



**SCUOLA DI DOTTORATO**  
**UNIVERSITÀ DEGLI STUDI DI MILANO-BICOCCA**

Department of  
**ECONOMICS, MANAGEMENT, AND STATISTICS**

---

Ph.D. program: **Economics and Statistics**  
Curriculum: **Statistics**

Cycle: **XXXIV°**

**DEVELOPMENTS IN DISCRETE LATENT VARIABLE MODELS:  
DEALING WITH LIKELIHOOD MULTIMODALITY  
AND CLUSTERING OF SIMPLE HYPERGRAPHS**

Surname: **BRUSA**

Name: **LUCA**

Registration number: **839359**

Supervisor: Prof. **FULVIA PENNONI**

Co-Supervisor: Prof. **FRANCESCO BARTOLUCCI**

Tutor: Prof. **GIORGIO VITTADINI**

**Academic Year: 2021-2022**



---

## Abstract

---

Latent variable models provide a fundamental statistical framework for the analysis of data in many different fields. These include, but are not limited to, education, psychology, sociology, economics, medicine, and engineering. The popularity of these models has significantly grown in the last decades due to the increasing availability of data and of computational resources. The thesis is organized as follows.

In Chapter 1 we provide basic background knowledge for the latent variable models based on a discrete distribution. In particular, we review in some details specific classes of the models at issue, namely latent class (LC), hidden Markov (HM) and stochastic block (SB) models. We recall the main underlying assumptions and analyze the standard maximum likelihood estimation of model parameters through the expectation-maximization (EM) algorithm.

In Chapter 2 we deal with the problem of multimodality of the log-likelihood function of discrete latent variable models, and the consequent possible convergence of the EM algorithm to a local maximum. We introduce a class of optimization methods, namely tempering or annealing, which employ a parameter known as temperature to re-scale the target function and monitor the prominence of all maxima. Exploiting the basic idea of these techniques, we propose a tempered EM algorithm to explore the parameter space adequately and increase the chance to reach the global maximum. We compare the proposal with the standard EM algorithm by an extensive Monte Carlo simulation study, evaluating both the ability to reach the global maximum of the log-likelihood function, and the computational time. We also employ the proposal on cross-sectional and longitudinal data referred to some application of interest, showing the main findings for LC and HM models.

In Chapter 3 we consider again the problem of convergence of the EM algorithm to a local maximum proposing a different approach. We explore the framework of evolutionary algorithms, a class of optimization methods working according to the biological principles of natural evolution. We discuss the mechanism of the main evolutionary operators and propose to apply it within the context of the EM algorithm for the estimation of the

---

parameters of discrete latent variable models. This approach allows us to explore more accurately the parameter space exploration and allows us to escape local maxima. The performance of the resulting algorithm is assessed relying on the same simulation scheme proposed in Chapter 2. This allows us to compare the two proposals, highlighting the benefits and drawbacks of both approaches.

In Chapter 4, in the context of SB model, we tackle the need to account for higher-order interactions, in order to include information deriving from groups of three or more subjects. We review the notion of hypergraphs and hyperedges, which extend the concept of graphs and edges respectively, and provide the most general mathematical formalization of high-order interactions. In particular we distinguish the notion of “simple” hypergraphs, where hyperedges are subsets of distinct nodes taking part into an interaction, from the notion of “multisets” hypergraphs, where repeated nodes are allowed in the same hyperedge; we illustrate how a proper choice has to rely on the specificity of each dataset. In this work, we focus on model-based clustering for simple hypergraphs, where literature is quite scarce, and computational challenges increase. We propose a general SB model for simple hypergraphs which allows us to capture the information of higher-order interactions. We perform maximum likelihood estimation of model parameters through a variational EM algorithm, and explore model selection using the Integrated Classification Likelihood criterion. The model is applied to both simulated and real data, and the performance of the proposal is assessed in terms of parameter estimation and ability to recover the clusters.

---

# Contents

---

|  |             |
|--|-------------|
| <b>Abstract</b>  | <b>iii</b>  |
| <b>Contents</b>  | <b>v</b>    |
| <b>List of Figures</b>   | <b>ix</b>   |
| <b>List of Tables</b>  | <b>xiii</b> |
| <b>1 Introduction</b>  | <b>1</b>    |
| 1.1 An overview of latent variable models . . . . .  | 1           |
| 1.1.1 General formulation of discrete latent variable models . . . . .                                     | 2           |
| 1.1.2 Maximum likelihood estimation . . . . .  | 3           |
| 1.1.3 Further estimation issues . . . . .  | 5           |
| 1.2 Relevant examples of discrete latent variable models . . . . .   | 5           |
| 1.2.1 Latent class model . . . . .   | 6           |
| 1.2.2 Hidden Markov model . . . . .  | 9           |
| 1.2.3 Stochastic block model . . . . .   | 16          |
| 1.3 Outline and main contributions . . . . .   | 19          |
| Bibliography . . . . .   | 21          |
| <b>2 Tempered Expectation-Maximization algorithm for the estimation of discrete latent variable models</b> | <b>25</b>   |
| 2.1 Introduction . . . . .   | 25          |
| 2.2 Convergence to local maxima . . . . .  | 27          |
| 2.2.1 Simulation study of multimodality for latent class model . . . . .                                   | 28          |
| 2.3 Annealing and tempering techniques . . . . .   | 37          |
| 2.3.1 The origin of simulated annealing . . . . .  | 37          |
| 2.3.2 The simulated tempering variant . . . . .  | 40          |

---

|          |  |           |
|----------|--|-----------|
| 2.3.3    | The parallel tempering version . . . . .   | 41        |
| 2.3.4    | Other features . . . . .   | 43        |
| 2.4      | Tempered Expectation-Maximization algorithm . . . . .  | 45        |
| 2.4.1    | Choice of the temperature parameter . . . . .  | 45        |
| 2.4.2    | Tuning of the tempering profiles . . . . .   | 46        |
| 2.4.3    | T-EM algorithm for the latent class model with categorical response variables . . . . .                                | 48        |
| 2.4.4    | T-EM algorithm for the hidden Markov model with categorical response variables . . . . .                               | 49        |
| 2.4.5    | T-EM algorithm for the hidden Markov model with continuous response variables . . . . .                                | 50        |
| 2.5      | Simulation study . . . . .   | 51        |
| 2.5.1    | Settings of the experimental scenarios . . . . .   | 51        |
| 2.5.2    | Simulation results . . . . .   | 52        |
| 2.5.3    | Results in terms of computational time . . . . .   | 57        |
| 2.5.4    | The role of the oscillating tempering profile . . . . .  | 58        |
| 2.5.5    | Analysis of the tempered Expectation-Maximization algorithm with fixed tempering profile . . . . .                     | 60        |
| 2.6      | Initialization of the tempered Expectation-Maximization algorithm . . . . .  | 61        |
| 2.7      | Applications . . . . .   | 65        |
| 2.7.1    | Evaluation of anxiety and depression . . . . .   | 65        |
| 2.7.2    | Discovering criminal trajectories . . . . .  | 67        |
| 2.7.3    | Analyzing countries development . . . . .  | 68        |
| 2.8      | Conclusions . . . . .  | 71        |
|          | Appendices . . . . .   | 73        |
| A        | Characteristics of the simulated scenarios . . . . .   | 73        |
| B        | Additional simulation results . . . . .  | 75        |
| C        | Simulation results with fixed tempering profiles . . . . .   | 77        |
| D        | Real data analysis in terms of computational time . . . . .  | 79        |
|          | Bibliography . . . . .   | 80        |
| <b>3</b> | <b>Evolutionary Expectation-Maximization algorithm for the estimation of discrete latent variable models</b> . . . . . | <b>85</b> |
| 3.1      | Introduction . . . . .   | 85        |
| 3.2      | Evolutionary algorithms . . . . .  | 86        |
| 3.2.1    | Design of evolutionary algorithms . . . . .  | 86        |

|                        |  |            |
|------------------------|--|------------|
| 3.2.2                  | Previous works . . . . .   | 89         |
| 3.3                    | Evolutionary Expectation-Maximization algorithm . . . . .                            | 89         |
| 3.3.1                  | Initialization and convergence criterion . . . . .                                   | 91         |
| 3.3.2                  | Model selection . . . . .  | 92         |
| 3.4                    | Simulation study . . . . .   | 93         |
| 3.5                    | Results with real-world data . . . . .   | 96         |
| 3.6                    | Conclusions . . . . .  | 99         |
| Appendices . . . . .   |  | 101        |
| A                      | Additional simulation results . . . . .  | 101        |
| B                      | Comparison with the oscillating T-EM algorithm . . . . .                             | 103        |
| C                      | Additional features about crossover and mutation . . . . .                           | 104        |
| Bibliography . . . . . |  | 106        |
| <b>4</b>               | <b>Model-based clustering in simple hypergraphs through a stochastic block-model</b> | <b>109</b> |
| 4.1                    | Introduction . . . . .   | 109        |
| 4.2                    | Interacting systems and hypergraphs . . . . .  | 111        |
| 4.2.1                  | Higher-order interactions representation: from graphs to hypergraphs                 | 111        |
| 4.2.2                  | The need for simple hypergraphs models . . . . .                                     | 112        |
| 4.2.3                  | Matrix representations of higher-order interactions . . . . .                        | 115        |
| 4.3                    | Preliminary works about hypergraph modeling . . . . .                                | 116        |
| 4.3.1                  | The bipartite graph representation and its limits . . . . .                          | 116        |
| 4.3.2                  | Hypergraphs modeling . . . . .   | 118        |
| 4.4                    | A stochastic block model for hypergraphs . . . . .                                   | 120        |
| 4.4.1                  | Model formulation . . . . .  | 120        |
| 4.4.2                  | Parameter identifiability . . . . .  | 123        |
| 4.4.3                  | Maximum likelihood estimation . . . . .  | 124        |
| 4.4.4                  | Model selection . . . . .  | 131        |
| 4.5                    | Simulation study . . . . .   | 131        |
| 4.5.1                  | Clustering performances . . . . .  | 132        |
| 4.5.2                  | Performance of model selection . . . . .   | 134        |
| 4.6                    | Analysis of a co-authorship dataset . . . . .  | 135        |
| 4.6.1                  | Dataset description . . . . .  | 135        |
| 4.6.2                  | Analysis with the <b>HyperSBM</b> package . . . . .                                  | 135        |
| 4.6.3                  | Comparison with Hypergraph Spectral Clustering . . . . .                             | 137        |
| 4.6.4                  | Comparison with a bipartite SB model . . . . .                                       | 138        |

|     |   |     |
|-----|---|-----|
| 4.7 | Conclusions . . . . .   | 139 |
|     | Appendices . . . . .  | 141 |
| A   | Proofs of theoretical results . . . . .                       | 141 |
| B   | Complete proof of the identifiability . . . . .               | 145 |
| C   | Artifacts induced by bipartite graph models . . . . .         | 154 |
| D   | Computational details . . . . .                               | 155 |
| E   | Spectral clustering . . . . .                                 | 157 |
| F   | More on the simulation study . . . . .                        | 158 |
| G   | The Hypergraph SB model is not a bipartite SB model . . . . . | 161 |
|     | Bibliography . . . . .  | 163 |



---

## List of Figures

---

|   |    |
|---|----|
| 1.1 Path diagram of the latent class model for multivariate data. . . . .   | 6  |
| 1.2 Path diagram of the basic hidden Markov model for multivariate data. . . .  | 10 |
| 1.3 Path diagram of the stochastic block model. . . . .   | 18 |
| 2.1 Graphical representation of the implemented Monte Carlo simulation study  | 27 |
| 2.2 Number of different local maxima obtained estimating the latent class model<br>with the Expectation-Maximization algorithm . . . . .  | 30 |
| 2.3 Percentage of global maximum obtained estimating the latent class model<br>with the Expectation-Maximization algorithm . . . . .  | 31 |
| 2.4 Configurations of parameters considered for both the simulation and the<br>estimation phase for the hidden Markov model with categorical response<br>variables . . . . .                                | 34 |
| 2.5 Number of different local maxima obtained estimating the hidden Markov<br>model with categorical response variables with the Expectation-Maximization<br>algorithm . . . . .                            | 35 |
| 2.6 Percentage of global maximum obtained estimating the hidden Markov model<br>with categorical response variables with the Expectation-Maximization algo-<br>rithm . . . . .                              | 36 |
| 2.7 Simulated annealing scheme . . . . .  | 40 |
| 2.8 Simulated tempering scheme . . . . .  | 42 |
| 2.9 Parallel tempering scheme . . . . .   | 43 |
| 2.10 Percentages of global maxima obtained using EM and M-T-EM algorithms<br>under simulated scenarios for the latent class and hidden Markov models<br>with correctly specified latent structure . . . . . | 53 |
| 2.11 Percentages of global maximum using EM and M-T-EM algorithms under<br>simulated scenarios for the latent class and hidden Markov models with mis-<br>specified latent structure . . . . .              | 54 |

|      |  |     |
|------|--|-----|
| 2.12 | Percentage of global maximum and mean distance from it using EM, M-T-EM, and O-T-EM algorithms on simulated data from an LC model correctly specified with 6 latent classes . . . . .                            | 59  |
| 2.13 | Maximized log-likelihood values for the anxiety and depression data using standard EM and O-T-EM algorithms for different numbers of the latent classes $k$ . . . . .  | 66  |
| 2.14 | Maximized log-likelihood values for the criminal data using standard EM and O-T-EM algorithms for different numbers of the latent states $k$ . . . . .   | 68  |
| 2.15 | Maximized log-likelihood values for the countries' economic conditions data using standard EM and O-T-EM algorithms with $k = 7$ latent states . . . . .   | 70  |
| 2.16 | Mean distance from the global maximum using EM and M-T-EM algorithms under simulated scenarios for the latent class and hidden Markov models with correctly specified latent structure . . . . .                 | 75  |
| 2.17 | Mean distance from the global maximum using EM and M-T-EM algorithms under simulated scenarios for the latent class and hidden Markov models with misspecified latent structure . . . . .                        | 76  |
| 3.1  | General scheme to describe the evolutionary algorithm . . . . .  | 87  |
| 3.2  | Percentages of global maxima obtained using EM, M-T-EM, and E-EM algorithms under simulated scenarios for the latent class and hidden Markov models with correctly specified latent structure . . . . .          | 94  |
| 3.3  | Percentages of global maxima obtained using EM, M-T-EM, and E-EM algorithms under simulated scenarios for the latent class and hidden Markov models with misspecified latent structure . . . . .                 | 95  |
| 3.4  | Maximized log-likelihood values for the anxiety and depression data using standard EM, O-T-EM, and E-EM algorithms . . . . .   | 97  |
| 3.5  | Maximized log-likelihood values for the countries' economic conditions data using standard EM, O-T-EM, and E-EM algorithms . . . . .   | 98  |
| 3.6  | Maximized log-likelihood values for the criminal data using standard EM, O-T-EM, and E-EM algorithms . . . . .   | 99  |
| 3.7  | Mean distance from the global maximum obtained using EM, M-T-EM, and E-EM algorithms under simulated scenarios for the latent class and hidden Markov models with correctly specified latent structure . . . . . | 101 |
| 3.8  | Mean distance from the global maximum using EM, M-T-EM, and E-EM algorithms under simulated scenarios for the latent class and hidden Markov models with misspecified latent structure . . . . .                 | 102 |

|     |   |     |
|-----|---|-----|
| 3.9 | Percentage of global maximum and mean distance from it using EM, M-T-EM, O-T-EM, and E-EM algorithms on simulated data from an latent class model correctly specified with 6 latent classes . . . . .   | 103 |
| 4.1 | Visualization of a set of higher-order interactions through a graph and a hypergraph . . . . .  | 112 |
| 4.2 | (a) A bipartite graph $\mathcal{G}$ ; (b) Projection of $\mathcal{G}$ into the multi-hypergraphs with self-loops space; (c) Projection of $\mathcal{G}$ on the simple hypergraphs subspace; (d) Embedding of the simple hypergraph in (c) in the bipartite graphs space . . | 117 |
| 4.3 | Mean Squared Error between true and estimated model parameters for different scenarios and number of nodes . . . . .  | 133 |
| 4.4 | Integrated Classification Likelihood index resulting from fitting the HSB model to the co-authorship dataset with number of latent groups ranging from 2 to 5 . . . . .   | 135 |



---

## List of Tables

---

|     |  |    |
|-----|--|----|
| 2.1 | Different values (with the corresponding frequency) obtained for the log-likelihood function on the basis of 100 repetitions of the Expectation-Maximization algorithm on the same sample, drawn from the latent class model according to scenario A and $n = 5,000$ . . . . .                         | 32 |
| 2.2 | Estimated parameters (weight of each latent class and conditional probabilities of each response variable given the latent class) referred to the results shown in Table 2.1 . . . . .   | 32 |
| 2.3 | Summary of all values defining the four simulated scenarios (first phase) for the hidden Markov model with categorical response variables . . . . .  | 33 |
| 2.4 | Different values (with the corresponding frequency) obtained for the log-likelihood function after 100 repetitions of the Expectation-Maximization algorithm on the same sample, drawn from the hidden Markov model with categorical response variables according to scenario C and time-heterogeneity | 37 |
| 2.5 | Estimated parameters (initial and transition probabilities) referred to the results shown in Table 2.4 . . . . .   | 38 |
| 2.6 | Number of samples in which the global maximum is reached with frequency $< 10\%$ , $> 50\%$ , or $> 95\%$ , using EM and M-T-EM algorithms under simulated scenarios for the latent class model . . . . .  | 55 |
| 2.7 | Number of samples in which the global maximum is reached with frequency $< 10\%$ , $> 50\%$ , or $> 95\%$ , using EM and M-T-EM algorithms under simulated scenarios for the hidden Markov model with categorical response variables .   | 56 |
| 2.8 | Number of samples in which the global maximum is reached with frequency $< 10\%$ , $> 50\%$ , or $> 95\%$ , using EM and M-T-EM algorithms under simulated scenarios for the hidden Markov model with continuous response variables .  | 56 |

|      |  |    |
|------|--|----|
| 2.9  | Mean square errors of the estimated model parameters with respect to the true model parameters, using EM and M-T-EM algorithms under simulated scenarios and estimating models with correct latent structure . . . . .                                   | 57 |
| 2.10 | Computational time in seconds of the EM and M-T-EM algorithms for each simulated scenario . . . . .  | 58 |
| 2.11 | Computational time in seconds of the EM, M-T-EM, and O-T-EM algorithms for the correctly specified latent class model with 6 latent classes . . . . .  | 60 |
| 2.12 | Influence of the $k$ -means (or $k$ -modes) initialization on the M-T-EM when the latent structure of the models is correctly specified . . . . .  | 63 |
| 2.13 | Influence of the $k$ -means (or $k$ -modes) initialization on the M-T-EM when the latent structure of the models is not correctly specified . . . . .  | 64 |
| 2.14 | Number of parameters, maximum log-likelihood and BIC index resulting from fitting a latent class model on the anxiety and depression data with EM and O-T-EM algorithms for different numbers of the latent classes $k$ . . . . .                        | 66 |
| 2.15 | Number of parameters, maximum log-likelihood and BIC index resulting from fitting a time heterogeneous hidden Markov model on the criminal data with EM and O-T-EM algorithms for different numbers of the latent states $k$ . . . . .                   | 67 |
| 2.16 | Number of parameters, maximum log-likelihood and BIC index resulting from fitting a time heterogeneous hidden Markov model on the countries' economic conditions data with EM and O-T-EM algorithms for different numbers of latent states $k$ . . . . . | 69 |
| 2.17 | Description of the simulated scenarios for the latent class model . . . . .  | 74 |
| 2.18 | Description of the simulated scenarios for the hidden Markov model with categorical response variables . . . . .   | 74 |
| 2.19 | Description of the simulated scenarios for the hidden Markov model with continuous response variables . . . . .  | 74 |
| 2.20 | Performance of the M-T-EM algorithm for latent class and hidden Markov models when the latent structure is correctly specified, using fixed configurations of tempering constants $\alpha$ and $\beta$ . . . . .   | 77 |
| 2.21 | Performance of the M-T-EM algorithm for latent class and hidden Markov models when the latent structure is not correctly specified, using fixed configurations of tempering constants $\alpha$ and $\beta$ . . . . .                                     | 78 |
| 2.22 | Computational times in seconds of the EM and O-T-EM algorithms for the anxiety and depression data . . . . .   | 79 |
| 2.23 | Computational times in seconds of the EM and O-T-EM algorithms for the criminal data . . . . .   | 79 |

|      |   |     |
|------|---|-----|
| 2.24 | Computational times in seconds of the EM and O-T-EM algorithms for the countries' economic conditions data . . . . .  | 79  |
| 3.1  | Number of samples in which the global maximum is reached with frequency < 5% or > 95%, using EM, M-T-EM, and E-EM algorithms under simulated scenarios for the latent class model . . . . .   | 96  |
| 3.2  | Number of samples in which the global maximum is reached with frequency < 5% or > 95%, using EM, M-T-EM, and E-EM algorithms under simulated scenarios for the hidden Markov model with categorical response variables . . . . .                  | 96  |
| 3.3  | Number of samples in which the global maximum is reached with frequency < 5% or > 95%, using EM, M-T-EM, and E-EM algorithms under simulated scenarios for the hidden Markov model with continuous response variables . . . . .                   | 97  |
| 4.1  | Summary of the parameters of the hypergraph stochastic block model . . . . .  | 122 |
| 4.2  | Number of parameters of the full hypergraph stochastic block model for given values of $Q$ (number of latent groups) and $M$ (largest hyperedge size) . . . . .   | 122 |
| 4.3  | Adjusted Rand Index for different scenarios and number of nodes . . . . .   | 132 |
| 4.4  | Results of model selection (performed by means of the ICL criterion): frequency of the selected number of groups and average Adjusted Rand Index of the classification obtained with $Q = 3$ depending on the selected number of groups . . . . . | 134 |
| 4.5  | Distribution of the number of distinct co-authors per author . . . . .  | 136 |
| 4.6  | Degree distribution of authors in the bipartite graph . . . . .   | 136 |
| 4.7  | Memory size of the matrix containing the products $\tau_{i_1 q_1} \cdots \tau_{i_m q_m}$ for given values of $n$ (number of nodes), $Q$ (number of latent groups) and $m$ (hyperedge size) . . . . .  | 156 |
| 4.8  | Performance of the proposed Variational Expectation-Maximization algorithm under different initialization strategies: spectral clustering, "soft" spectral clustering and graph-component absolute spectral clustering . . . . .                  | 158 |
| 4.9  | Adjusted Rand Index for different scenarios and number of nodes with respect to the soft spectral clustering initialization. Each value is obtained as the average over 10 simulated datasets . . . . .   | 159 |
| 4.10 | Computational time in seconds of the VEM algorithms for each setting, computed as the mean over 10 simulated samples . . . . .  | 160 |





# Introduction

---

## 1.1 An overview of latent variable models

Research workers in various disciplines often face the need to capture complex or conceptual properties of a system that are difficult to quantify or measure directly. Essentially, these situations may either refer to true quantities affected by measurement errors, or represent hypothetical constructs for which there exists no operational method for direct measurement. This behavior is especially true in the framework of social and behavioral sciences, but applications from other disciplines are becoming more and more common as the availability and complexity of data increases. These fields include, among others, medicine, economics, marketing, engineering, geography, and biology.

*Latent variables* provide a suitable probabilistic way to represent the above concepts: a latent variable is a random variable whose realizations are hidden from us, in contrast to manifest variables, where the realizations are observed ([von Eye and Clogg, 1994](#); [Borsboom et al., 2003](#)). Although latent variables are not directly observable themselves, certain of their effects on the manifest variables are observable, and hence subject to study. To this aim a broad variety of statistical models, formulated on the basis of latent variables, has been developed. Due to the wide range of applications, different definitions for such models are formulated according to specific literatures. Following [Bartolucci et al. \(2013\)](#), we define

a *latent variable model* as a statistical model which relies on specific assumptions on the conditional distribution of the response variables given the latent variables. See [Everitt \(1984\)](#), [Skrondal and Rabe-Hesketh \(2004\)](#) and [Bartholomew et al. \(2011\)](#) for a complete review of latent variable models. In general, a possible classification of the models at issue distinguishes between discrete and continuous latent variables. In the first case, a Gaussian distribution is typically assumed, thus resulting in a fully parametric model. In the second case, the model may be seen as semi-parametric, because no parametric assumptions are formulated on the distribution of the latent variables; in this way, the model is more flexible compared to other proposals. In the following, and throughout this whole work, the focus will be on discrete latent variable (DLV) models ([Bartolucci et al., 2022](#)). The following notation is borrowed from [Bartolucci et al. \(2013\)](#) and [Bartolucci et al. \(2022\)](#).

### 1.1.1 General formulation of discrete latent variable models

With reference to a single random unit drawn from the population of interest, let  $\mathbf{Y} = (Y_1, \dots, Y_r)$  denote the set of response variables and let  $\mathbf{U} = (U_1, \dots, U_l)$  denote the set of latent variables, having a discrete distribution. In the following we will assume that each latent variable has the same number  $k$  of latent components.

Given the structure of a latent variable model, it is generally possible to distinguish between two components:

- the measurements model, which describes the conditional distribution of the response variables given the latent variables:  $p(\mathbf{Y} = \mathbf{y} | \mathbf{U} = \mathbf{u})$ . Note that, under the assumption of local independence, we have  $p(\mathbf{Y} = \mathbf{y} | \mathbf{U} = \mathbf{u}) = \prod_{j=1}^r p(Y_j = y_j | \mathbf{U} = \mathbf{u})$ ;
- the latent model, which describes the distribution of the latent variables:  $p(\mathbf{U} = \mathbf{u})$ .

These two distributions depend on specific parameters according to the particular model, providing the basis for its formulation: by jointly considering them, we can obtain the manifest distribution, that is, the marginal distribution of the response variables, once the latent variables have been integrated out. This function assumes the following general expression

$$p(\mathbf{Y} = \mathbf{y}) = \sum_{\mathbf{u}} p(\mathbf{Y} = \mathbf{y} | \mathbf{U} = \mathbf{u}) p(\mathbf{U} = \mathbf{u}), \quad (1.1)$$

where the sum  $\sum_{\mathbf{u}}$  is over all possible configurations of  $\mathbf{U}$ . In the special case of a single latent variable ( $l = 1$ ), this summation corresponds to  $\sum_{u=1}^k$ . Finally, the posterior conditional distribution of the latent variables  $\mathbf{U}$  given the response variables  $\mathbf{Y}$  is obtained as

follows according to the Bayes' law:

$$p(\mathbf{U} = \mathbf{u} | \mathbf{Y} = \mathbf{y}) = \frac{p(\mathbf{Y} = \mathbf{y} | \mathbf{U} = \mathbf{u})p(\mathbf{U} = \mathbf{u})}{p(\mathbf{Y} = \mathbf{y})}.$$

These probabilities play a fundamental role both to estimate the model parameters and, once the model has been estimated, to assign each subject to a certain latent variable configuration.

It is worth noting that the above formulation of DLV models may be easily generalized by considering the inclusion of individual covariates either in the measurement model or in the latent model. These two extensions have very different interpretations: in the first case, covariates affect the response variables and the main interest is in explaining, through the latent variables, the heterogeneity between subjects that the observed covariates are not able to account for. In the second case, instead, covariates are assumed to affect the latent variables and the focus is on modeling the resulting effects.

### 1.1.2 Maximum likelihood estimation

As with any statistical model, estimation of DLV model parameters may be based on either a classical or a Bayesian approach. In this Section, as well as throughout the rest of the manuscript, we will focus on the first approach, which is typically carried out through the Expectation-Maximization (EM) algorithm (Baum et al., 1970; Dempster et al., 1977; McLachlan and Krishnan, 2008). What we obtain from the EM algorithm is the maximum likelihood estimate of the model parameters on the basis of a sample of  $n$  independent units.

To this aim, let  $\mathbf{y}_i$  denote the observed data for subject  $i$  and let  $\boldsymbol{\theta}$  denote the vector of all model parameters. Then the (incomplete data) log-likelihood function is equal to

$$\ell(\boldsymbol{\theta}) = \sum_{i=1}^n \log p(\mathbf{Y}_i = \mathbf{y}_i) = \sum_{\mathbf{y}} n_{\mathbf{y}} \log p(\mathbf{Y} = \mathbf{y}),$$

where the summation in the second expression is restricted to all the possible distinct configurations observed at least once,  $n_{\mathbf{y}} = \sum_{i=1}^n I(\mathbf{y}_i = \mathbf{y})$  is the frequency of configuration  $\mathbf{y}$ , and  $I(\cdot)$  denotes the indicator function. The latter formulation is generally more efficient from the computational point of view.

Maximization of  $\ell(\boldsymbol{\theta})$  by means of the EM algorithm requires the so-called *complete data*, corresponding to the pairs  $(\mathbf{y}_i, \mathbf{u}_i)$ ,  $i = 1, \dots, n$ , where  $\mathbf{u}_i$  is the latent configuration

for subject  $i$ . The complete data log-likelihood has then the following expression

$$\begin{aligned}\ell^*(\boldsymbol{\theta}) &= \sum_{i=1}^n \log p(\mathbf{Y}_i = \mathbf{y}_i, \mathbf{U}_i = \mathbf{u}_i) \\ &= \sum_{i=1}^n \log p(\mathbf{Y}_i = \mathbf{y}_i | \mathbf{U}_i = \mathbf{u}_i) + \sum_{i=1}^n \log p(\mathbf{U}_i = \mathbf{u}_i).\end{aligned}\tag{1.2}$$

Here the vectors  $\mathbf{u}_i$  are not directly observable. Hence, starting from an initial guess of the model parameters, the EM algorithm alternates the following two steps until a suitable convergence criterion is satisfied ( $h$  denotes the algorithm iteration number):

- **E-step:** compute the conditional expected value of  $\ell^*(\boldsymbol{\theta})$  given the observed data and the value of the parameters at the previous step:

$$\mathcal{Q}(\boldsymbol{\theta}; \boldsymbol{\theta}^{(h-1)}) = \mathbb{E}_{\boldsymbol{\theta}^{(h-1)}}[\ell^*(\boldsymbol{\theta}) | \mathbf{y}];$$

- **M-step:** maximize the expected value  $\mathcal{Q}(\boldsymbol{\theta}; \boldsymbol{\theta}^{(h-1)})$  and so update the model parameters:

$$\boldsymbol{\theta}^{(h)} = \arg \max_{\boldsymbol{\theta}} \mathcal{Q}(\boldsymbol{\theta}; \boldsymbol{\theta}^{(h-1)}).$$

The use of the EM algorithm ensure many advantages with respect to other maximization algorithms: among others, we mention the simplicity of implementation, the stable convergence to a local maximum of the log-likelihood function and the availability in many software packages. On the other hand, a well-known drawback of this approach is related to the multimodality of the log-likelihood function, which is especially observed when the model is based on discrete latent variables and has a certain degree of complexity. Consequently, the EM algorithm can converge to a local maximum, not corresponding to the global one.

Another potentially demanding aspect of the EM algorithm is related to the computation of the posterior conditional distribution  $p(\mathbf{U} = \mathbf{u} | \mathbf{Y} = \mathbf{y})$ , which could be infeasible for complex models. This problem could be sometimes addressed by implementing a backward-forward recursion (Baum et al., 1970; Welch, 2003). An alternative solution is to approximate the maximum likelihood estimates through a variational inference approach. In this case, instead of maximizing the intractable log-likelihood function, we optimize the lower bound that “best” (in terms of Kullback-Leibler divergence) approximates the log-likelihood function itself.

### 1.1.3 Further estimation issues

In this Section we briefly mention two further aspects related to the estimation of latent variable models. The first concerns the prediction of the latent variables, usually performed with the so-called maximum-a-posteriori rule (Goodman, 1974). Following this rule, a given unit of the sample is assigned to the latent support point with the highest estimated posterior probability. For simplest models, this corresponds to selecting a specific latent component. More complex models require instead the more challenging task to select the sequence of latent components for a certain sample unit (*e.g.*, the latent component for each time occasion for models on longitudinal data).

Finally, a relevant point in applying latent variable models is the choice of the number of latent components. Despite the existence of a specific criterion based on the likelihood ratio test, information criteria are commonly used in most applications. These criteria, in the case of maximum likelihood estimation (MLE), aim at penalizing the (incomplete) log-likelihood function in order to reach the best compromise between goodness-of-fit and model complexity. Among the others, we mention the Akaike information criterion (Akaike, 1973) and the Bayesian information criterion (Schwarz, 1978). The corresponding indices assume, respectively, the following general expressions:

$$\begin{aligned} \text{AIC} &= -2\hat{\ell} + 2\#par, \\ \text{BIC} &= -2\hat{\ell} + \log(n)\#par, \end{aligned}$$

where  $\hat{\ell}$  denotes the maximum value of the log-likelihood function and  $\#par$  the number of free parameters. Both criteria suggest selecting the number of components of the model corresponding to the smallest value of AIC or BIC. An alternative class of model selection criteria is widely used when the log-likelihood function may not be easily computed. Among them, the integrated classification likelihood (ICL) criterion (Biernacki et al., 2000) is particularly of interest, as it is based on a penalized form of the complete, rather than the incomplete, log-likelihood function. Many other model selection criteria have been proposed in the literature (Bacci et al., 2014; Bouveyron et al., 2019).

## 1.2 Relevant examples of discrete latent variable models

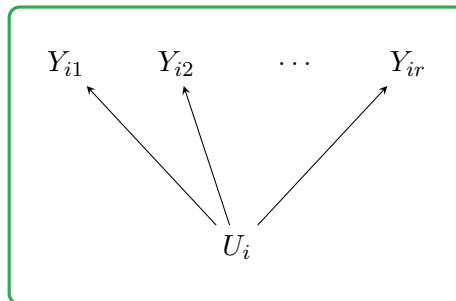
In this Section we introduce specific classes of DLV models: latent class (LC), hidden Markov (HM), and stochastic block (SB) models. Mainly borrowing from Bartolucci et al. (2013) and Bartolucci et al. (2022), we summarize model notation and formulation, and

describe the implementation of the standard MLE of the model parameters; see also [Bartolucci et al. \(2014\)](#), [Pandolfi et al. \(2021\)](#), and [Daudin et al. \(2008\)](#).

### 1.2.1 Latent class model

The LC model ([Lazarsfeld and Henry, 1968](#); [Goodman, 1974](#); [Lindsay et al., 1991](#)) is one of the most well-known latent variable models. It is typically applied for clustering a sample of subjects on the basis of a series of response variables, relying on the main assumption that the sample is drawn from a population made by different subpopulations or clusters, which are not directly observable. All units in the same cluster are assumed to have the same distribution of the response variables that, in general, is different from the one corresponding to other clusters.

**Model formulation** Considering cross-sectional data and for a sample of  $n$  independent individuals, let  $\mathbf{Y}_i = (Y_{i1}, \dots, Y_{ir})'$ ,  $i = 1, \dots, n$ , denote the vector of  $r$  response variables; we assume that each variable  $Y_{ij}$  is categorical with the same number  $c$  of categories, labeled from 0 to  $c-1$ . Note that the formulation of the model may be easily adapted to the case of response variables having a different number of categories; the case of continuous response variables instead is usually referred to as finite mixture model ([Titterington et al., 1985](#); [Lindsay, 1995](#); [McLachlan and Peel, 2000](#)). The LC model relies on individual-specific latent variables  $U_i$ ,  $i = 1, \dots, n$ , with  $k$  support points that identify the latent classes in the population, labeled from 1 to  $k$ . According to the assumption of local independence, the response variables are conditionally independent given the latent variable; the resulting path diagram is represented in Figure 1.1.



*Figure 1.1: Path diagram of the latent class model for multivariate data.*

The model parameters are the weights of each latent class, denoted by

$$\pi_u = p(U_i = u), \quad u = 1, \dots, k,$$

and the conditional probabilities of each response variable given the latent variable, denoted by

$$\phi_{jy|u} = p(Y_{ij} = y|U_i = u), \quad y = 0, \dots, c-1, \quad j = 1, \dots, r, \quad u = 1, \dots, k.$$

The probabilities  $\pi_u$  and  $\phi_{jy|u}$  must, obviously, be non-negative and satisfy the constraints  $\sum_{u=1}^k \pi_u = 1$  and  $\sum_{y=0}^{c-1} \phi_{jy|u} = 1$ ,  $j = 1, \dots, r$ ,  $u = 1, \dots, k$ . The number of free parameters of the model is then

$$\#par = \underbrace{k-1}_{\pi_u} + \underbrace{kr(c-1)}_{\phi_{jy|u}}.$$

We let  $\boldsymbol{\theta} = (\pi_u, \phi_{jy|u})_{u,y,j}$  denote the model parameter vector.

We can now make explicit the dependence of Equation (1.1) on the specific LC model parameters; the resulting manifest distribution for any individual  $i$  may be expressed as

$$p(\mathbf{Y}_i = \mathbf{y}) = \sum_{u=1}^k p(U_i = u)p(\mathbf{Y}_i = \mathbf{y}|U_i = u) = \sum_{u=1}^k \pi_u \prod_{j=1}^r \phi_{jy|u},$$

where  $p(\mathbf{Y}_i = \mathbf{y}|U_i = u) = \prod_{j=1}^r \phi_{jy|u}$  is the conditional probability of  $\mathbf{Y}_i$  given  $U_i = u$  (measurement model), and  $\mathbf{y} = (y_1, \dots, y_r)'$  denote a realization of  $\mathbf{Y}_i$ .

**Maximum likelihood estimation** As generically introduced in Section 1.1.2, in order to estimate the model parameters  $\boldsymbol{\theta}$ , we rely on the EM algorithm on the basis of the complete data log-likelihood. Accounting for the specific parameters of LC model, this function may be expressed as

$$\begin{aligned} \ell^*(\boldsymbol{\theta}) &= \sum_{i=1}^n \log p(\mathbf{Y}_i = \mathbf{y}_i|U_i = u_i) + \sum_{i=1}^n \log p(U_i = u_i) \\ &= \sum_{j=1}^r \sum_{u=1}^k \sum_{y=0}^{c-1} a_{juy} \log \phi_{jy|u} + \sum_{u=1}^k b_u \log \pi_u, \end{aligned} \quad (1.3)$$

where:

- $a_{juy} = \sum_{i=1}^n I(u_i = u, y_{ij} = y)$  is the frequency of subjects that are in latent class  $u$  and responded by  $y$  at the  $j$ -th response variable;
- $b_u = \sum_{i=1}^n I(u_i = u)$  is the number of sample units in latent class  $u$ .

In this setting, both expectation and maximization steps are simple to implement; in the following, referring to Section 1.1.2 for the general structure of the algorithm, we provide the explicit expressions for both steps:

- **E-step:** given the observed data and the value of the parameters at the previous step, compute the conditional expected values of each unknown frequency  $a_{juy}$  and  $b_u$  in (1.3). Computation of these expected values is based on the posterior probability that a subject with observed response configuration  $\mathbf{y}$  belongs to latent class  $u$ :

$$q(u|\mathbf{y}) = p(U_i = u | \mathbf{Y}_i = \mathbf{y}) = \frac{\pi_u \prod_{j=1}^r \phi_{jy|u}}{p(\mathbf{Y}_i = \mathbf{y})}, \quad u = 1, \dots, k. \quad (1.4)$$

The following closed-form expressions are then used:

$$\begin{aligned} \hat{b}_u &= \mathbb{E}[b_u] = \sum_{i=1}^n q(u|\mathbf{y}_i) = \sum_{\mathbf{y}} n_{\mathbf{y}} \cdot q(u|\mathbf{y}), \\ \hat{a}_{juy} &= \mathbb{E}[a_{juy}] = \sum_{i=1}^n I(y_{ij} = y) \cdot q(u|\mathbf{y}_i) = \sum_{\mathbf{y}} n_{\mathbf{y}} \cdot I(y_j = y) \cdot q(u|\mathbf{y}). \end{aligned}$$

In particular,  $\hat{b}_u$  must be computed for  $u = 1, \dots, k$ , whereas  $\hat{a}_{juy}$  must be computed for  $j = 1, \dots, r$ ,  $u = 1, \dots, k$ , and  $y = 0, \dots, c - 1$ .

- **M-step:** maximize the expected value of  $\ell^*(\boldsymbol{\theta})$  obtained by replacing each variable  $a_{juy}$  and  $b_u$  with  $\hat{a}_{juy}$  and  $\hat{b}_u$  respectively. As reported in [Bartolucci et al. \(2013\)](#), the following explicit solutions are available:

- *class weights:*

$$\pi_u = \frac{\hat{b}_u}{n},$$

to be computed for  $u = 1, \dots, k$ ;

- *conditional response probabilities:*

$$\phi_{jy|u} = \frac{\hat{a}_{juy}}{\hat{b}_u},$$

to be computed for  $y = 0, \dots, c - 1$ ,  $j = 1, \dots, r$ , and  $u = 1, \dots, k$ .



### 1.2.2 Hidden Markov model

The HM model (Wiggins, 1973; Bartolucci et al., 2013) represents a generalization of the LC model to the case of longitudinal data. It allows a dynamic clustering where each unit may move between clusters across time. Although our focus is on longitudinal data, HM models may be easily adapted to the analysis of time-series (Zucchini et al., 2016; MacDonald and Zucchini, 2016).

**Model formulation** For a sample of  $n$  individuals and for  $T$  time occasions, let  $\mathbf{Y}_i^{(t)} = (Y_{i1}^{(t)}, \dots, Y_{ir}^{(t)})'$  denote the occasion-specific response variables,  $i = 1, \dots, n$  and  $t = 1, \dots, T$ , and let  $\mathbf{Y}_i = (\mathbf{Y}_i^{(1)}, \dots, \mathbf{Y}_i^{(T)})$  denote the vector of response variables,  $i = 1, \dots, n$ . In order to model these responses, let us consider a sequence of latent variables  $U_i^{(1)}, \dots, U_i^{(T)}$ , collected in the vector  $\mathbf{U}_i$  and having a discrete distribution with  $k$  states. The latent model parameters are the initial probabilities, denoted by

$$\pi_u = p(U_i^{(1)} = u), \quad u = 1, \dots, k,$$

and the transition probabilities, denoted by

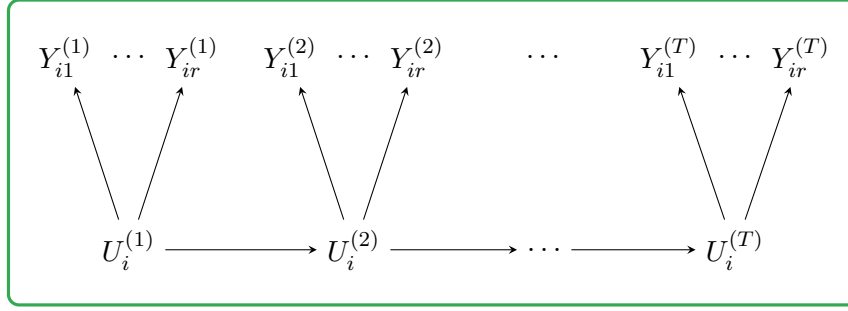
$$\pi_{u|\bar{u}}^{(t)} = p(U_i^{(t)} = u | U_i^{(t-1)} = \bar{u}), \quad t = 2, \dots, T, \quad \bar{u}, u = 1, \dots, k.$$

Note that it is possible to include a constraint corresponding to the hypothesis that the latent process is time homogeneous, such that the transition probabilities do not depend on time occasion  $t$ :  $\pi_{u|\bar{u}}^{(t)} = \pi_{u|\bar{u}}$ ,  $t = 2, \dots, T$ .

