Random graph clustering for collaborative filtering

Gabriel Frisch

▶ **To cite this version:**

Gabriel Frisch. Random graph clustering for collaborative filtering. Other [cs.OH]. Université de Technologie de Compiègne, 2021. English. NNT : 2021COMP2635 . tel-03581533

HAL Id: tel-03581533

https://tel.archives-ouvertes.fr/tel-03581533

Submitted on 19 Feb 2022

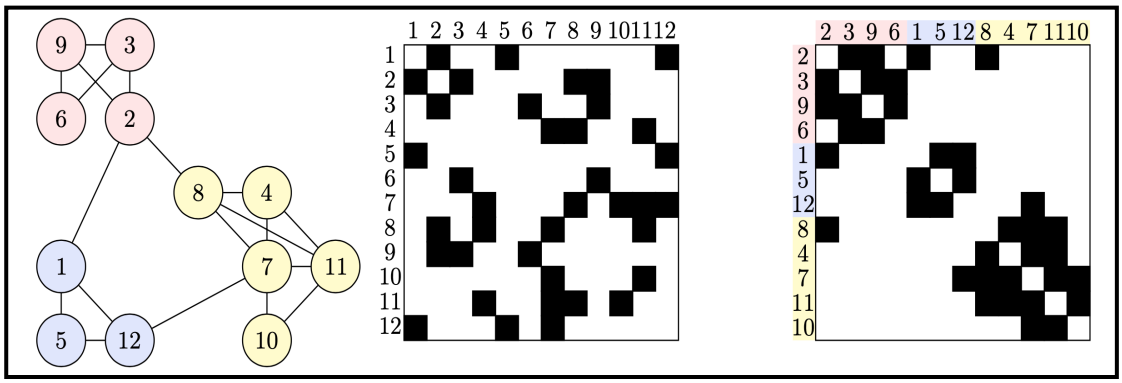
HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

Par **Gabriel FRISCH**

Random graph clustering for collaborative filtering

Thèse présentée
pour l'obtention du grade
de Docteur de l'UTC



Soutenue le 18 octobre 2021
Spécialité : Sciences et Technologies de l'Information et des Systèmes : Unité de recherche Heudyasic (UMR-7253)
D2635

Université de Technologie de Compiègne

Thèse de doctorat

Random graph clustering for collaborative filtering

Spécialité : Sciences et Technologies de l'Information et des Systèmes

Auteur:
Gabriel Frisch

Directeurs:
Jean-Benoist Leger
Yves Grandvalet

Jury:

Julien Chiquet	Université Paris-Saclay AgroParisTech, INRAE	*Examinateur*
Cédric Févotte	Institut de Recherche en Informatique de Toulouse, CNRS	*Rapporteur*
Gérard Govaert	Université de Technologie de Compiègne	*Examinateur*
Julie Josse	Institut Desbrest d'Épidémiologie et de Santé Publique, INRIA Montpellier	*Examinatrice*
Christine Keribin	Université Paris Saclay Laboratoire de Mathématiques d'Orsay	*Rapportrice*

Manuscrit présenté pour obtenir le titre de docteur de l'Université de Technologie de Compiègne

UMR Heudiasyc 7253
Équipe CID

18 octobre 2021

Remerciements

L'aboutissement de ma thèse n'aurait pas été possible sans l'aide et le soutien d'un grand nombre de personnes à qui je voudrais témoigner toute ma gratitude.

J'adresse tous mes remerciements à Jean-Benoist Leger et Yves Grandvalet pour leurs conseils et leur encadrement tout au long de ces trois dernières années. Vous avez toujours été à l'écoute, disponible et réactif face à mes nombreuses questions, qui je le conçois n'étaient pas toujours très pertinentes. Avec votre bonne humeur, votre patience, et votre persévérance, vous avez su m'encourager et me guider jusqu'à l'achèvement de ma thèse. C'est avec certitude, que je peux affirmer que vous avez été meilleur que le meilleur encadrant qu'un doctorant puisse espérer avoir.

Je tiens aussi à remercier les rapporteurs de cette thèse, madame Christine Keribin et monsieur Cédric Févotte pour leur relecture attentive et leurs remarques si judicieuses qui indéniablement abouti a une amélioration de mes travaux scientifiques. Je remercie également madame Julie Josse, monsieur Julien Chiquet et monsieur Gérard Govaert de l'honneur qu'ils m'ont fait en acceptant de faire parti du jury de cette thèse.

Je tiens aussi à remercier Benjamin Quost et Philippe Xu pour leur accompagnement lors de mes études et pour m'avoir donné l'envie de me lancer dans le monde de la recherche.

Je suis reconnaissant envers tous les autres membres du Laboratoire Heudiasyc avec lesquels j'ai pu partager de bons moments et de longues discussions scientifiques si intéressantes et enrichissantes: Yonatan, Yousseff, Joelle, Federico (motivation !), Anthony, Soundouss, Antoine, Corenthin et bien d'autres encore. Merci à Ayyoub, Freddy et Anthony qui ont eu la brillante idée (et un peu folle) de m'embarquer dans une grande aventure en vélo. Vélo qui par la suite, m'aura permis de m'échapper de bien nombreuses fois aux préoccupations de mon travail. Mens sana in corpore sano.

Enfin, je tiens à remercier ma famille et Aziliz qui m'ont beaucoup soutenu dans les moments difficiles de ce projet.

Résumé

Le filtrage collaboratif est une méthode qui vise à construire automatiquement des filtres personnalisés en utilisant les avis d'utilisateurs dans le but de leur proposer une liste restreinte d'objets qu'ils pourraient le plus apprécier. Les avis recueillis auprès de ces utilisateurs fournissent la base collaborative pour réaliser des prédictions sur les avis manquants.

La proportion de données manquantes, c'est à dire d'avis non fournis par les utilisateurs, est généralement importante dans les systèmes de recommandation. Dans certains cas, ces données manquantes peuvent être informatives et l'ignorer peut mener à des mauvaises conclusions. La plupart des méthodes de filtrage collaboratif n'utilisent pas ces données manquantes pour en extraire de l'information. Dans cette thèse, nous voulons exploiter ces données manquantes en proposant des modèles qui supposent que les avis des utilisateurs ne manquent pas au hasard (MNAR). L'approche se base sur le modèle à blocs latents, un modèle de co-clustering génératif dont les régularités permettent de prédire les avis manquants.

L'équité de la recommandation est un des autres problèmes majeurs du filtrage collaboratif. L'équité est souvent vaguement définie comme la qualité de traiter les gens avec justesse et impartialité. Bien qu'imprécise, cette définition stipule que l'équité du traitement fait référence à certains attributs protégés partagés par des groupes de personnes, tels que le sexe, l'âge, l'origine ethnique, le groupe socio-économique, etc. Ces dernières années, les travaux de recherche ont mis en évidence un manque d'équité dans les décisions prises par les algorithmes de filtrage collaboratif. Le modèle statistique que nous présentons dans cette thèse génère un co-clustering des utilisateurs et des objets du système tout en essayant de respecter une parité statistique des utilisateurs vis-à-vis de leurs attributs protégés. Nous donnons des garanties théoriques assurant une recommandation équitable quand la parité statistique des utilisateurs est bien respectée.

Abstract

Collaborative filtering is a method that aims at building automatically personalized filters by using feedback of users for the particular purpose of providing a shortlist of items that they might like the most. The ratings collected from these users provide the collaborative basis for making predictions about unseen items.

The datasets extracted from recommender systems have usually a high proportion of missing feedback, that is, feedback that were not provided by the users. In some situations, missing data can be informative and ignoring this information can lead to misleading conclusions. Most collaborative filtering methods do not have a principled method for extracting information from this missing data. In this thesis we take advantage of missing data by positing models of missingness that assume that the feedback are missing not at random (MNAR). The approach relies on the latent block model, a generative co-clustering model whose regularities allow to predict the missing ratings.

Another major issue in collaborative filtering is the fairness of recommendations. Fairness is often loosely defined as the quality of treating people with rightfulness. Although imprecise, this definition stipulates that fairness of treatment refers to certain sensitive attributes shared by groups of people, such as gender, age, ethnicity, socio-economic group, etc. In recent years, research has highlighted the lack of fairness in decisions made by recommendation system algorithms. The model we present in this thesis produces a co-clustering of users and items that respects a statistical parity of users with respect to the sensitive attributes. We give theoretical guarantees ensuring fair recommendations when the statistical parity of users is respected.

Contents

Notations

n_1 : number of vertices in a graph;
number of vertices of the first type in a bipartite graph;
number of users in a recommender system

n_2 : number of vertices of the second type in a bipartite graph;
number of items in a recommender system

k_1 : number of classes of vertices in a graph;
number of classes of vertices of the first type in a bipartite graph;
number of classes of users in a recommender system

k_2 : number of classes of vertices of the second type in a bipartite graph;
number of classes of items in a recommender system

i : index for vertices (of the first type) $i \in \{1, ..., n_1\}$

j : index for vertices of the second type $j \in \{1, ..., n_2\}$

q : index for classes of vertices (of the first type) $i \in \{1, ..., k_1\}$

l : index for classes vertices of the second type $j \in \{1, ..., k_2\}$
Note that bounds in summations or products will most often be implicit, for example $\sum_i$ will be a shortcut for $\sum_{i=1}^{n_1}$ and $\sum_{ijql}$ for $\sum_{i=1}^{n_1} \sum_{j=1}^{n_2} \sum_{q=1}^{k_1} \sum_{l=1}^{k_2}$.

$\boldsymbol{X}$: $(n_1 \times n_2)$ data matrix gathering the feedbacks of users on the items. Depending on the situation, X_{ij} can be binary or continuous. $\boldsymbol{X}_i$ is the vector of size n_2 gathering the feedbacks given by user i, and $\boldsymbol{X}_j$ is the vector of size n_1 gathering the feedbacks that item j received.

$\boldsymbol{X}^{(\mathrm{o})}$: $(n_1 \times k_1)$ "partially observed" data matrix, with missing entries. Missing entries are denoted $X_{ij} = \mathrm{NA}$.

$\boldsymbol{M}$: $(n_1 \times n_2)$ binary mask matrix that indicates the non-missing entries of $\boldsymbol{X}^{(\mathrm{o})}$: if $M_{ij} = 0$, then $X_{ij}^{(\mathrm{o})} = \mathrm{NA}$.

$\boldsymbol{Y}$: $(n_1 \times k_1)$ matrix of user embeddings; in clustering models, $\boldsymbol{Y}$ is the matrix of membership of users to groups: $Y_{iq} = 1$ if user i belongs to cluster q.

$\boldsymbol{Z}$: $(n_2 \times k_2)$ matrix of items embeddings; in clustering models, $\boldsymbol{Z}$ is the matrix of membership of items to groups: $Z_{jl} = 1$ if item j belongs to cluster l.

$\mathcal{B}$: Bernoulli distribution

$\mathcal{M}$: multinomial distribution

$\mathcal{N}$: normal distribution

$\mathbf{S}_{n-1}$: standard simplex of $\mathbb{R}^n$: $\mathbf{S}_{n-1} = \{x \in \mathbb{R}_+^n \,|\, \|x\|_1 = 1\}$

SBM : stochastic block model

LBM : latent block model

MCAR: missing completely at random

MAR : missing at random

MNAR: missing not at random

The hat over a symbol denotes an estimator or an estimated value, for example $\hat{\boldsymbol{X}}$ is the matrix of predicted feedbacks (estimator or estimate depending on the context).

Chapter 1

Introduction

Contents

1.1 Collaborative filtering

When searching for a product or a service on a platform, users can quickly be flooded by a very large choice of items. One way to handle a large volume of items is to provide one or several *filters* that reduce the list to some potentially more relevant items. Another solution is the *collaborative filtering*. Collaborative filtering is a method that aims at building automatically personalized filters by using feedback of people for the particular purpose of providing a shortlist of items that they might most enjoy. The reactions collected from people provide the *collaborative* basis for making predictions on other unseen items.

The feedback given by users can be implicit (browsing history, clicks, scrolling, etc.) or explicit. In the case of explicit evaluation data, users most often express their interest in items using a discrete rating scale that supposes an order between levels, for example from 1 to 5 expressing the worst opinion to the best one. As illustrated by Figure 1.1, the data extracted from recommender systems can be aggregated in a matrix where rows are users, columns are items and entries the feedback. These matrices are usually extremely sparse, with a high proportion of missing ratings, that is, ratings that were not provided by the users.

3	5	2	?	5	?	4	?	5	3	?	5	3
3	?	4	2	1	?	?	4	?	2	5	4	?
?	2	?	?	?	3	?	1	?	?	2	?	2
4	?	?	5	?	2	?	?	3	?	5	?	4
?	?	3	2	?	2	?	1	4	2	?	3	?
5	5	4	?	2	?	4	?	1	1	?	5	1
3	?	4	3	3	?	?	2	?	3	2	3	?
?	2	?	?	?	1	?	5	?	?	4	?	2
4	?	?	2	?	5	?	?	2	?	3	?	2
?	?	2	4	?	4	?	4	4	5	?	3	?
5	2	1	?	5	?	4	?	4	1	?	4	1
2	?	5	5	3	?	?	3	?	1	3	2	?
?	2	?	?	?	1	?	2	?	?	3	?	3
5	?	?	3	?	2	?	?	4	?	5	5	?
?	?	2	2	?	1	?	3	2	5	?	2	?

Figure 1.1: Illustration of the data extracted from a recommender system and aggregated in a matrix, denoted by $\boldsymbol{X}$ where rows are for users and columns for items. A collaborative filtering system aims at filling the missing values (?).

To predict the missing entries of the matrix, the numerous methods that have been proposed can be classified in either the category of neighborhood-based methods, or the category of model-based methods. This classification is obviously a bit reductive as some solutions known as *hybrids* are a mixture of both methods. The first type of approach attempts to predict the users opinions using a similarity based criteria while the second type, model-based, constructs a model of the observed feedback used to predict the missing values.

Both approaches have their pros and cons. Small recommender systems used in industry often stick with neighborhood methods as they are easy to understand and to implement. They provide as well *explainable* results and they help solving the *cold start* problem of giving satisfactory recommendations for a new entrant in the system, whether it is a user or an item. However, neighborhood methods fell into disrepute because they often provide less accurate results than the model-based methods and more importantly because they hardly scale up to very large systems. Model-based methods offer the possibility to describe various aspects of the data, enabling to target other issues in collaborative filtering; these include the *diversity* of recommendations, which aims to avoid user boredom, the *long-tail* problem, which aims to highlight less popular items, the *fairness* to ensure equal treatment, or the modeling of the missingness process to extract information from it.

The following sections provide a brief review of neighborhood methods and

a more specific review of the main model-based methods, namely clustering, matrix factorization and neural networks.

1.1.1 Neighborhood methods

This type of methods, which is also known in the literature as memory-based or similarity-based methods, computes the relationships between items or between users to predict the missing ratings. The simplest algorithms are directly derived from a basic nearest-neighbors approach [Sarwar et al., 2001, Breese et al., 2013, Bell and Koren, 2007] using a similarity measure between pairs of users or items that could typically be a cosine-based similarity or a difference-based similarity (mean-squared difference, mean-absolute difference) or a (Pearson) correlation-based coefficient, etc. When relationships are computed between users, the ratings are estimated as:

$$\hat{X}_{ij} = \frac{\sum_{i' \in N_j^k(i)} \text{similarity}(i, i') \cdot X_{i'j}}{\sum_{i' \in N_j^k(i)} \text{similarity}(i, i')}$$

with $N_j^k(i)$ being the set of the k-nearest neighbors of user i when considering item j. The Pearson correlation-based coefficient (PCC) is an example of similarity measure computed as:

$$\text{similarity}_{\text{PCC}}(i, i') = \frac{\sum_{j \in I_i \cap I_{i'}} \left(X_{ij} - \overline{X}_i\right)\left(X_{i'j} - \overline{X}_{i'}\right)}{\sqrt{\sum_{j \in I_i \cap I_{i'}} \left(X_{ij} - \overline{X}_i\right)^2} \sqrt{\sum_{j \in I_i \cap I_{i'}} \left(X_{i'j} - \overline{X}_{i'}\right)^2}}$$

with $\overline{X}_i$ the average rating of user i and $I_i \cap I_{i'}$ the set of co-rated items of two users i and i'. When relationships are computed between items, the ratings are estimated similarly.

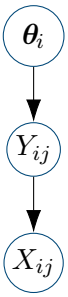
These methods are popular as an understandable explanation of the recommended items can be given to users which could in turn encourage them to leave more evaluations. Additionally, these methods easily deal with new items or users as they can provide recommendations without needing to re-estimate the whole set of parameters. However their poor performances compared to the latent factor models is their major drawback.

1.1.2 Clustering methods

Methods based on clustering give a natural way to identify groups of users with similar preferences. Clusters can then be used as the basis for rating prediction methods. A focus on some methods developed specifically for recommendation is given is this section.

User Rating Profile model: Marlin [2003] combines a latent Dirichlet allocation model [Blei et al., 2003] with a multinomial mixture model in a

model called the User Rating Profile model (URP). The URP model posits that each user i is characterized by its latent "attitude" that can be interpreted as its preferences profile. For each user the attitude is decomposed, into a set of typical preference patterns, by a mixture of user "attitudes", $\boldsymbol{Y}_i$, with Dirichlet mixing proportions $\boldsymbol{\theta}_i$. The individual Dirichlet mixing proportions allow the users to have different prior distributions over user attitudes. The ratings are generated independently, conditionally to user attitudes. The model is formally defined as:



$$\begin{cases} \forall i,\ \boldsymbol{\theta}_i \overset{\text{iid}}{\sim} \text{Dirichlet}(\boldsymbol{\alpha}), & \boldsymbol{\alpha} \in \mathbb{R}_+^{n_2} \\ \forall i,j,\ \boldsymbol{Y}_{ij} \overset{\text{iid}}{\sim} \mathcal{M}(1;\boldsymbol{\theta}_i), & \boldsymbol{\theta}_i \in \mathbf{S}_{n_2-1} \\ \forall i,j,\ X_{ij}|\boldsymbol{Y}_{ij} = y \overset{\text{ind}}{\sim} \mathcal{M}\big(1;\boldsymbol{\beta}_{yj}\big), & \boldsymbol{\beta}_{yj} \in \mathbf{S}_{n_v-1} \end{cases}$$

where $\mathcal{M}$ is the multinomial distribution, $\boldsymbol{\theta}_i \in \mathbf{S}_{n_2-1} = \{\boldsymbol{\theta}_i \in \mathbb{R}_+^{n_2} | \sum_j \theta_{ij} = 1\}$ and $\boldsymbol{\beta}_{yj} \in \mathbf{S}_{n_v-1} = \{\boldsymbol{\beta}_{yj} \in \mathbb{R}_+^{n_v} | \sum_v \beta_{yjv} = 1\}$.

Weighted Bregman co-clustering: George and Merugu [2005] proposed an approach based on a weighted Bregman co-clustering algorithm. The model simultaneously assigns users and items to clusters C_i and C_j and to co-clusters denoted C_{ij}. The predictions are generated based on the average ratings of the co-clusters and by taking into account the biases of the individual users and items. The estimated rating is given by:

$$\hat{X}_{ij} = \overline{C}_{ij} + \big(\mu_i - \overline{C}_i\big) + \big(\mu_j - \overline{C}_j\big) \ ,$$

where μ_i, μ_j are the average ratings of user i and item j, and $\overline{C}_{ij}$, $\overline{C}_i$, $\overline{C}_j$, are the average ratings of the corresponding co-cluster, user-cluster and item-cluster respectively. One interesting point of this approach is that it can support the entry of new users or items in the system by using an incremental and batched training procedure. Based on our experience, the method is faster and also more scalable than matrix factorization-based methods but gives poorer performance (see Section 3.2.3).

1.1.3 Matrix factorization methods

Matrix factorization (MF) algorithms decompose a matrix $\boldsymbol{X}$ of size $n_1 \times n_2$ into the product of two lower dimensionality rectangular matrices denoted $\boldsymbol{Y}$ and $\boldsymbol{Z}$ of size respectively $k_1 \times n_1$ and $k_1 \times n_2$. In the case of collaborative filtering, the first matrix $\boldsymbol{Y}$ embeds users and the second one $\boldsymbol{Z}$ embeds items. Both users and items share the same latent factor space of dimensionality k_1. In its vanilla form, matrix factorization estimates the rating given by user i to

item j by the product of the two latent vectors:

$$\hat{X}_{ij} = \boldsymbol{Y}_i^T \boldsymbol{Z}_j \ .$$

To infer the latent factors, MF methods usually but not always minimize the quadratic error between the true ratings X_{ij} and the predicted ratings $\hat{X}_{ij}$:

$$\underset{\boldsymbol{Y},\boldsymbol{Z}}{\arg\min} \sum_{i,j\in\kappa} \left(X_{ij} - \hat{X}_{ij}\right)^2 \tag{1.1}$$

where κ is the set of the (i,j) pairs for which X_{ij} is known. A regularization term on $\boldsymbol{Y}$ and $\boldsymbol{Z}$ is often added to this objective function to avoid over-fitting.

To minimize the criterion of Equation (1.1), stochastic gradient descent and alternating least squares are the two major approaches. Stochastic gradient descent updates the latent factors negatively proportional to the gradient of the objective function computed using one sampled rating at a time. This results in a very simple and rather fast running time algorithm. Alternating least squares (ALS) solves the problem optimally by alternatively fixing one of the two latent factors and minimizing the criterion with respect to the other one. The resulting algorithm consists in solving alternatively two quadratic objectives until convergence of the objective function. ALS is particularly well suited for handling very large matrices as training can be massively parallelized.

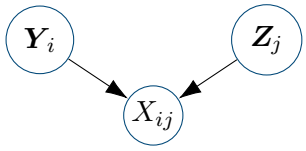
Matrix factorization is with no doubt the most commonly-used method to perform collaborative filtering since the Netflix prize challenge in 2009 that rewarded a 10% performance improvement in Netflix algorithms. One strength of matrix factorization is that it can easily be extended to improve predictions or to help solving well-known issues in collaborative filtering such as cold start [Zhou et al., 2011], fairness [Kamishima et al., 2018] or MNAR data [Hernández-Lobato et al., 2014]. MF models ordinal data by artificially creating a distance between the different rating levels, but as most collaborative filtering methods, it assumes the same distance between the different levels.

Biased MF: One extension to improve performances is the biased matrix factorization [Koren et al., 2009] where additional terms modeling the effects associated to users or items are added when estimating the ratings. The ratings estimation is broken down into four components: a global average μ, the user bias a_i, the item bias b_j and finally the dot product $\boldsymbol{Y}_i^T \boldsymbol{Z}_j$ that should capture the interaction between user i and item j only:

$$\hat{X}_{ij} = \boldsymbol{Y}_i^T \boldsymbol{Z}_j + a_i + b_j + \mu \ ,$$

with $a_i \in \mathbb{R}$, $b_j \in \mathbb{R}$, $\mu \in \mathbb{R}$. These biases are introduced to reflect the fact that each user has his/her own perception of the rating scale and that some items are valued more than others.

Probabilistic MF: Mnih and Salakhutdinov [2007] proposed a probabilistic view of the matrix factorization with the Probabilistic Matrix Factorization (PMF) model. PMF introduces a probabilistic linear model with Gaussian observation noise on the conditional distribution over the observed ratings and zero-mean spherical Gaussian priors on the two latent low-rank matrices:

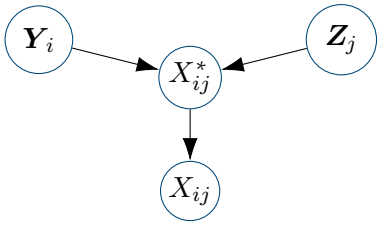


$$\begin{cases} \forall i,\ \boldsymbol{Y}_i \overset{\text{iid}}{\sim} \mathcal{N}\big(\boldsymbol{0}, \sigma_Y^2 \boldsymbol{I}\big) \\ \forall j,\ \boldsymbol{Z}_j \overset{\text{iid}}{\sim} \mathcal{N}\big(\boldsymbol{0}, \sigma_Z^2 \boldsymbol{I}\big) \\ \forall i,j,\ X_{ij} | \boldsymbol{Y}_i, \boldsymbol{Z}_j \overset{\text{ind}}{\sim} \mathcal{N}\big(\boldsymbol{Y}_i^T \boldsymbol{Z}_j, \sigma^2\big) \end{cases}$$

with $\mathcal{N}($,$the normal distribution, and \sigma_Y^2 \in \mathbb{R}_+,\ \sigma_Z^2 \in \mathbb{R}_+,\ \sigma^2 \in \mathbb{R}_+$. Standard matrix factorization models have trouble making accurate predictions for users who have very few ratings. For such users, the statistical distributions on latent factors in PMF tend to give a better generalization and regularization.

The PMF model is also proposed in a Bayesian framework by the same authors [Salakhutdinov and Mnih, 2008]. Some Gaussian-Wishart priors are introduced on the parameters of the distributions of $\boldsymbol{Y}$ and $\boldsymbol{Z}$ and the log-posterior is approximately maximized using MCMC methods [Neal, 1993]. The authors claim that the Bayesian version significantly increases the model's predictive accuracy, especially for the infrequent users, compared to the standard PMF model.

Ordinal NMF (OrdNMF): Gouvert et al. [2020] introduce a probabilistic non-negative matrix factorization (NMF) method to process ordinal data, called OrdNMF. In case of explicit feedbacks, users most often express their interest in items using a discrete ordered rating scale. Ordinal models take advantage of this ordinal data. Their model assumes that the data X_{ij} results from the quantization of a continuous latent variable X_{ij}^* with respect to an increasing sequence of thresholds. The latent variable X_{ij}^* links the factorization term $\boldsymbol{Y}_i^T \boldsymbol{Z}_j$ and the ordinal data X_{ij}. They jointly infer the latent variables as well as the sequence of thresholds to learn the distances between the different rating levels.



LLORMA: MF methods have the major drawback that they do not search for associations among a set of closely related user or items. Lee et al. [2016] relaxed the low-rank assumption of MF methods and proposed a Local Low-

Rank Matrix Approximation (LLORMA) assuming that the rating matrix behaves as a low-rank matrix in the vicinity of certain user-item combinations. They use several low-rank approximations instead of a single one and they estimate the matrix with a convex combination of low-rank matrices, each of which approximates the original matrix over a particular region centered on an anchor point:

$$\hat{X}_{ij} = \sum_q \frac{w_{ij}^q}{\sum_l w_{ij}^l} \left(\left(\boldsymbol{Y}_i^q \right)^T \boldsymbol{Z}_j^q \right)$$

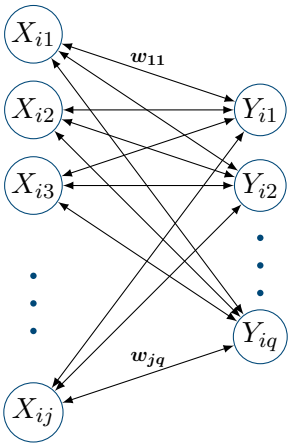
where w_{ij}^q is a weight based on the distance between the point (i, j) and the anchor point of the matrix q.

GLOMA: The previous model LLORMA is prone to overfitting when the region approximated by the local low-rank matrix is too sparse. To solve this problem, Chen et al. [2017] proposed to use Global information in the LOcal Matrix Approximations (GLOMA). A previously trained standard matrix factorization is used to help to train the local low-rank matrices such that each local matrix combines global and local latent factors of users and items.

1.1.4 Neural network based methods

Neural networks bring more flexibility to the models approximating any continuous function by composing nonlinear transformations. They are successfully used to introduce complex dependencies between users or between items. The methods presented below are examples of the most popular models taking advantage of neural networks.

Restricted Boltzmann machines: The Restricted Boltzmann Machine (RBM) is a two-layer undirected generative model proposed by Smolensky [1986]. The first layer of the model consists of "visible" units modelling the ratings and is connected to the second layer made up of latent or "hidden" units. The visible units $(X_{ij})_{ij}$ are assumed to be independent conditionally to the hidden units and follow a multinomial distribution to model discrete ratings. The hidden units $(Y_{iq})_{iq}$ are independent conditionally to the visible units and are Bernoulli distributed. The model is formally defined as follows with v being the value of the rating and q being the index of the hidden unit:



$$\begin{cases} p(X_{ij} = v|\boldsymbol{Y}_i) = \frac{\exp\left(b_j^v + \sum_q Y_{iq} w_{jq}^v\right)}{\sum_v \exp\left(b_j^v + \sum_q Y_{iq} w_{jq}^v\right)} \\ p(Y_{iq} = 1|\boldsymbol{X}_j) = \text{expit}\left(b_q + \sum_{jv} \mathbb{1}_{(X_{ij}=v)} w_{jq}^v\right) \end{cases}$$

with $b_q \in \mathbb{R}, b_j^v \in \mathbb{R}$, the unit weights $w_{jq}^v \in \mathbb{R}$ and $\text{expit}(x) = 1/(1 + \exp(-x))$

One disadvantage of the RBM model is its high number of unit weights $\boldsymbol{w}$ which is of the order of the number of items times the number of hidden units times the number of values of the rating scale. Applied to a standard collaborative filtering dataset, the model can easily reach one million parameters. To reduce model complexity, Salakhutdinov et al. [2007] propose to decompose the weight matrix $\boldsymbol{w}^v$ into a product of two lower-rank matrices.

Georgiev and Nakov [2013] extend the standard RBM using two independent hidden layers: one modeling correlations between items and another one modeling correlations between users. They also use real values in the visible layer as opposed to multinomial variables, thus taking advantage of the natural order between ratings.

Auto-encoder and variational auto-encoder: Auto-encoders for collaborative filtering are neural networks which take as input the vector $\boldsymbol{X}_i$ of length n_2 containing the partially observed binarized ratings of user i, project it into a low-dimensional $\mathbb{R}^d$ latent space, and then reconstruct $\boldsymbol{X}_i$ in the original space, thereby enabling the prediction of missing ratings. The model optimizes the following reconstruction error on observed ratings only, using a stochastic gradient descent algorithm:

$$\underset{\theta_{\text{enc}}, \theta_{\text{dec}}}{\arg\min} \sum_i \|\boldsymbol{X}_i - f_{\text{dec}}(f_{\text{enc}}(\boldsymbol{X}_i; \theta_{\text{enc}}); \theta_{\text{dec}})\|_{\mathcal{O}}^2$$

where $\|\cdot\|_{\mathcal{O}}^2$ only considers the contribution of observed ratings. f_{enc} is the encoding function mapping the original ratings to the low-dimensional latent space and is constituted of several deterministic nonlinear transformations. Likewise f_{dec} is a neural network but maps the low-dimensional latent vector to the original space. Sedhain et al. [2015] proposed an auto-encoder for collaborative filtering whose performance was competitive with Restricted Boltzmann Machines. The hyper-parameters of auto-encoders have to be carefully tuned and the parameters regularized to avoid over-fitting.

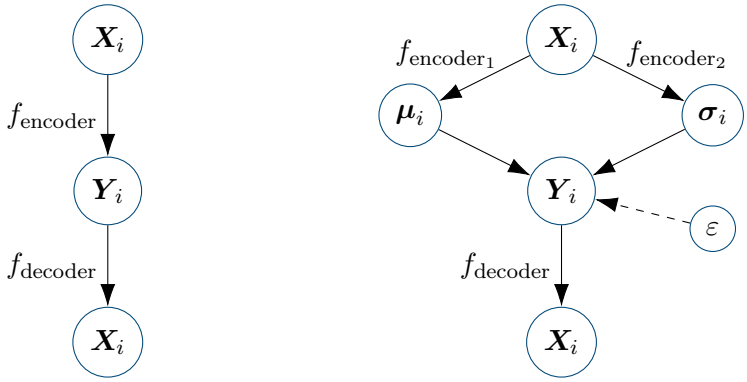


Figure 1.2: A taxonomy of auto-encoders. (a): the standard auto-encoder maps the original vector $\boldsymbol{X}_i$ to a low-dimensional latent vector $\boldsymbol{Y}_i$ and attempts to reconstruct it. (b): the variational auto-encoder maps the original vector to a low-dimensional multivariate Gaussian latent vector. The dotted arrow indicates a sampling operation performed during inference.

The variational auto-encoder (VAE) adds probabilistic distributions on the input vector $\boldsymbol{X}_i$ and on the latent vector $\boldsymbol{Y}_i$:

$$\begin{cases} \forall i,\ \boldsymbol{Y}_i \overset{\text{iid}}{\sim} \mathcal{N}(\boldsymbol{0}, \boldsymbol{I}_d) \\ \forall i\ \ \boldsymbol{X}_i | \boldsymbol{Y}_i \overset{\text{ind}}{\sim} \mathcal{B}(f_{\text{dec}}(\boldsymbol{Y}_i)) \end{cases}$$

with $\mathcal{B}$ standing for the Bernoulli distribution. The intractable posterior distribution is approximated using variational inference. The procedure introduces q_γ, a restricted parametric inference distribution defined on the latent variables of the model, and optimizes a lower bound on the log-likelihood. For an efficient stochastic gradient optimization of this lower bound, Kingma and Welling [2014] propose computational tricks based on the sampling of continuous latent variables. In the VAE model, the restricted variational distribution is set to be a fully factorized Gaussian distribution:

$$q_\gamma = \prod_i \mathcal{N}\big(\boldsymbol{\mu}_i, \boldsymbol{\sigma}_i^2 \boldsymbol{I}_d\big) \tag{1.2}$$

The neural network is defined so that the encoder outputs the variational parameters $(\boldsymbol{\mu}_i, \boldsymbol{\sigma}_i^2)_i$ of the latent Gaussian vector. Contrary to auto-encoders, VAE have the ability to capture user variances in the latent state $\boldsymbol{Y}_i$ and they are easier to train as the probabilistic distributions act as a regularization.

In their model, Liang et al. [2018] replace the Bernoulli conditional distribution of $\boldsymbol{X}_i$ with a multinomial arguing that it is better suited to modeling click data. Li and She [2017] make use of external data on items and they add a second latent vector to encode them.

Neural collaborative filtering (NCF): He et al. [2017] proposed a neural network architecture to model latent features of users and items and replace the inner product of matrix factorization with a neural architecture to generate ratings. As described by Figure 1.3, NCF takes as input the sparse representations of a user i and an item j which consists of the binarized vectors $\boldsymbol{X}_i$ and $\boldsymbol{X}_j$. The sparse vectors are fed to an embedding layer that outputs dense representations which in turn serves as the input of the multi-layer perceptron whose final output layer is the predicted rating $\hat{X}_{ij}$. The model is trained by minimizing the pointwise loss between the observed ratings X_{ij} and its estimated value $\hat{X}_{ij}$.

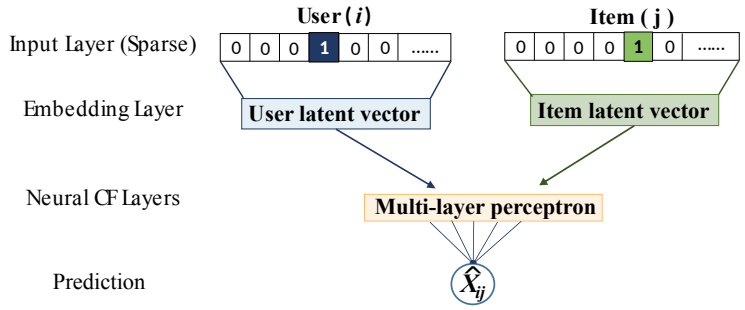


Figure 1.3: Main components of the Neural Collaborative filtering model. Using the sparse representation of user i and item j, the network outputs the ratings $\hat{X}_{ij}$. Redesigned illustration from He et al. [2017]

1.2 Graph clustering

As illustrated by Figure 1.4, a bipartite graph represents the data extracted from a recommender system in an equivalent manner as the matrix from Figure 1.1 does. The two sets of vertices are modelling users and items while an edge, possibly valued, is present if a feedback has been collected.

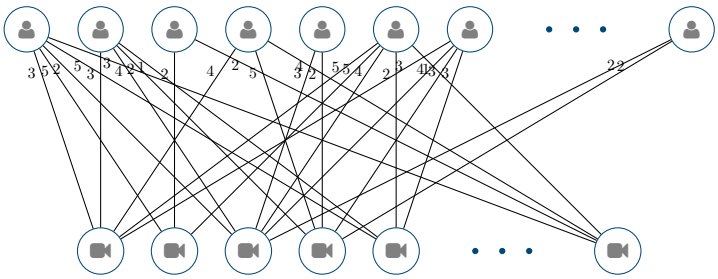


Figure 1.4: Illustration of the data extracted from a recommender system and aggregated in a bipartite graph. The first set of vertices is for users and the second one for items. The feedbacks are represented by edges.

Graph clustering groups the vertices of a graph into clusters taking into consideration the edge structure of the graph. Identifying groups of users having similar preferences for similar items is an important phase of collaborative filtering. Thus, existing methods for clustering vertices of a graph can provide such possibilities. Graph clustering methods can either be purely algorithmic using the graph properties or statistically based assuming that the graph is an observation of a generative random process.

The following sections are not intended to give an exhaustive review on graph clustering but only to give a few insights into some widely used methods or into some innovative methods that extend graph clustering.

1.2.1 Algorithmic graph clustering

Graphs come with various properties and some of which can be used to discriminate vertices such as degree or betweenness centrality of vertices, modularity or sub-graphs, strong connected components, etc.

Some methods use these properties to cluster vertices. For a careful and extensive analysis on graph clustering based on graph properties the reader is referred to Schaeffer [2007].

k-Spanning Tree A minimum spanning tree is a subset of edges of a weighted undirected graph that connects all vertices with no cycles and at a minimum cost.

To find a minimum spanning tree Kruskal's algorithm [Kruskal, 1956] starts by removing all edges from a graph and repeatedly add back the cheapest edge that does not create a cycle. As illustrated by Figure 1.5, Kruskal's algorithm

can be applied for graph clustering by running the algorithm until only k clusters remains. For a collaborative filtering purpose, the algorithm can be adapted by searching the maximum spanning tree in bipartite graphs.

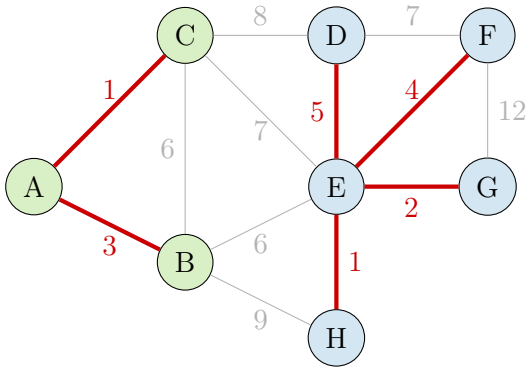


Figure 1.5: Minimum k-spanning tree clustering ($k = 2$) found with Kruskal's algorithm.

Hierarchical agglomerative clustering algorithm From a similarity criterion computed between vertices, an agglomerative algorithm groups iteratively the vertices into communities. At each step the number of classes decreases (if agglomerative; increases otherwise) which gives a hierarchical structure of communities that can be illustrated by a dendrogram (see Figure 1.6). The main advantage of these algorithms is that the number of classes is not predefined and could be chosen a posteriori. Their high computational complexity is usually their main flaw. Some attempts have been proposed to reduce this complexity.

Newman [2004] proposed a greedy algorithm starting with a state in which each vertex belongs to one of n classes, iteratively merges classes in pairs choosing at each step the merge that results in the greatest increase of a modularity based criterion.

Donetti and Muñoz [2004] proposed a method that exploits the spectral properties of the graph Laplacian-matrix allowing them to a project the network-nodes into an eigenvector-space (as in spectral clustering, see below) where it is possible to measure similarities between vertices.

Pons and Latapy [2005] used random walks on a graph with the intuition that walks tend to get trapped into densely connected parts of the graph. Using them, they define a measurement of the structural similarity between vertices.

Spectral Clustering: originally proposed by Donath and Hoffman [2003] clusters vertices of a graph using a similarity graph. Such a graph contains relationships between two vertices i and j if the distance between i and j is positive or larger than a certain threshold. This similarity graph is then used

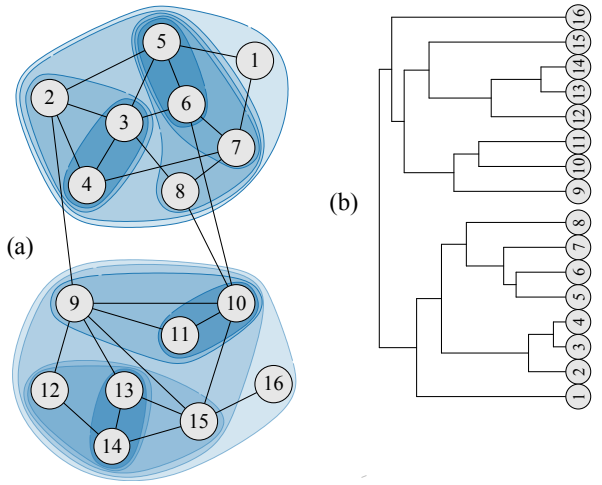


Figure 1.6: Illustration from Pons and Latapy [2005] (a): the community structure found by the random walk algorithm. (b): the dendrogram encoding the similarity between vertices shows if a different number of clusters may be relevant.

to compute the graph Laplacian matrix on which a dimensionality reduction is performed using its spectrum. Any standard clustering method such as k-means or hierarchical clustering can then be used on the reduced-dimensional matrix.

The spectral clustering method given in Algorithm 1 is one simple method based on the unnormalized graph Laplacian matrix. A detailed review on spectral clustering can be found in Von Luxburg [2007]. An interesting property used later in this thesis is the consistent identifiability proved by Rohe et al. [2011] of the Stochastic Block Model (see following section) by spectral clustering.

1.2.2 Random graph clustering

A random graph refers to a graph in which edges are generated according to a probability distribution. Random graphs may be described by probability distributions as in the Erdős-Rényi random graph Erdős and Rényi [1960] where each edge is included in the graph with probability p independently from every other edge, but also by more complex generative processes that may be partially observed. In particular, there are many latent space models assuming the existence of one or several latent random variables affecting the emission law of edges. These latent variables usually characterize the vertices sharing common properties by gathering them in classes or by bringing them together in the latent space. In this case, the generative model is often designed for clustering purposes. A few random graph models related to vertex clustering are presented below. For an extensive review on random graph models, the

Algorithm 1: Unormalized spectral clustering.

Input: Similarity matrix $\boldsymbol{S} \in \mathbb{R}^{n \times n}$, number of clusters k_1

begin

- Define a similarity graph using the similarity matrix. Let $\boldsymbol{W}$ be its weighted adjacency matrix.
- Define $\boldsymbol{D}$ the diagonal matrix of degrees, element of $\mathbb{R}^{n \times n}$: $D_{ii} = \sum_j W_{ij}$
- Define the unnormalized Laplacian as $\boldsymbol{L} = \boldsymbol{D} - \boldsymbol{W}$
- Find the k_1 eigenvectors corresponding to the k_1 smallest eigenvalues. Form the matrix $\boldsymbol{U} = [\boldsymbol{U_1}, ..., \boldsymbol{U_{k_1}}] \in \mathbb{R}^{n \times k_1}$ concatenating the eigenvectors into columns.

end

Result: clusters given by k-means with k_1 clusters on $\boldsymbol{U}$

reader is referred to Matias and Robin [2014] or Goldenberg et al. [2009].

Stochastic block model (SBM): The binary stochastic block model (SBM) is a probabilistic model that classifies the vertices of a graph. It is typically used to model the relationships (represented by edges) between homogeneous objects (represented by vertices). For instance, a social network can be represented by a graph, possibly directed, in which the vertices are people and the edges are their interactions. Each edge from vertex i to vertex j is associated to a random variable X_{ij} that codes for its presence: $X_{ij} = 1$ if an edge is present and $X_{ij} = 0$ otherwise. The SBM assumes a partition of the vertices that corresponds to a strong structure of the $(n_1 \times n_1)$ adjacency matrix $\boldsymbol{X}$ in homogeneous blocks. As illustrated in Figure 1.8, this block structure is unveiled by reordering the rows and columns of $\boldsymbol{X}$ according to their class index; for k_1 classes, the reordering reveals $k_1 \times k_1$ homogeneous blocks in the adjacency matrix. The partition is governed by the latent variable $\boldsymbol{Y}$, the $n_1 \times k_1$ indicator matrix of classes ($Y_{iq} = 1$ if vertex i belongs to class q and $Y_{iq} = 0$ otherwise). The class indicator of vertex i will be denoted $\boldsymbol{Y}_i$. The SBM makes several assumptions on the dependencies:

- Vertex classes are independent and identically distributed. The latent variables $\boldsymbol{Y}_i$ are independent and follow a multinomial distribution $\mathcal{M}(1; \boldsymbol{\alpha})$, where $\boldsymbol{\alpha} = (\alpha_1, ..., \alpha_{k_1})$ are the mixing proportions of vertices:

$$p(\boldsymbol{Y}; \boldsymbol{\alpha}) = \prod_i p(\boldsymbol{Y}_i; \boldsymbol{\alpha})$$
$$p(Y_{iq} = 1; \boldsymbol{\alpha}) = \alpha_q \ ,$$

with $\boldsymbol{\alpha} \in S_{(k_1-1)} = \{\boldsymbol{\alpha} \in \mathbb{R}_+^{k_1} | \sum_q \alpha_q = 1\}$.

- Given the vertex classes, the edge presences are independent and identically distributed. Given the vertex classes $\boldsymbol{Y}$, the edge presences $\boldsymbol{X}$ are independent and follow a Bernoulli distribution of parameter $\boldsymbol{\pi} = (\pi_{ql}; q = 1, ..., k_1; l = 1, ..., k_1)$: the probability of the presence of edge X_{ij} depends only on the classes of the two vertices i and j.

$$p(\boldsymbol{X}|\boldsymbol{Y}; \boldsymbol{\pi}) = \prod_{ij} p(X_{ij}|\boldsymbol{Y}_i, \boldsymbol{Y}_j; \boldsymbol{\pi})$$
$$p\big(X_{ij} = 1 \big| Y_{iq}Y_{jl} = 1; \boldsymbol{\pi}\big) = \pi_{ql} \quad .$$

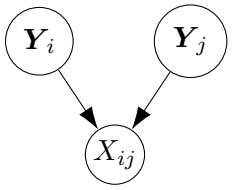


Figure 1.7: Random variable dependencies in the Stochastic Block Model

To summarize, the parameters of the SBM are $\theta = (\boldsymbol{\alpha}, \boldsymbol{\pi})$ and the probability mass function of $\boldsymbol{X}$ can be written as:

$$p(\boldsymbol{X}; \theta) = \sum_{\boldsymbol{Y} \in I} \left(\prod_{iq} {\alpha_q}^{Y_{iq}} \right) \left(\prod_{iqjl} \phi(X_{ij}; \pi_{ql})^{Y_{iq}Y_{jl}} \right) ,$$

where $\phi(X_{ij}; \pi_{ql}) = \pi_{ql}^{X_{ij}}(1 - \pi_{ql})^{1-X_{ij}}$ is the mass function of a Bernoulli variable, and where I denotes the set of all possible partitions of the n_1 vertices into k_1 groups.

The conditional distribution of X_{ij} can follow other various densities being possibly Gaussian or Poisson to deal with binary data, continuous or count data. The number of classes k_1 is supposed to be known otherwise statistical tools can be used for model selection (e.g. ICL from Biernacki et al. [2000], Daudin et al. [2008]).

Latent block model (LBM): The Latent Block Model (LBM) is a *co-clustering* model that classifies jointly the rows and the columns of a data matrix [Govaert and Nadif, 2008]. As illustrated in Figure 1.9, the LBM can be seen as an extended SBM that co-classifies the vertices of a bipartite graph.

This probabilistic generative model assumes a double partition on the rows and the columns of a $(n_1 \times n_2)$ data matrix $\boldsymbol{X}$ that corresponds to a strong structure of the matrix in homogeneous blocks. This structure is unveiled by reordering the rows and columns according to their respective cluster index; for k_1 row clusters and k_2 column clusters, the reordering reveals $k_1 \times k_2$

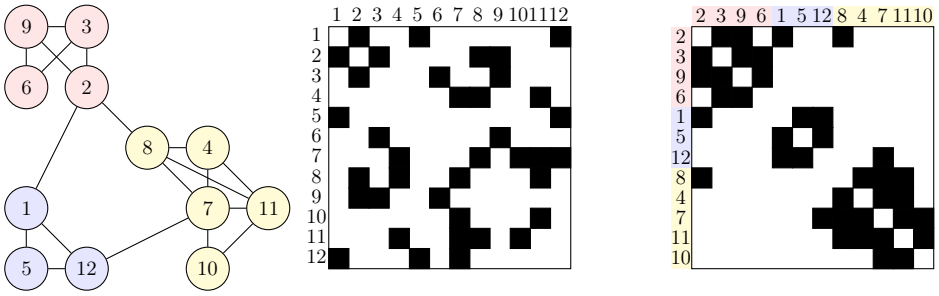


Figure 1.8: A binary graph on the left and its adjacency matrix on the center. The matrix on the right is the adjacency matrix reorganized according to the node clustering inferred by the stochastic block model.

homogeneous blocks in the data matrix. Note that we adopt here the original view where the data matrix is interpreted as a data table. If the matrix $\boldsymbol{X}$ is binary, it can also be interpreted as the biadjacency matrix of a bipartite graph, whose two sets of vertices correspond to the rows and columns of the data matrix. In this interpretation, $X_{ij} = 1$ if an edge is present between "row node" i and "column node" j, and $X_{ij} = 0$ otherwise.

For the $(n_1 \times n_2)$ data matrix $\boldsymbol{X}$, two partitions are defined by the latent variables $\boldsymbol{Y}$ and $\boldsymbol{Z}$, with $\boldsymbol{Y}$ being the $n_1 \times k_1$ indicator matrix of the latent row clusters ($Y_{iq} = 1$ if row i belongs to group q and $Y_{iq} = 0$ otherwise), and $\boldsymbol{Z}$ being the $n_2 \times k_2$ indicator matrix of the latent column cluster. The group indicator of row i will be denoted $\boldsymbol{Y}_i$, and similarly, the group indicator of column j will be denoted $\boldsymbol{Z}_j$. The LBM makes several assumptions on the dependencies:

- Independent rows and column clusters: the latent variables $\boldsymbol{Y}$ and $\boldsymbol{Z}$ are *a priori* independent.

$$p(\boldsymbol{Y}, \boldsymbol{Z}) = p(\boldsymbol{Y})p(\boldsymbol{Z}) \ .$$

  Note that *a priori* independence does not imply *a posteriori* independence: given the data matrix $\boldsymbol{X}$, the two partitions are (hopefully) not independent.

- Independent and identically distributed row clusters: the latent variables $\boldsymbol{Y}$ are independent and follow a multinomial distribution $\mathcal{M}(1; \boldsymbol{\alpha})$, where $\boldsymbol{\alpha} = (\alpha_1, ..., \alpha_{k_1})$ contains the mixing proportions of rows:

$$p(\boldsymbol{Y}; \boldsymbol{\alpha}) = \prod_i p(\boldsymbol{Y}_i; \boldsymbol{\alpha})$$
$$p(Y_{iq} = 1; \boldsymbol{\alpha}) = \alpha_q \ ,$$

  with $\boldsymbol{\alpha} \in S_{(k_1-1)} = \{\boldsymbol{\alpha} \in \mathbb{R}_+^{k_1} | \sum_q \alpha_q = 1\}$.