The HM model in its basic formulation (Bartolucci et al., 2013) relies on the following three main assumptions, which can be suitable relaxed:

1.  $\mathbf{Y}_i^{(1)}, \dots, \mathbf{Y}_i^{(T)}$  are conditionally independent given  $\mathbf{U}_i$ ;
2.  $Y_{i1}^{(t)}, \dots, Y_{ir}^{(t)}$  are conditionally independent given  $U_i^{(t)}$ , for  $t = 1, \dots, T$ ;
3.  $\mathbf{U}_i$  follows a first-order Markov chain with state space  $\{1, \dots, k\}$ , where  $k$  is the number of latent states.

The first two conditions represent a form of local independence; in other words, each occasion-specific response variable  $Y_{ij}^{(t)}$  is independent of  $Y_{ij}^{(1)}, \dots, Y_{ij}^{(t-1)}, Y_{ij}^{(t+1)}, \dots, Y_{ij}^{(T)}$  and of  $Y_h^{(t)}$ , for all  $h = 1, \dots, r$ ,  $h \neq j$ , given  $U_i^{(t)}$ . The resulting model is represented by the path diagram shown in Figure 1.2.



**Figure 1.2:** Path diagram of the basic hidden Markov model for multivariate data.

The proposed framework for HM models is sufficiently flexible to deal with responses of different nature; in the following we will distinguish between categorical and continuous response variables.

### 1.2.2.1 Hidden Markov model with categorical response variables

Let  $Y_{ij}^{(t)}$ ,  $j = 1, \dots, r$ ,  $t = 1, \dots, T$ , denote the categorical response variable with  $c$  categories for subject  $i$ . The corresponding conditional response probabilities are denoted by

$$\begin{aligned} \phi_{jy|u} &= p(Y_{ij}^{(t)} = y | U_i^{(t)} = u), & t = 1, \dots, T, \\ j &= 1, \dots, r, \quad y = 0, \dots, c-1, \quad u = 1, \dots, k, \end{aligned}$$

and may be collected into the vector

$$\phi_{\mathbf{y}|u} = \prod_{j=1}^r \phi_{jy|u} = p(Y_{i1}^{(t)} = y_1, \dots, Y_{ir}^{(t)} = y_r | U_i^{(t)} = u) = p(\mathbf{Y}_i^{(t)} = \mathbf{y} | U_i^{(t)} = u),$$

with  $\mathbf{y} = (y_1, \dots, y_r)'$ .

Accounting for the usual constraint on the above probabilities, the number of free parameters is

$$\#\text{par} = \underbrace{k-1}_{\pi_u} + \underbrace{(T-1)k(k-1)}_{\pi_{u|\bar{u}}^{(t)}} + \underbrace{kr(c-1)}_{\phi_{jy|u}}$$

if the model is time-heterogeneous, and reduces to

$$\#\text{par} = \underbrace{k-1}_{\pi_u} + \underbrace{k(k-1)}_{\pi_{u|\bar{u}}} + \underbrace{kr(c-1)}_{\phi_{jy|u}}$$

for a time-homogeneous model. Let us denote by  $\boldsymbol{\theta} = (\pi_{\mathbf{u}}, \pi_{\mathbf{u}|\bar{\mathbf{u}}}, \phi_{j\mathbf{y}|\mathbf{u}})_{\mathbf{u}, \bar{\mathbf{u}}, t, j, \mathbf{y}}$  the parameter vector.

The probability mass function of the distribution of  $\mathbf{U}_i$  may be expressed as

$$p(\mathbf{U}_i = \mathbf{u}) = \pi_{\mathbf{u}^{(1)}} \prod_{t=2}^T \pi_{\mathbf{u}^{(t)}|\mathbf{u}^{(t-1)}},$$

where  $\mathbf{u} = (u^{(1)}, \dots, u^{(T)})'$  denotes a realization of  $\mathbf{U}_i$ . In addition, the conditional distribution of  $\mathbf{Y}_i$  given  $\mathbf{U}_i = \mathbf{u}$  is

$$p(\mathbf{Y}_i = \mathbf{y}|\mathbf{U}_i = \mathbf{u}) = \prod_{t=1}^T \phi_{\mathbf{y}^{(t)}|\mathbf{u}^{(t)}} = \prod_{t=1}^T \prod_{j=1}^r \phi_{j\mathbf{y}^{(t)}|\mathbf{u}^{(t)}},$$

where  $\mathbf{y} = (\mathbf{y}^{(1)}, \dots, \mathbf{y}^{(T)})'$  denotes a realization of  $\mathbf{Y}_i$  and therefore, the resulting manifest distribution of  $\mathbf{Y}_i$  is

$$p(\mathbf{Y}_i = \mathbf{y}) = \sum_{\mathbf{u}} \pi_{\mathbf{u}^{(1)}} \pi_{\mathbf{u}^{(2)}|\mathbf{u}^{(1)}} \cdots \pi_{\mathbf{u}^{(T)}|\mathbf{u}^{(T-1)}} \phi_{\mathbf{y}^{(1)}|\mathbf{u}^{(1)}} \phi_{\mathbf{y}^{(2)}|\mathbf{u}^{(2)}} \cdots \phi_{\mathbf{y}^{(T)}|\mathbf{u}^{(T)}}.$$

It is important to note that computing the manifest distribution  $p(\mathbf{Y}_i = \mathbf{y})$  involves a summation over all possible  $k^T$  configurations of vector  $\mathbf{u}$ , which would require a considerable computational effort. In practice, this distribution is efficiently computed with a forward recursion (Baum et al., 1970; Welch, 2003). This recursion is efficiently implemented by using the matrix notation (Bartolucci, 2006; Bartolucci et al., 2007).

**Maximum likelihood estimation** The model parameters are estimated by means of the EM algorithm, on the basis of the complete data log-likelihood. In this case, the complete data are represented by the pairs  $(\mathbf{u}_i, \mathbf{y}_i)$ ,  $i = 1, \dots, n$ , where, with reference to subject  $i$ ,  $\mathbf{y}_i$  and  $\mathbf{u}_i$  denote the realization of  $\mathbf{Y}_i$  and  $\mathbf{U}_i$ , respectively. The corresponding complete data log-likelihood function may be written as follows:

$$\begin{aligned}
 \ell^*(\boldsymbol{\theta}) &= \sum_{i=1}^n \log p(\mathbf{Y}_i = \mathbf{y}_i | \mathbf{U}_i = \mathbf{u}_i) + \sum_{i=1}^n \log p(\mathbf{U}_i = \mathbf{u}_i) \tag{1.5} \\
 &= \sum_{i=1}^n \log \left( \prod_{t=1}^T \prod_{j=1}^r \phi_{jy^{(t)}|u^{(t)}} \right) + \sum_{i=1}^n \log \left( \pi_{u^{(1)}} \prod_{t=2}^T \pi_{u^{(t)}|u^{(t-1)}} \right) \\
 &= \sum_{i=1}^n \sum_{t=1}^T \sum_{j=1}^r \log \phi_{jy^{(t)}|u^{(t)}} + \sum_{i=1}^n \log \pi_{u^{(1)}} + \sum_{i=1}^n \sum_{t=2}^T \log \pi_{u^{(t)}|u^{(t-1)}} \\
 &= \sum_{t=1}^T \sum_{j=1}^r \sum_{u=1}^k \sum_{y=0}^{c-1} a_{juy}^{(t)} \log \phi_{jy|u} + \sum_{u=1}^k b_u^{(1)} \log \pi_u + \sum_{t=2}^T \sum_{\bar{u}=1}^k \sum_{u=1}^k b_{\bar{u}u}^{(t)} \log \pi_{u|\bar{u}},
 \end{aligned}$$

where:

- $a_{juy}^{(t)} = \sum_{i=1}^n I(u_i^{(t)} = u, y_{ij}^{(t)} = y)$  is the number of subjects that, at time occasion  $t$ , are in latent state  $u$  and have outcome  $y$  for the  $j$ -th response variable;
- $b_u^{(t)} = \sum_{i=1}^n I(u_i^{(t)} = u)$  is the number of subjects in latent state  $u$  at time occasion  $t$ ;
- $b_{\bar{u}u}^{(t)} = \sum_{i=1}^n I(u_i^{(t-1)} = \bar{u} | u_i^{(t)} = u)$  is the number of subjects that move from latent state  $\bar{u}$  to latent state  $u$  at time occasion  $t$ .

Since the frequencies  $a_{juy}^{(t)}$ ,  $b_u^{(t)}$ , and  $b_{\bar{u}u}^{(t)}$  are obviously unknown, the EM algorithm alternates the following two steps:

- **E-step:** given the observed data and the parameters value at the previous step, compute the expected value of every frequency  $a_{juy}^{(t)}$ ,  $b_u^{(t)}$ , and  $b_{\bar{u}u}^{(t)}$  in (1.5), so as to obtain the expected value of  $\ell^*(\boldsymbol{\theta})$ . This computation is based on the conditional posterior probabilities:

$$q^{(t)}(u|\mathbf{y}) = p(U_i^{(t)} = u | \mathbf{Y}_i = \mathbf{y}),$$

and

$$q^{(t)}(\bar{u}, u|\mathbf{y}) = p(U_i^{(t-1)} = \bar{u}, U_i^{(t)} = u | \mathbf{Y}_i = \mathbf{y}).$$

The following explicit expressions are available:

$$\begin{aligned}\hat{a}_{juy}^{(t)} &= \mathbb{E}[a_{juy}^{(t)}] = \sum_{i=1}^n I(y_{ij}^{(t)} = y) \cdot q^{(t)}(u|\mathbf{y}_i) = \sum_{\mathbf{y}} n_{\mathbf{y}} \cdot I(y_j^{(t)} = y) \cdot q^{(t)}(u|\mathbf{y}), \\ \hat{b}_u^{(t)} &= \mathbb{E}[b_u^{(t)}] = \sum_{i=1}^n q^{(t)}(u|\mathbf{y}_i) = \sum_{\mathbf{y}} n_{\mathbf{y}} \cdot q^{(t)}(u|\mathbf{y}), \\ \hat{b}_{\bar{u}u}^{(t)} &= \mathbb{E}[b_{\bar{u}u}^{(t)}] = \sum_{i=1}^n q^{(t)}(\bar{u}, u|\mathbf{y}_i) = \sum_{\mathbf{y}} n_{\mathbf{y}} \cdot q^{(t)}(\bar{u}, u|\mathbf{y}).\end{aligned}$$

In particular,  $\hat{a}_{juy}^{(t)}$  must be computed for  $j = 1, \dots, r$ ,  $u = 1, \dots, k$ ,  $y = 0, \dots, c - 1$ , and  $t = 1, \dots, T$ ,  $\hat{b}_u^{(t)}$  must be computed for  $u = 1, \dots, k$  and  $t = 1, \dots, T$ , whereas  $\hat{b}_{\bar{u}u}^{(t)}$  must be computed for  $\bar{u}, u = 1, \dots, k$  and  $t = 2, \dots, T$ .

- **M-step:** maximize the expected value of  $\ell^*(\boldsymbol{\theta})$  obtained as above. As shown in [Bartolucci et al. \(2013\)](#), the following explicit solutions are available for this aim:

– *Initial probabilities:*

$$\pi_u = \frac{\hat{b}_u^{(1)}}{n},$$

to be computed for  $u = 1, \dots, k$ ;

– *Transition probabilities:*

$$\pi_{u|\bar{u}}^{(t)} = \frac{\hat{b}_{\bar{u}u}^{(t)}}{\hat{b}_{\bar{u}}^{(t-1)}},$$

to be computed for  $\bar{u}, u = 1, \dots, k$  and  $t = 2, \dots, T$ ;

– *Conditional response probabilities:*

$$\phi_{jy|u} = \frac{\sum_{t=1}^T \hat{a}_{juy}^{(t)}}{\sum_{t=1}^T \hat{b}_u^{(t)}},$$

to be computed for  $j = 1, \dots, r$ ,  $y = 0, \dots, c - 1$ , and  $u = 1, \dots, k$ .

A fundamental point is the computation of the posterior probabilities  $q^{(t)}(u|\mathbf{y}) = p(U_i^{(t)} = u | \mathbf{Y}_i = \mathbf{y})$  and  $q^{(t)}(\bar{u}, u|\mathbf{y}) = p(U_i^{(t-1)} = \bar{u}, U_i^{(t)} = u | \mathbf{Y}_i = \mathbf{y})$  involved in the E-step. To this aim, we can rely on an efficient backward-forward recursion, strictly related to the forward recursion used for the computation of the manifest distribution ([Baum et al., 1970](#); [Welch, 2003](#)).

### 1.2.2.2 Hidden Markov model for continuous response variables

Let  $\mathbf{Y}_i^{(t)}$ ,  $t = 1, \dots, T$  denote the vector of  $r$  continuous response variables at time occasion  $t$  and for subject  $i$ . We assume that, conditionally on  $U_i^{(t)} = u$ ,  $\mathbf{Y}_i^{(t)}$  a Gaussian distribution, that is,

$$\mathbf{Y}_i^{(t)} | U_i^{(t)} = u \sim \mathcal{N}(\boldsymbol{\mu}_u, \boldsymbol{\Sigma}), \quad u = 1, \dots, k, \quad (1.6)$$

with state-specific mean vectors  $\boldsymbol{\mu}_u \in \mathbb{R}^r$ ,  $u = 1, \dots, k$ , and variance-covariance matrix  $\boldsymbol{\Sigma} \in \mathbb{R}^{r \times r}$  constant across latent states under the assumption of homoscedasticity. This latter assumption may be suitably relaxed to allow for heteroscedasticity across latent states.

Taking into consideration the usual constraints on probability vectors, the number of free parameters is

$$\# \text{par} = \underbrace{k-1}_{\pi_u} + \underbrace{(T-1)k(k-1)}_{\pi_{u|\bar{u}}^{(t)}} + \underbrace{kr}_{\boldsymbol{\mu}_u} + \underbrace{\frac{r^2+r}{2}}_{\boldsymbol{\Sigma}}$$

if the model is time-heterogeneous, and reduces to

$$\# \text{par} = \underbrace{k-1}_{\pi_u} + \underbrace{k(k-1)}_{\pi_{u|\bar{u}}} + \underbrace{kr}_{\boldsymbol{\mu}_u} + \underbrace{\frac{r^2+r}{2}}_{\boldsymbol{\Sigma}}$$

in the event of a time-homogeneous model. Again, let us collect the model parameters in the vector  $\boldsymbol{\theta}$ :  $\boldsymbol{\theta} = (\pi_u, \pi_{u|\bar{u}}^{(t)}, \boldsymbol{\mu}_u, \boldsymbol{\Sigma})_{u, \bar{u}, t}$ .

On the basis of this set of model parameters, the probability mass function of the distribution of the latent process  $\mathbf{U}_i$  may be expressed as

$$p(\mathbf{U}_i = \mathbf{u}) = \pi_{u^{(1)}} \prod_{t=2}^T \pi_{u^{(t)} | u^{(t-1)}},$$

where  $\mathbf{u} = (u^{(1)}, \dots, u^{(T)})'$  denotes a realization of  $\mathbf{U}_i$ . Moreover, according to (1.6), the conditional distribution of  $\mathbf{Y}_i^{(t)}$  given  $U_i^{(t)}$  simply follows the usual probability density function of a multivariate normal distribution:

$$f(\mathbf{Y}_i^{(t)} = \mathbf{y}^{(t)} | U_i^{(t)} = u) = (2\pi)^{-\frac{r}{2}} |\boldsymbol{\Sigma}|^{-\frac{1}{2}} \exp \left[ -\frac{1}{2} (\mathbf{y}^{(t)} - \boldsymbol{\mu}_u)' \boldsymbol{\Sigma}^{-1} (\mathbf{y}^{(t)} - \boldsymbol{\mu}_u) \right],$$

where  $\mathbf{y}^{(t)}$  denotes a realization of  $\mathbf{Y}_i^{(t)}$ . Finally, the resulting manifest distribution of  $\mathbf{Y}_i$  is

$$\begin{aligned} f(\mathbf{Y}_i = \mathbf{y}) &= \sum_{\mathbf{u}} \prod_{t=1}^T f(\mathbf{y}^{(t)} | u) p(\mathbf{u}) = \\ &= \sum_{\mathbf{u}} \prod_{t=1}^T f(\mathbf{y}^{(t)} | u) \pi_{u^{(1)}} \prod_{t=2}^T \pi_{u^{(t)} | u^{(t-1)}}. \end{aligned}$$

Similarly to the categorical case, an efficient computation of this distribution can be performed through a forward recursion method (Baum et al., 1970; Welch, 2003).

**Maximum likelihood estimation** In this case, accounting for the specific model parameters, the complete data log-likelihood function may be expressed as follows:

$$\ell^*(\boldsymbol{\theta}) = \sum_{i=1}^n \sum_{t=1}^T \sum_{u=1}^k z_{iu}^{(t)} \log f(\mathbf{y}_i^{(t)} | u) + \sum_{i=1}^n \sum_{u=1}^k z_{iu}^{(1)} \log \pi_u + \sum_{i=1}^n \sum_{t=2}^T \sum_{\bar{u}=1}^k \sum_{u=1}^k z_{i\bar{u}u}^{(t)} \log \pi_{u|\bar{u}},$$

where:

- $z_{iu}^{(t)} = I(u_i^{(t)} = u)$  is an indicator function equal to 1 if subject  $i$  is in latent state  $u$  at time occasion  $t$ ;
- $z_{i\bar{u}u}^{(t)} = I(u_i^{(t-1)} = \bar{u}) I(u_i^{(t)} = u)$  is an indicator function equal to 1 if subject  $i$  is in latent state  $\bar{u}$  at time  $t-1$  and moves to latent state  $u$  at time  $t$ .

As usual, this function provides the basis for the estimation of model parameters through the EM algorithm. In this specific setting, expectation and maximization steps are implemented as follows:

- **E-step:** compute the conditional expected values of the indicator functions  $z_{iu}^{(t)}$  and  $z_{i\bar{u}u}^{(t)}$ , given the observed data and the current value of the parameters. By substituting these indicator functions with the corresponding posterior expectation, we obtain the expected value of  $\ell^*(\boldsymbol{\theta})$ . In particular:

$$\begin{aligned} \hat{z}_{iu}^{(t)} &= p(U_i^{(t)} = u | \mathbf{Y}^t = \mathbf{y}_i^{(t)}), \quad t = 1, \dots, T, \quad u = 1, \dots, k, \\ \hat{z}_{i\bar{u}u}^{(t)} &= p(U_i^{(t)} = u, U_i^{(t-1)} = \bar{u} | \mathbf{Y}^t = \mathbf{y}_i^{(t)}), \quad t = 2, \dots, T, \quad \bar{u}, u = 1, \dots, k. \end{aligned} \tag{1.7}$$

In order to efficiently compute these quantities, we resort to forward-backward recursion; the implementation is similar to the one introduced for the categorical case.

- **M-step:** update the model parameters by maximizing the expected value of  $\ell^*(\boldsymbol{\theta})$  obtained as above. The following closed form solutions are available:

$$\begin{aligned}
 \boldsymbol{\mu}_u &= \frac{1}{\sum_{i=1}^n \sum_{t=1}^T \hat{z}_{iu}^{(t)}} \sum_{i=1}^n \sum_{t=1}^T \hat{z}_{iu}^{(t)} \mathbf{y}_i^{(t)}, \quad u = 1, \dots, k, \\
 \boldsymbol{\Sigma} &= \frac{1}{nT} \sum_{i=1}^n \sum_{t=1}^T \sum_{u=1}^k \hat{z}_{iu}^{(t)} (\mathbf{y}_i^{(t)} - \boldsymbol{\mu}_u)(\mathbf{y}_i^{(t)} - \boldsymbol{\mu}_u)', \\
 \pi_u &= \frac{\sum_{i=1}^n \hat{z}_{iu}^{(1)}}{n}, \quad u = 1, \dots, k, \\
 \pi_{u|\bar{u}} &= \frac{\sum_{i=1}^n \sum_{t=2}^T \hat{z}_{i\bar{u}u}^{(t)}}{\sum_{i=1}^n \sum_{t=2}^T \hat{z}_{iu}^{(t-1)}}, \quad \bar{u}, u = 1, \dots, k.
 \end{aligned} \tag{1.8}$$

### 1.2.3 Stochastic block model

While LC models are commonly applied to cross-sectional data, and HM models are typically employed in the analysis of longitudinal data (as well as of time-series), SB models represent a fundamental tool for dealing with network data.

**Notation and definitions** In mathematics and statistics, networks are usually referred to as *graphs*. In its simplest definition a graph  $\mathcal{G} = (\mathcal{V}, \mathcal{E})$  is made up of a set  $\mathcal{V} \neq \emptyset$  of  $n$  distinct nodes (or vertices, or points) and a set  $\mathcal{E}$  of edges (or lines, or links). For a given graph  $\mathcal{G}$ , it is often convenient to define the *adjacency matrix*  $A = (A_{ij})_{ij}$ . The generic element  $A_{ij}$  indicate whether nodes  $i$  and  $j$  are connected by an edge. Starting from this basic definition, many generalizations have been developed to account for more complex real-world systems. Directed edges were introduced to describe potential origin and destination of the interaction. Weighted edges allow us to model the strength of the relationship; this includes the special case of negative weights, with signed edges incorporating the distinction between constructive and damaging interactions. Multi-edges and self-loops describe the existence of more than one link between a pair of nodes, and the presence of edges connecting a node to itself, respectively. More recent developments include also the analysis of temporal graphs, where the interactions change dynamically across the time. We refer to [Whittaker \(1990\)](#) for a thorough review of graphical models.

SB models are an increasingly popular class of models in statistical analysis of graphs. Introduced in the early eighties in the field of social sciences ([Holland et al., 1983](#)), these models have successively flourished in many directions and application fields ([Snijders and](#)



Nowicki, 1997; Nowicki and Snijders, 2001). They assume that nodes are clustered into unobserved (or latent) groups and the connection probabilities between nodes (*i.e.*, the probability that two distinct nodes are connected into an edge) are driven by their group memberships.

**Model formulation** Considering a random graph with  $n$  nodes, let  $Y_{ij}$  denote the generic binary response variable ( $i, j = 1, \dots, n$ ), such that  $Y_{ij} = 1$  if there exists an edge connecting node  $i$  to node  $j$ , and  $Y_{ij} = 0$  otherwise. In this context, the set of response variables  $\mathbf{Y} = (Y_{ij})_{i,j=1,\dots,n}$  corresponds to the  $n \times n$  adjacency matrix of the graph. Hereafter we will rely on the following two assumptions on the response variables:

- $Y_{ij} = Y_{ji}$  for each pair of nodes  $i, j = 1, \dots, n$ , which restrict our focus on undirected graphs;
- $Y_{ii} = 0$  for each node  $i$ , which excludes self-loops.

Both assumptions may be suitably relaxed, extending the model to directed graphs containing self-loops; see Lee and Wilkinson (2019), among others, for a review of possible model specifications and extensions.

In the present formulation, the SB model relies on node-specific discrete latent variables  $U_i$  with  $k$  support points that describe the  $k$  unobserved blocks. Let  $\mathbf{U} = (U_1, \dots, U_n)$  denote the vector of the latent variables. Also in this case a form of local independence is assumed: variables  $Y_{ij}$  are conditionally independent given  $U_i = u$  and  $U_j = v$ . The path diagram of the corresponding model is represented in Figure 1.3. The model parameters are the probability that a node belongs to latent group  $u$ , denoted by

$$\pi_u = p(U_i = u), \quad u = 1, \dots, k,$$

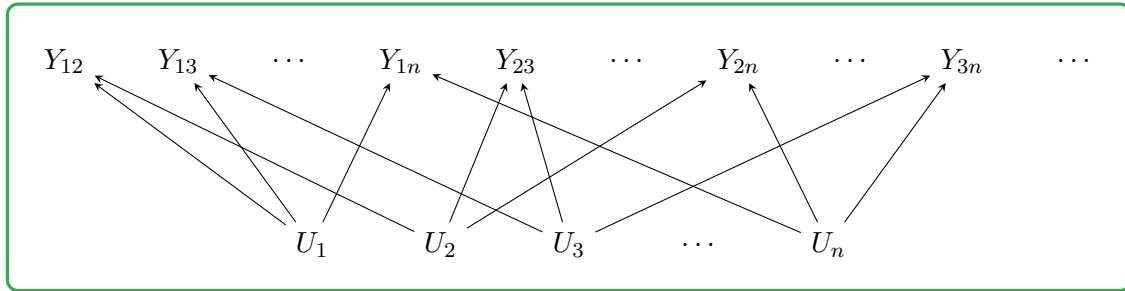
and the probability for a node from latent group  $u$  to be connected with a node from latent group  $v$ , denoted by

$$\pi_{uv} = p(Y_{ij} = 1 | U_i = u, U_j = v), \quad u, v = 1, \dots, k.$$

Since the graph is undirected, these probabilities must be symmetric:  $\pi_{uv} = \pi_{vu}$ , for each  $u, v = 1, \dots, k$ . This constraint, along with the usual ones on the above probabilities, determines the number of free parameters:

$$\#\text{par} = \underbrace{k - 1}_{\pi_u} + \underbrace{\frac{k^2 + k}{2}}_{\pi_{uv}}.$$

Finally, we let  $\boldsymbol{\theta} = (\pi_u, \pi_{uv})_{u,v}$  denote the vector of model parameters.



**Figure 1.3:** Path diagram of the stochastic block model.

The observed network distribution, corresponding to the model likelihood, may be obtained as

$$\begin{aligned} p(\mathbf{Y} = \mathbf{y}) &= \sum_{\mathbf{u}} p(\mathbf{Y} = \mathbf{y} | \mathbf{U} = \mathbf{u}) p(\mathbf{U} = \mathbf{u}) \\ &= \sum_{\mathbf{u}} \left[ \prod_{i=1}^{n-1} \prod_{j=i+1}^n p(Y_{ij} = y_{ij} | U_i = u_i, U_j = u_j) \right] \left[ \prod_{i=1}^n \pi_{u_i} \right], \end{aligned}$$

where  $\mathbf{y} = (y_{ij})_{ij}$  denotes a realization of  $\mathbf{Y} = (Y_{ij})_{ij}$  and  $\mathbf{u} = (u_1, \dots, u_n)$  is a realization of  $\mathbf{U} = (U_1, \dots, U_n)$ .

**Maximum likelihood estimation** Computing the above sum becomes prohibitive even when dealing with networks of limited size. This computational problem has a direct impact on the posterior distribution of the latent variables  $U_1, \dots, U_n$ , which becomes intractable itself, and hence on the EM algorithm which becomes complicated to apply. A typical remedy (Daudin et al., 2008) is to consider a variational approach: rather than maximizing the intractable log-likelihood function, this method aims at optimizing a lower bound of the log-likelihood itself. Denoting by  $\mathbb{Q}(\mathbf{U})$  a suitable approximation of the intractable posterior distribution  $p(\mathbf{U} = \mathbf{u} | \mathbf{Y} = \mathbf{y})$ , the lower bound is defined as

$$\mathcal{J}(\boldsymbol{\theta}) = \log p(\mathbf{Y} = \mathbf{y}) - \text{KL}[\mathbb{Q}(\mathbf{U}) || p(\mathbf{U} = \mathbf{u} | \mathbf{Y} = \mathbf{y})].$$

Here,  $\text{KL}[\cdot || \cdot]$  denotes the Kullback-Leibler divergence (Kullback and Leibler, 1951), measuring the (non-symmetric) distance between the two probability distribution. The variational EM algorithm alternates the following two steps until convergence; we refer to Daudin et al. (2008) for the explicit solutions of the two steps.

- **VE-step:** minimize the Kullback-Leibler divergence in order to determine the best approximation of  $p(\mathbf{U} = \mathbf{u} | \mathbf{Y} = \mathbf{y})$ ;
- **M-step:** maximize  $\mathcal{J}(\boldsymbol{\theta})$  in order to update the model parameters.

### 1.3 Outline and main contributions

In the present Chapter we provided basic background knowledge for the latent variable models based on a discrete distributions. In particular, we reviewed in some details specific classes of the models of interest, recalling the main underlying assumptions, summarizing the notations, and analyzing the standard MLE of model parameters through the EM algorithm. The remainder of this manuscript is comprised of separated and distinct research problems and is structured as follows.

In Chapter 2 we deal with the problem of multimodality of the log-likelihood function of DLV models, and the consequent possible convergence of the EM algorithm to a local maximum. We introduce a class of optimization methods, namely tempering or annealing, which employ a parameter known as temperature to re-scale the target function and monitor the prominence of all maxima. Exploiting the basic idea of these techniques, we propose a tempered EM algorithm to explore the parameter space adequately and increase the chance to reach the global maximum. We compare the proposal with the standard EM algorithm by an extensive Monte Carlo simulation study, evaluating both the ability to reach the global maximum of the log-likelihood function, and the computational time. We also employ the proposal on cross-sectional and longitudinal data referred to some application of interest, showing the main findings for LC and HM models respectively. All the results provide supporting evidence that the proposal outperforms the standard EM algorithm, and it significantly improves the chance to reach the global maximum.

In Chapter 3 we consider again the problem of convergence of the EM algorithm to a local maximum, proposing a different approach. We explore the framework of evolutionary algorithms, a class of optimization methods strongly inspired by the biological principles of natural evolution. We discuss the mechanism of the main evolutionary operators and propose their application in the context of the EM algorithm for DLV models estimation. This approach encourages a more accurate parameter space exploration and allows us to escape local maxima. The performance of the resulting algorithm is assessed relying on the same simulation scheme proposed in Chapter 2. This allow us to compare the two proposals, highlighting benefits and drawbacks of both approaches.

In Chapter 4, in the context of SB model, we tackle the need to account for higher-order

interactions, in order to include information deriving from groups of three or more subjects. We review the notion of hypergraphs and hyperedges, which extend the concept of graphs and edges respectively, and provide the most general mathematical formalization of higher-order interactions. In particular we distinguish the notion of “simple” hypergraphs, where hyperedges are subsets of *distinct* nodes taking part into an interaction, from the notion of “multisets” hypergraphs, where repeated nodes are allowed in the same hyperedge; we illustrate how a proper choice has to rely on the specificity of each dataset. In this work, we focus on model-based clustering for simple hypergraphs, where literature is quite scarce, and computational challenges increase. We propose a general SB model for simple hypergraphs which allows us to capture the information of higher-order interactions. We perform MLE of model parameters through a variational EM algorithm, and explore model selection using the ICL criterion. The model is applied to both simulated and real data, and the performance of the proposal is assessed in terms of parameter estimation and ability to recover the clusters.

## Bibliography

- AKAIKE, H. (1973). Information theory and an extension of the maximum likelihood principle.
- BACCI, S., PANDOLFI, S., AND PENNONI, F. (2014). A comparison of some criteria for states selection in the latent markov model for longitudinal data. *Adv. Data Anal. Classif.*, **8**, 125–145.
- BARTHOLOMEW, D., KNOTT, M., AND MOUSTAKI, I. (2011). *Latent Variable Models and Factor Analysis: A Unified Approach, 3rd Edition*. Wiley, Chichester.
- BARTOLUCCI, F. (2006). Likelihood inference for a class of latent markov models under linear hypotheses on the transition probabilities. *J. R. Stat. Soc. Ser. B*, **68**, 155–178.
- BARTOLUCCI, F., FARCOMENI, A., AND PENNONI, F. (2013). *Latent Markov Models for Longitudinal Data*. Chapman & Hall/CRC, Boca Raton.
- BARTOLUCCI, F., FARCOMENI, A., AND PENNONI, F. (2014). Latent markov models: A review of a general framework for the analysis of longitudinal data with covariates. *TEST*, **23**, 433–486.
- BARTOLUCCI, F., PANDOLFI, S., AND PENNONI, F. (2022). Discrete latent variable models. *Annu. Rev. Stat. Appl.*, **6**, 1–31.
- BARTOLUCCI, F., PENNONI, F., AND FRANCIS, B. (2007). A latent markov model for detecting patterns of criminal activity. *J. R. Stat. Soc. Ser. A*, **170**, 115–132.
- BAUM, L., PETRIE, T., SOULES, G., AND WEISS, N. (1970). A maximization technique occurring in the statistical analysis of probabilistic functions of markov chains. *Ann. Math. Stat.*, **41**, 164–171.
- BIERNACKI, C., CELEUX, G., AND GOVAERT, G. (2000). Assessing a mixture model for clustering with the integrated complete likelihood. *IEEE Trans. Pattern Anal. Mach. Intel.*, **22**, 719–725.
- BORSBOOM, D., MELLENBERGH, G., AND VAN HEERDEN, J. (2003). The theoretical status of latent variables. *Psychol. Rev.*, **110**, 203–219.
- BOUYEYRON, C., CELEUX, G., T.B., M., AND A.E., R. (2019). *Model-Based Clustering and Classification for Data Science: With Applications in R*. Cambridge University Press, Cambridge.

- DAUDIN, J., PICARD, F., AND ROBIN, S. (2008). A mixture model for random graphs. *Stat. Comput.*, **18**, 173–183.
- DEMPSTER, A., LAIRD, N., AND RUBIN, D. (1977). Maximum likelihood from incomplete data via the em algorithm (with discussion). *J. R. Stat. Soc. Ser. B*, **39**, 1–38.
- EVERITT, B. (1984). *An Introduction to Latent Variable Models*. Chapman & Hall/CRC, Boca Raton.
- GOODMAN, L. (1974). Exploratory latent structure analysis using both identifiable and unidentifiable models. *Biometrika*, **61**, 215–231.
- HOLLAND, P., LASKEY, K., AND LEINHARDT, S. (1983). Stochastic blockmodels: first steps. *Soc. Netw.*, **5**, 109–137.
- KULLBACK, S. AND LEIBLER, R. (1951). On information and sufficiency. *JSTOR*, **22**, 79–86.
- LAZARSELD, P. AND HENRY, N. (1968). *Latent Structure Analysis*. Houghton Mifflin, Boston.
- LEE, C. AND WILKINSON, D. (2019). A review of stochastic block models and extensions for graph clustering. *Appl. Netw. Sci.*, **4**, 1–50.
- LINDSAY, B. (1995). *Mixture Models: Theory, Geometry and Applications*. American Statistical Association, Arlington.
- LINDSAY, B., CLOGG, C., AND GREGO, J. (1991). Semiparametric estimation in the rasch model and related exponential response models, including a simple latent class model for item analysis. *J. Am. Stat. Assoc.*, **86**, 96–107.
- MACDONALD, I. AND ZUCCHINI, W. (2016). Hidden markov models for discrete-valued time series. In *Handbook of Discrete-Valued Time Series*, pages 267–286. Chapman & Hall/CRC, Boca Raton.
- MCLACHLAN, G. AND KRISHNAN, T. (2008). *The EM Algorithm and Extensions: 2nd Edition*. John Wiley and Sons, Hoboken.
- MCLACHLAN, G. AND PEEL, D. (2000). *Finite Mixture Models*. Wiley, New York.
- NOWICKI, K. AND SNIJDERS, T. (2001). Estimation and prediction for stochastic block-structures. *J. Am. Stat. Assoc.*, **96**, 1077–1087.

- PANDOLFI, S., BARTOLUCCI, F., AND PENNONI, F. (2021). Maximum likelihood estimation of hidden markov models for continuous longitudinal data with missing responses and dropout. *arXiv:2106.15948*, pages 1–36.
- SCHWARZ, G. (1978). Estimating the dimension of a model. *Ann. Stat.*, **6**, 461–464.
- SKRONDAL, A. AND RABE-HESKETH, S. (2004). *Generalized Latent Variable Modelling: Multilevel, Longitudinal and Structural Equation Models*. Chapman & Hall/CRC, Boca Raton.
- SNIJDERS, T. AND NOWICKI, K. (1997). Estimation and prediction for stochastic block-models for graphs with latent block structure. *J. Classif.*, **14**, 75–100.
- TITTERINGTON, D., SMITH, A., AND MAKOV, H. (1985). *Statistical Analysis of Finite Mixture Distributions*. Wiley, New York.
- VON EYE, A. AND CLOGG, C. (1994). *Latent Variables Analysis: Applications for Developmental Research*. SAGE Publications.
- WELCH, L. (2003). Hidden markov models and the baum-welch algorithm. *IEEE Inform. Theory Soc. Newsl.*, **53**, 9–13.
- WHITTAKER, J. (1990). *Graphical Models in Applied Multivariate Statistics*. John Wiley & Sons.
- WIGGINS, L. (1973). *Panel Analysis: Latent Probability Models for Attitude and Behaviour Processes*. Elsevier, Amsterdam.
- ZUCCHINI, W., MACDONALD, I., AND LANGROCK, R. (2016). *Hidden Markov Models for Time Series: An Introduction Using R, Second Edition*. Chapman & Hall/CRC, Boca Raton.





# Tempered Expectation-Maximization algorithm for the estimation of discrete latent variable models<sup>1</sup>

---

## 2.1 Introduction

As already mentioned in Section 1.1.2, maximum likelihood estimation (MLE) of discrete latent variable (DLV) models is usually performed by using the Expectation-Maximization (EM) algorithm (Baum et al., 1970; Dempster et al., 1977; McLachlan and Krishnan, 2008). A particular drawback of this approach is related to the multimodality of the log-likelihood function, which is especially observed with the DLV models. Consequently, the EM algorithm could converge to a local maximum, not corresponding to the global one. Multistart strategies employing both deterministic and random rules to initialize the model parameters are typically adopted. Although this approaches encourage a more accurate exploration of the parameter space, they are computationally intensive, and they do not ensure that the global maximum is reached.

In this Chapter, in order to face the multimodality of the likelihood function, we propose a tempered EM (T-EM) algorithm able to explore the parameter space adequately.

---

<sup>1</sup>Part of this Chapter has been published in: BRUSA, L., BARTOLUCCI, F, PENNONI, F. (2022). Tempered Expectation-Maximization algorithm for the estimation of discrete latent variable models. *Computational Statistics*. <https://link.springer.com/content/pdf/10.1007/s00180-022-01276-7.pdf>

Tempering and annealing (Kirkpatrick et al., 1983; Geyer, 1991; Geyer and Thompson, 1995; Falcioni and Deem, 1999) constitute a broad family of optimization methods; by means of a parameter known as temperature, they allow us to re-scale the target function and monitor the prominence of all maxima. In particular, these procedures are gradually attracted towards the global optimum by accurately defining a sequence of temperatures. The alternation of high and low values allows us to deal with two opposite but fundamental issues: on the one side, the algorithm is led to explore broad areas of the parameter space, thus escaping local sub-optimal solutions (high temperatures); on the other side, the algorithm is able to perform a sharp optimization of the target function in a small area of the parameter space (low temperatures).

In the following, dealing with DLV models, we propose a general approach. In particular, we explicitly focus on latent class (LC, see Lazarsfeld and Henry (1968), Goodman (1974), and Lindsay et al. (1991)) and hidden Markov (HM, see Wiggins (1973) and Bartolucci et al. (2013)) models because these are among the most utilized models in data analysis. However, the proposal can easily be adapted to the aforementioned finite mixture models and to other DLV models. We explore two different temperature sequences, including a non-monotone one, evaluating the ability to reach the global maximum along with the computational time efficiency by means of an extensive Monte Carlo simulation study. Up to our knowledge, for the first time, we deal with the problem of temperature sequence tuning, inspecting the performance of the T-EM algorithm with both optimally tuned and fixed temperature sequences. We also show the results for both LC and HM models, using the proposal on discrete and continuous, cross-sectional and longitudinal data in connection with some applications of interest.

The implemented code for the proposal is written for the open source software R (Team, 2022). It is based on some functions of the package `LMest` (Bartolucci et al., 2017), and it is available at the following link in the GitHub repository: <https://github.com/LB1304/T-EM>.

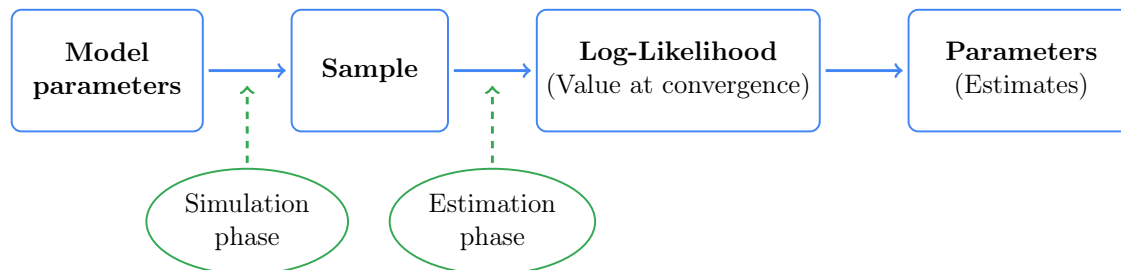
The rest of this Chapter is organized as follows. In Section 2.2, by means of an intensive simulation analysis, we illustrate the problem of multimodality for both LC and HM models. In Section 2.3 we introduce the theory of annealing and tempering methods. In Section 2.4 we provide details on the proposed T-EM algorithm for both models. In Section 2.5 we summarize the main findings of an extensive simulation study aimed to assess the performance of the proposal by comparing it with the standard EM algorithm for many different scenarios. In Section 2.6 we evaluate the proposed algorithm in connection with different initialization strategies. In Section 2.7 we apply the T-EM algorithm to estimate LC and HM models using a variety of data types. In Section 2.8 we provide some con-

clusion. Appendix A supplies more details of the settings used for the simulation studies, Appendices B and C provide additional simulation results, while Appendix D summarizes the performance comparison in terms of computational time for the real data analysis.

## 2.2 Convergence to local maxima

In this Section we address the problem of the convergence of EM algorithm to a local maximum of the log-likelihood function. To investigate the magnitude of this phenomenon, we carry out a simulation study, applied both to LC and to HM models with categorical response variables. This procedure, illustrated in Figure 2.1, consists of the following two steps:

1. given a chosen set of model parameters, we draw 50 samples from the corresponding model;
2. on the basis of every sample, we estimate the model parameters 100 times, using the EM algorithm with 100 different random starting values.



*Figure 2.1: Graphical representation of the implemented Monte Carlo simulation study*

For the whole study, the EM algorithm is implemented in the R programming language on the basis of the functions available in the two aforementioned packages. The convergence of the algorithm is checked on the basis of both the relative change in the log-likelihood at two consecutive steps, and the distance between the corresponding parameter vectors. As for the initialization of the algorithm, for all model parameters we adopt a random starting rule, based on suitably normalized random numbers drawn from a uniform distribution between 0 and 1. See also Section 2.5.1 for a more accurate description of these computational aspects.