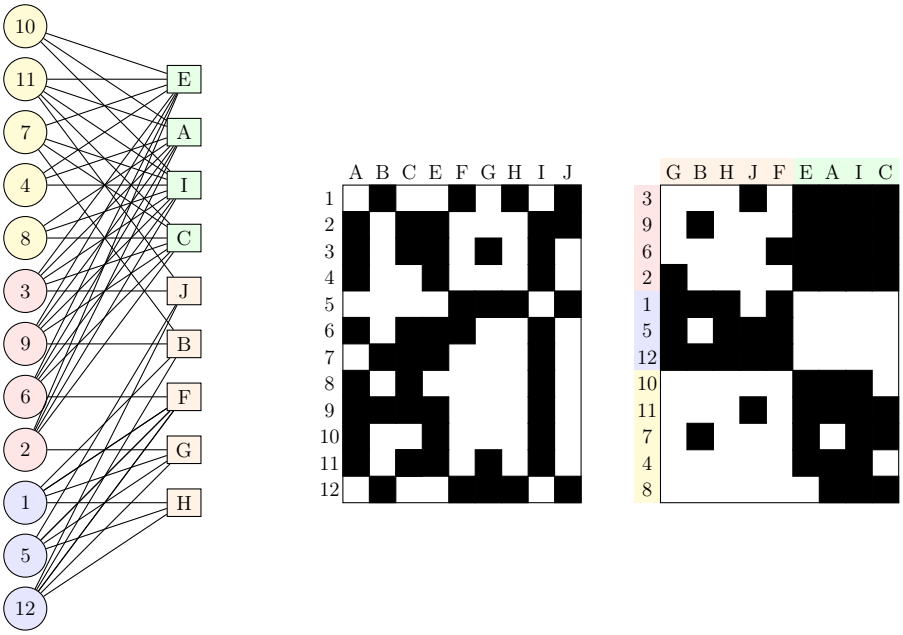


Figure 1.9: A binary bipartite graph on the left and its bi-adjacency matrix on the center. The matrix on the right is the bi-adjacency matrix reorganized according to the node clustering inferred by the latent block model.

- Independent and identically distributed column clusters: likewise, the latent variables $\boldsymbol{Z}$ are independent and follow a multinomial distribution $\mathcal{M}(1;\boldsymbol{\beta})$, where $\boldsymbol{\beta} = (\beta_1, ..., \beta_{k_2})$ contains the mixing proportions of columns:

$$p(\boldsymbol{Z};\boldsymbol{\beta}) = \prod_j p(\boldsymbol{Z}_j;\boldsymbol{\beta})$$
$$p(Z_{jl} = 1;\boldsymbol{\beta}) = \beta_l \ ,$$

  with $\boldsymbol{\beta} \in S_{(k_2-1)}$.

- Given row and column clusters, independent and identically distributed block entries: given the row and colum clusters $(\boldsymbol{Y}, \boldsymbol{Z})$, the entries X_{ij} are independent and follow a distribution: all elements of a block follow the same probability distribution.

$$p(\boldsymbol{X}|\boldsymbol{Y}, \boldsymbol{Z};\boldsymbol{\pi}) = \prod_{ij} p(X_{ij}|\boldsymbol{Y}_i, \boldsymbol{Z}_j;\boldsymbol{\pi})$$
$$X_{ij}|Y_{iq} = 1, Z_{jl} = 1 \overset{\text{ind}}{\sim} \phi_{ql}(X_{ij}) \ .$$

  with $\boldsymbol{\alpha} \in \mathbf{S}_{k_1-1}$, $\boldsymbol{\beta} \in \mathbf{S}_{k_2-1}$ and $\phi_{ql}(X_{ij})$ the density of the conditional distribution of X_{ij} depending on the group memberships of row i and column j.

As in the SBM, the conditional distribution of X_{ij} can follow various densities to model different data types. If the data matrix $\boldsymbol{X}$ is binary, the conditional distribution of X_{ij} is a Bernoulli of parameter $\boldsymbol{\pi} = (\pi_{ql}; q = 1, ..., k_1; l = 1, ..., k_2)$:

.

In the binary case, the parameters of the LBM are $\theta = (\boldsymbol{\alpha}, \boldsymbol{\beta}, \boldsymbol{\pi})$ and the probability mass function of $\boldsymbol{X}$ can be written as:

$$p(\boldsymbol{X}; \theta) = \sum_{(\boldsymbol{Y}, \boldsymbol{Z}) \in I \times J} \left(\prod_{i,q} {\alpha_q}^{Y_{iq}} \right) \left(\prod_{j,l} {\beta_l}^{Z_{jl}} \right) \left(\prod_{i,j,q,l} \phi(X_{ij}; \pi_{ql})^{Y_{iq} Z_{jl}} \right) ,$$

where $\phi(X_{ij}; \pi_{ql}) = \pi_{ql}^{X_{ij}} (1 - \pi_{ql})^{1 - X_{ij}}$ is the mass function of a Bernoulli variable and where I (resp. J) denotes the set of all possible partitions of rows (resp. columns) into k_1 (resp. k_2) groups.

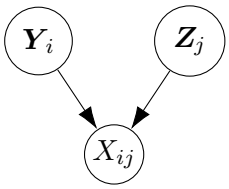


Figure 1.10: Random variable dependencies in the Latent Block Model

Latent Position Cluster Model (LPCM): The LPCM proposed by Handcock et al. [2007] extends the latent space model of Hoff et al. [2002] introducing a graph clustering based on latent distances between vertices. The model assumes that each vertex has an unobserved position in a d-dimensional Euclidean latent space and that the presence or absence of a edge between two vertices is independent of all other edges given the positions in space of these two vertices.

The latent positions $\boldsymbol{Y}_i$ are drawn from a finite mixture of k_1 multivariate normal distributions, each representing a different class of vertices. The conditional distribution of the presence of the edge X_{ij} is Bernoulli distributed parametrized by the distances between latent positions of vertices in latent space. The LPCM is formally defined as followed:

$$\begin{cases} \forall i,\ \boldsymbol{Y}_i \overset{\text{iid}}{\sim} \sum_q \alpha_q \mathcal{N}\left(\boldsymbol{\mu}_q, \sigma_q^2 \boldsymbol{I}_d\right) \\ \forall i, j,\ X_{ij} | \boldsymbol{Y}_i, \boldsymbol{Y}_j \overset{\text{ind}}{\sim} \mathcal{B}(\text{expit}\left(\gamma_0 Z_{ij} + \gamma_1 |\boldsymbol{Y}_i - \boldsymbol{Y}_j|\right)) \end{cases}$$

with expit $= 1/(1 + \exp(-x))$, $\boldsymbol{\mu}_q \in \mathbb{R}^d$, $\sigma_q^2 \in \mathbb{R}_+^*$, $\gamma_0 \in \mathbb{R}$, $\gamma_1 \in \mathbb{R}_+^*$, and $\boldsymbol{\alpha} \in \mathbf{S}_{k_1 - 1}$. α_q is the probability that a vertex belongs to the q^{th} class, and Z_{ij} being a covariate carrying a possible additional information about the edge between i

and j.

Mixed Membership SBM (MMSBM): The hard assignment of vertices to blocks in the SBM and LBM framework may be too restrictive for modeling complex networks as the diversity of interactions may not be well captured. Airoldi et al. [2008] proposed a model more expressive than the SBM, allowing nodes to pertain to multiple communities.

If $\boldsymbol{Y}_{i\to j}$ denotes the specific block membership of node i when it connects to node j, the MMSBM posits:

$$\begin{cases} \forall i,\ \boldsymbol{\alpha}_i \overset{\text{iid}}{\sim} \text{Dirichlet}(\boldsymbol{\gamma}), & \text{with } \boldsymbol{\gamma} \in \mathbb{R}_{\geq 0}^{k_1} \\ \forall i,j,\ \boldsymbol{Y}_{i\to j} \overset{\text{iid}}{\sim} \mathcal{M}(1;\boldsymbol{\alpha}_i) & \\ \forall i,j,\ \boldsymbol{Y}_{j\to i} \overset{\text{iid}}{\sim} \mathcal{M}(1;\boldsymbol{\alpha}_j) & \\ \forall i,j,\ X_{ij}|\boldsymbol{Y}_{i\to j},\boldsymbol{Y}_{j\to i} \overset{\text{ind}}{\sim} \mathcal{B}\big(\boldsymbol{Y}_{i\to j}^T \boldsymbol{\pi} \boldsymbol{Y}_{j\to i}\big) & \text{with } \boldsymbol{\pi} \in [0,1]^{k_1\times k_1} \end{cases}$$

The particularity of this model is that the group membership of each node is context dependent: each node may assume different membership when interacting with different peers.

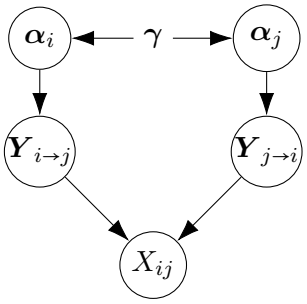


Figure 1.11: Random variable dependencies in the Mixed Membership Stochastic Block Model

Overlapping Stochastic block model (OSBM): in a similar vein of the MMSBM, Latouche et al. [2011] proposed the OSBM that is an extension of the SBM allowing each vertex to belong to zero or multiple classes. The constraint on the multinomial distribution parameter $\boldsymbol{\alpha} \in \mathbf{S}_{k_1-1}$ is relaxed implying that for each vertex i of the network, the latent vector $\boldsymbol{Y}$, of k_1 independent Boolean variables is drawn from a multivariate Bernoulli distribution allowing overlapping clusters:

$$\begin{cases} \forall i,\ \boldsymbol{Y}_i \overset{\text{iid}}{\sim} \prod_q \mathcal{B}(Y_{iq};\alpha_q) \\ \forall i,j,\ X_{ij}|\boldsymbol{Y}_i,\boldsymbol{Y}_j \overset{\text{ind}}{\sim} \mathcal{B}\big(\text{expit}\left(\boldsymbol{Y}_i^T \boldsymbol{W} \boldsymbol{Y}_j + \boldsymbol{Y}_i^T \boldsymbol{U} + \boldsymbol{Y}_j^T \boldsymbol{V} + W^*\right)\big) \end{cases}$$

with $\text{expit}(x) = 1/(1+\exp(-x))$, and $\alpha_q \in [0,1]$. $\boldsymbol{W}$ is a $(k_1 \times k_1)$-real weights matrix describing the interactions between classes, and $\boldsymbol{U}$ (resp. $\boldsymbol{V}$) a k_1-real weights vector describing the tendency of each classes to receive (resp. emit) an edge. Finally, the scalar $W^\star$ is the bias modeling the sparsity.

The product of Bernoulli distributions offers an elegant solution for modeling outliers as the model uses the null component such $\boldsymbol{Y}_i = \overrightarrow{0}$ rather than using an additional class for outliers if vertex i should not be classified.

Deep Generative Overlapping SBM (or DGLFRM): Mehta et al. [2019] proposed a deep generative model for overlapping community discovery and link prediction.

Each vertex is modelled with two latent variables : the first one is a binary random vector $\boldsymbol{A}_i$ encoding for the group membership and the second one $\boldsymbol{B}_i$ is a real-valued vector with a homoskedastic multivariate Gaussian prior encoding for the strength of the membership in each class. These two latent variables can be squeezed into a single one, denoted $\boldsymbol{Y}_i$, applying a dot product operation between $\boldsymbol{A}_i$ and $\boldsymbol{B}_i$. In this setting, each vertex is modelled as a sparse real-valued vector, providing a more flexible but less interpretable representation that the one provided by the OSBM, which assumes the node embedding $\boldsymbol{Y}_i$ to be a binary vector.

Given the node embeddings $\boldsymbol{Y}$, the model generates edges $(X_{ij})_{i,j}$ with independent Bernoulli distributions parametrized by the probability function $f(\boldsymbol{Y}_i, \boldsymbol{Y}_j)$; f being a function with one or several deterministic nonlinear transformations. The Deep Generative Overlapping SBM is formally defined as followed:

$$\begin{cases} \forall q,\ \pi_q \overset{\text{ind}}{\sim} \text{Beta}(\alpha, 1) \\ \forall i, q,\ A_{iq} \overset{\text{ind}}{\sim} \mathcal{N}\left(0, \sigma^2\right) \\ \forall i, q,\ B_{iq} \overset{\text{ind}}{\sim} \mathcal{B}\left(\prod_q \pi_q\right) \\ \forall i,\ \boldsymbol{Y}_i = \boldsymbol{A}_i \odot \boldsymbol{B}_i \\ \forall i, j,\ X_{ij} | \boldsymbol{Y}_i, \boldsymbol{Y}_j \overset{\text{ind}}{\sim} \mathcal{B}(f(\boldsymbol{Y}_i, \boldsymbol{Y}_j)) \end{cases}$$

with $\alpha \in \mathbb{R}_+^*$ and $\sigma^2 \in \mathbb{R}_+^*$. Modeling $\boldsymbol{B}_i$ using a stick-breaking prior Teh et al. [2007] enables learning the number of classes k_1.

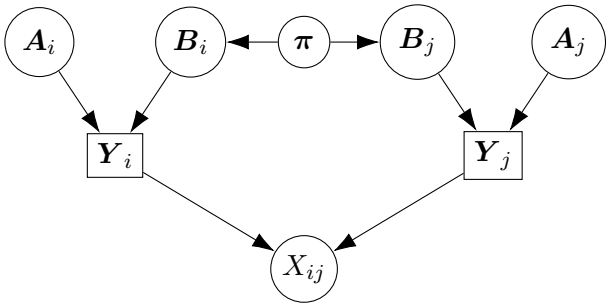


Figure 1.12: Random variable dependencies in the Deep Generative Overlapping SBM. $\boldsymbol{Y}_i$ is deterministic, generated by $\boldsymbol{A}_i \odot \boldsymbol{B}_j$.

1.3 Contributions

Collaborative filtering relies on a sparse rating matrix, where each user rates a few products, to propose recommendations. The approach consists to approximate the sparse rating matrix with a simple model whose regularities allow to fill in the missing entries. The latent block model is a generative co-clustering model that can provide such an approximation.

Like most standard clustering or co-clustering methods, the Latent Block Model assumes complete information and cannot be fitted on incomplete data, or may provide misleading conclusions when the missingness mechanism is informative. In Chapter 2, we introduce a missingness model that is coupled to the Latent Block Model (LBM) to process missing data. The overall model accounts for an informative absence of data (MNAR). We equip this model with a variational EM algorithm to infer its parameters and propose an Integrated Completed Likelihood (ICL) criterion to tackle model selection.

In Chapter 3, we are interested in providing the same treatment to users of the recommender system, regardless of a sensitive attribute such as gender, age, ethnicity, socio-economic group, etc. We propose a definition of fairness specific to recommender systems, requiring that the ranking of items be independent of the users' sensitive attribute. We propose a Latent Block Model that incorporates the exogenous sensitive attributes to ensure fair recommendations. We demonstrate that our model ensures approximately fair recommendations provided that the classification of users approximately respects statistical parity.

For block models, the standard inference algorithm presents computational difficulties in dealing with the large recommendation graphs that are handled in collaborative filtering. The first contribution of Chapter 4 is a variational inference algorithm for the stochastic block model and the latent block model for sparse graphs, which leverages on the sparseness of edges to scale up to a very large number of nodes. It is thanks to this inference that we have been able, in Chapter 2, to use our LBM with MNAR on real recommendation graphs. The second contribution of Chapter 4 is a generic inference algorithm for block models. This algorithm replaces the usual variational EM approach with a stochastic optimization of the variational criterion. This inference is particularly well suited to process large, possibly dense networks with elaborate block models. It was used to carry out the experiments in Chapter 3.

Bibliography

Edoardo Maria Airoldi, David M Blei, Stephen E Fienberg, and Eric P Xing. Mixed membership stochastic blockmodels. *Journal of machine learning research*, 2008. 21

Robert M Bell and Yehuda Koren. Scalable collaborative filtering with jointly derived neighborhood interpolation weights. In *Seventh IEEE International Conference on Data Mining (ICDM 2007)*, pages 43–52. IEEE, 2007. 5

Christophe Biernacki, Gilles Celeux, and Gérard Govaert. Assessing a mixture model for clustering with the integrated completed likelihood. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(7):719–725, 2000. doi: 10.1109/34.865189. 17

David M Blei, Andrew Y Ng, and Michael I Jordan. Latent dirichlet allocation. *the Journal of machine Learning research*, 3:993–1022, 2003. 5

John S Breese, David Heckerman, and Carl Kadie. Empirical analysis of predictive algorithms for collaborative filtering. *arXiv preprint arXiv:1301.7363*, 2013. 5

Chao Chen, Dongsheng Li, Qin Lv, Junchi Yan, Li Shang, and Stephen Chu. Gloma: Embedding global information in local matrix approximation models for collaborative filtering. *Proceedings of the AAAI Conference on Artificial Intelligence*, 31(1), Feb. 2017. URL https://ojs.aaai.org/index.php/AAAI/article/view/10752. 9

Jean-Jacques Daudin, Franck Picard, and Stephane Robin. A mixture model for random graphs. *Statistics and Computing*, 18(2):173–183, 2008. doi: 10.1007/s11222-007-9046-7. URL https://hal.archives-ouvertes.fr/hal-01197587. 17

William E Donath and Alan J Hoffman. Lower bounds for the partitioning of graphs. In *Selected Papers Of Alan J Hoffman: With Commentary*, pages 437–442. World Scientific, 2003. 14

Luca Donetti and Miguel Muñoz. Detecting network communities: a new systematic and efficient algorithm. *Journal of Statistical Mechanics: Theory and Experiment*, 2004, 04 2004. doi: 10.1088/1742-5468/2004/10/P10012. 14

Paul Erdős and Alfréd Rényi. On the evolution of random graphs. *Publ. Math. Inst. Hung. Acad. Sci*, 5(1):17–60, 1960. 15

Thomas George and Srujana Merugu. A scalable collaborative filtering framework based on co-clustering. In *Fifth IEEE International Conference on Data Mining (ICDM'05)*, pages 4–pp. IEEE, 2005. 6

Kostadin Georgiev and Preslav Nakov. A non-iid framework for collaborative filtering with restricted boltzmann machines. In *International conference on machine learning*, pages 1148–1156. PMLR, 2013. 10

Anna Goldenberg, Alice X Zheng, Stephen E Fienberg, and Edoardo M Airoldi. A survey of statistical network models, 2009. 16

Olivier Gouvert, Thomas Oberlin, and Cédric Févotte. Ordinal non-negative matrix factorization for recommendation. In Hal Daumé III and Aarti Singh, editors, *Proceedings of the 37th International Conference on Machine Learning*, volume 119 of *Proceedings of Machine Learning Research*, pages 3680–3689. PMLR, 13–18 Jul 2020. URL http://proceedings.mlr.press/v119/gouvert20a.html. 8

Gérard Govaert and Mohamed Nadif. Block clustering with Bernoulli mixture models: Comparison of different approaches. *Computational Statistics & Data Analysis*, 52(6):3233–3245, February 2008. doi: 10.1016/j.csda.2007.09.007. 17

Mark S Handcock, Adrian E Raftery, and Jeremy M Tantrum. Model-based clustering for social networks. *Journal of the Royal Statistical Society: Series A (Statistics in Society)*, 170(2):301–354, 2007. 20

Xiangnan He, Lizi Liao, Hanwang Zhang, Liqiang Nie, Xia Hu, and Tat-Seng Chua. Neural collaborative filtering. In *Proceedings of the 26th international conference on world wide web*, pages 173–182, 2017. 12

José Miguel Hernández-Lobato, Neil Houlsby, and Zoubin Ghahramani. Probabilistic matrix factorization with non-random missing data. In *Proceedings of the 31st International Conference on Machine Learning*, pages 1512–1520, 2014. 7

Peter D Hoff, Adrian E Raftery, and Mark S Handcock. Latent space approaches to social network analysis. *Journal of the american Statistical association*, 97(460):1090–1098, 2002. 20

Toshihiro Kamishima, Shotaro Akaho, Hideki Asoh, and Jun Sakuma. Recommendation independence. In *Conference on Fairness, Accountability and Transparency*, pages 187–201, 2018. 7

Diederik P Kingma and Max Welling. Auto-encoding variational bayes, 2014. 11

Yehuda Koren, Robert Bell, and Chris Volinsky. Matrix factorization techniques for recommender systems. *Computer*, 42(8):30–37, 2009. 7

Joseph B Kruskal. On the shortest spanning subtree of a graph and the traveling salesman problem. *Proceedings of the American Mathematical society*, 7(1):48–50, 1956. 13

Pierre Latouche, Etienne Birmelé, and Christophe Ambroise. Overlapping stochastic block models with application to the French political blogosphere. *The Annals of Applied Statistics*, 5(1):309–336, Mar 2011. ISSN 1932-6157. doi: 10.1214/10-aoas382. 21

Joonseok Lee, Seungyeon Kim, Guy Lebanon, Yoram Singer, and Samy Bengio. Llorma: Local low-rank matrix approximation. *Journal of Machine Learning Research*, 17(15):1–24, 2016. URL http://jmlr.org/papers/v17/14-301.html. 8

Xiaopeng Li and James She. Collaborative variational autoencoder for recommender systems. In *Proceedings of the 23rd ACM SIGKDD international conference on knowledge discovery and data mining*, pages 305–314, 2017. 11

Dawen Liang, Rahul G Krishnan, Matthew D Hoffman, and Tony Jebara. Variational autoencoders for collaborative filtering. In *Proceedings of the 2018 world wide web conference*, pages 689–698, 2018. 11

Benjamin M Marlin. Modeling user rating profiles for collaborative filtering. *Advances in neural information processing systems*, 16:627–634, 2003. 5

Catherine Matias and Stéphane Robin. Modeling heterogeneity in random graphs through latent space models: a selective review. *ESAIM: Proceedings and Surveys*, 47:55–74, 2014. 16

Nikhil Mehta, Lawrence Carin Duke, and Piyush Rai. Stochastic blockmodels meet graph neural networks. In Kamalika Chaudhuri and Ruslan Salakhutdinov, editors, *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pages 4466–4474. PMLR, 09–15 Jun 2019. URL http://proceedings.mlr.press/v97/mehta19a.html. 22

Andriy Mnih and Russ R Salakhutdinov. Probabilistic matrix factorization. *Advances in neural information processing systems*, 20:1257–1264, 2007. 7

Radford M Neal. *Probabilistic inference using Markov chain Monte Carlo methods*. Department of Computer Science, University of Toronto Toronto, Ontario, Canada, 1993. 8

M. E. J. Newman. Fast algorithm for detecting community structure in networks. *Physical Review E*, 69(6), Jun 2004. ISSN 1550-2376. doi: 10.1103/physreve.69.066133. URL http://dx.doi.org/10.1103/PhysRevE.69.066133. 14

Pascal Pons and Matthieu Latapy. Computing communities in large networks using random walks. In *International symposium on computer and information sciences*, pages 284–293. Springer, 2005. 14, 15

Karl Rohe, Sourav Chatterjee, and Bin Yu. Spectral clustering and the high-dimensional stochastic blockmodel. *The Annals of Statistics*, 39(4): 1878–1915, Aug 2011. ISSN 0090-5364. doi: 10.1214/11-aos887. URL http://dx.doi.org/10.1214/11-AOS887. 15

Ruslan Salakhutdinov and Andriy Mnih. Bayesian probabilistic matrix factorization using markov chain monte carlo. In *Proceedings of the 25th international conference on Machine learning*, pages 880–887, 2008. 8

Ruslan Salakhutdinov, Andriy Mnih, and Geoffrey Hinton. Restricted boltzmann machines for collaborative filtering. In *Proceedings of the 24th international conference on Machine learning*, pages 791–798, 2007. 10

Badrul Sarwar, George Karypis, Joseph Konstan, and John Riedl. Item-based collaborative filtering recommendation algorithms. In *Proceedings of the 10th international conference on World Wide Web*, pages 285–295, 2001. 5

Satu Elisa Schaeffer. Graph clustering. *Computer Science Review*, 1(1):27–64, 2007. ISSN 1574-0137. doi: https://doi.org/10.1016/j.cosrev.2007.05.001. URL https://www.sciencedirect.com/science/article/pii/S1574013707000020. 13

Suvash Sedhain, Aditya Krishna Menon, Scott Sanner, and Lexing Xie. Autorec: Autoencoders meet collaborative filtering. In *Proceedings of the 24th international conference on World Wide Web*, pages 111–112, 2015. 10

Paul Smolensky. Information processing in dynamical systems: Foundations of harmony theory. Technical report, Colorado Univ at Boulder Dept of Computer Science, 1986. 9

Yee Whye Teh, Dilan Grür, and Zoubin Ghahramani. Stick-breaking construction for the indian buffet process. In Marina Meila and Xiaotong Shen, editors, *Proceedings of the Eleventh International Conference on Artificial Intelligence and Statistics*, volume 2 of *Proceedings of Machine Learning Research*, pages 556–563, San Juan, Puerto Rico, 21–24 Mar 2007. PMLR. URL http://proceedings.mlr.press/v2/teh07a.html. 22

Ulrike Von Luxburg. A tutorial on spectral clustering. *Statistics and computing*, 17(4):395–416, 2007. 15

Ke Zhou, Shuang-Hong Yang, and Hongyuan Zha. Functional matrix factorizations for cold-start recommendation. In *Proceedings of the 34th international ACM SIGIR conference on Research and development in Information Retrieval*, pages 315–324, 2011. 7

Chapter 2

Missing data in random graph clustering

> When dealing with real data, the practising statistician should explicitly consider the process that causes missing data far more often than he does.
>
> Donald B. Rubin

Contents

2.1 Introduction

Missing data can be informative. Ignoring this information can lead to misleading conclusions when the data model does not allow information to be extracted from the missing data. The datasets extracted from recommender systems are usually extremely sparse, with a high proportion of missing ratings, that is, ratings that were not provided by the users. Most collaborative filtering methods do not have a principled method for extracting information from this missing data.

In this chapter, a discussion on existing Missing Not At Random data methods is first carried out. Then a co-clustering model, based on the Latent Block Model, is presented to take advantage of nonignorable nonresponse. A variational expectation-maximization algorithm is derived to perform inference, a model selection criterion is presented and we assess the proposed approach on a simulation study. We show that our model is useful on rather small datasets by using it on the voting records from the lower house of the French Parliament, where our analysis brings out relevant groups of MPs and texts, together with a sensible interpretation of the behavior of non-voters.

A shortcoming of the proposed inference is it can not be used on large standard recommender system datasets for computational reasons. Thus, we propose an adaptation of the inference to scales up to collaborative filtering data by taking advantage of the scarcity of the observed ratings.

2.1.1 Missing data

The problem of missing data has been an issue for a long time in data analysis as most statistical methods do not account for missing observations. When facing missing values in data, some ad-hoc solutions consist in ignoring the records with missing values (listwise or pairwise deletion) or in naive data imputation (mean, median, regression methods, etc.). These ad-hoc solutions may give satisfying results when the proportion of missing cases is low or when the missingness is generated by a particular mechanism (see MCAR definition below). In other cases, these methods may have a significant effect on the conclusions that can be drawn from the data.

Rubin [1976] gives a probabilistic definition to missingness in which the process that governs the observation probability is called the missing data mechanism. He classifies missing data mechanisms into three categories and draws consequences on model estimators when missingness mechanisms are improperly modeled. These three classes of missing data problems are:

1. *Missing Completely At Random* (MCAR): let M denote the missing data indicator, such that $M = 1$ when X is observed and $M = 0$ otherwise; in a Missing Completely at Random mechanism, the observation of X does

not depend on any data, either observed or missing. Mathematically speaking : $M \perp\!\!\!\perp X$.

2. *Missing At Random* (MAR): let $X^{(o)}$ and $X^{(m)}$ denote respectively the observed and the missing data; in a Missing At Random mechanism, the missing data indicator M doesn't depend on the missing data $X^{(m)}$ but depends on the observed data $X^{(o)}$. Mathematically speaking : $M \perp\!\!\!\perp X^{(m)} | X^{(o)}$.

3. *Missing Not At Random* (MNAR): the missing data indicator M depends on the value of the data X, either observed or missing. Mathematically speaking : $M \not\perp\!\!\!\perp X$. Missing Not At Random is also known as non-ignorable missingness.

Under the MAR or MCAR hypothesis, no information on the generation of data can be extracted from its absence. In likelihood-based imputation methods, the missingness is said to be ignorable and the parameters θ generating data X can be estimated via the maximum likelihood of observed data only: $\widehat{\theta} = \arg\max_\theta p(X^{(o)}, M; \theta, \phi) = \arg\max_\theta p(X^{(o)}; \theta)$, with ϕ the parameters generating missingness.

Under a MNAR missingness, the absence of data is informative, and ignoring this information in likelihood-based imputation methods may lead to strong biases in estimation [Little and Rubin, 1986]. A few models relying on likelihood imputation with MNAR assumptions are presented in the following section.

Another common approach to dealing with missing values is multiple imputation. It aims to fill in missing values by using multiple imputed values to take into account the uncertainty in the imputations whereas single imputation does not. Each set of imputed missing values is used to create a plausible completed dataset each of which is to be analyzed using standard statistical tools. One fundamental method of multiple imputations is the *repeated imputation* proposed by Rubin [1987] where "the repeated imputations are draws from the posterior predictive distribution of the missing values under a specific model". Repeated imputation is based on a generative Bayesian approach for both the data and the missing data mechanism. Although multiple imputation methods are widely and successfully used in fields such as epidemiology and clinical researches Sterne et al. [2009], Jakobsen et al. [2017], this thesis is focusing on likelihood-based imputation methods only.

2.1.2 Missing data in collaborative filtering

A common implicit assumption in collaborative filtering is that ratings are Missing At Random (MAR): the presence/absence of ratings is assumed to convey no information whatsoever about the value of this rating. However, this assumption is intuitively incorrect in many situations, and can often be

contradicted by showing that there is a clear dependence of rating frequency on the underlying preference level [Marlin et al., 2007, 2012]. This observation lends support to the hypothesis that ratings are generated by a Missing Not At Random (MNAR) process, where missingness depends on the actual rating that would be given by users.

Figure 2.1 exemplifies a situation where data is MAR or MNAR on the Horror-Lover toy dataset of Hernández-Lobato et al. [2014]. The left-hand plot shows the *complete data* matrix, that is, the full matrix of "true ratings" that would be given by all users to all items (users and items are ordered here according to their true class membership). When the probability to observe a rating is the same for all entries, the missing data is completely MAR, leading to the *observed data* at the center of Figure 2.1. On the right-hand plot is observed a setting where the probability to observe a rating is not the same for all entries (MNAR): the first half of users tend to rate what they like ("Positive" raters) and the second half tend to rate what they dislike ("Negative" raters). The rating frequency in all blocks of the matrix is clearly an indicator of the values of ratings. In this missingness scenario, not observing a rating is informative and the recommender system should take advantage of this information.

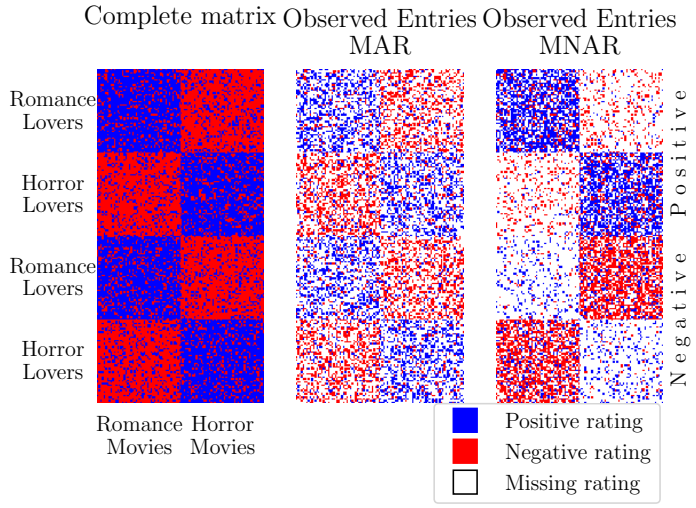


Figure 2.1: Ratings of the romance-horror toy dataset of Hernández-Lobato et al. [2014]: left, complete data; center, observed data with MAR entries; right, observed data with MNAR entries.

In a collaborative filtering context, the MNAR assumption is appropriate [Marlin et al., 2012]. Indeed, some people tend to give their opinion more often when they are satisfied with an item, whereas others rate mainly the items with which they are dissatisfied. A study conducted by Yahoo! Music's LaunchCast Radio service consisting of a user survey on how their preferences

affect which songs they choose to rate, indicates clearly that the choice to rate a song actually depends on the user's opinion (see Table 2.1). Thus, we can

Table 2.1: Study conducted by Yahoo! Music's LaunchCast Radioservice, in which users self-reported how often they rate songs based on their level of preference. Results reproduced from [Marlin et al., 2007].

Rating Frequency	Preference Level				
	Hate	Don't like	Neutral	Like	Love
Never	6.76%	4.69%	2.33%	0.11%	0.07%
Very Infrequently	1.59%	4.17%	9.46%	0.54%	0.35%
Infrequently	1.63%	4.44%	24.87%	1.48%	0.20%
Often	12.46%	22.50%	26.83%	25.30%	5.46%
Very Often	77.56%	64.20%	36.50%	72.57%	93.91%

safely assume that all users each have a propensity to give their opinion, and similarly, that each item has its own propensity to receive a rating. The models of missingness presented in this Chapter relies on several latent variables that encode some form of propensity. The propensity could represent the link between the notoriety of an item, measured by the associated ratings proportion, and the probability to observe its ratings. The more notorious an item is, the more likely it is to be evaluated. The propensity could also represent exogenous factors such as the price (the more expensive an item is, the more likely it is to be evaluated). This justifies the fact that, depending on the item, the propensities can be different.

2.1.3 Graph models for Missing Not At Random data

2.1.3.1 Double mixture models for MNAR data

Stochastic Block Model: Tabouy et al. [2020] deal with inference in the Stochastic Block Model when the network is not fully observed. They consider undirected binary networks where all the nodes are observed but information regarding the presence/absence of an edge is missing for some dyads. In this setting, the network can be represented with an adjacency matrix filled with zeros, ones and missing values. They study several NMAR missing designs and propose variational approaches to perform inference under this series of designs:

- double standard sampling where each dyad has the same probability to

be observed conditionally to its value.

$$p(\, M_{ij} = 1 | X_{ij} = 1 \,) = \pi_1$$
$$p(\, M_{ij} = 1 | X_{ij} = 0 \,) = \pi_0 \ ,$$

with $\pi_1,\ \pi_0 \in]0, 1[$.

- star sampling where all dyads from a node selected with probability

$$\pi_i = \text{expit}\left(a + b \sum_j X_{ij} \right) \ ,$$

are observed. In this setting, the probability to observe a dyad depends on the degree of its nodes.

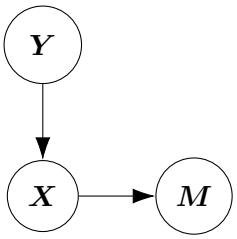
- class node sampling where all dyads from a node selected with probability depending on its group

$$\pi_q = p(Y_{iq} = 1) \ ,$$

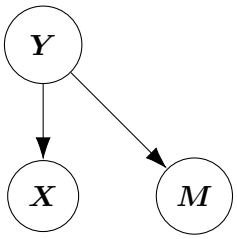
are observed.

- class dyad sampling where the probability to observe a dyad depends on the class (q, l) of its nodes

$$p\big(\, M_{ij} = 1 \big| Y_{iq} Y_{jl} = 1 \,\big) = \pi_{ql} \ .$$



(a) double standard and star sampling designs



(b) class node and class dyad sampling designs

Figure 2.2: Directed acyclic graph (DAG) of random variable dependencies in the MNAR Stochastic Block Models from Tabouy et al. [2020]. $\boldsymbol{M}$ is the mask matrix, $\boldsymbol{X}$ the adjacency matrix and $\boldsymbol{Y}$ the indicator matrix of the latent clusters.

Associated to these MNAR models, the authors propose a general inference for binary data based on a variational expectation-maximization algorithm (see Section 2.2.2). They propose the following restriction on the variational

distribution:

$$q_\gamma = \prod_i \mathcal{M}\left(1; \tau_i^{(Y)}\right) \times \prod_{(i,j)\in\mathcal{D}^m} \mathcal{B}(X_{ij}; \nu_{ij}) \ ,$$

where $\mathcal{D}^m$ is the set of all missing dyads. A drawback of this inference is that it could be computationally heavy if the network is large or if the proportion of missing value is important as a variational parameter is introduced for each missing dyad.

Latent Block Model: Up to our knowledge, all existing co-clustering methods consider that missing data is either MCAR or MAR [Selosse et al., 2020a, Jacques and Biernacki, 2018, Papalexakis et al., 2013], except one proposed by Corneli et al. [2020] used to co-cluster ordinal data. Corneli et al. [2020] introduce a Missing Not At Random mechanism in an ordinal Latent Block Model. Their model relies on the binary LBM to manage data sparsity:

$$\begin{aligned}
&\boldsymbol{Y}_i \overset{\text{iid}}{\sim} \mathcal{M}(1;\boldsymbol{\alpha}), && \boldsymbol{\alpha} \in \mathbf{S}_{k_1-1} \\
&\boldsymbol{Z}_j \overset{\text{iid}}{\sim} \mathcal{M}(1;\boldsymbol{\beta}), && \boldsymbol{\beta} \in \mathbf{S}_{k_2-1} \\
&\left(M_{ij} | Y_{iq} Z_{jl} = 1\right) \overset{\text{ind}}{\sim} \mathcal{B}\left(\pi_{ql}\right), && p \in [0,1] \ ,
\end{aligned}$$

Latent Gaussian random variables (denoted $\boldsymbol{X}^\star$) are used to generate ordinal entries in a data matrix $\boldsymbol{X}$:

$$\begin{aligned}
&\left(X^*_{ij} | Y_{iq} Z_{jl} = 1\right) \overset{\text{ind}}{\sim} \mathcal{N}\left(\mu_{ql}, \sigma^2\right), && \mu_{ql} \in \mathbb{R}, \sigma \in \mathbb{R}^*_+ \\
&\left(X_{ij} | X^*_{ij}, M_{ij}\right) = \begin{cases} \sum_{k=1}^K k \, \mathbb{1}_{]\zeta_{k-1};\zeta_k]}(X^*_{ij}) & \text{if} \quad M_{ij} = 1 \\ \text{NA} & \text{if} \quad M_{ij} = 0 \end{cases}
\end{aligned}$$

with $\zeta_0 = -\infty < \zeta_1 < \ldots < \zeta_{K-1} < \zeta_K = \infty$, fixed thresholds.

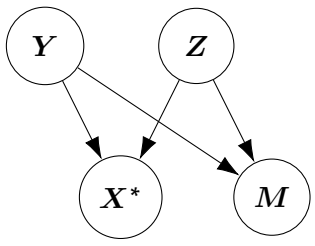


Figure 2.3: Variable dependencies in the MNAR Latent Block Model of Corneli et al. [2020]. $\boldsymbol{M}$ is the mask matrix, $\boldsymbol{X}^\star$ the latent continuous Gaussian random variable, $\boldsymbol{Y}$ and $\boldsymbol{Z}$ are respectively the indicator matrices of the row and column latent clusters.

Their model is very parsimonious as it assumes that both data and missingness are only dependent on the row and column clusters. In this setting, they are able to consider MNAR data even if they suppose that missingness depends

indirectly from the value of the data. Prediction performances on missing data in a collaborative filtering context can suffer from such a parsimony. This missingness mechanism is similar to the class dyad sampling design proposed by Tabouy et al. [2020] for the SBM. The model we propose in the next Section is less parsimonious, thus more flexible, as it supposes that missingness depends both on the value of the data and on the row and column indexes (not only on their respective cluster indexes). Our model can be easily reused for other probabilistic co-clustering models, as it is only weakly coupled to the generative model of the full data matrix.

2.1.3.2 Simple mixture models for MNAR data

In the simple clustering framework, few mixture models handling MNAR data have been proposed.

CPT-v and Logit-vd model: in a collaborative filtering context, Marlin et al. [2011] are the first to model missing not at random rating responses. They propose two multinomial mixture models to cluster users.

The first model includes a combination of the multinomial mixture model with a simple missing data mechanism where the probability that a rating is missing depends only on the value of that rating. They refer to this mechanism as CPT-v since it is parameterized using a simple conditional probability table:

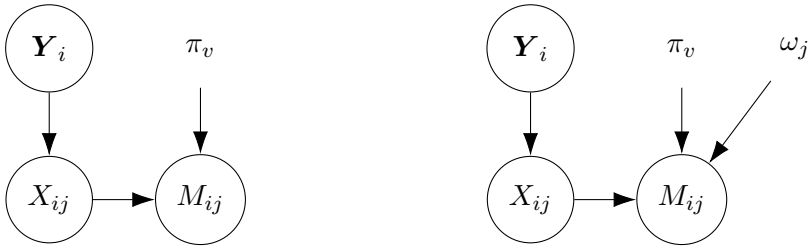
$$\begin{aligned} p(\boldsymbol{Y}_{iq} = 1) &= \alpha_q, & \boldsymbol{\alpha} &\in \mathbf{S}_{k_1-1} \\ p(X_{ij} = v | \boldsymbol{Y}_{iq} = 1) &= \beta_{vjq}, & \boldsymbol{\beta}_{jq} &\in \mathbf{S}_{k_v-1} \\ p(M_{ij} | X_{ij} = v) &= \pi_v, & \pi_v &\in]0,1[\ . \end{aligned}$$

The second model, Logit-vd, also uses the multinomial mixture model but its missingness model is more flexible as the probability of a data entry to be missing depends both on the value of the underlying data and on the characteristics of the items. The model makes use of an additive logistic model, hence its name:

$$\begin{aligned} p(\boldsymbol{Y}_{iq} = 1) &= \alpha_q, & \boldsymbol{\alpha} &\in \mathbf{S}_{k_1-1} \\ p(X_{ij} = v | \boldsymbol{Y}_{iq} = 1) &= \beta_{vjq}, & \boldsymbol{\beta}_{jq} &\in \mathbf{S}_{k_v-1} \\ p(M_{ij} | X_{ij} = v) &= \operatorname{expit}(\pi_v + \omega_j), & \pi_v &\in \mathbb{R},\ \omega_j \in \mathbb{R} \ . \end{aligned}$$

Bayesian-BM/OR model: still in a collaborative filtering context, Kim and Choi [2014] propose Bayesian-BM/OR, a simple mixture model of binomials in a Bayesian formalism.

The binomial distribution is suitable for modelling discrete, finite, and ordered rating values while the mixture can capture the simple intuition that



(a) CPT-v model (b) Logit-vd model

Figure 2.4: DAG of random variable dependencies in the CPT-v and Logit-vd models from Marlin et al. [2011]. $\boldsymbol{M}$ is the mask matrix, $\boldsymbol{X}$ the rating matrix and $\boldsymbol{Y}$ the indicator matrix of the user clusters.

users form clusters according to their preferences:

$$p(\boldsymbol{Y}_{iq} = 1) = \alpha_q, \qquad \boldsymbol{\alpha} \in \mathbf{S}_{k_1 - 1}$$
$$p(X_{ij} = v | \boldsymbol{Y}_{iq} = 1) = \binom{k_{v-1}}{v} \beta_{jq}^{v} (1 - \beta_{jq})^{k_{v-1} - v}, \qquad \beta_{jq} \in [0, 1] \ .$$

Their missingness is modeled by three factors, related to the user, the item and the rating value, all three being modeled by Bernoulli variables respectively named $\boldsymbol{U}$, $\boldsymbol{V}$, $\boldsymbol{R}$ and combined together by a "or" logical operator. The parameter of the random variable R_{ij} depends on the value of the ratings X_{ij}; the missingness mechanism is thus of type MNAR:

$$\begin{aligned} p(U_{ij} = 1) &= \mu_i \\ p(V_{ij} = 1) &= \omega_j \\ p(R_{ij} = 1 | X_{ij} = v) &= \pi_v \\ p(M_{ij} = 1 | X_{ij} = v) &= p(U_{ij} = 1) p(V_{ij} = 1) p(R_{ij} = 1 | X_{ij} = v) \ . \end{aligned}$$

The choice of this missingness model is motivated by algorithmic considerations. The proposed variational inference is indeed computationally efficient, since the complexity depends on the number of observed ratings instead of the usual dependence on the size of the rating data matrix. This missingness model offers more flexibility than the Logit-vd model as it considers a more realistic per user rating behaviour.

2.1.3.3 Matrix factorization with MNAR data

Also related to missing data but not to clustering, MNAR is also considered in the matrix factorization framework. The models presented is this section have been designed for a collaborative filtering application. The matrix factorization models exhibit state-of-the-art predictive performance in collaborative filtering but assume that data is missing at random. The following methods extend the matrix factorization framework to consider MNAR data while

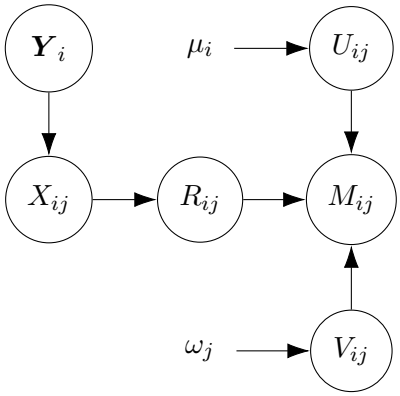


Figure 2.5: DAG of random variable dependencies in the Bayesian-BM/OR model from Kim and Choi [2014]. $\boldsymbol{M}$ is the mask matrix, $\boldsymbol{X}$ the rating matrix and $\boldsymbol{Y}$ the indicator matrix of the user clusters. Missingness is of type MNAR is the mask matrix depends indirectly on the rating matrix through the random variable R_{ij}.

inferring the low rank matrices or while predicting missing values.

Response Aware Model-Based: Ling et al. [2012] introduce a MNAR model in a simple probabilistic matrix factorization framework. The observation probability is a linear combination of the value of the entry of X_{ij} and of $\boldsymbol{Y}$ and $\boldsymbol{Z}$, the two Gaussian distributed low rank matrices that are also used to generate the rating matrix $\boldsymbol{X}$:

$$X_{ij} \overset{\text{iid}}{\sim} \mathcal{N}\left(\boldsymbol{Y}_i{}^T \boldsymbol{Z}_j, \sigma^2\right)$$
$$p(\, M_{ij} = 1 | X_{ij} = v \,) = \text{expit}\left(\boldsymbol{Y}_i{}^T \boldsymbol{\lambda_Y} + \boldsymbol{Z}_j{}^T \boldsymbol{\lambda_Z} + \lambda_v\right)$$

with $\boldsymbol{\lambda_Y} \in \mathbb{R}^{k_1}$ and $\boldsymbol{\lambda_Z} \in \mathbb{R}^{k_1}$, the parameters of the missingness model. The training complexity depends on the size of the rating data matrix $\boldsymbol{X}$.

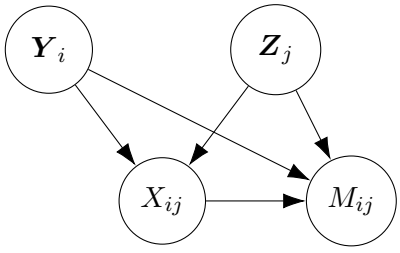


Figure 2.6: DAG of main random variable dependencies in the Response Aware Model-Based from Ling et al. [2012]. The matrix $\boldsymbol{X}$ of continuous ratings is generated by the two Gaussian distributed low rank matrices $\boldsymbol{Y}$ and $\boldsymbol{Z}$. The mask matrix $\boldsymbol{M}$ directly depends on the values of the ratings $\boldsymbol{X}$ as well as on the two low rank matrices.

Double probabilistic MF: Hernández-Lobato et al. [2014] use a double probabilistic matrix factorization (MF) model; one is for the complete data and one for the missing data, where users and items propensities are both modeled with low rank matrices. The dependencies of the main random variables of the model are presented in Figure 2.7.

The complete data model uses an ordinal likelihood for rating data (rather than the usual Gaussian likelihood), a variable noise ε_{ij}^X depending on each user and item and hierarchical priors to increase robustness to selection of hyper-parameter values. The discrete value of the rating X_{ij} is obtained from the interval defined by $\zeta_{j,0} = -\infty < \zeta_{j,1} < ... < \zeta_{j,K-1} < \zeta_K = \infty$ in which X_{ij} falls:

$$X_{ij} = \boldsymbol{Y}_i^{X^T} \boldsymbol{Z}_j^X + \varepsilon_{ij}^X \quad ,$$

with $\boldsymbol{Y}_i^X$, $\boldsymbol{Z}_j^X$ and ε_{ij}^X normally distributed.

The missing data model generates the binary mask matrix $\boldsymbol{M}$ using $\boldsymbol{Y}_i^M$ and $\boldsymbol{Z}_j^M$, two normally distributed low rank matrices, a variable noise ε_{ij}^M and the rating matrix $\boldsymbol{X}$ generated by the complete data model. The observation probability depends on the X_{ij} value with different dependence strengths across rows and columns, making this model very flexible.

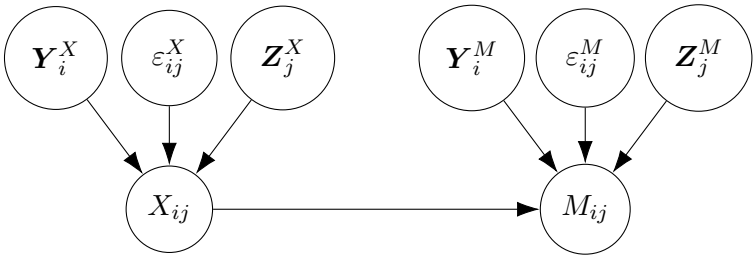


Figure 2.7: DAG of main random variable dependencies in the probabilistic matrix factorization with non random missing data from Hernández-Lobato et al. [2014]. The matrix $\boldsymbol{X}$ of continuous ratings is generated by the two normally distributed low rank matrices $\boldsymbol{Y}$ and $\boldsymbol{Z}$ and by the Gaussian noise $\boldsymbol{\varepsilon}$. The mask matrix $\boldsymbol{M}$ directly depends on the values of the ratings $\boldsymbol{X}$.

Weighted matrix factorization: Steck [2010] derives a weighted MF model and optimizes the parameters based on a metric that is robust to MNAR data. This work relies on two assumptions:

1. the relevant rating values are missing at random in the observed data. A rating is considered as relevant if it has the highest value (for example 5 on a five-star rating scale).

2. concerning the other rating values, an arbitrary missingness mechanism is allowed, as long as these values are missing with a higher probability than the relevant rating values do.

Under the above two assumptions, the author shows that the defined ranking metric remains unbiased even if computed from the observed MNAR data. The ranking metric proposed is computationally tractable for testing, but it remains too expensive to be used for training. For this reason, the author resort to an appropriate surrogate measure in which the key idea is to consider the ranking of all items whether their ratings are observed or missing in the data. This is done by the objective function called *AllRank-Regression*, that uses all values of $\boldsymbol{X}$ whether observed or not. A rating is imputed for all missing values and the ratings are predicted by a basic low-rank matrix-factorization model:

$$\begin{aligned} \hat{X}_{ij} &= {\boldsymbol{Y}_i}^T \boldsymbol{Z}_j + \mu \\ loss &= \sum_{ij} W_{ij} \cdot \left(\left(X_{ij} - \hat{X}_{ij} \right)^2 + \lambda \left(\sum_k Y_{i,k}^2 + Z_{j,k}^2 \right) \right) \\ \text{with } W_{ij} &= \begin{cases} w(X_{ij}) & \text{if} \quad X_{ij} \text{ observed} \\ w_m & \text{else} \end{cases} , \end{aligned}$$

with μ a fixed offset parameter chosen to be equal to the imputed missing rating. The weights W_{ij} are used to balance the proportion between observed and non observed data. The weight-values that are fixed tuning parameters, are different for each observed rating value X_{ij} but they are common for all missing values. The loss function is optimized with an alternating least squared strategy.

2.2 Learning from missing data with the binary Latent Block Model

Co-clustering simultaneously groups the rows and the columns of a data matrix. Co-clustering has found applications in many areas such as genomic analysis [Pontes et al., 2015, Kluger et al., 2003], text analysis [Dhillon et al., 2003, Selosse et al., 2020b], collaborative filtering [George and Merugu, 2005, Shan and Banerjee, 2008], or political analysis [Latouche et al., 2011, Wyse and Friel, 2012]. Co-clustering methods can be divided into categories such as, but not limited to, spectral methods [Dhillon, 2001, Kluger et al., 2003], mutual information methods [Dhillon et al., 2003], modularity based methods [Labiod and Nadif, 2011], non negative matrix tri-factorization [Ding et al., 2006] or model-based methods. Among the model-based methods, the Latent Block Model [Govaert and Nadif, 2008, Nadif and Govaert, 2010, Lomet, 2012, Keribin et al., 2015] relies on mixtures, assuming that the observations are generated from finite mixture components in rows and columns. Most standard methods of clustering or co-clustering presuppose complete information and cannot be applied with missing data, or may provide misleading conclusions when missingness is informative.

We aim at clustering the rows and columns of a binary data matrix whose entries are missing not at random. Equivalently, we consider the clustering of the vertices of a bipartite graph whose edges are missing not at random. For this purpose, we introduce a co-clustering model that combines a MNAR missingness model with the Latent Block Model (LBM).

In Section 2.2.1, we introduce our model, a LBM extended to a MNAR missingness process, and propose, in Section 2.2.2, a variational EM algorithm to infer its parameters. We also introduce, in Section 2.2.3, an Integrated Completed Likelihood (ICL) criterion to tackle model selection. We then conduct experiments on synthetic datasets in Section 2.2.4 to show that the overall approach is relevant to co-cluster and predict MNAR data. Finally, an analysis of the voting records of the lower house of the French Parliament and on the Yahoo! R3 dataset are presented in Section 2.2.5. The source code and the dataset of the voting records are provided for reproducibility purposes at `https://github.com/gfrisch/LBM-MNAR`.

2.2.1 Extension of the binary LBM to informative missingness

The standard Latent Block Model (Section 1.2.2) does not accommodate missing observations, that is, the data matrix $\boldsymbol{X}$ is fully observed. This section introduces our missingness model, which will be coupled to the LBM, thereby enabling to process missing data.

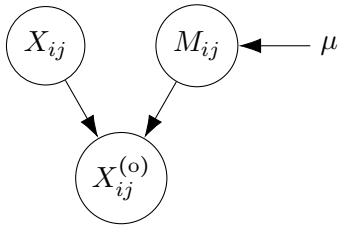


Figure 2.8: Graphical representation of the MCAR model. The partially observed entry $X_{ij}^{(o)}$ is generated by the corresponding entries of (i) the full matrix X_{ij} and (ii) the binary mask M_{ij}. The binary mask $\boldsymbol{M}$ does not depend on $\boldsymbol{X}$ and is defined here from a single global effect parameter μ.

We start by introducing some notation: from now on, $\boldsymbol{X}^{(o)}$ will denote the "partially observed" data matrix, with missing entries, whereas $\boldsymbol{X}$ denotes the "full" (unobserved) data matrix, without missing entries. The partially observed matrix $\boldsymbol{X}^{(o)}$ is identical to the full matrix $\boldsymbol{X}$ except for the missing entries; $\boldsymbol{X}^{(o)}$ takes its values in $\{0, 1, \text{NA}\}$, where NA denotes a missing value. It will be convenient to introduce a binary mask matrix $\boldsymbol{M}$ that indicates the *non-missing* entries of $\boldsymbol{X}^{(o)}$: if $M_{ij} = 0$, then $X_{ij}^{(o)} = \text{NA}$.

2.2.1.1 Models of Missingness

The three main types of missingness are Missing Completely At Random (MCAR), Missing At Random (MAR), and Missing Not At Random (MNAR). We propose here a model for each missingness type. Instead of directly modeling the probability of being missing, we will model a real variable P_{ij} that defines the log-odds of this probability. This log-odds will be called here the "propensity" to be observed:

$$\forall i, j \quad P_{ij} = \log\left(\frac{p(M_{ij} = 1)}{p(M_{ij} = 0)}\right) .$$

Missing Completely At Random (MCAR) Missingness does not depend on data, whether observed or not. A simple model of missingness is obtained by assuming that every entry of $\boldsymbol{X}^{(o)}$ has the same propensity of being missing. This is modeled by a single propensity parameter μ. The graphical representation of this model is shown in Figure 2.8.

Missing At Random (MAR) Missingness depends on the observed data $\boldsymbol{X}_{ij}^{(o)} \in \{0, 1, \text{NA}\}$ forall (i, j), but not on the unobserved data X_{ij} for (i, j) where $\boldsymbol{X}_{ij}^{(o)} = \text{NA}$. The previous missingness model can be enlarged by allowing the propensity of missingness to depend on the row and column indexes. To do so, we can introduce a latent variable for every row, denoted $\boldsymbol{A}$, and another

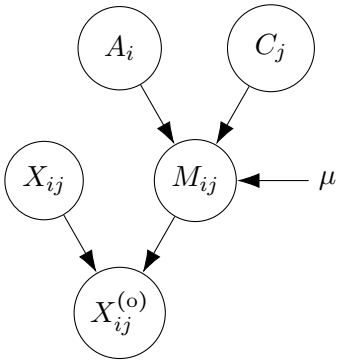


Figure 2.9: Graphical representation of the MAR model. The partially observed entry $X_{ij}^{(o)}$ is generated by the corresponding entries of (i) the full matrix X_{ij} and (ii) the binary mask M_{ij}. The binary mask $\boldsymbol{M}$ does not depend on $\boldsymbol{X}$ and is and is defined by a global effect parameter μ and two latent variables $\boldsymbol{A}$ and $\boldsymbol{C}$ that enable deviations from μ.

one for every column, denoted $\boldsymbol{C}$. For the sake of simplicity, all latent variables A_i and C_j are assumed independent. They allow deviations from the global propensity μ. The graphical representation of this model is shown in Figure 2.9.

Missing Not At Random (MNAR) Missingness here depends on unobserved data: the probability of observing the entries of the matrix depends on their values, whether observed or not. We equip the previous model with two additional latent variables to adapt the propensity of each entry of the data matrix to the unobserved data, that is, to X_{ij}. These new row and column latent variables, $\boldsymbol{B}$ and $\boldsymbol{D}$, adjust the propensity of missingness according to the actual value of X_{ij}. The graphical representation of this model is shown in Figure 2.10.

We model the latent variables $\boldsymbol{A}$, $\boldsymbol{B}$, $\boldsymbol{C}$, and $\boldsymbol{D}$ with Gaussian distributions centered at zero with free variances σ_A^2, σ_B^2, σ_C^2, and σ_D^2, respectively:

$$\begin{cases} \forall i, & A_i \overset{\text{iid}}{\sim} \mathcal{N}\big(0, \sigma_A^2\big), \quad B_i \overset{\text{iid}}{\sim} \mathcal{N}\big(0, \sigma_B^2\big) \\ \forall j, & C_j \overset{\text{iid}}{\sim} \mathcal{N}\big(0, \sigma_C^2\big), \quad D_j \overset{\text{iid}}{\sim} \mathcal{N}\big(0, \sigma_D^2\big) \end{cases} .$$

The global parameter μ and the latent variables define the propensity of missingness, that is, the log-odds of being missing as follows:

$$\forall i, j \quad P_{ij} = \begin{cases} \mu + A_i + B_i + C_j + D_j & \text{if} \quad X_{ij} = 1 \\ \mu + A_i - B_i + C_j - D_j & \text{if} \quad X_{ij} = 0 \end{cases} . \tag{2.1}$$

Then, given this propensity, every element M_{ij} of the mask matrix is independent and follows a Bernoulli distribution:

$$\forall i, j \quad M_{ij} | A_i, B_i, C_j, D_j, X_{ij} \overset{\text{ind}}{\sim} \mathcal{B}(\text{expit}\,(P_{ij})) \ , \tag{2.2}$$

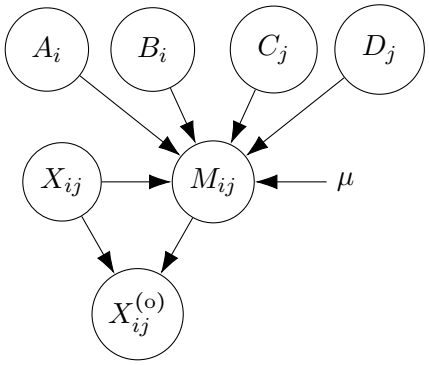


Figure 2.10: Graphical representation of the MNAR model. The partially observed entry $X_{ij}^{(o)}$ is generated by the corresponding entries of (i) the full matrix X_{ij} and (ii) the binary mask M_{ij}. The binary mask $\boldsymbol{M}$ depends on $\boldsymbol{X}$ and is defined by a global effect parameter μ, two latent variables $\boldsymbol{A}$ and $\boldsymbol{C}$ that enable deviations from μ, and two latent variables $\boldsymbol{B}$ and $\boldsymbol{D}$, which drive the deviations from the MAR model.

with $\text{expit}(x) = 1/(1+\exp(-x))$.

Note that, if we omit the latent variables B_i and D_j, the missingness model follows the MAR assumption since P_{ij}, and thus M_{ij}, is then independent of X_{ij}. If we also omit the latent variables A_i and C_j, the missingness model follows the MCAR assumption.

This model of missingness can be used for several applications. One of these, collaborative filtering, uses the history of user ratings to build a recommendation system. For this application, an MCAR modeling means that the probability of observing a rating for a particular item does not depend on the user nor the item; an MAR modeling means that missingness can depend on the user or the item; for example, some people give their opinion more often than others. The MAR simplifying assumption is often used in collaborative filtering. However, Marlin et al. [2007] show that there is often a dependency between the rating frequency and the underlying preference level, lending support to the hypothesis that ratings are generated by a MNAR process, where missingness depends on the actual rating that would be given. Some people give their opinion more often when they are satisfied and other ones when they are dissatisfied. Most collaborative filtering methods do not have a principled method for extracting information from missing data, which can lead to strong biases in estimations that may in turn drastically affect predictions [Hernández-Lobato et al., 2014]. Our missingness model allows one to account for the users' propensity to give their opinion, and for the items' propensity to be rated, that is, their notoriety. These propensities could also reflect exogenous factors such as price; for example, more expensive items could be evaluated more often.

2.2.1.2 LBM with MNAR data

We extend the standard *LBM* using the previous modeling to MNAR data.

Given the full matrix $\boldsymbol{X}$ and the mask matrix $\boldsymbol{M}$, all the elements of the observed matrix $\boldsymbol{X}^{(\mathrm{o})}$ are generated as follows:

$$X_{ij}^{(\mathrm{o})} = \begin{cases} X_{ij} & \text{if} \quad M_{ij} = 1 \\ \mathrm{NA} & \text{if} \quad M_{ij} = 0 \end{cases} . \tag{2.3}$$

Figure 2.11 summarizes the LBM extented to MNAR data.

$\boldsymbol{X}^{(\mathrm{o})}$ taking its values in $(0, 1, \mathrm{NA})$, the same model can be rewritten with a categorial distribution (that is, a multinomial distribution with one trial) using directly the latent variables of both models:

$$\begin{aligned} &\forall i,\ \boldsymbol{Y}_i \overset{\text{iid}}{\sim} \mathcal{M}(1; \boldsymbol{\alpha}) && \forall j,\ \boldsymbol{Z}_j \overset{\text{iid}}{\sim} \mathcal{M}(1; \boldsymbol{\beta}) \\ &\forall i,\ A_i \overset{\text{iid}}{\sim} \mathcal{N}\left(0, \sigma_A^2\right) && \forall j,\ C_j \overset{\text{iid}}{\sim} \mathcal{N}\left(0, \sigma_C^2\right) \\ &\forall i,\ B_i \overset{\text{iid}}{\sim} \mathcal{N}\left(0, \sigma_B^2\right) && \forall j,\ D_j \overset{\text{iid}}{\sim} \mathcal{N}\left(0, \sigma_D^2\right) \end{aligned} \tag{2.4}$$

$$X_{ij}^{(\mathrm{o})} \Big| Y_{iq} = 1, Z_{jl} = 1, A_i, B_i, C_j, D_j \overset{\text{ind}}{\sim} \mathrm{cat}\left(\begin{bmatrix} 0 \\ 1 \\ \mathrm{NA} \end{bmatrix}, \begin{bmatrix} p_0 \\ p_1 \\ 1 - p_0 - p_1 \end{bmatrix}\right)$$

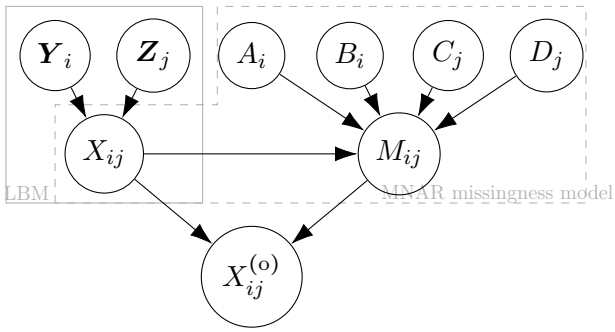
with

$$p_0 = \left(1 - \pi_{ql}\right) \operatorname{expit}\left(\mu + A_i - B_i + C_j - D_j\right) \tag{2.5}$$

$$p_1 = \pi_{ql} \operatorname{expit}\left(\mu + A_i + B_i + C_j + D_j\right) . \tag{2.6}$$

The parameters of the LBM with MNAR data are $\theta = (\boldsymbol{\alpha}, \boldsymbol{\beta}, \boldsymbol{\pi}, \mu, \sigma_A^2, \sigma_B^2, \sigma_C^2, \sigma_D^2)$ with size $k_1 + k_2 + k_1 \times k_2 + 5$.

Keribin et al. [2015] proved under mild assumptions that for a probability distribution $p_\theta(X)$ of a binary Latent Block Model, there exists a unique set of parameters $\theta = (\boldsymbol{\pi}, \boldsymbol{\alpha}, \boldsymbol{\beta})$ up to a permutation of row and column labels. Our missingness model belongs to the linear mixed effects model family whose identifiability depends on the covariance structures. Wang [2013] derives conditions of identifiability for the covariance parameters in a linear mixed effect model and study some commonly used covariance structures. We did not reach a proof of identifiability for the extended LBM with MNAR missingness model. However, the stability of the inference shown by experiments on synthetic data (Section 2.2.4 and Annex 2.A.3.2) suggests that there may be conditions that there may be conditions for which the joint model should be identifiable.



Latent Block Model

$$\boldsymbol{Y}_i \overset{\text{iid}}{\sim} \mathcal{M}(1;\boldsymbol{\alpha}), \qquad \boldsymbol{\alpha} \in \mathbf{S}_{k_1-1}$$

$$\boldsymbol{Z}_j \overset{\text{iid}}{\sim} \mathcal{M}(1;\boldsymbol{\beta}), \qquad \boldsymbol{\beta} \in \mathbf{S}_{k_2-1}$$

$$\big(X_{ij} \big| Y_{iq} = 1, Z_{jl} = 1\big) \overset{\text{ind}}{\sim} \mathcal{B}\big(\pi_{ql}\big), \qquad \pi_{ql} \in (0,1)$$

MNAR model

$$A_i \overset{\text{iid}}{\sim} \mathcal{N}\big(0, \sigma_A^2\big), \qquad \sigma_A^2 \in \mathbb{R}_+^*$$

$$B_i \overset{\text{iid}}{\sim} \mathcal{N}\big(0, \sigma_B^2\big), \qquad \sigma_B^2 \in \mathbb{R}_+^*$$

$$C_j \overset{\text{iid}}{\sim} \mathcal{N}\big(0, \sigma_C^2\big), \qquad \sigma_C^2 \in \mathbb{R}_+^*$$

$$D_j \overset{\text{iid}}{\sim} \mathcal{N}\big(0, \sigma_D^2\big), \qquad \sigma_D^2 \in \mathbb{R}_+^*$$

$$\big(M_{ij} | A_i, B_i, C_j, D_j, X_{ij} = 1\big) \overset{\text{ind}}{\sim} \mathcal{B}(\text{expit}\,(\mu + A_i + B_i + C_j + D_j))$$

$$\big(M_{ij} | A_i, B_i, C_j, D_j, X_{ij} = 0\big) \overset{\text{ind}}{\sim} \mathcal{B}(\text{expit}\,(\mu + A_i - B_i + C_j - D_j))$$

Observations are generated according to:

$$X_{ij}^{(o)} = \begin{cases} X_{ij} & \text{if} \quad M_{ij} = 1 \\ \text{NA} & \text{if} \quad M_{ij} = 0 \end{cases}$$

Figure 2.11: Graphical view and summary of the Latent Block Model extended to MNAR missingness process. The observed data $X_{ij}^{(o)}$ is generated by the necessary information carried by the class and propensity of row i and by the class and propensity of the column j.

2.2.2 Inference in the extented LBM

The dependency between the full data matrix $\boldsymbol{X}$ and the mask matrix $\boldsymbol{M}$ requires a joint inference of the LBM with the MNAR model. As the standard maximum likelihood approach cannot be applied directly, we adopt a strategy based on a variational EM.

During inference, we use the reformulation of Equation (2.4). We can split our random variables into two sets: the set of unobserved latent variables and the set of observed variables consisting of $\boldsymbol{X}^{(\mathrm{o})}$ only. An observation of $\boldsymbol{X}^{(\mathrm{o})}$ only is called the incomplete data, and an observation of $\boldsymbol{X}^{(\mathrm{o})}$ together with the latent variables $\boldsymbol{A}$, $\boldsymbol{B}$, $\boldsymbol{C}$, $\boldsymbol{D}$, $\boldsymbol{Y}$ and $\boldsymbol{Z}$ is called the complete data. Given the incomplete data, our objective is to infer the model parameters θ via maximum likelihood $\widehat{\theta} = \arg\max_\theta p(\boldsymbol{X}^{(\mathrm{o})};\theta)$. This likelihood of the incomplete data set should be obtained by marginalization over all latent variables $p(\boldsymbol{X}^{(\mathrm{o})};\theta) = \sum_{\boldsymbol{Y}\boldsymbol{Z}} \int_{\boldsymbol{A}\,\boldsymbol{B}\,\boldsymbol{C}\,\boldsymbol{D}} p(\boldsymbol{X}^{(\mathrm{o})}, \boldsymbol{Y}, \boldsymbol{Z}, \boldsymbol{A}, \boldsymbol{B}, \boldsymbol{C}, \boldsymbol{D}; \theta)$, which is rapidly untractable as it involves an exponentially growing sum over all possible values of the latent variables.

We resort to the Expectation Maximization (EM) algorithm to maximize $p(\boldsymbol{X}^{(\mathrm{o})};\theta)$ without explicitly calculating it. The EM algorithm iteratively applies the two following steps:

E-step Expectation step: from the current estimate $\theta^{(t)}$ of θ, compute the criterion $\mathcal{Q}(\theta|\theta^{(t)})$ defined as the expectation of the complete log-likelihood, conditionally on the observations $\boldsymbol{X}^{(\mathrm{o})}$:

$$\mathcal{Q}(\theta|\theta^{(t)}) = \mathbb{E}_{\boldsymbol{Y},\boldsymbol{Z},\boldsymbol{A},\boldsymbol{B},\boldsymbol{C},\boldsymbol{D}|\boldsymbol{X}^{(\mathrm{o})},\theta^{(t)}}\left[\log p\Big(\boldsymbol{X}^{(\mathrm{o})}, \boldsymbol{Y}, \boldsymbol{Z}, \boldsymbol{A}, \boldsymbol{B}, \boldsymbol{C}, \boldsymbol{D}; \theta\Big)\right]$$

M-step Maximization step: find the parameters that maximize $\mathcal{Q}(\theta|\theta^{(t)})$.

$$\theta^{(t+1)} = \arg\max_\theta \; \mathcal{Q}(\theta|\theta^{(t)})$$

The computation of the complete log-likelihood at the E-step requires the posterior distribution of the latent variables $p(\boldsymbol{Y}, \boldsymbol{Z}, \boldsymbol{A}, \boldsymbol{B}, \boldsymbol{C}, \boldsymbol{D}|\boldsymbol{X}^{(\mathrm{o})})$ which is intractable, because the search space of the latent variables is combinatorially too large. This problem is well known in the context of co-clustering; for the Latent Block Model, Celeux and Diebolt [1985], Keribin et al. [2015] propose a stochastic E-step with Monte Carlo sampling, but this strategy is not suited to large-scale problems. We follow the original strategy proposed by Govaert and Nadif [2008], which relies on a variational formulation of the problem, since it is more efficient in high dimension.

2.2.2.1 Variational EM

The variational EM (VEM) [Jordan et al., 1999, Jaakkola, 2000] introduces $q(\cdot)$, a parametric inference distribution defined over the latent variables $\boldsymbol{Y}$, $\boldsymbol{Z}$, $\boldsymbol{A}$, $\boldsymbol{B}$, $\boldsymbol{C}$, $\boldsymbol{D}$ and optimizes the following lower bound on the log-likelihood of the incomplete data:

$$\mathcal{J}(q,\theta) = \log p\Big(\boldsymbol{X}^{(\mathrm{o})};\theta\Big) - KL\Big(q(\cdot) \parallel p(\cdot|\boldsymbol{X}^{(\mathrm{o})};\theta)\Big) \ , \tag{2.7}$$

where KL stands for the Kullback-Leibler divergence and $q(\cdot)$ denotes the variational distribution over the latent variables $\boldsymbol{Y}$, $\boldsymbol{Z}$, $\boldsymbol{A}$, $\boldsymbol{B}$, $\boldsymbol{C}$ and $\boldsymbol{D}$. It can be shown that $\mathcal{J}(q,\theta)$ is a concave function of the variational distribution q and that its maximum is reached for $q(\cdot) = p(\cdot|\boldsymbol{X}^{(\mathrm{o})};\theta)$. Thus, maximizing the criterion $\mathcal{J}$ is equivalent to minimizing the discrepancy between $q(\cdot)$ and $p(\cdot|\boldsymbol{X}^{(\mathrm{o})};\theta)$, as measured by the KL-divergence, and is also equivalent to maximizing the likelihood. The minimization of this KL-divergence requires one to explore the whole space of latent distributions; the difficulty of the problem is equivalent, in terms of complexity, to the initial problem.

The criterion $\mathcal{J}(q,\theta)$ can also be expressed as the sum of a negative "energy" and the entropy of q hence its name "negative variational free energy" in analogy with the thermodynamic free energy:

$$\mathcal{J}(q,\theta) = \mathcal{H}(q) + \mathbb{E}_q\Big[\log p\Big(\boldsymbol{X}^{(\mathrm{o})},\boldsymbol{Y},\boldsymbol{Z},\boldsymbol{A},\boldsymbol{B},\boldsymbol{C},\boldsymbol{D};\theta\Big)\Big] \ , \tag{2.8}$$

where $\mathcal{H}(q)$ is the entropy of the variational distribution and $\mathbb{E}_q$ is the expectation with respect to the variational distribution. The criteria $\mathcal{J}$ can become tractable if an exploration of a subspace, noted q_γ, of the latent distributions is made. However, this solution comes with the cost that the maximum found will be a lower bound of the initial criterion:

$$\mathcal{J}(q,\theta) \geq \mathcal{J}(q_\gamma,\theta) \tag{2.9}$$

$\mathcal{J}(q_\gamma,\theta)$ is also known as the "Evidence Lower BOund" (ELBO) emphasizing the lower bound property on the evidence of the data.

A wise choice of the restriction on the variational distribution leads to a feasible computation of the criterion. We choose to consider the following

posterior shapes on the latent variables:

$$
\begin{aligned}
\forall i \qquad & \boldsymbol{Y}_i|\boldsymbol{X}^{(\mathrm{o})} \underset{q_\gamma}{\sim} \mathcal{M}\Big(1;\tau_i^{(Y)}\Big) \\
\forall j \qquad & \boldsymbol{Z}_j|\boldsymbol{X}^{(\mathrm{o})} \underset{q_\gamma}{\sim} \mathcal{M}\Big(1;\tau_j^{(Z)}\Big) \\
\forall i \qquad & A_i|\boldsymbol{X}^{(\mathrm{o})} \underset{q_\gamma}{\sim} \mathcal{N}\Big(\nu_i^{(A)},\rho_i^{(A)}\Big) \\
\forall i \qquad & B_i|\boldsymbol{X}^{(\mathrm{o})} \underset{q_\gamma}{\sim} \mathcal{N}\Big(\nu_i^{(B)},\rho_i^{(B)}\Big) \\
\forall j \qquad & C_j|\boldsymbol{X}^{(\mathrm{o})} \underset{q_\gamma}{\sim} \mathcal{N}\Big(\nu_j^{(C)},\rho_j^{(C)}\Big) \\
\forall j \qquad & D_j|\boldsymbol{X}^{(\mathrm{o})} \underset{q_\gamma}{\sim} \mathcal{N}\Big(\nu_j^{(D)},\rho_j^{(D)}\Big) \ .
\end{aligned}
$$

We also impose the conditional independence of the latent variables to get a feasible computation of the entropy and of the negative "energy" (Equation 2.8) under q_γ. This conditional independence is widely known as the "mean field approximation" [Parisi, 1988]. We finally get the following fully factorized shape:

$$
\begin{aligned}
q_\gamma = & \prod_{i=1}^{n_1} \mathcal{M}\Big(1;\tau_i^{(Y)}\Big) \times \prod_{j=1}^{n_2} \mathcal{M}\Big(1;\tau_j^{(Z)}\Big) \\
& \times \prod_{i=1}^{n_1} \mathcal{N}\Big(\nu_i^{(A)},\rho_i^{(A)}\Big) \times \prod_{i=1}^{n_1} \mathcal{N}\Big(\nu_i^{(B)},\rho_i^{(B)}\Big) \\
& \times \prod_{j=1}^{n_2} \mathcal{N}\Big(\nu_j^{(C)},\rho_j^{(C)}\Big) \times \prod_{j=1}^{n_2} \mathcal{N}\Big(\nu_j^{(D)},\rho_j^{(D)}\Big) \ ,
\end{aligned}
$$

where $\gamma = (\boldsymbol{\tau}^{(Y)},\boldsymbol{\tau}^{(Z)},\boldsymbol{\nu}^{(A)},\boldsymbol{\rho}^{(A)},\boldsymbol{\nu}^{(B)},\boldsymbol{\rho}^{(B)},\boldsymbol{\nu}^{(C)},\boldsymbol{\rho}^{(C)},\boldsymbol{\nu}^{(D)},\boldsymbol{\rho}^{(D)})$ denotes the set of hyper-parameters' concatenation of the restricted variational distribution q_γ.

The new criterion $\mathcal{J}(\gamma,\theta)$ that we want to optimize from now on is:

$$
\mathcal{J}(\gamma,\theta) = \mathcal{H}(q_\gamma) + \mathbb{E}_{q_\gamma}\Big[\log p\Big(\boldsymbol{X}^{(\mathrm{o})},\boldsymbol{Y},\boldsymbol{Z},\boldsymbol{A},\boldsymbol{B},\boldsymbol{C},\boldsymbol{D};\theta\Big)\Big] \tag{2.10}
$$

and the initial estimates of the model parameters $\widehat{\theta}$ are inferred as:

$$
\widehat{\theta} = \arg\max_{\theta}\Big(\max_{\gamma}\mathcal{J}(\gamma,\theta)\Big) \ . \tag{2.11}
$$

This double maximization is realized with an iterative strategy and can be seen as an extension of the EM algorithm. The two steps are described in Algorithm 2.

Algorithm 2: Variational Expectation Maximization algorithm.

Input: observed data $\boldsymbol{X}^{(o)}$, k_1 and k_2 number of row groups and column groups ;

- Initialize $\gamma^{(0)}$ and $\theta^{(0)}$;
- **while** *not convergence of criterion* $\mathcal{J}$ **do**

VE-step: find the variational parameters $\gamma^{(t+1)}$ that optimize $\mathcal{J}\left(\gamma, \theta^{(t)}\right)$

$$\gamma^{(t+1)} = \arg\max_{\gamma} \mathcal{J}\left(\gamma, \theta^{(t)}\right)$$

M-step: find the model parameters $\theta^{(t+1)}$ that optimize $\mathcal{J}\left(\gamma^{(t)}, \theta\right)$:

$$\theta^{(t+1)} = \arg\max_{\theta} \mathcal{J}\left(\gamma^{(t)}, \theta\right)$$

end

Result: θ and γ: model and variational parameters

2.2.2.2 Computation of the variational criterion

The restriction on the space of the variational distribution simplifies the computation of $\mathcal{H}(q_\gamma)$ as entropy is additive across independent variables:

$$\begin{aligned}\mathcal{H}(q_\gamma) = &- \sum_{iq} \tau_{iq}^{(Y)} \log \tau_{iq}^{(Y)} - \sum_{jl} \tau_{jl}^{(Z)} \log \tau_{jl}^{(Z)} \\ &+ \frac{1}{2} \sum_{i} \log\left(2\pi e \rho_i^{(A)}\right) + \frac{1}{2} \sum_{i} \log\left(2\pi e \rho_i^{(B)}\right) \\ &+ \frac{1}{2} \sum_{j} \log\left(2\pi e \rho_j^{(C)}\right) + \frac{1}{2} \sum_{j} \log\left(2\pi e \rho_j^{(D)}\right) \ .\end{aligned}$$

The independence of latent variables allows one to rewrite the expectation of the complete log-likelihood as:

$$\begin{aligned}\mathbb{E}_{q_\gamma}\left[\log p\left(\boldsymbol{X}^{(o)}, \boldsymbol{Y}, \boldsymbol{Z}, \boldsymbol{A}, \boldsymbol{B}, \boldsymbol{C}, \boldsymbol{D}\right)\right] = &\ \mathbb{E}_{q_\gamma}[\log p(\boldsymbol{Y})] \\ &+ \mathbb{E}_{q_\gamma}[\log p(\boldsymbol{Z})] + \mathbb{E}_{q_\gamma}[\log p(\boldsymbol{A})] + \mathbb{E}_{q_\gamma}[\log p(\boldsymbol{B})] \\ &+ \mathbb{E}_{q_\gamma}[\log p(\boldsymbol{C})] + \mathbb{E}_{q_\gamma}[\log p(\boldsymbol{D})] \\ &+ \mathbb{E}_{q_\gamma}\left[\log p\left(\boldsymbol{X}^{(o)} \middle| \boldsymbol{Y}, \boldsymbol{Z}, \boldsymbol{A}, \boldsymbol{B}, \boldsymbol{C}, \boldsymbol{D}\right)\right] \ .\end{aligned} \tag{2.12}$$

Despite the variational approximation, the expectation of the complete log-likelihood (2.12) cannot be exactly computed as its last term involves an

expectation under q_γ of nonlinear functions:

$$\begin{aligned}\mathbb{E}_{q_\gamma}\Big[\log p\Big(\boldsymbol{X}^{(\mathrm{o})}\Big|\boldsymbol{Y},\boldsymbol{Z},\boldsymbol{A},\boldsymbol{B},\boldsymbol{C},\boldsymbol{D}\Big)\Big] &= \sum_{ijql:X_{ij}^{(\mathrm{o})}=0} \tau_{iq}^{(Y)}\tau_{jl}^{(Z)}\mathbb{E}_{q_\gamma}[\log(p_0)] \\ &+ \sum_{ijql:X_{ij}^{(\mathrm{o})}=1} \tau_{iq}^{(Y)}\tau_{jl}^{(Z)}\mathbb{E}_{q_\gamma}[\log(p_1)] \\ &+ \sum_{ijql:X_{ij}^{(\mathrm{o})}=\mathrm{NA}} \tau_{iq}^{(Y)}\tau_{jl}^{(Z)}\mathbb{E}_{q_\gamma}[\log(1-p_0-p_1)]\ , \end{aligned} \tag{2.13}$$

with p_0 and p_1 defined in Equations (2.5)–(2.6).

These expectations can be approximated by Taylor expansions assuming a small variance of the Gaussian variational variables. This method has similarities with the delta method [Wasserman, 2004, p. 79] with normal asymptotics with variances tending to zero. Using a first order Taylor expansion would lead to a criterion without maximum, so we use a second order Taylor expansion. The full expression of the criterion is given in Appendix 2.A.

2.2.2.3 Maximization of the variational criterion

The VEM Algorithm 2 alternates between a maximization with respect to the variational parameters γ and a maximization w.r.t the model parameters θ. For our model, there is no explicit solution for the two maximizations of the criterion $\mathcal{J}(\gamma,\theta)$, which are carried out by the Limited-memory Broyden–Fletcher–Goldfarb–Shanno (L-BFGS) algorithm. We used automatic differentiation to compute the gradients needed for L-BFGS and for the Taylor series used in the variational criterion. We chose the Autograd library from HIPS and the submodule Autograd from PyTorch [Paszke et al., 2019]. These libraries rely on a reverse accumulation computational graph to compute exact gradients. Their high efficiency, even with large graphs, thanks to GPU acceleration, makes them particularly well adapted for the VEM algorithm.

2.2.2.4 Initialization

VEM does not ensure convergence towards a global optimum. The EM-like algorithms are known to be sensitive to the initialization, particularly when applied to models with discrete latent space, and may get stuck in unsatisfactory local maxima [Biernacki et al., 2003, Baudry and Celeux, 2015].

A simple solution consists in training for a few iterations from several random initializations, and pursuing optimization with the solution with highest value of the variational criterion [see, e.g., small EM for mixtures Baudry and Celeux, 2015]. This exploration strategy spends a great deal of computing resources to bring out only a few good estimates. Another solution is to rely on simpler clustering methods, such as k-means or spectral clustering, to initialize the algorithm [Shireman et al., 2015].

The parameters of the Stochastic Block Model, a close relative of the Latent Block Model for graphs, can be consistently identified by spectral clustering [Rohe et al., 2011]. Following this idea, we use a double spectral clustering [with absolute eigenvalues of the Laplacian as in Rohe et al., 2011] on rows and columns on the similarity matrices $\boldsymbol{X}\boldsymbol{X}^T$ and $\boldsymbol{X}^T\boldsymbol{X}$, to initialize our algorithm. Although this method is not designed for MNAR data, it can be expected to provide a satisfying initialization of the Latent Block Model if the missingness is not predominant. The parameters of our missingness model cannot be initialized with this procedure; they are randomly initialized. The overall initialization procedure is described in Appendix 2.A.

2.2.2.5 Improving the computational efficiency

The proposed inference cannot scale to collaborative filtering data for computational reasons. In case of collaborative filtering data, the data is parsimonious, that is most of the data is missing. In such settings, we can improve the computational expenses of our inference using approximations. These additional approximations makes the training procedure of our model possible with standard collaborative filtering datasets. The procedure is further detailed in Chapter 4.

With our proposed approximations, detailed in Appendix 2.B, the training procedure scans only the observed data $\boldsymbol{X}^{(\mathrm{o})}$ whereas the original procedure scans all entries, observed or not, of the data matrix. With our approach, each computation of the criterion has complexity $\mathcal{O}(k_1 \cdot k_2 \cdot \#\boldsymbol{X}^{(\mathrm{o})})$, where $\#\boldsymbol{X}^{(\mathrm{o})}$ is the number of observed entries, instead of $\mathcal{O}(k_1 \cdot k_2 \cdot n_1 \cdot n_2)$.

2.2.2.6 Prediction on non-observed data

Several predictors could be proposed to predict ratings from the model fitted by variational inference. Ideally, the quantity we would use for predicting an unobserved rating (i, j) is the probability that the rating is positive conditionally to all observed data: $p(X_{ij} = 1|\boldsymbol{X}^{(\mathrm{o})}; \theta)$. As the rating (i, j) is unobserved, we approximate this quantity using our generative model with $p(X_{ij} = 1|\widehat{\boldsymbol{Y}}_i, \widehat{\boldsymbol{Z}}_j, M_{ij} = 0; \widehat{\theta})$.

Let the following quantities:

$$\widehat{m}_{ij}^{(1)} = 1 - \operatorname{expit}\left(\widehat{\mu} + \nu_i^{(A)} + \nu_i^{(B)} + \nu_j^{(C)} + \nu_j^{(D)}\right)$$
$$\widehat{m}_{ij}^{(0)} = 1 - \operatorname{expit}\left(\widehat{\mu} + \nu_i^{(A)} - \nu_i^{(B)} + \nu_j^{(C)} - \nu_j^{(D)}\right)$$

$$\widetilde{p}_{ij} = \sum_{ql} \tau_{iq}^{(Y)} \widehat{\pi}_{ql} \tau_{jl}^{(Z)}$$

$$\widehat{p}_{ij} = \frac{\widehat{m}_{ij}^{(1)}\, \widetilde{p}_{ij}}{\widehat{m}_{ij}^{(1)}\, \widetilde{p}_{ij} + \widehat{m}_{ij}^{(0)}\, (1 - \widetilde{p}_{ij})}$$

Proposition 1 *With the previously defined quantities:*

- $\widehat{m}_{ij}^{(1)}$ *is an estimator of* $p(M_{ij} = 0 | X_{ij} = 1; \theta)$
- $\widehat{m}_{ij}^{(0)}$ *is an estimator of* $p(M_{ij} = 0 | X_{ij} = 0; \theta)$
- $\widetilde{p}_{ij}$ *is an estimator of* $p(X_{ij} = 1; \theta)$
- $\widehat{p}_{ij}$ *is an estimator of* $p(X_{ij} = 1 | M_{ij} = 0; \theta)$

$\widehat{m}_{ij}^{(1)}$ and $\widehat{m}_{ij}^{(0)}$ are built from the maximum *a posteriori* estimates of the latent variables A, B, C, D and with $\widehat{\theta}$ the maximum likelihood estimate obtained with the variational EM algorithm.
$\widehat{m}_{ij}^{(1)}$ (resp. $\widehat{m}_{ij}^{(0)}$) represents the probability that the rating is missing conditionally to the fact that the rating is positive (resp. negative).
$\widetilde{p}_{ij}$ is built with the expected *a posteriori* of the latent variables $\boldsymbol{Y}, \boldsymbol{Z}$ and with $\widehat{\theta}$.
$\widehat{p}_{ij}$ is built with the previous estimates by applying the Bayes theorem.
$\widehat{p}_{ij}$ represents the probability that the rating is positive conditionally to the fact that this rating is missing in $\boldsymbol{X}^{(\mathrm{o})}$.

2.2.3 Model selection

2.2.3.1 Integrated Completed Likelihood criterion (ICL)

ICL, inspired by the Bayesian Information Criterion, was originally proposed to select a relevant number of classes for mixture models [Biernacki et al., 1998]. It was extended to select an appropriate number of (row and column) clusters in the standard Latent Block Model [Keribin et al., 2012]: for k_1 row classes and k_2 column classes, the criterion is defined as

$$\log \int p(\boldsymbol{X}, \boldsymbol{Y}, \boldsymbol{Z} | \theta; k_1, k_2) p(\theta; k_1, k_2) \mathrm{d}\theta \ , \tag{2.14}$$

with $p(\theta; k_1, k_2)$ the prior distribution of parameters. By taking into account the latent variables $\boldsymbol{Y}, \boldsymbol{Z}$, ICL is a clustering-oriented criterion, whereas BIC or AIC are driven by the faithfulness to the distribution of $\boldsymbol{X}$ [Biernacki et al., 1998].

For the LBM with MNAR missingness, ICL requires priors on the parameters of the missingness model. We chose independent InverseGamma(1, 1) distributions for the parameters σ_A^2, σ_B^2, σ_C^2 and σ_D^2. As in [Keribin et al.,

2012], we use non-informative Dirichlet distribution priors on the $\boldsymbol{\alpha}$ and $\boldsymbol{\beta}$ parameters of class mixing proportions.

Proposition 2 *The asymptotic ICL criterion,*

$$\begin{aligned} ICL^{\infty}(k_1,k_2) = & \max_{\theta,\boldsymbol{Y},\boldsymbol{Z},\boldsymbol{A},\boldsymbol{B},\boldsymbol{C},\boldsymbol{D}} \log p\Big(\boldsymbol{X}^{(\mathrm{o})},\boldsymbol{Y},\boldsymbol{Z},\boldsymbol{A},\boldsymbol{B},\boldsymbol{C},\boldsymbol{D};\theta\Big) \\ & - \frac{k_1-1}{2}\log(n_1) - \frac{k_2-1}{2}\log(n_2) \\ & - \frac{k_1k_2+1}{2}\log(n_1n_2) - \log(n_1n_2) \ , \end{aligned}$$

for the LBM with the MNAR model of Section 2.2.1.2 is, up to an irrelevant constant, an asymptotic expansion of the log integrated completed likelihood

$$\log \int p(\boldsymbol{X},\boldsymbol{Y},\boldsymbol{Z},\boldsymbol{A},\boldsymbol{B},\boldsymbol{C},\boldsymbol{D}|\theta;k_1,k_2)p(\theta;k_1,k_2)\mathrm{d}\theta \ .$$

See proof in Appendix 2.A

As the maximized completed log-likelihood required for the asymptotic ICL cannot be calculated (see Section 2.2.2), in practice we use the expectation of the completed log-likelihood under the variational posterior on latent space defined as the difference between the lower bound provided by the variational approximation and the entropy of the variational distribution (see Equation 2.10). The asymptotic ICL is approximated by:

$$\begin{aligned} \mathcal{J}\big(\widehat{\gamma},\widehat{\theta}\big) - \mathcal{H}(q_{\widehat{\gamma}}) - \frac{k_1-1}{2}\log(n_1) - \frac{k_2-1}{2}\log(n_2) \\ - \frac{k_1k_2+1}{2}\log(n_1n_2) - \log(n_1n_2) \ , \end{aligned}$$

where $\big(\widehat{\gamma},\widehat{\theta}\big) = \arg\max_{\gamma,\theta} \mathcal{J}\big(\widehat{\gamma},\widehat{\theta}\big)$.

An asymptotic ICL criterion for the LBM with MAR data can be constructed in the same way, allowing for comparison with the MNAR model as the models are nested (see details in Appendix 2.A).

2.2.4 Experiments on simulated data

Simulated data brings all the elements to assess clustering algorithms in controlled settings. Using controlled datasets provides the means to properly test the ability of an algorithm to recover the known underlying structure.

2.2.4.1 Difficulty of a co-clustering task

In co-clustering, several loss functions are suited for measuring the discrepancy between the underlying classes ($\boldsymbol{Y}$, $\boldsymbol{Z}$) and some predictions ($\widehat{\boldsymbol{Y}}$, $\widehat{\boldsymbol{Z}}$). For

our experiments, we will use the measure defined by Govaert and Nadif [2008], that is, the ratio of misclassified entries in the data matrix:

$$l_{item}(\boldsymbol{Y}, \boldsymbol{Z}, \widehat{\boldsymbol{Y}}, \widehat{\boldsymbol{Z}}) = 1 - \max_{t\in\Omega_1, s\in\Omega_2} \frac{1}{n_1 n_2} \sum_{ijql} Y_{iq}, \widehat{Y}_{it(q)} Z_{jl}, \widehat{Z}_{js(l)} \ ,$$

where Ω_1 (resp. Ω_2) is the set of all possible permutations of $\{1, \ldots, k_1\}$ (resp. $\{1, \ldots, k_2\}$), introduced to take into account the fact that cluster indexes are known only up to a permutation.

In standard clustering, the difficulty of a task is often assessed by its Bayes risk, that is, by the minimum of the expectation of the loss function, which is typically approximated by Monte Carlo on simulated data. Co-clustering poses specific difficulties. Adding more rows or more columns alters its difficulty because the dimensions of the spaces where the clustering is performed are expanded. The duality between the rows and the columns implies that the size of the matrix is a characteristic of a co-clustering problem. In other words, given a fixed generative distribution, as the matrix size increases, the difficulty of the task decreases, in contrast to simple clustering, where the difficulty, as measured by the Bayes risk, remains constant when more examples (that is, rows) are added.

A simple Monte Carlo approximation of the risk consists in averaging over many statistical units. In simple clustering, this means generating a great number of rows in a data matrix. In co-clustering, the statistical unit is the whole matrix, implying that a Monte Carlo approximation of the risk is obtained by generating a great number of data matrices, which then involves a great computational time. Furthermore, estimating the Bayes risk from a single data matrix is very inconstant; the risk may be very different between two data matrices of the same size generated from the same distribution. Hence the usual notion of Bayes risk is not appropriate for co-clustering. Lomet et al. [2012] argue that conditioning the Bayes risk on the observed matrix is more appropriate. They give a protocol to simulate data matrices in which the difficulty of the clustering task is controlled by the following *conditional Bayes risk*:

$$r_{item}(\widehat{\boldsymbol{Y}}, \widehat{\boldsymbol{Z}}) = \mathbb{E}\Big[l_{item}(\boldsymbol{Y}, \boldsymbol{Z}, \widehat{\boldsymbol{Y}}, \widehat{\boldsymbol{Z}}) \Big| \boldsymbol{X}^{(\mathrm{o})} \Big] \ , \tag{2.15}$$

where the expectation is taken over $\boldsymbol{Y}, \boldsymbol{Z}$ only and $\widehat{\boldsymbol{Y}}, \widehat{\boldsymbol{Z}}$ are the clusterings returned by the *conditional Bayes classifier*, that is, the maximum *a posteriori*:

$$\big(\widehat{\boldsymbol{Y}}, \widehat{\boldsymbol{Z}}\big) = \operatorname*{arg\,min}_{(\boldsymbol{Y}, \boldsymbol{Z})} r_{item}(\boldsymbol{Y}, \boldsymbol{Z}) = \operatorname*{arg\,max}_{(\boldsymbol{Y}, \boldsymbol{Z})} \sum_{ij} p\Big(Y_i, Z_j \Big| \boldsymbol{X}^{(\mathrm{o})} \Big) \ .$$

The expectation (2.15) involves the non tractable posterior of the latent variables $p(\boldsymbol{Y}, \boldsymbol{Z} | \boldsymbol{X}^{(\mathrm{o})})$ (see Section 2.2.2). The expectation is approximated using an empirical distribution obtained with a Gibbs sampling procedure to draw labels from $p(\boldsymbol{Y}, \boldsymbol{Z} | \boldsymbol{X}^{(\mathrm{o})})$.

Lomet et al. [2012] released data sets, with different sizes and difficulties, simulated from the Latent Block Model. Using their protocol, we generated new data according the LBM with a MNAR missingness process. Data sets are generated according to the LBM with three row and column classes, with parameters

$$\boldsymbol{\alpha} = \boldsymbol{\beta} = \begin{pmatrix} 1/3 \\ 1/3 \\ 1/3 \end{pmatrix} \qquad \text{and} \qquad \boldsymbol{\pi} = \begin{pmatrix} \varepsilon & \varepsilon & 1-\varepsilon \\ \varepsilon & 1-\varepsilon & 1-\varepsilon \\ 1-\varepsilon & 1-\varepsilon & \varepsilon \end{pmatrix} , \tag{2.16}$$

where ε defines the difficulty of the clustering task. The parameters of the MNAR process are

$$\mu = 1, \quad \sigma_A^2 = 1, \quad \sigma_B^2 = 1, \quad \sigma_C^2 = 1, \quad \sigma_D^2 = 1 \ , \tag{2.17}$$

which gives an average proportion of 35% of missing values.

2.2.4.2 Class prediction

We test here the ability of the proposed inference scheme to recover row and column classes. To conduct the experiments, we generate an initial data matrix of size $n_1 = n_2 = 500$ with a conditional Bayes risk of 5% set by choosing ε (2.16) by trial and error. The size of this matrix is then progressively reduced, removing rows and columns, to increase the difficulty of the classification task. The conditional Bayes risk is re-estimated on each sub-matrix to provide a reference. Our algorithm is then run on these data matrices using 20 initializations for each run, as described in Section 2.2.2.4. We then predict the row and column classes $(\boldsymbol{Y}, \boldsymbol{Z})$ with their maximum *a posteriori* estimators on the variational distribution. This whole process is repeated 20 times, leading to the results presented in Figure 2.12.

As expected, the conditional Bayes risk decreases as the data matrices grow. The predictions returned by our algorithm follow the same pattern, with a diminishing gap to the conditional Bayes risk as the data matrices grow, which is consistent with our expectations. Appendix 2.A.3.2 provides additional experimental results that suggest consistent estimations of the model parameters.

2.2.4.3 MNAR versus MAR model for MNAR data

The importance of using the right missingness model is tested by comparing the classifications returned by an LBM with and without an MNAR model. A data set is generated according to the LBM with MNAR values where the parameters $\boldsymbol{\alpha}$, $\boldsymbol{\beta}$ and $\boldsymbol{\pi}$ of the LBM are fixed as in (2.16), and ε is chosen in order to get a conditional Bayes risk of 12%, for data matrices of size $n_1 = n_2 = 100$; the MNAR model parameters μ, σ_A^2 and σ_C^2 are all set to one

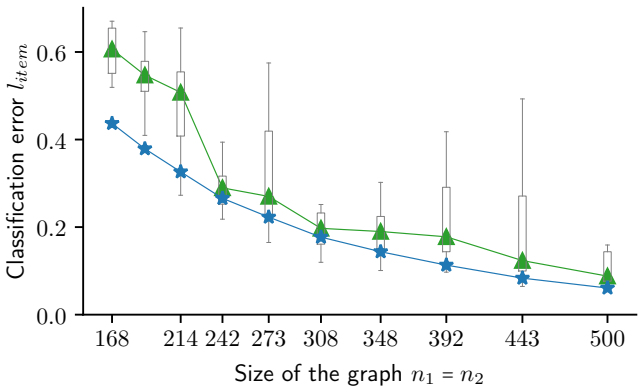


Figure 2.12: Classification error with respect to the size of the data matrix (lower is better); ⋆ is the median of the conditional Bayes risk; ▲ is the median prediction error obtained by our algorithm.

which gives an average proportion of 35% of missing values. Several data matrices are generated using these parameters while varying the value of the σ_B^2 and σ_D^2 parameters that govern the MNAR effects; these variations do not affect the conditional Bayes risk nor the proportion of missing values as latent variables $\boldsymbol{B}$ and $\boldsymbol{D}$ follow Gaussian distributions centered at zero (see Figure 2.11). For each data matrix, we train the LBM with either the MAR or the MNAR model. We also train a categorical LBM [Keribin et al., 2015] considering missing values to be level of the categorical distribution using the "blockcluster" package [Bhatia et al., 2014]. This process is repeated 20 times, starting from the generation of a new fully observed data matrix.

The median of the classification errors l_{item} are presented in Figure 2.13 as a function of the MNAR effect. They are essentially constant and close to the conditional Bayes risk for the LBM with the MNAR model, whereas the LBM with the MAR model is badly affected by MNAR data, eventually leading to a classification close to a totally random allocation [1]. Ignoring the nature of the missingness process leads here to strong biases in estimation that in turn drastically affect classification. Thankfully, the ICL criterion may be of great help to select the right missingness model as shown in Section 2.2.4.5. The trend of classification errors obtained with the categorical LBM is always close to the classification error of the totally random allocation. Explaining missingness by cluster membership is not relevant here and impairs the fit of the model. For similar reasons, poor performances were found (not shown here) using the ordinal LBM with MNAR missingness proposed by Corneli et al. [2020] in which data and missingness dependent on the same row and column clusters.

[1] With equal class proportions, the expected classification error of a random allocation is $\frac{k_1-1}{k_1} + \frac{k_2-1}{k_2} - \frac{k_1-1}{k_1}\frac{k_2-1}{k_2}$, that is, 0.89 here where $k_1 = k_2 = 3$.

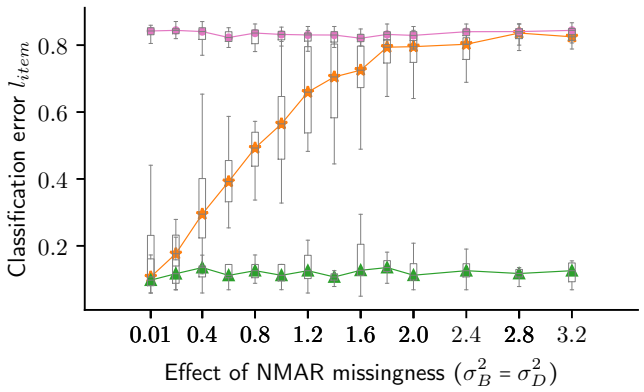


Figure 2.13: Classification error with respect to an increase of the MNAR effect (lower is better); ⋆ is the median prediction error obtained with the MAR model; ▲ is the median prediction error obtained with the MNAR model; ; ● is the median prediction error obtained with the categorical LBM.