### 2.2.1 Simulation study of multimodality for latent class model

As for the LC model, we test the convergence behavior of the EM algorithm considering two different scenarios with increasing levels of complexity, from an estimation point of view. In addition to the true values of the model parameters  $\pi_u$  (the weight of each latent class) and  $\phi_{jy|u}$  (the conditional probability of each response variable given the latent variable), the complexity of the scenario depends on: the sample size  $n$ , the number of response variables  $r$ , the number of categories of each response variable  $c$ , and the number of latent classes  $k$ . The two considered scenarios are summarized hereafter:

- **Scenario A.** The first scenario considers favorable conditions for the estimation algorithm: low values of  $r$  e  $c$ , and over all, a small number of latent classes. In more details, the adopted values are:

- $r = 6, c = 3, \text{ and } k = 3;$
- $\pi = [1/3, 1/3, 1/3]';$
- $\phi_{jy|u}$  defined by 
$$\begin{bmatrix} 0.7 & 0.15 & 0.1 \\ 0.2 & 0.7 & 0.2 \\ 0.1 & 0.15 & 0.7 \end{bmatrix}, \quad j = 1, \dots, r.$$

- **Scenario B.** The second scenario, instead, is based on more challenging conditions for the estimation algorithm, mainly due to the higher number of latent classes. In this case, the chosen values are:

- $r = 10, c = 6, \text{ and } k = 6;$
- $\pi = [1/6, 1/6, 1/6, 1/6, 1/6, 1/6]';$
- $\phi_{jy|u}$  defined by 
$$\begin{bmatrix} 0.95 & 0.25 & 0 & 0 & 0 & 0 \\ 0.05 & 0.65 & 0.1 & 0 & 0 & 0 \\ 0 & 0.1 & 0.7 & 0.2 & 0 & 0 \\ 0 & 0 & 0.2 & 0.7 & 0.1 & 0 \\ 0 & 0 & 0 & 0.1 & 0.65 & 0.05 \\ 0 & 0 & 0 & 0 & 0.25 & 0.95 \end{bmatrix}, \quad j = 1, \dots, r.$$

For each scenario, two different sample sizes are considered,  $n = 500$  and  $n = 5,000$ , for a total of four different configurations of parameters; for each of the 4 resulting situations, 50 different random samples from the corresponding LC model are simulated.

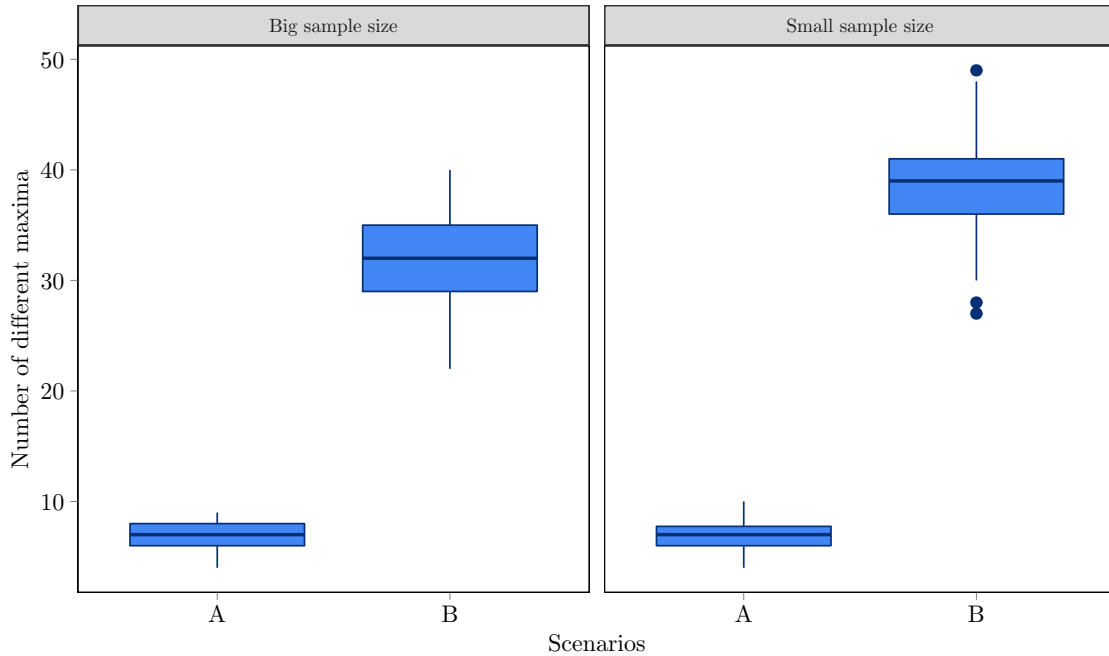
In the second phase of the simulation study, for every sample, we estimate 100 times both a correctly specified model, by selecting the true number of latent components (namely,

$k = 3$  for Scenario A, and  $k = 6$  for Scenario B), and a misspecified model, by considering a number of latent states different from the true one (in particular,  $k = 4$  for Scenario A and  $k = 7$  for Scenario B). Each estimation is initialized with a different random starting value, in order to test the overall convergence behavior of the EM algorithm. Finally, on the basis of these 100 estimation results, we count the number of different local maxima for each sample. Once log-likelihood values at convergence have been suitably ordered, a couple of consecutive log-likelihood values can correspond to two different maxima, if their distance is significant, or correspond to the same maximum (with a slightly variation due to approximation) otherwise. A tolerance threshold is therefore required in this process, in order to distinguish between the two cases. In this regard, throughout the simulation study, all values are suitably rounded.

**Results** Figure 2.2 summarizes the main results of the study, focusing on the case of misspecified models. The latter represents a realistic situation in many real-world situations, and allows us to best emphasize the extent of the problem of multiple local maxima. We refer to Section 2.5 for more simulation studies, including the analyses of results related to correctly specified models. Dealing with Scenario A, which should represent a sort of favorable situation for the estimation procedure, the corresponding plots in Figure 2.2 show the existence of a high number of different local optima. With none of the 50 analyzed samples, the EM algorithm is able to converge repeatedly to the same maximum; even in best cases, at least 4 different local maxima are reached, and this number grows to 10 different values in worst instances. Finally, by performing a comparison between the case with  $n = 500$  and that with  $n = 5,000$ , we can notice a very slight better behavior when a larger sample size is employed.

As expected, the situation is even much more complicated when the second scenario is considered. Here, the number of different local maxima dramatically increases: regardless of the sample size, the EM algorithm can converge, even in the best cases, to more than 20 different values, reaching up to about 50 distinct local maxima in the worst cases. Again, a slightly better situation can be observed with  $n = 5,000$ , compared to  $n = 500$ , thus confirming the theory that the frequency of reaching the global maximum increases with the sample size.

A second, but equally important, question is how often the EM algorithm gets to the global maximum: an hypothetical situation in which the EM algorithm converges to only two different optima, but the global maximum is reached less frequently than the local maximum is generally much worse with respect to a circumstance in which the EM algorithm

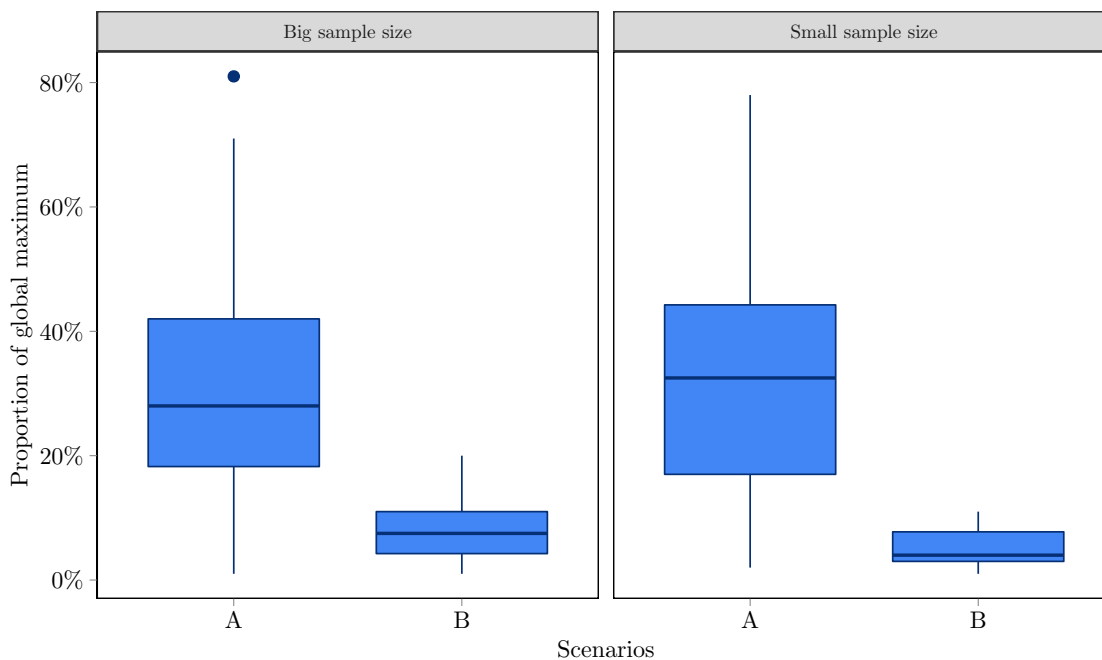


**Figure 2.2:** Number of different local maxima obtained estimating the LC model with the Expectation-Maximization algorithm; for each scenario, 50 samples are simulated and for each sample, the model is estimated 100 times using the Expectation-Maximization algorithm. The plots show the frequency of samples with a given number of different local maxima

can converge to a higher number of different optima, but the global maximum is reached with an overwhelming probability. In this regard, the results are shown in Figure 2.3.

Throughout the analysis, a distinction between the two scenarios is required; considering Scenario A (Figures 2.3), although the behavior is clearly problematic, we can still find some samples where the EM algorithm is able to reach the global maximum relatively often, or at least with a probability greater than 50%. All samples obtained according to the second scenario, instead, show a completely bad result: the probability to reach the global optimum is never sufficiently high, being always lower than 20%, and in most cases even not greater than 10%. Finally, the same conclusions, drawn before about the effect of different sample sizes, hold here.

To sum up, what emerges from the simulation study at issue is a rather critical situation: in the worst, but not infrequent, cases a wide number of different optima are obtained and the true global maximum is reached only in a small minority of times. Besides the obvious



**Figure 2.3:** Percentage of global maximum obtained estimating the latent class model with the Expectation-Maximization algorithm; for each scenario, 50 samples are simulated and for each sample, the model is estimated 100 times using the Expectation-Maximization algorithm. The plots show the frequency of samples with a given percentage

problem to achieve convergence toward this value, it is clear that in such a situation, it also appears uncertain that this value is the global maximum, and there is not another, higher value, which has never been reached.

Finally, a last interesting issue is the effect of these different optima of the log-likelihood function on the parameter estimates. In this regard, since we are purposely dealing with misspecified models, we cannot directly compare the true model parameters with their estimated values; then, we just show differences and similarities between the estimates corresponding to different maxima. We consider, for instance, a sample obtained according to the first scenario, with  $n = 5,000$ , in order to focus on the most favorable scenario from an estimation perspective. Consistently with the results just summarized, and as shown in Table 2.1, the considered sample exhibits 6 different maxima, and the global optimum is reached 17% of the times.

The impact on the estimated parameters is quite clear: by comparing the optimal estimates, corresponding to the global maximum, with the ones obtained in the case of the

|           |         |         |         |         |         |         |
|-----------|---------|---------|---------|---------|---------|---------|
| Value     | -28,690 | -28,688 | -28,687 | -28,686 | -28,685 | -28,684 |
| Frequency | 1       | 3       | 33      | 17      | 29      | 17      |

**Table 2.1:** Different values (with the corresponding frequency) obtained for the log-likelihood function on the basis of 100 repetitions of the Expectation-Maximization algorithm on the same sample, drawn from the latent class model according to scenario A and  $n = 5,000$

most frequent local optimum (equal to  $-28,687$ ), significant differences can be detected. In particular, in Table 2.2 the results for the weights  $\pi_u$  and the conditional probabilities  $\phi_{jy|u}$  (for the first response variable, i.e.  $r = 1$ ) are shown.

|  | Log-Likelihood = -28,687  | Log-Likelihood = -28,684  |
|--|---|---|
| Weights ( $\pi_u$ )  | [0.33, 0.33, 0.17, 0.17]'   | [0.02, 0.31, 0.33, 0.34]'   |
| Conditional probabilities for the first response ( $\phi_{jy u}$ for $j = 1$ ) | $\begin{bmatrix} 0.70 & 0.15 & 0.14 & 0.06 \\ 0.21 & 0.69 & 0.19 & 0.20 \\ 0.09 & 0.16 & 0.67 & 0.74 \end{bmatrix}$ | $\begin{bmatrix} 0.47 & 0.71 & 0.15 & 0.10 \\ 0.38 & 0.20 & 0.70 & 0.20 \\ 0.15 & 0.09 & 0.15 & 0.70 \end{bmatrix}$ |

**Table 2.2:** Estimated parameters (weight of each latent class and conditional probabilities of each response variable given the latent class) referred to the results shown in Table 2.1. The parameters resulting from the global maximum are shown on the right, the estimates obtained from the most frequent local maximum on the left

Even accounting for the label switching among latent states, we notice some very clear dissimilarities:

- considering the estimated parameter associated to the global maximum, we notice that the first class has a weight approximately equal to 0 zero, while analyzing the estimates related to the local optimum, all classes have a strictly positive weight;
- analogously, by taking into consideration the conditional probability matrix obtained in correspondence to the global maximum, we can observe that the first column (related to the first latent class) is completely different from every column of the same matrix in the local maximum case.

### 2.2.1.1 Simulation analysis of multimodality for Hidden Markov model

A simulation study similar to the previous one has been carried out for HM models; in this case, to assess the convergence behavior of the EM algorithm, four different configurations



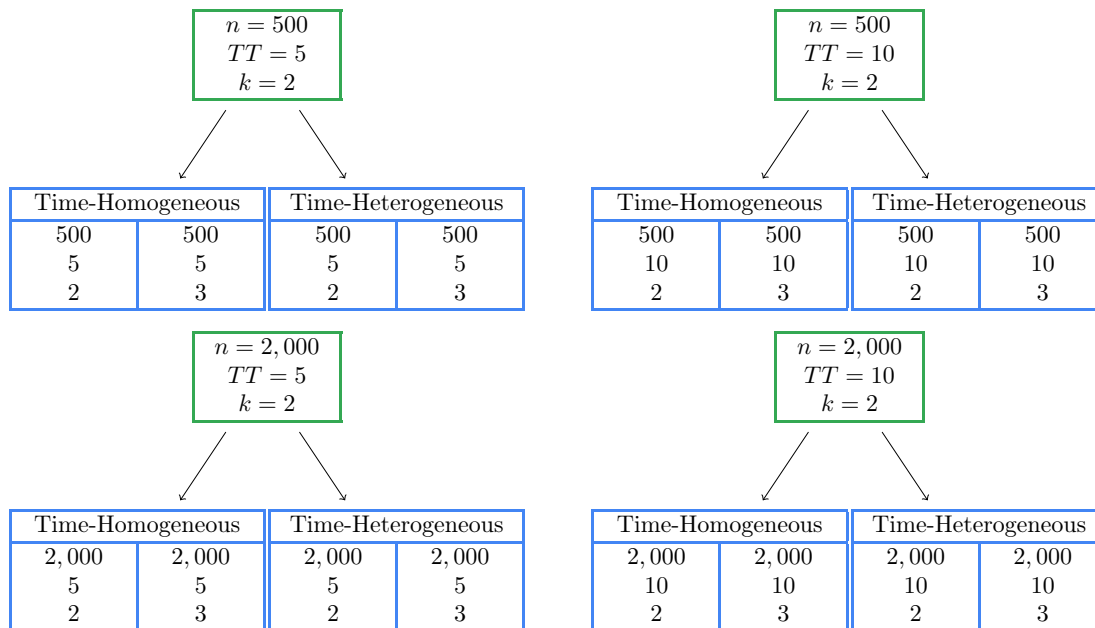
of the parameters, or scenarios, have been considered, according to the chosen sample size  $n$  and to the number of time occasions  $T$ ; in particular, samples are made up of 500 or 2,000 subjects and the study is referred to 5 or 10 time occasions. Moreover, for all samples 5 binary response variables are considered and 2 latent states are always assumed. In Table 2.3 the simulated scenarios are summarized. Let us remark that all samples are drawn from a time-homogeneous HM model, being  $\pi_{u|\bar{u}}^{(t)} = \pi_{u|\bar{u}}$ ,  $t = 2, \dots, T$ . Following the same procedure adopted in the simulation study for the LC model, for each of these scenarios we draw 50 different random samples.

| Notation               | Scenarios  |  |   |  |
|------------------------|--|--|---|--|
|                        | A  | B  | C   | D  |
| $n$                    | $n = 500$  | $n = 500$  | $n = 5,000$   | $n = 5,000$  |
| $T$                    | $T = 5$  | $T = 10$   | $T = 5$   | $T = 10$   |
| $k$                    | $k = 2$  |  |   |  |
| $r$                    | $r = 5$  |  |   |  |
| $c$                    | $c = 2$  |  |   |  |
| $\pi_u$                | $[0.5, 0.5]'$  |  |   |  |
| $\pi_{\bar{u}u}^{(t)}$ | $\underbrace{\begin{bmatrix} 0 & 0 \\ 0 & 0 \end{bmatrix}}_{t=0}$                                      | $\underbrace{\begin{bmatrix} 0.9 & 0.1 \\ 0.1 & 0.9 \end{bmatrix}}_{t=1}$                              | $\underbrace{\begin{bmatrix} 0.9 & 0.1 \\ 0.1 & 0.9 \end{bmatrix}}_{t=2}$ | $\dots, \underbrace{\begin{bmatrix} 0.9 & 0.1 \\ 0.1 & 0.9 \end{bmatrix}}_{t=T}$                       |
| $\phi_{jy u}$          | $\underbrace{\begin{bmatrix} 0.8 & 0.2 \\ 0.2 & 0.8 \end{bmatrix}}_{1^{st} \text{ response variable}}$ | $\underbrace{\begin{bmatrix} 0.8 & 0.2 \\ 0.2 & 0.8 \end{bmatrix}}_{2^{nd} \text{ response variable}}$ | $\dots,$  | $\underbrace{\begin{bmatrix} 0.8 & 0.2 \\ 0.2 & 0.8 \end{bmatrix}}_{r^{th} \text{ response variable}}$ |

**Table 2.3:** Summary of all values defining the four simulated scenarios (first phase) for the hidden Markov model with categorical response variables

In the second phase of the study, for each sample, 4 different models are estimated: a time-homogeneous model with  $k = 2$  latent states (it is the only model correctly specified); a time-homogeneous model with  $k = 3$  latent states; two time-heterogeneous models, with  $k = 2$  and  $k = 3$  latent states (all these models are misspecified because they are not in agreement with to the simulation scheme). Figure 2.4 summarizes all the simulated scenarios and the configurations of the estimated HM models.

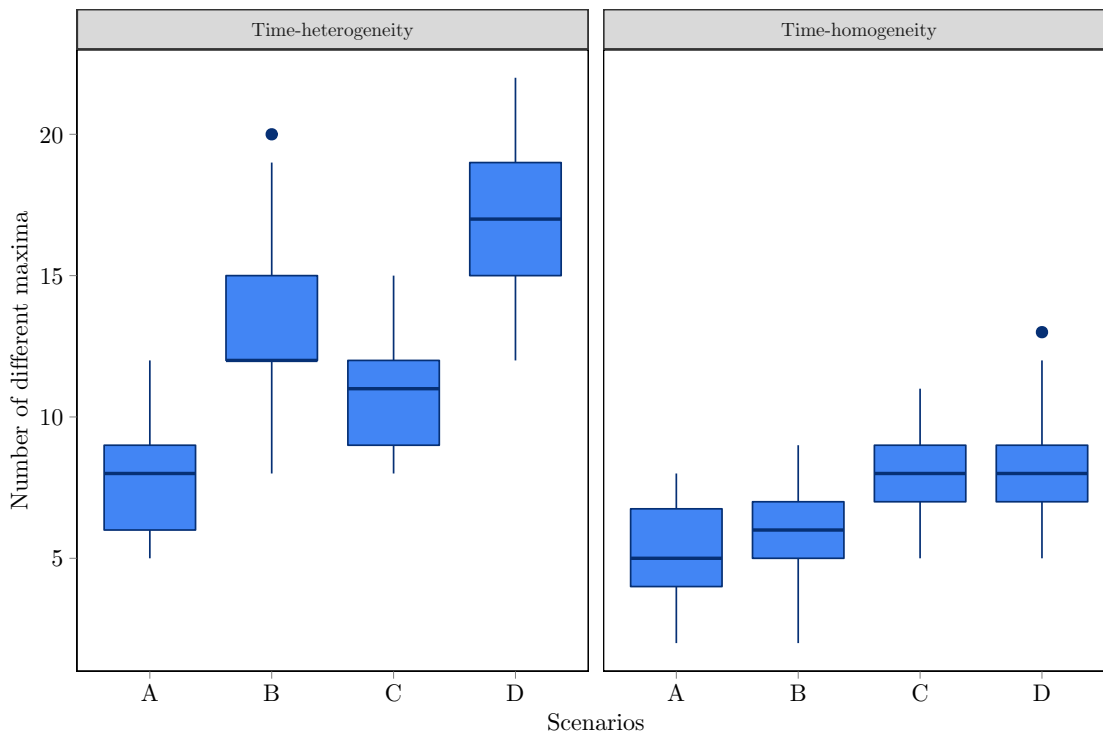
In order to monitor the convergence behavior of the EM algorithm, the log-likelihood function at convergence is evaluated many times: in particular, each model is estimated 100



**Figure 2.4:** Configurations of parameters considered for both the simulation (in green) and the estimation (in blue) phases for the hidden Markov model with categorical response variables. The involved parameters are the sample size  $n$  and the number of time occasions  $TT$  and of latent states  $k$

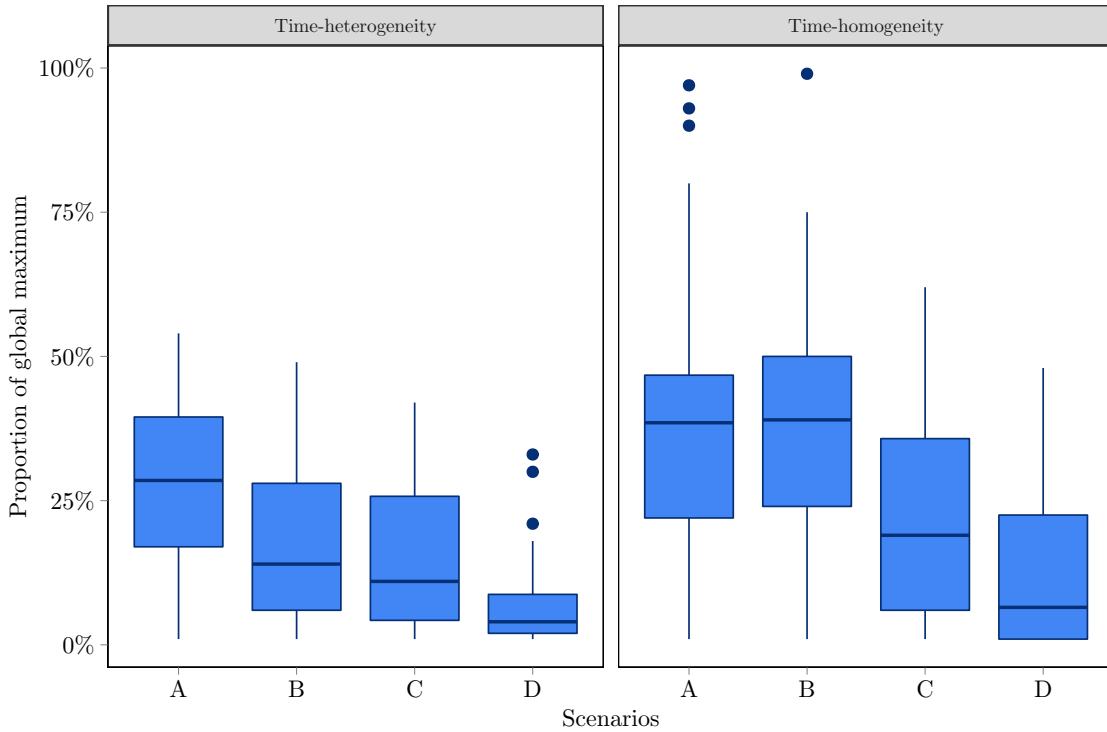
times on the same sample, starting from 100 random initial values. This procedure allows us to count the number of possible local maxima. The same remark expressed for LC model needs to be taken into consideration also in this context: two different values could either constitute two actual distinct maxima, or represent the same optimum with a slightly different approximation. Therefore, a tolerance threshold is needed to distinguish between these two alternatives; in the following, all log-likelihood values are suitably rounded.

**Results** Similarly to the previous simulation study, we only show the results obtained using the wrong number of latent components ( $k = 3$  for all scenarios). This analysis illustrate, once again, the extent of the problem of multiple local maxima: as shown in Figures 2.5 and 2.6, regardless of the considered scenario, a large number of different values is reached at convergence of the EM algorithm. In the following, we list the three factors affecting the number of local maxima in the simulation study, summarizing their actual observed effect.



**Figure 2.5:** Number of different local maxima obtained estimating the hidden Markov model with categorical response variables with the Expectation-Maximization algorithm; for each scenario, 50 samples are simulated and, for each sample, the model is estimated 100 times. The plots show the frequency of samples with a given number of different local maxima

1. An important feature that contributes to increase or decrease the number of different local maxima is the sample size. As already noted in the LC model analysis, in each scenario more local maxima are detected when the sample size increases. Moreover, we also detect a considerable decrease in the proportion of global maximum; in this regard, it is important to observe that the only rare samples for which the EM algorithm converges to a unique maximum, are for  $n = 500$  and when time-homogeneity is assumed.
2. Under a misspecified heterogeneous HM model, another important characteristic with a substantial impact on the number of local maxima is the number of the time occasions: increasing  $T$ , indeed, leads to a significant rise of the number of local maxima and to an important decrease in the proportion of global maxima. This behavior, however, does not occur when the model is assumed as time homogeneous.



**Figure 2.6:** Percentage of global maximum obtained estimating the hidden Markov model with categorical response variables with the Expectation-Maximization algorithm; for each scenario, 50 samples are simulated and, for each sample, the model is estimated 100 times. The plots show the frequency of samples with a given percentage

- Finally, the hypothesis of time-heterogeneity rather than time-homogeneity is the feature that seems to affect most the number of local maxima. As expected, all the misspecified models estimated assuming a time heterogeneous transition probability matrix show a significantly higher number of different values of local maxima. This behavior is particularly evident when the number of time occasions is high (e.g.,  $T = 10$  in our study).

In summary, when misspecified models are estimated, we detect the highest number of different local maxima when  $n$  is small and  $T$  is big; moreover, the models with time-heterogeneous transition probabilities always contribute to provide more local maxima. The most relevant conclusion that we can draw from this simulation study is that, irrespective of the selected scenario, the number of different local maxima is always extremely high.

Finally, it could be interesting to show in more detail the actual effect of the described behavior on the estimated HM model parameters. Let us consider, as an example, a single

sample obtained according to the third scenario ( $n = 5,000$  and  $T = 5$ ) in the time-homogeneous case; in Table 2.4 different values (rounded up to the nearest integer) assumed by the log-likelihood function after convergence of the EM algorithm are shown.

|           |                |         |         |         |         |         |         |
|-----------|----------------|---------|---------|---------|---------|---------|---------|
| Value     | $\leq -70,745$ | -70,744 | -70,743 | -70,742 | -70,741 | -70,740 | -70,739 |
| Frequency | 30             | 4       | 3       | 8       | 47      | 3       | 5       |

**Table 2.4:** Different values (with the corresponding frequency) obtained for the log-likelihood function after 100 repetitions of the Expectation-Maximization algorithm on the same sample, drawn from the hidden Markov model with categorical response variables according to scenario C and time-heterogeneity

In the case described above, the presence of many local maxima gives rise to a serious issue, since the highest value, equal to  $-70,739$ , occurs only 5 times (out of 100). Moreover, comparing the estimated parameters resulting in this instance, with the ones obtained in the case of the most frequent value (equal to  $-70,741$ ), we can detect significant differences. In particular, as shown in Table 2.5, even accounting for label switching among latent states, we notice significant dissimilarities:

- when the global maximum estimates are considered, the second latent state has initial probability very close to zero ( $\pi_2 = 0.01$ ); moreover, according to the transition probability matrix, the probability to move from a latent state to a different one is always very low (all elements outside the diagonal are close to zero);
- on the contrary, if the estimates corresponding to the local maximum are taken into consideration, all latent states have initial probability far from zero and the transition probability between any couple of latent states is generally higher.

## 2.3 Annealing and tempering techniques

In this Section we summarize the theory of annealing and tempering techniques, laying the groundwork for the implementation of the T-EM algorithm.

### 2.3.1 The origin of simulated annealing

The origin of the word *annealing* is in the metallurgy and material science (Aarts and Korst, 1989, Section 2.2), where annealing is a common heat treatments, which is used

|   | Log-Likelihood = -70,741   | Log-Likelihood = -70,739   |
|---|--|--|
| Initial Probabilities<br>( $\pi_u$ )              | [0.24, 0.25, 0.51]'  | [0.51, 0.01, 0.48]'  |
| Transition Probabilities<br>( $\pi_{u \bar{u}}$ ) | $\begin{bmatrix} 0.54 & 0.27 & 0.19 \\ 0.31 & 0.67 & 0.02 \\ 0.06 & 0.05 & 0.89 \end{bmatrix}$ | $\begin{bmatrix} 0.89 & 0.01 & 0.10 \\ 0.00 & 0.84 & 0.16 \\ 0.10 & 0.01 & 0.89 \end{bmatrix}$ |

**Table 2.5:** Estimated parameters (initial and transition probabilities) referred to the results shown in Table 2.4. The parameters resulting from the global maximum are shown on the right, the estimates obtained from the most frequent local maximum on the left

to alter physical, mechanical and sometimes chemical properties of metals; in particular, the metal is heated to a specific temperature, where re-crystallization can occur, and then cooled at a very slow and controlled rate. During this process, the free energy of the solid is minimized. Practice shows that the cooling must be done carefully, in order not to get trapped into locally optimal structure with crystal imperfections: if the decrease of temperature is sufficiently slow, a perfect crystalline solid will form.

In a mathematical context, we can define a similar process by establishing a strong underlying analogy between this physical annealing process of metals and the problem of solving complex optimization problems; this similarity, in particular, is based on the following equivalences:

- solutions in the optimization problem are equivalent to states of the metal physical system;
- the value of a solution is equivalent to the free energy of a state.

From these analogies, a class of mathematical and statistical techniques, widely employed in global optimization and probabilistic sampling, was developed. The origin of these methods can be traced back to the early 1980's, when the concepts of annealing were introduced in combinatorial optimization (Kirkpatrick et al., 1983). The resulting approach, known as simulated annealing (SA), soon became very popular, being used in various fields, such as image and signal processing (Chen and Luk, 1999), biology (Svergun, 1999), geophysics (Billings, 1994), finance (Crama and Schyns, 2003) and production managements (Koulamas et al., 1994).

Consider finding the global maximum of a non-convex function  $f(\boldsymbol{\theta})$  with respect to the vector  $\boldsymbol{\theta}$ . In general, an optimization procedure will lead to one of the local optima

closest to the starting point: maxima with higher values would only emerge through a more accurate exploration of the function profile. Annealing technique encourages such an exploration behavior by introducing a parameter  $\tau \in \mathbb{R}^+$ , known as the temperature, which provides a rescaling of the target function. In particular, following the general scheme of its metallurgical counterpart, SA operates by flattening the shape of the objective function and then gradually warping the substitute, flat profile towards the original, bumpy one. If this warping process is sufficiently slow, as all modes gradually reappear, the algorithm remains close enough to the dominant mode (Kirkpatrick et al., 1983).

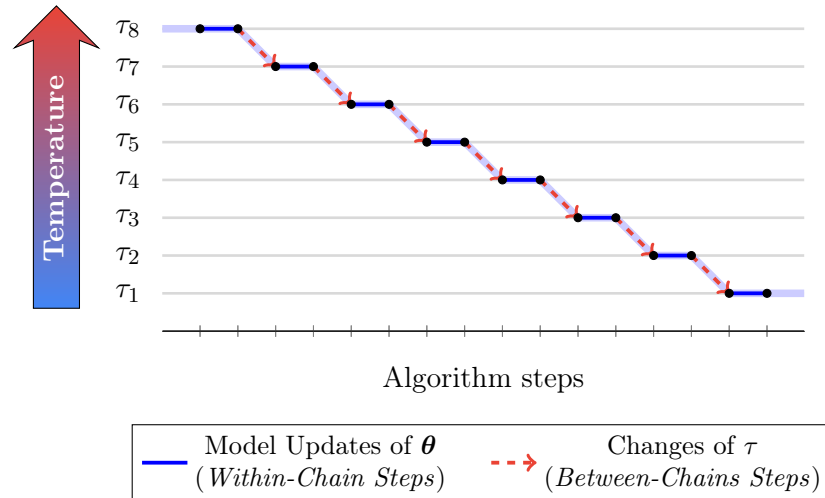
The procedure is based on the definition of the following general family of modified functions:

$$f^{(\tau)}(\boldsymbol{\theta}) \propto f(\boldsymbol{\theta})^{1/\tau}, \quad (2.1)$$

where the proportionality is up to possible normalizing constants; we refer to  $f^{(\tau)}(\boldsymbol{\theta})$  as an annealed (or tempered, or heated) version of  $f(\boldsymbol{\theta})$ . In this context, the value of  $\tau$  plays a key role, since:

- as temperature is raised, the shape of  $f^{(\tau)}(\boldsymbol{\theta})$  becomes relatively flat and all maxima are less pronounced; this makes it much easier to escape from local optima and allows exploring wide regions of the parameters space;
- as temperature is lowered, on the contrary, the shape of  $f^{(\tau)}(\boldsymbol{\theta})$  becomes similar to the original shape of  $f(\boldsymbol{\theta})$ ; this guarantees a sharp optimization in a local region of the parameters space.

SA makes use of a simple, strictly decreasing sequence of  $M$  temperature values  $(\tau_m)_{m=1\dots M}$ ; for each fixed value  $\tau_m$ , the procedure maximizes the corresponding annealed function  $f^{(\tau_m)}(\boldsymbol{\theta})$  and then moves to the next temperature level. The sequence needs to satisfy the following two fundamental requirements: (i) the initial temperature  $\tau_1$  should be sufficiently high so that the corresponding annealed function  $f^{(\tau_1)}(\boldsymbol{\theta})$  is relatively flat; (ii) for each level  $\tau_m$ , the sequence decreases to immediately following, lower, temperature  $\tau_{m+1}$  and it is not allowed us to increase back to a previous, higher, value:  $\mathbb{P}(\tau_{m+1} \mid \tau_m) = 1$ , where  $\mathbb{P}(\tau_{m+1} \mid \tau_m)$  denotes the probability to move from temperature  $\tau_m$  to temperature  $\tau_{m+1}$ . In this way, as temperature is adjusted from high to low values, the new objective function is increasingly more peaked, until the original profile is restored (in correspondence of  $\tau_M = 1$ ). Hence, SA involves deterministic transitions between two successive temperature levels; in general, in order to avoid to be entrapped in local optima, these transitions



**Figure 2.7:** Simulated annealing scheme: the procedure moves from high to low temperature levels  $(\tau_m)_{m=1,\dots,M}$ ; for each level  $\tau_m$ , the algorithm maximizes the annealed function  $f^{(\tau_m)}(\theta)$ .

must be done by taking small enough decrements in the temperature. Figure 2.7 illustrates a simple example of this procedure when  $M = 8$  temperature levels are considered.

### 2.3.2 The simulated tempering variant

Though SA provides a general framework that, in principle, can be applied to any optimization problem, its performance is deeply influenced by how slow temperature transitions occur. However, in general, precise rules are not provided in this regard: experience shows that this issue is problem-dependent and a specific tuning is often required for each application (Sambridge, 2014).

Simulated tempering (ST) (Geyer and Thompson, 1995; Sambridge, 2014) is a variant of annealing, that attempts to address this issue. The general structure is similar to SA: a strictly decreasing temperature sequence  $(\tau_m)_{m=1\dots M}$  is required and, for each fixed level  $\tau_m$ , the procedure seeks the maximum of the corresponding annealed (or tempered) function  $f^{(\tau_m)}(\theta)$ . In ST, however, temperature transitions gain two original elements; first, the temperature may either increase or decrease, according to specific probabilities: for each level  $\tau_m$ , we possibly have  $\mathbb{P}(\tau_{m+1} | \tau_m) \geq 0$  and  $\mathbb{P}(\tau_{m-1} | \tau_m) \geq 0$ , with the only, obvious constraint that  $\mathbb{P}(\tau_{m+1} | \tau_m) + \mathbb{P}(\tau_{m-1} | \tau_m) = 1$ . Secondly, the decision to change temperature is now stochastic: a new proposed level may be accepted or rejected according to a specific probability (Geyer and Thompson, 1995; Sambridge, 2014). Therefore, the



process describing temperature evolution follows a Markov chain. In particular, when a jump is proposed between temperatures  $\tau_m$  and  $\tau_l$ , with  $l = m \pm 1$ , the transition is accepted, in accordance with the Metropolis-Hastings rule (Hastings, 1970), with probability  $\lambda(m, l) = \min(1, r)$ , where

$$r = \frac{f^{(\tau_l)}(\boldsymbol{\theta}) c_l q_{m|l}}{f^{(\tau_m)}(\boldsymbol{\theta}) c_m q_{l|m}} \quad (2.2)$$

and  $c_m$  and  $c_l$  are normalizing constants:

$$c_m = \left[ \int f^{(\tau_m)}(\boldsymbol{\theta}) d\boldsymbol{\theta} \right]^{-1},$$

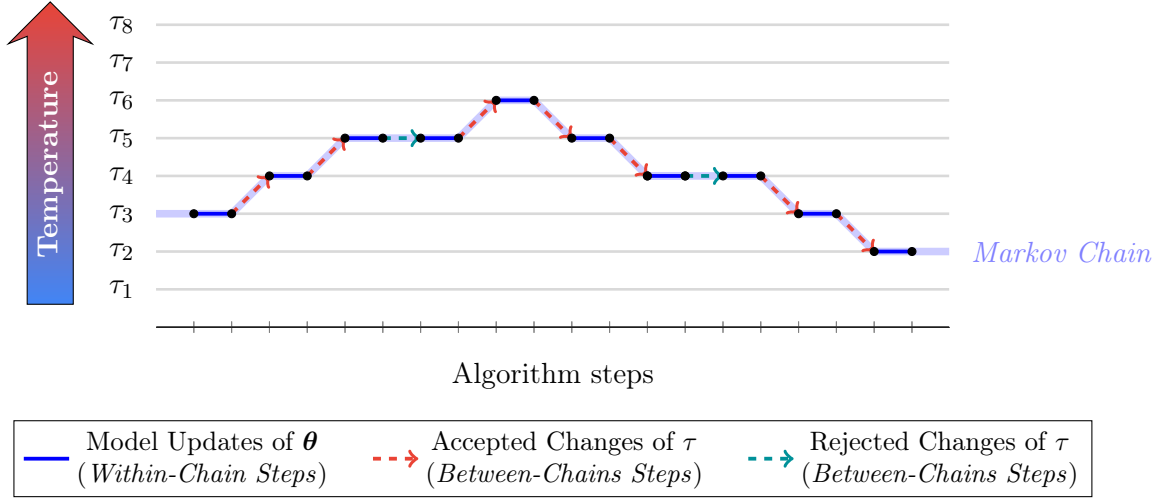
$$c_l = \left[ \int f^{(\tau_l)}(\boldsymbol{\theta}) d\boldsymbol{\theta} \right]^{-1}.$$

In the previous expressions  $q_{l|m} = \mathbb{P}(\tau_l | \tau_m)$  and  $q_{m|l} = \mathbb{P}(\tau_m | \tau_l)$  are the probabilities of proposing a move from temperature level  $m$  to  $l$  and from temperature level  $l$  to  $m$ , respectively. Standard values for ST are the following:  $q_{l|m} = q_{m|l} = 1/2$ , with  $q_{2|1} = q_{M|M-1} = 1$ .

It is straightforward to prove that ST becomes identical to SA when  $l = m + 1$  and  $\lambda = 1$ , but, in general, this will not be the case. In this regard, an important advantage of tempering over annealing is that the former guarantees a detailed balance between a finite set of temperature levels by satisfying the Metropolis-Hastings stochastic accept-reject rule, whereas the latter violates such condition (Earl and Deem, 2005). Conversely, in ST the computation of normalizing constants is required for evaluation of  $\lambda(m, l)$ ; this may result in difficulties in the implementation when such constants must be determined numerically. Figure 2.8 provides a simple example of ST with 8 temperature levels.

### 2.3.3 The parallel tempering version

Among the appealing features of ST, a considerable aspect is the possibility to include a parallel execution in a quite natural way; the resulting approach is known as parallel tempering (PT) (Geyer, 1991; Falcioni and Deem, 1999; Earl and Deem, 2005; Sambridge, 2014). In this case, a temperature sequence  $(\tau_m)_{m=1, \dots, M}$  is again employed and, for each fixed level  $\tau_m$ , the respective tempered function  $f^{(\tau_m)}(\boldsymbol{\theta})$  is maximized. The difference between ST and PT lies in the nature of transition between different temperature levels: indeed, instead of a single Markov chain either increasing or decreasing the value of temperature, in PT an



**Figure 2.8:** Simulated tempering scheme: a single Markov Chain moves randomly either up or down a level in the temperature sequence; for each level  $\tau_m$ , the algorithm maximizes the tempered function  $f^{(\tau_m)}(\theta)$ .