2.2.4.4 Selecting the number of classes

We reuse the parameters (2.16) and (2.17) to analyze the behavior of the ICL criterion. We consider different sizes of data matrices, between (30,30) and (150,150), with varying difficulty for each matrix size, with a conditional Bayes risk (2.15) of respectively 5%, 12% and 20%

The results in Figure 2.14 show that, as expected, the ICL criterion tends to select more often the right number of classes as the data matrices get larger and also when classes are more separated. We also observe that the ICL criterion tends to be conservative for small data matrices, by underestimating the number of classes. It could come to the fact that the size of the matrix is not large enough to consider the asymptotic approximation as valid and/or it could come from the approximations used to compute the log-likelihood $\mathcal{J}$ (variational restriction and delta method).

2.2.4.5 Selecting the adequate missingness model

We use the models fitted in Section 2.2.4.3 to analyze the ability of the ICL criterion to select the right missingness model (MNAR or MAR). The difference in asymptotic ICL between the MAR and MNAR models is computed for each data matrix, assuming that the right numbers of classes (k_1, k_2) are known.

The results, presented in Figure 2.15, show that ICL rightfully opts for the MNAR model almost everywhere, demonstrating the ability of this criterion to select the adequate missingness model. The MAR model is only chosen for some experiments with the lowest MNAR effect ($\sigma_B^2 = \sigma_D^2 = 0.01$), where the prediction performances are almost identical (see Figure 2.13).

			$r_{item}(\widehat{\boldsymbol{Y}},\widehat{\boldsymbol{Z}})$ = 5%				$r_{item}(\widehat{\boldsymbol{Y}},\widehat{\boldsymbol{Z}})$ = 12%				$r_{item}(\widehat{\boldsymbol{Y}},\widehat{\boldsymbol{Z}})$ = 20%			
			k_2				k_2				k_2			
			2	3	4	5	2	3	4	5	2	3	4	5
$n_1 = n_2 = 30$	k_1	2	4	3	1		7	5			10	3		
		3	3	**9**			5	**1**	1		5	**1**		
		4					1					1		
		5												
$n_1 = n_2 = 40$	k_1	2	3	4			10	4	1		12	2	1	
		3		**12**				**5**			4	**1**		
		4		1										
		5												
$n_1 = n_2 = 50$	k_1	2		1			6	2			15	1	2	
		3	2	**16**				**11**			1	**0**		
		4		1			1				1			
		5												
$n_1 = n_2 = 75$	k_1	2	3				10	1			16			
		3		**16**				**8**				**4**		
		4					1							
		5	1											
$n_1 = n_2 = 100$	k_1	2					6				17			
		3		**20**				**14**				**2**	1	
		4												
		5												
$n_1 = n_2 = 150$	k_1	2		1			4	1			15		1	
		3		**18**				**15**				**4**		
		4												
		5		1										

Figure 2.14: Count number of (k_1, k_2) models selected by the asymptotic ICL criterion among 20 data matrices for different difficulties, as measured by the conditional Bayes risk, and different matrix sizes. All matrices are generated with the same number of row and column classes: $k_1 = k_2 = 3$.

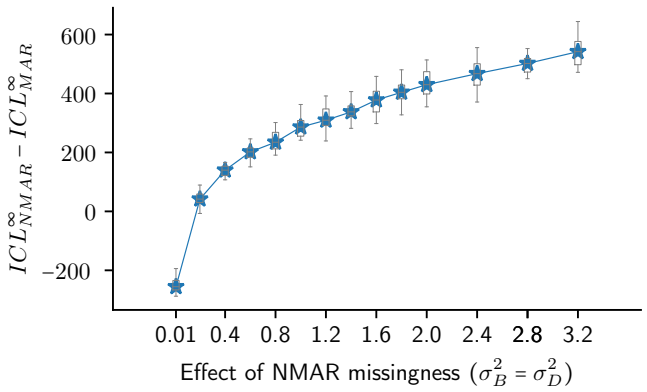


Figure 2.15: Difference in asymptotic ICL between the MAR and MNAR models with respect to an increase of the MNAR effect, where ⋆ is the median. The MNAR model is selected when the difference in ICL is positive.

2.2.4.6 Missing data prediction

As a first step, we use a toy dataset to illustrate the differences between a model with an MCAR missingness mechanism and a model with an MNAR missingness mechanism. The missingness in the dataset generated below is of MNAR type but it was not simulated according to our missingness model.

Horror-Romance Dataset We follow the generation protocol of the horror-romance dataset of Hernández-Lobato et al. [2014] and Schnabel et al. [2016]. The item clusters gather romance and horror movies; there are four user clusters by considering the cartesian product of romance lover or horror lover and positive raters or negative raters.

We consider binary ratings (positive/negative opinion). Romance-lovers rate positively romance movies with probability $p = 0.75$ and rate positively horror movies with probability $p = 0.25$. Similarly, horror-lovers rate positively horror movies with probability $p = 0.75$ and rate positively romance movies with probability $p = 0.25$.

Some users give their opinions more often when they like a movie (positive raters), whereas other users give their opinions more often when they dislike a movie (negative raters). The probability of observing a rating thus depends on the rating itself: the data is MNAR. The probabilities to observe a rating is set to 0.175 if the user belongs to the category of positive raters, and 0.025 if the user belongs to the category of negative raters.

We generate an observation rating matrix $\boldsymbol{X}^{(o)}$ with 2000 users and 1000 items. Half of the users (resp. items) are romance lovers (resp. movies) and the other half are horror lovers (resp. movies). Each user category is split equally in two sub-categories gathering respectively positive and negative raters.

Experimental Protocol We compare our LBM-MNAR against a LBM assuming MCAR data, whose model of missingness requires a single parameter, the overall probability of being missing. We recall that, for this last model, the missingness model does not affect the fit of the ratings model.

We split the entries of the observed rating matrix $\boldsymbol{X}^{(\mathrm{o})}$ into a training set (with 99% of the data) and a validation set (with 1% of the data). The validation set is kept small to reduce interference with the missingness mechanism. We train our model, the MNAR-LBM, and the classical LBM assuming MCAR data. We suppose that (k_1, k_2), the number of row and column clusters, are known.

As the LBM estimates are sensitive to initialization, we repeat the training process 50 times from different random initializations, and we select the model with lower mean absolute error on the validation set. As we are using a synthetic dataset, the final performance of each model can be evaluated with the mean absolute error on all non-observed ratings data $\boldsymbol{X}^{(\mathrm{m})}$. The whole process is repeated 20 times to compute average results based on the mean absolute error (MAE) defined as:

$$MAE = \frac{\sum_{ij \in \boldsymbol{X}^{(\mathrm{m})}} \left| \hat{X}_{ij} - X_{ij}^{(\mathrm{m})} \right|}{\# \left\{ ij : X_{ij}^{(\mathrm{m})} = 1 \wedge X_{ij}^{(\mathrm{m})} = 0 \right\}}$$

Results According to a paired t-test (Student test), the prediction performances of the MNAR-LBM with a mean absolute error of 0.349 on the probability predictions are significantly better that the MCAR-LBM with a mean absolute error of 0.377. One unexpected result is that the standard deviation of the MAE is higher with the MNAR-LBM (0.0196) than with the MCAR-LBM (0.0013). This variability could reflect some greater optimization difficulties due to the higher number of parameters used in the MNAR-LBM.

Figure 2.16 shows an observed rating matrix $\boldsymbol{X}^{(\mathrm{o})}$, ordered according to the true groups, and reordered according to the estimated groups. The matrix on the right shows that our model is able to recover all the groups, whereas the matrix in the center shows that the MCAR-LBM only distinguishes two main groups of users and empties the two additional groups. Because of its wrong missingness assumption, the MCAR-LBM model is not able to recover the underlying user propensities and creates homogeneous but biased blocks of users that will impact the rating predictions.

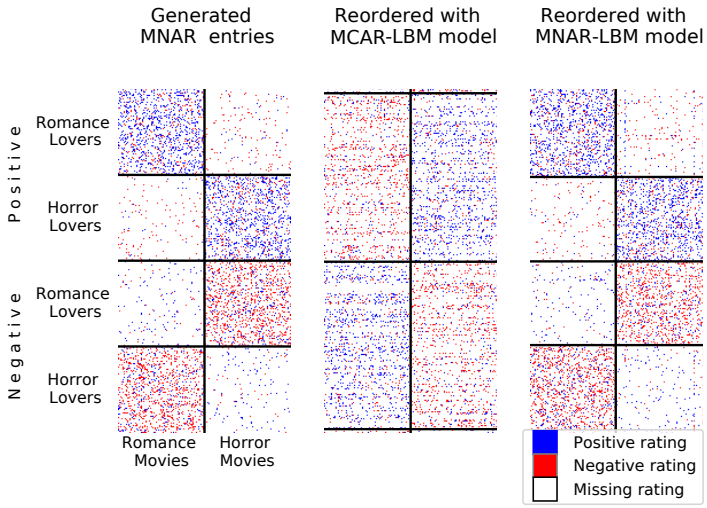


Figure 2.16: Observed MNAR ratings matrix of a romance-horror dataset: left, original matrix ordered with true groups; center, matrix re-ordered by an MCAR-LBM model; right, matrix re-ordered by our MNAR-LBM model. Users and items groups are artificially separated with plain lines.

2.2.5 Experiments on real data

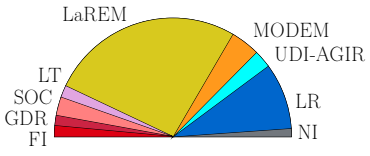
2.2.5.1 Class prediction on French Parliament

We consider voting records[2] from the lower house of the French Parliament (*Assemblée Nationale*). This dataset gathers the results of the 1256 ballots of year 2018 of the 577 French members of parliament (MPs) for the procedural motions and amendments for the 15th legislature (June 2017). For each ballot, the vote of each MP is recorded as a 4-level categorical response: "yes", 'no", "abstained" or "absent". Using our model, we bring out some relevant groups of ballots and MPs, as well as some structure in the behavior of nonvoters.

We gather the data in a matrix where each row represents an MP and each column represents a ballot. To use our model, we reduced the 4 response levels to 3 ("yes", 'no", "missing") assuming that merging the "abstained" and "absent" categories would not affect much the underlying missingness process ("abstained" votes represent about 4% of the expressed votes, "missing" responses represent 85% of all votes).

At the lower house of French Parliament, MPs may group together according to their political affinities. Groups with fewer than 15 members or MPs who choose to be independent are gathered under the "Non inscrits" (NI) label,

[2]Votes from the French National Assembly are available from `http://data.assemblee-nationale.fr/travaux-parlementaires/votes`.



Political groups from left-wing to right-wing

FI (17): France Insoumise
GDR (16): Groupe de la Gauche démocrate et républicaine
SOC (29): Socialistes
LT (19): Libertés et territoires
LaREM (304): La République En Marche
MODEM (46): Mouvement démocrate
UDI-AGIR (28): Les Constructifs
LR (104): Les Républicains
NI (13): Non inscrits (mixed left and right wings)

Figure 2.17: Hemicycle of the political groups of the French National Assembly

giving a heterogeneous range of political hues inside it. The names of the groups and their frequency are detailed in Figure 2.17.

The ICL criterion, used to select both the numbers of classes and the type of missingness, favors a MNAR missingness with $k_1 = 14$ MP classes and $k_2 = 14$ ballot classes (see Figure 2.20) against a MAR model with 19 MP classes 23 ballot classes. The reordered data matrix derived from this block clustering is displayed in Figure 2.18. Fewer classes lead to over-aggregated components hiding the subtleties of the network, but since they still correspond to well-identified groups and are more friendly to visual analysis, we provide them as additional material in Appendix 2.A.3.2.

In Figure 2.18, classes of MPs are coherent to their political orientation: class 0 and 1 are mainly made up of left-wing MPs from the groups SOC, FI, GDR, LT, classes 2 and 3 are mainly made up of right-wing MPs from LR and the classes from 6 to 13 are mainly made up of centrist MPs from LaREM and MODEM who are known to be political allies. Classes of ballots can be analyzed with the available metadata. A bipartite opposition system appears from classes A and C. Ballots from class A refer to the original articles of law proposed by the government and are unsurprisingly voted positively by the MPs classes from 6 to 13 as they are from the same political mould as the French government. Ballots from class C mainly refer to amendments proposed by minority and are voted positively by both the left wing (class 0 and 1) and the right wing (classes 2 and 3) and negatively by the MPs supporting the government (classes 6 to 13). The left and right wings are yet divided by usual issues such as immigration regulation amendments gathered in classes G and M or general economic matters gathered in classes H and I.

In our model, the latent variables $\boldsymbol{A}$ and $\boldsymbol{B}$ characterize the propensity of MPs to cast a vote. Figure 2.19 displays the scatter plot of $\nu_i^{(A)}$ and $\nu_i^{(B)}$, the maximum *a posteriori* estimates of A_i and B_i for all MPs under the variational distribution. The abscissa represents the propensity to vote[3], with higher

[3]More rigorously, the abscissa represents the *global deviation from the average* propensity to vote.

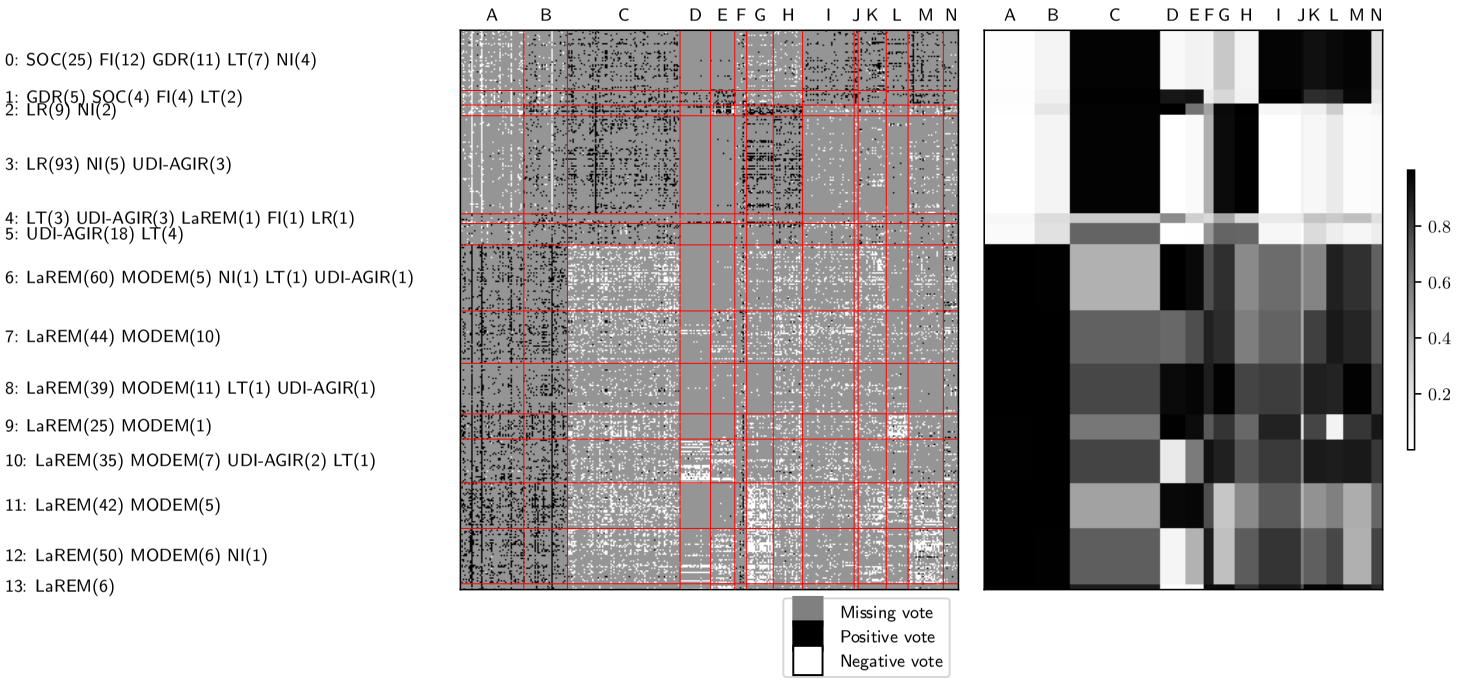


Figure 2.18: Left: matrix of votes reordered according to the row and column classes, for the MNAR LBM model selected by ICL, with 14 MP classes and 14 ballot classes. The red lines delineate class boundaries. The counts of MPs belonging to their political groups in each MP class is given on the left. Right: summary of the inferred opinions (expressed or not) for all classes of ballots and MPs, as given by the estimated probability to approve a resolution in each block of the reordered matrix.

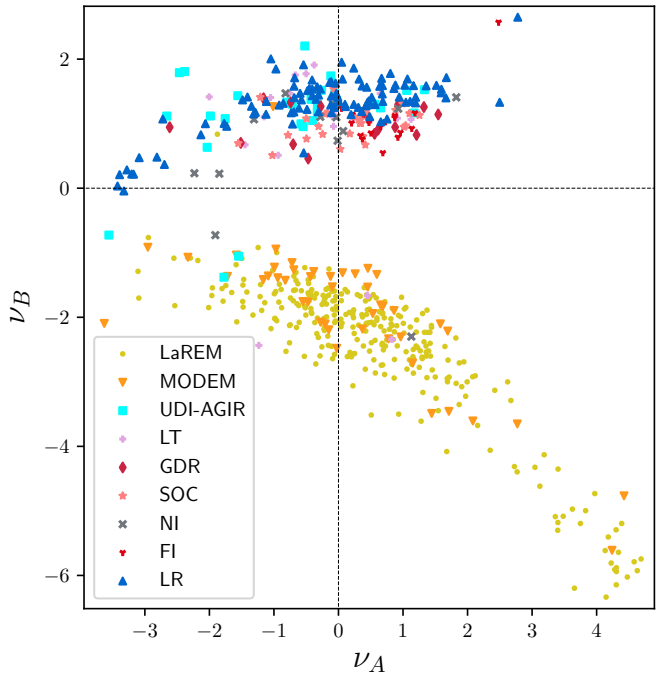


Figure 2.19: Maximum *a posteriori* estimates of the MPs propensities ($\nu_i^{(A)}$, $\nu_i^{(B)}$), with their political group memberships. $\nu_i^{(A)}$ drives the MAR effect and $\nu_i^{(B)}$ drives the MNAR one.

values of $\boldsymbol{\nu}^{(A)}$ corresponding to a higher propensity to vote, and the ordinate $\boldsymbol{\nu}^{(B)}$ represents the additional effect of casting a vote when approving the resolution. The membership of MPs to their political group is indicated by the plotting symbol.

We see two obvious clusters separated by the vertical axis $\boldsymbol{\nu}^{(B)}$: the bottom cluster is essentially formed by MPs from the LaREM and MODEM political groups, which support the government, whereas the top cluster is formed by the opposition political groups. The $\boldsymbol{\nu}^{(B)}$ estimates for the opposition cluster are positive, meaning that these MPs come to parliament to vote positively. This behavior is not surprising because the MPs of the opposition parties are outnumbered by the MPs supporting the government, so they must be diligent if they want their tabled motion or amendment passed. The dependency between the political groups and the MNAR effect encoded in the estimates $\boldsymbol{\nu}^{(B)}$, which is confirmed by an ANOVA test (with a p-value smaller than numerical error), supports that the missingness patterns captured by our model are relevant for the problem at hand. A similar analysis is developed on ballots in Appendix 2.A.3.2.

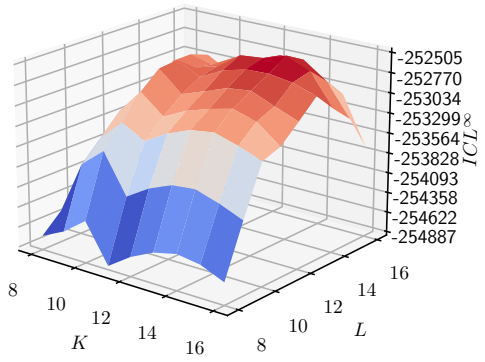


Figure 2.20: ICL curve. Maximum is reached for k_1=14 and k_2=14

2.2.5.2 Missing data prediction on Yahoo! R3 dataset

The standard protocol to assess the performances of a model uses a subset of the observed data to train the model and a different subset of the observed data to test it. If the observed data is MNAR, the standard testing procedure is biased as the distribution of the observed data is different from the distribution of the missing data. To properly assess the performances of a model that posits an MNAR missingness, the test set should consist of a random sample of ratings (MAR or MCAR) for each user, while the training set would consist of MNAR ratings.

The R3-Yahoo! Music dataset[4] contains ratings for songs collected from users during normal interaction with Yahoo! Music services and also from users during an online survey conducted by Yahoo! Research. The users from the survey had to give their opinion on some randomly selected songs they had to listen. The ratings range from 1 to 5 for 15,400 users and 1000 songs. The data contains exactly ten ratings of randomly selected songs for 5400 users of the dataset. A training set is built with the ratings collected from users during normal interaction and a test set is built with the ratings collected from the randomly selected songs. The distribution of ratings in Figure 2.21 shows the inadequacy of the usual MAR assumption.

Few models for collaborative filtering handling MNAR data have been proposed and tested on the Yahoo! R3 dataset. Marlin et al. [2011] are the pioneer to model some missing not at random rating responses with a multinomial mixture model denoted *MM CPT-v* (see Section 2.1.3.3) in which the missing data mechanism is very simple. The results based on the

[4] http://webscope.sandbox.yahoo.com/

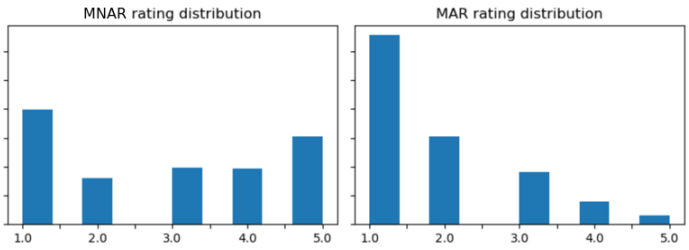


Figure 2.21: Rating distribution of R3-Yahoo! Music dataset. Left: distribution of the training set. Right: distribution of the test set that consists of ratings from randomly selected songs. The two ratings distributions are very different showing that the usual implicit assumption of Missing At Random (MAR) ratings is incorrect.

root-mean-square error (RMSE):

$$RMSE = \sqrt{\frac{\sum_{ij:X^{\text{test}}} \left(\hat{X}_{ij} - X_{ij}\right)^2}{\#\left\{ij : X^{\text{test}}\right\}}}$$

and displayed in Table 2.2 show that this model greatly improve performances over the multinomial mixture with MAR assumption. Also, they propose *MM Logit-vd*, a second model with a more complex missingness using characteristics of the items. Yet, this model performs worse than the MM CPT-v.

Hernández-Lobato et al. [2014] propose a double probabilistic MF model, one for the complete data and one for the MNAR missing data, but their model do not perform better than the similar one with MAR assumption (in Table 2.2, RMSE is 1.483 vs 1.480) and far worse than MM CPT-v and MM Logit-vd.

Schnabel et al. [2016] propose an empirical risk minimization framework to derive a propensity scored matrix factorization method that can account for selection bias. Their standard model, *Prop-MF MNAR naive*, gives poor prediction performances (RMSE is 1.375). A variant of their model, using Inverse-Propensity-Scoring estimator (*Prop-MF MNAR IPS*) offers good performances (RMSE is 0.994) but the method needs a small MCAR sample (5% of the test set) for eliciting the marginal rating distribution needed to estimate the propensities. Thus, the method can not be fairly compared with other baselines.

Finally, the constant predictor that outputs the most frequent rating of the test set, regardless of the user and song, performs very close to MM CPTv, the best baseline that also predicts almost always predicts the same value for missing ratings. Surprisingly, in all these baselines, the simplest methods perform best. A reasonable hypothesis is that the test set is too small and

little informative. Ten ratings per user does not seem enough to estimate the true MCAR distribution. For example, about 11% of users gave the lowest rating to all the test items they were exposed to. Additionally, the hypothesis made in the Yahoo! study that users would not change their behavior when forced to give feedback is questionable.

Table 2.2: Average root-mean-square error on the MAR test set of Yahoo dataset. Results taken from supplementary of Hernández-Lobato et al. [2014], from Schnabel et al. [2016] and Marlin et al. [2011]

Methods	RMSE
Prop-MF MNAR IPS Schnabel et al. [2016]	<u>0.994</u>
Prop-MF MNAR naive Schnabel et al. [2016]	1.375
MF MNAR Hernández-Lobato et al. [2014]	1.483
MF MAR Hernández-Lobato et al. [2014]	1.480
MM MAR Marlin et al. [2011]	1.500
MM CPTv MNAR Marlin et al. [2011]	**1.056**
MM Logitvd MNAR Marlin et al. [2011]	1.141
Oracle	1.057

Nonetheless, we conduct some experiments with the model LBM-MNAR proposed in Chapter 2. We binarize the ratings by considering that ratings above 2 are positive. Figure 2.22 illustrates the rating distributions of the training and test sets. The number of user clusters and item clusters in co-clustering are arbitrarily set to ten. To measure the ranking performance of algorithms, we use the Normalized Discounted Cumulative Gain (NDCG) that measures ranking quality by a penalized sum of the relevance scores of the ranking results:

$$NDCG@k = \frac{DCG@k}{IDCG@k} \text{ with } DCG@k = \sum_{i=1}^{k} \frac{rel_i}{\log(i+1)} \ ,$$

rel_i, the relevance of the results at each rank i before k and $IDCG@k$ being the $DCG@k$ computed with a perfect ranking. The LBM-MNAR is compared with the MM-CPTv proposed by Marlin et al. [2011] and with the strategy that returns a randomly ordered list of items for each user. Table 2.3 depicts the ranking performance of models with the NDCG, averaged over all users,

for a recommendation list of 5 or 10 items. Unsurprisingly, because the NDCG is based on ranks, the MM CPTv model has poor performance as it predicts almost all the time the same rating.

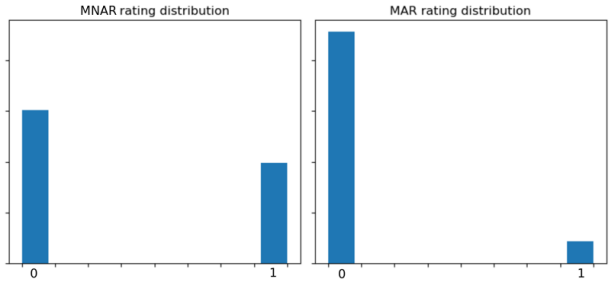


Figure 2.22: Rating distribution of the binarized R3-Yahoo! Music dataset. Left: distribution of the training set. Right: distribution of the test set that consists of ratings from randomly selected songs. Ratings above 2 are considered as positive.

Table 2.3: Normalized Discounted Cumulative Gain estimated on the binarized Yahoo Mar test set (the higher the better).

Methods	NDCG@5	NDCG@10
LBM-MNAR	0.7529	0.8764
MM-CPTv Marlin et al. [2011]	0.7470	0.8734
Random	0.7473	0.8734

Conclusion: We come to the conclusion that the Yahoo study had a major impact on the collaborative filtering domain by showing a clear dependence of rating frequency on the underlying preference level. The common implicit assumption of Missing At Random (MAR) ratings in collaborative filtering is incorrect and the bias induced by using a test set with MNAR data to assess the performances of a recommender system is most probably significant. However, the test set from the Yahoo! R3 dataset appears to be small, making it difficult to properly assess any predictor. Additionally, it is likely that behavior of users change when they are forced to give feedback. The lack of other dataset with MAR test set is a problematic barrier to progress in the domain of recommender system.

2.2.5.3 Missing data prediction with MAR assumption on test sets

We conduct some experiments on datasets classically used for investigating recommender systems. In these datasets, an absence of rating most probably conveys some information as the feedbacks are collected from users who freely chose the items they rate. However these datasets do not have a test set made of a random selection of unrated items. ⚠ *Thus standard protocols to assess the performances of a model considering MNAR data will be biased.* The following experiments presented in this section are only illustrations to show that the computational efficiency improvements made on our MNAR-LBM model (Section 2.2.2.5) give the opportunity to handle standard recommendation datasets.

We compare our LBM-MNAR with methods based on matrix factorization (MF). We use Singular Value Decomposition (SVD) and its extension SVD++, both popularized during the Netflix challenge [**?**]. We also include Non-negative Matrix Factorization (NMF), an algorithm similar to SVD but where user and item factors are kept positive [**?**]. Finally, we add a matrix factorization method that accounts for selection bias: Propensity-weighted Matrix Factorization (Prop. MF) [Schnabel et al., 2016]. We were not able to compare with Hernández-Lobato et al. [2014] MF model due to outdated scientific libraries used in their implementation.

Datasets The MovieLens 100k and 1M datasets [**?**] contain ratings given by users to movies. The original ratings range from 1 to 5 (the favorite). We binarize the ratings by considering that ratings above 2 are positive so that the rate between positive and negative ratings are roughly balanced.

The Movie Tweetings dataset [**?**] is based on public and well-structured tweets. The tweets were parsed to extract the ratings given by user to movies. The dataset was first released by **?**. The original ratings range from 1 to 10. We binarize the ratings by considering that ratings above 7 are positive so that the positive and negative ratings are roughly balanced.

The NIPS 2013 dataset [**?**] is a set of ratings given by scientific reviewers during the bidding process of the 2013 NIPS conference. A rating represents the declared interest to review a given paper : (*not willing*, *in a pinch*, *willing*, *eager*). The ratings are binarized as positive when the levels are *willing* or *eager*.

For computational stability reasons, we removed, for each dataset, rows and columns which have less than 10 ratings. Characteristics of the datasets, after having removed rows and columns, are summarized in Table 2.4.

Experimental Protocol We use the experimental protocol described in Section 2.2.4.6. However, as the ground truth is unknown for missing ratings, we split the observed ratings into three parts: a training set, a validation set

Table 2.4: Characteristics of the datasets

Dataset	n_1	n_2	Nb ratings	Sparsity
Synth.	2000	1000	200000	10%
ML100k	943	1349	99287	7.8%
ML1M	6040	3260	998539	5%
Movie Tweets	3272	3069	146459	1.5%
NIPS	950	1381	52009	4%

Table 2.5: Area Under Curve ROC with MAR assumption on test sets.

Dataset	LBM MNAR	LBM	Prop. MF	SVD	SVD++	NMF
ML100k	0.766	**0.786**	0.766	0.766	0.771	0.781
ML1M	0.792	0.797	0.790	0.793	**0.801**	0.790
NIPS	0.827	0.830	**0.840**	0.809	0.821	0.816
Movie Tweets	0.829	0.837	0.8235	0.831	0.842	**0.849**

and a test set, with proportions of 98%, 1% and 1% respectively. The numbers of clusters, k_1 for users and k_2 for items, are not known in the MNAR-LBM, and the classical LBM. These two hyper-parameters are estimated during the validation step. As validation and test sets are built from observed data, the ratings are estimated using a predictor on $p(X_{ij} = 1|M_{ij} = 1; \theta)$ rather that $p(X_{ij} = 1|M_{ij} = 0; \theta)$.

Results and Discussion Table 2.5 shows the values of the area under the ROC curve obtained using each method on the datasets. In accordance with our expectations as the test sets are not made of a random selection of unrated items, LBM-MNAR and Propensity MF, the two models that accounts for selection bias, don't consistently outperformed other standard baselines. A surprising fact is that the LBM would compare well to classical matrix factorization. A possible explanation is that the data binarization affected the performance of models initially designed for more complex data.

Table 2.6 displays the trained parameters of our missingness model on the selected datasets. We remind that σ_A and σ_B are related to the dispersion of user propensity and that σ_B drives the MNAR effect. Similarly, σ_C and σ_D are related to the dispersion of item propensity with σ_D driving the item MNAR effect. On all datasets, it is clear from the low values of σ_D that the model does not detect a strong MNAR effect imputable to items.

Table 2.6: Parameters learnt by our missingness model. σ_A and σ_B are related to user propensity while σ_C and σ_D are related to item propensity.

Dataset	μ	σ_A	σ_B	σ_C	σ_D
ML100k	-3.46	0.94	0.66	1.26	0.32
ML1M	-3.77	1.08	0.53	1.44	0.26
Movie Tweets	-5.21	0.89	0.55	0.92	0.14
NIPS	-3.69	1.00	1.66	0.08	0.17

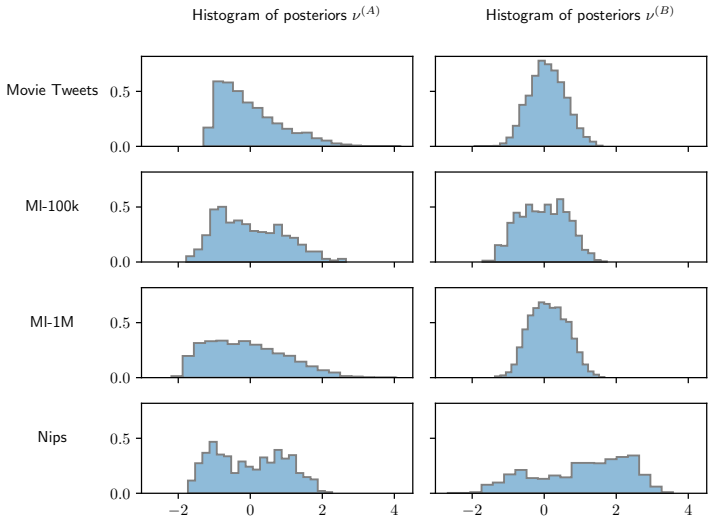


Figure 2.23: Histograms of maximum a posteriori of variables $\boldsymbol{A}$ and $\boldsymbol{B}$ according to several datasets. $\boldsymbol{A}$ and $\boldsymbol{B}$ characterize users propensity to give their opinion.

On the NIPS dataset, the very low value of σ_C conveys that the bid distribution is not influenced by any effect of the papers however we suspect this property is a consequence of the NIPS bidding process; the reviewers are representative of the subjects of the submitted papers. Still on this dataset, the high value of σ_B reveals that users have heterogeneous bidding behaviors. The histograms of the maximum *a posteriori* estimates of users propensities on Figures 2.23 and 2.24 confirm an unusual rating behavior on the NIPS dataset. A source of this heterogeneous rating behavior could come from several types of bidding strategy: some reviewers prefer to bid positively on seemingly interesting papers, some other ones prefer to bid negatively on papers they don't want to review, and other ones have a mixed strategy.

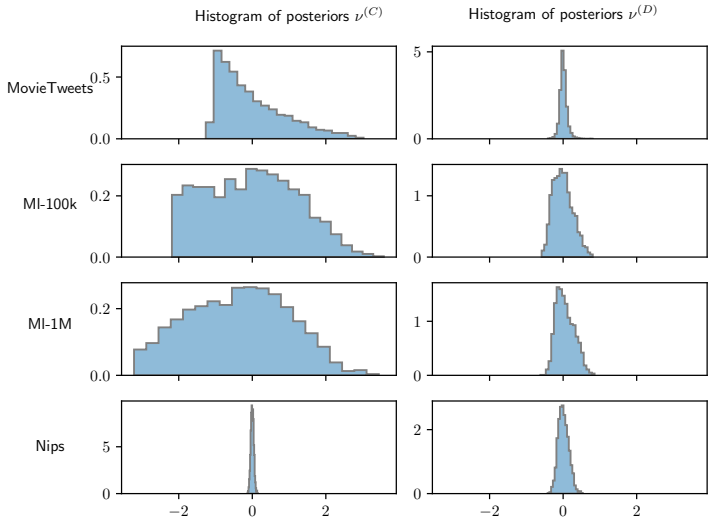


Figure 2.24: Histograms of maximum a posteriori of variables $\boldsymbol{C}$ and $\boldsymbol{D}$ according to several datasets. $\boldsymbol{C}$ and $\boldsymbol{D}$ characterize items propensity to receive a rating.

2.3 Conclusion

In many estimation problems, the absence of data conveys some information on the underlying phenomenon that should be exploited for its modeling. We proposed a co-clustering model that accounts for this absence of data; it aims at retrieving groups of rows and columns based on the complete data matrix instead of considering only the partitioning of the observed data matrix. This model consists of two building blocks: a co-clustering model (Latent Block Model) of the full data matrix, and a model of the missingness of the entries of the data matrix. This missingness model preserves the symmetry of the co-clustering model by allowing two MNAR effects, one on the rows and the other on the columns. The overall model of the observed data matrix results from the combination of the model of the complete data matrix with the missingness model.

We used variational techniques and Taylor series to obtain a tractable approximation of the lower bound of the observed log-likelihood. We proposed a model selection criterion to select both the number of classes and the type of missingness (MAR versus MNAR).

Our experiments on synthetic datasets have shown that ignoring an informative missingness can lead to catastrophic co-clustering estimates, supporting the value of using expressive missingness models on such type of data. We also illustrated the use of our model on real-world cases where the missingness model provides an interesting basis for analyzing and interpreting the motivations of nonvoters. These experiments can be reproduced using the source

code and the dataset available at https://github.com/gfrisch/LBM-MNAR.

Our model should also be useful in other fields such as in ecology, where the probability of observing interaction between species derives from some factors that also explain the true interactions [Vázquez et al., 2009], or in collaborative filtering, where the probability of observing a rating depends on the actual rating that would be given by the user [Marlin et al., 2007]. In the latter application, the data sizes generally encountered in recommendation would require computational improvements in inference. Another useful future work is to extend our model to non-binary data.

Bibliography

Jean-Patrick Baudry and Gilles Celeux. EM for mixtures. *Statistics and Computing*, 25(4):713–726, 2015. doi: 10.1007/s11222-015-9561-x. 51

Parmeet Bhatia, Serge Iovleff, and Gérard Govaert. blockcluster: An R Package for Model Based Co-Clustering. working paper or preprint, December 2014. URL https://hal.inria.fr/hal-01093554. 57

Christophe Biernacki, Gilles Celeux, and Gérard Govaert. Assessing a mixture model for clustering with the integrated classification likelihood. Technical Report RR-3521, INRIA, October 1998. URL https://hal.inria.fr/inria-00073163. 53

Christophe Biernacki, Gilles Celeux, and Gérard Govaert. Choosing starting values for the EM algorithm for getting the highest likelihood in multivariate Gaussian mixture models. *Computational Statistics & Data Analysis*, 41: 561–575, 01 2003. doi: 10.1016/S0167-9473(02)00163-9. 51

G. Celeux and J. Diebolt. The SEM algorithm: a probabilistic teacher algorithm derived from the EM algorithm for the mixture problem. *Computational Statistics Quarterly*, 2:73–82, 1985. 47

Marco Corneli, Charles Bouveyron, and Pierre Latouche. Co-clustering of ordinal data via latent continuous random variables and not missing at random entries. *Journal of Computational and Graphical Statistics*, 2020. doi: 10.1080/10618600.2020.1739533. URL https://hal.archives-ouvertes.fr/hal-01978174. 35, 57, 165

Inderjit S. Dhillon. Co-clustering documents and words using bipartite spectral graph partitioning. In *Proceedings of the seventh ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 269–274, aug 2001. 41

Inderjit S. Dhillon, Subramanyam Mallela, and Dharmendra S. Modha. Information-theoretic co-clustering. In *Proceedings of the Ninth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '03, pages 89—-98, New York, NY, USA, 2003. Association for Computing Machinery. ISBN 1581137370. doi: 10.1145/956750.956764. URL https://doi.org/10.1145/956750.956764. 41

Chris Ding, Tao Li, Wei Peng, and Haesun Park. Orthogonal nonnegative matrix t-factorizations for clustering. In *Proceedings of the Twelfth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, volume 2006, pages 126–135, 01 2006. doi: 10.1145/1150402.1150420. 41

Thomas George and Srujana Merugu. A scalable collaborative filtering framework based on co-clustering. In *Fifth IEEE International Conference on Data Mining (ICDM)*, 12 2005. ISBN 0-7695-2278-5. doi: 10.1109/ICDM.2005.14. 41

Gérard Govaert and Mohamed Nadif. Block clustering with Bernoulli mixture models: Comparison of different approaches. *Computational Statistics & Data Analysis*, 52(6):3233–3245, February 2008. 41, 47, 55

José Miguel Hernández-Lobato, Neil Houlsby, and Zoubin Ghahramani. Probabilistic matrix factorization with non-random missing data. In *Proceedings of the 31st International Conference on Machine Learning*, pages 1512–1520, 2014. 32, 39, 44, 60, 67, 68, 70

Tommi S. Jaakkola. Tutorial on variational approximation methods. In Manfred Opper and David Saad, editors, *Advanced Mean Field Methods: Theory and Practice*, pages 129–159. MIT Press, 2000. 48

Julien Jacques and Christophe Biernacki. Model-Based Co-clustering for Ordinal Data. *Computational Statistics & Data Analysis*, 123:101–115, July 2018. doi: 10.1016/j.csda.2018.01.014. URL https://hal.inria.fr/hal-01448299. 35

Janus Jakobsen, Christian Gluud, Jørn Wetterslev, and Per Winkel. When and how should multiple imputation be used for handling missing data in randomised clinical trials - a practical guide with flowcharts. *BMC Medical Research Methodology*, 17, 12 2017. doi: 10.1186/s12874-017-0442-1. 31

Michael I. Jordan, Zoubin Ghahramani, Tommi S. Jaakkola, and Lawrence K. Saul. An introduction to variational methods for graphical models. *Machine Learning*, 37(2):183—-233, November 1999. ISSN 0885-6125. doi: 10.1023/A:1007665907178. URL https://doi.org/10.1023/A:1007665907178. 48

Christine Keribin, Vincent Brault, Gilles Celeux, and Gérard Govaert. Model selection for the binary latent block model. In *Proceedings of COMPSTAT*, 08 2012. 53, 83

Christine Keribin, Vincent Brault, Gilles Celeux, and Gérard Govaert. Estimation and selection for the latent block model on categorical data. *Statistics and Computing*, 25(6):1201–1216, 2015. 41, 45, 47, 57, 85

Yong-Deok Kim and Seungjin Choi. Bayesian binomial mixture model for collaborative prediction with non-random missing data. In *Eighth ACM Conference on Recommender Systems (RecSys)*, page 201–208, 2014. 36, 38

Yuval Kluger, Ronen Basri, Joseph Chang, and Mark Gerstein. Spectral biclustering of microarray data: Coclustering genes and conditions. *Genome Research*, 13:703–716, 05 2003. doi: 10.1101/gr.648603. 41

Lazhar Labiod and Mohamed Nadif. Co-clustering for binary and categorical data with maximum modularity. In *11th IEEE International Conference on Data Mining (ICDM)*, pages 1140–1145, 2011. doi: 10.1109/ICDM.2011.37. 41

Pierre Latouche, Etienne Birmelé, and Christophe Ambroise. Overlapping stochastic block models with application to the French political blogosphere. *The Annals of Applied Statistics*, 5(1):309–336, Mar 2011. ISSN 1932-6157. doi: 10.1214/10-aoas382. URL http://dx.doi.org/10.1214/10-AOAS382. 41

Guang Ling, Haiqin Yang, Michael R. Lyu, and Irwin King. Response aware model-based collaborative filtering. In *Proceedings of the Twenty-Eighth Conference on Uncertainty in Artificial Intelligence*, UAI'12, page 501–510, Arlington, Virginia, USA, 2012. AUAI Press. ISBN 9780974903989. 38

Roderick J. A. Little and Donald B. Rubin. Introduction. In *Statistical Analysis with Missing Data*, chapter 1, pages 1–23. John Wiley & Sons, 1986. ISBN 9781119013563. 31

Aurore Lomet. *Sélection de modèle pour la classification croisée de données continues*. PhD thesis, Université de technologie de Compiègne, 2012. URL http://www.theses.fr/2012COMP2041. Thèse de doctorat dirigée par Govaert, Gérard et Grandvalet, Yves Technologies de l'information et des systèmes Compiègne 2012. 41

Aurore Lomet, Gérard Govaert, and Yves Grandvalet. Design of artificial data tables for co-clustering analysis. Technical report, Université de technologie de Compiègne, France, 2012. 55

Benjamin M. Marlin, Richard S. Zemel, Sam T. Roweis, and Malcolm Slaney. Collaborative filtering and the missing at random assumption. In *Twenty-Third Conference on Uncertainty in Artificial Intelligence (UAI)*, pages 267–275, 2007. 32, 33, 44, 74

Benjamin M. Marlin, Richard S. Zemel, Sam T. Roweis, and Malcolm Slaney. Recommender systems, missing data and statistical model estimation. In *Proceedings of the 22nd International Joint Conference on Artificial Intelligence (IJCAI)*, pages 2686–2691, 2011. 36, 37, 66, 68, 69

Benjamin M. Marlin, Richard S. Zemel, Sam T. Roweis, and Malcolm Slaney. Collaborative filtering and the missing at random assumption. *CoRR*, abs/1206.5267, 2012. URL http://arxiv.org/abs/1206.5267. 32

Mohamed Nadif and Gérard Govaert. Latent block model for contingency table. *Communications in Statistics—Theory and Methods*, 39(3):416–425, 01 2010. doi: 10.1080/03610920903140197. 41

Evangelos E. Papalexakis, Nikolaos Sidiropoulos, and Rasmus Bro. From k-means to higher-way co-clustering: Multilinear decomposition with sparse latent factors. *IEEE Transactions on Signal Processing*, 61(2):493–506, January 2013. ISSN 1053-587X. doi: 10.1109/TSP.2012.2225052. 35

Giorgio Parisi. *Statistical field theory*. Frontiers in Physics. Addison-Wesley, 1988. URL https://cds.cern.ch/record/111935. 49

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. Pytorch: An imperative style, high-performance deep learning library. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems 32*, pages 8024–8035. Curran Associates, Inc., 2019. 51

Beatriz Pontes, Raúl Giráldez, and Jesús S. Aguilar-Ruiz. Biclustering on expression data: A review. *Journal of Biomedical Informatics*, 57: 163–180, 2015. ISSN 1532-0464. doi: https://doi.org/10.1016/j.jbi.2015.06.028. URL http://www.sciencedirect.com/science/article/pii/S1532046415001380. 41

Karl Rohe, Sourav Chatterjee, and Bin Yu. Spectral clustering and the high-dimensional stochastic blockmodel. *The Annals of Statistics*, 39(4): 1878–1915, Aug 2011. ISSN 0090-5364. doi: 10.1214/11-aos887. URL http://dx.doi.org/10.1214/11-AOS887. 52

D. B. Rubin. *Multiple Imputation for Nonresponse in Surveys*. John Wiley & Sons, 1987. 31

Donald B. Rubin. Inference and missing data. *Biometrika*, 63(3):581–592, 1976. ISSN 00063444. URL http://www.jstor.org/stable/2335739. 30

Tobias Schnabel, Adith Swaminathan, Ashudeep Singh, Navin Chandak, and Thorsten Joachims. Recommendations as treatments: Debiasing learning and evaluation. In *Proceedings of the 33nd International Conference on Machine Learning (ICML)*, pages 1670–1679, 2016. URL http://proceedings.mlr.press/v48/schnabel16.html. 60, 67, 68, 70

Margot Selosse, Julien Jacques, and Christophe Biernacki. Model-based co-clustering for mixed type data. *Computational Statistics & Data Analysis*, 144:106866, 2020a. ISSN 0167-9473. doi: https://doi.org/10.1016/j.csda.2019.106866. URL http://www.sciencedirect.com/science/article/pii/S016794731930221X. 35

Margot Selosse, Julien Jacques, and Christophe Biernacki. Textual data summarization using the self-organized co-clustering model. *Pattern Recognition*, 103:107315, 2020b. ISSN 0031-3203. doi: https://doi.org/10.1016/j.patcog.2020.107315. URL http://www.sciencedirect.com/science/article/pii/S0031320320301199. 41

Hanhuai Shan and Arindam Banerjee. Bayesian co-clustering. In *2008 Eighth IEEE International Conference on Data Mining*, pages 530–539, 2008. 41

Emilie Shireman, Douglas Steinley, and Michael Brusco. Examining the effect of initialization strategies on the performance of Gaussian mixture modeling. *Behavior Research Methods*, 49, 12 2015. doi: 10.3758/s13428-015-0697-6. 51

Harald Steck. Training and testing of recommender systems on data missing not at random. In *16th ACM International Conference on Knowledge Discovery and Data Mining (SIGKDD)*, pages 713–722, 2010. 39

Jonathan AC Sterne, Ian R White, John B Carlin, Michael Spratt, Patrick Royston, Michael G Kenward, Angela M Wood, and James R Carpenter. Multiple imputation for missing data in epidemiological and clinical research: potential and pitfalls. *Bmj*, 338, 2009. 31

Timothée Tabouy, Pierre Barbillon, and Julien Chiquet. Variational inference for stochastic block models from sampled data. *Journal of the American Statistical Association*, 115(529):455–466, 2020. 33, 34, 36

Diego P Vázquez, Nico Blüthgen, Luciano Cagnolo, and Natacha P Chacoff. Uniting pattern and process in plant–animal mutualistic networks: a review. *Annals of botany*, 103(9):1445–1457, 2009. 74

Wei Wang. Identifiability of linear mixed effects models. *Electronic Journal of Statistics*, 7:244–263, 2013. 45

Larry Wasserman. *All of Statistics: A Concise Course in Statistical Inference*. Springer, 2004. 51, 94

Jason Wyse and Nial Friel. Block clustering with collapsed latent block models. *Statistics and Computing*, 22(2):415–428, 2012. 41

2.A Appendix - Learning from missing data with the Latent Block Model

2.A.1 Computing the criterion $\mathcal{J}(q_\gamma, \theta)$

The criterion to be optimized is :

$$\mathcal{J}(q_\gamma, \theta) = \mathcal{H}(q_\gamma) + \mathbb{E}_{q_\gamma}[\log p(\boldsymbol{X}^{(o)}, \boldsymbol{Y}, \boldsymbol{Z}, \boldsymbol{A}, \boldsymbol{B}, \boldsymbol{C}, \boldsymbol{D}; \theta)] \ ,$$

where θ is the list of all model parameters: $\theta = (\boldsymbol{\alpha}, \boldsymbol{\beta}, \boldsymbol{\pi}, \mu, \sigma_A^2, \sigma_B^2, \sigma_C^2, \sigma_D^2)$. We restrict the form of the variational distribution q_γ to get a fully factorized form:

$$q_\gamma = \prod_{i=1}^{n_1} \mathcal{M}\left(1; \boldsymbol{\tau}_i^{(Y)}\right) \times \prod_{j=1}^{n_2} \mathcal{M}\left(1; \boldsymbol{\tau}_j^{(Z)}\right) \times \prod_{i=1}^{n_1} \mathcal{N}\left(\nu_i^{(A)}, \rho_i^{(A)}\right) \times \prod_{i=1}^{n_1} \mathcal{N}\left(\nu_i^{(B)}, \rho_i^{(B)}\right) \times \prod_{j=1}^{n_2} \mathcal{N}\left(\nu_j^{(C)}, \rho_j^{(C)}\right) \times \prod_{j=1}^{n_2} \mathcal{N}\left(\nu_j^{(D)}, \rho_j^{(D)}\right) \ ,$$

where γ denotes the list of parameters of the distribution: $\gamma = (\boldsymbol{\tau}^{(Y)}, \boldsymbol{\tau}^{(Z)}, \boldsymbol{\nu}^{(A)}, \boldsymbol{\rho}^{(A)}, \boldsymbol{\nu}^{(B)}, \boldsymbol{\rho}^{(B)}, \boldsymbol{\nu}^{(C)}, \boldsymbol{\rho}^{(C)}, \boldsymbol{\nu}^{(D)}, \boldsymbol{\rho}^{(D)})$.