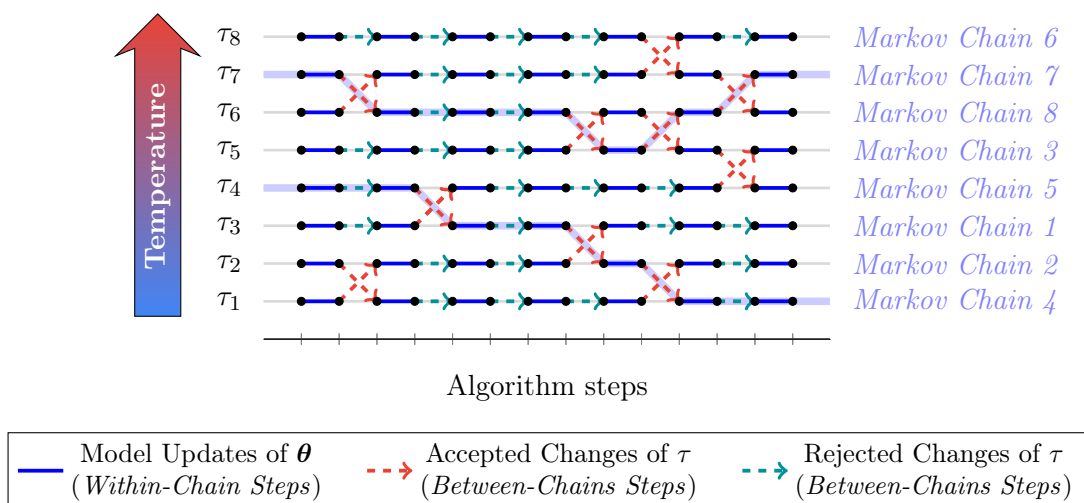
ensemble of Markov chains is distributed across all levels of the sequence and, at specified intervals, a swap between a pair of neighboring chains is proposed:

$$(\theta_m, \tau_m), (\theta_l, \tau_l) \rightarrow (\theta_m, \tau_l), (\theta_l, \tau_m), \quad m, l = 1, \dots, M, \quad l \neq m;$$

immediately before the proposed swap,  $\theta_m$  and  $\theta_l$  are current maximum points of  $f^{(\tau_m)}(\theta)$  and  $f^{(\tau_l)}(\theta)$ , respectively. As in ST, in order to guarantee the detailed balance property, the proposed swap is accepted with probability  $\lambda(m, l) = \min(1, r)$  (see also [Sambridge \(2014\)](#)), with

$$r = \frac{f^{(\tau_l)}(\theta_m) c_l q_{m|l}}{f^{(\tau_m)}(\theta_m) c_m q_{l|m}} \cdot \frac{f^{(\tau_m)}(\theta_l) c_m q_{l|m}}{f^{(\tau_l)}(\theta_l) c_l q_{m|l}} = \frac{f^{(\tau_l)}(\theta_m) f^{(\tau_m)}(\theta_l)}{f^{(\tau_m)}(\theta_m) f^{(\tau_l)}(\theta_l)}, \quad (2.3)$$

The above expression highlights a considerable/significant improvement of PT over ST: indeed, in the former, the computation of normalizing constants is no longer required, thereby ensuring a substantial reduction of the computational effort. It can also make efficient use of large CPU clusters, when maximization processes at different temperatures can be run in parallel. Figure 2.9 illustrates a simple example of this approach.



**Figure 2.9:** Parallel tempering scheme: an ensemble of Markov Chains is distributed across all levels of the temperature sequence, and at points along the Markov Chains, a swap of models at two neighboring temperature levels is proposed and accepted or rejected according to a certain probability; for each temperature level  $\tau_m$ , the algorithm maximizes the tempered function  $f^{(\tau_m)}(\boldsymbol{\theta})$ .

### 2.3.4 Other features

We conclude this Section with a brief list of final observations and remarks:

1. All methods illustrated above may be used in combination with many existing optimization techniques. In particular, a common approach is described in [Sambridge \(2014\)](#): instead of directly maximizing the objective function  $f(\boldsymbol{\theta})$ , it establishes to sample through a statistical approach from a distribution with the following probability density function:

$$\pi(\boldsymbol{\theta}, \tau) = e^{f(\boldsymbol{\theta})/\tau}.$$

Indeed, for  $\tau > 0$ , the maximum point of  $f(\boldsymbol{\theta})$  corresponds to the maximum point of  $\pi(\boldsymbol{\theta}, \tau)$ ; thereby, samples drawn from  $\pi(\boldsymbol{\theta}, \tau)$  will be attracted to the peak of the distribution and hence the global maximum of  $f(\boldsymbol{\theta})$ .

2. The main difference between SA and PT lies in the opposite way they distribute wide exploration and localized search across the procedure. SA clearly separates these two activities, by starting at high temperatures (that allow exploring wide regions of the solution space) and then gradually moving to lower temperatures (that guarantee a sharp optimization in a localized region of solution space). With PT, on the other

hand, at each iteration step, both activities spread across all temperature levels: while low temperature chains explore locally, higher temperature chains explore more globally and communication between these chains is always ensured by performing random swaps (see Figure 2.9). Therefore, SA requires a careful tuning of the temperature sequence, in order to avoid entrapment in local optima; in PT, instead, this issue is avoided.

3. The application of a tempering technique to the EM algorithm, however, faces many difficulties, mainly related to the nature of the objective function; in particular, since an explicit form for the log-likelihood function is often intractable, the employment of tempering requires a specific and careful adjustment and it makes it impossible the computation of the acceptance probability  $\lambda$  as defined through (2.2) or (2.3). Moreover, for the same reason we need to employ decreasing temperature levels during the iterations of the algorithm and tempering techniques do not ensure such behavior. Hence, in the following, we will consider two possible solutions:
  - (a) a standard SA procedure, which does not involve any acceptance probability computation and guarantees a strict decrease of temperature;
  - (b) an adjusted ST procedure, where temperature may either decrease or increase but, with respect to the original version, the accept-reject rule is removed and the temperature sequence has a general decreasing trend.

In both cases, the detailed balance condition is violated, and the temperature sequence needs to be carefully tuned; even so, tempering approach shows a clear advantage since the procedure will spend much time at all temperatures. A possible interesting development is to include a PT scheme. In the following, for simplicity, we will generally refer to both methods as “tempering”, since this procedure also includes annealing as a special case.

4. Tempering techniques are employed, among others, in [Barbu and Zhu \(2013\)](#) and [Robert et al. \(2018\)](#) for simulating from complex multimodal statistical distributions by means of Markov chain Monte Carlo methods ([Metropolis et al., 1953](#); [Hastings, 1970](#)). On the other hand, the use of these procedures within the EM algorithm is quite scarce. [Hofmann \(1999\)](#) proposed tempering techniques for the EM algorithm in the context of probabilistic latent semantic analysis. For what concerns finite Gaussian mixture models, recently, [Lartigue et al. \(2021\)](#) proposed a general class of deterministic approximated versions of the EM algorithm following previous proposals in [Yuille et al. \(1994\)](#), [Ueda and Nakano \(1998\)](#), and [Zhou and Lange \(2010\)](#).

## 2.4 Tempered Expectation-Maximization algorithm

The T-EM algorithm is implemented by adjusting the computation of the expected frequencies in the E-step. As illustrated in Section 1.2, this computation is based on conditional distributions, namely,  $q(u|\mathbf{y})$  for the LC model, and either  $q^{(t)}(u|\mathbf{y})$  or  $q^{(t)}(\bar{u}, u|\mathbf{y})$  for the HM model. In the following, generically referring to these probabilities as  $q(\cdot)$ , we show details of the T-EM algorithm for the LC and HM models, and we define some general rules for the tempering constants.

The family of tempered probabilities has the following expression:

$$\tilde{q}^{(\tau)}(\cdot) = m^{-1}q(\cdot)^{1/\tau}, \quad (2.4)$$

where  $q(\cdot)$  denotes the original conditional probability,  $\tau$  is a suitable value for the temperature parameter, varying over the interval  $[1, +\infty]$ , and  $m$  is a suitable normalizing constant in order to sum to 1. At each E-step of the T-EM algorithm, the conditional expected frequencies are computed accordingly.

### 2.4.1 Choice of the temperature parameter

Given the above setting, it is clear that the temperature value  $\tau$  has a deep impact on the performance of the proposed T-EM algorithm, and needs to be carefully tuned. In fact, as analyzed in Section 2.4, increasing the temperature value has the effect of flattening the profile of the log-likelihood, thereby reducing the chance that the algorithm will get trapped into a local maximum. In particular, the choice  $\tau \rightarrow +\infty$  yields  $\tilde{q}^{(\tau)}(\cdot)$  to a uniform distribution, while  $\tau = 1$  recovers the original posterior probability  $q(\cdot)$ . In other words, a decrease in the value of  $\tau \in (1, +\infty)$  implies a change in the shape of  $\tilde{q}^{(\tau)}(\cdot)$  from uniform to original distribution. Therefore, we define a sequence of temperature values  $(\tau_h)_{h \geq 1}$ , where  $h$  is the T-EM algorithm iteration number, so that:

1. the initial temperature value  $\tau_1$  is sufficiently large, implying that the corresponding tempered distribution  $q^{(\tau_1)}(\cdot)$  is relatively flat;
2. the temperature value  $\tau_h$  tends towards 1 as the algorithm iteration counter increases:  $\tau_h \rightarrow 1$  as  $h \rightarrow +\infty$ .

Any resulting sequence, denoted as *tempering profile*, guarantee a proper convergence of the algorithm (Lartigue et al., 2021). This high flexibility in the definition of the temperature sequence allows us to consider different tempering profiles, in order to evaluate and compare the performance of the resulting algorithm:

- a monotonically decreasing exponential profile, which is defined as

$$\tau_h = 1 + e^{\beta-h/\alpha}, \quad (2.5)$$

where,  $\alpha \geq 1$  and  $\beta \geq 0$  are two tuning constants chosen so as to ensure flexibility in the profile shape;

- a non-monotonic profile with oscillations of gradually smaller amplitude, which is expressed as

$$\tau_h = \tanh\left(\frac{h}{2\rho}\right) + \left(\tau_0 - \beta \cdot \frac{2\sqrt{2}}{3\pi}\right) \cdot \alpha^{h/\rho} + \beta \cdot \text{sinc}\left(\frac{3\pi}{4} + \frac{h}{\rho}\right), \quad (2.6)$$

with constants  $r, \tau_0, \beta > 0$  and  $0 < \alpha < 1$ . This profile has more parameters to tune, but it guarantees a very high level of flexibility. Here,  $\tanh(\cdot)$  indicates the hyperbolic tangent, while  $\text{sinc}(x) = \sin(\pi x)/(\pi x)$  (with  $\text{sinc}(0) = 1$ ) denotes the normalized sine cardinal function. In this case, the sequence  $(\tau_h)_{h \geq 1}$  could assume values that are smaller than 1 or even negative, for some combinations of the involved constants; although this is not an issue from a strictly mathematical perspective, a tempering step with negative temperature lacks a proper interpretation. Therefore, in practice, we can force the tempering profile to be always greater or equal to 1 by taking  $\tau_h = \max\{\tau_h, \delta\}$ , with  $\delta \geq 1$  (we fix  $\delta = 1$ ).

The abbreviations M-T-EM and O-T-EM are used for monotonic (2.5) and oscillating (2.6) tempering profiles.

### 2.4.2 Tuning of the tempering profiles

The selection of optimal tempering constants for both profiles may be carried out through a grid-search procedure; in the following, the term *grid* will denote the sequence of values considered for a constant, while the term *step-size* will refer to the distance between two consecutive values. For the monotonic profile the only two constants are simple to interpret:  $\beta$  controls the value of the initial temperature, while  $\alpha$  adjusts the decrease rate of the temperature. Lower values of both make the contribution of tempering insignificant; at the extreme,  $\alpha = 1$  and  $\beta = 0$  recover the standard EM algorithm. Although it is not possible to provide precise and rigorous rules for the selection of these constants, some guidelines hold in general: (i) avoid very high values of  $\alpha$  and  $\beta$ . Indeed, beyond certain values, the target function can not be flattened further, and only the computational time would increase.

This sort of “threshold” values are unfortunately data-dependent, but we recommend not exceeding  $\alpha = 15$  and  $\beta = 5$ ; *(ii)* choose step-sizes for each grid such that the distance between two consecutive values of  $\alpha$  will result much smaller than the one between two successive values of  $\beta$ . Indeed, the monotonic profile is much more sensitive to variations in  $\alpha$  than in  $\beta$ ; we suggest, for example, a ratio of about 1 to 10; *(iii)* avoid increasing  $\beta$  without a corresponding growth of  $\alpha$  (while the opposite has no shortcomings). This would lead to a fast decrease in the value of the temperature; accordingly, the target function would not be warped back to its original shape in a gradual way, and the algorithm could possibly be brought far from the global mode; *(iv)* typically, for each type of data there are many possible suitable tempering configurations, and an important step is to locate a rough range for the constants. After that, although the tuning process can be further refined, most of the tempering configurations chosen within that range would provide good results; *(v)* various factors such as number of observations, of response variables, and of latent components would guide the choice of this “unrefined” range. For example, estimating a model with many latent components typically requires higher values of  $\alpha$  and  $\beta$  with respect to a model with fewer components.

The same guidelines illustrated above should also be taken into account for the oscillating profile, where, however, there are more constants to tune. Their practical interpretation is, in this case, slightly different:  $T_0$  controls the initial temperature,  $\rho$  the distance between two consecutive peaks of the profile,  $\beta$  the amplitude of the oscillations, and  $\alpha$  the global decrease rate.

The following steps for tuning the tempering profile are derived from the aforementioned rules:

1. define grids for all the tempering constants, starting with large step-sizes;
2. estimate the model using the T-EM algorithm with these “unrefined” grids for the tempering constants employing a much smaller number of starting values with respect to that required with the EM algorithm;
3. identify the optimal tempering constants by comparing values of the log-likelihood function at convergence;
4. improve the tuning procedure, if necessary, in a smaller region of the tempering constants space and repeat the same procedure (points 2 and 3) using the same small number of different starting values.

The previous rules are successfully employed to estimate the models for the applications presented in Section 2.7.

A final note, which is effective for both profiles, is that in order to achieve a proper convergence, the algorithm needs to be run until the temperature is steadily close to 1. After that, the last step is conducted with the temperature precisely equal to 1 in order to retrieve the shape of the original log-likelihood function. Typically, this approach increases the number of steps that are required for the algorithm to converge, especially in the case of the oscillating profile. The code written for this proposal is implemented in R and it is freely available at the following link in the GitHub repository <https://github.com/LB1304/T-EM>.

### 2.4.3 T-EM algorithm for the latent class model with categorical response variables

In the following, we provide some details of the tempered distribution (2.4) defined for the LC model with categorical response variables considering a suitable tempering profile  $\tau_h$ :

$$\tilde{q}^{(\tau_h)}(u|\mathbf{y}_i) = \frac{q(u|\mathbf{y}_i)^{1/\tau_h}}{\sum_{v=1}^k q(v|\mathbf{y}_i)^{1/\tau_h}}.$$

The corresponding pseudo-code is shown in the box Algorithm 1. On the basis of this tempered distribution, the E-step and M-step of the resulting version of the algorithm are implemented as follows:

- **E-step:** compute the revised conditional expected values of  $a_{juy}$  and  $b_u$  revised according to the rules

$$\tilde{b}_u^{(\tau)} = \sum_{i=1}^n \tilde{q}^{(\tau)}(u|\mathbf{y}_i) \quad \text{and} \quad \tilde{a}_{juy}^{(\tau)} = \sum_{i=1}^n I(y_{ij} = y) \tilde{q}^{(\tau)}(u|\mathbf{y}_i); \quad (2.7)$$

to obtain the conditional expected value  $\mathcal{Q}(\boldsymbol{\theta}; \boldsymbol{\theta}^{(h-1)})$ .

- **M-step:** maximize  $\mathcal{Q}(\boldsymbol{\theta}; \boldsymbol{\theta}^{(h-1)})$ , thus updating the parameters as:

$$\pi_u^{(\tau)} = \frac{\tilde{b}_u^{(\tau)}}{n} \quad \text{and} \quad \phi_{jy|u}^{(\tau)} = \frac{\tilde{a}_{juy}^{(\tau)}}{\tilde{b}_u^{(\tau)}}. \quad (2.8)$$



**Algorithm 1** Tempered Expectation-Maximization algorithm for the latent class model with categorical response variables

---

- 1: Define a tempering profile  $(\tau_h)_{h \geq 1}$ .
  - 2:  $\boldsymbol{\theta} \leftarrow \boldsymbol{\theta}^{(0)}$  and  $h \leftarrow 0$ .
  - 3: **while** (Convergence Condition = FALSE) **do**
  - 4:      $h \leftarrow h + 1$ ;
  - 5:     **E-Step**: compute  $\tilde{a}_{j|uy}^{(\tau_h)}$  and  $\tilde{b}_u^{(\tau_h)}$ , as in (2.7);
  - 6:     **M-Step**: compute  $\pi_u^{(\tau_h)}$  and  $\phi_{j|y|u}^{(\tau_h)}$  as in (2.8).
  - 7: **end while**
- 

#### 2.4.4 T-EM algorithm for the hidden Markov model with categorical response variables

A more refined formulation for the tempered distribution in (2.4) is required to estimate the HM model with categorical response variables. Once the tempering profile  $\tau_h$  is chosen, we obtain the following tempered distributions:

$$\tilde{q}^{(t;\tau_h)}(u|\mathbf{y}_i) = \frac{q^{(t)}(u|\mathbf{y}_i)^{1/\tau_h}}{\sum_{v=1}^k q^{(t)}(v|\mathbf{y}_i)^{1/\tau_h}}$$

and

$$\tilde{q}^{(t;\tau_h)}(\bar{u}, u|\mathbf{y}_i) = \frac{q^{(t)}(\bar{u}, u|\mathbf{y}_i)^{1/\tau_h}}{\sum_{v=1}^k \sum_{\bar{v}=1}^k q^{(t)}(\bar{v}, v|\mathbf{y}_i)^{1/\tau_h}}.$$

The pseudo-code is shown in the box below Algorithm 2. In this setting, the steps of the T-EM algorithm are:

- **E-step**: compute the revised conditional expected value of every frequency  $a_{j|uy}^{(t)}$ ,  $b_u^{(t)}$ , and  $b_{\bar{u}u}^{(t)}$ , so as to obtain the conditional expected value  $\mathcal{Q}(\boldsymbol{\theta}; \boldsymbol{\theta}^{(h-1)})$ ; in particular, we have the following explicit expressions

$$\begin{aligned} \tilde{a}_{j|uy}^{(t;\tau_h)} &= \sum_{i=1}^n I(y_{ij}^{(t)} = y) \cdot \tilde{q}^{(t;\tau_h)}(u|\mathbf{y}_i) = \sum_{\mathbf{y}} n_{\mathbf{y}} \cdot I(y_j^{(t)} = y) \cdot \tilde{q}^{(t;\tau_h)}(u|\mathbf{y}), \\ \tilde{b}_u^{(t;\tau_h)} &= \sum_{i=1}^n \tilde{q}^{(t;\tau_h)}(u|\tilde{\mathbf{y}}_i) = \sum_{\tilde{\mathbf{y}}} n_{\tilde{\mathbf{y}}} \cdot \tilde{q}^{(t;\tau_h)}(u|\mathbf{y}), \\ \tilde{b}_{\bar{u}u}^{(t;\tau_h)} &= \sum_{i=1}^n \tilde{q}^{(t;\tau_h)}(\bar{u}, u|\mathbf{y}_i) = \sum_{\tilde{\mathbf{y}}} n_{\tilde{\mathbf{y}}} \cdot \tilde{q}^{(t;\tau_h)}(\bar{u}, u|\mathbf{y}). \end{aligned} \tag{2.9}$$

Similarly to the standard EM algorithm, the posterior probabilities  $\tilde{q}^{(t;\tau_h)}(u|\mathbf{y}_i)$  and

$\tilde{q}^{(t;\tau_h)}(\bar{u}, u|\mathbf{y}_i)$  may be efficiently computed by a backward recursion; see [Bartolucci et al. \(2013, pp. 61-64\)](#) for further details.

- **M-step:** by maximizing  $\mathcal{Q}(\boldsymbol{\theta}; \boldsymbol{\theta}^{(h-1)})$ , update the parameters as follows:

$$\pi_u^{(\tau_h)} = \frac{\tilde{b}_u^{(1; \tau_h)}}{n}, \quad \pi_{u|\bar{u}}^{(t; \tau_h)} = \frac{\tilde{b}_{\bar{u}u}^{(t; \tau_h)}}{\tilde{b}_{\bar{u}}^{(t-1; \tau_h)}} \quad \text{and} \quad \phi_{jy|u}^{(\tau_h)} = \frac{\sum_{t=1}^T \tilde{a}_{juy}^{(t; \tau_h)}}{\sum_{t=1}^T \tilde{b}_u^{(t; \tau_h)}}. \quad (2.10)$$

---

**Algorithm 2** Tempered Expectation-Maximization algorithm for the hidden Markov model with categorical response variables

---

- 1: Define a tempering profile  $(\tau_h)_{h \geq 1}$ .
  - 2:  $\boldsymbol{\theta} \leftarrow \boldsymbol{\theta}^{(0)}$  and  $h \leftarrow 0$ .
  - 3: **while** (Convergence Condition = FALSE) **do**
  - 4:      $h \leftarrow h + 1$ ;
  - 5:     **E-Step:** compute  $\tilde{a}_{juy}^{(t; \tau_h)}$ ,  $\tilde{b}_u^{(t; \tau_h)}$ , and  $\tilde{b}_{\bar{u}u}^{(t; \tau)}$ , as in (2.9);
  - 6:     **M-Step:** compute  $\pi_u^{(\tau_h)}$ ,  $\pi_{u|\bar{u}}^{(t; \tau_h)}$ , and  $\phi_{yj|u}^{(\tau_h)}$  as in (2.10).
  - 7: **end while**
- 

#### 2.4.5 T-EM algorithm for the hidden Markov model with continuous response variables

Regarding the HM model with continuous response variables, the pseudo-code is shown in box Algorithm 3. Similarly to the previous case, denoting by  $\tau_h$  a suitable tempering profile, the steps of the resulting T-EM algorithm are as follows:

- **E-step:** compute the conditional expected value  $\mathcal{Q}(\boldsymbol{\theta}; \boldsymbol{\theta}^{(h-1)})$  considering  $z_{iu}^{(t)}$  and  $z_{i\bar{u}u}^{(t)}$ :

$$\tilde{z}_{iu}^{(t;\tau_h)} = \tilde{q}^{(t;\tau_h)}(u|\mathbf{y}_i) \quad \text{and} \quad \tilde{z}_{i\bar{u}u}^{(t;\tau_h)} = \tilde{q}^{(t;\tau_h)}(\bar{u}, u|\mathbf{y}_i). \quad (2.11)$$

- **M-step:** maximize  $\mathcal{Q}(\boldsymbol{\theta}; \boldsymbol{\theta}^{(h-1)})$  and update the model parameters as follows:

$$\begin{aligned}
 \boldsymbol{\mu}_u^{(\tau_h)} &= \frac{1}{\sum_{i=1}^n \sum_{t=1}^T \tilde{z}_{iu}^{(t;\tau_h)}} \sum_{i=1}^n \sum_{t=1}^T \tilde{z}_{iu}^{(t;\tau_h)} \mathbf{y}_i^{(t)}, \\
 \boldsymbol{\Sigma}^{(\tau_h)} &= \sum_{i=1}^n \sum_{t=1}^T \sum_{u=1}^k \frac{\tilde{z}_{iu}^{(t;\tau_h)} (\mathbf{y}_i^{(t)} - \boldsymbol{\mu}_u) (\mathbf{y}_i^{(t)} - \boldsymbol{\mu}_u)'}{nT}, \\
 \pi_u^{(\tau_h)} &= \frac{\sum_{i=1}^n \tilde{z}_{iu}^{(1;\tau_h)}}{n}, \\
 \pi_{u|\bar{u}}^{(t;\tau_h)} &= \frac{\sum_{i=1}^n \sum_{t=2}^T \tilde{z}_{i\bar{u}u}^{(t;\tau_h)}}{\sum_{i=1}^n \sum_{t=2}^T \tilde{z}_{iu}^{(t-1;\tau_h)}}.
 \end{aligned} \tag{2.12}$$

---

**Algorithm 3** Tempered Expectation-Maximization algorithm for the hidden Markov model with continuous response variables

---

- 1: Define a tempering profile  $(\tau_h)_{h \geq 1}$ .
  - 2:  $\boldsymbol{\theta} \leftarrow \boldsymbol{\theta}^{(0)}$  and  $h \leftarrow 0$ .
  - 3: **while** (Convergence Condition = FALSE) **do**
  - 4:      $h \leftarrow h + 1$ ;
  - 5:     **E-Step:** compute  $\tilde{z}_{iu}^{(t;\tau_h)}$  and  $\tilde{z}_{i\bar{u}u}^{(t;\tau_h)}$ , as in (2.11);
  - 6:     **M-Step:** compute  $\boldsymbol{\mu}_u^{(\tau_h)}$ ,  $\boldsymbol{\Sigma}^{(\tau_h)}$ ,  $\pi_u^{(\tau_h)}$ , and  $\pi_{u|\bar{u}}^{(t;\tau_h)}$ , as in (2.12).
  - 7: **end while**
- 

## 2.5 Simulation study

We conduct an extensive Monte Carlo simulation study to evaluate the performance of the T-EM algorithm. In the following, we illustrate the simulation scheme for each different model specification and summarize the main results.

### 2.5.1 Settings of the experimental scenarios

In this Section we illustrate the simulation schemes for each different model specification. The settings involved in each model are related to sample size  $n$ , and number of response variables  $r$ , categories for each variable  $c$ , time occasions  $T$ , and latent components  $k$ . We define a baseline scenario (setting A, see Tables 2.17, 2.18, and 2.19 in Appendix A) for each model, characterized by  $n = 500$ ,  $r = 6$ ,  $c = 3$ ,  $T = 5$  and  $k = 3$ . In addition,

more scenarios (settings from B to F) are obtained by doubling, one at a time, the value of each feature. These settings are summarized in Tables 2.17, 2.18, and 2.19 in Appendix A, where also the values of the models' parameters are presented. Following the scheme already introduced in Section 2.3, for each scenario we draw 50 different samples.

For each of the simulated samples, we estimate 100 times both the model with correctly specified latent structure and that with misspecified latent structure, using each time different starting values randomly selected and employing the standard EM algorithm and the two proposed versions of the T-EM algorithm. The choice to also fit misspecified models allows us to show in more detail the features of the proposed tempering approach.

The convergence of the algorithms is checked on the basis of both the relative change in the log-likelihood of two consecutive steps, and the distance between the corresponding parameter vectors. We stop the algorithm when both criteria are satisfied:

$$\frac{\ell(\boldsymbol{\theta}^{(h)}) - \ell(\boldsymbol{\theta}^{(h-1)})}{|\ell(\boldsymbol{\theta}^{(h)})|} < \varepsilon_1$$

and

$$\max_s |\theta_s^{(h)} - \theta_s^{(h-1)}| < \varepsilon_2,$$

where  $\boldsymbol{\theta}^{(h)}$  is the parameter estimate obtained at the  $h$ -th iteration of the M-step and  $\varepsilon_1$  and  $\varepsilon_2$  are tolerance levels equal to  $10^{-8}$  and  $10^{-4}$ , respectively.

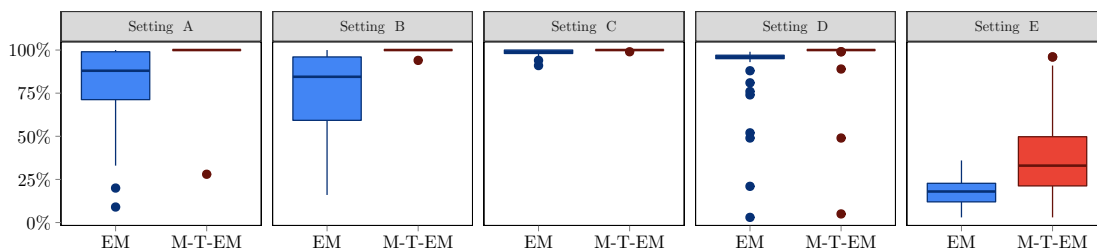
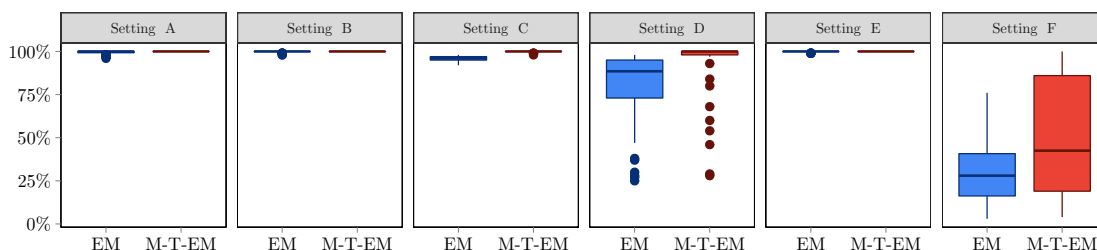
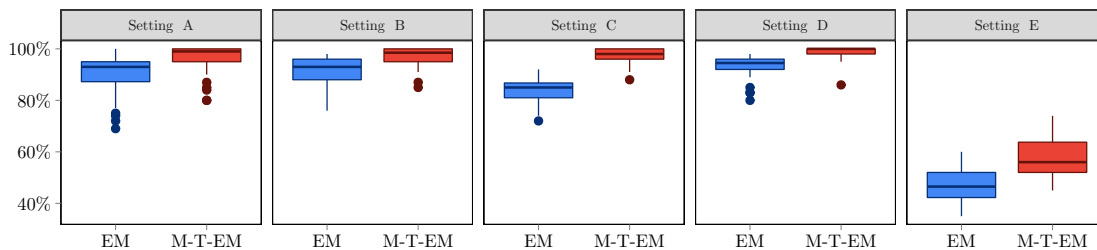
Regarding the algorithm initialization, we adopt a starting rule based on normalized random numbers (Bartolucci et al., 2013). In more details, each initial ( $\pi_u$ ) and transition ( $\pi_{u|\bar{u}}^{(t)}$ ) probability is initialized with a random number drawn from a uniform distribution between 0 and 1. Then, they are normalized such that  $\sum_{u=1}^k \pi_u = 1$  and  $\sum_{u=1}^k \pi_{u|\bar{u}}^{(t)} = 1$ . Similarly, we draw each  $\phi_{jy|u}$  from the uniform distribution and we normalize these parameters so that  $\sum_{y=0}^{c-1} \phi_{jy|u} = 1$ . In the case of continuous response variables, the mean vectors  $\boldsymbol{\mu}_u$  are drawn from a multivariate Gaussian distribution, whereas  $\boldsymbol{\Sigma}$  is initialized with the observed variance-covariance matrix.

### 2.5.2 Simulation results

The EM and T-EM algorithms are compared according to the following criteria:

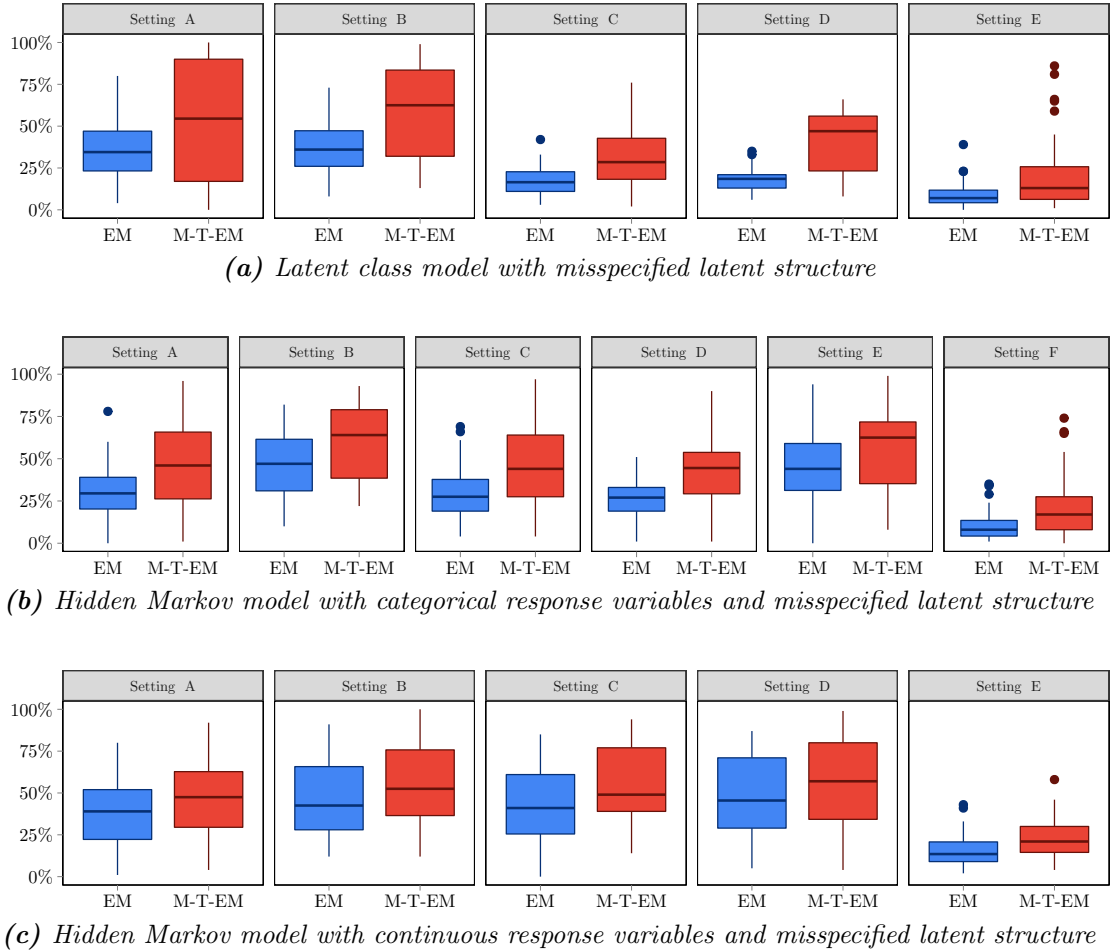
1. *Global maximum achievement*: the highest of the maximized log-likelihood values over all 100 initial values, denoted by  $\hat{\ell}_{\text{MAX}}$ , is considered as the global maximum, and a log-likelihood value at convergence denoted by  $\hat{\ell}$  is considered close to this value once it satisfies  $(\hat{\ell}_{\text{MAX}} - \hat{\ell})/|\hat{\ell}_{\text{MAX}}| < \tilde{\varepsilon}$ , where  $\tilde{\varepsilon}$  is a suitable threshold;

2. *Average distance from the global maximum* computed over the 100 log-likelihood values  $\hat{\ell}_1, \dots, \hat{\ell}_{100}$  and expressed as  $\sum_{s=1}^{100} (\hat{\ell}_{\text{MAX}} - \hat{\ell}_s)/100$ ;
3. *Low mean square error* of the estimated model parameters with respect to the true model parameters, computed only for models with a correctly specified latent structure;
4. *Low mean and median of the log-likelihood values at convergence.*

(a) *Latent class model with correctly specified latent structure*(b) *Hidden Markov model with categorical response variables and correctly specified latent structure*(c) *Hidden Markov model with continuous response variables and correctly specified latent structure*

**Figure 2.10:** Percentages of global maxima obtained using EM and M-T-EM algorithms under simulated scenarios presented in Tables 2.17, 2.18, and 2.19 of the Appendix A for the latent class and hidden Markov models with correctly specified latent structure

In particular, in this first part of the simulation study, we analyze the performance of the M-T-EM algorithm when the tempering profile is optimally tuned through a grid-search



**Figure 2.11:** Percentages of global maximum using EM and M-T-EM algorithms under simulated scenarios presented in Tables 2.17, 2.18, and 2.19 of the Appendix A for the latent class and hidden Markov models with misspecified latent structure

procedure. The following values for the tempering constants are kept fixed throughout the simulation studies:  $\alpha$  ranging from 1 to 15 with a step-size equal to 1 and  $\beta$  ranging from 0 to 2, with a step-size equal to 0.1. In order to show the flexibility of the method, we use the same grid for each model. However, efficient ad hoc grids may be set according to the model and observed data. The results are summarized in the following, while the full outcomes related to every sample of each simulated scenario are reported in the SI.

Criterion 1 is the most important, providing a suitable measure of performance of the algorithm. In this regard, the main results are summarized in Figures 2.10 and 2.11, representing the frequencies of global maximum with respect to the LC model, HM model

with categorical response variables, and HM model with continuous response variables, respectively. From all these figures it clearly emerges that the M-T-EM algorithm ensures better performance in each considered scenario.

Regarding the estimation of models whose latent structure is correctly specified, in particular (see Figure 2.10), the improvement with respect to the standard EM algorithm is very relevant: the M-T-EM is generally able to detect the global maximum in the overwhelming majority of cases, and the frequency of convergence to the global mode is very close, or even equal, to 100%. Only in estimating models with many latent states (up to 6), this percentage is slightly reduced, even if the M-T-EM still remains the algorithm providing the best performance. As an example, we consider the HM model with categorical response variables and in the particular setting F (see the last plot in Figure 2.10b): in this case the frequencies of convergence to the global maximum is, on average, equal to 29% when the standard EM algorithm is used, and up to 52% when the M-T-EM algorithm is employed. Moreover, this frequency is always lower than 75% with the EM, while it reaches 100% with M-T-EM (though only in a few cases).

All the algorithms are less efficient in steadily detecting the global mode when models with misspecified latent components are estimated (see Figures 2.11). The M-T-EM algorithm always provides the best performance, and in many scenarios the improvement is very relevant: in setting D of the LC model (Figure 2.11a) the frequency of convergence to the global mode increases from 18% to 41%; in setting C of the HM model with categorical responses (Figure 2.11b) for some samples this frequency reaches 100%.

| Scenario | Correctly specified |         |         | Misspecified |         |       |
|----------|---------------------|---------|---------|--------------|---------|-------|
|          | < 10%               | > 50%   | > 95%   | < 10%        | > 50%   | > 95% |
| A        | 1 - 0               | 43 - 49 | 20 - 49 | 7 - 2        | 9 - 25  | 0 - 8 |
| B        | 0 - 0               | 41 - 50 | 15 - 49 | 1 - 0        | 10 - 33 | 0 - 3 |
| C        | 0 - 0               | 50 - 50 | 47 - 50 | 9 - 4        | 0 - 9   | 0 - 0 |
| D        | 1 - 1               | 47 - 48 | 32 - 47 | 4 - 1        | 0 - 21  | 0 - 0 |
| E        | 10 - 5              | 0 - 13  | 0 - 2   | 32 - 18      | 0 - 5   | 0 - 0 |

**Table 2.6:** Number of samples in which the global maximum is reached with frequency < 10%, > 50%, or > 95%, using EM (highlighted in blue) and M-T-EM (highlighted in red) algorithms under simulated scenarios presented in Table 2.17 of the Appendix A for the latent class model

In Tables 2.6, 2.7, and 2.8, for each one of the simulated scenarios, we show the number of samples in which the global maximum is reached at least half of the times (> 50%), almost always (> 95%), or almost never (< 10%). These results provide supporting evidence for

| Scenario | Correctly specified |         |         | Misspecified |         |       |
|----------|---------------------|---------|---------|--------------|---------|-------|
|          | < 10%               | > 50%   | > 95%   | < 10%        | > 50%   | > 95% |
| A        | 0 - 0               | 50 - 50 | 50 - 50 | 4 - 1        | 6 - 24  | 0 - 1 |
| B        | 0 - 0               | 50 - 50 | 50 - 50 | 0 - 0        | 20 - 35 | 0 - 0 |
| C        | 0 - 0               | 50 - 50 | 35 - 50 | 4 - 1        | 5 - 21  | 0 - 1 |
| D        | 0 - 0               | 43 - 47 | 11 - 41 | 3 - 2        | 1 - 17  | 0 - 0 |
| E        | 0 - 0               | 50 - 50 | 50 - 50 | 3 - 1        | 20 - 32 | 0 - 2 |
| F        | 7 - 6               | 6 - 23  | 0 - 7   | 27 - 17      | 0 - 5   | 0 - 0 |

**Table 2.7:** Number of samples in which the global maximum is reached with frequency < 10%, > 50%, or > 95%, using EM (highlighted in blue) and M-T-EM (highlighted in red) algorithms under simulated scenarios presented in Table 2.18 of the Appendix A for the hidden Markov model with categorical response variables











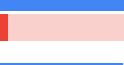





| Scenario | Correctly specified |         |         | Misspecified |         |       |
|----------|---------------------|---------|---------|--------------|---------|-------|
|          | < 10%               | > 50%   | > 95%   | < 10%        | > 50%   | > 95% |
| A        | 0 - 0               | 50 - 50 | 12 - 36 | 3 - 2        | 13 - 23 | 0 - 0 |
| B        | 0 - 0               | 50 - 50 | 14 - 36 | 0 - 0        | 20 - 26 | 0 - 1 |
| C        | 0 - 0               | 50 - 50 | 0 - 40  | 2 - 0        | 19 - 24 | 0 - 0 |
| D        | 0 - 0               | 50 - 50 | 18 - 47 | 4 - 4        | 20 - 28 | 0 - 1 |
| E        | 0 - 0               | 15 - 40 | 0 - 0   | 17 - 5       | 0 - 1   | 0 - 0 |

**Table 2.8:** Number of samples in which the global maximum is reached with frequency < 10%, > 50%, or > 95%, using EM (highlighted in blue) and M-T-EM (highlighted in red) algorithms under simulated scenarios presented in Table 2.19 of the Appendix A for the hidden Markov model with continuous response variables

the conclusions drawn so far. In particular, when the considered models are estimated with the correct latent structure, M-T-EM algorithm performs really well, significantly increasing the performance with respect to standard EM algorithm. For example, this enhancement is evident in setting C of the HM model with continuous response variables, where we observe that 40 samples reach the global mode with high frequency compared to none with the standard EM algorithm. An analogous improvement is noticeable for the case with 6 latent states but referred to the frequency of convergence to the global maximum more than half the time. In the case of models estimated with the wrong latent structure and many components we show another important result, not highlighted so far: the number of samples in which the global maximum is almost never reached (< 10% of times) diminishes when the M-T-EM algorithm is employed.



We also consider the mean distance from the global mode to measure how far is the obtained maximum from the global one. In particular, although all settings provide similar results, we notice that when dealing with correctly specified models, the mean distance decreases to zero when the M-T-EM algorithm is employed, thus confirming that the global maximum is almost always reached. In Appendix B, all detailed results are provided in Figures 2.16 and 2.17.

| Scenario | LC   | Categorical HM   | Continuous HM  |
|----------|--|--|--|
| A        |  0.0013<br>0.0012 |  0.0006<br>0.0002   |  0.0643<br>0.0272 |
| B        |  0.0007<br>0.0006 |  0.0003<br>0.0001   |  0.0556<br>0.0294 |
| C        |  0.0022<br>0.0010 |  0.0046<br>0.0003   |  0.1603<br>0.0433 |
| D        |  0.0020<br>0.0006 |  0.0027<br>0.0002   |  0.0322<br>0.0094 |
| E        |  0.0584<br>0.0544 |  0.0002<br>0.0001   |  0.1384<br>0.1168 |
| F        | -  |  0.0202<br>0.0179 | -  |

**Table 2.9:** Mean square errors of the estimated model parameters with respect to the true model parameters, using EM (highlighted in blue) and M-T-EM (highlighted in red) algorithms for each simulated scenario. Each value is computed as the average over 50 samples and 100 starting values as presented in Section 2.5.2. The true parameters for each model are summarized in Appendix A

Finally, we also provide the mean square error of the estimated model parameters with respect to the true ones, once the models are estimated with the correct latent structure. The results, summarized in Table 2.9, show that the mean square error values are always smaller with the M-T-EM algorithm than with the standard EM algorithm, thus highlighting that the estimated model parameters are more accurate by employing the former.

### 2.5.3 Results in terms of computational time

Having assessed the good performance of the proposed M-T-EM algorithm in locating the global maximum, we also compare the computational time required for convergence with that required by the EM algorithm for the same simulation settings illustrated above. Tempering constants are chosen as presented in Section 2.5.2. The estimation is performed

by employing an Intel(R) Core(TM) i7-8700T CPU @ 2.40GHz Windows desktop with 8 GB of RAM.

| Scenario                   | LC            | Categorical HM | Continuous HM   |
|----------------------------|---------------|----------------|-----------------|
| Correctly specified models |               |                |                 |
| A                          | 0.039 - 0.055 | 0.178 - 0.643  | 3.499 - 3.442   |
| B                          | 0.042 - 0.067 | 0.288 - 1.109  | 6.225 - 6.381   |
| C                          | 0.046 - 0.052 | 0.212 - 0.583  | 7.475 - 7.224   |
| D                          | 0.035 - 0.039 | 0.191 - 0.775  | 5.974 - 5.897   |
| E                          | 0.466 - 0.537 | 0.270 - 1.206  | 9.545 - 9.495   |
| F                          | -             | 1.728 - 11.237 | -               |
| Misspecified models        |               |                |                 |
| A                          | 0.205 - 0.484 | 1.114 - 7.282  | 11.114 - 12.480 |
| B                          | 0.268 - 0.348 | 2.045 - 13.258 | 18.646 - 21.016 |
| C                          | 0.294 - 0.407 | 1.396 - 6.747  | 24.670 - 29.081 |
| D                          | 0.244 - 0.364 | 1.173 - 7.365  | 19.981 - 23.852 |
| E                          | 0.581 - 0.630 | 2.022 - 13.217 | 18.513 - 19.714 |
| F                          | -             | 2.523 - 15.867 | -               |

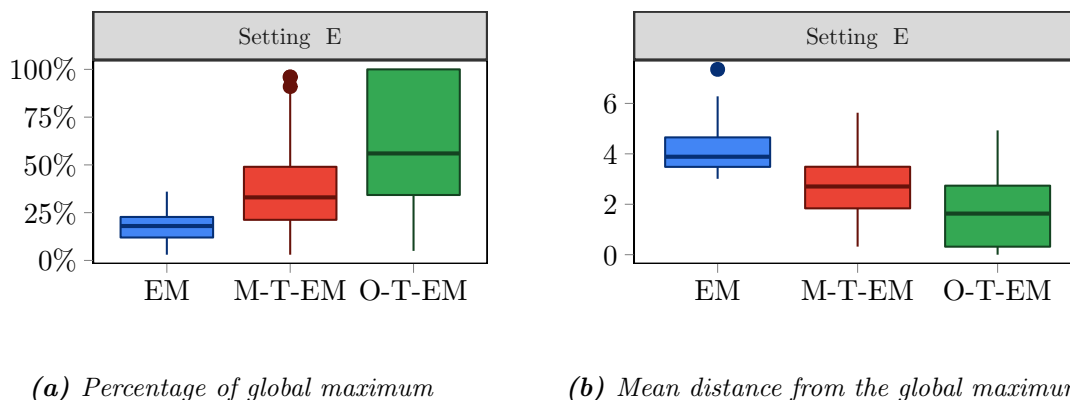
**Table 2.10:** Computational time in seconds of the EM (highlighted in blue) and M-T-EM (highlighted in red) algorithms for each simulated scenario, computed as the mean over 50 samples and 100 starting values as presented in Section 2.5.2

The main results, summarized in Table 2.10, show that when estimating LC and HM models with continuous response variables, the EM and M-T-EM algorithms show very similar computing times. The EM algorithm generally remains the fastest even if the difference with the M-T-EM is negligible. When dealing with correctly specified HM models with continuous response variables, M-T-EM algorithm is faster than the EM algorithm. Conversely, for the case of the HM model with categorical response variables it is the slower, requiring up to 6.5 times the computational time of the EM algorithm. These two opposite behaviors are due to the different implementations of the T-EM algorithm: the one for the HM model with categorical responses involves the addition of a for loop to the code, which is not required for the other two models.

#### 2.5.4 The role of the oscillating tempering profile

Although the M-T-EM algorithm ensures significant improvements in terms of ability to detect the global maximum, in some cases the frequency of convergence to this global mode remains inferior to 100%. A possible remedy is represented by the oscillating profile, which

is able to explore the parameter space more deeply than the monotonic one. In the following we focus only on the LC model, comparing the O-T-EM algorithm with the EM and M-T-EM algorithms; this is due to the higher computing time associated with this profile. The main results are summarized in Figure 2.12, where we show the percentage of times the global maximum is reached and the mean distance from the global maximum for the three versions of the algorithm.



(a) Percentage of global maximum

(b) Mean distance from the global maximum

**Figure 2.12:** Percentage of global maximum and mean distance from it using EM, M-T-EM, and O-T-EM algorithms on simulated data from an LC model correctly specified with 6 latent classes

Employing the oscillating profile, we notice a further improvement compared to the results analyzed in Section 2.5.2: the global maximum is reached on average about 18% of times with the standard EM algorithm, which increases up to 38% with the M-T-EM algorithm, and up to 60% with the oscillating version. It is also interesting to evaluate the number of samples in which the global maximum is reached almost surely ( $< 95\%$ ); this number, as reported in Table 2.6, was equal to 0 and 2 with EM and M-T-EM algorithms, respectively. Using the O-T-EM algorithm instead it increases to 18 samples. As for the mean distance from the global maximum, we notice that this value decreases accordingly, following the general advantage of the O-T-EM algorithm over the monotonic version. This optimal performance of the tempered algorithm with oscillating profile results, however, in a much higher computational time, as reported in Table 2.11.

The aspect mentioned above sometimes makes the employment of the O-T-EM algorithm rather complex; in particular, when it is applied to the HM model with categorical responses, the convergence is extremely slow, and the M-T-EM could be the most appropriate choice.

| Algorithm | Minimum | Median | Mean  | Maximum |
|-----------|---------|--------|-------|---------|
| EM        | 0.07    | 0.51   | 0.466 | 1.74    |
| M-T-EM    | 0.11    | 0.59   | 0.537 | 1.78    |
| O-T-EM    | 0.08    | 6.08   | 7.91  | 24.51   |

**Table 2.11:** Computational time in seconds of the EM, M-T-EM, and O-T-EM algorithms for the correctly specified latent class model with 6 latent classes, computed as the mean over 50 samples and 100 starting values, as presented in Section 2.5.2

### 2.5.5 Analysis of the tempered Expectation-Maximization algorithm with fixed tempering profile

Lastly, we check the performance of the T-EM algorithm when it is not optimally tuned, but the tempering constants are fixed in advance. With this aim, for each inspected scenario, a short list of different configurations of tempering constants is considered for applying the M-T-EM algorithm to all samples. In the analysis of the results, the tempered version is considered as the best choice only when it outperforms the standard EM algorithm with respect to all the four criteria introduced in Sect 2.5.2. Otherwise, if at least one criterion shows a better result with the standard EM algorithm, the latter is preferred. In this way, we carry out a very rigorous analysis.

Tables 2.20 and 2.21 in the Appendix C report for each scenario the configuration of tempering constants which exhibits the best performance. Results are highly satisfactory in most cases: given a fixed configuration, the M-T-EM algorithm outmatches the standard version in around 50% of samples in almost all the analyzed scenarios. In other words, once a configuration of tempering constants is set appropriately by a grid-search procedure over a specific sample, it generally remains valid for around 50% of other samples. This percentage increases up to 100% in some scenarios, especially when the latent structure of the model is correctly specified: the considered configuration of tempering constants provides optimal results in all samples. Similar results are achieved in the case of oscillating tempering profile analyzing setting E of the LC model when the latent structure is correctly defined: the best configuration of tempering constants ( $\alpha = 0.9$ ,  $\beta = 50$ ,  $\rho = 5$ , and  $T_0 = 10$ ) performs well with 62% of considered samples. It is clear that there are still some cases that require experimenting with the tempering constants to yield good performance; however, in our opinion, this represents a first significant improvement that allows avoiding specific settings for models and types of data.

## 2.6 Initialization of the tempered Expectation-Maximization algorithm

In this Section we consider different initialization strategies for the model parameters to evaluate the effect of different choices in detecting the global maximum and reducing the computational time.

Laird (1978) was the first to suggest a grid search; however, this method is unfeasible when parameter space has a high dimension. As suggested in Bartolucci et al. (2013), a multi-try strategy is typically adopted, combining deterministic and random rules. In particular, once an estimate  $\hat{\theta}_0$  is obtained starting with a deterministic rule, this approach suggests performing the algorithm again, starting from a suitable number  $R$  of randomly chosen points of the parameters space, and obtaining the estimates  $\hat{\theta}_1, \dots, \hat{\theta}_R$ . Then, we compare  $\ell(\hat{\theta}_0)$  with the maximum of  $\ell(\hat{\theta}_1), \dots, \ell(\hat{\theta}_R)$  and we take as global maximum of  $\ell(\theta)$  the solution corresponding to the highest log-likelihood value. In this way, provided that  $R$  is large enough, the random rule allows us to adequately explore the parameter space. However, this approach is computationally intensive and it may fail to reach the global maximum, in particular with many model parameters.

More refined initialization strategies have been investigated. Recently, Maruotti and Punzo (2021) carried out a detailed study about the effect of the initial values on the outcome of the EM algorithm and compared different initialization methods, some of which are based on preliminary cluster analysis (Everitt et al., 2011). In particular, they examine the following strategies:

- *Random Short EM*: this procedure establishes to perform a suitable number of “short” runs of the EM algorithm, each with random initialization. Every run is limited to a small number of iterations, without waiting for the reaching of convergence. The resulting parameter vector providing the largest likelihood is then used as initial value for the classic EM.
- *Partitional Clustering*: this method, proposed by Leroux and Puterman (1992) following McLachlan and Basford (1988), suggests to classify the observations into  $k$  clusters, where  $k$  is the number of latent states or latent classes of the model to be estimated; then the estimate derived by the application of this clustering algorithm is considered as the initial point of the EM algorithm. A typical example, when dealing with continuous responses, is represented by  $k$ -means clustering (Forgy, 1965; MacQueen, 1967), which partitions units so that the within-cluster sum of squares is minimized. Then the starting values for the EM algorithm are simply computed on

the basis of this partition; for instance the transition probabilities are obtained as the proportions of transition and persistence. A similar procedure for generating starting values based on data partitions when response variables are categorical makes use of the  $k$ -modes algorithm (Huang, 1998).

































- *Gaussian Mixtures*: this strategy prescribes to obtain a preliminary partition using Gaussian mixtures (Fraley and Raftery, 2002; Titterton et al., 1985); then the initial values for the EM algorithm are computed from this partition as in the previous method.

In the following we will focus on the partitioning clustering approach, being the method that provides the best results in Maruotti and Punzo (2021). Considering  $k$ -means and  $k$ -modes algorithms as partitioning clustering methods for categorical and continuous response variables respectively, initial values are computed as follows:

- proportion of observations assigned to cluster  $u$  at the first time occasion for the initial probabilities ( $\pi_u$ );
- proportion of transition (or persistence) estimated from cluster  $\bar{u}$  to cluster  $u$  for the transition probabilities ( $\pi_{u|\bar{u}}^{(t)}$ );
- proportion of observations assigned to cluster  $u$  who responded with category  $j$  to the response variable  $y$  for the conditional probabilities ( $\phi_{jy|u}$ );
- maximum likelihood estimator on the observations of cluster  $u$  for the mean vectors ( $\mu_u$ );
- maximum likelihood estimator on all the observations under the hypothesis of homoscedasticity for the variance-covariance matrix ( $\Sigma$ ).

We consider the same samples and starting values used in Section 2.6, comparing the performance of EM and M-T-EM algorithms. In general, when the estimation of correctly specified models is considered, the standard EM algorithm benefits from the adoption of a  $k$ -means initialization; using this kind of strategy, therefore, the results obtained with EM and M-T-EM algorithms are very similar.

































In the second column of Table 2.12, for each scenario, we report the number of samples in which the standard EM algorithm with  $k$ -means initialization does not converge to the global maximum, which is instead reached by the M-T-EM algorithm with the same starting values. It is important to remark that M-T-EM does not behave worse than the standard EM in all the other samples, but both algorithms converge to the same value (further

| Scenario                         | Perc. > EM<br>( $k$ -means)   | Perc. Glob. Max.<br>( $k$ -means)  | Iterations<br>(Random) | Iterations<br>( $k$ -means) |
|----------------------------------|---|--|------------------------|-----------------------------|
| LC model                         |   |  |                        |                             |
| A                                | 98%    | 98%     | 26.49                  | 25.10                       |
| B                                | 98%    | 98%     | 28.88                  | 28.32                       |
| C                                | 96%    | 100%    | 11.00                  | 6.82                        |
| D                                | 76%    | 92%     | 11.42                  | 8.54                        |
| E                                | 60%    | 0%      | -                      | -                           |
| HM model (categorical responses) |   |  |                        |                             |
| A                                | 62%    | 100%    | 10.09                  | 5.48                        |
| B                                | 58%    | 100%    | 10.00                  | 5.14                        |
| C                                | 0%     | 100%    | 8.89                   | 5.78                        |
| D                                | 76%    | 92%     | 12.53                  | 9.70                        |
| E                                | 38%    | 100%    | 9.50                   | 5.16                        |
| F                                | 84%    | 36%     | 176.47                 | 164.96                      |
| HM model (continuous responses)  |   |  |                        |                             |
| A                                | 14%    | 100%    | 42.73                  | 11.84                       |
| B                                | 8%     | 100%    | 37.97                  | 10.76                       |
| C                                | 0%    | 100%   | 45.39                  | 11.06                       |
| D                                | 0%   | 98%   | 34.21                  | 10.62                       |
| E                                | 28%  | 100%  | 83.89                  | 14.44                       |

**Table 2.12:** Percentage of samples in which the global maximum is reached by the M-T-EM algorithm with  $k$ -means initialization, but not by the standard EM algorithm with the same starting values, percentage of samples in which the M-T-EM algorithm with  $k$ -means initialization reaches the global maximum and number of iterations until convergence with random and  $k$ -means (or  $k$ -modes) initialization when the latent structure of the models is correctly specified

analyses conducted on correctly specified HM models with continuous response variables and  $k = 2$  latent states highlight that in such case the global maximum is always reached also by the EM algorithm with  $k$ -means initialization). We also compare random and  $k$ -means initializations for the M-T-EM algorithm. The results, summarized in the last three columns of Table 2.12, show that the  $k$ -means initialization works properly. Indeed this strategy significantly reduces the number of iterations required for convergence, and hence the computational time. In particular we report, along with the number of samples in which the M-T-EM algorithm with  $k$ -means initialization reaches the global maximum, the average number of iterations required by the two initialization strategies to converge.

We notice that apart from some cases with many latent components, the global maximum is almost always reached by the M-T-EM algorithm when initialized with the  $k$ -means approach. As for the decrease in the number of iterations, the advantage is particularly evident when dealing with HM model with continuous responses; in this case, it is dropped up to one sixth.

| Scenario                         | Perc. > EM<br>( $k$ -means)   | Perc. Glob. Max.<br>( $k$ -means)   | Iterations<br>(Random) | Iterations<br>( $k$ -means) |
|----------------------------------|---|---|------------------------|-----------------------------|
| LC model                         |   |   |                        |                             |
| A                                | 60%    | 16%    | 359.44                 | 216.00                      |
| B                                | 60%    | 22%    | 136.21                 | 121.27                      |
| C                                | 38%    | 0%     | -                      | -                           |
| D                                | 60%    | 26%    | 112.04                 | 99.04                       |
| E                                | 78%    | 0%     | -                      | -                           |
| HM model (categorical responses) |   |   |                        |                             |
| A                                | 86%    | 40%    | 148.55                 | 140.29                      |
| B                                | 76%    | 40%    | 134.40                 | 121.61                      |
| C                                | 82%    | 32%    | 122.13                 | 110.48                      |
| D                                | 42%   | 30%   | 147.47                 | 132.06                      |
| E                                | 70%  | 32%  | 116.50                 | 106.26                      |
| F                                | 78%  | 18%  | 263.00                 | 253.79                      |
| HM model (continuous responses)  |   |   |                        |                             |
| A                                | 44%  | 44%  | 142.33                 | 142.33                      |
| B                                | 70%  | 56%  | 117.07                 | 97.32                       |
| C                                | 62%  | 50%  | 141.78                 | 136.68                      |
| D                                | 68%  | 44%  | 116.41                 | 94.09                       |
| E                                | 62%  | 26%  | 136.21                 | 89.54                       |

**Table 2.13:** Percentage of samples in which the global maximum is reached by the M-T-EM algorithm with  $k$ -means initialization, but not by the standard EM algorithm with the same starting values, percentage of samples in which the M-T-EM algorithm with  $k$ -means initialization reaches the global maximum and number of iterations until convergence with random and  $k$ -means (or  $k$ -modes) initialization when the latent structure of the models is not correctly specified

In the case of models where the latent structure is not correctly specified, the situation is less well defined: likewise the previous case, the results obtained comparing EM and M-T-EM algorithms initialized with  $k$ -means strategy are very similar for some samples (second column in Table 2.13), highlighting that the standard EM algorithm may sometimes benefit



from the adoption of this initialization strategy. However, when M-T-EM is employed, this improvement does not always correspond to an advantage of  $k$ -means initialization with respect to the random one. As shown in Table 2.13, the number of samples that benefit from this initialization strategy is quite limited and usually does not reach the 50%. Finally, also in this case the  $k$ -means initialization provides some benefits from the point of view of the number of iterations until convergence, even if less pronounced than in the case of models with correctly specified latent structures.

## 2.7 Applications

To explore the performance of the T-EM algorithm when dealing with real-world cases, we apply it to cross-sectional and longitudinal data; we specifically address the problem of selecting the best number of components for LC and HM models.

### 2.7.1 Evaluation of anxiety and depression

We consider data derived from the administration of 14 ordinal items measuring anxiety and depression in a sample of 201 oncological Italian patients (Zigmond and Snaith, 1983). Items are measured according to four response categories ranging from 0 to 3 and corresponding to the lowest and to the highest level of anxiety or depression, respectively. Data are available in the R package `MultiLCIRT` (Bartolucci et al., 2014).

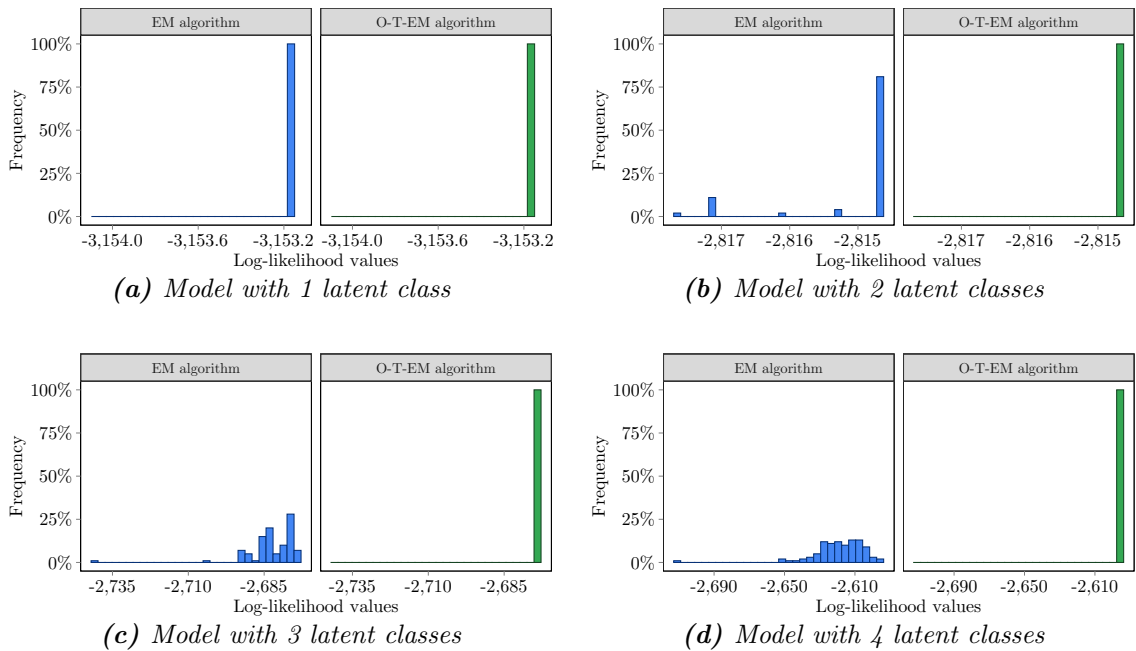
The LC model allows to discover subpopulations of patients with similar intensity levels of these two pathologies. The model is estimated with both EM and O-T-EM algorithms with a number of latent components  $k$  ranging from 1 to 4 to perform model selection. Bayesian Information Criterion ( $BIC$ , Schwarz, 1978) is employed at this purpose penalizing the maximized log-likelihood function for the model complexity. For the O-T-EM algorithm the following configuration of tempering constants is used and held fixed over the values of  $k$ :  $\rho = 90$ ,  $\tau_0 = 10$ ,  $\beta = 20$ , and  $\alpha = 0.8$ .

Results of the estimation of LC model with EM and O-T-EM algorithms are reported in Table 2.14, where it can be seen that both algorithms are able to detect the global maximum for every number of latent classes. The resulting optimal number of components, corresponding to the minimum value of  $BIC$ , is three. It is important to remark that the results are always obtained using the same configuration of tempering constants as presented above. Therefore, we highlight the considerable level of flexibility of the method again.

In this case, the main advantage resulting from the employment of the tempered algo-

| $k$ | #par | EM           |                 | O-T-EM       |                 |
|-----|------|--------------|-----------------|--------------|-----------------|
|     |      | $\hat{\ell}$ | BIC             | $\hat{\ell}$ | BIC             |
| 1   | 42   | -3,153.15    | 6,529.04        | -3,153.15    | 6,529.04        |
| 2   | 85   | -2,814.64    | 6,080.05        | -2,814.64    | 6,080.05        |
| 3   | 128  | -2,674.48    | <b>6,027.79</b> | -2,674.48    | <b>6,027.79</b> |
| 4   | 171  | -2,595.47    | 6,097.83        | -2,595.47    | 6,097.83        |

**Table 2.14:** Number of parameters, maximum log-likelihood and BIC index resulting from fitting a latent class model on the anxiety and depression data with EM and O-T-EM algorithms for different numbers of the latent classes  $k$ . For both algorithms, bold value represents the best result



**Figure 2.13:** Maximized log-likelihood values for the anxiety and depression data using standard EM (in blue) and O-T-EM (in green) algorithms. Four different choices for the number of latent classes are analyzed, with 100 random starting values each

rihm is the frequency of convergence to the global maximum. Figure 2.13 refers to the maximum log-likelihood values reached by each algorithm for each model; as it is evident, while the EM algorithm spreads out over a wide range of values, the O-T-EM algorithm always converge to a single value appearing as the global mode. The same results are obtained also with the M-T-EM for different configurations of tempering constants.

### 2.7.2 Discovering criminal trajectories

We consider longitudinal data on conviction histories of a cohort of  $n = 10,000$  offenders followed from the age of criminal responsibility (10 years) until age 40. As described in [Research Development and Statistics Directorate \(1998\)](#), offenses are grouped into the following 10 typologies: violence against the person, sexual offenses, burglary, robbery, theft and handling stolen goods, fraud and forgery, criminal damage, drug offenses, motoring offenses, and other offenses. Binary response variables ( $r = 10$ ) indicate if the offender has committed a crime during six age bands ( $T = 6$ ) of length equal to 5 years. An HM model was proposed for the analysis of these data in [Bartolucci et al. \(2007\)](#) and [Pennoni \(2014\)](#) to identify typologies of criminal behaviors and types of criminal career specialization over time. Data are available in the R package `LMest` ([Bartolucci et al., 2017](#)).

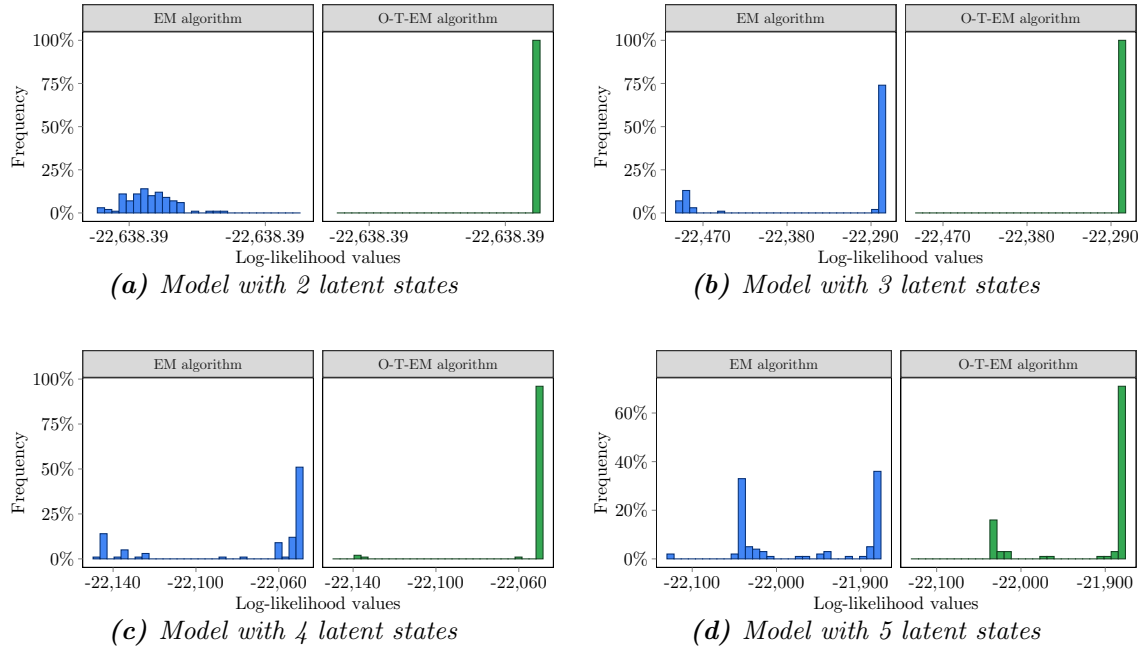
Results of estimating a time heterogeneous HM model with the EM and O-T-EM algorithms for a number of latent states ranging from 1 to 5 are reported in Table 2.15. The optimal number of latent states corresponding to the minimum value of BIC is four. As for the tempered algorithm, the same configuration of tempering constants is kept fixed for all values of  $k$ :  $\alpha = 0.2$ ,  $\beta = 10$ ,  $\rho = 5$  and  $T_0 = 10$ .

| $k$ | #par | EM           |                  | O-T-EM       |                  |
|-----|------|--------------|------------------|--------------|------------------|
|     |      | $\hat{\ell}$ | BIC              | $\hat{\ell}$ | BIC              |
| 1   | 10   | -27,936.35   | 55,964.81        | -27,936.35   | 55,964.81        |
| 2   | 31   | -22,638.39   | 45,562.30        | -22,638.39   | 45,562.30        |
| 3   | 62   | -22,275.05   | 45,121.14        | -22,275.05   | 45,121.14        |
| 4   | 103  | -22,051.55   | <b>45,051.77</b> | -22,051.55   | <b>45,051.77</b> |
| 5   | 154  | -21,881.36   | 45,181.12        | -21,881.36   | 45,181.12        |

**Table 2.15:** Number of parameters, maximum log-likelihood and BIC index resulting from fitting a time heterogeneous hidden Markov model on the criminal data with EM and O-T-EM algorithms for different numbers of the latent states  $k$ . For both algorithms, bold value represents the best result

Results in Figure 2.14 show the comparison of maximum log-likelihood values reached by each algorithm for each model: according to the same procedure illustrated in Section 2.5.2, for each value of  $k$ , 100 different starting values are randomly chosen to initialize both versions of the algorithm. It is clear that, also in this context, the O-T-EM guarantees a better performance. Employing this algorithm the global maximum is reached with a very

high frequency in all cases, and in particular for the selected model with  $k = 4$  latent states. Conversely, the standard EM algorithm converges to a broad range of different maxima.



**Figure 2.14:** Maximized log-likelihood values for the criminal data using standard EM (in blue) and O-T-EM (in green) algorithms. Four different choices for the number of latent states are analyzed, with 100 random starting values each

More specifically, with the proposed algorithm, the frequency of global maximum is higher: the M-T-EM algorithm reaches the global mode 96 times, while the standard EM algorithm only 63. Moreover, the mean distance from the global optimum decreases to almost zero, and the mean of log-likelihood values increases accordingly; only the median value remains essentially unchanged, with just a very slight enhancement.

### 2.7.3 Analyzing countries development

We consider data obtained from the World Bank’s World Development Indicators ([The World Bank Group, 2018](#)) on  $n = 175$  countries collected for  $T = 5$  years (from 2011 to 2015) on  $r = 6$  continuous response variables: life expectancy at birth, total population between the ages 0 to 14, percentage of population with access to electricity, percentage of population using the internet, share of electricity generated by renewable power plants, and fertility rate. A logit transformation was applied to the variables expressed in a percentage

scale, and a Box-Cox transformation (Box and Cox, 1964) to all the variables. In order to check the assumption on the conditional distribution we check the posterior density of each response variable once the units are allocated according to maximum a posteriori rule; results (available from the authors upon request) seem satisfactory. Results of the estimation of a time heterogeneous HM model on the transformed data with EM and O-T-EM algorithms for a number of states ranging from 1 to 10 are reported in Table 2.16. In

| $k$ | #par | EM           |                  | O-T-EM       |                  |
|-----|------|--------------|------------------|--------------|------------------|
|     |      | $\hat{\ell}$ | BIC              | $\hat{\ell}$ | BIC              |
| 1   | 42   | -18,100.06   | 36,339.58        | -18,100.06   | 36,339.58        |
| 2   | 57   | -17,299.80   | 34,816.53        | -17,299.80   | 34,816.53        |
| 3   | 80   | -16,891.00   | 34,117.72        | -16,887.96   | 34,111.63        |
| 4   | 111  | -16,386.89   | 33,269.60        | -16,386.89   | 33,269.60        |
| 5   | 150  | -16,161.01   | 33,019.26        | -16,161.01   | 33,019.26        |
| 6   | 197  | -16,006.90   | 32,953.79        | -16,002.67   | 32,945.33        |
| 7   | 252  | -15,859.53   | 32,943.11        | -15,821.86   | <b>32,867.78</b> |
| 8   | 315  | -15,692.55   | <b>32,934.54</b> | -15,676.37   | 32,902.18        |
| 9   | 386  | -15,569.32   | 33,054.78        | -15,531.69   | 32,979.51        |
| 10  | 465  | -15,459.35   | 33,242.85        | -15,428.07   | 33,180.30        |

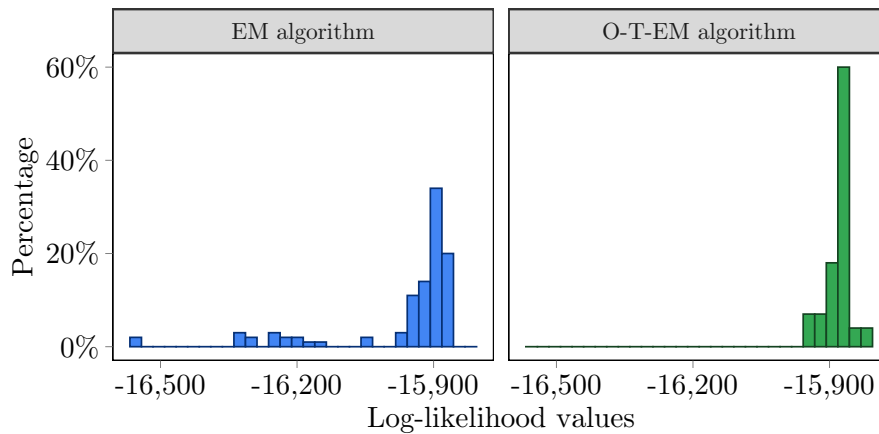
**Table 2.16:** Number of parameters, maximum log-likelihood and BIC index resulting from fitting a time heterogeneous hidden Markov model on the countries' economic conditions data with EM and O-T-EM algorithms for different numbers of latent states  $k$ . For both algorithms, bold values represent the best results

this case, the advantages of using the tempering approach are even more evident:

1. it guarantees convergence to the global maximum. Indeed, for most values of  $k$ , the maximized log-likelihood value is higher than that of the EM algorithm, showing that the standard EM algorithm cannot correctly detect such a value. Moreover, the mean distance from the global maximum also shows significant improvements, assuming much smaller values when the O-T-EM algorithm is used, thus showing that it is able to converge repeatedly to the global maximum.
2. it allows us to select a more parsimonious model. Model selection performed with the standard EM leads us to choose eight components, whereas the T-EM algorithm suggests seven components. BIC values are always smaller than those obtained with standard algorithm.

3. it exhibits an appealing level of flexibility: there is no need to change the optimal set of tempering constants (fixed at  $\alpha = 0.6$ ,  $\beta = 110$ ,  $\rho = 5$ , and  $\tau_0 = 20$ ) once the HM model is fitted for a number of states ranging from 5 to 10. For values of  $k$  from 2 to 4, another unique configuration of tempering constants proves to be the best ( $\alpha = 0.5$ ,  $\beta = 120$ ,  $\rho = 5$ , and  $\tau_0 = 10$ ).

Focusing on the log-likelihood values shown in Figure 2.15 related to the selected model with seven states, we notice that the O-T-EM algorithm always avoids lower values in favor of the higher ones of the maximized log-likelihood. These are reached much more frequently with respect to the EM algorithm.



**Figure 2.15:** Maximized log-likelihood values for the countries’ economic conditions data using standard EM (left, in blue) and O-T-EM (right, in green) algorithms with  $k = 7$  latent states, using 100 random starting values

As already illustrated with the simulation study presented in Section 2.7 and also shown in Table 2.24 of Appendix D, the O-T-EM algorithm is more demanding in terms of computational time with respect to the EM algorithm; however, its performance is superior. Moreover, we notice that on average, a single execution of the T-EM algorithm requires the same time of approximately 10 runs of the standard algorithm. It is important to note that after 1,000 executions performed with 1,000 different random starting values, the EM algorithm is still unable to detect the global maximum (according to the definition provided by the first criterion in Section 2.6) obtained with the O-T-EM algorithm, and equal to  $-15,821.86$ , since its highest reached value is  $-15,834.97$ . Neither a higher number of random starting values (up to 10,000 in our study), nor the  $k$ -means initialization strategy allows us to improve its performance.

## 2.8 Conclusions

The likelihood of DLV models is typically multimodal, and convergence to a point that it is not the global maximum is a severe limitation of all the algorithms employed for maximum likelihood estimation of the model parameters. To reduce the chance of local maxima at convergence when the Expectation-Maximization (EM) algorithm is employed, the model parameters are typically initialized with a multiple-try strategy, employing deterministic and random values to initialize the model parameters. Then, maximum likelihood estimate of the parameters corresponds to the highest log-likelihood at convergence of the algorithm.

In this Chapter, a new robust estimation algorithm based on annealing and tempering techniques is proposed in this context. The underlying idea of the tempered EM (T-EM) algorithm is flattening the target function and then gradually warping it back towards the original one. The ability of the algorithm to remain close enough to the dominant maximum is related to the slowness and the graduation of the warping process, which, in turn, is controlled by a sequence of parameters known as the temperature or tempering profile. Two main classes of such profiles usable with many models to be estimated are tested and compared: a monotonically decreasing exponential profile, easy to tune, and an oscillating profile, having more parameters to tune and ensuring the best performances with a very high level of flexibility.

An accurate Monte Carlo simulation study is carried out considering two general classes of DLV models: latent class and hidden Markov models. We compare the performance of the standard EM algorithm with the proposed ones. This comparison is carried out by evaluating the ability to reach the global maximum and the computational time. From the results of the simulation study and those of the applications we show that the proposed algorithms outperform the standard EM, increasing significantly the chance to of reaching the global maximum in the overwhelming majority of cases. In particular, when an optimally tuned tempering profile is employed, the improvement with respect to the EM algorithm is remarkable: the T-EM algorithm can reach the global mode with a high frequency, generally escaping all local sub-optimal maxima. We detect that the variant with the oscillating profile shows the best performance, slightly outperforming also the monotonic version in most cases.

Estimating the models with the proposed algorithms on categorical and continuous data, having a cross-sectional or longitudinal structure, we also show their good performance in choosing the proper number of latent components. According to the results obtained for the LC and HM models we argue that the proposal may be especially useful for the estimation of the model parameters with complex data structures involving the inclusion of covariates,

missing values, and drop-out.

An additional appealing feature of the proposal is the high level of flexibility of the tempering profiles: once a grid-search procedure is employed to set the tempering constants, these constants remain valid also when data with similar characteristics are used to estimate the model parameters. Moreover, a broad range of values generally performs optimally in many different applied contexts.

Future works may consider the relevant issue of finding a new family of tempering profiles that combines the excellent performance of the oscillating profile with the simple tuning procedure and the fast execution time of the monotonic profile. Other relevant research directions include the exploration of the T-EM algorithm in connection with other maximization algorithms; the most natural choice in this regard is to apply a tempering approach to a direct maximization algorithm, such as Newton-Raphson. The algorithm would also benefit from a more efficient implementation, through the C++ language in order to reduce the computation time. Finally, another possible research line, explored in the next Chapter, would be to incorporate the EM algorithm in the context of evolutionary algorithms ([Ashlock, 2004](#)).



## Appendices

### A Characteristics of the simulated scenarios

Tables 2.17, 2.18, and 2.19 summarize the specific values used to simulate data for the estimation of the LC model, HM model with categorical responses, and HM with continuous responses presented in Section 2.5.1. The following parameters are considered:

- weights of the latent classes (for the LC model) and initial probabilities of the latent states (for the HM models) are defined in such a way that each latent component has the same probability:  $\pi_u = 1/k, \forall u = 1, \dots, k$ ;
- transition probabilities of the HM models are defined to favor persistence in each state; in particular, for  $k = 3$  the transition matrix is defined as follows:

$$\begin{bmatrix} 0.800 & 0.150 & 0.050 \\ 0.100 & 0.800 & 0.100 \\ 0.050 & 0.150 & 0.800 \end{bmatrix};$$

- conditional response probabilities are kept fixed considering scenario A (see Tables 2.17, 2.18, and 2.19); for each response variable we define the corresponding matrix as follows:

$$\begin{bmatrix} 0.800 & 0.100 & 0.050 \\ 0.150 & 0.800 & 0.150 \\ 0.050 & 0.100 & 0.800 \end{bmatrix};$$

- for the HM model with continuous response variables the same conditional distribution holds for all response variables; for example, with  $k = 3$  latent states, the mean vector  $\boldsymbol{\mu} = [-2, 0, 2]'$  is fixed for each response variable;
- the variance-covariance matrix  $\boldsymbol{\Sigma}$  is computed as the sample covariance matrix of the data.

| Scenario | $n$   | $r$ | $c$ | $k$ |
|----------|-------|-----|-----|-----|
| A        | 500   | 6   | 3   | 3   |
| B        | 1,000 | 6   | 3   | 3   |
| C        | 500   | 12  | 3   | 3   |
| D        | 500   | 6   | 6   | 3   |
| E        | 500   | 6   | 3   | 6   |

**Table 2.17:** Description of the simulated scenarios for the latent class model: sample size ( $n$ ), response variables ( $r$ ), categories ( $c$ ), and latent classes ( $k$ )

| Scenario | $n$   | $r$ | $c$ | $T$ | $k$ |
|----------|-------|-----|-----|-----|-----|
| A        | 500   | 6   | 3   | 5   | 3   |
| B        | 1,000 | 6   | 3   | 5   | 3   |
| C        | 500   | 12  | 3   | 5   | 3   |
| D        | 500   | 6   | 6   | 5   | 3   |
| E        | 500   | 6   | 3   | 10  | 3   |
| F        | 500   | 6   | 3   | 5   | 6   |

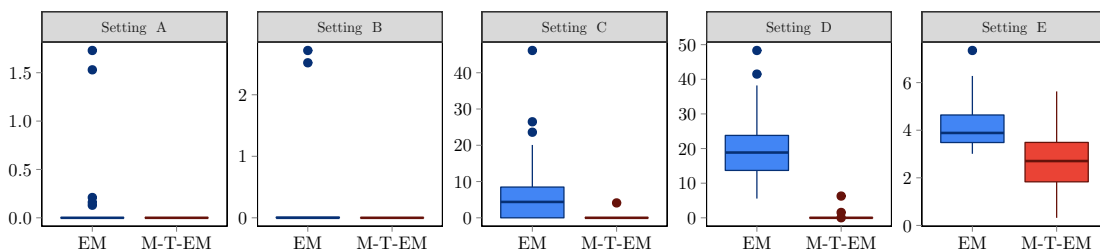
**Table 2.18:** Description of the simulated scenarios for the hidden Markov model with categorical response variables: sample size ( $n$ ), number of response variables ( $r$ ), categories ( $c$ ), time occasions ( $T$ ), and latent states ( $k$ )

| Scenario | $n$   | $r$ | $T$ | $k$ |
|----------|-------|-----|-----|-----|
| A        | 500   | 6   | 5   | 3   |
| B        | 1,000 | 6   | 5   | 3   |
| C        | 500   | 12  | 5   | 3   |
| D        | 500   | 6   | 10  | 3   |
| E        | 500   | 6   | 5   | 6   |

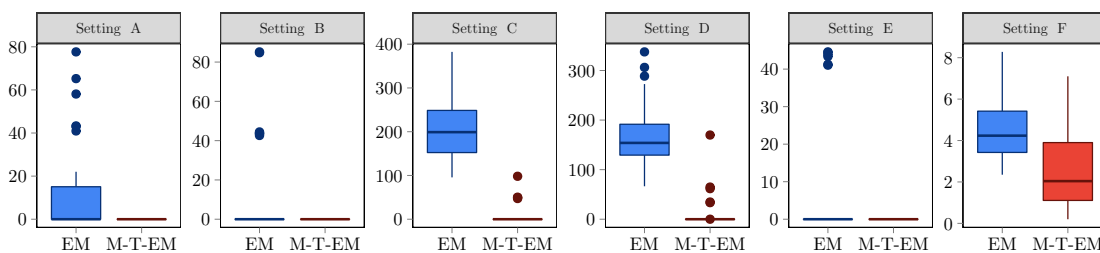
**Table 2.19:** Description of the simulated scenarios for the hidden Markov model with continuous response variables: sample size ( $n$ ), number of response variables ( $r$ ), time occasions ( $T$ ), and latent states ( $k$ )

## B Additional simulation results

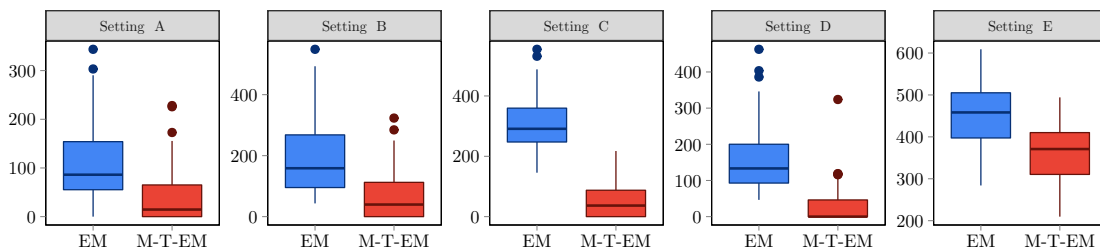
In this Section we report additional details on the results of the simulation study in Section 2.5.2. In particular, for each considered scenario (see Tables 2.17, 2.18, and 2.19), Figures 2.16 and 2.17 show the distribution of the mean distance from the global maximum through boxplots.