The entropy is additive across independent variables, so we get:

$$\begin{aligned}\mathcal{H}(q_\gamma) = &- \sum_{iq} \tau_{iq}^{(Y)} \log \tau_{iq}^{(Y)} - \sum_{jl} \tau_{jl}^{(Z)} \log \tau_{jl}^{(Z)} \\ &+ (n_1 + n_2)(\log(2\pi) + 1) \\ &+ \frac{1}{2} \sum_i \left(\log \rho_i^{(A)} + \log \rho_i^{(B)}\right) + \frac{1}{2} \sum_j \left(\log \rho_j^{(C)} + \log \rho_j^{(D)}\right) \ .\end{aligned}$$

The independence of the latent variables allows one to rewrite the expectation of the complete log-likelihood as:

$$\begin{aligned}\mathbb{E}_{q_\gamma}\left[\log p\left(\boldsymbol{X}^{(o)}, \boldsymbol{Y}, \boldsymbol{Z}, \boldsymbol{A}, \boldsymbol{B}, \boldsymbol{C}, \boldsymbol{D}\right)\right] = &\ \mathbb{E}_{q_\gamma}[\log p(\boldsymbol{Y})] \\ &+ \mathbb{E}_{q_\gamma}[\log p(\boldsymbol{Z})] + \mathbb{E}_{q_\gamma}[\log p(\boldsymbol{A})] + \mathbb{E}_{q_\gamma}[\log p(\boldsymbol{B})] \\ &+ \mathbb{E}_{q_\gamma}[\log p(\boldsymbol{C})] + \mathbb{E}_{q_\gamma}[\log p(\boldsymbol{D})] \\ &+ \mathbb{E}_{q_\gamma}\left[\log p\left(\boldsymbol{X}^{(o)} \middle| \boldsymbol{Y}, \boldsymbol{Z}, \boldsymbol{A}, \boldsymbol{B}, \boldsymbol{C}, \boldsymbol{D}\right)\right] \ ,\end{aligned}$$

with the following terms:

$$\begin{aligned}
\mathbb{E}_{q_\gamma}[\log p(\boldsymbol{Y})] &= \sum_{iq} \mathbb{E}_{q_\gamma} Y_{iq} \log \alpha_q = \sum_{iq} \tau_{iq}^{(Y)} \log \alpha_q \\
\mathbb{E}_{q_\gamma}[\log p(\boldsymbol{Z})] &= \sum_{jl} \mathbb{E}_{q_\gamma} Z_{jl} \log \beta_l = \sum_{jl} \tau_{jl}^{(Z)} \log \beta_l \\
\mathbb{E}_{q_\gamma}[\log p(\boldsymbol{A})] &= -\frac{n_1}{2} \log 2\pi - \frac{n_1}{2} \log \sigma_A^2 - \frac{1}{2\sigma_A^2} \sum_i \mathbb{E}_{q_\gamma} A_i^2 \\
&= -\frac{n_1}{2} \log 2\pi - \frac{n_1}{2} \log \sigma_A^2 \\
&\quad - \frac{1}{2\sigma_A^2} \sum_i \left(\left(\nu_i^{(A)}\right)^2 + \rho_i^{(A)} \right)
\end{aligned}$$

$$\begin{aligned}
\mathbb{E}_{q_\gamma}[\log p(\boldsymbol{B})] &= -\frac{n_1}{2} \log 2\pi - \frac{n_1}{2} \log \sigma_B^2 \\
&\quad - \frac{1}{2\sigma_B^2} \sum_i \left(\left(\nu_i^{(B)}\right)^2 + \rho_i^{(B)} \right) \\
\mathbb{E}_{q_\gamma}[\log p(\boldsymbol{C})] &= -\frac{n_2}{2} \log 2\pi - \frac{n_2}{2} \log \sigma_C^2 \\
&\quad - \frac{1}{2\sigma_C^2} \sum_j \left(\left(\nu_j^{(C)}\right)^2 + \rho_j^{(C)} \right) \\
\mathbb{E}_{q_\gamma}[\log p(\boldsymbol{D})] &= -\frac{n_2}{2} \log 2\pi - \frac{n_2}{2} \log \sigma_D^2 \\
&\quad - \frac{1}{2\sigma_D^2} \sum_j \left(\left(\nu_j^{(D)}\right)^2 + \rho_j^{(D)} \right)
\end{aligned}$$

$$\begin{aligned}
\mathbb{E}_{q_\gamma}\left[\log p\left(\boldsymbol{X}^{(\mathrm{o})} \middle| \boldsymbol{Y}, \boldsymbol{Z}, \boldsymbol{A}, \boldsymbol{B}, \boldsymbol{C}, \boldsymbol{D}\right)\right] &= \sum_{ql,ij:X_{ij}^{(\mathrm{o})}=1} \tau_{iq}^{(Y)} \tau_{jl}^{(Z)} \mathbb{E}_{q_\gamma}[\log p_1] \\
&\quad + \sum_{ql,ij:X_{ij}^{(\mathrm{o})}=0} \tau_{iq}^{(Y)} \tau_{jl}^{(Z)} \mathbb{E}_{q_\gamma}[\log p_0] \\
&\quad + \sum_{ql,ij:X_{ij}^{(\mathrm{o})}=\mathrm{NA}} \tau_{iq}^{(Y)} \tau_{jl}^{(Z)} \mathbb{E}_{q_\gamma}[\log(1 - p_0 - p_1)] \ , \qquad (2.18)
\end{aligned}$$

with p_0 and p_1 defined in Equations (2.5)–(2.6).

Equation (2.18) involves the computation of the expectations of the following nonlinear functions:

$$\begin{aligned}
f_1(x, y) &= \log\left(\pi_{ql} \operatorname{expit}(\mu + x + y)\right) \\
f_0(x, y) &= \log\left((1 - \pi_{ql}) \operatorname{expit}(\mu + x - y)\right) \\
f_{\mathrm{NA}}(x, y) &= \log\Big(1 - \pi_{ql} \operatorname{expit}(\mu + x + y) \\
&\qquad - (1 - \pi_{ql}) \operatorname{expit}(\mu + x - y)\Big) \ .
\end{aligned}$$

The approximation of these expectations given by the second-order Taylor series with independent random variables X and Y reads:

$$\mathbb{E}[f(X,\,Y)] \approx f(\mathbb{E}X,\,\mathbb{E}Y) + \frac{1}{2}\,var(X)\,\frac{\partial^2 f(\mathbb{E}[X],\,\mathbb{E}[Y])}{\partial(X)^2} + \frac{1}{2}\,var(Y)\,\frac{\partial^2 f(\mathbb{E}[X],\,\mathbb{E}[Y])}{\partial(Y)^2}\ ,$$

which yields in our case:

$$\begin{aligned}\mathbb{E}_{q_\gamma}[f(A_i + C_j,\ B_i + D_j)] \approx\ & f\Big(\nu_i^{(A)} + \nu_j^{(C)},\ \nu_i^{(B)} + \nu_j^{(D)}\Big) \\ & + \frac{1}{2}\Big(\rho_i^{(A)} + \rho_j^{(C)}\Big)\,\frac{\partial^2 f\Big(\nu_i^{(A)} + \nu_j^{(C)},\ \nu_i^{(B)} + \nu_j^{(D)}\Big)}{\partial\Big(\nu_i^{(A)} + \nu_j^{(C)}\Big)^2} \\ & + \frac{1}{2}\Big(\rho_i^{(B)} + \rho_j^{(D)}\Big)\,\frac{\partial^2 f\Big(\nu_i^{(A)} + \nu_j^{(C)},\ \nu_i^{(B)} + \nu_j^{(D)}\Big)}{\partial\Big(\nu_i^{(B)} + \nu_j^{(D)}\Big)^2}\ .\end{aligned}$$

The criterion is now fully computable.

2.A.2 Initialization of the VEM algorithm with spectral clustering: Algorithm 3.

2.A.3 Asymptotic form of the Integrated Completed Likelihood

2.A.3.1 ICL of the MNAR model

The asymptotic criterion,

$$\begin{aligned}ICL^{\infty}(k_1,k_2) = \max_{\theta}\ & \log p\Big(\boldsymbol{X}^{(\mathrm{o})},\boldsymbol{Y},\boldsymbol{Z},\boldsymbol{A},\boldsymbol{B},\boldsymbol{C},\boldsymbol{D};\theta\Big) \\ & - \frac{k_1-1}{2}\log(n_1) - \frac{k_2-1}{2}\log(n_2) \\ & - \frac{k_1k_2+1}{2}\log(n_1n_2) - \log(n_1n_2)\ ,\end{aligned}$$

for the LBM with the MNAR model of Section 2.2.1.2 is, up to an irrelevant constant, an asymptotic expansion of the log integrated completed likelihood

$$\log \int p(\boldsymbol{X},\boldsymbol{Y},\boldsymbol{Z},\boldsymbol{A},\boldsymbol{B},\boldsymbol{C},\boldsymbol{D}|\theta;k_1,k_2)p(\theta;k_1,k_2)\mathrm{d}\theta\ .$$

Proof: With independent latent variables and independent priors on the parameters, the log integrated completed likelihood reads

$$\begin{aligned}
&\log \int p\Big(\boldsymbol{X}^{(\mathrm{o})}, \boldsymbol{Y}, \boldsymbol{Z}, \boldsymbol{A}, \boldsymbol{B}, \boldsymbol{C}, \boldsymbol{D}\Big|\boldsymbol{\theta}\Big)p(\boldsymbol{\theta})\mathrm{d}\boldsymbol{\theta} \\
&= \log \int p\Big(\boldsymbol{X}^{(\mathrm{o})}\Big|\boldsymbol{Y}, \boldsymbol{Z}, \boldsymbol{A}, \boldsymbol{B}, \boldsymbol{C}, \boldsymbol{D}, \boldsymbol{\pi}\Big)p(\boldsymbol{\pi})p(\mu)\mathrm{d}\boldsymbol{\pi}\mathrm{d}\mu \\
&+ \log \int p(\boldsymbol{Y}|\boldsymbol{\alpha})p(\boldsymbol{\alpha})\mathrm{d}\boldsymbol{\alpha} + \log \int p(\boldsymbol{Z}|\boldsymbol{\beta})p(\boldsymbol{\beta})\mathrm{d}\boldsymbol{\beta} \\
&+ \log \int p\big(\boldsymbol{A}|\sigma_A^2\big)p\big(\sigma_A^2\big)\mathrm{d}\sigma_A^2 + \log \int p\big(\boldsymbol{B}|\sigma_B^2\big)p\big(\sigma_B^2\big)\mathrm{d}\sigma_B^2 \\
&+ \log \int p\big(\boldsymbol{C}|\sigma_C^2\big)p\big(\sigma_C^2\big)\mathrm{d}\sigma_C^2 + \log \int p\big(\boldsymbol{D}|\sigma_D^2\big)p\big(\sigma_D^2\big)\mathrm{d}\sigma_D^2.
\end{aligned} \tag{2.19}$$

As in the ICL developed by Keribin et al. [2012] for the standard LBM, we set non-informative Dirichlet distribution $\mathcal{D}(a, ..., a)$ priors on $\boldsymbol{\alpha}$ and $\boldsymbol{\beta}$:

$$\begin{aligned}
\log p(\boldsymbol{Y}) &= \log \int p(\boldsymbol{Y}|\boldsymbol{\alpha})p(\boldsymbol{\alpha}; a)\mathrm{d}\boldsymbol{\alpha} \\
&= \log \int \prod_{iq} (\alpha_q)^{Y_{iq}} \frac{1}{\mathcal{B}(a)} \prod_{iq} (\alpha_q)^{a-1}\mathrm{d}\boldsymbol{\alpha} \\
&= \log \mathcal{B}(a + \sum_i \boldsymbol{Y}_i) - \log \mathcal{B}(a) \\
&= \sum_q \log \Gamma(Y_{:q} + a) + \log \Gamma(k_1 a) - \log \Gamma(n_1 + k_1 a) \\
&\quad - k_1 \log \Gamma(a) \ ,
\end{aligned}$$

where $Y_{:q} = \sum_i Y_{iq}$. The Stirling expansion $\log \Gamma(x) = x \log x - x - \frac{1}{2}\log x + o(\log x)$ leads to the following asymptotic development of $\log p(\boldsymbol{Y})$:

$$\begin{aligned}
\log p(\boldsymbol{Y}) &= \sum_q \log \Gamma(Y_{:q} + a) - \log \Gamma(n_1 + k_1 a) + o(\log n_1) \\
&= \sum_q Y_{:q} \log Y_{:q} - n_1 - \frac{1}{2} n_1 \\
&\quad - \left(n_1 \log n_1 + k_1 a \log n_1 - n_1 - \frac{1}{2}\log n_1\right) + o(\log n_1).
\end{aligned}$$

With the non-informative Jeffrey prior $a = \frac{1}{2}$, this gives:

$$\begin{aligned}
\log p(\boldsymbol{Y}) &= \sum_q Y_{:q} \log\left(\frac{1}{n_1} Y_{:q}\right) - \frac{k_1 - 1}{2} \log n_1 + o(\log n_1) \\
&= \max_{\boldsymbol{\alpha}} \log p(\boldsymbol{Y}; \boldsymbol{\alpha}) - \frac{k_1 - 1}{2} \log n_1 + o(\log n_1).
\end{aligned} \tag{2.20}$$

Similarly, we get:

$$\begin{aligned}\log p(\boldsymbol{Z}) &= \sum_l \log\Gamma(Z_{:l}+a) + \log\Gamma(k_2 a) - \log\Gamma(n_2+k_2 a) \\ &\quad - k_2\log\Gamma(a) \\ &= \max_{\boldsymbol{\beta}} \log p(\boldsymbol{Z};\boldsymbol{\beta}) - \frac{k_2-1}{2}\log n_2 + o(\log n_2) \ , \end{aligned} \tag{2.21}$$

where $Z_{:l} = \sum_j Z_{jl}$.

An InverseGamma(ξ, ξ) prior is set on σ_A^2:

$$\begin{aligned} p(\boldsymbol{A}) &= \int p\big(\boldsymbol{A}|\sigma_A^2\big)p\big(\sigma_A^2;\xi\big)\,\mathrm{d}\sigma_A^2 \\ &= \int \big(2\pi\sigma_A^2\big)^{-\frac{n_1}{2}} \exp\left(-\frac{\sum_i A_i^2}{2\sigma_A^2}\right)\frac{\xi^\xi}{\Gamma(\xi)}\exp\left(-\frac{\xi}{\sigma_A^2}\right)\big(\sigma_A^2\big)^{-\xi-1}\,\mathrm{d}\sigma_A^2 \\ &= \frac{\xi^\xi}{\Gamma(\xi)}(2\pi)^{\left(-\frac{n_1}{2}\right)}\int \sigma_A^{2\left(-\frac{n_1}{2}-\xi-1\right)}\exp\left(-\frac{2\xi+\sum_i A_i^2}{2}\cdot\frac{1}{\sigma_A^2}\right)\mathrm{d}\sigma_A^2 \\ &= \frac{\xi^\xi}{\Gamma(\xi)}2^\xi\pi^{-\frac{n_1}{2}}\left(2\xi+\sum_i A_i^2\right)^{\left(-\frac{n_1}{2}-\xi\right)}\Gamma\big(\frac{n_1}{2}+\xi\big) \ . \end{aligned}$$

Setting ξ to 1, one gets:

$$p(\boldsymbol{A}) = 2\,\pi^{-\frac{n_1}{2}}\left(2+\sum_i A_i^2\right)^{-\frac{n_1}{2}-1}\Gamma\left(\frac{n_1}{2}+1\right) \ ,$$

therefore,

$$\begin{aligned}\log p(\boldsymbol{A}) &= \log 2 - \frac{n_1}{2}\log\pi + \log\Gamma\left(\frac{n_1}{2}+1\right) - \left(\frac{n_1}{2}+1\right)\log\left(2+\sum_i A_i^2\right) \\ &= \log 2 - \frac{n_1}{2}\log\pi + \log\Gamma\left(\frac{n_1}{2}+1\right) - \left(\frac{n_1}{2}+1\right)\log n_1 \\ &\quad - \left(\frac{n_1}{2}+1\right)\log\left(\frac{2}{n_1}+\frac{1}{n_1}\sum_i A_i^2\right) \ . \end{aligned}$$

Using a Taylor expansion on the last term with $n_1 \to +\infty$ and using the fact

that $\frac{1}{n_1}\sum_i A_i^2$ tends to a constant, we obtain:

$$\begin{aligned}
\log p(\boldsymbol{A}) &= \log 2 - \frac{n_1}{2}\log\pi + \log\Gamma\left(\frac{n_1}{2}+1\right) - \left(\frac{n_1}{2}+1\right)\log n_1 \\
&\quad - \left(\frac{n_1}{2}+1\right)\log\left(\frac{1}{n_1}\sum_i A_i^2\right)\left(1+O\left(\frac{2}{n_1}\right)\right) \\
&= \log 2 - \frac{n_1}{2}\log\pi + \log\Gamma\left(\frac{n_1}{2}+1\right) - \left(\frac{n_1}{2}+1\right)\log n_1 \\
&\quad - \left(\frac{n_1}{2}+1\right)\log\left(\frac{1}{n_1}\sum_i A_i^2\right) + O(1) \\
&= \log 2 - \frac{n_1}{2}\log\pi + \log\Gamma\left(\frac{n_1}{2}+1\right) - \left(\frac{n_1}{2}+1\right)\log n_1 \\
&\quad - \frac{n_1}{2}\log\left(\frac{1}{n_1}\sum_i A_i^2\right) + O(1)
\end{aligned}$$

Using the property of the gamma function $\Gamma(x+1) = x\Gamma(x)$ and applying the Stirling expansion of $\log\Gamma(x)$, we get for $n_1 \to +\infty$:

$$\begin{aligned}
\log p(\boldsymbol{A}) &= \log 2 - \frac{n_1}{2}\log\pi + \log\frac{n_1}{2} + \frac{n_1}{2}\log\frac{n_1}{2} - \frac{n_1}{2} \\
&\quad - \frac{1}{2}\log\frac{n_1}{2} - \left(\frac{n_1}{2}+1\right)\log n_1 \\
&\quad - \frac{n_1}{2}\log\left(\frac{1}{n_1}\sum_i A_i^2\right) + o(\log n_1) \\
&= -\frac{n_1}{2}\log(2\pi) - \frac{n_1}{2} - \frac{n_1}{2}\log\left(\frac{1}{n_1}\sum_i A_i^2\right) \\
&\quad - \frac{1}{2}\log n_1 + o(\log n_1) \\
&= \max_{\sigma_A^2} \log p\left(\boldsymbol{A};\sigma_A^2\right) - \frac{1}{2}\log n_1 + o(\log n_1) \quad . \qquad (2.22)
\end{aligned}$$

Similarly, with an identical prior on σ_B^2, σ_C^2 and σ_D^2 we get:

$$\begin{aligned}
\log p(\boldsymbol{B}) &= \max_{\sigma_B^2} \log p\left(\boldsymbol{B};\sigma_B^2\right) - \frac{1}{2}\log n_1 + o(\log n_1) \\
\log p(\boldsymbol{C}) &= \max_{\sigma_C^2} \log p\left(\boldsymbol{C};\sigma_C^2\right) - \frac{1}{2}\log n_2 + o(\log n_2) \qquad (2.23) \\
\log p(\boldsymbol{D}) &= \max_{\sigma_D^2} \log p\left(\boldsymbol{D};\sigma_D^2\right) - \frac{1}{2}\log n_2 + o(\log n_2) \quad .
\end{aligned}$$

Using a Laplace approximation as realized in the BIC, the penalty term differs from the categorical LBM [Keribin et al., 2015] as the levels of the

distribution are linked (see (2.5) and (2.6) from Section 2.2.1.2). We have:

$$\log p\Big(\boldsymbol{X}^{(\mathrm{o})}|\boldsymbol{Y},\boldsymbol{Z},\boldsymbol{A},\boldsymbol{B},\boldsymbol{C},\boldsymbol{D}\Big) \tag{2.24}$$

$$= \log \int p\Big(\boldsymbol{X}^{(\mathrm{o})}|\boldsymbol{Y},\boldsymbol{Z},\boldsymbol{A},\boldsymbol{B},\boldsymbol{C},\boldsymbol{D},\boldsymbol{\pi},\mu\Big)p(\boldsymbol{\pi})p(\mu)\mathrm{d}\boldsymbol{\pi}\mathrm{d}\mu$$

$$= \max_{\boldsymbol{\pi},\mu} \log p\Big(\boldsymbol{X}^{(\mathrm{o})}|\boldsymbol{Y},\boldsymbol{Z},\boldsymbol{A},\boldsymbol{B},\boldsymbol{C},\boldsymbol{D};\boldsymbol{\pi},\mu\Big) \tag{2.25}$$

$$+ \frac{k_1k_2+1}{2}\log(n_1n_2) + o(\log n_1) + o(\log n_2)\,,$$

as the number of free parameters in the conditional distribution of $X^{(o)}$ is $k_1 \times k_2 + 1$ which comes from the (k_1, k_2)-matrix of probabilities π and from μ, governing the global missingness rate. The asymptotic ICL criterion (Proposition 2) is directly derived from Equations (2.19), (2.20), (2.21), (2.22), (2.23) and (2.25).

□

2.A.3.2 ICL of the LBM with MAR data

We consider the following LBM extended with the MAR missingness process:

Latent Block Model

$$Y_i \overset{\text{iid}}{\sim} \mathcal{M}(1;\boldsymbol{\alpha}), \qquad \boldsymbol{\alpha} \in \mathbf{S}_{k_1-1}$$

$$Z_j \overset{\text{iid}}{\sim} \mathcal{M}(1;\boldsymbol{\beta}), \qquad \boldsymbol{\beta} \in \mathbf{S}_{k_2-1}$$

$$(X_{ij}|Y_i = q, Z_j = l) \overset{\text{ind}}{\sim} \mathcal{B}\big(\pi_{ql}\big), \qquad \pi_{ql} \in [0,1]$$

MAR data model

$$A_i \overset{\text{iid}}{\sim} \mathcal{N}\big(0,\sigma_A^2\big), \qquad \sigma_A^2 \in \mathbb{R}_+^*$$

$$C_j \overset{\text{iid}}{\sim} \mathcal{N}\big(0,\sigma_C^2\big), \qquad \sigma_C^2 \in \mathbb{R}_+^*$$

$$(M_{ij}|A_i,C_j) \overset{\text{ind}}{\sim} \mathcal{B}(\operatorname{expit}(\mu + A_i + C_j))$$

Observations are generated according to:

$$X_{ij}^{(\mathrm{o})} = \begin{cases} X_{ij} & \text{if} \quad M_{ij} = 1 \\ \text{NA} & \text{if} \quad M_{ij} = 0 \end{cases}$$

The asymptotic ICL of this model is:

$$ICL^{\infty}(k_1,k_2) = \max_{\boldsymbol{\theta}} \log p\Big(\boldsymbol{X}^{(\mathrm{o})},\boldsymbol{Y},\boldsymbol{Z},\boldsymbol{A},\boldsymbol{C};\boldsymbol{\theta}\Big) \tag{2.26}$$

$$- \frac{k_1 k_2 + 1}{2}\log(n_1 n_2) - \frac{k_1 - 1}{2}\log(n_1) - \frac{k_2 - 1}{2}\log(n_2) - \frac{1}{2}\log(n_1 n_2)\ . \tag{2.27}$$

2.A.4 Supplemental figures for simulated data

This section provides additional experimental results that suggest a consistent estimation of the model parameters. We reuse the data matrices generated by the LBM with missing data from Section 2.2.4.2. An initial data matrix of size $n_1 = n_2 = 500$ with a conditional Bayes risk of 5% was generated and progressively reduced, removing rows and columns, to increase the difficulty of the classification task.

Figure 2.25 displays the maximum absolute error made on the parameters $\boldsymbol{\pi}$ of the Bernoulli distributions that model the probability of $\boldsymbol{X}$ conditionally to the row and column classes. This error decreases as the size of the data matrices grows, which is consistent with our expectations.

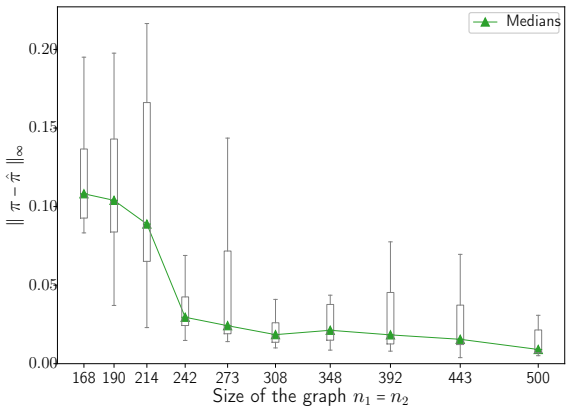


Figure 2.25: Maximum error between the true ($\boldsymbol{\pi}$) and the estimated ($\hat{\boldsymbol{\pi}}$) probabilities associated to the blocks of the data matrix $\boldsymbol{X}$ as a function of its size.

Figure 2.26 displays the mean squared error (MSE) between the generated and estimated values of the latent variables $\boldsymbol{A}$, $\boldsymbol{B}$, $\boldsymbol{C}$, $\boldsymbol{D}$ responsible for the individual variability of missingness. The estimated values are given by the maximum *a posteriori* of their corresponding variational distribution. The

MSE curves of the variables $\boldsymbol{A}$ and $\boldsymbol{C}$ are comparable as well as the curves of the variables $\boldsymbol{B}$ and $\boldsymbol{D}$. This is expected as the data matrices are generated with symmetric characters in rows and columns.

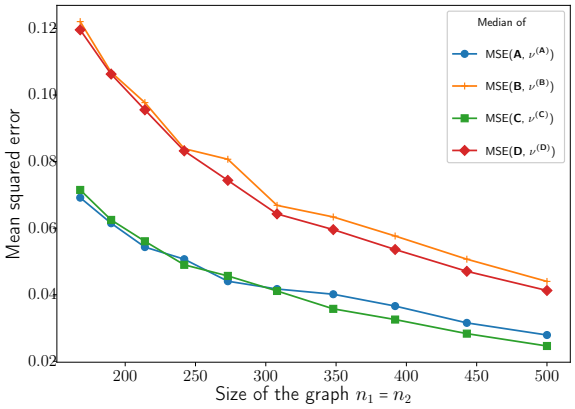


Figure 2.26: Mean squared error of the maximum *a posteriori* estimates of the latent variables $\boldsymbol{A}$, $\boldsymbol{B}$, $\boldsymbol{C}$, $\boldsymbol{D}$ governing the propensity of missingness.

Figure 2.27 compares the estimated values of $\boldsymbol{A}$, $\boldsymbol{B}$, $\boldsymbol{C}$ and $\boldsymbol{D}$ to their true generated values for two different sizes of data matrices, all other parameters being equal. A linear trend is exhibited from these scatter plots showing a good aptitude of the proposed inference to recover extreme negative and positive values.

2.A.5 Supplemental figures for the French national assembly votes analysis

Figure 2.28 displays the reordered matrix of votes derived from a block clustering with a small number of classes. Such a simplification may be helpful for identifying global trends. With this model, the three MP classes are broadly identified as gathering the right-wing (first class) and left-wing (second class) opposition parties, the last class being formed of the political groups supporting the government. The opposition systems appear clearly: on the ballots from classes A and E, the votes contrast the membership to the opposition parties versus the governmental alliance, whereas on the ballots from classes C and D, they separate the left-wing from the right-wing oppositions. Class B gathers various ballots on topics of rather general agreement pertaining to social or health matters.

Going back to the model selected by ICL described in Section 2.2.5, we analyze the resolution propensities to be voted upon and to be positively perceived by nonvoters. These propensities are encoded in the values of the latent variables $\boldsymbol{C}$ and $\boldsymbol{D}$. Figure 2.29 displays the scatter plot of $\nu_j^{(C)}$ and $\nu_j^{(D)}$, the maximum *a posteriori* estimates of C_j and D_j under the variational

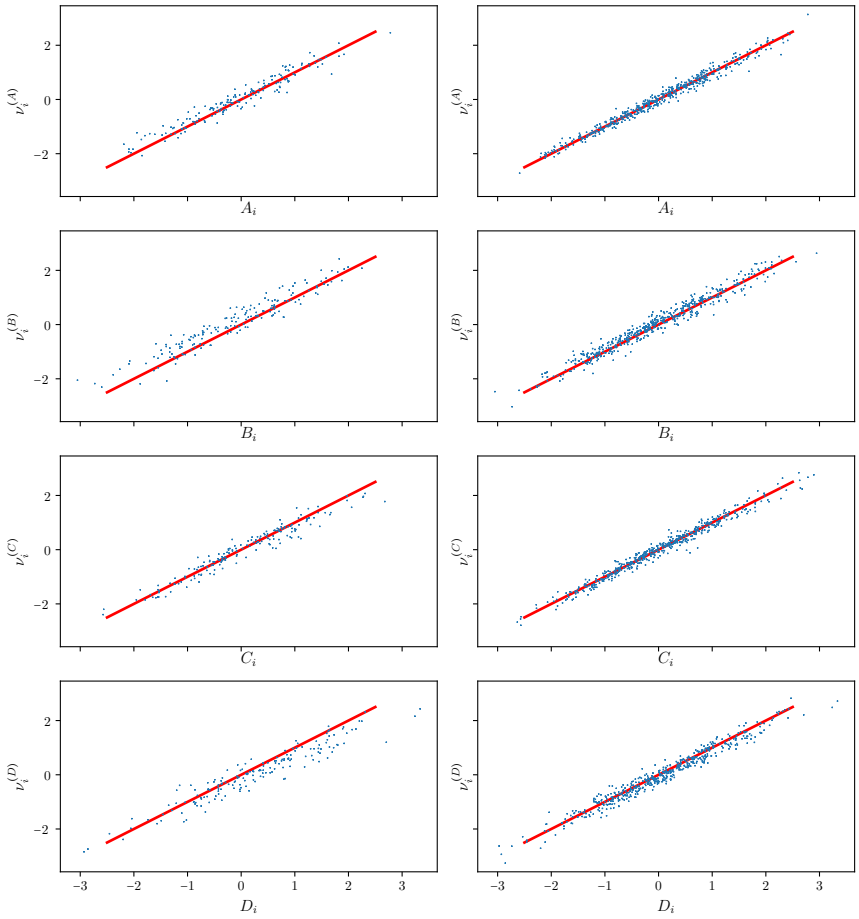


Figure 2.27: Maximum *a posteriori* estimates of the latent variables governing the propensity of missingness versus their true generated values. Left: $n_1 = n_2 = 168$ and the conditional Bayes risk is 0.44; right: $n_1 = n_2 = 500$ and the conditional Bayes risk is 0.05. The identity line is drawn in red for reference.

Figure 2.28: Left: matrix of votes reordered according to the row and column classes, for the MNAR LBM with 3 MP classes and 5 ballot classes. The red lines delineate class boundaries. The counts of MPs belonging to the three most represented political groups in each MP class is given on the left. Right: summary of the inferred opinions (expressed or not) for all classes of ballots and MPs, as given by the estimated probability to approve a resolution in each block of the reordered matrix.

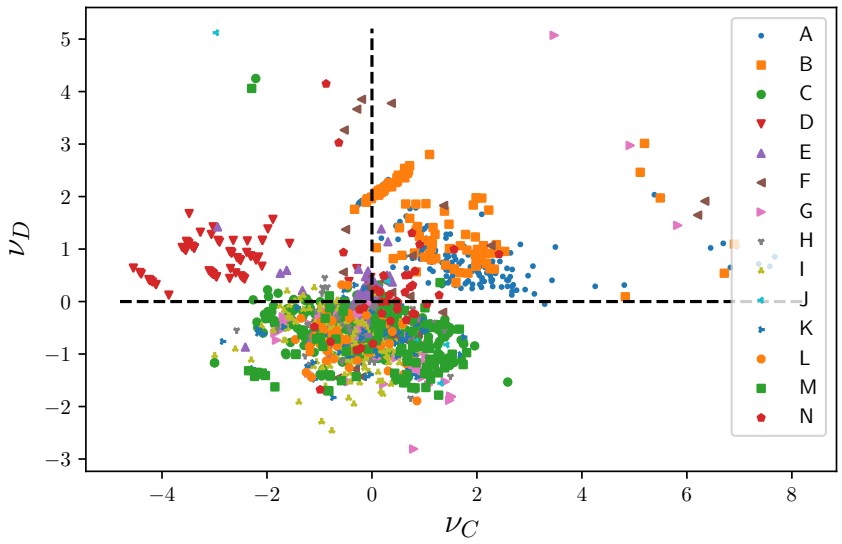


Figure 2.29: maximum *a posteriori* estimates of the resolution propensities $(\nu_j^{(C)}, \nu_j^{(D)})$, with their clustering class memberships. $\nu_j^{(C)}$ drives the MAR effect and $\nu_j^{(D)}$ drives the MNAR one.

distribution, for all ballots. The abscissa $\boldsymbol{\nu}^{(C)}$ reflects the mobilization on the ballots, with higher mobilization for higher values, and the ordinate $\boldsymbol{\nu}^{(D)}$ represents the additional effect of mobilizing specifically supporting voters. The fourteen-cluster membership of ballots (there is no obvious relevant classification for ballots) is indicated by the plotting symbol.

Some relationship between missingness and membership to ballot classes emerge from this plot. A first cluster of ballot appears in the positive quadrant, with propositions mainly proposed by the government, categorized in ballot classes A and B. A second cluster, smaller, on the upper left, is mainly formed by ballots categorized in class D, voted positively by few voters. All these propositions are related to the same law project regarding housing and were voted over a short period (06/03/2018 and 06/08/2018). The largest cluster, on the lower part of the graph, gathers most of the remaining ballots, that would have a tendency to be voted negatively by nonvoters. These propositions were proposed by either the right-wing or left-wing opposition, and get little support from a vast majority of MPs. Note also that the small group of highly voted propositions, on the right-hand side, is made of ballots belonging to six ballot classes. This reflects the fact that our model does not link the MNAR effect to the LBM memberships.

Algorithm 3: Initialization of the VEM algorithm with spectral clustering.

Input:

- observed data $\boldsymbol{X}^{(o)}$
- k_1 and k_2 number of row groups and column groups

Function `SpectralClustering`(W *adjacency matrix,* k *number of clusters*)**:**

- Define $\boldsymbol{D}$ the diagonal matrix, element of $\mathbb{R}^{n \times n}$: $D_{ii} = \sum_q W_{iq}$
- Define $\boldsymbol{L} = \boldsymbol{D}^{-1/2} \boldsymbol{W} \boldsymbol{D}^{-1/2}$
- Find the eigenvectors corresponding to the k eigenvalues of $\boldsymbol{L}$ that are largest in absolute value. Form the matrix $\boldsymbol{U} = [U_1, ..., U_k] \in \mathbb{R}^{n \times k}$ concatenating the eigenvectors into columns.

Return results of k-means with k clusters on $\boldsymbol{U}$.

begin

- Build $\boldsymbol{Y}$ the $n_1 \times k_1$ indicator matrix of the row cluster memberships with results of $SpectralClustering(\boldsymbol{X}\boldsymbol{X}^T, k_1)$
- Build $\boldsymbol{Z}$ the $n_2 \times k_2$ indicator matrix of the column cluster memberships with results of $SpectralClustering(\boldsymbol{X}^T\boldsymbol{X}, k_2)$.
- $\boldsymbol{\alpha}$, $\boldsymbol{\beta}$ and $\boldsymbol{\pi}$ estimated from $\boldsymbol{Y}$ and $\boldsymbol{Z}$
- μ initialized such as expit(μ) is the global missingness rate
- σ_A^2, σ_B^2, σ_C^2 and σ_D^2 sampled from $U_{]0,1]}$

end

Result:

- $\theta = \left(\boldsymbol{\alpha}, \boldsymbol{\beta}, \boldsymbol{\pi}, \mu, \sigma_A^2, \sigma_B^2, \sigma_C^2, \sigma_D^2\right)$ the model parameters
- $\boldsymbol{Y}$ and $\boldsymbol{Z}$ the row and column cluster memberships

2.B Appendix - MNAR LBM on sparse graphs

2.B.1 Computation of the variational log-likelihood criterion

The criterion we want to optimize is:

$$\mathcal{J}(q_\gamma, \theta) = \mathcal{H}(q_\gamma) + \mathbb{E}_{q_\gamma}\left[\mathcal{L}\left(\boldsymbol{X}^{(\mathrm{o})}, Y, Z, A, B, C, D; \theta\right)\right] \ , \tag{2.28}$$

and the details are given in Appendix 2.A. Here we are interested in the most expensive calculation that is the expectation of the conditional log-likelihood of $X_{ij}^{(\mathrm{o})}$ as it iterates on all values of the data matrix:

$$\mathbb{E}_{q_\gamma}\mathcal{L}\left(X_{ij}^{(\mathrm{o})} \middle| A, B, C, Y, Z\right) = \sum_{ijql:X_{ij}^{(\mathrm{o})}=1} \tau_{iq}^{(Y)}\tau_{jl}^{(Z)}\mathbb{E}_{q_\gamma}\left[\log\left(\pi_{ql}\operatorname{expit} P_{ij:X_{ij}=1}\right)\right] \tag{2.29}$$

$$+\sum_{ijql:X_{ij}^{(\mathrm{o})}=0} \tau_{iq}^{(Y)}\tau_{jl}^{(Z)}\mathbb{E}_{q_\gamma}\left[\log\left((1-\pi_{ql})\operatorname{expit} P_{ij:X_{ij}=0}\right)\right] \tag{2.30}$$

$$+\sum_{ijql:X_{ij}^{(\mathrm{o})}=\mathrm{NA}} \tau_{iq}^{(Y)}\tau_{jl}^{(Z)}\mathbb{E}_{q_\gamma}\left[\log\left(1-\pi_{ql}\operatorname{expit} P_{ij:X_{ij}=1}-(1-\pi_{ql})\operatorname{expit} P_{ij:X_{ij}=0}\right)\right] \ , \tag{2.31}$$

with $P_{ij:X_{ij}=0} = \mu + A_i - B_i + C_j - D_j$ and $P_{ij:X_{ij}=1} = \mu + A_i + B_i + C_j + D_j$.

To take advantage of the scarcity of the typical collaborative filtering data matrices, we propose an effective inference to scale to real data. The main idea of an effective inference is to only iterate on the observed ratings $X_{ij}^{(\mathrm{o})} = 1$ and $X_{ij}^{(\mathrm{o})} = 0$. However, in Equation (2.31) the sum is realized on all non-observed ratings. Using the two following approximations, we separate the double sum on i and j, eventually leading to Equation (2.34):

$$\begin{aligned}\operatorname{expit} x &= \exp(x) + O(\exp(2x)) && \text{with } x \to -\infty\\ &= \exp(x) + o(\exp(x)) && \text{with } x \to -\infty\end{aligned}$$

$$\begin{aligned}\log(1-x) &= -x + O\left(x^2\right) && \text{with } x \to 0\\ &= -x + o(x) && \text{with } x \to 0 \ .\end{aligned}$$

We use these approximations because the overall probability to observe a rating

is low. The term in Equation (2.31) can be re-written as:

$$\begin{aligned}&\mathbb{E}_{q_\gamma}\left[\log\left(1-\pi_{ql}\operatorname{expit}P_{ij:X_{ij}=1}-(1-\pi_{ql})\operatorname{expit}P_{ij:X_{ij}=0}\right)\right]=\\&\quad\mathbb{E}_{q_\gamma}\left[-\pi_{ql}\exp(P_{ij:X_{ij}=1})\right]+\mathbb{E}_{q_\gamma}\left[-(1-\pi_{ql})\exp(P_{ij:X_{ij}=0})\right] \qquad (2.32)\\&\quad+O(\exp(2\mu)) \qquad (\text{with } \mu\to-\infty)\end{aligned}$$

Equations (2.29), (2.30) and (2.32) involve the computation of an expectation of some nonlinear function. These four nonlinear functions are:

$$\begin{aligned}f_1(X,Y)&=\log\left(\pi_{ql}\operatorname{expit}\mu+X+Y\right)\\f_0(X,Y)&=\log\left((1-\pi_{ql})\operatorname{expit}\mu+X-Y\right)\\f_{\mathrm{NA}}(X,Y)&=\exp(\mu+X+Y)\\g_{\mathrm{NA}}(X,Y)&=\exp(\mu+X-Y)\ .\end{aligned}$$

Applying the Delta method [Wasserman, 2004, p. 79] with second order Taylor developments with X and Y independent random variables, leads to:

$$\mathbb{E}_{q_\gamma}[f(X,\,Y)]=\quad f(\mathbb{E}X,\,\mathbb{E}Y)+\tfrac{1}{2}\,var(X)\,\frac{\partial^2 f(\mathbb{E}X,\mathbb{E}Y)}{\partial(X)^2}+\tfrac{1}{2}\,var(Y)\,\frac{\partial^2 f(\mathbb{E}X,\mathbb{E}Y)}{\partial(Y)^2}\quad,\qquad(2.33)$$

which yields:

$$\begin{aligned}\forall ij: X_{ij}^{(\mathrm{o})}=1,\quad&\mathbb{E}_{q_\gamma}[f_1(A_i+C_j,\ B_i+D_j)]\approx\log\left(\pi_{ql}\cdot\operatorname{expit}\mathrm{vP}_{ij|X_{ij}=1}\right)\\&\quad-\frac{1}{2}\cdot\text{v-var}\cdot\operatorname{expit}\mathrm{vP}_{ij|X_{ij}=1}\cdot(1-\operatorname{expit}\mathrm{vP}_{ij|X_{ij}=1})\\\forall ij: X_{ij}^{(\mathrm{o})}=0,\quad&\mathbb{E}_{q_\gamma}[f_0(A_i+C_j,\ B_i+D_j)]\approx\log\left((1-\pi_{ql})\cdot\operatorname{expit}\mathrm{vP}_{ij|X_{ij}=0}\right)\\&\quad-\frac{1}{2}\cdot\text{v-var}\cdot\operatorname{expit}\mathrm{vP}_{ij|X_{ij}=0}\cdot(1-\operatorname{expit}\mathrm{vP}_{ij|X_{ij}=0})\end{aligned}$$

$$\begin{aligned}\forall ij: X_{ij}^{(\mathrm{o})}=\mathrm{NA},\quad&\mathbb{E}_{q_\gamma}[f_{\mathrm{NA}}(A_i+C_j,\ B_i+D_j)]\approx\exp(\mathrm{vP}_{ij|X_{ij}=1})\\&\quad-\frac{1}{2}\cdot\text{v-var}\cdot\exp(\mathrm{vP}_{ij|X_{ij}=1})\\\forall ij: X_{ij}^{(\mathrm{o})}=\mathrm{NA},\quad&\mathbb{E}_{q_\gamma}[g_{\mathrm{NA}}(A_i+C_j,\ B_i+D_j)]\approx\exp(\mathrm{vP}_{ij|X_{ij}=0})\\&\quad-\frac{1}{2}\cdot\text{v-var}\cdot\exp(\mathrm{vP}_{ij|X_{ij}=0})\ ,\end{aligned}$$

where the v prefix denotes quantities under variational approximation:

$$\begin{aligned}
\text{vP}_{ij|X_{ij}=1} &= \mu + \nu_i^{(A)} + \nu_i^{(B)} + \nu_j^{(C)} + \nu_j^{(D)} \\
\text{vP}_{ij|X_{ij}=0} &= \mu + \nu_i^{(A)} - \nu_i^{(B)} + \nu_j^{(C)} - \nu_j^{(D)} \\
\text{v-var} &= \rho_i^{(A)} + \rho_i^{(B)} + \rho_j^{(C)} + \rho_j^{(D)}
\end{aligned}$$

To sum up, the expectation of the observed log-likelihood (Equations (2.29), (2.30) and (2.31)), conditionally to the latent variables is:

$$\mathbb{E}_{q_\gamma} \mathcal{L}\Big(X_{ij}^{(\text{o})} \Big| A, B, C, Y, Z \Big) \approx$$

$$\sum_{qlij:X_{ij}^{(\text{o})}=+1} \tau_{iq}^{(Y)} \tau_{jl}^{(Z)} \left(\begin{array}{l}
\log\left(\pi_{ql} \cdot \text{expit}\, \text{vP}_{ij|X_{ij}=1}\right) \\
-\frac{1}{2} \cdot \text{v-var} \cdot \text{expit}\, \text{vP}_{ij|X_{ij}=1} \cdot \left(1 - \text{expit}\, \text{vP}_{ij|X_{ij}=1}\right) \\
+\pi_{ql} \cdot \exp\left(\text{vP}_{ij|X_{ij}=1}\right) + \left(1 - \pi_{ql}\right) \cdot \exp\left(\text{vP}_{ij|X_{ij}=0}\right) \\
-\frac{1}{2} \cdot \text{v-var} \cdot \left(\pi_{ql} \cdot \exp\left(\text{vP}_{ij|X_{ij}=1}\right) + \left(1 - \pi_{ql}\right) \cdot \exp\left(\text{vP}_{ij|X_{ij}=0}\right)\right)
\end{array} \right)$$

$$+ \sum_{qlij:X_{ij}^{(\text{o})}=0} \tau_{iq}^{(Y)} \tau_{jl}^{(Z)} \left(\begin{array}{l}
\log\left(\pi_{ql} \cdot \text{expit}\, \text{vP}_{ij|X_{ij}=0}\right) \\
-\frac{1}{2} \cdot \text{v-var} \cdot \text{expit}\, \text{vP}_{ij|X_{ij}=0} \cdot \left(1 - \text{expit}\, \text{vP}_{ij|X_{ij}=0}\right) \\
+\pi_{ql} \cdot \exp\left(\text{vP}_{ij|X_{ij}=1}\right) + \left(1 - \pi_{ql}\right) \cdot \exp\left(\text{vP}_{ij|X_{ij}=0}\right) \\
-\frac{1}{2} \cdot \text{v-var} \cdot \left(\pi_{ql} \cdot \exp\left(\text{vP}_{ij|X_{ij}=1}\right) + \left(1 - \pi_{ql}\right) \cdot \exp\left(\text{vP}_{ij|X_{ij}=0}\right)\right)
\end{array} \right)$$

$$- \exp\left(\mu\right) \sum_{ql} \left(\begin{array}{l}
\pi_{ql} \Big(\sum_i \left(\tau_{iq}^{(Y)} \exp\left(\nu_i^{(A)} + \nu_i^{(B)} \right) \right) \cdot \sum_j \left(\tau_{jl}^{(Z)} \exp\left(\nu_j^{(C)} + \nu_j^{(D)} \right) \right) \Big) \\
+ (1 - \pi_{ql}) \Big(\sum_i \left(\tau_{iq}^{(Y)} \exp\left(\nu_i^{(A)} - \nu_i^{(B)} \right) \right) \cdot \sum_j \left(\tau_{jl}^{(Z)} \exp\left(\nu_j^{(C)} - \nu_j^{(D)} \right) \right) \Big)
\end{array} \right) \tag{2.34}$$

Chapter 3

Fairness in recommender systems

> When you invent the ship, you also invent the shipwreck; when you invent the plane you also invent the plane crash; and when you invent electricity, you invent electrocution... Every technology carries its own negativity, which is invented at the same time as technical progress.
>
> Paul Virilio

Contents

3.1 Introduction

In simple terms, fairness is often loosely defined as the quality of treating people equally, with impartiality and rightfulness. Although imprecise, this definition stipulates that equal treatment refers to certain sensitive attributes shared by groups of people, such as gender, age, ethnicity, socio-economic group, etc. In recent years, intensive research has highlighted the lack of fairness in decisions made by machine learning algorithms [Binns, 2018].

There are several stakeholders in a recommendation scenario. In the terminology of Burke et al. [2018], we target consumer-fairness, where the objective is to provide the same treatment to users of the recommender system, regardless of their sensitive attribute. We target recommender systems relying on collaborative filtering, which aims at building recommendations from the history of user ratings. These observed ratings are the basis for making automatic predictions about non-rated items, under the assumption that users can be clustered according to their past opinion behavior. Although sensitive attributes are not used to fit the models, some disparate treatments may nevertheless exist, possibly due to some societal or cultural effects that bias the sampling of data [Daymonti and Andrisani, 1984]. When the sensitive attribute is observed, its processing can help mitigate these discriminatory effects.

Several proposals have already been made on how fairness should be formally defined in collaborative filtering [Gajane, 2017, Räz, 2021]. One common approach is the recommendation independence [Kamishima et al., 2018], that requires the unconditional statistical independence between recommendations and a specified sensitive attribute s_i of user i:

$$p(X_{ij} > X_{ij'}|s_i = -1) = p(X_{ij} > X_{ij'}|s_i = 1) \ ,$$

with $X_{ij} > X_{ij'}$ when item j is preferred to item j' for user i. Assuming the access to the value of the sensitive attribute s_i of all users, we give the following definition of a fair recommender system:

Definition 1 (ε-fair recommendation, binary sensitive attribute) *A recommender system is said to be ε-fair with respect to attribute s if for any two items j and j':*

$$\left| \frac{\#\{i|s_i = 1 \wedge (X_{ij} > X_{ij'})\}}{\#\{i|s_i = 1\}} - \frac{\#\{i|s_i = -1 \wedge (X_{ij} > X_{ij'})\}}{\#\{i|s_i = -1\}} \right| \leq \varepsilon \ , \qquad (3.1)$$

where $\varepsilon \in \mathbb{R}_+$ measures the gap to exact fairness

In essence, an ε-fair recommender system ensures that, for any two items, the proportion of users with the same preference is approximately identical in all the subpopulations of users defined by identical sensitive attributes. This equal treatment does not ensure equal impact, which argues for equal

recommendation quality between sensitive groups. Although some works [Yao and Huang, 2017] have argued that recommendation independence may be overly restrictive, resulting in a poor quality of recommendations, we use here this definition to propose fair recommendations exempted from any stereotypes.

3.1.1 Fairness in collaborative filtering methods

In this section, we review the recent works that have raised the issue of fairness in recommender systems.

Independence-enhanced recommendation in PMF: Kamishima et al. [2018] have proposed methods for improving fairness, formalized as the independence of the predicted ratings with the sensitive attribute. Their methods are based on probabilistic matrix factorization (see Section 1.1.3), modified so that the ratings are dependent to the sensitive attribute:

$$\hat{X}_{ij} = \boldsymbol{Y}_i^{s_i T} \boldsymbol{Z}_j^{s_i} + b_i^{s_i} + b_j^{s_i} + \mu^{s_i} \quad . \tag{3.2}$$

They minimize the quadratic error between the true ratings X_{ij} and the predicted ratings $\hat{X}_{ij}$ regularized by criteria to enhance the recommendation independence. By controlling the moments of the distributions of rating among sensitive groups the regularization term favors independence. The first independence term they proposed is the *mean matching* designed so as to match the means of the distributions $p(\boldsymbol{X}|\boldsymbol{s}=0) = p(\boldsymbol{X}|\boldsymbol{s}=1)$. To deal with the second moment of distributions in addition to the first moment, they proposed an alternative independence term using the *negative Bhattacharyya distance*:

$$\log \int \sqrt{p(\boldsymbol{X}|\boldsymbol{s}=0)p(\boldsymbol{X}|\boldsymbol{s}=1)}\mathrm{d}\boldsymbol{X}$$

Finally to extend to the case where the sensitive feature is categorical type, they employed a third independence term based on the *negative mutual information*:

$$-I(\boldsymbol{X};\boldsymbol{s}) = -(\mathcal{H}(\boldsymbol{X}) - \mathcal{H}(\boldsymbol{X}|\boldsymbol{s})) \quad ,$$

where $\mathcal{H}$ stands for the differential entropy.

The dependency on the sensitive attribute imposed to the rating prediction implies that the number of latent factors needed to model users and items, is multiplied by the number of values the sensitive attribute s can takes. Thus the number of values the sensitive attribute can take is rather limited.

Fairness-Aware Tensor-Based Recommendation: Tensor-based methods extend the matrix factorization framework to n-dimensional arrays with the aim of modelling multi-aspect interactions. For example, between users, items, and time of day.

Using the same definition of fairness as Kamishima et al. [2018], Zhu et al. [2018] proposed a tensor method that isolates the impact of the sensitive attribute by adding dedicated dimensions to the latent factor matrix. For example and as illustrated by Figure 3.1, if one of the sensitive attributes is the binary gender, two dimensions encoding for the sensitive value are added to the user latent factor matrix (upper right). These two sensitive dimensions are kept constant while inferring the others.

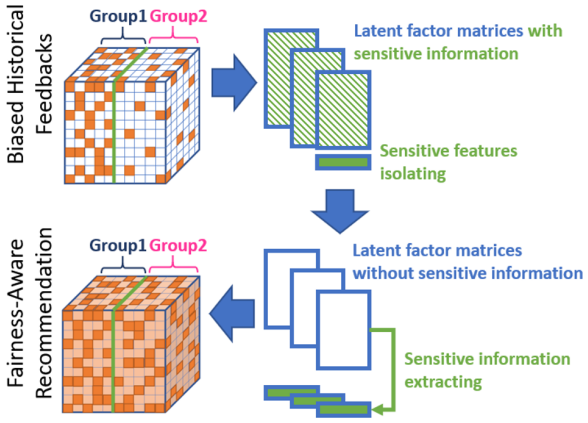


Figure 3.1: Illustration of Fairness-Aware Tensor-Based Recommendation (FATR) from Zhu et al. [2018]: sensitive features are isolated (top right), then sensitive information is extracted (bottom right), resulting in fairness-aware recommendation.

As there may still be sensitive information that resides in non-sensitive dimensions, the authors maintain the additional constraint of orthogonality between sensitive and non-sensitive dimensions adding to the objective function the term $\lambda \| \boldsymbol{s}^T \boldsymbol{Y}^k \|_{\mathcal{F}}^2$ where $\boldsymbol{Y}^k$ is the k^{th} factor matrix of the user latent tensor and λ a trade off parameter.

The fair ratings are predicted using the remaining non-sensitive dimensions only. Unlike many other methods, this solution is capable of handling multiple and non-binary sensitive attributes.

Other fairness objectives beyond parity: Yao and Huang [2017] proposed four new metrics that deal with different types of unfairness and used them as penalty functions in augmented matrix factorization objectives.

The first two metrics are based on the average reconstruction errors between two groups of users distinguished by their binary protected attribute. They defined the *value unfairness* as the signed error averaged across the groups of users. A high value unfairness occurs when one class of user is consistently given higher or lower predictions than their true preferences while predictions for other classes are consistently better estimated. However if the errors in prediction are evenly balanced between overestimation and underestimation

the value unfairness becomes small. Thus, the authors also proposed the *absolute unfairness* being the unsigned error averaged across the groups of users.

The last two metrics are the *underestimation unfairness* and the *overestimation unfairness* which measure inconsistency in how much the predictions respectively underestimate and overestimate the true ratings. For example, in a course recommender, *underestimation unfairness* could lead to a top student not being recommended to explore a topic they would excel in. In the opposite, *overestimation unfairness* would lead to a situation where the user may be so overwhelmed by recommendations that it would become detrimental.

Pairwise comparisons: All of the above methods are based on the fairness of predicted ratings, but an approximate fairness of ratings may not entail an approximate fairness of the recommender system that provides users with a short list of relevant items. With this in mind, Beutel et al. [2019] provided new metrics based on pairwise comparisons to aim equalized odds between items and proposed a novel pairwise regularization approach to improve the fairness of the recommender system during training. Equalized odds also called the equality of opportunity is an other common definition of a fair recommendation, less restrictive that the independence of the predicted ratings with the sensitive attribute. In this setting a classifier is said to be fair if the items from different sensitive groups have equal probabilities to be recommended:

$$p(X_{ij} = 1 | Y_i = y, Z_j = z, s_j = 0) = p(X_{ij} = 1 | Y_i = y, Z_j = z, s_j = 1) \ ,$$

conditionally to Y_i and Z_j attributes characterizing user i and item j.

The authors extended this definition to a *pairwise fairness* that considers a system to be fair if the likelihood of a clicked item being ranked above another relevant unclicked item is the same across all sensitive groups, conditioned to the fact that items have been engaged with the same amount.

Considering a pairwise comparison only may be problematic as it does not distinguish situations where the system would systematically recommend items from one sensitive group before the others, independently of user preferences. Thus, the authors proposed the *intra-group pairwise accuracy* and the *inter-group pairwise accuracy*, two criterion measuring the pairwise accuracy between items respectively from the same sensitive group and from different sensitive groups.

Statistical parity in clustering: Finally, further from recommender systems but still related to the contribution proposed in the next section, the notion of statistical parity is often considered for fairness in clustering methods [Abbasi et al., 2021, Ghadiri et al., 2020, Bera et al., 2019].

3.1.2 Our contribution

In the following section, we propose to incorporate the exogenous sensitive attributes in the latent block model to ensure fair recommendations. Introducing the sensitive attribute in the latent block model leads to a classification of users that is independent from the sensitive attribute. Since users are only characterized by their ratings and their sensitive attribute, fairness is measured here by a parity criterion. We propose a definition of fairness for the recommender system that expresses that the ranking of items should be independent of the sensitive attribute. We show that our model ensures approximately fair recommendations provided that the classification of users approximately respects statistical parity.

3.2 Co-clustering for fair recommendation

This Section is available as a preprint on HAL Frisch et al. [2021].

In this Section, we aim at producing fair recommendations using a co-clustering of users and items that respects statistical parity of users with respect to some sensitive attributes. For this purpose, we introduce a co-clustering model based on the Latent Block Model (LBM) that relies on an ordinal regression model that takes as inputs the sensitive attributes. We demonstrate that our model ensures approximately fair recommendations provided that the clustering of users approximately respects statistical parity. Finally, we conduct experiments on a real-world dataset to show that the proposed approach can help alleviate unfairness.

3.2.1 Model proposed

The data used to build recommender systems can be aggregated in a matrix where rows are users, columns are items and entries the feedbacks. The model we propose is based on the Latent Block Model that considers a data matrix to group users and items based on their opinions.

The user feedback used for collaborative filtering can be implicit (history, browsing history, clicks...) or explicit. In the case of explicit evaluation data, users most often express their interest in items using a discrete rating scale. This rating scale suppose an order between levels, for example from 1 to 5 expressing the worst opinion to the best one. Models handling this type of data can assume that these scales are a discretization of the opinion of a user that may be better handled by a continuous variable. The method we propose to model ratings is based on a statistical co-clustering using ordered probit regression to model ordinal responses. Covariates encoding a sensitive user attribute can easily be included in the probit regression framework.

Ordered probit in Latent Block Model The ordered probit model [Daykin and Moffatt, 2002] assumes the existence of a continuous, Gaussian distributed latent random variable, denoted $\boldsymbol{X}^\star$. In a collaborative filtering context, this latent variable represents the underlying value, assumed to be continuous, assigned to an item by the user. The assumption of a single underlying continuous variable leading to ordinal ratings may be appropriate when ratings are not the result of a sequential process [Bürkner and Vuorre, 2019]. The discrete observed ratings $\boldsymbol{X}$ are the result of the partition of the continuous space of $\boldsymbol{X}^\star$ by a set of thresholds $\boldsymbol{\zeta}$ such that: $X_{ij} = 1$ if $-\infty < X^*_{ij} < \zeta_1$, $X_{ij} = 2$ if $\zeta_1 < X^*_{ij} < \zeta_2$, $\ldots$, $X_{ij} = K$ if $\zeta_{K-1} < X^*_{ij} < +\infty$ (see Figure 3.2).

We use the ordered probit model within a Latent Block Model (see Section 1.2.2), assuming that conditionally to row and column group assignments,

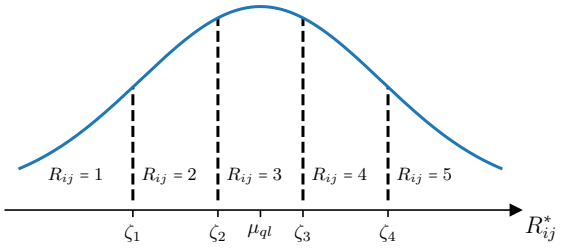


Figure 3.2: The conditional density function of X^*_{ij} and its relationship to X_{ij}. Fixed thresholds ζ_k, defines the discretization of X^*_{ij}.

the entries of $\boldsymbol{X}^\star$ are independent and identically distributed with Gaussian distribution:

$$p(\, X^*_{ij} \big| Y_{iq} Z_{jl} = 1; \pi_{ql}, \sigma\,) = \phi\big(X^*_{ij}; \pi_{ql}, \sigma^2\big) \;,\; \pi_{ql} \in \mathbb{R} \text{ and } \sigma \in \mathbb{R}^*_+ \tag{3.3}$$

with $\phi\big(\cdot; \pi_{ql}, \sigma^2\big)$ the probability density function of the Gaussian distribution with mean π_{ql} and variance σ^2. The conditional probability that a user i gives to the item j the rating with value k is then:

$$\begin{aligned} p(\, X_{ij} = k \big| Y_{iq} Z_{jl} = 1; \pi_{ql}\,) &= p(\, \zeta_{k-1} < X^*_{ij} < \zeta_k \big| Y_{iq} Z_{jl} = 1; \pi_{ql}\,) \\ &= \Phi\big(\zeta_k; \pi_{ql}, \sigma^2\big) - \Phi\big(\zeta_{k-1}; \pi_{ql}, \sigma^2\big) \;, \end{aligned}$$

with $\Phi\big(\cdot; \pi_{ql}, \sigma^2\big)$ being the normal cumulative distribution function. To ensure model identifiability, the thresholds $\boldsymbol{\zeta}$ are fixed to equidistant predefined values.

Individual row and column effects The Latent Block Model is well suited to collaborative filtering, in that it searches for users and items that share the same opinion patterns. However, a model that assumes that users in a given cluster share exactly the same opinion patterns is very restrictive. Instead, we assume here that opinions may be slightly different within a cluster, using a richer model than Equation (3.3) for the conditional distribution of X^*_{ij}. In addition to the cluster effect π_{ql} derived solely from the group memberships of users and items, one deviation is induced by the user i and another by the item j :

$$p(\, X^*_{ij} \big| Y_{iq} Z_{jl} = 1, A_i, B_j; \pi_{ql}\,) = \phi\big(X^*_{ij}; \pi_{ql} + A_i + B_j, \sigma^2\big) \;, \tag{3.4}$$

with latent variables $\boldsymbol{A}$ and $\boldsymbol{B}$ independently and identically distributed with:

$$\begin{aligned} A_i &\overset{\text{iid}}{\sim} \mathcal{N}\big(0, \sigma^2_A\big), \qquad \sigma^2_A \in \mathbb{R}^*_+ \\ B_i &\overset{\text{iid}}{\sim} \mathcal{N}\big(0, \sigma^2_B\big), \qquad \sigma^2_B \in \mathbb{R}^*_+ \end{aligned}$$

These two variables encode different rating patterns for users and items such as systematic over- or under-rating relative to the user or item populations.

Sensitive attribute We assume that, in addition to the matrix of ratings, we have access to a sensitive attribute s_i, describing here a binary feature of user i that should not intervene in the recommendation of items (more general sensitive attributes are considered in Appendix 3.A). We introduce a latent variable C_j for each object j assuming that they interact with different strengths with the sensitive attribute. This interaction between the object j and the sensitive attribute s_i is added to the conditional distribution of X^*_{ij} (Equation 3.4):

$$p(X^*_{ij} | Y_{iq} Z_{jl} = 1, A_i, B_j, s_i, C_j; \pi_{ql}) = \phi\big(X^*_{ij}; \pi_{ql} + A_i + B_j + s_i C_j, \sigma^2\big) \ ,$$

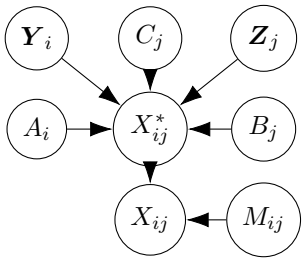
with

$$C_j \overset{\text{iid}}{\sim} \mathcal{N}\big(0, \sigma_C^2\big), \ \ \sigma_G^2 \in \mathbb{R}_+^* \ .$$

This model explains the ratings by $\pi_{ql} + A_i + B_j + s_i C_j$ and σ^2; the co-clustering is driven by π_{ql}, and provided the effects of the sensitive attribute are well captured by $s_i C_j$, we expect the co-clustering to be independent of the sensitive attribute, which ensures fair recommendations as shown in Section 3.2.2.2. A summary of the model we propose is presented in Figure 3.3.

Modelling missingness The datasets extracted from recommender systems are usually extremely sparse, with a high proportion of missing ratings, that is, ratings that were not provided by the users. The model we proposed so far does not accommodate missing observations, and suppose a fully observed data matrix $\boldsymbol{X}$.

The study of missing data identifies three main type of missingness [Rubin, 1976]: Missing Completely At Random (MCAR) and Missing At Random (MAR) referring to the mechanisms in which the probability of being missing does not depend on the variable of interest (here $\boldsymbol{X}^\star$); and finally Missing Not At Random (MNAR) referring to the mechanisms in which the probability of being missing depends on the actual value of the missing data. A common implicit assumption in collaborative filtering is that ratings are MAR or MCAR: the presence/absence of ratings is assumed to convey no information whatsoever about the value of these ratings. For simplicity of statistical modelling we take the same assumption, although previous studies [Marlin et al., 2007, 2012] have shown a potential dependence between the presence of ratings and the underlying opinion. We introduce a simple Bernoulli missingness model generating $\boldsymbol{M} \in \{0,1\}^{n_1 \times n_2}$, a mask matrix where each entry M_{ij} is one with probability p and indicates whether the rating is observed: $M_{ij} = 1$ if X_{ij} is observed and 0 otherwise. Given the complete data matrix $\boldsymbol{X}^\star$ and the mask



$$
\begin{aligned}
\boldsymbol{Y}_i &\overset{\text{iid}}{\sim} \mathcal{M}(1;\boldsymbol{\alpha}), && \boldsymbol{\alpha} \in \mathbf{S}_{k_1-1} \\
\boldsymbol{Z}_j &\overset{\text{iid}}{\sim} \mathcal{M}(1;\boldsymbol{\beta}), && \boldsymbol{\beta} \in \mathbf{S}_{k_2-1} \\
A_i &\overset{\text{iid}}{\sim} \mathcal{N}\left(0,\sigma_A^2\right), && \sigma_A^2 \in \mathbb{R}_+^* \\
B_j &\overset{\text{iid}}{\sim} \mathcal{N}\left(0,\sigma_B^2\right), && \sigma_B^2 \in \mathbb{R}_+^* \\
C_j &\overset{\text{iid}}{\sim} \mathcal{N}\left(0,\sigma_C^2\right), && \sigma_C^2 \in \mathbb{R}_+^*
\end{aligned}
$$

$$
\left(X^*_{ij} \middle| Y_{iq}=1, Z_{jl}=1, A_i, B_j\right) \overset{\text{ind}}{\sim} \mathcal{N}\left(\pi_{ql} + A_i + B_j + s_i C_j, \sigma^2\right)
$$

$$
\left(X_{ij} \middle| X^*_{ij}, M_{ij}\right) = \begin{cases} \sum_{k=1}^{K} k\, \mathbb{1}_{]\theta_{k-1};\theta_k]}(X^*_{ij}) & \text{if} \quad M_{ij}=1 \\ \text{NA} & \text{if} \quad M_{ij}=0 \end{cases}
$$

with $M_{ij} \overset{\text{iid}}{\sim} \mathcal{B}(p), \qquad p \in [0,1]$

and $\zeta_0 = -\infty < \zeta_1 < \ldots < \zeta_{K-1} < \zeta_K = \infty$, fixed thresholds

Figure 3.3: Graphical view and summary of the ordered probit Latent Block Model with protected attribute $\boldsymbol{s}$. The discrete observed data X_{ij} is generated by the underlying continuous data X^*_{ij} and the mask entry M_{ij}.

matrix $\boldsymbol{M}$, the elements of the observed ratings $\boldsymbol{X}$ are generated as follows:

$$
\left(X_{ij} \middle| X^*_{ij}, M_{ij}\right) = \begin{cases} \sum_{k=1}^{K} k\, \mathbb{1}_{]\zeta_{k-1};\zeta_k]}(X^*_{ij}) & \text{if} \quad M_{ij}=1 \\ \text{NA} & \text{if} \quad M_{ij}=0 \end{cases}
$$

Any generative model under a MCAR or MAR process can be fitted separately from the missingness model as the overall likelihood can be factorized between the observed and non observed data. Under such assumptions, we show in Appendix3.A that ignoring non-observed ratings results in a proper fitting.

3.2.2 Inference and fair recommendations

3.2.2.1 A stochastic batch gradient descent of the variational criterion

The log-likelihood of the model is not tractable as it involves a sum that is combinatorially too large [Brault and Mariadassou, 2015]. We resort to a variational inference procedure [Jaakkola, 2000] that introduces q_γ, a restricted parametric inference distribution defined on the latent variables of the model, to optimize the following lower bound on the log-likelihood:

$$
\mathcal{J}(\gamma,\theta) = \log p(\boldsymbol{X};\theta) - \mathrm{KL}\left(q_\gamma \,\|p(L|\boldsymbol{X})\right)
$$

where KL stands for the Kullback-Leibler divergence, $\mathcal{H}$ for the differential entropy, $\theta = (\boldsymbol{\alpha}, \boldsymbol{\beta}, \boldsymbol{\pi}, \sigma^2, \sigma_A^2, \sigma_B^2, \sigma_C^2, p)$ is the concatenation of the model pa-

rameters, and $L = (\boldsymbol{Y}, \boldsymbol{Z}, \boldsymbol{A}, \boldsymbol{B}, \boldsymbol{C})$ is the concatenation of the latent variables.

The variational distribution q_γ is chosen so that the computation of the criterion becomes easier:

$$\begin{aligned}
&\forall i, \quad \boldsymbol{Y}_i|\boldsymbol{X} \underset{q_\gamma}{\sim} \mathcal{M}\left(1; \boldsymbol{\tau}_i^{(Y)}\right) && \forall j, \quad \boldsymbol{Z}_j|\boldsymbol{X} \underset{q_\gamma}{\sim} \mathcal{M}\left(1; \boldsymbol{\tau}_j^{(Z)}\right) \\
&\forall i, \quad A_i|\boldsymbol{X} \underset{q_\gamma}{\sim} \mathcal{N}\left(\nu_i^{(A)}, \rho_i^{(A)}\right) && \forall j, \quad B_j|\boldsymbol{X} \underset{q_\gamma}{\sim} \mathcal{N}\left(\nu_j^{(B)}, \rho_j^{(B)}\right) \\
& && \forall j, \quad C_j|\boldsymbol{X} \underset{q_\gamma}{\sim} \mathcal{N}\left(\nu_j^{(C)}, \rho_j^{(C)}\right) .
\end{aligned}$$

We also enforce the conditional independence of the latent variables, leading to the following fully factorized form:

$$\begin{aligned}
q_\gamma = &\prod_{i=1}^{n_1} \mathcal{M}\left(1; \boldsymbol{\tau}_i^{(Y)}\right) \times \prod_{j=1}^{n_2} \mathcal{M}\left(1; \boldsymbol{\tau}_j^{(Z)}\right) \\
&\times \prod_{i=1}^{n_1} \mathcal{N}\left(\nu_i^{(A)}, \rho_i^{(A)}\right) \times \prod_{j=1}^{n_2} \mathcal{N}\left(\nu_j^{(B)}, \rho_j^{(B)}\right) \\
&\times \prod_{j=1}^{n_2} \mathcal{N}\left(\nu_j^{(C)}, \rho_j^{(C)}\right) ,
\end{aligned} \tag{3.5}$$

where γ denotes the concatenation of all parameters of the variational distribution[1]. This conditional independence of the latent variables to $\boldsymbol{X}$ simplifies the criterion $\mathcal{J}(\gamma, \theta)$ to:

$$\mathcal{J}(\gamma, \theta) = \mathbb{E}_{q_\gamma}[\log p(\boldsymbol{X}|L)] - \mathrm{KL}\left(q_\gamma \| p(L; \theta)\right) . \tag{3.6}$$

As explained in Section 3.2.1, the optimization criterion relies only on the non-missing entries of $\boldsymbol{X}$ because the data is assumed to be missing at random. The full expansion of the criterion is given in Appendix 3.A.

We resort to a batch stochastic optimization to maximize the variational criterion using noisy estimates of its gradient [Ranganath et al., 2014]. Samples are drawn from the variational distribution (Equation 3.5) to estimate a noisy but unbiased gradient of the expectation of the conditional log-distribution of $\boldsymbol{X}$ (first term of Equation 3.6), which we then use to update our parameters as follows:

$$(\gamma, \theta)^{(t+1)} = (\gamma, \theta)^{(t)} + \eta \cdot \nabla_{(\gamma,\theta)} \mathcal{J}\left(\boldsymbol{X}_{(i:i+n),(j:j+n)}; \gamma, \theta\right) ,$$

where n is the batch size and η is the adaptive learning rate based on the past gradients that were computed (Adam optimizer Kingma and Ba [2014]).

Using a stochastic gradient algorithm instead of the usual EM algorithm alleviates the well-known initialization problems of the Latent Block Model, which result in unsatisfactory local maxima [Biernacki et al., 2003, Baudry and Celeux, 2015]. However, it requires the use of differentiable functions to back-propagate gradients through the automatic differentiation graph [Paszke et al., 2019]. For this purpose, the multinomial distributions are replaced by a

[1] $\gamma = (\boldsymbol{\tau}^{(Y)}, \boldsymbol{\tau}^{(Z)}, \boldsymbol{\nu}^{(A)}, \boldsymbol{\rho}^{(A)}, \boldsymbol{\nu}^{(B)}, \boldsymbol{\rho}^{(B)}, \boldsymbol{\nu}^{(C)}, \boldsymbol{\rho}^{(C)})$

differentiable Gumbel-Softmax distribution [Jang et al., 2016].

3.2.2.2 Fair recommendations

This section describes a theoretical result establishing a guarantee on the fairness of recommendations. This guarantee is subject to an assumption about the parity of the clustering of users that can be tested in practice, and that holds true for the experiments reported in Section 3.2.3 and Appendix 3.A. We develop here the case of a binary sensitive attribute to simplify the exposition. The result is more general and applies to any discrete sensitive attribute. It is proven in this general sense in Appendix 3.A.

Recommendations are partial orders between items. In collaborative filtering, the usual approach to producing recommendations is to estimate a relevance score for each item, which is then used to define a total order through numerical comparisons. With the parameters obtained by variational inference, we define the relevance score of item j for user i as:

$$\hat{X}_{ij} = \boldsymbol{\tau}_i^{(Y)} \hat{\boldsymbol{\pi}} \boldsymbol{\tau}_j^{(Z)^T} + \nu_i^{(A)} + \nu_j^{(B)} \quad . \tag{3.7}$$

This relevance score is computed from the maxima *a posteriori* of the latent variables encoding the user and item group memberships $(\boldsymbol{\tau}_i^{(Y)}, \boldsymbol{\tau}_j^{(Z)})$, that is, the trend related to the co-cluster to which (i, j) belongs, and the global effects related to user i and item j. It does not use the user's sensitive attribute s_i which is considered here as a nuisance parameter, properly taken into account during inference and then ignored when predicting a relevance score. It then becomes possible to compare items fairly with respect to the sensitive attribute.

Definition 2 (Fair comparison of items) *Given user i and any two items j and j', the comparison of items j and j' is said to be fair if it is freed from the evaluation bias regarding the sensitive attribute s: item j is fairly preferred to item j' if $\hat{X}_{ij} > \hat{X}_{ij'}$, that is:*

$$\boldsymbol{\tau}_i^{(Y)} \hat{\boldsymbol{\pi}} \boldsymbol{\tau}_j^{(Z)^T} + \nu_i^{(A)} + \nu_j^{(B)} > \boldsymbol{\tau}_i^{(Y)} \hat{\boldsymbol{\pi}} \boldsymbol{\tau}_{j'}^{(Z)^T} + \nu_i^{(A)} + \nu_{j'}^{(B)} \quad .$$

The modelling of the observed data $\boldsymbol{X}$ incorporates the term $\nu_j^{(B)} s_i$, interpreted here as a spurious opinion bias related to the sensitive attribute. While it is important to ignore this term for a fair comparison of items, its inclusion into the model is important to allow the construction of clusters that are not affected by this spurious effect. These clusters can then be expected to be representative of all subpopulations defined by their sensitive attribute value, and thus to respect the statistical parity of users.

Definition 3 (Clustering ε-parity, binary sensitive attribute) *The*

clustering of users is said to respect ε-parity with respect to attribute s iff:

$$\forall q, \quad \left| \frac{\#\{i|s_i = 1 \wedge u_{iq} = 1\}}{\#\{i|s_i = 1\}} - \frac{\#\{i|s_i = -1 \wedge u_{iq} = 1\}}{\#\{i|s_i = -1\}} \right| \leq \varepsilon \ , \tag{3.8}$$

where $\varepsilon \in \mathbb{R}_+$ measures the gap to exact parity, u_{iq} is the (hard) membership of user i to cluster q,and $\#\{i|\Omega\}$ is the number of users defined by the cardinality of the set Ω.

In essence, clustering ε-parity requires that subpopulations of users defined by identical sensitive attributes be represented approximately equally in each user group. For the Latent Block Model, the hard membership u_{iq} of Definition 3 is given by the maximum *a posteriori* of the latent variable $\tau_{iq}^{(Y)}$.

Our theoretical guarantee ensures that this approximate statistical parity in clusters is sufficient to get approximately fair recommendations (Definition 1) from our model:

Theorem 1 (Fair recommendation from clustering parity) *If the clustering of users in k_1 groups respects ε-parity (Definition 3 or Definition 4) then the recommender system relying on the relevance score defined in Equation (3.7) is $(k_1\varepsilon)$-fair (Definition 1 or Definition 5).*

Proof: see Appendix 3.A.

3.2.3 Experiment on MovieLens dataset

The final goal of a recommender system is to provide users with a shortlist of items that they might most enjoy. We choose here to directly assess the quality of the ranking rather than using proxy measures, such as root mean square error on ratings, that ignore relative rankings.

To measure the ranking performance of algorithms, we use the Normalized Discounted Cumulative Gain [Järvelin and Kekäläinen, 2017] (NDCG) that measures ranking quality by a penalized sum of the relevance scores of the ranking results:

$$NDCG@k = \frac{DCG@k}{IDCG@k} \text{ with } DCG@k = \sum_{i=1}^{k} \frac{rel_i}{\log(i+1)} \ ,$$

rel_i, the relevance of the results at each rank i before k and $IDCG@k$ being the $DCG@k$ computed with a perfect ranking.

We use the MovieLens 1M dataset [movie lens] that contains one million ratings given by 6,040 users to 3,900 movies scaling from 1 to 5 (from least liked to most liked). The dataset also contains additional information about users: gender (binary), age category (seven levels) or occupation. We give here some experimental results where gender is the sensitive attribute, and additional results, in particular with age considered as the sensitive attribute, can be found in Appendix 3.A.

3.2.3.1 Experimental Protocol

We estimate the average performances by predicting preferences on ratings that are concealed during training. These concealed ratings form our test set, with 20 ratings per user, which is about 10% of the available data. This process is repeated 5 times, with independent random draws, to produce stable average performances.

We compare our model (referred to as Parity LBM) with the baseline LBM that does not use the sensitive variable in the modelling (referred to as Standard LBM). We expect the latter model to create groups of users that do not respect clustering parity and to generate unfair recommendations. We also compare to another co-clustering algorithm, weighted Bregman co-clustering [George and Merugu, 2005, Banerjee et al., 2007] (referred to as Bregman co-clust) to compare the statistical parity of user groups inferred from another baseline. Finally, we compare with Singular Value Decomposition (SVD), a method popularized during the Netflix challenge [Koren et al., 2009] that still remains state of the art in collaborative filtering [Rendle et al., 2019]. All these baselines are implemented in the Python module `Surprise` [Hug, 2020].

The number of clusters in co-clustering and the number of factors in matrix factorization are both arbitrarily set to fifteen. Another comparison with more clusters, provided in Appendix 3.A, produces qualitatively similar results.

We repeat the learning process 25 times from different random initializations to mitigate the initialization dependence that affects all optimization procedures. We select the best solution based on the optimization criteria, that is, the one with the highest likelihood for the LBM models and the lowest training reconstruction error for the other baselines.

3.2.3.2 Results and Discussion

Gender as sensitive attribute User gender (binary in this dataset) is used as the sensitive attribute s_i. In the dataset, 27% of users self-identified as females, this proportion must be met in each group to respect clustering parity. To measure the dependence between gender and user group memberships, we compute the χ^2 statistic constructed from the contingency table of males and females counts in each group. Table 3.1 reports the p-value for testing the independence between groups and genders, with an asymptotical test. We recall that, under the null hypothesis of independence, the test statistic with k degrees of freedom has mean k and variance $2k$. The results show that the methods that do not consider the sensitive variable in the modelling create groups that are dependent on gender. In contrast, our Parity-LBM model is consistent with the clustering parity hypothesis: the gender representation in groups is representative of the gender distribution in the overall dataset.

The fairness of recommendations resulting from this clustering parity is

Table 3.1: Measures of statistical gender parity among user clusters. The number of user groups is $k_1 = 15$. The χ^2 statistic (with 14 degrees of freedom) is averaged over the five replicates of the experiment. A high value of the χ^2 statistic (or a low p-value) leads to the rejection of the clustering parity hypothesis.

Model	Parity LBM	Standard LBM	Bregman co-clust
χ^2 statistic	18.0	44.4	187
p-value	0.20	$5.1 \cdot 10^{-5}$	$< 10^{-15}$.

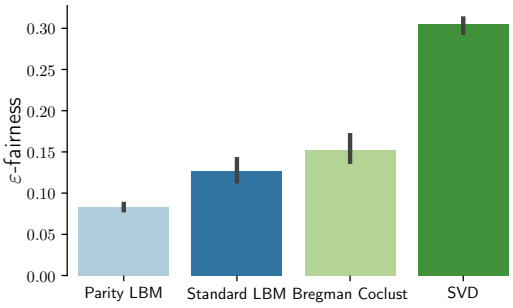


Figure 3.4: Gaps ε for the ε-fair recommendations (see Definition 1) provided by each model: a smaller ε-fairness indicates fairer recommendations.

ascertained by computing the gap ε from exactly fair recommendations, as defined in Definition 1. Figure 3.4 displays these gaps, with lower values indicating a fairer recommendation; our model provides a significantly fairer recommendation compared to the standard Latent Block Model, which is itself much fairer than the two other baselines. The order observed in Table 3.1 is followed.

Figure 3.5 depicts the ranking performance of algorithms with the NDCG, averaged over all users, for a recommendation list of 10 items. SVD gets the best overall result, followed by the Latent Block Models that outperform Bregman co-clustering. The overall performances of our model and the standard LBM are not significantly different. Figure 3.5 also reports the average NDCG within each sensitive group. This performance measure shows that female users receive significantly less relevant recommendations than males with all algorithms. This disparate treatment can be related to equalized odds [Hardt et al., 2016] in the classification framework, in that it focuses on truly relevant recommendations. The performance gap between the sensitive groups is reduced by our parity LBM compared to the standard LBM. Although the difference is the smallest among all comparisons, our model does not eliminate disparate treatment. As a cautionary note, although it is likely that the

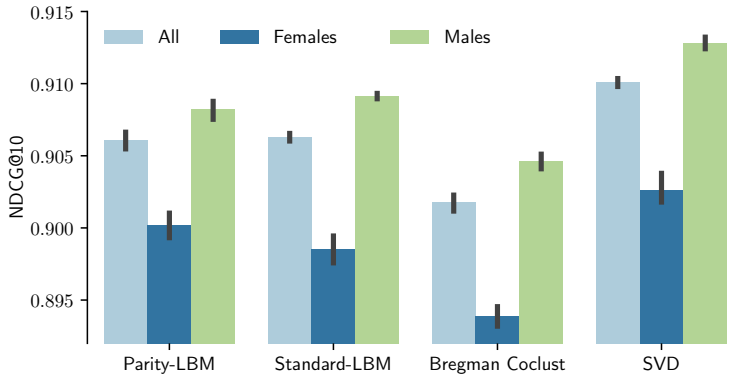


Figure 3.5: Normalized Discounted Cumulative Gain estimated on MovieLens-1M (the higher the better)

recommendations are less relevant to female users, under the assumption that the observed ratings are somewhat influenced by gender stereotypes, it is not possible to satisfactorily measure the performance of fair recommendations from the original rating matrix.

To reduce the disparate performances between males and females, a possible solution is to systematically favors items identified as overrated by females. We recall that the latent variable C_j, which is until now not used for fair prediction, captures the difference in opinion trends between female and male users on movie j. A high absolute value of C_j indicates a strongly gendered opinion for movie j. With our encoding of genres, negative C_j indicate a relative overrating by males and positive C_j indicate a relative overrating by females. If the relevance score of Equation (3.7) is modified by using this latent variable C_j, but not the protected attribute, some stereotype is equally injected in the recommendations of all users:

$$\hat{X}_{ij} = \boldsymbol{\tau}_i^{(Y)} \hat{\boldsymbol{\pi}} \boldsymbol{\tau}_j^{(Z)^T} + \nu_i^{(A)} + \nu_j^{(B)} + \lambda\ \nu_j^{(C)} \quad , \tag{3.9}$$

with $\lambda \in \mathbb{R}$ being the strength of the stereotype injected in recommendations. A positive value of λ will favors movies identified as overrated by females while a negative value will favors movies identified as overrated by males. It should be noted that altering the relevance score to inject stereotype for all users do not modify the gap ε from exactly fair recommendations of the method. Figure 3.6 shows that the gap of performances between males and females can be reduced up to be equalized. Although these results show that seeking for an equality of impact is possible, it comes with the cost that some sort of stereotype, originally present in the dataset, would remain in the recommendations which is ethically arguable and should be used with cautious for each type of recommender system.

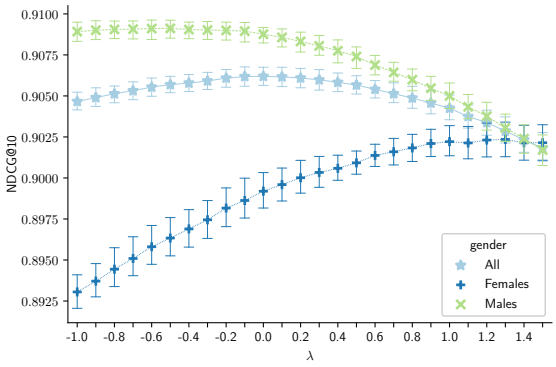


Figure 3.6: NDCG estimated on MovieLens-1M (the higher the better) using relevance scores exploiting stereotype (Equation 3.9). A positive (resp. negative) value of λ favors movies overrated by females (resp. males).

Finally, we present some insights provided by our model on movies. We display the empirical cumulative distribution function (CDF) of C_j for movies conditionally on their genre (for some handpicked archetypal genres). The dominance of the CDF for a given genre expresses that, according to our model, female users have a higher opinion than male users for the movies belonging to that genre. Figure 3.7 shows the results, which reflect stereotypes that women are more likely than men to positively evaluate musical films and dramas, while men are similarly inclined toward westerns and action films. These stereotypes are incorporated into our model to fit actual ratings, but ignored to deliver fair recommendations using Equation (3.7). The lists of extreme movies based on extreme (positive and negative) values of C_j is given in Appendix 3.A.4.1.

The latent variable B_j encodes the overall opinion trend about movie j. Two interesting observations can be made from the scatter plot of B_j versus movie popularity (see bottom of Figure 3.7). First, unpopular movies are also the least appreciated according to our model; this supports the hypothesis that ratings are generated by a MNAR (Missing Not At Random) process, where a missing rating can be considered as weak negative feedback, assuming that users primarily rate items they like. This missingness process must still be taken into account in our model. Second, it shows that the most liked movies (according to our model) are not necessarily the most popular (and will be recommended); the recommendations are not affected by popularity bias.

3.3 Conclusion

We proposed a new co-clustering method for fair recommendation. Our model combines the Gaussian Latent Block Model with an ordinal regression model. The sensitive attribute is adequately accounted for in the model,

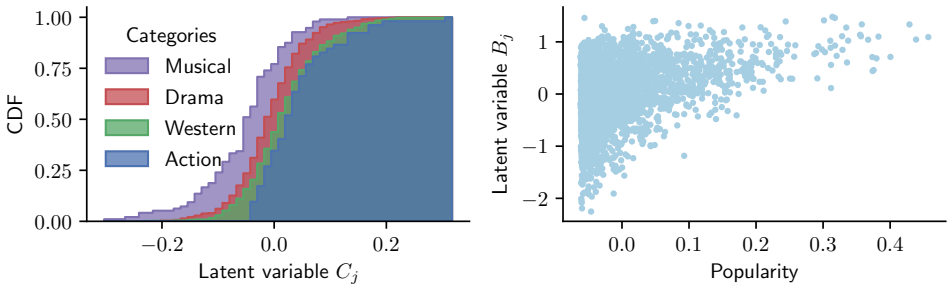


Figure 3.7: Left: cumulative distribution function of latent variable C_j conditionally on the genre of the movie. A dominating CDF indicates a genre for which females' opinions are more positive than males'. Right: scatter plot of the movie latent variable B_j versus popularity (ratio of ratings). High positive values of B_j (resp. popularity) correspond to movies that are the most liked (resp. popular).

allowing the clustering of users to be unaffected by the effects of this attribute on ratings. This results in user clusters that approximately respect statistical parity. We base recommendation on a relevance score that ignores the sensitive attribute in order to compare items fairly.

We provide theoretical guarantees ensuring approximately fair recommendations, for any known discrete sensitive attribute, provided that the clustering of users respects an approximate statistical parity that can be assessed in practice. Our analysis focuses on the fairness of preferences, as defined by the ranking of ratings, rather than on the predicted values themselves, which are less relevant for recommendation. Through experiments on real-world data, we show that our method significantly mitigates the unfairness of recommendations. Furthermore, the latent variables inferred by the model are also amenable to analyses that can help identify recommendation bias.

Our study supports that the absence of rating conveys some information that should be exploited. Previous work [Marlin et al., 2007, 2012] has already shown that the data used for collaborative filtering datasets can be strongly influenced by observational bias, motivating the use of Missing Not At Random (MNAR) models. Societal biases may have a significant contribution to missingness, leading to an additional source of unfairness if missingness is not properly modeled. Studying fairness with MNAR processes is a highly relevant but extremely challenging direction for future research, as assessing the relevance of MNAR models in real situations requires data that are typically produced by online randomized experiments.

Bibliography

Mohsen Abbasi, Aditya Bhaskara, and Suresh Venkatasubramanian. Fair clustering via equitable group representations. In Madeleine Clare Elish, William Isaac, and Richard S. Zemel, editors, *ACM Conference on Fairness, Accountability, and Transparency (FAccT)*, pages 504–514, 2021. URL https://doi.org/10.1145/3442188.3445913. 101

Arindam Banerjee, Inderjit Dhillon, Joydeep Ghosh, Srujana Merugu, and Dharmendra S Modha. A generalized maximum entropy approach to bregman co-clustering and matrix approximation. *The Journal of Machine Learning Research*, 8:1919–1986, 2007. 110

Jean-Patrick Baudry and Gilles Celeux. EM for mixtures. *Statistics and Computing*, 25(4):713–726, 2015. doi: 10.1007/s11222-015-9561-x. 107

Suman K. Bera, Deeparnab Chakrabarty, Nicolas J. Flores, and Maryam Negahbani. Fair algorithms for clustering, 2019. 101

Alex Beutel, Jilin Chen, Tulsee Doshi, Hai Qian, Li Wei, Yi Wu, Lukasz Heldt, Zhe Zhao, Lichan Hong, Ed H. Chi, and Cristos Goodrow. *Fairness in Recommendation Ranking through Pairwise Comparisons*, pages 2212—-2220. 2019. ISBN 9781450362016. URL https://doi.org/10.1145/3292500.3330745. 101

Christophe Biernacki, Gilles Celeux, and Gérard Govaert. Choosing starting values for the EM algorithm for getting the highest likelihood in multivariate Gaussian mixture models. *Computational Statistics & Data Analysis*, 41: 561–575, 2003. doi: 10.1016/S0167-9473(02)00163-9. 107

Reuben Binns. Fairness in machine learning: Lessons from political philosophy. In Sorelle A. Friedler and Christo Wilson, editors, *Proceedings of the 1st Conference on Fairness, Accountability and Transparency*, volume 81 of *Proceedings of Machine Learning Research*, pages 149–159, New York, NY, USA, 23–24 Feb 2018. PMLR. URL http://proceedings.mlr.press/v81/binns18a.html. 98

Vincent Brault and Mahendra Mariadassou. Co-clustering through latent bloc model: A review. *Journal de la Société Française de Statistique*, 156(3): 120–139, 2015. URL http://journal-sfds.fr/article/view/474/448. 106

Robin Burke, Nasim Sonboli, and Aldo Ordonez-Gauger. Balanced neighborhoods for multi-sided fairness in recommendation. In *1st Conference on Fairness, Accountability and Transparency*, volume 81 of *PMLR*, pages 202–214, 2018. URL http://proceedings.mlr.press/v81/burke18a.html. 98

Paul-Christian Bürkner and Matti Vuorre. Ordinal regression models in psychology: A tutorial. *Advances in Methods and Practices in Psychological Science*, 2(1):77–101, 2019. 103

Anne R. Daykin and Peter G. Moffatt. Analyzing ordered responses: A review of the ordered probit model. *Understanding Statistics*, 1(3):157–166, 2002. doi: 10.1207/S15328031US0103_02. 103

Thomas N. Daymonti and Paul J. Andrisani. Job preferences, college major, and the gender gap in earnings. *Journal of Human Resources*, 19(3):408–428, 1984. URL https://EconPapers.repec.org/RePEc:uwp:jhriss:v:19:y:1984:i:3:p:408-428. 98

Gabriel Frisch, Jean-Benoist Leger, and yves Grandvalet. Co-clustering for fair recommendation. working paper or preprint, May 2021. URL https://hal.archives-ouvertes.fr/hal-03239856. 103

Pratik Gajane. On formalizing fairness in prediction with machine learning. *CoRR*, abs/1710.03184, 2017. URL http://arxiv.org/abs/1710.03184. 98

Thomas George and Srujana Merugu. A scalable collaborative filtering framework based on co-clustering. In *Fifth IEEE International Conference on Data Mining (ICDM)*, 2005. 110

Mehrdad Ghadiri, Samira Samadi, and Santosh Vempala. Socially fair k-means clustering. *arXiv preprint arXiv:2006.10085*, 2020. 101

Moritz Hardt, Eric Price, and Nathan Srebro. Equality of opportunity in supervised learning. In *Advances in Neural Information Processing Systems 29*, pages 3315–3323, 2016. URL https://proceedings.neurips.cc/paper/2016/hash/9d2682367c3935defcb1f9e247a97c0d-Abstract.html. 111

Nicolas Hug. Surprise: A python library for recommender systems. *Journal of Open Source Software*, 5(52):2174, 2020. doi: 10.21105/joss.02174. 110

Tommi S. Jaakkola. Tutorial on variational approximation methods. In *Advanced Mean Field Methods: Theory and Practice*, pages 129–159. MIT Press, 2000. 106

Eric Jang, Shixiang Gu, and Ben Poole. Categorical reparameterization with gumbel-softmax. *arXiv preprint arXiv:1611.01144*, 2016. 108

Kalervo Järvelin and Jaana Kekäläinen. IR evaluation methods for retrieving highly relevant documents. In *ACM SIGIR Forum*, volume 51, pages 243–250, 2017. 109

Toshihiro Kamishima, Shotaro Akaho, Hideki Asoh, and Jun Sakuma. Recommendation independence. In *Conference on Fairness, Accountability and Transparency*, pages 187–201, 2018. 98, 99, 100

Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014. 107

Y. Koren, R. Bell, and C. Volinsky. Matrix factorization techniques for recommender systems. *Computer*, 42(8):30–37, Aug 2009. 110

Benjamin M. Marlin, Richard S. Zemel, Sam T. Roweis, and Malcolm Slaney. Collaborative filtering and the missing at random assumption. In *Twenty-Third Conference on Uncertainty in Artificial Intelligence (UAI)*, pages 267–275, 2007. 105, 114

Benjamin M. Marlin, Richard S. Zemel, Sam T. Roweis, and Malcolm Slaney. Collaborative filtering and the missing at random assumption. *CoRR*, abs/1206.5267, 2012. URL http://arxiv.org/abs/1206.5267. 105, 114

movie lens. Movielens 1M datasets. https://grouplens.org/datasets/movielens/. 109

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. Pytorch: An imperative style, high-performance deep learning library. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems 32*, pages 8024–8035. Curran Associates, Inc., 2019. 107

Rajesh Ranganath, Sean Gerrish, and David Blei. Black Box Variational Inference. In Samuel Kaski and Jukka Corander, editors, *Proceedings of the Seventeenth International Conference on Artificial Intelligence and Statistics*, volume 33 of *Proceedings of Machine Learning Research*, pages 814–822, Reykjavik, Iceland, 22–25 Apr 2014. PMLR. URL http://proceedings.mlr.press/v33/ranganath14.html. 107

Tim Räz. Group fairness: Independence revisited. *arXiv preprint arXiv:2101.02968*, 2021. 98

Steffen Rendle, Li Zhang, and Yehuda Koren. On the difficulty of evaluating baselines: A study on recommender systems. *arXiv preprint arXiv:1905.01395*, 2019. 110

Donald B. Rubin. Inference and missing data. *Biometrika*, 63(3):581–592, 1976. ISSN 00063444. URL http://www.jstor.org/stable/2335739. 105

Sirui Yao and Bert Huang. Beyond parity: Fairness objectives for collaborative filtering. *CoRR*, abs/1705.08804, 2017. URL http://arxiv.org/abs/1705.08804. 99, 100

Ziwei Zhu, Xia Hu, and James Caverlee. Fairness-aware tensor-based recommendation. In *27th ACM International Conference on Information and Knowledge Management*, pages 1153—-1162, 2018. ISBN 9781450360142. doi: 10.1145/3269206.3271795. 100

3.A Appendix - Co-clustering for fair recommendation

3.A.1 Computation of the variational log-likelihood criterion

The criterion we want to optimize is:

$$\mathcal{J}(q_\gamma, \theta) = \mathcal{H}(q_\gamma) + \mathbb{E}_{q_\gamma}\left[\mathcal{L}(\boldsymbol{X}, \boldsymbol{Y}, \boldsymbol{Z}, \boldsymbol{A}, \boldsymbol{B}, \boldsymbol{C}; \theta)\right] \ . \tag{3.10}$$

We chose to restrict the space of the variational distribution q_γ in order to get a fully factorized form:

$$\begin{aligned} q_\gamma = &\prod_{i=1}^{n_1} \mathcal{M}\left(1; \tau_i^{(Y)}\right) \times \prod_{j=1}^{n_2} \mathcal{M}\left(1; \tau_j^{(Z)}\right) \\ &\times \prod_{i=1}^{n_1} \mathcal{N}\left(\nu_i^{(A)}, \rho_i^{(A)}\right) \times \prod_{j=1}^{n_2} \mathcal{N}\left(\nu_j^{(B)}, \rho_j^{(B)}\right) \\ &\times \prod_{j=1}^{n_2} \mathcal{N}\left(\nu_j^{(C)}, \rho_j^{(C)}\right) \end{aligned} \tag{3.11}$$

where γ denotes the parameters concatenation of the variational distribution[2] q_γ. The entropy is additive across independent variables so we get:

$$\mathcal{H}(q_\gamma) = \mathcal{H}(q_\gamma(\boldsymbol{Y})) + \mathcal{H}(q_\gamma(\boldsymbol{Z})) + \mathcal{H}(q_\gamma(\boldsymbol{A})) + \mathcal{H}(q_\gamma(\boldsymbol{B})) + \mathcal{H}(q_\gamma(\boldsymbol{C})) \ ,$$

with the following terms:

$$\begin{aligned} \mathcal{H}(q_\gamma(\boldsymbol{Y})) &= -\sum_{iq} \tau_{iq}^{(Y)} \log \tau_{iq}^{(Y)} \\ \mathcal{H}(q_\gamma(\boldsymbol{Z})) &= -\sum_{jl} \tau_{jl}^{(Y)} \log \tau_{jl}^{(Z)} \\ \mathcal{H}(q_\gamma(\boldsymbol{A})) &= \frac{1}{2} \sum_i \log \rho_i^{(A)} + \frac{n_1}{2}(\log 2\pi + 1) \\ \mathcal{H}(q_\gamma(\boldsymbol{B})) &= \frac{1}{2} \sum_j \log \rho_j^{(B)} + \frac{n_2}{2}(\log 2\pi + 1) \\ \mathcal{H}(q_\gamma(\boldsymbol{C})) &= \frac{1}{2} \sum_j \log \rho_j^{(C)} + \frac{n_2}{2}(\log 2\pi + 1) \end{aligned}$$

[2] $\gamma = (\boldsymbol{\tau}^{(Y)}, \boldsymbol{\tau}^{(Z)}, \boldsymbol{\nu}^{(A)}, \boldsymbol{\rho}^{(A)}, \boldsymbol{\nu}^{(B)}, \boldsymbol{\rho}^{(B)}, \boldsymbol{\nu}^{(C)}, \boldsymbol{\rho}^{(C)})$

The independence of the latent variables allows to rewrite the expectation of the complete log-likelihood as:

$$\begin{aligned}\mathbb{E}_{q_\gamma}[\mathcal{L}(\boldsymbol{X},\boldsymbol{Y},\boldsymbol{Z},\boldsymbol{A},\boldsymbol{B},\boldsymbol{C})] = \mathbb{E}_{q_\gamma}[\mathcal{L}(\boldsymbol{Y})] + \mathbb{E}_{q_\gamma}[\mathcal{L}(\boldsymbol{Z})] \\ + \mathbb{E}_{q_\gamma}[\mathcal{L}(\boldsymbol{A})] + \mathbb{E}_{q_\gamma}[\mathcal{L}(\boldsymbol{B})] + \mathbb{E}_{q_\gamma}[\mathcal{L}(\boldsymbol{C})] \\ + \mathbb{E}_{q_\gamma}[\mathcal{L}(\boldsymbol{X}|\boldsymbol{Y},\boldsymbol{Z},\boldsymbol{A},\boldsymbol{B},\boldsymbol{C})] \ ,\end{aligned}$$

with the following terms:

$$\mathbb{E}_{q_\gamma}\mathcal{L}(\boldsymbol{Y}) = \mathbb{E}_{q_\gamma}\left[\sum_{iq} Y_{iq}\log\alpha_q\right] = \sum_{iq}\tau_{iq}^{(Y)}\log\alpha_q$$

$$\mathbb{E}_{q_\gamma}\mathcal{L}(\boldsymbol{Z}) = \mathbb{E}_{q_\gamma}\left[\sum_{jl} Z_{jl}\log\beta_l\right] = \sum_{jl}\tau_{jl}^{(Z)}\log\beta_l$$

$$\begin{aligned}\mathbb{E}_{q_\gamma}\mathcal{L}(\boldsymbol{A}) &= -\frac{n_1}{2}\log 2\pi - \frac{n_1}{2}\log\sigma_A^2 - \frac{1}{2\sigma_A^2}\sum_i \mathbb{E}_{q_\gamma}A_i^2 \\ &= -\frac{n_1}{2}\log 2\pi - \frac{n_1}{2}\log\sigma_A^2 - \frac{1}{2\sigma_A^2}\sum_i\left(\left(\nu_i^{(A)}\right)^2 + \rho_i^{(A)}\right) \\ \mathbb{E}_{q_\gamma}\mathcal{L}(\boldsymbol{B}) &= -\frac{n_2}{2}\log 2\pi - \frac{n_2}{2}\log\sigma_B^2 - \frac{1}{2\sigma_B^2}\sum_i\left(\left(\nu_i^{(B)}\right)^2 + \rho_i^{(B)}\right) \\ \mathbb{E}_{q_\gamma}\mathcal{L}(\boldsymbol{C}) &= -\frac{n_2}{2}\log 2\pi - \frac{n_2}{2}\log\sigma_C^2 - \frac{1}{2\sigma_C^2}\sum_j\left(\left(\nu_j^{(C)}\right)^2 + \rho_j^{(C)}\right)\end{aligned}$$

and as the entries of the data matrix $\boldsymbol{X}$ are independent and identically distributed:

$$\mathbb{E}_{q_\gamma}\mathcal{L}(\boldsymbol{X}|\boldsymbol{A},\boldsymbol{B},\boldsymbol{C},\boldsymbol{Y},\boldsymbol{Z}) = \mathbb{E}_{q_\gamma}\mathcal{L}\left(\boldsymbol{X}^{(\mathrm{o})}\middle|\boldsymbol{A},\boldsymbol{B},\boldsymbol{C},\boldsymbol{Y},\boldsymbol{Z}\right) + \mathcal{L}\left(\boldsymbol{X}^{(\neg o)}\right) \quad (3.12)$$

where $\boldsymbol{X}^{(\mathrm{o})}$ denotes the set of observed ratings and $\boldsymbol{X}^{(\neg o)}$, the set of non-observed ratings, where X_{ij} = NA. From Equation 3.12, it becomes clear that maximizing $\mathbb{E}_{q_\gamma}\mathcal{L}(\boldsymbol{X}^{(\neg o)})$ is not necessary to infer the model parameters used for prediction and therefore ignoring the non-observed data is correct. The expectation of the conditional log-likelihood (first term of right side of Equation 3.12) is numerically estimated by sampling from q_γ.