(a) Latent class model with correctly specified latent structure

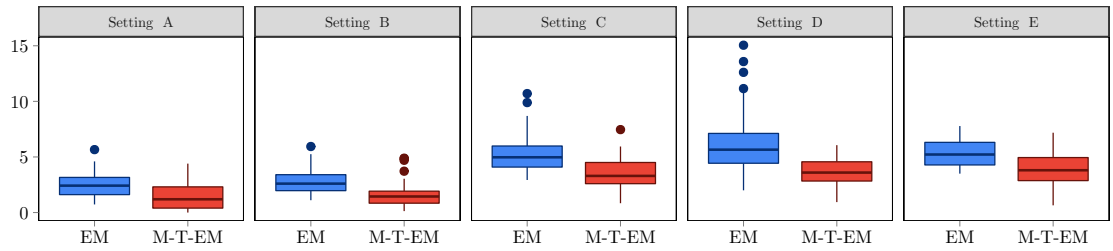


(b) Hidden Markov model with categorical response variables and correctly specified latent structure

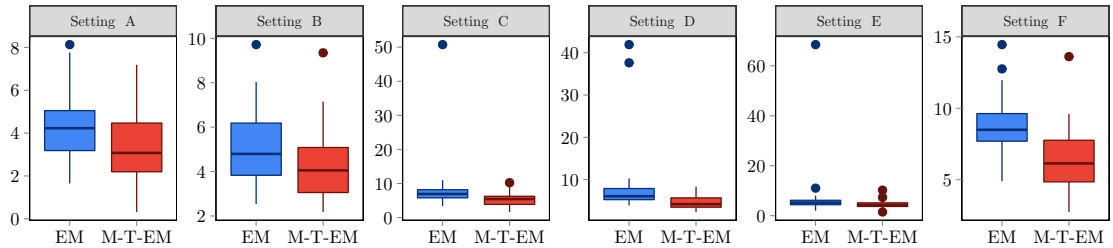


(c) Hidden Markov model with continuous response variables and correctly specified latent structure

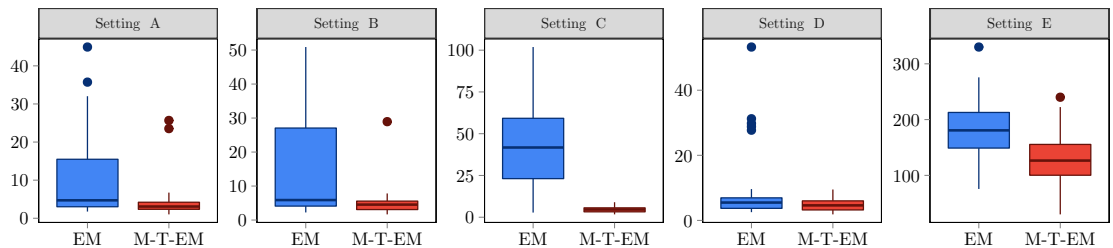
**Figure 2.16:** Mean distance from the global maximum using EM and M-T-EM algorithms under simulated scenarios presented in Tables 2.17, 2.18, and 2.19 of the Appendix A for the latent class and hidden Markov models with correctly specified latent structure



(a) Latent class model with misspecified latent structure



(b) Hidden Markov model with categorical response variables and misspecified latent structure



















(c) Hidden Markov model with continuous response variables and misspecified latent structure

















**Figure 2.17:** Mean distance from the global maximum using EM and M-T-EM algorithms under simulated scenarios presented in Tables 2.17, 2.18, and 2.19 of the Appendix A for the latent class and hidden Markov models with misspecified latent structure

### C Simulation results with fixed tempering profiles

In this Section we present results obtained from the simulation studies comparing the EM algorithm with the T-EM algorithm with fixed tempering profiles. Table 2.20 reports the results for LC model, HM model with categorical response variables and HM model with continuous response variables model (according to simulated scenarios presented in Tables 2.17, 2.18, and 2.19 in the Appendix A) when the latent structure is correctly specified; Table 2.21 considers the same models when the latent structure is not correctly specified. The analysis carried out on the basis of the results is reported in Section 2.5.5.

| Scenario                                 | Tempering profile |         | Percentage   |
|--|-------------------|---------|--|
| Correctly specified LC model             |                   |         |  |
|  | $\alpha$          | $\beta$ |  |
| A  | 6                 | 0.7     | 58%     |
| B  | 6                 | 0.6     | 62%     |
| C  | 1                 | 0.7     | 78%     |
| D  | 1                 | 1.8     | 72%     |
| E  | 1                 | 1.5     | 64%     |
| Correctly specified categorical HM model |                   |         |  |
|  | $\alpha$          | $\beta$ |  |
| A  | 1                 | 0.7     | 96%   |
| B  | 1                 | 0.6     | 100%  |
| C  | 1                 | 1.9     | 74%   |
| D  | 3                 | 1.8     | 70%   |
| E  | 1                 | 0.6     | 100%  |
| F  | 14                | 1.1     | 64%   |
| Correctly specified continuous HM model  |                   |         |  |
|  | $\alpha$          | $\beta$ |  |
| A  | 1                 | 1.3     | 92%   |
| B  | 1                 | 1.1     | 100%  |
| C  | 2                 | 0.3     | 96%   |
| D  | 2                 | 0.2     | 98%   |
| E  | 1                 | 1.1     | 100%  |

**Table 2.20:** Performance of the M-T-EM algorithm for latent class and hidden Markov models when the latent structure is correctly specified, using fixed configurations of tempering constants  $\alpha$  and  $\beta$ . The last column shows the percentage of samples for which the M-T-EM algorithm outperforms the EM algorithm

| Scenario                          | Tempering profile |         | Percentage  |
|-----------------------------------|-------------------|---------|---|
| Misspecified LC model             |                   |         |   |
|                                   | $\alpha$          | $\beta$ |   |
| A                                 | 2                 | 0.6     | 50%    |
| B                                 | 2                 | 0.6     | 52%    |
| C                                 | 2                 | 0.6     | 44%    |
| D                                 | 2                 | 0.6     | 58%    |
| E                                 | 2                 | 0.6     | 62%    |
| Misspecified categorical HM model |                   |         |   |
|                                   | $\alpha$          | $\beta$ |   |
| A                                 | 5                 | 2.0     | 60%    |
| B                                 | 5                 | 1.9     | 54%    |
| C                                 | 6                 | 0.0     | 66%   |
| D                                 | 3                 | 1.7     | 64%  |
| E                                 | 1                 | 1.9     | 52%  |
| F                                 | 15                | 0.1     | 42%  |
| Misspecified continuous HM model  |                   |         |   |
|                                   | $\alpha$          | $\beta$ |   |
| A                                 | 2                 | 0.4     | 84%  |
| B                                 | 1                 | 0.9     | 92%  |
| C                                 | 1                 | 1.2     | 80%  |
| D                                 | 2                 | 0.2     | 80%  |
| E                                 | 3                 | 0.0     | 86%  |

**Table 2.21:** Performance of the M-T-EM algorithm for latent class and hidden Markov models when the latent structure is not correctly specified, using fixed configurations of tempering constants  $\alpha$  and  $\beta$ . The last column shows the percentage of samples for which the M-T-EM algorithm outperforms the EM algorithm

## D Real data analysis in terms of computational time

In this Section, considering the real data presented and analyzed in Section 2.7 we briefly inspect the performance of the different algorithms in terms of computational time. Results for the LC model, the HM model with categorical response variables and the HM model with continuous response variables are summarized in Tables 2.22, 2.23, and 2.24, respectively. All values are expressed in seconds.

| Algorithm | Minimum | Median | Mean | Maximum |
|-----------|---------|--------|------|---------|
| EM        | 0.03    | 0.09   | 0.15 | 0.78    |
| M-T-EM    | 0.42    | 0.44   | 0.45 | 0.76    |
| O-T-EM    | 1.22    | 1.24   | 1.26 | 1.56    |

**Table 2.22:** Computational times in seconds of the EM and O-T-EM algorithms. The analysis refers to the estimation of the latent class model for the anxiety and depression data using  $k = 3$  latent classes and on the basis of 100 random starting values

| Algorithm | Minimum  | Median   | Mean     | Maximum  |
|-----------|----------|----------|----------|----------|
| EM        | 16.85    | 26.37    | 23.27    | 45.24    |
| M-T-EM    | 152.23   | 287.78   | 299.58   | 529.13   |
| O-T-EM    | 1,952.25 | 2,541.36 | 2,774.72 | 3,247.59 |

**Table 2.23:** Computational times in seconds of the EM and O-T-EM algorithms. The analysis refers to the estimation of the hidden Markov model with categorical response variables for the criminal data using  $k = 4$  latent states and on the basis of 100 random starting values

| Algorithm | Minimum | Median | Mean  | Maximum |
|-----------|---------|--------|-------|---------|
| EM        | 0.56    | 2.08   | 2.33  | 5.13    |
| M-T-EM    | 0.52    | 3.02   | 3.13  | 7.87    |
| O-T-EM    | 2.03    | 22.72  | 28.44 | 105.69  |

**Table 2.24:** Computational times in seconds of the EM and O-T-EM algorithms. The analysis refers to the estimation of the hidden Markov model with continuous response variables for the countries' economic conditions data using  $k = 7$  latent states and on the basis of 100 random starting values

## Bibliography

- AARTS, E. AND KORST, J. (1989). *Simulated Annealing and Boltzmann Machines: A Stochastic Approach to Combinatorial Optimization and Neural Computing*. John Wiley and Sons, New York.
- ASHLOCK, D. (2004). *Evolutionary Computation for Modeling and Optimization*. Springer, New York.
- BARBU, A. AND ZHU, S. (2013). *Monte Carlo Methods*. Springer, Singapore.
- BARTOLUCCI, F., BACCI, S., AND GNALDI, M. (2014). MultiLCIRT: An R package for multidimensional latent class item response models. *Computational Statistics and Data Analysis*, **71**, 971–985.
- BARTOLUCCI, F., FARCOMENI, A., AND PENNONI, F. (2013). *Latent Markov Models for Longitudinal Data*. Chapman & Hall/CRC, Boca Raton.
- BARTOLUCCI, F., PANDOLFI, S., AND PENNONI, F. (2017). LMest: An R package for latent markov models for longitudinal categorical data. *Journal of Statistical Software*, **81**, 1–38.
- BARTOLUCCI, F., PENNONI, F., AND FRANCIS, B. (2007). A latent markov model for detecting patterns of criminal activity. *Journal of the Royal Statistical Society. Series A (Statistics in Society)*, **170**, 114–132.
- BAUM, L., PETRIE, T., SOULES, G., AND WEISS, N. (1970). A maximization technique occurring in the statistical analysis of probabilistic functions of markov chains. *The Annals of Mathematical Statistics*, **41**, 164–171.
- BILLINGS, S. (1994). Simulated annealing for earthquake location. *Geophysical Journal International*, **118**, 680–692.
- BOX, G. AND COX, D. (1964). An analysis of transformations. *J. R. Stat. Soc. Series. B Stat. Methodol.*, **26**, 211–243.
- CHEN, S. AND LUK, B. (1999). Adaptive simulated annealing for optimization in signal processing applications. *Signal Processing*, **79**, 117–128.
- CRAMA, Y. AND SCHYNS, M. (2003). Simulated annealing for complex portfolio selection problems. *European Journal of Operational Research*, **150**, 546–571.



- DEMPSTER, A., LAIRD, N., AND RUBIN, D. (1977). Maximum likelihood from incomplete data via the em algorithm (with discussion). *Journal of the Royal Statistical Society, Series B*, **39**, 1–38.
- EARL, D. J. AND DEEM, M. W. (2005). Parallel Tempering: theory, applications, and new perspectives. *Phys. Chem. Chem. Phys.*, **7**, 3910–3916.
- EVERITT, B. S., LANDAU, S., LEESE, M., AND STAHL, D. (2011). *Cluster Analysis, 5th ed.* New York: John Wiley.
- FALCIONI, M. AND DEEM, M. (1999). A Biased Monte Carlo Scheme for Zeolite Structure Solution. *Journal of Chemical Physics*, **110**, 1754–1766.
- FORGY, E. (1965). Cluster analysis of multivariate data: Efficiency versus interpretability of classifications. *Biometrics*, **21**, 768–780.
- FRALEY, C. AND RAFTERY, A. (2002). Model-based clustering, discriminant analysis, and density estimation. *Journal of the American Statistical Association*, **97**, 611–634.
- GEYER, C. J. (1991). Markov Chain Monte Carlo Maximum Likelihood. In *Computing Science and Statistics, Proceedings of the 23rd Symposium on the Interface*, Computing science and statistics, pages 156–163. Interface Foundation of North America.
- GEYER, C. J. AND THOMPSON, E. A. (1995). Annealing Markov Chain Monte Carlo with Applications to Ancestral Inference. *Journal of the American Statistical Association*, **90**, 909–920.
- GOODMAN, L. (1974). Exploratory latent structure analysis using both identifiable and unidentifiable models. *Biometrika*, **61**, 215–231.
- HASTINGS, W. K. (1970). Monte Carlo Sampling Methods Using Markov Chains and their Application. *Biometrika*, **57**, 97–109.
- HOFMANN, C. J. (1999). Probabilistic latent semantic analysis. In *Proceedings of the Fifteenth conference on Uncertainty in artificial intelligence*, UAI’99, pages 289–296. Morgan Kaufmann Publisher Inc., San Francisco, CA, USA.
- HUANG, Z. (1998). Extensions to the k-means algorithm for clustering large data sets with categorical values. *Data Mining and Knowledge Discovery*, **2**, 283–304.
- KIRKPATRICK, S., GELATT, C., AND VECCHI, M. (1983). Optimization by Simulated Annealing. *Science (New York, N.Y.)*, **220**, 671–680.

- KOULAMAS, C., ANTONY, S., AND JAEN, R. (1994). A survey of simulated annealing applications to operations-research problems. *OMEGA-International Journal of Management Science*, **22**, 41–56.
- LAIRD, N. (1978). Nonparametric maximum likelihood estimation of a mixing distribution. *Journal of the American Statistical Association*, **73**, 805–811.
- LARTIGUE, T., DURRLEMAN, S., AND ALLASSONNIÈRE, S. (2021). Deterministic approximate EM algorithm; application to the riemann approximation EM and the tempered EM. *arXiv:2003.10126*, pages 1–32.
- LAZARSELD, P. AND HENRY, N. (1968). *Latent Structure Analysis*. Houghton Mifflin, Boston.
- LEROUX, B. AND PUTERMAN, M. (1992). Maximum-penalized-likelihood estimation for independent and markov-dependent mixture models. *Biometrics*, **48**, 545–558.
- LINDSAY, B., CLOGG, C., AND GREGO, J. (1991). Semiparametric estimation in the rasch model and related exponential response models, including a simple latent class model for item analysis. *Journal of the American Statistical Association*, **86**, 96–107.
- MACQUEEN, J. (1967). Some methods for classification and analysis of multivariate observations. In *In 5-th Berkeley Symposium on Mathematical Statistics and Probability*, pages 281–297.
- MARUOTTI, A. AND PUNZO, A. (2021). Initialization of hidden markov and semi-hidden markov: A critical evaluation of several strategies. *International Statistical Review*.
- MCLACHLAN, G. AND BASFORD, K. (1988). *Mixture Models: Inference and Applications to Clustering*. Marcel Dekker, New York.
- MCLACHLAN, G. AND KRISHNAN, T. (2008). *The EM Algorithm and Extensions: 2nd Edition*. John Wiley and Sons, Hoboken.
- METROPOLIS, N., ROSENBLUTH, A., ROSENBLUTH, M., TELLER, A.-H., AND TELLER, E. (1953). Equation of state calculations by fast computing machines. *Journal of Chemical Physics*, **21**, 1087–1092.
- PENNONI, F. (2014). *Issues on the Estimation of Latent Variable and Latent Class Models*. Scholar’s Press, Saarbrücken.

- RESEARCH DEVELOPMENT AND STATISTICS DIRECTORATE (1998). The offenders index: codebook. Available from <https://webarchive.nationalarchives.gov.uk/ukgwa/20130128103514/http://homeoffice.gov.uk/rds/pdfs/oicodes.pdf>.
- ROBERT, C., ELVIRA, V., TAWN, N., AND WU, C. (2018). Accelerating mcmc algorithms. *WIREs Computational Statistics*, **10**, 1–14.
- SAMBRIDGE, M. (2014). A parallel tempering algorithm for probabilistic sampling and multimodal optimization. *Geophysical Journal International*, **196**, 357–374.
- SCHWARZ, G. (1978). Estimating the dimension of a model. *Annals of Statistics*, **6**, 461–464.
- SVERGUN, D. (1999). Restoring low resolution structure of biological macromolecules from solution scattering using simulated annealing. *Biophysical Journal*, **76**, 2879–2886.
- TEAM, R. C. (2022). *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, Vienna, Austria.
- THE WORLD BANK GROUP (2018). Data catalog: World development indicators. Available from <https://datacatalog.worldbank.org/dataset/world-development-indicators>.
- TITTERINGTON, D., SMITH, A., AND MAKOV (1985). *Statistical Analysis of Finite Mixture Distributions*. John Wiley and Sons, New York.
- UEDA, N. AND NAKANO, R. (1998). Deterministic annealing EM algorithm. *Neural Netw.*, **11**, 271–282.
- WIGGINS, L. (1973). *Panel Analysis: Latent Probability Models for Attitude and Behaviour Processes*. Elsevier, Amsterdam.
- YUILLE, A., STOLORZ, P., AND UTANS, J. (1994). Statistical Physics, Mixture of Distributions, and the EM Algorithm. *Neural Computation*, **6**, 334–340.
- ZHOU, H. AND LANGE, K. (2010). On the bumpy road to the dominant mode. *Scand. J. Stat.*, **37**, 612–631.
- ZIGMOND, A. AND SNAITH, R. (1983). The hospital anxiety and depression scale. *Acta Psychiatr. Scand.*, **67**, 361–70.



## Evolutionary Expectation-Maximization algorithm for the estimation of discrete latent variable models

---

### 3.1 Introduction

In this Chapter we consider the problem of convergence of the log-likelihood function to local maxima again and explore a different tactic, proposing the use of a quite new technique known as evolutionary algorithms (EAs). Inspired by the basic principles of the Darwinian theory of evolution, this class of computational methods is commonly employed to solve complex optimization problems in an iterative manner. At each step of the procedure, many candidate solutions are considered and evaluated. The best candidates, according to some quality measure, are selected for successive step, while the worst ones are discarded. To favor an adequate exploration of the solution domain, other natural-based operations, such as mutation and crossover, may increase the diversity of the candidates during the process.

Following these principles, we propose an evolutionary version of the EM algorithm able to repeatedly discard local maxima in favor of solutions closer to the global optimum; this approach significantly increases the chance of converging to the global maximum. In continuity with the content of the previous Chapter, we specifically address the case of LC

and HM models; it is anyway straightforward to extend the procedure for the estimation of many other DLV models. The performance of the proposed evolutionary approach is assessed in terms of ability to converge to the global maximum relying on the same simulation studies designed in Chapter 2, so that we can compare the proposal with both the standard and the tempered EM algorithms. The implemented code for the proposal is written for the open source software R and will be made available.

The rest of this Chapter is organized as follows. In Section 3.2 we introduce the theory of evolutionary algorithms, examining its main features in full detail. In Section 3.3 we provide the general formulation of the evolutionary EM (E-EM) algorithm; we deal with the specific settings of the algorithm and propose an efficient model selection approach. In Section 3.4 we analyze the performance of the proposed algorithm through an extensive Monte Carlo simulation study. In Section 3.5 we apply the E-EM algorithm to the real-world data introduced and used in Section 2.6 of the previous Chapter. Section 3.6 concludes the present Chapter with a brief comparison between the T-EM and the E-EM algorithms. Appendices A and B include additional simulation results, while Appendix C provides some technical details about the main evolutionary operators.

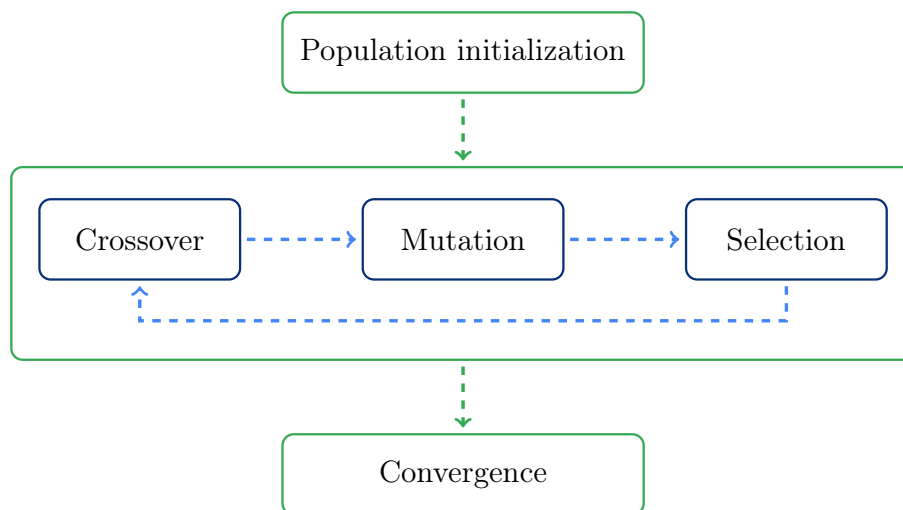
## 3.2 Evolutionary algorithms

In this Section we provide a generic introduction to the theory of evolutionary algorithms. Note that this field is quite vast and active, and just a brief outline is provided here; for more detail see, among the others, [Bäck \(1996\)](#), [Deb \(2001\)](#), and [Ashlock \(2004\)](#) for a comprehensive coverage of the topic.

### 3.2.1 Design of evolutionary algorithms

In the natural context, evolution can be described as the process by which individuals of a population become fitter in a specific environment through mutation, adaptation, natural selection, and selective breeding. Evolutionary algorithms aim at modeling these principles in a computational field operating on “populations” of data structures. In particular, these methods are commonly employed to solve difficult optimization problems, for both continuous and discrete functions. In this context, each population member represents a possible solution of the optimization problem. After creating an initial population, a *fitness function* assigns a numerical estimate of quality to each population member and is used to decide which of them deserve further attention. This procedure is known as *selection* and it aims at favoring data structures with a higher fitness function value. Each selected

population member is then made to evolve through some *variation operators*, mimicking the natural process of evolution. Here we consider the most relevant ones in evolutionary computation literature: *mutation*, which introduces variation by making random changes in data structures, and *crossover*, which produces a new generation of individuals from the previous ones. The entire process is repeated until one or more stopping conditions are met. In the following we will examine in more detail the main components of an EA, while the general scheme of an EA is sketched in Figure 3.1.



*Figure 3.1: General scheme to describe the evolutionary algorithm*

**Initial population** The choice of the initial population is a crucial aspect, having a direct impact on the performance of an EA. In general there exist different ways to specify the initial population: it may be filled in at random, designed to some standard, or be the output of some other algorithm. However, regardless of the initialization procedure, a main requirement is the ability to represent the entire search space of the parameters, increasing diversity; to this aim note that an initial population including only points from a restricted region will typically result into optimal solutions close to that same region, thus wasting the exploration power of the evolutionary algorithm. Another fundamental feature is the choice of the initial population size (generally kept constant in evolutionary algorithms): a small number of population members ensures a fast execution of the algorithm, but limits the representation of the entire search space. On the contrary, a large population helps increasing such a diversity, but results into a higher computational time. A thoughtful

choice needs to balance these two requirements, taking into account the complexity of the problem at issue, the computational cost of the fitness function, and the availability of computational resources.

**Crossover** Also known as recombination, it is an evolutionary operator used to produce a new generation of individuals from the previous ones. Crossover techniques can be classified into three main types:

- asexual crossover, when a single parent is selected to generate an offspring;
- sexual crossover, when two parents are selected and combined in some fashion to produce one or two offsprings;
- multi-recombination, when more than two parents are used to generate one or more offsprings.

Crossover may be either applied to all individuals of a given generation, or performed on a restricted number of individuals, selected according to a probability distribution (a typical choice is to favor crossover in the individuals associated with the worst values of the fitness function). The latter option helps preserving the information of the parents without increasing the computational time of the algorithm.

**Mutation** It is the process by which changes are introduced into the structure of an already existing population member, so as to provide a source of minor variation. Likewise the crossover operator, mutation may either affect the whole population or be mainly used on the least fit individuals in order to encourage exploration. Note that, unlike the biological and natural evolution, in evolutionary algorithms the difference between asexual crossover and mutation, may appear very subtle. We will examine this point in more detail and from a computational perspective in Section 3.3.

**Selection** A major difference between biological evolution and evolutionary algorithms lies in the selection process. Natural selection “multiplies the incidence of beneficial mutations over the generations and eliminates harmful ones”, thus enhancing “the preservation of a group of organisms that are best adjusted to the physical and biological conditions of their environment” ([Encyclopaedia Britannica, 2022](#)). This “selection of the fittest” procedure is not always appropriate in evolutionary algorithms. Consider, for example, a selection scheme which chooses very often the best individuals and very rarely the worst ones; the consequence is a phenomenon known as *strong selective pressure*, which diminishes



the diversity in the population and may lead to a premature convergence of the algorithm (Wieczorek and Czech, 2002). The problem of loss of population diversity has garnered increasing attention in the computer science literature, but is beyond the scope of this work; see, for example, Goldberg and Deb (1991), Back (1994), and Blickle and Thiele (1995). To prevent this behavior, it is important to balance between *exploitation* of the individuals with the best values of the fittest function and *exploration* of population member associated with the worst values of the fitness function. This results into a broad variety of selection methods, ranging from elitism (*i.e.*, selection of the best percent of the current population, representing pure exploitation) to random selection (representing pure exploration). For a thorough collection of selection methods see, among the others, Ashlock (2004, Chapter 2).

### 3.2.2 Previous works

The evolutionary paradigm is successfully applied to clustering. In this context several criteria used for assessing partitions can serve as fitness function, and the role of evolutionary algorithms is mainly limited to minimizing some notion of distance between points and clusters. Hruschka et al. (2009) present a thorough overview of this subject, providing a large number of references, and describing applications of evolutionary algorithms for clustering in different domains.

In the field of model-based clustering Pernkopf and Bouchaffra (2005) introduce an EA for Gaussian mixture models, focusing on the evolution of the parameter space: the initial population is made up of different copies of the model parameters, crossover and mutation aims at evolving these values, and selection extracts a new population of model parameters. Andrews and McNicholas (2013) and McNicholas et al. (2021) address the same problem following a slightly different approach and considering evolution among cluster membership labels. Kampo (2021) extends the approach to the Gaussian mixture model with missing values, and to the family of Gaussian parsimonious clustering models. Up to our best knowledge no applications of EA for LC, HM, or other DLV models have been proposed in statistical literature.

## 3.3 Evolutionary Expectation-Maximization algorithm

In this Section we provide details on the proposed E-EM algorithm; its basic idea is to utilize the most appealing features of both algorithms, namely, the well-known convergence properties towards a maximum of the log-likelihood function characteristic of the EM algorithm, and the ability to accurately explore the parameter space typical of the evolutionary

framework. In addition, we also exploit the fact that, although the EM algorithm typically has a slow convergence, the first few steps considerably increase the value of the log-likelihood function.

The proposed E-EM algorithm defines a population in which each “individual” represents a possible solution of the model estimation. In particular, we associate with each individual the array containing the initial posterior probabilities of the latent states for each response configuration (and for each time occasion, in the case of longitudinal data). This choice is similar to [McNicholas et al. \(2021\)](#), which directly focuses on evolving the cluster membership labels. In literature, a possible alternative is sometimes defined, considering evolution of the parameter space (see *e.g.* [Pernkopf and Bouchaffra, 2005](#)) and performing crossover and mutation on the model parameters. The latter approach seems to involve some drawbacks, especially when categorical response variable are taken into account. See Appendix C for a quick theoretical comparison between the two approaches.

In the following we discuss in more detail the feature of the proposed E-EM algorithm. Hereafter we will denote the evolving population by letter  $P$ . Moreover we let  $n_p$  and  $n_c$  denote the number of parent and offspring individuals respectively. At the beginning of each step of the E-EM algorithm the current population  $P_0$  always consists of  $n_p$  parents. After being initialized, the algorithm alternates the following operations until convergence:

1. **Update.** Each individual from population  $P_0$  is updated by performing a small number  $R$  of cycles of the EM algorithm; the resulting updated population is denoted by  $P_1$ . The value of  $R$  should be kept sufficiently small so as not to increase the computational time. In the following the choice  $R = 20$  will be considered for both LC and HM models. Moreover, convergence is checked on the basis of the relative change in the log-likelihood of two consecutive steps; if this condition is fulfilled, the EM algorithm is interrupted without performing the remaining cycles.
2. **Crossover.** The crossover operator recombines individuals from population  $P_1$  to obtain the  $n_c$  offspring of new population  $P_2$ . More specifically, two parents are randomly selected among the individuals of population  $P_1$ , the same row is randomly chosen from the two corresponding arrays, which are swapped from that line to the end. This operator is usually known as single-point crossover ([Bäck, 1996](#); [Michalewicz and Fogel, 2000](#)). For the HM model this operation is repeated for every time occasion independently. Note that the same pair of parents could be selected multiple times; this surely happens if  $n_p(n_p - 1)/2 < n_c$ .
3. **Update.** Each offspring individual from population  $P_2$  is updated through  $R$  steps of

the EM algorithm, generating the updated population  $P_3$ . This operation is identical to the previous update performed at point 1.

4. **Selection.** The selection operator pertains to individuals from both the parent population  $P_1$  and the offspring population  $P_3$  collectively. The complete data log-likelihood is employed as a fitness function, so that fittest individuals are those with the highest log-likelihood value. Refer to Chapter 1 for the expression of the log-likelihood function for each specific model. The selection strategy is inspired from [Back and Schwefel \(1996\)](#): individuals from populations  $P_1$  and  $P_3$  are considered jointly and the  $n_p$  with the highest fitness value are selected for the next generation  $P_4$ . This elitist approach allows us to preserve the monotonic convergence of the EM algorithm.
5. **Mutation.** Differently from the crossover operator, mutation introduces variation working on a single individual at a time. More specifically, given a row of the corresponding array, mutation operator swaps the highest value with a random one. In other words, we are changing the latent component to which a subject is assigned. Each row is selected with a certain probability  $p_M$ , equal to 0.02 in the following. To preserve the elitism of the algorithm, the best individual of the current population is always copied unaltered to the next generation (denoted by  $P_5$ ).

After convergence of the E-EM algorithm, the best individual from population  $P_4$  is selected and updated one last time through a complete run of the EM algorithm until convergence is reached. The resulting E-EM algorithm is a very flexible procedure, in which the specificities of each model are limited to the few steps of the EM algorithm. It is valid for both the LC and the HM models, and it can be applied with minimal effort to other DLV models. The pseudo-code of the algorithm is presented in Algorithm 4.

### 3.3.1 Initialization and convergence criterion

Given the evolutionary framework of the algorithm and the related need to represent the entire search space, a random initialization of the population represents the most appropriate choice. To this aim, model parameters are randomly drawn as described in Section 2.5.1, and used to compute the array of the estimated posterior probabilities. The process is repeated for each of the  $n_p$  individuals of the initial population. The use of more refined initialization strategies, such as  $k$ -means or  $k$ -modes algorithms, does not show any significant benefit and is therefore discarded.

**Algorithm 4** General scheme of the evolutionary Expectation-Maximization algorithm

- 
- 1: Initialize  $n_p$ ,  $n_c$ , and  $R$ .
  - 2: Initialize  $P_0$ .
  - 3: **while** (Convergence Condition = FALSE) **do**
  - 4:  $P_1 \leftarrow \text{Update}(P_0)$ : run  $R$  steps of the EM algorithm;
  - 5:  $P_2 \leftarrow \text{Crossover}(P_1)$ ;
  - 6:  $P_3 \leftarrow \text{Update}(P_2)$ : run  $R$  steps of the EM algorithm;
  - 7:  $P_4 \leftarrow \text{Select}(P_1 \cup P_3)$ ;
  - 8:  $P_5 \leftarrow \text{Mutate}(P_4)$ ;
  - 9:  $P_0 \leftarrow P_5$ .
  - 10: **end while**
  - 11: Select the best result from population  $P_4$  and run the EM algorithm until convergence.
- 

Another fundamental aspect is how to check for the algorithm convergence. Here we adapt a simple criterion commonly used for the EM algorithm: at each step of the E-EM algorithm, before mutation is applied, the best solution is selected and its log-likelihood is computed. The algorithm is stopped when the relative change in the log-likelihood of two consecutive steps is smaller than a suitable tolerance level  $\varepsilon$  (equal to  $10^{-8}$  in the following; see also Section 2.5.1). More restrictive criteria exist in the evolutionary literature, considering for example the lack of progress of the top  $n_p$  solutions over a certain number of generations (Pernkopf and Bouchaffra, 2005; Andrews and McNicholas, 2013), or the absence of change in the mean fitness value. These approaches are more time demanding, without showing any significant benefit in our case, and are not taken into consideration.

### 3.3.2 Model selection

The flexibility of the proposed E-EM algorithm allows us to directly incorporate a selection process for the optimal number of latent components  $k$ . The structure of the algorithm remains mainly unaltered, the only necessary change being in the nature of the fitness function. A simple approach is to define some model selection criterion, such as AIC or BIC (see Section 1.1.3), and use this as fitness function instead of the log-likelihood. A minor adjustment regards the initial population, which has to include a suitable number of individuals for each value of  $k$ . In general, higher values of  $k$  require a more accurate exploration of the parameter space to converge to the global maximum, and hence a higher number of individuals in the initial population. On the contrary, if the number of solutions for high values of  $k$  is overwhelming, the risk is that individuals related to small values of  $k$  may be dropped prematurely. In this sense a control may be added to force the algorithm to carry on at least one solution for each value of  $k$ . The resulting procedure is able to evolve

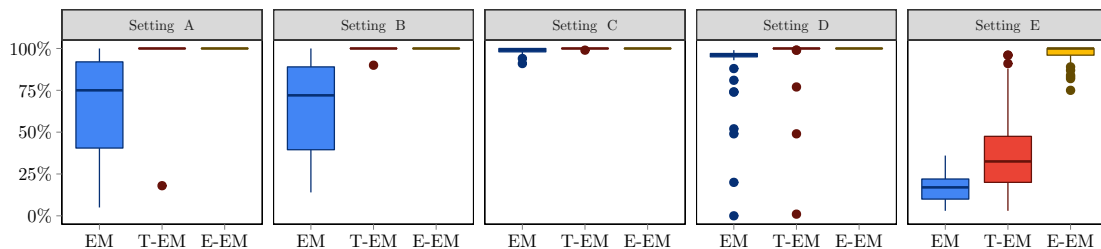
the population, gradually favoring solutions with the optimal number of latent components, and thus converging in a natural and automatic way towards the correct model.

### 3.4 Simulation study

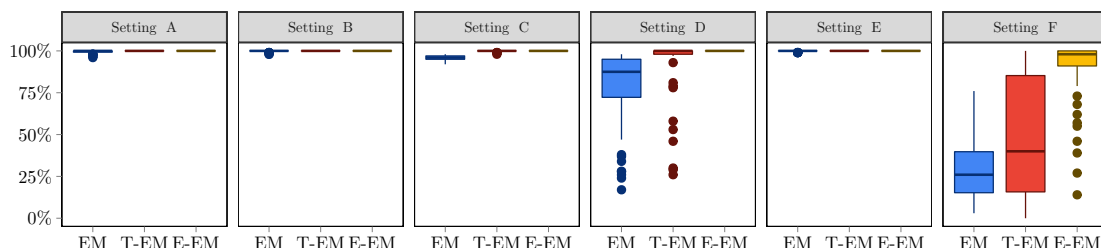
In this Section we present the results of a simulation study aimed at assessing the performance of the E-EM algorithm. In order to compare the proposal with both the EM and the T-EM algorithms, we employ here the same simulation schemes and the same comparison criteria presented in Section 2.5 (see also Appendix A of the previous Chapter for more details about the specific settings). In the following study, in particular, we mainly rely on the percentage of times that global maximum is reached by each algorithm; the corresponding results are summarized in Figures 3.2 and 3.3. Note that in all the plots the E-EM algorithm is compared to both the standard and the tempered versions of the algorithm. As for the latter, the monotonic version is considered throughout the whole Section.

Considering initially the estimation of models with a correctly specified latent structure (see Figure 3.2), the E-EM algorithm shows a clear advantage with respect to the standard EM algorithm. This improvement is generally analogous to the one obtained with the T-EM algorithm when models with a few latent components are considered; on the contrary, focusing on the cases with more latent components, the performance of the E-EM algorithm is considerably superior also with respect to the tempered approach. In particular the probability to reach the global maximum is, on average, very close to 100% when the E-EM algorithm is used.

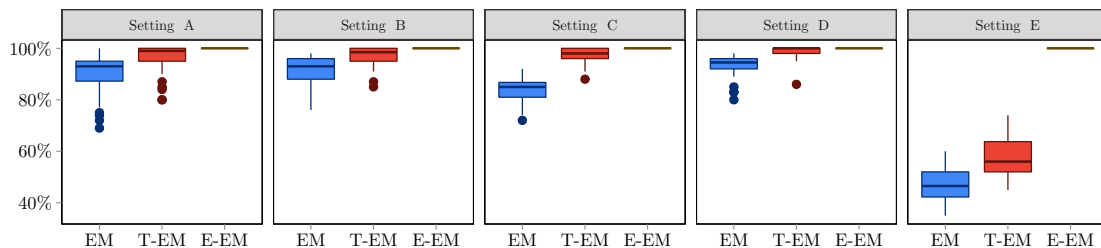
Regarding the estimation of models with misspecified latent structure (see Figure 3.3), the advantage of the E-EM algorithm over both the standard and the tempered version is even more significant. The evolutionary approach allows us to reach the global maximum with a very high incidence in all the considered scenarios; the frequency is usually very close to 100%, highlighting that the E-EM algorithm is generally able to avoid all local maxima. Only the most challenging settings (namely, the ones which involve the estimation of models with many latent components) show lower frequencies, ensuring anyway a considerable advantage with respect to the results obtained with the other two methods. As an example let us consider Setting F for the HM model with categorical response variables; in this case the average frequency of achieved global maximum is approximately equal to 60%, thus making this scenario the most complex from an estimation perspective. The corresponding values for the standard EM and the tempered EM algorithms are 8% and 17% respectively, highlighting the clear superiority of the evolutionary algorithm.



(a) Latent class model with correctly specified latent structure



(b) Hidden Markov model with categorical response variables and correctly specified latent structure

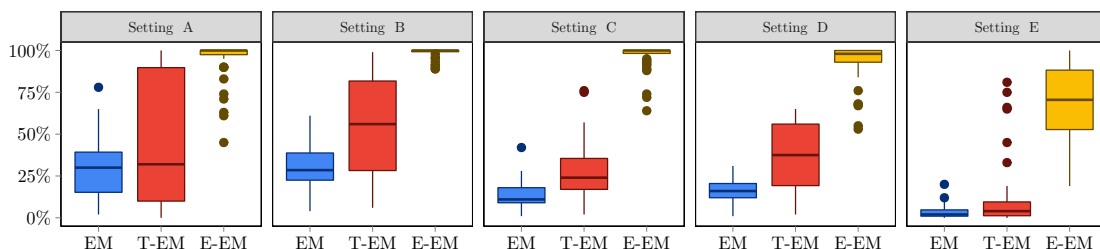


(c) Hidden Markov model with continuous response variables and correctly specified latent structure

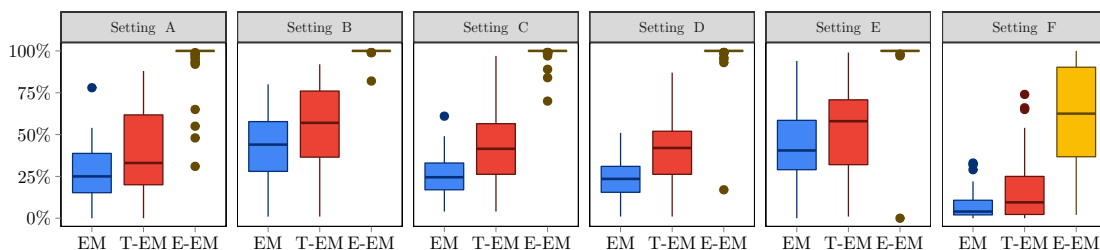
**Figure 3.2:** Percentages of global maxima obtained using EM, M-T-EM, and E-EM algorithms under simulated scenarios presented in Tables 2.17, 2.18, and 2.19 of the Appendix A for the latent class and hidden Markov models with correctly specified latent structure

To conclude this part of the simulation study, we also show, for each simulated scenario, the number of samples in which the global maximum is reached with very low ( $< 10\%$ ) or very high ( $> 95\%$ ) frequency. The results are summarized in Tables 3.1, 3.2, and 3.3 for LC, HM with categorical response variables, and HM with continuous response variables respectively.

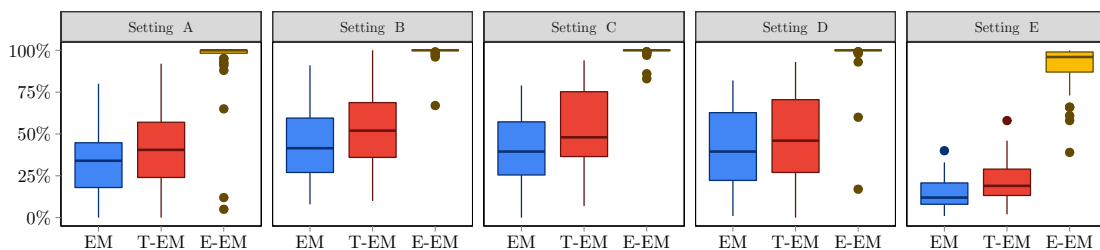
The superior performance of the E-EM algorithm is confirmed by the analysis of these results; although the advantage is evident also in the estimation of models whose latent structure is correctly specified (see, *e.g.*, Setting C and E of the HM model with continuous response variables), the most significant improvements are shown when the models with



(a) Latent class model with misspecified latent structure



(b) Hidden Markov model with categorical response variables and misspecified latent structure



(c) Hidden Markov model with continuous response variables and misspecified latent structure

**Figure 3.3:** Percentages of global maximum using EM, M-T-EM, and E-EM algorithms under simulated scenarios presented in Tables 2.17, 2.18, and 2.19 of the Appendix A for the latent class and hidden Markov models with misspecified latent structure

misspecified latent components are considered. In these cases, the number of samples in which the global maximum is reached with very high frequency ( $> 95\%$ ) is generally superior to 40 (out of a total of 50); on the contrary, the samples with a very low frequency ( $< 10\%$ ) of global maximum is usually negligible. These results confirm that the evolutionary approach outperforms the standard EM algorithm and shows a clear superiority also with respect to the tempered EM algorithm.