Stochastic gradient optimization To optimize the criterion with stochastic gradient descent, we express the variational log-likelihood criterion on a single

rating:

$$\begin{aligned}\mathcal{J}(X_{ij};q_\gamma,\theta) = \mathbb{E}_{q_\gamma}\Big[\mathcal{L}\Big(X_{ij}^{(o)}\Big|\boldsymbol{Y}_i,\boldsymbol{Z}_j,A_i,B_j,C_j\Big)\Big] \\ + \frac{1}{n_2}\big(\mathcal{H}(q_\gamma(\boldsymbol{Y}_i)) + \mathcal{H}(q_\gamma(A_i)) + \mathbb{E}_{q_\gamma}[\mathcal{L}(\boldsymbol{Y}_i)] + \mathbb{E}_{q_\gamma}[\mathcal{L}(A_i)]\big) \\ + \frac{1}{n_2}\big(\mathcal{H}(q_\gamma(\boldsymbol{Z}_j)) + \mathcal{H}(q_\gamma(B_j)) + \mathbb{E}_{q_\gamma}[\mathcal{L}(\boldsymbol{Z}_j)] + \mathbb{E}_{q_\gamma}[\mathcal{L}(B_j)]\big) \\ + \frac{1}{n_2}\big(\mathcal{H}(q_\gamma(C_j)) + \mathbb{E}_{q_\gamma}[\mathcal{L}(C_j)]\big)\end{aligned}$$

A batch of data, $\boldsymbol{X}_{(i:i+n),(j:j+n)}$, consists of a $(n \times n)$ sub-matrix randomly sampled from the original matrix $\boldsymbol{X}$.

3.A.2 Clustering ε-parity and ε-fair recommendation for arbitrary discrete sensitive attribute

Definition 4 (Clustering ε-parity, arbitrary discrete sensitive attribute) *The clustering of users is said to respect ε-parity with respect to the discrete attribute $s \in \mathcal{S}$ iff:*

$$\forall (t,t') \in \mathcal{S}^2,\ \forall q,\ \left|\frac{\#\{i|s_i = t \wedge u_{iq} = 1\}}{\#\{i|s_i = t\}} - \frac{\#\{i|s_i = t' \wedge u_{iq} = 1\}}{\#\{i|s_i = t'\}}\right| \leq \varepsilon\ , \quad (3.13)$$

where $\varepsilon \in \mathbb{R}_+$ measures the gap to exact parity, u_{iq} is the (hard) membership of user i to cluster q,and $\#\{i|\Omega\}$ is the number of users defined by the cardinality of the set Ω.

Definition 5 (ε-fair recommendation, arbitrary discrete sensitive attribute) *A recommender system is said to be ε-fair with respect to the dicrete attribute $s \in \mathcal{S}$ if for any two items j and j':*

$$\forall (t,t') \in \mathcal{S}^2,\ \left|\frac{\#\left\{i|s_i = t \wedge (\hat{X}_{ij} > \hat{X}_{ij'})\right\}}{\#\{i|s_i = t\}} - \frac{\#\left\{i|s_i = t' \wedge (\hat{X}_{ij} > \hat{X}_{ij'})\right\}}{\#\{i|s_i = t'\}}\right| \leq \varepsilon\ , \quad (3.14)$$

where $\varepsilon \in \mathbb{R}_+$ measures the gap to exact fairness

3.A.3 Proof of Theorem 1

Theorem 2 (Fair recommendation from clustering parity) *If the clustering of users in k_1 groups respects ε-parity (Definition 3 or Definition 4) then the recommender system relying on the relevance score defined in Equation (3.7) is $(k_1\varepsilon)$-fair (Definition 1 or Definition 5).*

Proof: *Suppose that $\boldsymbol{\tau}^{(Y)}$, the maximum a posteriori of $\boldsymbol{Y}$, is a binary matrix; $\boldsymbol{\tau}^{(Y)}$ is thus a $n_1 \times k_1$ indicator matrix of row classes membership.*

Then, given user i, item j is said to be preferred to item j' if $\hat{X}_{ij} > \hat{X}_{ij'}$, that is:

$$\begin{aligned}
\hat{X}_{ij} > \hat{X}_{ij'} &\iff \boldsymbol{\tau}_i^{(Y)} \hat{\boldsymbol{\pi}} \boldsymbol{\tau}_j^{(Z)^T} + \nu_i^{(A)} + \nu_j^{(B)} > \boldsymbol{\tau}_i^{(Y)} \hat{\boldsymbol{\pi}} \boldsymbol{\tau}_{j'}^{(Z)^T} + \nu_i^{(A)} + \nu_{j'}^{(B)} \\
&\iff \boldsymbol{\tau}_i^{(Y)} \hat{\boldsymbol{\pi}} \left(\boldsymbol{\tau}_j^{(Z)} - \boldsymbol{\tau}_{j'}^{(Z)} \right)^T > \nu_{j'}^{(B)} - \nu_j^{(B)} \\
&\iff \boldsymbol{\tau}_i^{(Y)} \boldsymbol{a} > b \\
&\iff \boldsymbol{a}_{d_i} > b \ , \qquad (3.15)
\end{aligned}$$

with $\boldsymbol{a} \in \mathbb{R}^{k_1}$ defined by $\boldsymbol{a} = \hat{\boldsymbol{\pi}} \left(\boldsymbol{\tau}_j^{(Z)} - \boldsymbol{\tau}_{j'}^{(Z)} \right)^T$, $b \in \mathbb{R}$ defined by $b = \nu_{j'}^{(B)} - \nu_j^{(B)}$ and $d_i \in \{1, \cdots, k_1\}$ being the group indicator of user i: $\tau_{i,d_i}^{(Y)} = 1$.

Suppose ε-parity, from Definition 4 (Definition 3 is a particular case of Definition 4), we have

$$\forall (t,t'), \qquad \forall q, \qquad \left| \frac{\#\{i | s_i = t \wedge d_i = q\}}{\#\{i | s_i = t\}} - \frac{\#\{i | s_i = t' \wedge d_i = q\}}{\#\{i | s_i = t'\}} \right| \leq \varepsilon$$

therefore,

$$\forall (t,t'), \quad \forall q, \quad \left| \mathbb{1}_{\boldsymbol{a}_{d_i} > b} \frac{\#\{i | s_i = t \wedge d_i = q\}}{\#\{i | s_i = t\}} - \mathbb{1}_{\boldsymbol{a}_{d_i} > b} \frac{\#\{i | s_i = t' \wedge d_i = q\}}{\#\{i | s_i = t'\}} \right| \leq \varepsilon \mathbb{1}_{\boldsymbol{a}_{d_i} > b}$$

By summing over all groups, we get:

$$\forall (t,t'), \quad \sum_q \left| \frac{\mathbb{1}_{\boldsymbol{a}_{d_i} > b} \#\{i | s_i = t \wedge d_i = q\}}{\#\{i | s_i = t\}} - \frac{\mathbb{1}_{\boldsymbol{a}_{d_i} > b} \#\{i | s_i = t' \wedge d_i = q\}}{\#\{i | s_i = t'\}} \right| \leq \varepsilon \sum_q \mathbb{1}_{\boldsymbol{a}_{d_i} > b}$$

and from the triangular inequality,

$$\begin{aligned}
&\forall (t,t'), \quad \left| \frac{\sum_q \mathbb{1}_{\boldsymbol{a}_{d_i} > b} \#\{i | s_i = t \wedge d_i = q\}}{\#\{i | s_i = t\}} - \frac{\sum_q \mathbb{1}_{\boldsymbol{a}_{d_i} > b} \#\{i | s_i = t' \wedge d_i = q\}}{\#\{i | s_i = t'\}} \right| \leq \varepsilon \sum_q \mathbb{1}_{\boldsymbol{a}_{d_i} > b} \\
&\forall (t,t'), \quad \left| \frac{\#\{i | s_i = t \wedge \boldsymbol{a}_{d_i} > b\}}{\#\{i | s_i = t\}} - \frac{\#\{i | s_i = t' \wedge \boldsymbol{a}_{d_i} > b\}}{\#\{i | s_i = t'\}} \right| \leq \varepsilon k_1
\end{aligned}$$

And, applying (3.15), the result is obtained:

$$\forall (t,t'), \quad \left| \frac{\#\left\{i | s_i = t \wedge (\hat{X}_{ij} > \hat{X}_{ij'})\right\}}{\#\{i | s_i = t\}} - \frac{\#\left\{i | s_i = t' \wedge (\hat{X}_{ij} > \hat{X}_{ij'})\right\}}{\#\{i | s_i = t'\}} \right| \leq \varepsilon k_1$$

□

Table 3.2: List of movies with the largest gap in opinion between females and males for which females have a better opinion than males

Title	Year	Genders	Cj
Dirty Dancing	1987	Musical|Romance	0.31
Rocky Horror Picture Show, The	1975	Comedy|Horror|Musical|Sci-Fi	0.26
Sound of Music, The	1965	Musical	0.24
Grease	1978	Comedy|Musical|Romance	0.23
Jumpin' Jack Flash	1986	Action|Comedy|Romance|Thriller	0.23
Gone with the Wind	1939	Drama|Romance|War	0.22
Newsies	1992	Children's|Musical	0.21
Strictly Ballroom	1992	Comedy|Romance	0.21
Steel Magnolias	1989	Drama	0.20
Sense and Sensibility	1995	Drama|Romance	0.20
Full Monty, The	1997	Comedy	0.19
Much Ado About Nothing	1993	Comedy|Romance	0.18
Thelma & Louise	1991	Action|Drama	0.18
Swing Kids	1993	Drama|War	0.17
Fried Green Tomatoes	1991	Drama	0.17
Ever After: A Cinderella Story	1998	Drama|Romance	0.17
Anastasia	1997	Animation|Children's|Musical	0.17
Little Women	1994	Drama	0.17
Color Purple, The	1985	Drama	0.17
To Wong Foo, Thanks for Everything!	1995	Comedy	0.17

3.A.4 Supplemental results for MovieLens 1M

3.A.4.1 Gender as sensitive attribute

Supplemental analysis of the model We list in Tables 3.2 and 3.3 the most extreme movies according to the inferred value of their latent variable C_j. Variable C_j encodes the difference in opinion between the sensitive groups, not the overall opinion. For example, a movie may well be liked by most people but liked even more by males. Table 3.2 lists movies for which females have a better opinion than males and Table 3.3 lists movies for which males have a better opinion than females.

Higher number of groups We did not optimize the hyper-parameters of the compared models. We present here additional experiments to illustrate that the conclusions of Section 4.2.3 apply to different hyper-parameter settings. Using a substantially larger number of groups ($k_1 = 50$ user groups and $k_2 = 50$ item groups) or a larger dimension of latent factors for SVD (also 50), the statistical gender parity measures given in Table 3.4 and the recommendation performance given in Figure 3.8 are qualitatively similar to the ones given in Table 3.1 and Figure 3.5.

Table 3.3: List of movies with the largest gap in opinion between females and males for which males have a better opinion than females

Title	Year	Genders	Cj
Good, The Bad and The Ugly, The	1966	Action|Western	-0.32
Animal House	1978	Comedy	-0.30
Caddyshack	1980	Comedy	-0.27
Dumb & Dumber	1994	Comedy	-0.27
Exorcist, The	1973	Horror	-0.24
Clockwork Orange, A	1971	Sci-Fi	-0.24
Patton	1970	Drama|War	-0.23
Godfather: Part II, The	1974	Action|Crime|Drama	-0.22
Reservoir Dogs	1992	Crime|Thriller	-0.22
Saving Private Ryan	1998	Action|Drama|War	-0.22
Airplane!	1980	Comedy	-0.21
Eyes Wide Shut	1999	Drama	-0.21
Aliens	1986	Action|Sci-Fi|Thriller|War	-0.21
Predator	1987	Action|Sci-Fi|Thriller	-0.20
Apocalypse Now	1979	Drama|War	-0.20
Unforgiven	1992	Western	-0.20
Evil Dead II (Dead By Dawn)	1987	Action|Adventure|Comedy|Horror	-0.20
Big Trouble in Little China	1986	Action|Comedy	-0.20
Godfather, The	1972	Action|Crime|Drama	-0.20

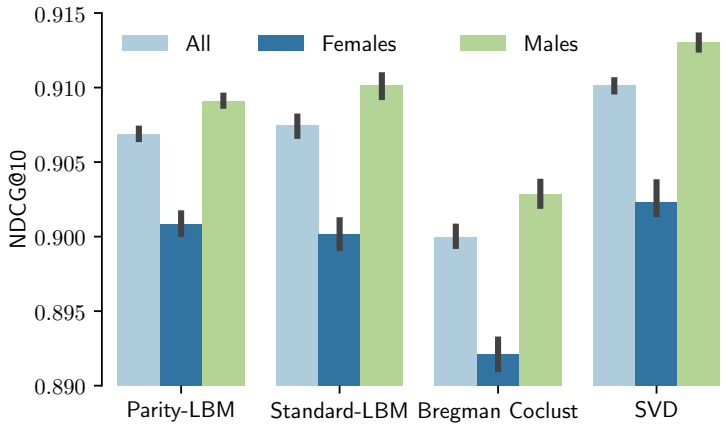


Figure 3.8: Normalized Discounted Cumulative Gain estimated on MovieLens-1M with $k_1 = k_2 = 50$ groups for clustering methods and 50 factors for the SVD.

Table 3.4: Measures of gender statistical parity. The number of user groups is $k_1 = 50$. The χ^2 statistic (with 49 degrees of freedom) is averaged over the five replicates of the experiment. A high value of the χ^2 statistic (or a low p-value) leads to the rejection of the statistical parity hypothesis.

Model	Parity LBM	Standard LBM	Bregman co-clust
χ^2 statistic	20	94	105
p-value	0.999	$1.1 \cdot 10^{-4}$	$5.8 \cdot 10^{-6}$

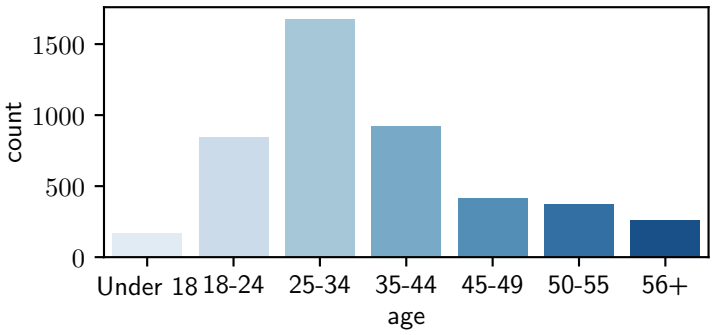


Figure 3.9: Count of users in each age category.

3.A.4.2 Age as sensitive attribute

The age range of the users is indicated within the following intervals: 'Under 18','18-24', '25-34', '35-44', '45-49', '50-55' and '56+'. The counts of users in each age category is displayed in Figure 3.9.

User age is treated as sensitive: we introduce seven binary sensitive attributes s_i encoding for the seven categories of user age. We use a one-hot encoding of the seven categories of user age and introduce for the purpose seven binary sensitive attributes $s_i^1, \cdots, s_i^7$ and their item associated latent variables $C_j^1, \cdots, C_j^7$. We use the protocol described in Section 4.2.3 with the exception that our Parity-LBM is initialized from estimates obtained with the Standard-LBM. Table 3.5 presents results of the χ^2 statistics constructed from the contingency table of user age counts in each group. The methods that do not consider the sensitive variable in the modelling create groups that are dependent on the age and assuming the statistical parity with our Parity-LBM model is reasonable.

Finally, we illustrate the interpretability of the estimates of the latent variables $C_j^1, \cdots, C_j^7$ related to movies. For each age category k, we select the thirty movies with the largest value of the latent variables C_j^k. These movies have the largest positive opinion bias for users in the given age category.

Table 3.5: Measures of statistical parity with respect to age category. The number of group of users is $k_1 = 15$. A high value of the χ^2 statistic (or a low p-value) leads to the rejection of the statistical parity hypothesis. The χ^2 statistic is averaged on the five folds of the cross-validation. Degrees of freedom is 14.

Model	Parity LBM	Standard LBM	Bregman co-clust
χ^2 statistic	99	144	577
p-value	0.12	$5.1 \cdot 10^{-5}$	$< 10^{-15}$

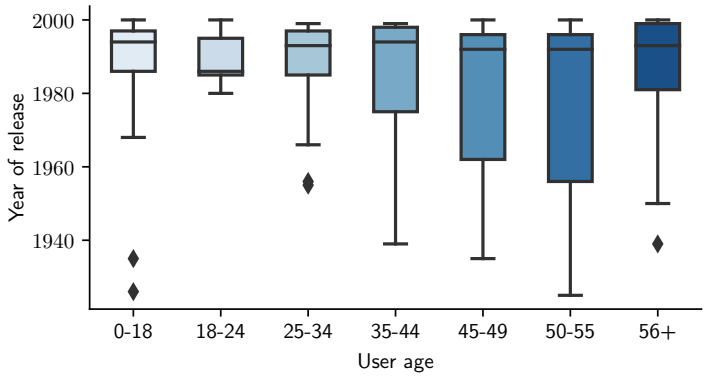


Figure 3.10: Release years of the thirty most extreme movies according to the inferred positive value of the latent variables $C_j^1, \cdots, C_j^7$. Each latent variable C_j^k is matched with its corresponding user age category.

Figure 3.10 displays a boxplot of the release years of these films for all user age categories. The greater variability in the distribution for older users means that they have a comparatively higher opinion of older movies than younger users. If user age were the sensitive attribute, the recommendations would not account for these differences.

Chapter 4

Handling large graphs for recommendation

Every time computing power increases by a factor of ten we should totally rethink how and what we compute

Chuck Dickens

Contents

4.1 Introduction

Block models are clustering or coclustering tools that are commonly used for community detection in graph data. This chapter studies block model inference for processing large graphs, which are commonly encountered in applications such as collaborative filtering or text mining where nodes represent words. In such contexts, the size of the graph poses a computational problem for handling the model, be it the stochastic block model (SBM) or the latent block model (LBM).

As a first contribution, we present a variational inference algorithm for the stochastic block model and the latent block model for sparse graphs, which leverages on the sparsity of edges to scale up to a very large number of nodes. This algorithm is implemented in SparseBM, a Python module that takes advantage of the hardware speed up provided by graphics processing units (GPU). As a second contribution, we propose a generic inference for block models based on a stochastic optimization of the variational criterion, partially numerically estimated. With experiments on synthetic data, we show that the proposed inference can bring out better estimates than the standard variational EM procedure.

4.2 SparseBM: A Python module for handling large sparse graphs with block models

This section is available as a preprint on HAL [Frisch et al., 2021].

Dense graphs are usually represented by adjacency matrices as illustrated in Figure 4.1. When the average degree of vertices is low, most elements of the adjacency matrix are zero; the matrix is sparse. Such type of graph is commonly found in datasets generated from social networks or collaborative systems. For instance, the Movielens-25M dataset [Harper and Konstan, 2015] can be model by a bipartite network made of 120,000 users and 60,000 movies vertices with an average degree of 112 or by a biadjacency matrix with a sparsity rate of 97.7%. In such a context, the size of the adjacency matrix poses a computational problem for handling the model, be it the stochastic block model (SBM) [Holland et al., 1983] or the latent block model (LBM) [Govaert and Nadif, 2008, Nadif and Govaert, 2010].

These generative models for random graphs rely on mixtures, assuming that the observations are generated from finite mixture components in rows and columns. They have found applications in many areas such as text analysis [Selosse et al., 2020], genomic analysis [Aubert et al., 2016], ecology [Bar-Hen et al., 2020], collaborative filtering [Corneli et al., 2020], or political analysis [Latouche et al., 2011, Wyse and Friel, 2012]. These probabilistic models provide a co-clustering analysis of the nodes of a graph that can be compared, among others, spectral methods [Dhillon, 2001, Kluger et al., 2003], mutual information methods [Dhillon et al., 2003], modularity based methods [Labiod and Nadif, 2011] or non-negative matrix tri-factorization [Ding et al., 2006].

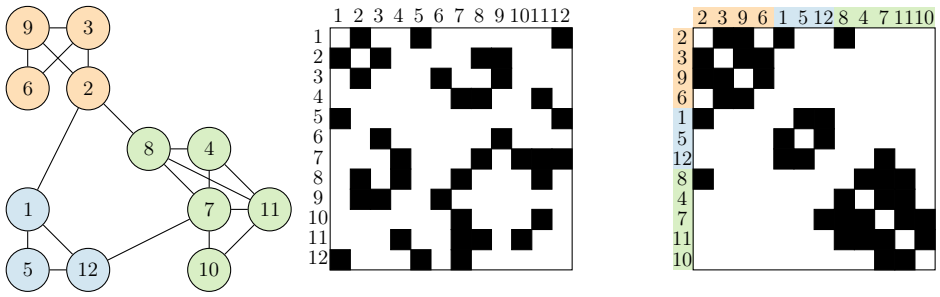


Figure 4.1: A binary graph on the left and its adjacency matrix on the center. The matrix on the right is the adjacency matrix reorganized according to the node clustering inferred by the stochastic block model.

Though adjacency lists are routinely used to represent sparse graphs in a compact way, the packages developed for SBM and LBM [Leger, 2016, Bhatia et al., 2017] rely on computations on dense adjacency matrices to benefit from the computational efficiency offered by matrix calculus. In this article, we show how to efficiently conduct inference with the SBM and LBM in very large

sparse graphs, using a computational representation based on an adjacency list instead of an adjacency matrix. This inference optimized for sparse graphs is implemented in **SparseBM**, a Python module that also takes advantage of the hardware speed up provided by graphics processing units (GPUs). Our contributions allow the analysis of graphs whose size is beyond the reach of current SBM and LBM implementations.

The article is organized as follows; Section 4.2.1 describes the original variational inferences and the ones we propose to reduce the complexity for sparse graphs. We provide an overview of the functionalities of the **SparseBM** module through the various examples of Section 4.2.2. Section 4.2.3 then reports experiments on synthetic datasets that show that our computational tricks are relevant to analyze sparse matrices.

Notation Let n_1 be the number of vertices of a graph and n_2 the number of vertices of the second set when considering a bipartite graph. Sums and products relative to the first and second set of vertices (if bipartite) will be indexed respectively by i and j, and the classes of these two types of vertices will be indexed by $q \in \{1, \dots, k_1\}$ and $l \in \{1, \dots, k_2\}$. The bounds of summations or products will be implicit, for example $\sum_i$ will be a shorthand for $\sum_{i=1}^{n_1}$ and $\prod_{ijql}$ for $\prod_{i=1}^{n_1} \prod_{j=1}^{n_2} \prod_{q=1}^{k_1} \prod_{l=1}^{k_2}$. We use set-builder notation to describe sets that are defined by a predicate, rather than explicitly enumerated, a colon separator in sums and products specifying this domain. For example, $\sum_{ijql:\, i \neq j,\, X_{ij}=1}$ is the quadruple sum on i, j, q, and l, such as the indices i and j are not equal and $X_{ij} = 1$, that is, an edge is present from vertex i to vertex j.

4.2.1 Estimation procedure

4.2.1.1 Computationally efficient variational inference for sparse graphs

The generative modelling the SBM and LBM can be split into a set of unobserved latent variables and a set of observed variables consisting of $\boldsymbol{X}$ only. An observation of $\boldsymbol{X}$ is referred to as incomplete data, and an observation of $\boldsymbol{X}$ together with the latent variables is referred to as complete data.

Given the incomplete data, the objective is to infer the model parameters θ via maximum likelihood $\widehat{\theta} = \arg\max_\theta p(\boldsymbol{X}; \theta)$. When applying the Expectation Maximization (EM) algorithm to the SBM or to the LBM to maximize $p(\boldsymbol{X}; \theta)$, the computation of the complete log-likelihood at the E-step requires the posterior distribution of the latent variables, which is intractable, because the search space of the latent variables is combinatorially too large [Brault and Mariadassou, 2015].

This problem is well known in the context of co-clustering; for both SBM and LBM, some methods [Celeux and Diebolt, 1985, Keribin et al., 2015] rely

on a stochastic E-step with Monte Carlo sampling, but these strategies are not suited to large-scale problems. We follow the variational reformulation of the problem that is more efficient in high dimension. The variational EM (VEM) [Jordan et al., 1999, Jaakkola, 2000] introduces q_γ, a restricted set of parametric distributions defined over the latent variables, and maximizes the following lower bound on the log-likelihood of the incomplete data:

$$\mathcal{J}(q_\gamma, \theta) = \log p(\boldsymbol{X}; \theta) - KL(q_\gamma \parallel p(\cdot|\boldsymbol{X}; \theta)) \ , \tag{4.1}$$

where KL stands for the Kullback-Leibler divergence and q_γ denotes the variational distribution over the latent variables. The criterion $\mathcal{J}(q_\gamma, \theta)$ can be rewritten as as the sum of a negative "energy" and the entropy of q_γ:

$$\mathcal{J}(q_\gamma, \theta) = \mathbb{E}_{q_\gamma}[\log p(\boldsymbol{X}, \cdot\,; \theta)] + \mathcal{H}(q_\gamma) \ , \tag{4.2}$$

where $\mathbb{E}_{q_\gamma}$ is the expectation with respect to the variational distribution and $\mathcal{H}(q_\gamma)$ is the entropy of the variational distribution. The variational distribution q_γ is restricted to belong to a set of distributions that lead to a tractable computation of the criterion of Equation 4.2. Here, as is usually done in variational inference, the conditional independence of the latent variables is assumed; this is known as the "mean-field approximation" [Parisi, 1988].

Variational inference of the stochastic block model The mean-field approximation applied to the stochastic block model leads to the following form of the variational distribution over the latent variable $\boldsymbol{Y}$:

$$q_\gamma = \prod_i \mathcal{M}(1; \boldsymbol{\tau}_i)$$

where $\boldsymbol{\tau}_i \in \mathbf{S}_{k_1 - 1}$ are the parameters of the variational multinomial distributions. Using the conditional independence of the latent variable, the criterion $\mathcal{J}(q_\gamma, \theta)$ is expanded as:

$$\mathcal{J}(q_\gamma, \theta) = \mathbb{E}_{q_\gamma}[\log p(\boldsymbol{X}|\boldsymbol{Y}; \theta)] + \mathbb{E}_{q_\gamma}[\log p(\boldsymbol{Y}; \boldsymbol{\alpha})] + \mathcal{H}(q_\gamma) \ ,$$

where

$$\begin{aligned} \mathbb{E}_{q_\gamma}[\log p(\boldsymbol{X}|\boldsymbol{Y}; \theta)] &= \sum_{ijql:\, i \neq j} \tau_{iq}\tau_{jl}\big(X_{ij} \log \pi_{ql} + (1 - X_{ij}) \log(1 - \pi_{ql})\big) \\ \mathbb{E}_{q_\gamma}[\log p(\boldsymbol{Y}; \boldsymbol{\alpha})] &= \sum_{iq} \tau_{iq} \log \alpha_q \\ \mathcal{H}(q_\gamma) &= -\sum_{iq} \tau_{iq} \log \tau_{iq} \ . \end{aligned} \tag{4.3}$$

As Equation 4.3 involves a sum on all the non-diagonal elements of the adjacency matrix $\boldsymbol{X}$, the computation of the criterion $\mathcal{J}(q_\gamma, \theta)$ is of complexity $\mathcal{O}({n_1}^2 {k_1}^2)$ where n_1 is the number of vertices and k_1 is the number of classes.

When the considered graph is large, this complexity becomes problematic: the adjacency matrix may not fit in memory and/or the computation time may be prohibitive. However, Equation (4.3) can be rewritten by summing only the non-zero elements of the adjacency matrix, lowering the complexity to $\mathcal{O}(\#\{ij : X_{ij} = 1\}{k_1}^2)$ where $\#\{ij : X_{ij} = 1\}$ is the number of non-zero entries in $\boldsymbol{X}$:

$$\mathbb{E}_{q_\gamma}[\log p(\boldsymbol{X}|\boldsymbol{Y};\theta)] = \sum_{ijql:\, i\neq j,\, X_{ij}=1} \tau_{iq}\tau_{jl}\big(\log \pi_{ql} - \log(1-\pi_{ql})\big) + \sum_{ql} \log\big(1-\pi_{ql}\big)\left(\left(\sum_i \tau_{iq}\right)\left(\sum_j \tau_{jl}\right) - \sum_i \tau_{iq}\tau_{il}\right) .$$

We give the parameter estimates as defined for the original VEM inference in Algorithm 4 and for the VEM inference optimized for sparse graphs in Algorithm 5. The memory complexity of the algorithm, originally in $\mathcal{O}({n_1}^2)$, is reduced to $\mathcal{O}(\#\{ij : X_{ij} = 1\})$.

Algorithm 4: VEM - Original version

Data: Adjacency matrix $\boldsymbol{X}$

Inititialize $\boldsymbol{\tau}$, $\boldsymbol{\alpha}$, $\boldsymbol{\pi}$

while $\mathcal{J}^{(t)} - \mathcal{J}^{(t-1)} > atol$ **do**

  repeat

    $Q_{il} = \sum_{j:\, j\neq i} \tau_{jl} X_{ij}$

    $R_l = \sum_{j:\, j\neq i} \tau_{jl}$

    $\tau_{iq} \propto \alpha_q \prod_l \pi_{ql}^{Q_{il}} (1-\pi_{ql})^{R_l - Q_{il}}$

  until *convergence*;

  $\alpha_q = \frac{1}{n_1}\sum_i \tau_{iq}$ $\qquad$ $\pi_{ql} = \frac{\sum_{ij:\, i\neq j} \tau_{iq}\tau_{jl}X_{ij}}{\sum_{ij:\, i\neq j} \tau_{iq}\tau_{jl}}$

Algorithm 5: VEM - Sparse graph

Data: Sparse adjacency matrix $\boldsymbol{X}$

Inititialize $\boldsymbol{\tau}$, $\boldsymbol{\alpha}$, $\boldsymbol{\pi}$

while $\mathcal{J}^{(t)} - \mathcal{J}^{(t-1)} > atol$ **do**

  repeat

    $Q_{il} = \sum_{j:\, j\neq i, X_{ij}=1} \tau_{jl}$

    $\tau_{iq} \propto \alpha_q \prod_{jl:\, j\neq i} (1-\pi_{ql})^{\tau_{jl}} \prod_l \left(\frac{\pi_{ql}}{(1-\pi_{ql})}\right)^{Q_{il}}$

  until *convergence*;

  $\alpha_q = \frac{1}{n_1}\sum_i \tau_{iq}$ $\qquad$ $\pi_{ql} = \frac{\sum_{ij:\, i\neq j, X_{ij}=1} \tau_{iq}\tau_{jl}}{(\sum_i \tau_{iq})(\sum_j \tau_{jl}) - \sum_i \tau_{iq}\tau_{il}}$

Variational inference of the latent block model The mean-field approximation applied to the Latent Block Model leads to the following form of the variational distribution over the latent variables $\boldsymbol{Y}$ and $\boldsymbol{Z}$:

$$q_\gamma = \prod_i \mathcal{M}\Big(1; \boldsymbol{\tau}_i^{(Y)}\Big) \prod_j \mathcal{M}\Big(1; \boldsymbol{\tau}_j^{(Z)}\Big) \ ,$$

where $\boldsymbol{\tau}_i^{(Y)}$ and $\boldsymbol{\tau}_j^{(Z)}$ are respectively the parameters of the variational multinomial distributions over the latent variables $\boldsymbol{Y}$ and $\boldsymbol{Z}$. Using the conditional independence of the latent variable, the criterion $\mathcal{J}(q_\gamma, \theta)$ is expanded as:

$$\mathcal{J}(q_\gamma, \theta) = \mathbb{E}_{q_\gamma}[\log p(\boldsymbol{X}|\boldsymbol{Y}, \boldsymbol{Z}; \theta)] + \mathbb{E}_{q_\gamma}[\log p(\boldsymbol{Y}; \boldsymbol{\alpha})] + \mathbb{E}_{q_\gamma}[\log p(\boldsymbol{Z}; \boldsymbol{\beta})] + \mathcal{H}(q_\gamma) \ ,$$

where

$$\mathbb{E}_{q_\gamma}[\log p(\boldsymbol{X}|\boldsymbol{Y}, \boldsymbol{Z}; \theta)] = \sum_{ijql} \tau_{iq}^{(Y)} \tau_{jl}^{(Z)} \Big(X_{ij} \log \pi_{ql} + (1 - X_{ij}) \log(1 - \pi_{ql}) \Big) \tag{4.4}$$

$$\begin{aligned}
\mathbb{E}_{q_\gamma}[\log p(\boldsymbol{Y}; \boldsymbol{\alpha})] &= \sum_{iq} \tau_{iq}^{(Y)} \log \alpha_q \\
\mathbb{E}_{q_\gamma}[\log p(\boldsymbol{Z}; \boldsymbol{\beta})] &= \sum_{jl} \tau_{jl}^{(Z)} \log \beta_l \\
\mathcal{H}(q_\gamma) &= -\sum_{iq} \tau_{iq}^{(Y)} \log \tau_{iq}^{(Y)} - \sum_{jl} \tau_{jl}^{(Y)} \log \tau_{jl}^{(Z)} \ .
\end{aligned}$$

Equation 4.4 is rewritten analogously to Equation 4.3, reducing the computational complexity of $\mathcal{J}(q_\gamma, \theta)$ from $\mathcal{O}(n_1 n_2 k_1 k_2)$ to $\mathcal{O}(\#\{ij : X_{ij} = 1\} k_1 k_2)$:

$$\begin{aligned}
\mathbb{E}_{q_\gamma}[\log p(\boldsymbol{X}|\boldsymbol{Y}, \boldsymbol{Z}; \theta)] = & \sum_{ijql:\, X_{ij}=1} \tau_{iq}^{(Y)} \tau_{jl}^{(Z)} \Big(\log \pi_{ql} - \log(1 - \pi_{ql}) \Big) \\
& + \sum_{ql} \log\big(1 - \pi_{ql}\big) \Big(\sum_i \tau_{iq}^{(Y)} \Big) \Big(\sum_j \tau_{jl}^{(Z)} \Big) \ .
\end{aligned}$$

We give the parameter estimates as defined for the original VEM inference in Algorithm 6 and for the VEM inference for sparse graphs in Algorithm 7.

Algorithm 6: VEM - Original version

Data: Adjacency matrix $\boldsymbol{X}$

Inititialize $\boldsymbol{\tau}^{(Y)}$, $\boldsymbol{\tau}^{(Z)}$, $\boldsymbol{\alpha}$, $\boldsymbol{\beta}$, $\boldsymbol{\pi}$

while $\mathcal{J}^{(t)} - \mathcal{J}^{(t-1)} > atol$ **do**

  repeat

    $Q_{il} = \sum_j \tau_{jl}^{(Z)} X_{ij}$

    $R_l = \sum_j \tau_{jl}^{(Z)}$

    $\tau_{iq}^{(Y)} \propto \alpha_q \prod_l \pi_{ql}^{Q_{il}} (1 - \pi_{ql})^{R_l - Q_{il}}$

    $S_{jq} = \sum_i \tau_{iq}^{(Y)} X_{ij}$

    $T_q = \sum_i \tau_{iq}^{(Y)}$

    $\tau_{jl}^{(Z)} \propto \beta_l \prod_q \pi_{ql}^{S_{jq}} (1 - \pi_{ql})^{T_q - S_{jq}}$

  until *convergence*;

  $\alpha_q = \frac{\sum_i \tau_{iq}^{(Y)}}{n_1}$ $\qquad$ $\beta_l = \frac{\sum_j \tau_{jl}^{(Z)}}{n_2}$ $\qquad$ $\pi_{ql} = \frac{\sum_{ij} \tau_{iq}^{(Y)} \tau_{jl}^{(Z)} X_{ij}}{\sum_{ij} \tau_{iq}^{(Y)} \tau_{jl}^{(Z)}}$

Algorithm 7: VEM - Sparse graph

Data: Sparse adjacency matrix $\boldsymbol{X}$

Inititialize $\boldsymbol{\tau}^{(Y)}$, $\boldsymbol{\tau}^{(Z)}$, $\boldsymbol{\alpha}$, $\boldsymbol{\beta}$, $\boldsymbol{\pi}$

while $\mathcal{J}^{(t)} - \mathcal{J}^{(t-1)} > atol$ **do**

  repeat

    $Q_{il} = \sum_{j:\, X_{ij}=1} \tau_{jl}^{(Z)}$

    $\tau_{iq}^{(Y)} \propto \alpha_q \prod_{jl} (1 - \pi_{ql})^{\tau_{jl}^{(Z)}} \prod_l \frac{\pi_{ql}^{Q_{il}}}{(1-\pi_{ql})^{Q_{il}}}$

    $S_{jq} = \sum_{i:\, X_{ij}=1} \tau_{iq}^{(Y)}$

    $\tau_{jl}^{(Z)} \propto \beta_l \prod_{iq} (1 - \pi_{ql})^{\tau_{iq}^{(Y)}} \prod_q \frac{\pi_{ql}^{S_{jq}}}{(1-\pi_{ql})^{S_{jq}}}$

  until *convergence*;

  $\alpha_q = \frac{\sum_i \tau_{iq}^{(Y)}}{n_1}$ $\qquad$ $\beta_l = \frac{\sum_j \tau_{jl}^{(Z)}}{n_2}$ $\qquad$ $\pi_{ql} = \frac{\sum_{ij:X_{ij}=1} \tau_{iq}^{(Y)} \tau_{jl}^{(Z)}}{\sum_i \tau_{iq}^{(Y)} \sum_j \tau_{jl}^{(Z)}}$

4.2.1.2 Initialization

The optimization process does not ensure convergence towards a global optimum of the criterion $\mathcal{J}(q_\gamma, \theta)$. EM-like algorithms are known to be sensitive to initialization, particularly when applied to models with discrete latent spaces, and may get stuck into unsatisfactory local maxima [Biernacki et al., 2003, Baudry and Celeux, 2015].

A simple heuristic consists in training for a few iterations from several random initializations, and pursuing optimization for the solutions with highest value of the variational criterion [see, e.g., small EM for mixtures Baudry and Celeux, 2015]. Another approach is to rely on cheaper clustering methods, such as k-means or spectral clustering, to initialize the algorithm [Shireman et al., 2015]. These methods bring out good estimates but spend a great deal of computing and memory resources when the data matrices get bigger. Some existing methods such as online k-means [MacQueen, 1967] are adapted to handle large matrices and could be used. However for simplicity reasons, the initialization procedure implemented in **SparseBM** is limited to multiple random initializations.

4.2.1.3 Selection of the number of classes

The Integrated Completed Likelihood criterion (ICL), inspired by the Bayesian Information Criterion, was originally proposed to select a relevant number of classes for mixture models [Biernacki et al., 2000]. It was extended to select an appropriate number of classes in the SBM [Daudin et al., 2008] and in the LBM [Keribin et al., 2012].

The ICL criterion for the standard SBM reads:

$$\begin{aligned} ICL_{SBM}(k_1) &= \log \int p(\boldsymbol{X}, \boldsymbol{Y}|\theta; k_1) p(\theta; k_1) \mathrm{d}\theta \\ &= \max_{\theta} \log p(\boldsymbol{X}, \boldsymbol{Y}; \theta) - \frac{{k_1}^2}{2} \log\left(n_1(n_1-1)\right) - \frac{k_1 - 1}{2} \log n_1 \\ &\quad + o(\log n_1) \ , \end{aligned}$$

with $p(\theta; k_1)$ the prior distribution of parameters as set by Daudin et al. [2008].

The ICL criterion for the standard LBM reads:

$$\begin{aligned} ICL_{LBM}(k_1, k_2) &= \log \int p(\boldsymbol{X}, \boldsymbol{Y}, \boldsymbol{Z}|\theta; k_1, k_2) p(\theta; k_1, k_2) \mathrm{d}\theta \\ &= \max_{\theta} \log p(\boldsymbol{X}, \boldsymbol{Y}, \boldsymbol{Z}; \theta) - \frac{k_1 k_2}{2} \log(n_1 n_2) \\ &\quad - \frac{k_1 - 1}{2} \log n_1 - \frac{k_2 - 1}{2} \log n_2 + o(\log n_1) + o(\log n_2) \ , \end{aligned}$$

with $p(\theta; k_1, k_2)$ the prior distribution of parameters as modeled by Keribin et al. [2012]. In practice, as the log-likelihood maximum can not be computed, its variational approximation is used. By taking into account the latent variables, ICL is clustering-oriented, whereas BIC or AIC are driven by the faithfulness to the distribution of $\boldsymbol{X}$ [Biernacki et al., 2000].

Being dependent on the log-likelihood, the ICL criterion is also sensitive to the VEM solution, and thus to its initialization, which usually leads to an irregular ICL behavior during the exploration of the number of groups. To get a smoother ICL response, **SparseBM** implements a procedure, known as

"split and merge" or "forward and backward" [Tabouy, 2019], that relies on the two alternated strategies to "split" and "merge" groups. Starting from a trained model with k_1 groups, the split strategy explores all models obtained by splitting one of the k_1 groups and keeps the best model estimation in terms of ICL. The split strategy brings out models with more and more groups until no model improves upon the best ICL criterion found so far, and thus for a few iterations. In our implementation the number of groups considered should not exceeds $min(1.5 \cdot n_q^{best}, n_q^{best} + 10)$ with n_q^{best} being the number of groups of the best model found so far in the split strategy. The merge strategy then starts backward, from the model with the highest number of groups, and explores all models obtained by merging two groups. It generates new model estimations with a decreasing number of groups until merging becomes pointless (e.g., from a SBM with only two groups). The split and merge procedure is repeated until no best model estimations comes out for a few iterations (two in the implementation we propose).

4.2.2 Block clustering with SparseBM

SparseBM is a Python module that implements the Bernoulli Latent Block Model and stochastic block model variational inference, optimized for large and sparse graphs. The estimation procedure is fully written with tensor expressions to easily leverage parallel computing. The module can optionally make use of the **CuPy** library that provides GPU accelerated computing. As **CuPy** and **NumPy** share the same interface, only one agnostic code is implemented. The **SparseBM** module is distributed through the PyPI repository (`https://pypi.org/project/sparsebm/`) and the documentation is available at `https://sparsebm.readthedocs.io/`.

4.2.2.1 Installation guidelines

As the module is available through PyPI repository, it can be installed with the package installer **pip**:

```
pip install sparsebm
```

To leverage GPU accelaration, the **CuPy** module must be installed with **pip** or **anaconda** or directly with the extra argument when installing SparseBM:

```
pip install sparsebm[gpu]
```

For users that do not have GPU, we advise the free serverless Jupyter notebook environment provided by Google Colab (`https://colab.research.google.com/`) where the **Cupy** module is already installed and ready to use with one GPU.

4.2.2.2 SparseBM: A Python interface

The main features exposed to the user are:

- `generate_SBM_dataset` and `generate_LBM_dataset`, two functions optimized to generate large and sparse graphs using either the SBM or the LBM;
- `SBM` and `LBM`, two classes implementing the stochastic block model and Latent Block Model inference optimized for sparse graphs and using the multiple random initialisations strategy;
- `ModelSelection` a class implementing the model selection algorithm based on split-merge strategy and making use of the SBM or LBM for inference.

In the following sections, we give more details and provide examples of the use of these algorithm.

Sparse network generation: network generation avoids the manipulation of dense matrices by creating the adjacency matrix $\boldsymbol{X}$ block by block.

The function `generate_SBM_dataset` generates a network from the SBM with a specified number of nodes n_1, a number of classes k_1, class proportions ($\boldsymbol{\alpha} \in \mathbf{S}_{k_1-1}$), and array of connection probabilities ($\boldsymbol{\pi} \in [0,1]^{k_1 \times k_1}$) between classes. The argument `symmetric` indicates wether the adjacency matrix is symmetric, when clustering an undirected graph. The generated sparse adjacency matrix $\boldsymbol{X}$ (from class `scipy.sparse.coo_matrix`) and the generated indicator matrix of the latent classes $\boldsymbol{Y}$ are returned in a dictionary at keys "data" and "cluster_indicator".

```
>>> from sparsebm import generate_SBM_dataset
>>> import numpy as np
>>>
>>> connection_probabilities = np.array(
...     [
...         [0.1, 0.036, 0.012, 0.0614],
...         [0.036, 0.074, 0.0, 0.0],
...         [0.012, 0.0, 0.11, 0.024],
...         [0.0614, 0.0, 0.024, 0.086],
...     ]
... )
>>>
>>> dataset = generate_SBM_dataset(
...     number_of_nodes=10 ** 3,
...     number_of_clusters=4,
...     connection_probabilities=connection_probabilities,
```

```
...     cluster_proportions=np.array([0.25, 0.25, 0.25, 0.25]),
...     symmetric=True,
... )
>>> graph = dataset["data"]
>>> cluster_indicator = dataset["cluster_indicator"]
```

If no argument is given to `generate_SBM_dataset`, a random affiliation graph [Matias and Miele, 2017] is generated:

```
>>> from sparsebm import generate_SBM_dataset
>>> dataset = generate_SBM_dataset()
```

A similar function called `generate_LBM_dataset` generates a bipartite network following the LBM and returns a dictionary that contains the adjacency matrix and the indicator matrices of the row and column latent classes.

Stochastic block model: the SBM is encapsulated in the `SBM` class that inherits from the `sklearn.base.BaseEstimator` that is the base class for all estimators in scikit-learn. A number of classes k_1 should be specified with the parameter `n_clusters`, otherwise the default value 5 is used. If the **Cupy** module is installed, the class uses the GPU with the largest memory available. The parameter `use_gpu` can disable this behaviour and the parameter `gpu_index` can enforce the use of a specific GPU.

The class implements the random initializations strategy that corresponds to the execution of `n_iter_early_stop` EM steps on `n_init` random initializations, followed by iterations until the convergence of the criterion for the `n_init_total_run`-best preliminary results; `n_iter_early_stop`, `n_init` and `n_init_total_run` are parameters of the class.

The convergence of the criterion $\mathcal{J}(q_\gamma, \theta)$ is declared when

$$\mathcal{J}^{(t)}(q_\gamma, \theta) - \mathcal{J}^{(t-5)}(q_\gamma, \theta) \leq (atol + rtol \cdot |\mathcal{J}^{(t)}(q_\gamma, \theta)|) \ ,$$

with $atol = 1e-4$ and $rtol = 1e-10$ being respectively the absolute tolerance and the relative tolerance.

```
>>> from sparsebm import SBM
>>> model = SBM(
...     n_clusters=4,
...     max_iter=10000,
...     n_init=100,
...     n_init_total_run=10,
...     n_iter_early_stop=10,
...     rtol=1e-10,
...     atol=1e-4,
...     verbosity=1,
...     use_gpu=True,
```

```
...     gpu_index=0,
... )
```

The class implements a `fit` method to learn from the adjacency matrix of a graph, being either sparse (class `scipy.sparse`) or not (class `numpy.array`):

```
>>> model.fit(graph)

---------- START RANDOM INITIALIZATIONS ----------
100 of 100 Initializations:  [100% ] [ Elapsed Time: 0:00:03 ]
---------- START TRAINING BEST INITIALIZATIONS ----------
10 of 10 Runs:  [100% ] [ Elapsed Time: 0:00:01 ]
```

When successfully inferred, the class proportions $\boldsymbol{\alpha}$ of the SBM, the array $\boldsymbol{\pi}$ of connection probabilities and the labels of the classes are available by the model properties `group_membership_probability`, `group_connection_probabilities` and `labels`. The Integrated Completed Loglikelihood can be computed with the method `get_ICL`. The inferred labels can be compared with the true ones using the adjusted rand index [Hubert and Arabie, 1985] that computes a similarity measure between two clusterings:

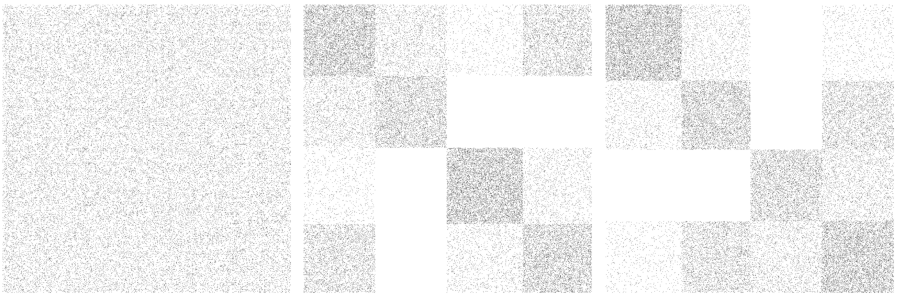
```
>>> from sparsebm.utils import ARI
>>> ari = ARI(cluster_indicator.argmax(1), model.labels)
>>> print("Adjusted Rand index is {:.2f}".format(ari))
>>> print("ICL is {:.4f}".format(model.get_ICL()))

Adjusted Rand index is 1.00
ICL is -74473.8386
```

The function `reorder_rows` reorders the rows of a sparse matrix enabling an easy visualization (see Figure 4.2) of the adjacency matrix reordered according to the estimated or true classes:

```
>>> from sparsebm.utils import reorder_rows
>>> reorder_rows(graph, np.argsort(model.labels))
>>> graph = graph.transpose()
>>> reorder_rows(graph, np.argsort(model.labels))
>>> graph = graph.transpose()
```

Latent block model: the `LBM` class encapsulates the Latent Block Model and its random initialisation procedure. Its usage is similar to the `SBM` class and we refer the reader to the documentation of the **SparseBM** module or examples for more details. To measure the agreement between co-clustering partitions, the module proposes an implementation of the co-clustering adjusted rand index (CoARI) [Robert et al., 2020], which is an extension of the adjusted rand index for co-clustering.



(a) (b) (c)

Figure 4.2: Adjacency matrix of a network with $n_1 = 1000$ nodes generated by a SBM. The size of black pixels representing edges is enlarged for visualization reasons: (a) original adjacency matrix, (b) adjacency matrix reordered according to the true classes, (c) adjacency matrix reordered according to the classes returned by inference. Note that the permutation of classes observed between (b) and (c) is irrelevant for clustering purposes.