We also inspect the mean distance from the global maximum, whose results are summarized in Tables 3.7 and 3.8 in Appendix A. Throughout all the considered scenarios, the E-EM algorithm shows values very close to zero, thus confirming that the global maximum

| Scenario | Correctly specified |              | Misspecified |            |
|----------|---------------------|--------------|--------------|------------|
|          | < 10%               | > 95%        | < 10%        | > 95%      |
| A        | 1 - 0 - 0           | 20 - 49 - 50 | 7 - 2 - 0    | 0 - 8 - 40 |
| B        | 0 - 0 - 0           | 15 - 49 - 50 | 1 - 0 - 0    | 0 - 3 - 44 |
| C        | 0 - 0 - 0           | 47 - 50 - 50 | 9 - 4 - 0    | 0 - 0 - 40 |
| D        | 1 - 1 - 0           | 32 - 47 - 50 | 4 - 1 - 0    | 0 - 0 - 31 |
| E        | 10 - 5 - 0          | 0 - 2 - 40   | 32 - 18 - 0  | 0 - 0 - 4  |

**Table 3.1:** Number of samples in which the global maximum is reached with frequency < 5% or > 95%, using EM (highlighted in blue), M-T-EM (highlighted in red), and E-EM (highlighted in yellow) algorithms under simulated scenarios presented in Table 2.17 of the Appendix A for the latent class model

| Scenario | Correctly specified |              | Misspecified |            |
|----------|---------------------|--------------|--------------|------------|
|          | < 10%               | > 95%        | < 10%        | > 95%      |
| A        | 0 - 0 - 0           | 50 - 50 - 50 | 4 - 1 - 0    | 0 - 1 - 43 |
| B        | 0 - 0 - 0           | 50 - 50 - 50 | 0 - 0 - 0    | 0 - 0 - 49 |
| C        | 0 - 0 - 0           | 35 - 50 - 50 | 4 - 1 - 0    | 0 - 1 - 47 |
| D        | 0 - 0 - 0           | 11 - 41 - 50 | 3 - 2 - 0    | 0 - 0 - 47 |
| E        | 0 - 0 - 0           | 50 - 50 - 50 | 3 - 1 - 1    | 0 - 2 - 49 |
| F        | 7 - 6 - 0           | 0 - 7 - 29   | 27 - 17 - 6  | 0 - 0 - 10 |

**Table 3.2:** Number of samples in which the global maximum is reached with frequency 5% or > 95%, using EM (highlighted in blue), M-T-EM (highlighted in red), and E-EM (highlighted in yellow) algorithms under simulated scenarios presented in Table 2.18 of the Appendix A for the hidden Markov model with categorical response variables

is almost always reached.

### 3.5 Results with real-world data

In this Section we briefly examine the behavior of the E-EM algorithm in connection with real-world cases. We consider here the same applications introduced and analyzed in Section 2.7, which we refer to for a preliminary description of the data. In the following we assess the performance of the E-EM algorithm, comparing it with the standard EM and the T-EM algorithms; we refer again to Section 2.7 for a more detailed analysis of the results.

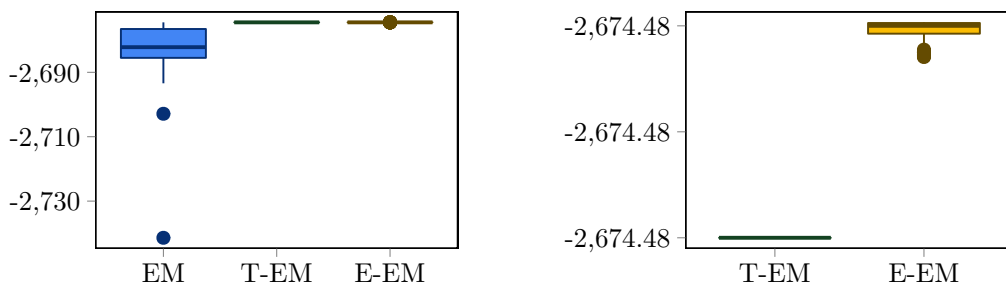
Starting with the LC model, we analyze data derived from the administration of 14 ordinal items measuring anxiety and depression in a sample of 201 oncological Italian pa-



| Scenario | Correctly specified |              | Misspecified |            |
|----------|---------------------|--------------|--------------|------------|
|          | < 10%               | > 95%        | < 10%        | > 95%      |
| A        | 0 - 0 - 0           | 12 - 36 - 50 | 3 - 2 - 1    | 0 - 0 - 41 |
| B        | 0 - 0 - 0           | 14 - 36 - 50 | 0 - 0 - 0    | 0 - 1 - 49 |
| C        | 0 - 0 - 0           | 0 - 40 - 50  | 2 - 0 - 0    | 0 - 0 - 48 |
| D        | 0 - 0 - 0           | 18 - 47 - 50 | 4 - 4 - 0    | 0 - 1 - 47 |
| E        | 0 - 0 - 0           | 0 - 0 - 50   | 17 - 5 - 0   | 0 - 0 - 26 |

**Table 3.3:** Number of samples in which the global maximum is reached with frequency 5% or > 95%, using EM (highlighted in blue), M-T-EM (highlighted in red), and E-EM (highlighted in yellow) algorithms under simulated scenarios presented in Table 2.19 of the Appendix A for the hidden Markov model with continuous response variables

tients (Zigmond and Snaith, 1983). We estimate the model using the E-EM algorithm with a number of latent classes  $k$  ranging from 1 to 4, and we employ the BIC (Schwarz, 1978) index to perform model selection. Consistently with the findings shown in Section 2.7.1, the resulting optimal number of components is three. In Figure 3.4 we show a comparison of the results obtained with the three estimation algorithms when the model with three latent classes is estimated.



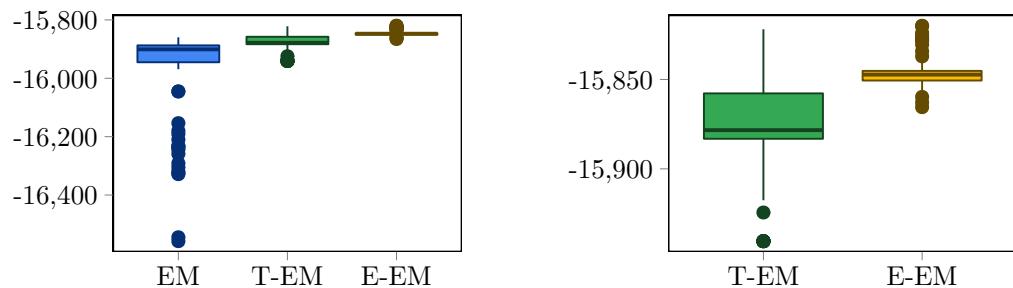
(a) Comparison between EM, O-T-EM, and E-EM algorithms

(b) Detail on the comparison between O-T-EM and E-EM algorithms

**Figure 3.4:** (a) Maximized log-likelihood values for the anxiety and depression data using standard EM (in blue), O-T-EM (in green), and E-EM (in yellow) algorithms. (b) Zoom on the comparison between O-T-EM and E-EM algorithms

The T-EM and E-EM algorithms show a similar superiority compared to the standard version, ensuring a much more frequent convergence to the global maximum. In particular, the former slightly improves the value of the maximized log-likelihood function also with respect to the O-T-EM algorithm.

A similar behavior is obtained with the HM model with continuous response variables, analyzing from the World Bank’s World Development Indicators ([The World Bank Group, 2018](#)) on  $n = 175$  countries collected for  $T = 5$  years (from 2011 to 2015) on  $r = 6$  continuous social and economic response variables. The time heterogeneous HM model is estimated with the E-EM algorithm, for a number of latent states ranging from 1 to 10. Relying on the BIC index to select the optimal number of latent components, we obtain a model with 7 latent states; this result is in accordance with the conclusion drawn with the O-T-EM algorithm in Section 2.7.2 (while the standard EM algorithm leads to select 8 latent states, with higher values of the BIC index). A comparison between the performance of the three algorithms in estimating the optimal model is shown in Figure 3.5.



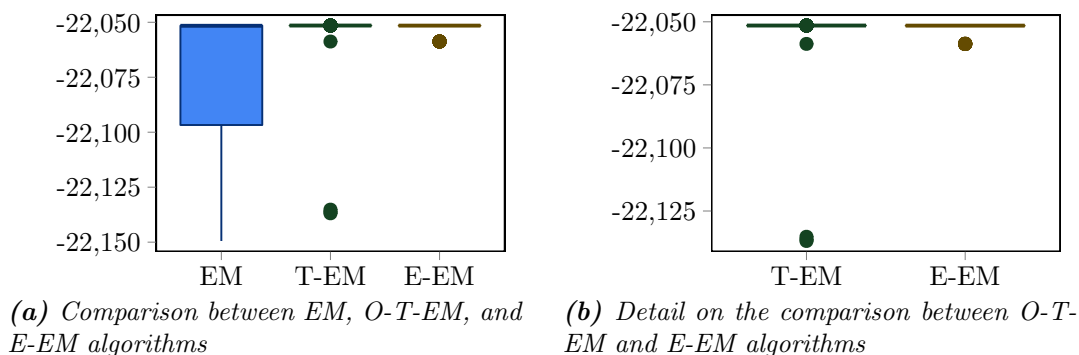
(a) Comparison between EM, O-T-EM, and E-EM algorithms (b) Detail on the comparison between O-T-EM and E-EM algorithms

**Figure 3.5:** (a) Maximized log-likelihood values for the countries’ economic conditions data using standard EM (in blue), O-T-EM (in green), and E-EM (in yellow) algorithms. (b) Zoom on the comparison between O-T-EM and E-EM algorithms

In this case, it is even more evident the different performance of the three algorithms: while the standard EM algorithm converges to a wide range of values, the two modified versions avoids low values of the log-likelihood functions, always being very close to the global maximum. In particular, the E-EM algorithm shows a slight superiority also with respect to the T-EM algorithm, reaching, on average, higher values. As a final note, the global maximum reached by the E-EM algorithm (approximately equal to  $-18,820$ ) outperforms the highest obtained by the T-EM algorithm (equal to  $-15,822$ ).

Finally, dealing with the HM model with categorical response variables, we consider longitudinal data on conviction histories of a cohort of  $n = 10,000$  offenders. A time heterogeneous HM model is estimated with the E-EM algorithm for a number of latent states ranging from 1 to 5. According to the BIC index, and coherently with the results

obtained in Section 2.7.3, the optimal number of latent states is four. Figure 3.6 summarizes the comparison between the behavior of the three algorithms in estimating this optimal model.



**Figure 3.6:** (a) Maximized log-likelihood values for the criminal data using standard EM (in blue), O-T-EM (in green), and E-EM (in yellow) algorithms. (b) Zoom on the comparison between O-T-EM and E-EM algorithms

The situation is here slightly different: both the O-T-EM and the E-EM algorithms outperform the standard EM algorithm, but in this case the evolutionary version does not provide any significant advantage with respect to the tempered algorithm, and both of them are able to steadily reach the global maximum.

### 3.6 Conclusions

In this Chapter, considering the problem of convergence of the log-likelihood function to local maxima again, a new estimation algorithm is explored. The underlying idea of this evolutionary EM (E-EM) algorithm is incorporating the standard E-step and M-step into an evolutionary framework. In this context, clearly inspired by the basic principles of the Darwinian theory of natural evolution, a population is considered, in which each individual represents a candidate solution of the model parameters estimation problem. Selection of the best individuals is performed through a fitness function that determines the quality of each candidate solution: the best ones are selected for the successive steps, and the worst ones are discarded. Other evolutionary operators, such as crossover and mutation, ensure adequate parameter space exploration.

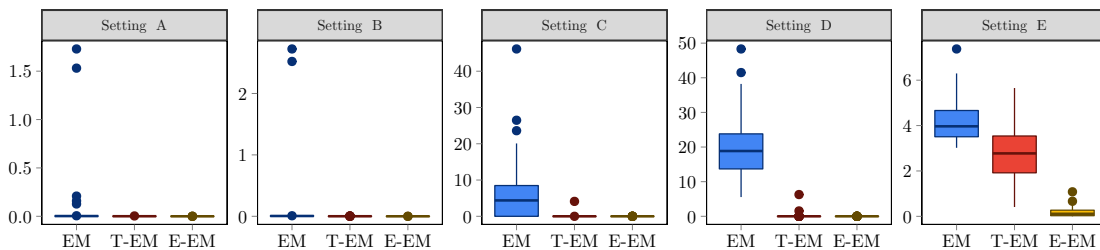
An accurate Monte Carlo simulation study is carried out to assess the proposal's performance for two general classes of DLV models, namely latent class and hidden Markov

models. In particular, the ability to reach the global maximum is evaluated, allowing us to compare the performance of the E-EM algorithm with the standard and the tempered versions. To conclude the present Chapter, we summarize the main differences, advantages and disadvantages of the two proposals. Both approaches clearly show superior performance with respect to the standard EM algorithm; this behavior is distinctly evident throughout all the simulation studies and by applying the proposed algorithms to real-world data. The E-EM algorithm generally provides the best results, even compared to the tempered version; in particular, it allows us (i) to improve the value of the global maximum, and (ii) to increase the percentage of times that the global maximum is reached. Although both the proposed algorithms rely on some parameters (namely,  $\alpha$ ,  $\beta$ ,  $T_0$ , and  $\rho$  for the T-EM algorithm, and  $n_p$  and  $n_c$  for the E-EM algorithm), the evolutionary approach ensures a much more straightforward interpretation:  $n_p$  and  $n_c$  represent the number of parent and children individuals respectively. None of the tempering parameters has such a clear meaning. The effect of a change in the value of the parameters on the performance of the E-EM algorithm is fully known: it is evident that increasing the value of  $n_p$  and  $n_c$  (number of parents and offspring, respectively) ensures a better performance (obviously, at the cost of a higher computational time). Conversely, as  $n_p$  and  $n_c$  decrease, the algorithm resembles the standard EM version. These fully-known effects allow us to control the algorithm behavior (regardless of the considered model). On the contrary, regarding the T-EM algorithm, we may only suggest a broad range of values for which the algorithm shows a good performance. These are quite general, but it is impossible to state that they hold in every instance. In conclusion, no precise rules explaining and interpreting the dependence of the algorithm performance based on the parameter's value is available for the T-EM algorithm. Finally, this different behavior implies a relevant advantage for the E-EM algorithm in terms of computational time. Indeed it does not need a tuning procedure for the parameters, which, on the contrary, is often essential for the T-EM algorithm.

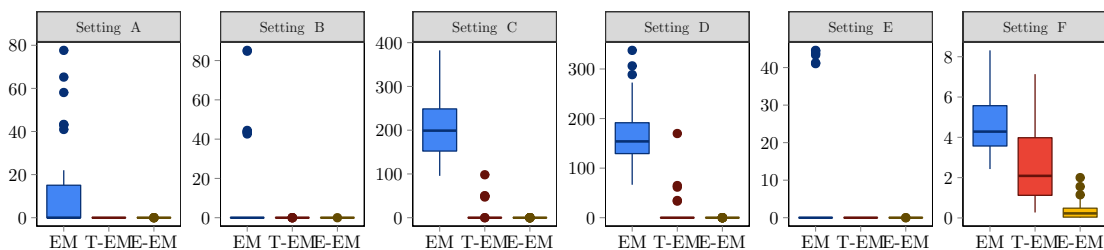
## Appendices

### A Additional simulation results

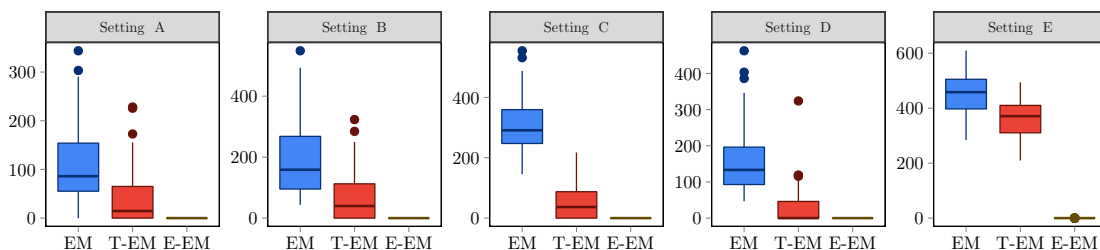
In this Section we report additional details on the results of the simulation study in Section 3.4. In particular, for each considered scenario, Figures 3.7 and 3.8 show the distribution of the mean distance from the global maximum through boxplots.



(a) Latent class model with correctly specified latent structure

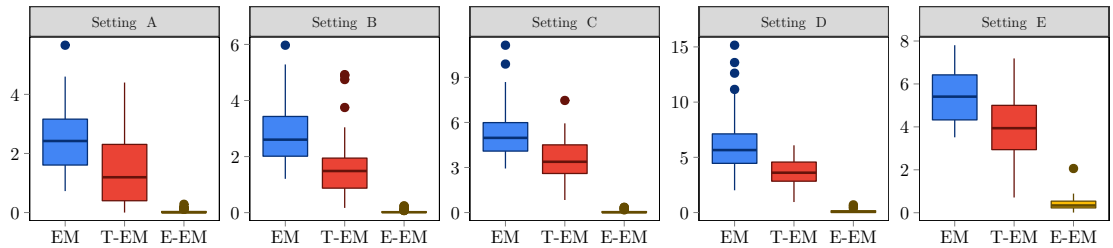


(b) Hidden Markov model with categorical response variables and correctly specified latent structure

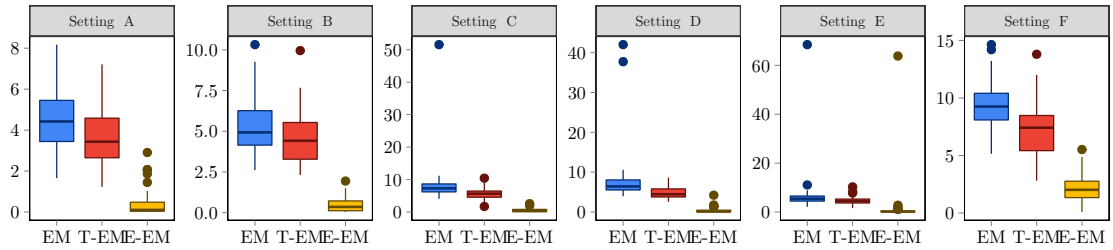


(c) Hidden Markov model with continuous response variables and correctly specified latent structure

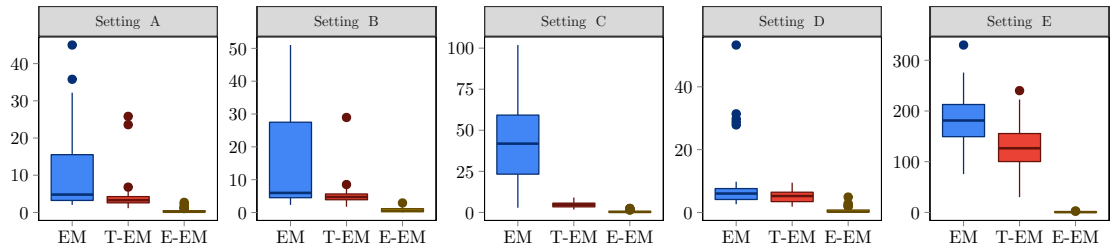
**Figure 3.7:** Mean distance from the global maximum obtained using EM, M-T-EM, and E-EM algorithms under simulated scenarios presented in Tables 2.17, 2.18, and 2.19 of the Appendix A for the latent class and hidden Markov models with correctly specified latent structure



(a) Latent class model with misspecified latent structure



(b) Hidden Markov model with categorical response variables and misspecified latent structure

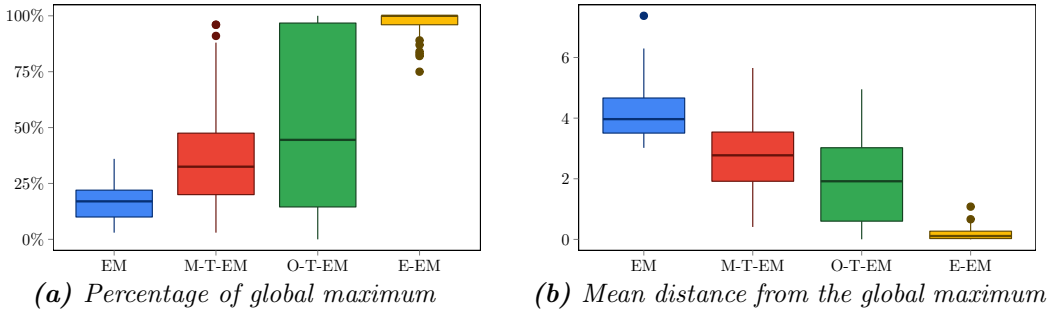


(c) Hidden Markov model with continuous response variables and misspecified latent structure

**Figure 3.8:** Mean distance from the global maximum using EM, M-T-EM, and E-EM algorithms under simulated scenarios presented in Tables 2.17, 2.18, and 2.19 of the Appendix A for the latent class and hidden Markov models with misspecified latent structure

## B Comparison with the oscillating T-EM algorithm

We compare here the E-EM algorithm with the oscillating version of the T-EM algorithm. We recall that, although the latter outperforms the monotonic version, this gain in performance comes with a much higher computational time, which makes the employment of the O-T-EM algorithm rather complex in many cases (see Section 2.5.4). The results of this study are summarized in Figure 3.9.



**Figure 3.9:** Percentage of global maximum and mean distance from it using EM, M-T-EM, O-T-EM, and E-EM algorithms on simulated data from an latent class model correctly specified with 6 latent classes

As clearly shown in Figure 3.9, the E-EM algorithm outperforms all the other versions. The global maximum is reached on average about 17% of times with the standard EM algorithm (this percentage rarely exceeds the 25%), which increases up to 37% with the M-T-EM algorithm and up to 45% with the oscillating version. The E-EM algorithm shows a superior performance, allowing us to steadily reach the global maximum: the corresponding percentage is on average equal to 97%, and it never decreases below 75%. Employing this algorithm, the median percentage value is equal to 100%, highlighting that for many samples the global maximum is always reached. The mean distance from the global maximum decreases accordingly.

## C Additional features about crossover and mutation

Evolutionary operators such as crossover and mutation can be defined in multiple ways; throughout this Chapter, they are performed on the arrays containing the posterior probabilities of the latent components for each response configuration. A possible alternative, sometimes analyzed in the literature, considers the evolution of the parameter space instead. As already stated in Section 4.3, the latter approach involves some drawbacks, especially when categorical response variables are taken into account. In the following, in order to justify the choice made, we briefly compare the peculiarities of the two approaches.

First, apart from the mean vectors and the variance-covariance matrix in the hidden Markov model with continuous response variables, all the models' parameters are probabilities subject to some kind of constraints (*e.g.* the weights, or initial probabilities, must sum to 1 in all models). This aspect complicates the implementation of crossover on the model parameters: as a simple example, let us consider the following two initial probability vectors:  $\pi_1 = [0.4, 0.6]$  and  $\pi_2 = [0.2, 0.8]$ . Performing crossover between  $\pi_1$  and  $\pi_2$  requires to swap the (for example) first element of  $\pi_1$  with the corresponding one of  $\pi_2$ ; this operation results into two new vectors that are not normalized. A normalization step would obviously affect each element of both vectors, exceeding the purposes of the crossover operator. The same happens for the transition and conditional response probabilities, and a similar issue also affects the mutation operator. To this aim, note that in [Pernkopf and Bouchaffra \(2005\)](#) and in [Andrews and McNicholas \(2013\)](#) an evolutionary approach performing crossover and mutation on the parameters is actually defined for the Gaussian mixture model. However mutation and crossover operators are only applied to the mean vector and the covariance matrix, without including the weights vector.

Secondly, the parameters-based approach would require considering many classes of parameters: for example, in the case of hidden Markov models with categorical response variables, mutation and crossover should be performed on the initial probabilities, on the transition probabilities, and on the conditional response probabilities. Employing mutation and crossover on the a posteriori probabilities, on the contrary, allows us to operate on a unique structure (the posterior probability matrix); this ensures an advantage from the point of view of the computational complexity. Moreover, this also allows us to define a single common implementation of the E-EM algorithm for all the models, without having to consider specific changes according to each model and to its parameters.

Moreover, employing mutation and crossover on the posterior probabilities ensures a more straightforward interpretation of the evolutionary operators (mutation simply changes the latent component to which a subject is assigned; crossover implies allocating a portion



of the subjects according to one of the two parents, and the remaining subjects according to the other parent). Such a clear meaning is not available for a potential version operating on the model parameters.

Finally, [Andrews and McNicholas \(2013\)](#) compared the two approaches in the case of Gaussian mixture models, without showing any significant advantage deriving from performing mutation and crossover on the parameters.

## Bibliography

- ANDREWS, J. AND MCNICHOLAS, P. (2013). Using evolutionary algorithms for model-based clustering. *Pattern Recognition Letters*, **34**, 987–992.
- ASHLOCK, D. (2004). *Evolutionary Computation for Modeling and Optimization*. Springer, New York.
- BACK, T. (1994). Selective pressure in evolutionary algorithms: A characterization of selection mechanisms. In *Proceedings of the first IEEE conference on evolutionary computation. IEEE World Congress on Computational Intelligence*, pages 57–62. IEEE.
- BACK, T. AND SCHWEFEL, H.-P. (1996). Evolutionary computation: An overview. In *Proceedings of IEEE International Conference on Evolutionary Computation*, pages 20–29. IEEE.
- BÄCK, T. (1996). *Evolutionary Algorithms in Theory and Practice: Evolution Strategies, Evolutionary Programming, Genetic Algorithms*. Oxford University Press, New York.
- BLICKLE, T. AND THIELE, L. (1995). A comparison of selection schemes used in genetic algorithms. tik-report 11, tik institut für technische informatik und kommunikationsnetze. *Computer Engineering and Networks Laboratory, ETH, Swiss Federal Institute of Technology, Gloriastrasse*, **35**, 279–284.
- DEB, K. (2001). *Multi-Objective Optimization using Evolutionary Algorithms*. John Wiley and Sons, Chichester.
- ENCYCLOPAEDIA BRITANNICA (2022). Natural selection. <https://www.britannica.com/science/natural-selection>.
- GOLDBERG, D. AND DEB, K. (1991). A comparative analysis of selection schemes used in genetic algorithms. In *Foundations of genetic algorithms*, volume 1, pages 69–93. Elsevier.
- HRUSCHKA, E., CAMPELLO, R., FREITAS, A., AND PONCE LEON F. DE CARVALHO, A. (2009). A survey of evolutionary algorithms for clustering. *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, **39**, 133–155.
- KAMPO, R. (2021). *Evolutionary Algorithms for Model-Based Clustering*. PhD thesis, McMaster University, Hamilton, Ontario, Canada.
- MCNICHOLAS, S., MCNICHOLAS, P., AND ASHLOCK, D. (2021). An evolutionary algorithm with crossover and mutation for model-based clustering. *J Classif*, **38**, 264–279.

- MICHALEWICZ, Z. AND FOGEL, D. (2000). *How to Solve It: Modern Heuristics*. Springer, Berlin, Heidelberg.
- PERNKOPF, F. AND BOUCHAFFRA, D. (2005). Genetic-based em algorithm for learning gaussian mixture models. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **27**, 1344–1348.
- SCHWARZ, G. (1978). Estimating the dimension of a model. *Annals of Statistics*, **6**, 461–464.
- THE WORLD BANK GROUP (2018). Data catalog: World development indicators. Available from <https://datacatalog.worldbank.org/dataset/world-development-indicators>.
- WIECZOREK, W. AND CZECH, Z. (2002). Selection schemes in evolutionary algorithms. In *Intelligent Information Systems 2002*, pages 185–194. Physica-Verlag HD.
- ZIGMOND, A. AND SNAITH, R. (1983). The hospital anxiety and depression scale. *Acta Psychiatr. Scand.*, **67**, 361–70.



# Model-based clustering in simple hypergraphs through a stochastic blockmodel<sup>1</sup>

---

## 4.1 Introduction

Over the past two decades a broad variety of models has been developed for pairwise interactions, encoded in graphs. These allow to better comprehend the complexity of many real-world systems: interconnected components often display properties that cannot be deduced from the analysis of the individual elements. Systems of this kind, characterized by richness of interactions among their units, emerge in almost every branch of science and technology, including chemistry, physics, biology, geology, ecology, engineering, and even psychology, economy, and social sciences. See [Newman \(2010, Chapters 1 to 5\)](#) for a thorough list of interesting examples. However, modern applications in various fields highlight the need to account for higher-order interactions, to include the information deriving from groups of three or more nodes. *Hypergraphs* provide the most general formalization of higher-order interactions: similarly to a graph, we define a hypergraph as a set of nodes and a set of hyperedges, where each hyperedge is a subset of distinct nodes taking part in

---

<sup>1</sup>Part of this chapter has been submitted for publication in an international statistical journal: BRUSA, L., MATIAS, C. Model-based clustering in simple hypergraphs through a stochastic blockmodel.

an interaction. In particular, here we distinguish these *simple* hypergraphs from *multisets-hypergraphs* where multiset hyperedges are allowed. A multiset is the generalization of a set where each element may appear with some multiplicity; thus, multiset hyperedges occur when repeated nodes appear. It is clear that, depending on each specific real-world case, only one of the two notions of hypergraph is appropriate.

Despite an increasing interest for these higher-order interactions, the statistical literature on this topic remains quite scarce. Statistics such as centrality or clustering coefficient have been extended from graphs to hypergraphs (Estrada and Rodríguez-Velázquez, 2006). These help to understand the structure and extract information from the data but do not fill the need for hypergraphs models. In the graphs context, as mentioned in Section 1.2.3, stochastic block (SB) models were introduced in the early eighties (Frank and Harary, 1982; Holland et al., 1983) and have flourished in many directions. These models assume that nodes are clustered into groups and the connection probabilities between nodes are driven by their groups memberships. Variants handling weighted graphs and degree corrected versions have been developed among others. Despite these new improvements, generalizations of the SB model for hypergraphs are currently limited to some restrictive hypotheses. In particular, to our best knowledge, extensions in the context of simple hypergraphs are still missing in the literature. In order to overcome this limitation, in the present Chapter we focus on model-based clustering for simple hypergraphs and study stochastic hypergraph block models.

The algorithm implementation in C++ is freely available as an R package called `HyperSBM` at <https://github.com/LB1304/HyperSBM>. The files to reproduce the synthetic experiments and the dataset analysis are available at <https://github.com/LB1304/Hypergraph-Stochastic-Blockmodel>.

The remaining of this chapter is organized as follows. In Section 4.2 we introduce the general context of higher-order interactions and their representation through hypergraphs. We furthermore discuss the multisets-hypergraphs assumption, often presented as a harmless one in the literature, and highlight its consequences on datasets analysis. These consequences motivate our focus on simple hypergraphs, where much less has been done, while computational challenges are higher. In Section 4.3 we present an overview of models for hypergraphs existing in previous literature. In Section 4.4, we formulate a general stochastic simple hypergraphs block model as well as different submodels. Parameter inference is performed through a variational expectation-maximization (VEM) algorithm (Section 4.4.3) and model selection relies on an integrated completed likelihood (ICL) criterion (Section 4.4.4). In Sections 4.5 and 4.6 we illustrate the performance of our method through an analysis on synthetic and real datasets. In Section 4.7 we provide some conclu-

sions. All the proofs are postponed to Appendices A and B, while Appendices from C to G contains additional details on both computational and theoretical aspects.

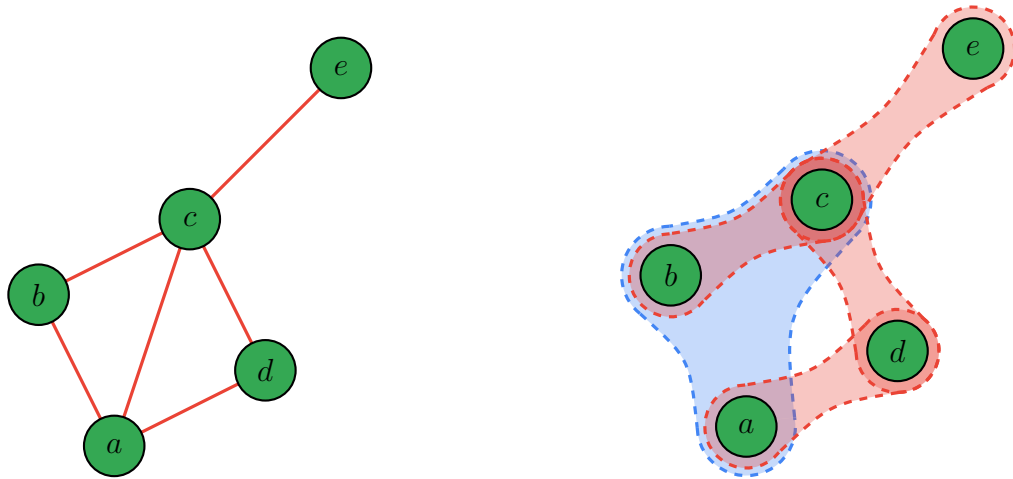
## 4.2 Interacting systems and hypergraphs

In this Section we analyze in more details the general structure of a complex, or interacting, system, formally introducing the concept of interaction and discussing its mathematical representations. Following [Battiston et al. \(2020\)](#), we define an  $m$ -interaction as a set  $I = [p_1, \dots, p_m]$ , where  $p_1, \dots, p_m$  denote the  $m$  elements of the system among which the interaction takes place. We refer to the case  $m = 2$  as a binary interaction, and to  $m \geq 3$  as higher-order interaction. We refer to [Battiston et al. \(2020\)](#), [Bick et al. \(2021\)](#), and [Torres et al. \(2021\)](#) for recent reviews on higher-order interactions.

### 4.2.1 Higher-order interactions representation: from graphs to hypergraphs

The simplest and most natural mathematical entity used to represent interactions among the basic elements of a complex system is the *graph* structure. As introduced in Section 1.2.3, a graph  $\mathcal{G} = (\mathcal{V}, \mathcal{E})$  is defined as a set  $\mathcal{V} \neq \emptyset$  of distinct nodes, representing the various elements of the complex system, and of a set  $\mathcal{E}$  of edges, consisting of pairs of distinct elements of  $\mathcal{V}$ . Despite recent developments and improvements, a serious limitation of graphs remains the inability to account for higher-order interactions, capturing interactions between groups of more than two nodes. This task is nowadays fundamental in many modern applications in various field; simple example include (but are not limited to) triadic and larger groups interactions in social networks (whose importance has early been acknowledged in [Simmel, 1950](#)), scientific co-authorship ([Roy and Ravindran, 2015](#)), interactions between more than two species in ecological systems ([Muyinda et al., 2020](#); [Singh and Baruah, 2021](#)) or higher-order correlations between neurons in brain networks ([Chelaru et al., 2021](#)).

A common approach to provide a graph-based representation of a higher-order interaction, is to unfold it in terms of binary interactions; for example, a simple triadic interaction  $I = [a, b, c]$  can be easily decomposed into the following collection of binary interactions:  $I_B = \{[a, b], [b, c], [c, a]\}$ . This description, however, lacks a proper interpretation: although they are associated to then same graph representation,  $I$  and  $I_B$  do not convey the same information. Let us consider, for example, co-authorship networks; in this context there is a significant difference between a single paper written by three authors (interaction  $I$ ),



**Figure 4.1:** Visualization of the set of higher-order interactions  $\{[a, b, c], [a, d], [c, d], [c, e]\}$  through a graph (on the left) and a hypergraph (on the right). The latter provides a much more accurate representation

and three different papers written by pairs of those authors (set of interactions  $I_B$ ). In other words, through this representation, it is impossible to state whether any higher-order interaction is actually present or not.

*Hypergraphs* (Battiston et al., 2020) give an explicit encoding of higher-order interactions, thus providing the most appropriate description. Similarly to a graph, a hypergraph  $\mathcal{H} = (\mathcal{V}, \mathcal{E})$  is defined as a set of nodes  $\mathcal{V} \neq \emptyset$  and a set of hyperedges  $\mathcal{E}$ . Each hyperedge is a non-empty collection of  $m$  distinct nodes taking part within an interaction. This entity naturally includes the one of simple graphs, by simply considering  $m = 2$  for each hyperedge  $e \in \mathcal{E}$ . Note that a hypergraph can contain a hyperedge of size 3  $[a, b, c]$  without any requirement on the existence of the hyperedges of size 2  $[a, b]$ ,  $[a, c]$ , and  $[b, c]$ . An example of hypergraph representing a higher-order interacting system is represented in shown in Figure 4.1.

### 4.2.2 The need for simple hypergraphs models

In this Section, we discuss modeling differences between multisets-hypergraphs where multiset hyperedges are allowed versus simple hypergraphs where hyperedges are subsets of distinct nodes. We first recall that a multiset is the generalization of a set where each element may appear with some multiplicity. Thus, multiset hyperedges occur when nodes may be repeated in a hyperedge.

Multiset hyperedges generalize in some sense the notion of self-loops in graphs, thus



constituting a natural extension to consider. However, they are not appropriate in all contexts. For instance, a co-authorship dataset cannot contain hyperedges with repeated nodes (but it may contain a self-loop of a unique author). In the same way, a social interaction hypergraph does not contain multiset hyperedges: triadic interactions are restricted to 3 different individuals and self-loops are not allowed. In the meantime, they are natural in other contexts; *e.g.* chemical reaction hypergraphs where the multiplicity plays the role of the stoichiometric coefficients (Flamm et al., 2015). We first argue that multiset hypergraphs models are inappropriate for analyzing simple hypergraphs.

#### 4.2.2.1 A motivating example

For the sake of simplicity, we restrict our attention to uniform size-3 hypergraphs on a set of  $n$  nodes and consider two different models. The first one, denoted as MH, acts on 3-uniform multisets-hypergraphs and draws a hyperedge between any 3 nodes, not necessarily distinct, with probability  $p_{\text{MH}}$ . The second one, denoted as SH, acts on 3-uniform simple hypergraphs and draws a hyperedge between any 3 distinct nodes, with probability  $p_{\text{SH}}$ .

Let us consider the toy example of a simple hypergraph  $\mathcal{H}$  with  $n = 3$  nodes and only one hyperedge  $e = \{1, 2, 3\}$ . This dataset could correspond for instance to observing one publication with 3 authors. When analyzed under the MH model, the density of our observed hypergraph is estimated by

$$\hat{p}_{\text{MH}} = 1/27$$

because there are  $n^3 = 27$  possible size-3 multiset hyperedges under this model, and just one of these is observed. On the contrary, when analyzed under the SH model, we infer a density of

$$\hat{p}_{\text{SH}} = 1$$

because the only possible size-3 hyperedge is observed. As a consequence, the statistical conclusions drawn on this dataset will highly differ depending on whether we restrict attention to simple hypergraphs or work with more general multisets-hypergraphs. This simple and elementary example shows that it is not possible to statistically analyze a simple hypergraph with a multisets-hypergraphs model without erroneous conclusions. This choice of the ambient space has to be made according to the specificities of the dataset.

### 4.2.2.2 Computational challenge in the simple hypergraph case

The main technical difference between multisets-hypergraphs and simple hypergraphs analysis comes from the enumeration of the size- $m$  tuples of nodes. In the multisets-hypergraphs setting, the summations over multisets of nodes  $\{i_1, \dots, i_m\} \in \{1, \dots, n\}^m$  factorize into  $m$  independent sums. On the contrary, in the simple hypergraph setting, the summations involve sets of nodes  $\{i_1, \dots, i_m\}$  that are constrained to be distinct. As a consequence, such a factorization is impossible.

Let us consider a concrete example. In the context of interaction data, modularity is a widely used criterion for clustering entities. It is designed to obtain specific clusters, called communities, characterized by large intra-group and low inter-group connections (see also Section 4.3.2 for a more detailed analysis of modularity). When dealing with hypergraphs modularity criteria have been proposed only in the multisets-hypergraphs setting ([Kamiński et al., 2019](#); [Chodrow et al., 2021](#)). Modularities are generally constructed as deviation measures of the number of hyperedges from their expected number under a null model. For instance in the graphs context, the Newman and Girvan modularity of a partition  $(C_1, \dots, C_Q)$  of the nodes into  $Q$  clusters is computed as

$$\begin{aligned} \text{Modularity}(C_1, \dots, C_Q) &= \frac{1}{2|E|} \sum_{q=1}^Q \sum_{i,j \in C_q} \left( A_{ij} - \frac{d_i d_j}{2|E|} \right) \\ &= \frac{1}{2|E|} \sum_{q=1}^Q \sum_{i,j \in C_q} A_{ij} - \frac{1}{2|E|} \sum_{q=1}^Q \sum_{i,j \in C_q} \frac{d_i d_j}{2|E|}, \end{aligned}$$

where  $A = (A_{ij})_{i,j}$  is the graph adjacency matrix,  $d_i$  is the degree of node  $i$  and  $2|E| = \sum_i d_i$  is twice the number of edges. While the first part of these criteria enumerates only the (present) hyperedges, a quantity that is small in general as most hypergraph datasets are sparse, the second part needs to account for all tuples of nodes in the graph (or at least in the same group  $C_q$ ). In the case of multisets-hypergraphs, this second term factorizes to

$$\frac{1}{2|E|} \sum_{q=1}^Q \sum_{i,j \in C_q} \frac{d_i d_j}{2|E|} = \frac{1}{2|E|} \sum_{q=1}^Q \frac{(\sum_{i \in C_q} d_i)(\sum_{j \in C_q} d_j)}{2|E|} = \sum_{q=1}^Q \frac{\text{Vol}(q)^2}{(2|E|)^2},$$

where the time complexity of the volume  $\text{Vol}(q) = \sum_{i \in C_q} d_i$  computation is  $O(n)$ . On the contrary, in the simple hypergraph setting, enumerating all constrained tuples of nodes

requires enumerating

$$\sum_{m=2}^M \binom{n}{m}$$

elements for a hypergraph with  $n$  nodes and maximum hyperedge size  $M$ . This quantity is huge and represents the main computational limit when analyzing hypergraphs. Our approach to this issue is detailed in Appendix D.

### 4.2.3 Matrix representations of higher-order interactions

To conclude this Section, we briefly describe the representation of higher-order interactions in terms of matrices. These structures generalize the standard notions existing in the graph literature.

The *incidence matrix* of a hypergraph encodes the relation of node-hyperedge pairs. In particular, let us denote by  $H = (H_{ie})$  the  $n \times N$  matrix, where  $n$  and  $N$  are the numbers of nodes and hyperedges, respectively, and the entry  $H_{ie}$  is equal to 1 if and only if node  $i$  belongs to hyperedge  $e$ , and 0 otherwise. The incidence matrix provides important properties of the hypergraphs. The degree of a node  $i$  (*i.e.* the number of hyperedges incident to the node) can be defined as the sum of the elements of the  $i$ -th row of the incidence matrix:  $\deg(i) = \sum_{e=1}^N h_{ie}$ . Conversely, the sum of the elements in the  $e$ -th columns defines the size of hyperedge  $e$  (*i.e.*, number of nodes taking part in the hyperedge):  $|e| = \sum_{i=1}^n h_{ie}$ .