Model selection: the `ModelSelection` class encapsulates the model selection algorithm based on the split and merge strategy. The argument `model_type` specifies the model to use and `n_clusters_max` specifies the upper bound on the number of groups the algorithm can explore. The split strategy stops when the number of classes is greater than $min(1.5 \cdot nnq\_best, nnq\_best + 10, n\_clusters\_max)$ with nnq_best being the number of classes of the best model found so far during the split strategy. The merge strategy stops when the minimum relevant number of classes is reached. The split and merge strategy alternates until no best model is found for two iterations.

The argument `plot` specifies if an illustration is displayed to the user during the learning process (see Figure 4.3).

```
>>> from sparsebm import ModelSelection
>>> sbm_model_selection = ModelSelection(
...     model_type="SBM",
...     n_clusters_max=30,
...     plot=True,
...     use_gpu=True,
...     gpu_index=None,
... )
```

To learn from a sparse network, the class implements the `fit` method and returns the best model found.

```
>>> sbm_selected = sbm_model_selection.fit(graph, symmetric=True)
>>> number_of_clusters = dataset['cluster_indicator'].shape[1]
>>> print(f"Best ICL is {sbm_selected.get_ICL():.4f}")
```

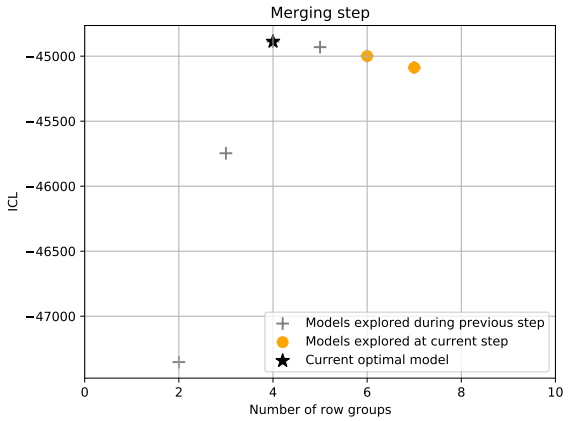


Figure 4.3: Illustration displayed during model selection with merge and split strategy for a SBM.

```
>>> print(f"The original number of classes was {number_of_clusters}")
>>> print(f"The model selection picked {sbm_selected.n_clusters} classes")

Best ICL is -44162.1115
The original number of classes was 4
The model selection picked 4 classes
```

Sckikit-learn integration: the SBM and LBM implemented in **Sparsebm** use the **Scikit-learn** interface style; models are thus compatible with the pipelines, model selection, and evaluation metrics. We illustrate the integration with **Scikit-learn** with a gridsearch algorithm to select the best number of classes. In this example, the `GridSearchCV` instance receives the `SBM` model and runs the algorithm with the numbers of classes specified. The model are compared together with the Integrated Completed Likelihood criterion implemented in the `SBM` model. A number of jobs to run in parallel is specified with the argument `n_jobs`; the **SparseBM** module is using all GPUs available in the system. The number of jobs in parallel should never be higher that the number of GPUs in the system.

```
>>> from sparsebm import SBM
>>> import sklearn
>>> from sklearn import metrics
>>>
>>> graph = dataset["data"]
>>> clusters_index = dataset["cluster_indicator"].argmax(1)
>>> number_of_nodes = graph.shape[0]
>>>
```

```
>>> model = SBM(verbosity=0)
>>> train = test = np.arange(number_of_nodes)
>>> n_clusters = [1, 2, 3, 4, 5, 6, 7, 8]
>>> grid = sklearn.model_selection.GridSearchCV(
...     estimator=model,
...     n_jobs=4,
...     param_grid={"n_clusters": n_clusters},
...     cv=[[train, test]],
...     verbose=1,
... )
>>> print("Start grid search algorithm")
>>> grid.fit(graph, symmetric=True)
>>> ari = metrics.adjusted_rand_score(
...     clusters_index, grid.best_estimator_.labels
... )
>>> print(
...     "Best number of classes is {} according to ICL".format(
...         grid.best_params_["n_clusters"]
...     )
... )
```

```
Start grid search algorithm
Fitting 1 folds for each of 8 candidates, totalling 8 fits
[Parallel(n_jobs=4)]: Using backend LokyBackend with 4 concurrent workers.
[Parallel(n_jobs=4)]: Done   8 out of   8 | elapsed:   22.4s finished
Best number of classes is 4 according to ICL
Adjusted Rand Index is 0.97891220269305
```

4.2.2.3 SparseBM: A command line interface

The **SparseBM** module comes with a command line interface to run the LBM and SBM inference and to generate networks. The SBM/LBM model selection algorithm to chose the best number of classes according to the ICL criterion is available. The command `sparsebm` must be followed by the positional argument `sbm` or `lbm` or `modelselection` or `generate` to use respectively the stochastic block model inference or the Latent Block Model inference or the model selection algorithm or to generate a network with one of these models.

Latent block model: `sparsebm lbm` command line returns a JSON file that contains the two partitions and the estimated parameters of the model. The usage of the command is detailed below:

```
sparsebm lbm --help
```

```
usage: sparsebm lbm [-h] [-k1 N_ROW_CLUSTERS] [-k2 N_COLUMN_CLUSTERS]
                    [-o OUTPUT] [-sep SEP] [-niter MAX_ITER]
                    [-ninit N_INIT] [-early N_ITER_EARLY_STOP]
                    [-ninitt N_INIT_TOTAL_RUN] [-t TOL] [-v VERBOSITY]
                    [-gpu USE_GPU] [-idgpu GPU_INDEX] ADJACENCY_MATRIX

optional arguments:
  -h, --help            show this help message and exit

mandatory arguments:
  ADJACENCY_MATRIX      List of edges in CSV format
  -k1 N_ROW_CLUSTERS, --n_row_clusters N_ROW_CLUSTERS
                        number of row clusters
  -k2 N_COLUMN_CLUSTERS, --n_column_clusters N_COLUMN_CLUSTERS
                        number of row clusters

output:
  -o OUTPUT, --output OUTPUT
                        File path for the json results.

optional arguments:
  -sep SEP, --sep SEP   CSV delimiter to use. Default is ','
  -niter MAX_ITER, --max_iter MAX_ITER
                        Maximum number of EM step
  -ninit N_INIT, --n_init N_INIT
                        Number of initializations that will be run
  -early N_ITER_EARLY_STOP, --n_iter_early_stop N_ITER_EARLY_STOP
                        Number of EM steps to perform for each initialization.
  -ninitt N_INIT_TOTAL_RUN, --n_init_total_run N_INIT_TOTAL_RUN
                        Number of the best initializations that will be run
                        until convergence.
  -t TOL, --tol TOL     Tolerance of likelihood to declare convergence.
  -v VERBOSITY, --verbosity VERBOSITY
                        Degree of verbosity. Scale from 0 (no message
                        displayed) to 3.
  -gpu USE_GPU, --use_gpu USE_GPU
                        Specify if a GPU should be used.
  -idgpu GPU_INDEX, --gpu_index GPU_INDEX
                        Specify the gpu index if needed.
```

Stochastic block model: `sparsebm sbm` command line returns a JSON file that contains the partition and the estimated parameters of the model. A

summary of the usage of the command is given:

sparsebm sbm --help

```
usage: sparsebm sbm [-h] [-sep SEP] [-o OUTPUT] [-k N_CLUSTERS]
                    [-s SYMMETRIC] [-niter MAX_ITER]
                    [-ninit N_INIT][-early N_ITER_EARLY_STOP]
                    [-ninitt N_INIT_TOTAL_RUN] [-t TOL]
                    [-v VERBOSITY] [-gpu USE_GPU] [-idgpu GPU_INDEX]
                    ADJACENCY_MATRIX
```

Model selection: `sparsebm modelselection sbm` or `sparsebm modelselection lbm` command line returns a JSON file that contains the partition (two if LBM is used) and the estimated parameters of the best model found.

sparsebm modelselection --help

```
usage: sparsebm modelselection [-h] -t TYPE [-gpu USE_GPU]
                               [-idgpu GPU_INDEX][-s SYMMETRIC]
                               [-p PLOT] [-o OUTPUT]
                               ADJACENCY_MATRIX
```

Graph generation: `sparsebm generate` command line returns a JSON file that contains the partition and a CSV file that contains the adjacency list of the graph. A summary of the usage of the command is given:

sparsebm generate --help

```
positional arguments:
  {sbm,lbm}   model to generate data with
    sbm       use the stochastic block model to generate data
    lbm       use the latent block model to generate data
```

Example: with the two following commands, a network is generated and trained with a SBM using the model selection algorithm:

sparsebm generate sbm
sparsebm modelselection edges.csv -t=sbm

```
---------- START Graph Generation ----------
25 of 25 Generating block:  [100% ]
Groups and params saved in ./groups.json
Edges saved in ./edges.csv
Spliting
```

```
        Explore models from 1 classes
        ...
Merging
        Explore models from 5 classes
        ...
Best icl is -53481.1475
Model has been trained successfully.
Value of the Integrated Completed Loglikelihood is -53481.1475
The model selection picked 3 classes
Results saved in results.json
```

4.2.3 Experiments and discussion

4.2.3.1 Benefit of our inference optimized for sparse graphs.

We compare our inference optimized for sparse graphs to the original inference designed for dense graphs. We provide here experiments on the latent block model only, similar results can be obtained for the stochastic block model.

Fixed graph size, varying sparsity A network is generated following an LBM with $n_1 = 10\,000$ nodes of type (1) equally divided in three classes and $n_2 = 5\,000$ nodes of type (2) equally divided in four classes, with parameters

$$\boldsymbol{\alpha} = \begin{pmatrix} 1/3 \\ 1/3 \\ 1/3 \end{pmatrix} \quad \text{and} \quad \boldsymbol{\beta} = \begin{pmatrix} 1/4 \\ 1/4 \\ 1/4 \\ 1/4 \end{pmatrix} \quad \text{and} \quad \boldsymbol{\pi} = 2^{-\varepsilon} \cdot \begin{pmatrix} 1 & 1/4 & 1/4 & 1/2 \\ 1/4 & 1/4 & 1/4 & 1/4 \\ 1/2 & 1/4 & 1/2 & 1/2 \end{pmatrix}, \tag{4.5}$$

where $\varepsilon \in \{1, \cdots, 6\}$ defines the sparsity level of the graph. For each value of ε, a network is generated using these model parameters; the size of the generated networks is fixed, and their sparsity increases with ε.

The model parameters are estimated for each network using the original variational inference and the one optimized for sparse graphs (Algorithms 6 and 7, respectively). This process is repeated 100 times for each graph size.

The medians of the computation times are presented in Figure 4.4 as a function of the sparsity of the graph (that is, one minus the expected ratio of actual edges to the $n_1 \times n_2$ edges of the complete bipartite graph). The execution times reported here correspond to the overall estimation protocol, that is, (i) 20 EM steps from 100 random initializations, followed by (ii) iterations until convergence of the criterion for the 10 best results reached after these 20 initial steps (see Section 4.2.2). The architecture used is a NVIDIA DGX Server with a Tesla V100-SXM2-32GB GPU.

The execution times of the original inference ($\star$ in Figure 4.4) are nearly

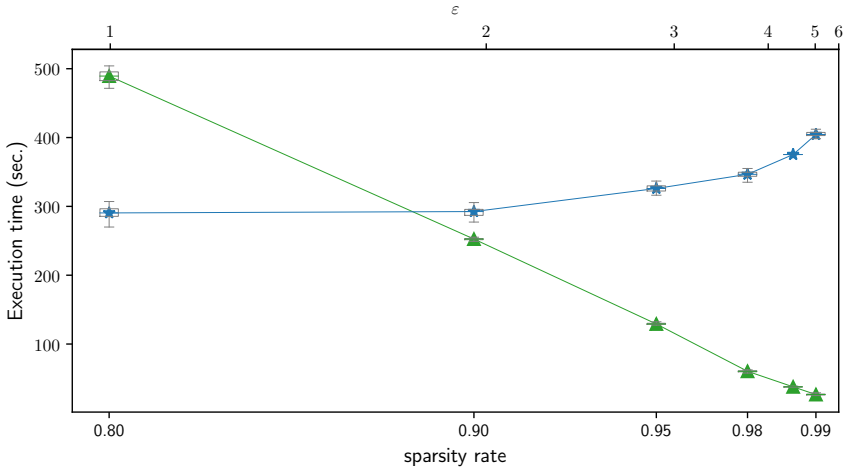


Figure 4.4: Median computation times for inferring the parameters of a Latent Block Model as a function of the sparsity of the bipartite graph (size $10\,000 \times 5\,000$); ▲ is for the algorithm optimized for sparse graph; ⋆ is for the original algorithm.

constant, except for high sparsity levels, where the difficulty of estimation is increased, requiring more EM steps to reach convergence. For our inference optimized for sparse graphs (▲ in Figure 4.4), the quasi-linear trend of execution times in relation to the sparsity rate gives an experimental confirmation of the $\mathcal{O}(\#\{ij : X_{ij} = 1\}k_1k_2)$ computational complexity.

Fixed graph sparsity, varying size A second series of network is generated following the LBM, using the parameters of the previous experiment, except that ε is now fixed to 5, leading to a sparsity rate of 98.76%, and that the sizes of the bipartite graph, n_1 and n_2, vary. The model parameters are estimated for each network using the original variational inference and the one optimized for sparse graphs. The random initialization strategy and the hardware architecture used are as in the previous experiments.

The medians of the computation times are reported in Figure 4.5 as a function of the size of the bipartite graph $n_1 \times n_2$. Using the original inference (⋆ in Figure 4.5), the GPU is out of memory (OOM) with graphs bigger than $10\,000 \times 5\,000$ due to the $\mathcal{O}(n_1n_2)$ memory complexity of the algorithm. The inference implemented in **SparseBM** (▲ in Figure 4.5) can be applied to much bigger graphs as its memory complexity is in $\mathcal{O}(\#\{ij : X_{ij} = 1\}$ and gets some execution times scaling linearly with the size of the graphs.

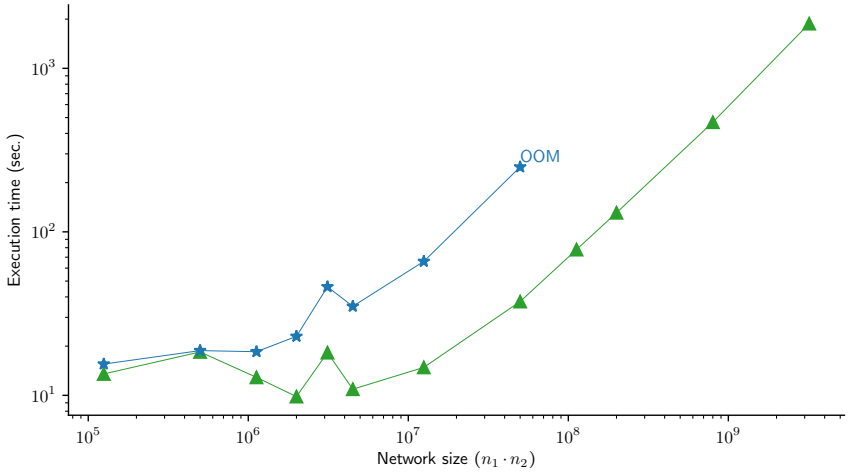


Figure 4.5: Median computation times for inferring the parameters of a Latent Block Model as a function of the graph size $n_1 \times n_2$ (fixed sparsity rate of 98.76%); ▲ is for the algorithm optimized for sparse graph, ⋆ is for the original algorithm.

4.2.3.2 Comparing SparseBM with existing R packages.

We compare the LBM inference from **SparseBM**, **Blockcluster** [Bhatia et al., 2017] and **Blockmodels** [Leger, 2016], using the previous experimental setup with varying graph size.

For a fair comparison between packages, the architecture used is an Intel Xeon Gold 6138 CPU (2.00GHz) with 16 GB RAM (**Blockcluster** and **Blockmodels** are not designed for GPU). Due to this limited computation power, we lighten the previous optimization protocol: we still use 100 random initializations, but they are only updated for 10 EM steps (a single step for **Blockmodels** as this number is hard-coded); then, only the (single) best initialization is selected to pursue until the convergence of the criterion.

The medians on a hundred repetitions of the execution times are reported in Figure 4.6. The algorithms from **Blockmodels** and **Blockcluster** are saturating the RAM memory with networks of sizes respectively $(15\,000 \times 7\,500)$ and $(40\,000 \times 20\,000)$, while the implementation of **SparseBM** allows bigger networks as shown in Section 4.2.3.1. Note that the limited memory footprint of **SparseBM** provided by the sparse reformulation of the inference is essential to reach the low computation times (real elapsed time) with GPU (• in Figure 4.6). Indeed, using LBM on large networks would not be possible otherwise due to the very limited memory size available in common GPUs.

We verify that the solutions obtained by the different packages are of comparable accuracy by calculating their similarity with the true generated coclustering. To measure this similarity, we use the coclustering adjusted

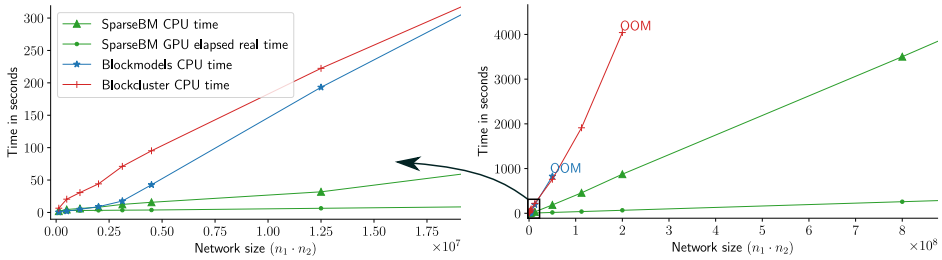


Figure 4.6: Median computation times (CPU), using existing implementations for inferring the parameters of a Latent Block Model as a function of the graph size $n_1 \times n_2$ with a fixed sparsity rate of 98.76%; ▲ is for **SparseBM**, ⋆ is for **Blockmodels** and + is for **Blockcluster**. The median of the computation real elapsed time using **SparseBM** with GPU is also displayed (•) for reference. The graphic on the left zooms in on smaller networks.

rand index scores (CoARI) [Robert et al., 2020], whose median values are reported in Table 4.1. The scores increase for bigger networks as the inference problems gets easier; the scores for **SparseBM** and **Blockcluster** are similar, and we suppose that the poorer performance of **Blockmodels** is mainly due to the lighter initialization procedure.

Table 4.1: Median of the coclustering adjusted rand index (CoARI, a similarity measure between two coclusterings), using existing implementations, as a function of the graph size $n_1 \times n_2$ with a fixed sparsity rate of 98.76%.

	CoARI measured with packages		
Network size $(n_1 \cdot n_2)$	**SparseBM**	**Blockcluster**	**Blockmodels**
1.25×10^5	0.05	0.05	0.05
5.00×10^5	0.11	0.11	0.09
1.13×10^6	0.18	0.18	0.12
2.00×10^6	0.26	0.26	0.15
3.13×10^6	0.33	0.32	0.18
4.50×10^6	0.41	0.41	0.20
1.25×10^7	0.68	0.68	0.25
5.00×10^7	0.93	0.93	0.30
1.13×10^8	0.98	0.98	OOM
2.00×10^8	1.00	1.00	OOM

4.2.4 Conclusion

SparseBM is a Python module for estimating Bernoulli block models in large and sparse networks, relying on the stochastic and latent block models. After a brief review of the mathematical foundations of these models, we presented the details of the calculations that are used in this package to reduce the complexity of the original formulation of the variational inference.

These computation tricks enable the modeling of large sparse networks for which computational and memory requirements prohibit the use of the original approach.

We presented the command line interface and the **Scikit-learn** compatible Python API of the module through examples, and we conducted experiments on synthetic datasets showing that this inference is computationally efficient, enabling to analyze many more networks in a given computation time, and more importantly, much larger sparse networks than the ones that can be handled by current packages. In future releases of the **SparseBM** module, we plan to extend the models to other probability distributions that may result in sparse graphs, such as the zero-inflated Poisson.

4.3 Stochastic variational inference for large and complex graph models

When graphs are dense and large, handling lists of adjacency instead of adjacency matrices is no longer a good solution to efficiently conduct inference with the block models. Additionally, when using more complex graph models, the computational tricks to scale up to a very large number of nodes is impossible to apply without any further approximation (as shown in Appendix 2.B). In this section, a stochastic optimization algorithm for mean-field variational inference is derived for large graph and/or complex models.

4.3.1 Stochastic gradient descent on variational criterion

Gradient descent is a way to minimize an objective function by updating the parameters in the opposite direction of the gradient of the objective function with respect to the parameters. The alternated double maximization performed in the variational EM algorithm (see Section 2.2.2) can be realized using gradient descent:

E-step : update the variational parameters $\boldsymbol{\gamma}$ until convergence of the criterion $\mathcal{J}(\boldsymbol{X};\boldsymbol{\theta},\boldsymbol{\gamma})$:

$$\boldsymbol{\gamma}^{(t+1)} = \boldsymbol{\gamma}^{(t)} + \eta^{(t)} \cdot \nabla_{\boldsymbol{\gamma}} \mathcal{J}(\boldsymbol{X};\boldsymbol{\theta},\boldsymbol{\gamma})$$

M-step : update the model parameters $\boldsymbol{\theta}$ until convergence of the criterion $\mathcal{J}(\boldsymbol{X};\boldsymbol{\theta},\boldsymbol{\gamma})$:

$$\boldsymbol{\theta}^{(t+1)} = \boldsymbol{\theta}^{(t)} + \eta^{(t)} \cdot \nabla_{\boldsymbol{\theta}} \mathcal{J}(\boldsymbol{X};\boldsymbol{\theta},\boldsymbol{\gamma}) \ ,$$

with $\eta^{(t)} \in \mathbb{R}_+$ the learning rate that may change over time.

Apart from the fact that all datasets may not fit in memory, this strategy is inefficient [Hoffman et al., 2013] for large datasets because all the variational parameters $\boldsymbol{\gamma}$ (whose number scales with the size of the dataset) must be optimized before re-estimating the model parameters $\boldsymbol{\theta}$.

In stochastic gradient descent, those drawbacks are naturally removed as the gradient is approximated by a gradient computed on a single example randomly sampled from the data. In a co-clustering setting with block models, each entry X_{ij} of the data matrix is considered to be an example. The whole data matrix is no longer required to be stored in memory as only sampled entries are loaded on the fly. A necessary condition to use stochastic gradient descent is to re-write the variational criterion as a sum on all values of the

data matrix: $\mathcal{J}(\boldsymbol{X};\boldsymbol{\theta},\boldsymbol{\gamma}) = \sum_{ij}\mathcal{J}(X_{ij};\boldsymbol{\theta},\boldsymbol{\gamma})$ (see example on the binary Latent Block Model below).

The parameters of the objective function are updated after each noisy estimation of the gradient computed with a single training example. In the case of the variational expectation maximization algorithm and by using *adaptive learning rates* on each parameters, the alternated double maximization can be replaced by a single maximization with respect to all parameters.

$$\left(\boldsymbol{\gamma}^{(t+1)},\boldsymbol{\theta}^{(t+1)}\right) = \left(\boldsymbol{\gamma}^{(t)},\boldsymbol{\theta}^{(t)}\right) + \boldsymbol{\eta}\cdot\nabla_{\boldsymbol{\gamma},\boldsymbol{\theta}}\mathcal{J}(X_{ij};\boldsymbol{\theta},\boldsymbol{\gamma}) \ ,$$

with $\boldsymbol{\eta}$ the per-parameter vector of learning rate. The *adaptive moment estimation* (Adam, [Kingma and Ba, 2014]) is an example of algorithm that computes adaptive learning rates for each parameter from estimates of first and second moments of the gradients. When using stochastic gradient descent with adaptive learning rates on block models, the learning rates of the model parameters θ are typically slower than the ones of the variational parameters γ which prevent the divergence of the variational criterion.

Mini-batch gradient descent combines the advantages of both gradient descent and stochastic gradient descent by estimating the gradient with several examples sub-sampled from the data. By updating less often the parameters, the convergence rate increases and the computational expenses are reduced. Estimating the gradient with more examples also help to get a smoother convergence as the variance of the gradient is reduced. Figure 4.7 illustrates a co-clustering setting with block models, where a mini-batch of data, $\boldsymbol{X}^{p}_{(i:i+m_1),(j:j+m_2)}$, consists of a $(m_1 \times m_2)$ sub-matrix sampled from the original matrix $\boldsymbol{X}$ whose rows and columns are randomly permuted at each epoch. The parameters are updated using the gradient estimated on the sub-matrix:

$$\left(\boldsymbol{\gamma}^{(t+1)},\boldsymbol{\theta}^{(t+1)}\right) = \left(\boldsymbol{\gamma}^{(t)},\boldsymbol{\theta}^{(t)}\right) + \boldsymbol{\eta}\cdot\nabla_{\boldsymbol{\gamma},\boldsymbol{\theta}}\mathcal{J}\left(\boldsymbol{X}^{p}_{(i:i+m_1),(j:j+m_2)};\boldsymbol{\theta},\boldsymbol{\gamma}\right) \ .$$

Stopping conditions: Using stochastic or mini batch gradient descent rather than expectation maximization ultimately complicates convergence detection as the criterion is not ensured to improve after each step. Although the use of adaptive learning rates helps to reduce the oscillations of the criterion, a Savitzky–Golay filter [Savitzky and Golay, 1964] can be used for the purpose of smoothing the curve of the optimization criterion. Then, a possible solution is to slowly decrease the learning rate when the smoothed optimization curve has stopped improving over time and completely stop the algorithm when the learning rate reaches a minimum threshold (10^{-6} in our experiments). We empirically observed that estimates often benefit from reducing the learning rate by a factor 5 once the variational criterion has been stagnating for around 20 epochs.

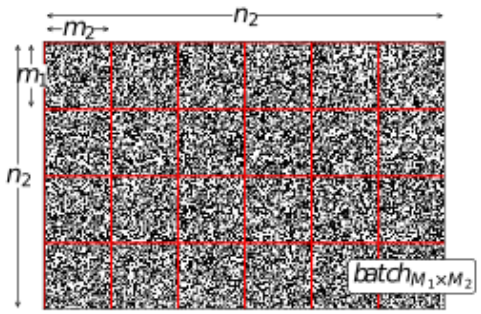


Figure 4.7: Sampling a mini-batch in a co-clustering setting consists of randomly sample a $(m_1 \times m_2)$ sub-matrix of the original $(n_1 \times n_2)$ matrix $\boldsymbol{X}$. After each epoch, the sub-matrices are re-make up by randomly swapping rows and columns of the original matrix.

4.3.2 Numerical estimation of analytic criterion

When handling complex block models, getting an analytical expression of the variational criterion may be difficult or event impossible without further approximations. The inference of the Latent Block Model handling MNAR data from Section 2.2.1.2 uses a Taylor approximation. The Overlapping Stochastic Block Model [Latouche et al., 2011] needs a second level of approximation (the ξ-transformation) as the variational criterion involves an expectation with a nonlinear function. The Bernoulli Latent Block Model using covariates requires a polynomial approximation to get acceptable inference times (see execution times in [Leger, 2016]). All these approximations complicate the final optimization criterion and often lead to expectation-maximization steps where a numerical maximization algorithm is finally necessary as no explicit formula can be found.

With such models, we side-step this problem by replacing the analytical expression of the intractable terms with their numerical estimate. In the models cited above, the non tractable term is the expectation of the conditional log-likelihood of the observed data: $\mathbb{E}_{q_\gamma}[\log p(\boldsymbol{X}|L)]$ where q_γ is the variational distribution and L gathers the latent variables of the model. The estimate of a noisy but unbiased gradient of this expectation via Monte Carlo samples from q_γ can be optimized with a stochastic gradient descent:

$$\mathbb{E}_{q_\gamma}[\log p(\boldsymbol{X}|L)] \approx \frac{1}{N}\sum_{n}^{N} \log p(\boldsymbol{X}|L_n^*) \ , \tag{4.6}$$

where L_n^* is the n^{th} sample from the variational distribution. In the limit, ones can only use a single sample L^* as a very noisy approximate of the expectation.

However, to optimize a numerical estimation of (4.6), we need estimates of the gradients with respect to the variational distribution parameters which is in practice not possible. The *reparameterization trick* [Kingma and Welling, 2014] restates the problem for continuous random variables, in such a way that the gradient is estimated with respect to the parameters of a deterministic function. For example, to get the gradient of x sampled from a Gaussian distribution $\mathcal{N}(\mu, \sigma^2)$, one can restate the problem to compute the gradient with respect to the parameters of the deterministic function $f(x) = \mu + \sigma\varepsilon$ where ε is an auxiliary noise variable $\varepsilon \sim \mathcal{N}(0, 1)$. This trick can be used in settings where a tractable inverse cumulative distribution function exists (e.g., Cauchy, Pareto, Gumbel, Weibull, etc.) or with distributions from the location-scale family (e.g., Gaussian, Student, Uniform, Laplace, etc.). However, difficulties arise when the model uses discrete latent variables such as in the Latent Block Models because their distributions are not reparameterizable.

Jang et al. [2016] use the Gumbel-softmax distribution, a continuous distribution over the simplex that approximates samples from a categorical or Bernoulli distribution and whose parameter gradients can be easily computed via the reparameterization trick. The Gumbel-softmax distribution generates sample vectors $\boldsymbol{y} \in S_{(k-1)}$ based on the softmax function and the Gumbel(0, 1) distribution:

$$y_q = \frac{\exp((\log \alpha_q + g_q)/\tau)}{\sum_l \exp((\log \alpha_l + g_l)/\tau)} \quad \text{for } q = 1, \ldots, k$$

with $g_1, \ldots, g_k$ are i.i.d samples drawn from Gumbel(0,1) and $\alpha_1, \ldots, \alpha_k$ are the class probabilities. The density of the Gumbel-softmax distribution:

$$p(\boldsymbol{y}, \boldsymbol{\alpha}, \tau) = \Gamma(k)\tau^{k-1}\left(\sum_q \alpha_q / y_q^{\tau}\right)^{-k} \prod_q \left(\alpha_q / y_q^{\tau+1}\right)$$

is characterized by a temperature τ. Samples from Gumbel-softmax approach samples from a categorical distribution as $\tau \to 0$ but also samples from a uniform distribution as $\tau \to \infty$. Using the Gumbel-softmax distribution when doing a stochastic inference in the block models allow gradients to flow through the differentiation graph built by automatic differentiation libraries.

Example on the binary Latent Block Model: To optimize the variational criterion of the binary Latent Block Model (see Section 1.2.2) with

stochastic gradient descent, the criterion is expressed on a batch:

$$\mathcal{J}\big(\boldsymbol{X}_{(i:i+m_1),(j:j+m_2)};\gamma,\theta\big) = \mathbb{E}_{q_\gamma}\big[\mathcal{L}\big(\boldsymbol{X}_{(i:i+m_1),(j:j+m_2)}\big|\boldsymbol{Y}_{i:i+m_1},\boldsymbol{Z}_{j:j+m_2}\big)\big] \tag{4.7}$$

$$+\frac{m_2}{n_2}\big(\mathcal{H}(q_\gamma(\boldsymbol{Y}_{i:i+m_1})) + \mathbb{E}_{q_\gamma}[\mathcal{L}(\boldsymbol{Y}_{i:i+m_1})]\big) \tag{4.8}$$

$$+\frac{m_1}{n_1}\big(\mathcal{H}(q_\gamma(\boldsymbol{Z}_{j:j+m_2})) + \mathbb{E}_{q_\gamma}[\mathcal{L}(\boldsymbol{Z}_{j:j+m_2})]\big)\ , \tag{4.9}$$

with (n_1, n_2) and (m_1, m_2) the dimensions of respectively the original data matrix $\boldsymbol{X}$ and the sub-sampled data matrix, and q_γ being the variational distribution over the latent variables $\boldsymbol{Y}$ and $\boldsymbol{Z}$:

$$q_\gamma = \prod_i \mathcal{M}\big(1;\boldsymbol{\tau}_i^{(Y)}\big)\ \prod_j \mathcal{M}\big(1;\boldsymbol{\tau}_j^{(Z)}\big)\ .$$

Terms from (4.8) and (4.9) are detailed in Equation 4.4 from Section 4.2.1.1. They are normalized to the batch size. The expectation of the conditional log-likelihood of the observation (right-hand term of 4.7), is numerically estimated (Equation 4.6) by sampling approximately from q_γ using the Gumbel-softmax distribution. This numerical estimation only serves as an illustration of the methodology as the analytical expression of this expectation is here easy to get.

4.3.3 Avoiding poor local minima with stochastic gradient descent

The EM-like algorithms are known to be sensitive to the initialization and may get stuck in unsatisfactory local maxima [Biernacki et al., 2003]. Strategies consisting in exploring from many random initializations, spend a great deal of computing resources to bring out only a few good estimates.

We test here the ability of the stochastic gradient descent inference scheme to bring out good estimates without using a costly initialization strategy. To do that, we compare the row and column classes recovering when using the proposed inference scheme from Section 4.3.2 against the standard variational-EM inference designed by Govaert and Nadif [2008]. We provide here experiments on the binary latent block model only, similar results can be obtained for the stochastic block model.

To conduct the experiments, we generate data matrices from the binary Latent Block Model(Section 1.2.2) , with size $n_1 = n_2 = 1,000$ and four row

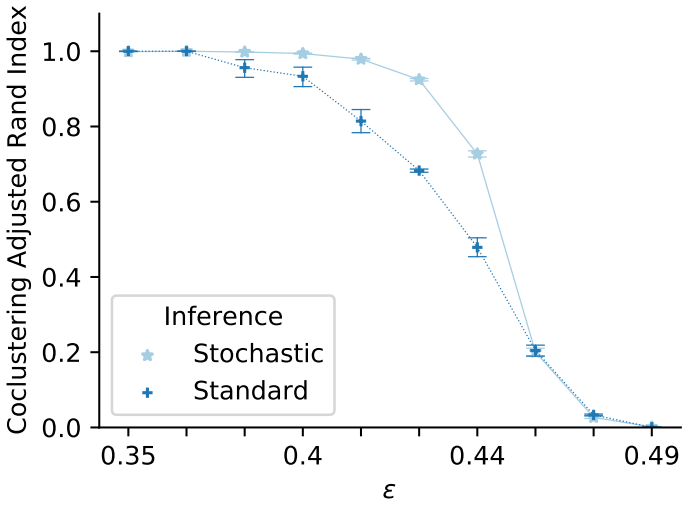


Figure 4.8: Median of the coclustering adjusted rand index (CoARI, a similarity measure between two coclusterings), using the stochastic and the standard variational inference of the Latent Block Model, as a function of the difficulty of the coclustering task.

and four column classes, with parameters:

$$\boldsymbol{\alpha} = \begin{pmatrix} 1/4 \\ 1/4 \\ 1/4 \\ 1/4 \end{pmatrix} \quad \text{and} \quad \boldsymbol{\beta} = \begin{pmatrix} 1/4 \\ 1/4 \\ 1/4 \\ 1/4 \end{pmatrix} \quad \text{and} \quad \boldsymbol{\pi} = \begin{pmatrix} \varepsilon & \varepsilon & \varepsilon & 1-\varepsilon \\ \varepsilon & \varepsilon & 1-\varepsilon & \varepsilon \\ \varepsilon & 1-\varepsilon & 1-\varepsilon & \varepsilon \\ \varepsilon & \varepsilon & \varepsilon & \varepsilon \end{pmatrix},$$

and an increasing difficulty of the clustering task as ε gets closer to 0.5. The matrix $\boldsymbol{\pi}$ is structured hard enough such that using simple clustering algorithms based on the the graph degrees would not give satisfying results.

The model parameters are estimated for each data matrix using the original variational-EM inference (see Section 4.2.1.1) and the one using a mini-batch stochastic gradient descent on the variational criteria.

With the original variational-EM inference, we used an initialization protocol, that is, (i) 10 EM steps from 100 random initializations, followed by (ii) iterations until convergence of the criterion for the 10 best results reached after these 10 initial steps. With the stochastic gradient descent inference, no restart strategy is used and a single random initialization is performed.

With both inferences, we then predict the row and column classes $(\boldsymbol{Y},\boldsymbol{Z})$ with their maximum a posteriori estimators on the variational distribution and we calculate their similarity with the true generated coclustering. To measure this similarity, we use the coclustering adjusted rand index scores (CoARI) [Robert et al., 2020]. This whole process is repeated 50 times, leading to the results presented in Figure 4.8.

The dominance of the coARI curve of the stochastic gradient descent optimization expresses that this procedure is less inclined to be trapped is poor local minima. Even though a better initialization scheme could be found relying on simple clustering methods, such as k-means or spectral clustering, to initialize the standard variational EM algorithm [Shireman et al., 2015], avoiding this initialization step to alleviate the whole training procedure is an undeniable advantage of the stochastic gradient descent optimization.

4.4 Conclusion

We have proposed calculation tricks to reduce the complexity of the original algorithm of inference of Bernoulli block models, enabling the modeling of large and sparse networks. To have a more general inference procedure, we proposed to use a stochastic optimization algorithm for mean-field variational inference in block models.

Our algorithm enables to process large and dense networks because the whole data matrix no longer needs to be loaded into memory. When using elaborate block models, some non tractable analytical expressions can be replaced with their numerical estimates using a sampling step. The stochasticity of the gradients helps the algorithm to get out of poor local minima, thereby avoiding the use of an initialization procedure to bring out better estimates.

Bibliography

Julie Aubert, Sophie Schbath, and Stephane Robin. Latent block model for metagenomic data. European Conference on Computational Biology (ECCB 2016), Workshop "Recent Computational Advances in Metagenomics (RCAM)", September 2016. URL https://hal.archives-ouvertes.fr/hal-01661258. 129

Avner Bar-Hen, Pierre Barbillon, and Sophie Donnet. Block models for multipartite networks. applications in ecology and ethnobiology, 2020. 129

Jean-Patrick Baudry and Gilles Celeux. EM for mixtures. *Statistics and Computing*, 25(4):713–726, 2015. doi: 10.1007/s11222-015-9561-x. 134, 135

Parmeet Singh Bhatia, Serge Iovleff, and Gérard Govaert. Blockcluster: An r package for model-based co-clustering. *Journal of Statistical Software*, 76 (9):1–24, 2017. ISSN 1548-7660. doi: 10.18637/jss.v076.i09. URL https://www.jstatsoft.org/v076/i09. 129, 147

Christophe Biernacki, Gilles Celeux, and Gérard Govaert. Assessing a mixture model for clustering with the integrated completed likelihood. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(7):719–725, 2000. doi: 10.1109/34.865189. 135

Christophe Biernacki, Gilles Celeux, and Gérard Govaert. Choosing starting values for the EM algorithm for getting the highest likelihood in multivariate Gaussian mixture models. *Computational Statistics & Data Analysis*, 41: 561–575, 01 2003. doi: 10.1016/S0167-9473(02)00163-9. 134, 154

Vincent Brault and Mahendra Mariadassou. Co-clustering through latent bloc model: A review. *Journal de la Société Française de Statistique*, 156(3): 120–139, 2015. URL http://journal-sfds.fr/article/view/474/448. 130

G. Celeux and J. Diebolt. The SEM algorithm: A probabilistic teacher algorithm derived from the EM algorithm for the mixture problem. *Computational Statistics Quarterly*, 2:73–82, 1985. 130

Marco Corneli, Charles Bouveyron, and Pierre Latouche. Co-clustering of ordinal data via latent continuous random variables and not missing at random entries. *Journal of Computational and Graphical Statistics*, 2020. doi: 10.1080/10618600.2020.1739533. URL https://hal.archives-ouvertes.fr/hal-01978174. 129

Jean-Jacques Daudin, Franck Picard, and Stephane Robin. A mixture model for random graphs. *Statistics and Computing*, 18(2):173–183, 2008. doi: 10.1007/s11222-007-9046-7. URL https://hal.archives-ouvertes.fr/hal-01197587. 135

Inderjit S. Dhillon. Co-clustering documents and words using bipartite spectral graph partitioning. In *Proceedings of the seventh ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 269–274, aug 2001. doi: 10.1145/502512.502550. 129

Inderjit S. Dhillon, Subramanyam Mallela, and Dharmendra S. Modha. Information-theoretic co-clustering. In *Proceedings of the Ninth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '03, pages 89—-98, New York, NY, USA, 2003. Association for Computing Machinery. ISBN 1581137370. doi: 10.1145/956750.956764. 129

Chris Ding, Tao Li, Wei Peng, and Haesun Park. Orthogonal nonnegative matrix t-factorizations for clustering. In *Proceedings of the Twelfth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, volume 2006, pages 126–135, 01 2006. doi: 10.1145/1150402.1150420. 129

Gabriel Frisch, Jean-Benoist Leger, and Yves Grandvalet. SparseBM: A Python Module for Handling Sparse Graphs with Block Models. working paper or preprint, February 2021. URL `https://hal.archives-ouvertes.fr/hal-03139586`. 129

Gérard Govaert and Mohamed Nadif. Block clustering with Bernoulli mixture models: Comparison of different approaches. *Computational Statistics & Data Analysis*, 52(6):3233–3245, February 2008. doi: 10.1016/j.csda.2007.09.007. 129, 154

F. Maxwell Harper and Joseph A. Konstan. The movielens datasets: History and context. *ACM Trans. Interact. Intell. Syst.*, 5(4), December 2015. ISSN 2160-6455. doi: 10.1145/2827872. URL `https://doi.org/10.1145/2827872`. 129

Matthew D. Hoffman, David M. Blei, Chong Wang, and John Paisley. Stochastic variational inference. *Journal of Machine Learning Research*, 14(4): 1303–1347, 2013. URL `http://jmlr.org/papers/v14/hoffman13a.html`. 150

Paul W. Holland, Kathryn Blackmond Laskey, and Samuel Leinhardt. Stochastic blockmodels: First steps. *Social Networks*, 5(2):109 – 137, 1983. 129

Lawrence Hubert and Phipps Arabie. Comparing partitions. *Journal of Classification*, 2(1):193–218, Dec 1985. ISSN 1432-1343. doi: 10.1007/BF01908075. 139

Tommi S. Jaakkola. Tutorial on variational approximation methods. In Manfred Opper and David Saad, editors, *Advanced Mean Field Methods: Theory and Practice*, pages 129–159. MIT Press, 2000. 131

Eric Jang, Shixiang Gu, and Ben Poole. Categorical reparameterization with gumbel-softmax. *arXiv preprint arXiv:1611.01144*, 2016. 153

Michael I. Jordan, Zoubin Ghahramani, Tommi S. Jaakkola, and Lawrence K. Saul. An introduction to variational methods for graphical models. *Machine Learning*, 37(2):183—-233, November 1999. ISSN 0885-6125. doi: 10.1023/A:1007665907178. URL https://doi.org/10.1023/A:1007665907178. 131

Christine Keribin, Vincent Brault, Gilles Celeux, and Gérard Govaert. Model selection for the binary latent block model. In *Proceedings of COMPSTAT*, 08 2012. 135

Christine Keribin, Vincent Brault, Gilles Celeux, and Gérard Govaert. Estimation and selection for the latent block model on categorical data. *Statistics and Computing*, 25(6):1201–1216, 2015. 130

Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014. 151

Diederik P Kingma and Max Welling. Auto-encoding variational bayes, 2014. 153

Yuval Kluger, Ronen Basri, Joseph Chang, and Mark Gerstein. Spectral biclustering of microarray data: Coclustering genes and conditions. *Genome Research*, 13:703–716, 05 2003. doi: 10.1101/gr.648603. 129

Lazhar Labiod and Mohamed Nadif. Co-clustering for binary and categorical data with maximum modularity. In *11th IEEE International Conference on Data Mining (ICDM)*, pages 1140–1145, 2011. doi: 10.1109/ICDM.2011.37. 129

Pierre Latouche, Etienne Birmelé, and Christophe Ambroise. Overlapping stochastic block models with application to the French political blogosphere. *The Annals of Applied Statistics*, 5(1):309–336, Mar 2011. ISSN 1932-6157. doi: 10.1214/10-aoas382. 129, 152

Jean-Benoist Leger. Blockmodels: A r-package for estimating in latent block model and stochastic block model, with various probability functions, with or without covariates, 2016. 129, 147, 152

J. MacQueen. Some methods for classification and analysis of multivariate observations. In *Proceedings of the Fifth Berkeley Symposium on Mathematical Statistics and Probability, Volume 1: Statistics*, pages 281–297, Berkeley, Calif., 1967. University of California Press. URL https://projecteuclid.org/euclid.bsmsp/1200512992. 135

Catherine Matias and Vincent Miele. Statistical clustering of temporal networks through a dynamic stochastic block model. *Journal of the Royal Statistical Society Series B*, 79(4):1119–1141, 2017. doi: 10.1111/rssb.12200. 138

Mohamed Nadif and Gérard Govaert. Latent block model for contingency table. *Communications in Statistics—Theory and Methods*, 39(3):416–425, 01 2010. doi: 10.1080/03610920903140197. 129

Giorgio Parisi. *Statistical Field Theory*. Frontiers in Physics. Addison-Wesley, 1988. URL https://cds.cern.ch/record/111935. 131

Valerie Robert, Yann Vasseur, and Vincent Brault. Comparing high-dimensional partitions with the co-clustering adjusted rand index. *Journal of Classification*, Nov 2020. ISSN 1432-1343. doi: 10.1007/s00357-020-09379-w. 139, 148, 155

Abraham Savitzky and Marcel JE Golay. Smoothing and differentiation of data by simplified least squares procedures. *Analytical chemistry*, 36(8): 1627–1639, 1964. 151

Margot Selosse, Julien Jacques, and Christophe Biernacki. Textual data summarization using the self-organized co-clustering model. *Pattern Recognition*, 103:107315, 2020. ISSN 0031-3203. doi: https://doi.org/10.1016/j.patcog.2020.107315. URL http://www.sciencedirect.com/science/article/pii/S0031320320301199. 129

Emilie Shireman, Douglas Steinley, and Michael Brusco. Examining the effect of initialization strategies on the performance of Gaussian mixture modeling. *Behavior Research Methods*, 49, 12 2015. doi: 10.3758/s13428-015-0697-6. 135, 156

Timothée Tabouy. *Impact of Sampling on Structure Inference in Networks : Application to Seed Exchange Networks and to Ecology*. PhD thesis, Université Paris-Saclay, September 2019. URL https://tel.archives-ouvertes.fr/tel-02414300. 136

Jason Wyse and Nial Friel. Block clustering with collapsed latent block models. *Statistics and Computing*, 22(2):415–428, 2012. 129

Chapter 5

Conclusion

In collaborative filtering, the reactions collected from people provide the *collaborative* basis for making predictions about other unseen items. Random graph clustering models are well suited to collaborative filtering, in that they search for users and items that share the same opinion patterns. The Latent block models (LBM) belong to these methods by clustering jointly the vertices of a bipartite graph. They offer an easy framework to target side issues in collaborative filtering such as handling informative missing data or fairness of the recommendations.

In collaborative filtering as in many other estimation problems, the absence of data conveys some information on the underlying phenomenon that should be exploited for its modeling. In Chapter 2, we had been interested in the clustering of the vertices of a bipartite graph whose edges are missing not at random. For this purpose, we introduced a co-clustering model that combines the Latent Block Model (LBM) with a MNAR missingness model that manages the censoring of data. The overall model accounts for an informative absence of data by retrieving groups of vertices based on the complete graph instead of considering only the partitioning of the observed network. Our missingness model preserves the symmetry of the co-clustering model by allowing each vertex to be characterized by a latent variable that defines the log-odd of the probability of its edges to be missing.

We proposed a variational fitting procedure using a Taylor series to obtain a tractable approximation of the lower bound of the observed log-likelihood. We provided a model selection criterion to select both the number of classes and the appropriate type of missingness. With experiments on synthetic datasets, we showed that ignoring an informative missingness can lead to catastrophic co-clustering estimates, supporting the value of using expressive missingness models on such type of data. Our model is appropriate for collaborative filtering purposes in which previous studies showed that the probability of observing a rating depends on the actual rating that would be given by the user.

With some calculation tricks (detailled in Chapter 4) and additional approximations used in the inference procedure, we were able to handle datasets classically used for investigating recommender systems. In collaborative filtering, the common implicit assumption of ratings Missing At Random (MAR) is most probably incorrect. To properly assess the performances of a collaborative

filtering model a test set should consist of ratings selected at random (MCAR or MAR) for each user. A Yahoo study claims to contain such type of data and also shows a clear dependence of rating frequency on the underlying preference level. However using this dataset for assessment is problematic as results from the existing literature tend to point out that it was probably not collected correctly.

Producing fair recommendations, another common issue in collaborative filtering is discussed in Chapter 3. Fairness refers to certain sensitive attributes shared by groups of people, such as gender, age, ethnicity, socio-economic group, etc. Fairness by recommendation independence requires the unconditional statistical independence between recommendations and these specified sensitive attributes. Based on this concept, we proposed a definition of fairness specific to recommender systems, requiring item rankings to be independent of the users' sensitive attribute.

Our model combines the Gaussian Latent Block Model with an ordinal regression model in which the sensitive attribute is adequately accounted for allowing the clustering of users to be unaffected by the effects of this attribute. Our theoretical guarantee ensures approximately fair recommendations provided that the clustering of users respects a practically verifiable property, namely approximate statistical parity. Through experiments on a dataset classically used for investigating recommender systems, we first showed that in situations where sensitive attributes can be collected, it is preferable to design algorithms that process sensitive attributes to remove their influence, rather than simply ignore them. Furthermore, the latent variables inferred by our model are also amenable to analyses that can help identify recommendation bias. We also showed that an equality of treatment does not necessarily imply an equality of impact. A model ensuring an equal performances can still provide stereotypical recommendations. Thus, fairness criteria based on equality of impact can be ethically disputable.

Not related to common issue in collaborative filtering, Chapter 4 focuses on optimization techniques for block models. First, we presented some calculation tricks that reduce the complexity of the variational inference with Bernoulli block models provided that the network is sparse. This algorithm implemented in a Python module called SparseBM, was assessed on synthetic datasets showing its relevance to analyze sparse and much larger matrices than the other baselines can handle. When applied to complex models (such as in Chapter 2), the reformulation trick we proposed to reduce the complexity of the variational inference often entangles the analytical criterion. To side-step this problem, we proposed to use a general algorithm for block models based on a stochastic optimization of the variational criterion. By using numerical estimates of some non-tractable expressions of the criterion, this algorithm gives the opportunity to process large and not necessarily sparse networks with elaborate block models. The models presented in Chapter 2 and 3 couldn't have been applied to standard collaborative filtering datasets without using

these optimization techniques.

During this thesis, we worked on the Latent Block Models for collaborative filtering. Many issues remain partially or even unresolved due to time constraints and would deserve further investigation. A extension of our LBM with MNAR data to non-binary data would be valuable to treat properly the ordinal ratings given by users. A Gaussian Latent Block Model with an ordinal probit regression model as used in Chapter 3 could be appropriate if the missing data model is modified accordingly. However, the lack of dataset with an appropriate MAR test set would still be problematic for assessing the estimated ratings or recommendations. It could be worthwhile to launch an initiative to build a new dataset with industrial having such type of data.

We believe that there is a link between fairness of recommendation and Missing Not At Random data. We think it is most likely that the observed ratings are somewhat influenced by some societal phenomena related to the protected attributes. One could easily imagine a stereotypical, simple-minded example where males (or females) are less prone to rate romance (horror) movies even if they like them because they would be afraid to experience feelings of shame, even if their ratings are anonymous. In this example, a social pressure would encourage or discourage people to rate a movie conditionally to their true opinion and to their protected attribute. Investigating unfairness in data collection seems to us very interesting but also very hard to achieve as it would require a dataset gathering random sample of ratings (MAR or MCAR) as well as the user's protected attributes.

List of Figures