The *adjacency matrix* of a hypergraph encodes the relation of node-node pairs. It is defined as the  $n \times n$  matrix  $A = (A_{ij})$ , where the entry  $A_{ij}$ , with  $i \neq j$ , counts the number of hyperedges containing both nodes  $i$  and  $j$ . As long as we do not allow for self-loops, the diagonal entries  $a_{ii}$  are all equal to 0. The adjacency matrix is related to the incidence matrix by the following simple formula:  $A = II' - D$ , where  $D$  is the diagonal matrix, with entry  $d_{ii} = \deg(i)$  equal to the number of hyperedges node  $i$  belongs to.

A hypergraph *Laplacian operator* (Zhou et al., 2006) can be defined on the basis of the above matrices as follows:

$$L = I_n - D^{-1/2} H \Delta^{-1} H' D^{-1/2} \in \mathbb{R}^{n \times n},$$

where  $I$  is the identity matrix of size  $n$ ,  $H$  is the incidence matrix,  $D$  and  $\Delta$  are diagonal matrices such that  $d_{ii} = \deg(i)$  and  $\delta_{ee} = |e|$ . Similarly to the simple graph context, Laplacian matrix and the distribution of its eigenvalues convey many important properties of hypergraphs; we will address a simple example in Section 4.4.3.3. Note that several other

notions of hypergraph Laplacian have been proposed in the literature; see, among others, [Bolla \(1993\)](#) and [Rodríguez \(2002\)](#).

### 4.3 Preliminary works about hypergraph modeling

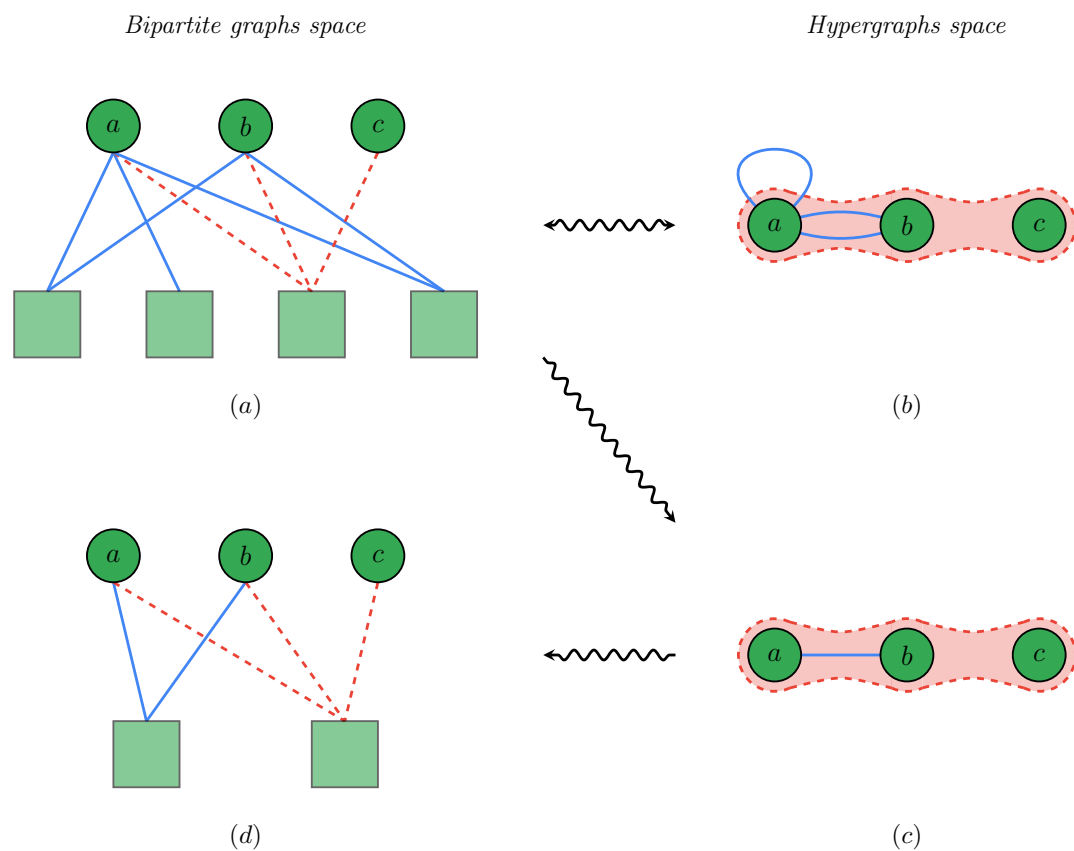
In this Section we briefly summarize the main contributions available in the statistical literature in the field of higher-order interactions and hypergraphs.

#### 4.3.1 The bipartite graph representation and its limits

Some early analyses of hypergraphs rely on the embedding of the former into the space of bipartite graphs (see for *e.g.* [Battiston et al., 2020](#)). A bipartite graph is defined by two sets of nodes ( $\mathcal{U}, \mathcal{W}$ ) and a set of edges; each edge  $e$  must connect a node from set  $\mathcal{U}$  and a node from set  $\mathcal{W}$ , *i.e.*  $e = (u, w)$  with  $u \in \mathcal{U}$  and  $w \in \mathcal{W}$ . Any hypergraph  $\mathcal{H} = (\mathcal{V}, \mathcal{E})$  may be represented as a bipartite graph:  $\mathcal{U}$  is chosen as the set of hypergraphs nodes, so  $\mathcal{U} = \mathcal{V}$ , while  $\mathcal{W}$  is the set of hyperedges, so that  $\mathcal{W} = \mathcal{E}$ : there is a link between  $v \in \mathcal{V}$  and  $e \in \mathcal{E}$  whenever node  $v$  belongs to hyperedge  $e$  in the original hypergraph  $\mathcal{H}$ .

It is possible to define a “converse” application from bipartite graphs to hypergraphs. Indeed, any bipartite graph can be projected into two distinct hypergraphs, by choosing one of the two parts as the nodes set and forming a hyperedge with any set of nodes that are neighbors (in the bipartite graph) of the same node (belonging to the second part). A major difference appears whether we consider simple hypergraphs or multi-hypergraphs with self-loops. In multi-hypergraphs (not to be confused with multisets-hypergraphs) hyperedges may appear several time so that these are weighted hypergraphs with integer valued weights. In multi-hypergraphs, we also allow for self-loops, *i.e.* hyperedges of cardinality 1. Then, this application from bipartite graphs to hypergraphs slightly differs depending on whether we allow the image of a bipartite graph to be a multi-hypergraphs with self-loops or a simple hypergraph. In the first case, all the information from the bipartite graph will be encoded in the multi-hypergraphs with self-loops; while in the second case, part of the information will be lost. This is illustrated on a toy example in [Figure 4.2](#).

The embedding of the simple hypergraphs space into the bipartite graphs space is not the inverse of the natural projection of bipartite graphs into simple hypergraphs. Thus, models of bipartite graphs are inappropriate to handle simple hypergraphs, as the former generally put mass on any bipartite graph, notwithstanding the fact that not all of these may be realized as the image of a simple hypergraph. For the same reason, preferential attachment models of bipartite graphs ([Guillaume and Latapy, 2004](#)) may not be directly



**Figure 4.2:** (a) A bipartite graph  $\mathcal{G}$ ; (b) Projection of  $\mathcal{G}$  into the multi-hypergraphs with self-loops space, choosing the top nodes as the new set of nodes. Hyperedges are  $\{a\}, \{a, b\}, \{a, b, c\}$ . The applications from (a) to (b) are invertible bijections, one being the inverse of the other; (c) Projection of  $\mathcal{G}$  on the simple hypergraphs subspace. Hyperedges are  $\{a, b\}, \{a, b, c\}$ ; (d) Embedding of the simple hypergraph in (c) in the bipartite graphs space. Note that (a) and (d) are not the same bipartite graph

used for simple hypergraphs as they would produce unconstrained bipartite graphs that do not necessarily come from simple hypergraphs. Appendix C contains further considerations in the same line.

### 4.3.2 Hypergraphs modeling

A first simple and natural model for hypergraphs was introduced generalizing Erdős-Rényi’s model of random graphs and leading to random uniform hypergraphs. It consists in drawing uniformly at random from the set of all  $m$ -uniform hypergraphs (*i.e.* with hyperedges of fixed cardinality  $m$ ) over a set of  $n$  nodes. Each of the  $\binom{n}{m}$   $m$ -elements subsets of the set of nodes is chosen to be an edge of a random hypergraph with probability  $p$  independently of all other subsets. The number of edges of such a random hypergraph has a binomial distribution with expected value equal to  $\binom{n}{m}p$ . A slightly more general definition of random hypergraphs allows for hyperedges of varying cardinality: each possible subset of nodes of size  $m$  is selected as a hyperedge with probability  $p_m$ ,  $0 \leq p_m \leq 1$ ,  $m = 2, \dots, M$ , independently of all other subsets of nodes of the same and different sizes. However, similarly to Erdős-Rényi, these models are too simple and homogeneous to be used to statistically analyze datasets.

The configuration model for random graphs constitutes a slightly more advanced model; it draws uniformly at random from the set of all graphs over a set of  $n$  nodes with some prescribed degrees sequence. A first generalization to the hypergraph context appears in Ghoshal et al. (2009) focusing on tripartite and 3-uniform hypergraphs, while Chodrow (2020) extends it to a more general hypergraphs setup: given a fixed degree sequence  $\mathbf{d}$  (*i.e.*, the vector whose  $i$ -th element is the degree of node  $i$ ), and a fixed dimension sequence  $\boldsymbol{\delta}$  (*i.e.*, the vector whose  $e$ -th element is the size of hyperedge  $e$ ), and denoted by  $\mathcal{H}_{\mathbf{d}, \boldsymbol{\delta}}$  the space of all hypergraphs with the specified nodes degree and hyperedges size sequences, the configuration model is the uniform distribution on this space. The configuration model is useful to sample (hyper)-graphs with the same nodes degrees (and same hyperedges sizes) as an observed one through shuffling algorithms, and thus is often used as a null model in a statistical perspective. However sampling exactly (and not approximately) from this model is challenging, in particular in the hypergraph case. We refer to Section 4 in Chodrow (2020) for a thorough discussion on this issue.

Hypergraph nodes clustering has recently received some attention. Ghoshdastidar and Dukkipati (2014) introduce a planted partition model for uniform hypergraphs, which is a particular case of a SB model. More precisely, they assume that nodes are clustered into equally-sized groups and two parameters determine intra-groups and inter-groups connec-

tion probabilities, the former always being larger than the latter. They develop a spectral partitioning method and establish its consistency. This result was extended to the non-uniform and weighted sparse (*i.e.* most weights are close to zero) setting in Ghoshdastidar and Dukkipati (2017). We will include these formulations as particular cases of our general model. Introducing hypergraphons, Balasubramanian (2021) extends the hypergraph SB model ideas to a nonparametric setting; this approach is very general, but its least-squares estimator of a hypergraphon model is intractable. Besides, Algorithm 1 in that reference is dedicated to community detection and does not recover general groups. In the same way, the references (Ke et al., 2020; Ahn et al., 2018; Chien et al., 2019) all focus on community detection and do not find clusters that are not communities. In a parallel vein, Turnbull et al. (2021) recently proposed a latent space model for hypergraphs, by generalizing random geometric graphs to hypergraphs, though not designed to capture clustering. A proposal linked to SB models appears in Vazquez (2009), where nodes belong to latent groups and participate in a hyperedge with a probability that depends on their group and that hyperedge.

Modularity is a widely used criterion for clustering entities in the context of interaction data. It is designed to obtain specific clusters, called communities, and characterized by large intra-group and low inter-group connections (exactly as in the above partition model from Ghoshdastidar and Dukkipati, 2014). In the hypergraph context, the definition of modularity is not unique. In particular, Kamiński et al. (2019) introduce a “strict” modularity criterion such that only hyperedges with all their nodes belonging to the same group contribute to an increase in the modularity. Their criterion measures a deviation of the number of these homogeneous hyperedges from a new null model: a configuration-like model for hypergraphs where the average values of the degrees are kept fixed. Further in this direction, Chodrow et al. (2021) introduce a very general degree-corrected hypergraph SB model and propose two new modularity criteria. Similarly to Kamiński et al. (2019), one of these criteria relies on an “all-or-nothing” affinity function that distinguishes only whether a given edge is contained entirely within a single cluster. In this setup, they establish a link between approximate maximum likelihood estimation (MLE) and their modularity criterion. This echoes the work of Newman (2016) in the graph context. It is important to note that the developments in Kamiński et al. (2019); Chodrow et al. (2021) are done in a multisets-hypergraphs context where hyperedges are multisets, *i.e.* nodes are allowed to appear with a certain multiplicity in each hyperedge. The multisets-hypergraphs setup simplifies computational challenges raised by the computation of the modularity and to our knowledge modularity approaches still lack instantiation in the simple hypergraph case. As already argued in Section 4.2.2.1, while both approaches are grounded, they give rise to

different statistical analyses. The choice of which should be used depends on the type of data at hand. Focusing on community detection, random walks approaches have also been used for hypergraph clustering (Swan and Zhan, 2021), as well as low-rank tensor decompositions (Ke et al., 2020). The misclassification ratio for the community detection problem in hypergraphs and its limits have been analyzed in various contexts (see for instance Ahn et al., 2018; Chien et al., 2019; Cole and Zhu, 2020). We mention that a recent approach has proposed to cluster hyperedges (Ng and Murphy, 2021) while our focus in this work is on nodes clustering.

To conclude this Section, we mention that the literature about higher-order interactions often discusses simplicial complexes in parallel with hypergraphs. Borrowing the notation from algebraic topology (Hatcher, 2002), a simplex is generally defined as a set of nodes, and a simplicial complex is a collection of simplices. However the peculiarity of these structures (namely the fact that each subset of a simplex should also be a simplex) puts them out of the scope of the current state of the art (Battiston et al., 2020).

## 4.4 A stochastic block model for hypergraphs

### 4.4.1 Model formulation

Let  $\mathcal{H} = (\mathcal{V}, \mathcal{E})$  denote a binary hypergraph, where  $\mathcal{V} = \{1, \dots, n\}$  is a set of  $n$  nodes and  $\mathcal{E}$  is the set of hyperedges. We indicate by  $M = \max_{e \in \mathcal{E}} |e|$  the largest size of hyperedges in  $\mathcal{E}$  (so that  $M \geq 2$ , with  $M = 2$  for graphs). Let us denote by

$$\begin{aligned} \mathcal{V}^{(m)} &= \{\{i_1, \dots, i_m\} : i_1, \dots, i_m \in \mathcal{V} \text{ and } i_1 \neq \dots \neq i_m\}, \\ \mathcal{E}^{(m)} &= \{\{i_1, \dots, i_m\} \in \mathcal{V}^{(m)} : \{i_1, \dots, i_m\} \in \mathcal{E}\}, \end{aligned}$$

the sets of unordered node tuples and hyperedges of size  $m$  respectively. Obviously it holds that

$$\mathcal{E} = \bigcup_{m=2}^M \mathcal{E}^{(m)} \subseteq \bigcup_{m=2}^M \mathcal{V}^{(m)}.$$

In particular, for each tuple  $\{i_1, \dots, i_m\} \in \mathcal{V}^{(m)}$ , we define the indicator variable

$$Y_{i_1, \dots, i_m} = \mathbb{1}_{\{i_1, \dots, i_m\} \in \mathcal{E}} = \begin{cases} 1 & \text{if } \{i_1, \dots, i_m\} \in \mathcal{E}, \\ 0 & \text{if } \{i_1, \dots, i_m\} \notin \mathcal{E}. \end{cases}$$

We let  $\mathbf{Y} = (Y_{i_1, \dots, i_m})_{\{i_1, \dots, i_m\} \in \mathcal{V}^{(m)}}$  represent a random hypergraph.



Likewise the formulation of the SB model for graphs, we assume that the nodes belong to  $Q$  unobserved groups. Let  $U_1, \dots, U_n$  denote  $n$  independent and identically distributed (i.i.d.) latent random variables having a discrete distribution with  $Q$  support points  $\{1, \dots, Q\}$ ; for each  $q = 1, \dots, Q$ ,  $\pi_q = \mathbb{P}(U_i = q)$  is the prior distribution such that  $\pi_q \geq 0$  and  $\sum_{q=1}^Q \pi_q = 1$ . With a slight abuse of notation, we sometimes write  $U_i = (U_{i1}, \dots, U_{iQ}) \in \{0, 1\}^Q$ , with only one value  $U_{iq}$  equal to 1. We also let  $\mathbf{U} = (U_1, \dots, U_n)$ .

Every  $m$ -tuple of nodes is associated with a latent configuration, simply defined as the set of latent groups these nodes belong to. We denote by

$$\mathcal{Q}^{(m)} = \{Y_{q_1, \dots, q_m} : q_1, \dots, q_m \in \{1, \dots, Q\}\},$$

the set of all possible latent configurations of elements in  $\mathcal{V}^{(m)}$ . Conditionally on the latent variables  $U_i$ , all indicator variables  $Y_{i_1, \dots, i_m}$  are assumed to be independent and follow a Bernoulli distribution whose parameter depends on the latent configuration:

$$Y_{i_1, \dots, i_m} | \{U_{i_1} = q_1, \dots, U_{i_m} = q_m\} \sim \mathcal{B}(B_{q_1, \dots, q_m}^{(m)}) \quad \text{for any } \{i_1, \dots, i_m\} \in \mathcal{V}^{(m)}.$$

Here  $B_{q_1, \dots, q_m}^{(m)}$  denotes the probability that  $m$  unordered nodes with latent configuration  $\{q_1, \dots, q_m\}$  are connected into a hyperedge. Therefore,  $\forall \{i_1, \dots, i_m\} \in \mathcal{V}^{(m)}$ , the following holds:

$$\mathbb{P}(Y_{i_1, \dots, i_m} | U_{i_1} = q_1, \dots, U_{i_m} = q_m) = (B_{q_1, \dots, q_m}^{(m)})^{Y_{i_1, \dots, i_m}} \cdot (1 - B_{q_1, \dots, q_m}^{(m)})^{1 - Y_{i_1, \dots, i_m}}.$$

Note that each  $B^{(m)}$  is a fully symmetric tensor of rank  $m$ , namely

$$B_{q_1, \dots, q_m}^{(m)} = B_{q_{\sigma(1)}, \dots, q_{\sigma(m)}}^{(m)}, \quad \forall q_1, \dots, q_m \text{ and } \forall \sigma \text{ permutation of } \{1, \dots, m\}. \quad (4.1)$$

We let  $\boldsymbol{\theta} = (\pi_q, B_{q_1, \dots, q_m}^{(m)})_{q, m, q_1, \dots, q_m}$  denote the parameter vector and  $\mathbb{P}_{\boldsymbol{\theta}}, \mathbb{E}_{\boldsymbol{\theta}}$  the corresponding probability distribution and expectation, respectively. Moreover, to simplify the notation, we will denote as  $\sum_{\mathcal{V}^{(m)}}$  and  $\sum_{\mathcal{Q}^{(m)}}$  the summations over all possible unordered node tuples and over all possible latent configurations respectively:

$$\sum_{\mathcal{V}^{(m)}} = \sum_{\{i_1, \dots, i_m\} \in \mathcal{V}^{(m)}}, \quad \text{and} \quad \sum_{\mathcal{Q}^{(m)}} = \sum_{\{q_1, \dots, q_m\} \in \mathcal{Q}^{(m)}}.$$

The model parameters of our hypergraph stochastic block (HSB) model are summarized in Table 4.1.

**Lemma 1.** *The number of different parameters in each tensor  $B^{(m)}$  is  $\binom{Q+m-1}{m}$ .*

| Parameter                   | Description  | Range                                    |
|-----------------------------|--|--|
| $\pi_q$                     | Prior probability of latent blocks                                   | $q = 1, \dots, Q$                        |
| $B_{q_1, \dots, q_m}^{(m)}$ | Conditional probability of hyperedges given the latent configuration | $m = 2, \dots, M$<br>$q_j = 1, \dots, Q$ |

**Table 4.1:** Summary of the parameters of the hypergraph stochastic block model

As a consequence, the total number of free parameters is given by

$$(Q - 1) + \sum_{m=2}^M \binom{Q + m - 1}{m}.$$

As better shown in Table 4.2, the number of parameters increases quite rapidly as the values

| $M$ | $Q$ |    |     |     |     |      |
|-----|-----|----|-----|-----|-----|------|
|     | 2   | 3  | 4   | 5   | 6   | 7    |
| 3   | 4   | 10 | 20  | 35  | 56  | 84   |
| 4   | 5   | 15 | 35  | 70  | 126 | 210  |
| 5   | 6   | 21 | 56  | 126 | 252 | 462  |
| 6   | 7   | 28 | 84  | 210 | 462 | 924  |
| 7   | 8   | 36 | 120 | 330 | 792 | 1716 |

**Table 4.2:** Number of parameters of the full hypergraph stochastic block model for given values of  $Q$  (number of latent groups) and  $M$  (largest hyperedge size)

of  $Q$  and  $M$  grow. To significantly reduce the model complexity, we introduce submodels by assuming the equality of some conditional probabilities  $B_{q_1, \dots, q_m}^{(m)}$ . We mention that (Chodrow et al., 2021) have also defined submodels in the context of degree-corrected HSB models. In particular, we consider two ‘‘affiliation’’ submodels given by

$$B_{q_1, \dots, q_m}^{(m)} = \begin{cases} \alpha^{(m)} & \text{if } q_1 = \dots = q_m, \\ \beta^{(m)} & \text{if there exist at least } q_i \neq q_j \end{cases} \quad (\mathbf{Aff-m})$$

and

$$B_{q_1, \dots, q_m}^{(m)} = \begin{cases} \alpha & \text{if } q_1 = \dots = q_m \\ \beta & \text{if there exist at least } q_i \neq q_j \end{cases} \quad \forall m = 2, \dots, M. \quad (\mathbf{Aff})$$

The number of parameters is dropped to  $(Q - 1) + 2(M - 1)$  and to  $(Q - 1) + 2$  under

Assumptions **(Aff-m)** and **(Aff)**, respectively. These submodels reflect the same ideas as in [Kamiński et al. \(2019\)](#); [Chodrow et al. \(2021\)](#) when they consider that only hyperedges whose nodes all belong to the same group should increase the modularities.

**The choice of  $M$ .** It is important to stress that when analyzing a dataset,  $M$  is not necessarily the maximum observed value of the hyperedges sizes but rather a modeling choice. Indeed, take for example a co-authorship dataset with  $n$  authors and only 3 co-authors at most. If nothing prevents 4 persons to be co-authors, then the fact that there are no hyperedges of size 4 gives as much information as if all the possible size-4 hyperedges would be present. In the same way, the amount of information contained in a dataset where all but say 5 possible size-4 hyperedges are present is the same as the amount of information contained in the same dataset but with only 5 occurring size-4 hyperedges. In other words, occurring hyperedges and possible but non-occurring hyperedges carry the same amount of information (0 and 1 values play a similar role). As a consequence,  $M$  should be chosen by the statistician, depending on the characteristics of the dataset at hand and on computational resources (see “Algorithm complexity” below for more on that point). One should keep in mind that on any dataset, choosing  $M > 2$  is already an improvement (in the sense of taking into account more information) with respect to a graph analysis of the data at hand.

**Generalizations.** Our model could allow for self-loops without any important changes (by authorizing  $m = 1$ ). It could also be easily generalized to multiple hypergraphs (with or without self-loops) by putting a (zero-inflated or deflated) Poisson law on the conditional distribution of the hyperedges. More generally, the conditional Bernoulli distribution could be replaced by any parametric distribution to handle weighted hypergraphs (*e.g.* a Poisson or a degree-corrected Poisson as in [Chodrow et al. \(2021\)](#)). The case of multisets-hypergraphs could also be handled and would result in a fastest algorithm (though requiring a distinct implementation, which is not provided in our R package).

#### 4.4.2 Parameter identifiability

In this Section, we first establish generic identifiability of the parameters of a HSB model restricted to simple  $m$ -uniform hypergraphs for any  $m \geq 2$ . Generic identifiability (in a parametric context) means that every parameter  $\theta$ , except possibly for some lying in a subset whose dimension is strictly smaller than the dimension of the full parameter space, uniquely defines the distribution  $\mathbb{P}_\theta$ . In other words, when picking at random (*w.r.t.* Lebesgue measure) a parameter  $\theta$ , this uniquely defines  $\mathbb{P}_\theta$  almost surely (*w.r.t.* Lebesgue measure).

Identifiability is established up to label switching on the node groups, as in any discrete latent variable model. The case  $m = 2$  corresponds to Theorem 2 in [Allman et al. \(2011\)](#). Our proof follows the same ideas, building on a key result by [Kruskal \(1977\)](#), and relying in our case on a sufficient condition for a sequence of non-negative integers to be the degree sequence of a simple  $m$ -uniform hypergraph ([Behrens et al., 2013](#)).

**Theorem 1.** *For any  $m \geq 2$  and  $Q \in \mathbb{N}_{\neq 0}$ , the set  $\theta^{(m)} = (\pi_q, B_{q_1, \dots, q_m}^{(m)})_{1 \leq q \leq Q, 1 \leq q_1 \leq \dots \leq q_m \leq Q}$  of parameter of the HSB model restricted to  $m$ -uniform simple hypergraphs over  $n$  nodes, is generically identifiable, up to label switching on the node groups, for large enough  $n$  (depending only on  $m, Q$ ).*

The case of fixed group proportions (e.g., equal group proportions  $\pi_q = 1/Q$ ) needs special attention. Indeed, our main result does not explicitly characterize the subspace of the parameter space on which identifiability may not be satisfied (we only know that its dimension is less than that of the full parameter space). When restricting to fixed group proportions, we are exactly on a lower dimensional space and may not obtain identifiability without specific care. In the same way, our result does not apply in the affiliation cases (**Aff-m**) and (**Aff**) that correspond to a restriction of the parameter space to a lower-dimensional subspace.

The result stated for  $m$ -uniform hypergraphs is enough to imply a similar one for non-uniform simple hypergraphs, as stated in the following corollary.

**Corollary 2.** *For any  $Q \in \mathbb{N}_{\neq 0}$ , the set  $\theta = (\pi_q, B_{q_1, \dots, q_m}^{(m)})_{1 \leq q \leq Q, 1 \leq q_1 \leq \dots \leq q_m \leq Q, 2 \leq m \leq M}$  of parameters of the HSBM for simple hypergraphs over  $n$  nodes, is generically identifiable, up to label switching on the node groups, for large enough  $n$  (depending only on  $M, Q$ ).*

Our proof of Corollary 2 specifically requires all the  $\pi_q$ 's are distinct (a generic condition, thus not explicitly stated) and does not apply for instance in the restricted case of equal group proportions. In that case, it is not sufficient to identify the parameters for each value of  $m$  separately.

### 4.4.3 Maximum likelihood estimation

On the basis of the model parameters introduced in Section 4.4.1, the distribution of  $\mathbf{U}$  may be expressed as

$$\mathbb{P}_{\theta}(\mathbf{U}) = \prod_{i=1}^n \mathbb{P}_{\theta}(U_i) = \prod_{i=1}^n \prod_{q=1}^Q \pi_q^{U_{iq}} = \prod_{q=1}^Q \pi_q^{\sum_{i=1}^n U_{iq}},$$

where  $\mathbb{P}_\theta(U_i) = \prod_{q=1}^Q \pi_q^{U_{iq}}$  denotes the distribution of latent variable  $U_i$ , and the corresponding logarithmic transformation as

$$\log \mathbb{P}_\theta(\mathbf{U}) = \sum_{q=1}^Q \sum_{i=1}^n U_{iq} \log \pi_q.$$

Moreover, for the conditional distribution of  $\mathbf{Y}$  given  $\mathbf{U}$ , we have

$$\mathbb{P}_\theta(\mathbf{Y}|\mathbf{U}) = \prod_{m=2}^M \prod_{\gamma^{(m)}} \mathbb{P}_\theta(Y_{i_1, \dots, i_m} | U_{i_1}, \dots, U_{i_m}),$$

given the assumption of independence of indicator hyperedges given the latent variables. Again, the corresponding logarithmic transformation assumes the following expression:

$$\begin{aligned} \log \mathbb{P}_\theta(\mathbf{Y}|\mathbf{U}) &= \sum_{m=2}^M \sum_{\gamma^{(m)}} \log \mathbb{P}_\theta(Y_{i_1, \dots, i_m} | U_{i_1}, \dots, U_{i_m}) \\ &= \sum_{m=2}^M \sum_{\gamma^{(m)}} \sum_{\mathcal{Q}^{(m)}} U_{i_1 q_1} \cdots U_{i_m q_m} \log \mathbb{P}_\theta(Y_{i_1, \dots, i_m} | U_{i_1} = q_1, \dots, U_{i_m} = q_m) \\ &= \sum_{m=2}^M \sum_{\gamma^{(m)}} \sum_{\mathcal{Q}^{(m)}} U_{i_1 q_1} \cdots U_{i_m q_m} \log \left[ (B_{q_1, \dots, q_m}^{(m)})^{Y_{i_1, \dots, i_m}} (1 - B_{q_1, \dots, q_m}^{(m)})^{1 - Y_{i_1, \dots, i_m}} \right] \\ &= \sum_{m=2}^M \sum_{\gamma^{(m)}} \sum_{\mathcal{Q}^{(m)}} U_{i_1 q_1} \cdots U_{i_m q_m} \left[ Y_{i_1, \dots, i_m} \log(B_{q_1, \dots, q_m}^{(m)}) + (1 - Y_{i_1, \dots, i_m}) \log(1 - B_{q_1, \dots, q_m}^{(m)}) \right]. \end{aligned}$$

Finally, the (incomplete data) likelihood function may be obtained as

$$\begin{aligned} \mathbb{P}_\theta(\mathbf{Y}) &= \sum_{q_1=1}^Q \cdots \sum_{q_n=1}^Q \mathbb{P}_\theta(U_1 = q_1, \dots, U_n = q_n) \mathbb{P}_\theta(\mathbf{Y} | U_1 = q_1, \dots, U_n = q_n) \quad (4.2) \\ &= \sum_{q_1=1}^Q \cdots \sum_{q_n=1}^Q \left( \prod_{i=1}^n \mathbb{P}_\theta(U_i = q_i) \right) \prod_{m=2}^M \prod_{\gamma^{(m)}} \mathbb{P}_\theta(Y_{i_1, \dots, i_m} | U_{i_1} = q_{i_1}, \dots, U_{i_m} = q_{i_m}) \\ &= \sum_{q_1=1}^Q \cdots \sum_{q_n=1}^Q \left( \prod_{i=1}^n \pi_{q_i} \right) \prod_{m=2}^M \prod_{\gamma^{(m)}} (B_{q_{i_1}, \dots, q_{i_m}}^{(m)})^{Y_{i_1, \dots, i_m}} (1 - B_{q_{i_1}, \dots, q_{i_m}}^{(m)})^{1 - Y_{i_1, \dots, i_m}}. \end{aligned}$$

As it usually happens with latent variable models, the computation of the model likelihood is generally intractable. Indeed, Equation (4.2) involves a summation over all possible  $Q^n$  different latent configurations, which is too computationally heavy, unless  $n$  and  $q$  are

small. As mentioned in Section 1.1.2, latent variable models often rely on Expectation-Maximization (EM) algorithm (Dempster et al., 1977) to solve this problem. Nonetheless, the expectation step of the EM algorithm is typically based on the conditional probability of the latent variable, which is intractable itself in the context of (H)SB models (see *e.g.* Matias and Robin, 2014, for a more detailed explanation of why the EM algorithm is not feasible in these cases). A possible remedy is to rely on variational approximations of EM algorithm (VEM, Jordan et al., 1999).

#### 4.4.3.1 Variational Expectation-Maximization algorithm

In order to illustrate the implementation of the VEM algorithm for the HSB model, we first have to introduce the complete data log-likelihood function:

$$\begin{aligned} \ell^*(\boldsymbol{\theta}) &= \log \mathbb{P}_{\boldsymbol{\theta}}(\mathbf{Y}, \mathbf{U}) = \log \mathbb{P}_{\boldsymbol{\theta}}(\mathbf{U}) + \log \mathbb{P}_{\boldsymbol{\theta}}(\mathbf{Y}|\mathbf{U}) \\ &= \sum_{q=1}^Q \sum_{i=1}^n U_{iq} \log \pi_q \\ &\quad + \sum_{m=2}^M \sum_{\mathcal{V}^{(m)}} \sum_{Q^{(m)}} U_{i_1 q_1} \cdots U_{i_m q_m} [Y_{i_1 \dots i_m} \log(B_{q_1, \dots, q_m}^{(m)}) + (1 - Y_{i_1 \dots i_m}) \log(1 - B_{q_1, \dots, q_m}^{(m)})]. \end{aligned} \tag{4.3}$$

The variational approach follows the same iterative two-steps structure as in the EM algorithm; the core idea is to replace the intractable posterior distribution  $\mathbb{P}_{\boldsymbol{\theta}}(\mathbf{U}|\mathbf{Y})$  by the best approximation (with respect to Kullback-Leibler divergence) in a class of simpler (often factorized) distributions. We thus introduce the class of factorized probability distributions  $\mathbb{Q}_{\tau}$  over  $\mathbf{U} = (U_1, \dots, U_n)$  given by

$$\mathbb{Q}_{\tau}(\mathbf{U}) = \prod_{i=1}^n \mathbb{Q}_{\tau}(U_i) = \prod_{i=1}^n \prod_{q=1}^Q \tau_{iq}^{U_{iq}},$$

with the variational parameter  $\tau_{iq} = \mathbb{Q}_{\tau}(U_i = q) \in [0, 1]$  and  $\sum_{q=1}^Q \tau_{iq} = 1$ , for any  $i = 1, \dots, n$  and  $q = 1, \dots, Q$ . Let us denote by  $\mathbb{E}_{\mathbb{Q}_{\tau}}$  the expectation under distribution  $\mathbb{Q}_{\tau}$  and by  $\mathcal{H}(\mathbb{Q}_{\tau}) = \mathbb{E}_{\mathbb{Q}_{\tau}}[-\log \mathbb{Q}_{\tau}(\mathbf{U})]$  the entropy (Shannon, 1948) of  $\mathbb{Q}_{\tau}$ .

We define the evidence lower bound (ELBO) as follows:

$$\begin{aligned}
 \mathcal{J}(\boldsymbol{\theta}, \tau) &= \mathbb{E}_{\mathbb{Q}_\tau}[\log \mathbb{P}_\theta(\mathbf{Y}, \mathbf{U})] + \mathcal{H}(\mathbb{Q}_\tau) \\
 &= \mathbb{E}_{\mathbb{Q}_\tau}[\log \mathbb{P}_\theta(\mathbf{Y}, \mathbf{U})] - \mathbb{E}_{\mathbb{Q}_\tau}[\log \mathbb{Q}_\tau(\mathbf{U})] \\
 &= \sum_{q=1}^Q \sum_{i=1}^n \tau_{iq} \log \frac{\pi_q}{\tau_{iq}} \\
 &\quad + \sum_{m=2}^M \sum_{\mathcal{Q}^{(m)}} \sum_{\mathcal{Y}^{(m)}} \tau_{i_1 q_1} \cdots \tau_{i_m q_m} \left[ Y_{i_1, \dots, i_m} \log(B_{q_1, \dots, q_m}^{(m)}) + (1 - Y_{i_1, \dots, i_m}) \log(1 - B_{q_1, \dots, q_m}^{(m)}) \right].
 \end{aligned} \tag{4.4}$$

Proposition 1 shows that  $\mathcal{J}(\boldsymbol{\theta}, \tau)$  is a lower bound of the model log-likelihood  $\log \mathbb{P}_\theta(\mathbf{Y})$ .

**Proposition 1.** *The function  $\mathcal{J}(\boldsymbol{\theta}, \tau)$ , as defined in Equation (4.4), satisfies*

$$\mathcal{J}(\boldsymbol{\theta}, \tau) = \log \mathbb{P}_\theta(\mathcal{E}) - KL(\mathbb{Q}_\tau(\mathbf{U}) || \mathbb{P}_\theta(\mathbf{U} | \mathcal{E})), \tag{4.5}$$

where  $KL()$  denotes the Kullback-Leibler divergence.

The VEM algorithm alternates the following two steps until a suitable convergence criterion is satisfied:

- **VE-Step:** maximizes  $\mathcal{J}(\boldsymbol{\theta}, \tau)$  with respect to  $\tau$

$$\hat{\tau}^{(t)} = \arg \max_{\tau} \mathcal{J}(\boldsymbol{\theta}^{(t-1)}, \tau); \quad \text{s.t.} \quad \sum_{q=1}^Q \tau_{iq} = 1 \quad \forall i = 1, \dots, n \tag{4.6}$$

this is equivalent to minimizing the Kullback-Leibler divergence term in (4.5), and thus finding the “best” approximation of the conditional distribution  $\mathbb{P}_\theta(\mathbf{U} | \mathbf{Y})$ ;

- **M-Step:** maximizes  $\mathcal{J}(\boldsymbol{\theta}, \tau)$  with respect to  $\boldsymbol{\theta}$

$$\hat{\boldsymbol{\theta}}^{(t)} = \arg \max_{\boldsymbol{\theta}} \mathcal{J}(\boldsymbol{\theta}, \tau^{(t-1)}), \quad \text{s.t.} \quad \sum_{q=1}^Q \pi_q = 1, \tag{4.7}$$

thus updating the value of the model parameters  $\pi_q$  and  $B_{q_1, \dots, q_m}^{(m)}$ .

In the following we provide the solutions of the two maximization problems in Equations (4.6) and (4.7).

**Proposition 2** (VE-Step). *Given the current model parameters  $(\pi_q, B_{q_1, \dots, q_m}^{(m)})_{q, m, q_1, \dots, q_m}$  at any iteration of the VEM algorithm, the corresponding optimal values of the variational*

parameters  $(\widehat{\tau}_{iq})_{i,q}$  defined in Equation (4.6) should satisfy the following fixed point equation:

$$\begin{aligned} \log \widehat{\tau}_{iq} = \log \pi_q + \sum_{m=1}^{M-1} \sum_{\mathcal{Q}^{(m)}} \sum_{\substack{\{i_1, \dots, i_m\} \in \mathcal{V}^{(m)} \\ \text{s.t. } \{i, i_1, \dots, i_m\} \in \mathcal{V}^{(m+1)}}} \widehat{\tau}_{i_1 q_1} \cdots \widehat{\tau}_{i_m q_m} \\ \times \left[ Y_{ii_1 \dots i_m} \log(B_{qq_1 \dots q_m}^{(m+1)}) + (1 - Y_{ii_1 \dots i_m}) \log(1 - B_{qq_1 \dots q_m}^{(m+1)}) \right] + c_i, \end{aligned} \quad (4.8)$$

for any  $1 \leq i \leq n$  and  $1 \leq q \leq Q$  and where  $c_i$  are normalizing constants such that  $\sum_q \widehat{\tau}_{iq} = 1$ .

**Proposition 3** (M-Step). *Given the current variational parameters  $(\tau_{iq})_{i,q}$  at any iteration of the VEM algorithm, the corresponding optimal values of the model parameters  $(\widehat{\pi}_q, \widehat{B}_{q_1 \dots q_m}^{(m)})_{q,m,q_1, \dots, q_m}$  defined in Equation (4.7) are expressed as:*

$$\widehat{\pi}_q = \frac{1}{n} \sum_{i=1}^n \tau_{iq} \quad \text{and} \quad \widehat{B}_{q_1 \dots q_m}^{(m)} = \frac{\sum_{\mathcal{V}^{(m)}} \tau_{i_1 q_1} \cdots \tau_{i_m q_m} Y_{i_1 \dots i_m}}{\sum_{\mathcal{V}^{(m)}} \tau_{i_1 q_1} \cdots \tau_{i_m q_m}}.$$

We now express the solutions of the M-Step under the submodels given by **(Aff-m)** and **(Aff)**. Note that the VE-Step is unchanged under these settings.

**Proposition 4** (M-Step, affiliation setup). *In the particular affiliations submodels given by **(Aff-m)** and **(Aff)** respectively, given variational parameters  $(\tau_{iq})_{i,q}$  at any iteration of the VEM algorithm, the corresponding optimal values of  $(\widehat{\alpha}^{(m)}, \widehat{\beta}^{(m)})_m$  and  $\widehat{\alpha}, \widehat{\beta}$  maximizing  $\mathcal{J}$  as in Equation (4.7) are now expressed as:*

- under Assumption **(Aff-m)**,

$$\begin{aligned} \widehat{\alpha}^{(m)} &= \frac{\sum_{q=1}^Q \sum_{\mathcal{V}^{(m)}} \tau_{i_1 q} \cdots \tau_{i_m q} Y_{i_1 \dots i_m}}{\sum_{q=1}^Q \sum_{\mathcal{V}^{(m)}} \tau_{i_1 q} \cdots \tau_{i_m q}}, \\ \widehat{\beta}^{(m)} &= \frac{\sum_{\substack{\{q_1, \dots, q_m\} \in \mathcal{Q}^{(m)} \\ |\{q_1, \dots, q_m\}| \geq 2}} \sum_{\mathcal{V}^{(m)}} \tau_{i_1 q_1} \cdots \tau_{i_m q_m} Y_{i_1 \dots i_m}}{\sum_{\substack{\{q_1, \dots, q_m\} \in \mathcal{Q}^{(m)} \\ |\{q_1, \dots, q_m\}| \geq 2}} \sum_{\mathcal{V}^{(m)}} \tau_{i_1 q_1} \cdots \tau_{i_m q_m}}; \end{aligned}$$



- under Assumption **(Aff)**,

$$\hat{\alpha} = \frac{\sum_{m=2}^M \sum_{q=1}^Q \sum_{\mathcal{V}^{(m)}} \tau_{i_1 q} \cdots \tau_{i_m q} Y_{i_1 \dots i_m}}{\sum_{m=2}^M \sum_{q=1}^Q \sum_{\mathcal{V}^{(m)}} \tau_{i_1 q} \cdots \tau_{i_m q}},$$

$$\hat{\beta} = \frac{\sum_{m=2}^M \sum_{\substack{\{q_1, \dots, q_m\} \in \mathcal{Q}^{(m)} \\ |\{q_1, \dots, q_m\}| \geq 2}} \sum_{\mathcal{V}^{(m)}} \tau_{i_1 q_1} \cdots \tau_{i_m q_m} Y_{i_1 \dots i_m}}{\sum_{m=2}^M \sum_{\substack{\{q_1, \dots, q_m\} \in \mathcal{Q}^{(m)} \\ |\{q_1, \dots, q_m\}| \geq 2}} \sum_{\mathcal{V}^{(m)}} \tau_{i_1 q_1} \cdots \tau_{i_m q_m}}.$$

**Algorithm complexity.** The complexity of our algorithm is of the order  $O(nQ^M \binom{n}{M})$ , which is rather prohibitive for large datasets when  $M$  becomes large. We recall here that the value of  $M$  must be chosen, and is not necessarily the largest observed hyperedge size (see paragraph “The choice of  $M$ ” above). Thus, for large datasets, we recommend limiting the analysis to  $M = 3$  or 4.

#### 4.4.3.2 Fixed point

From Proposition 2, the  $\tau_i$ 's in the VE-Step are obtained using a fixed point algorithm. In practice at iteration  $h$  of the VEM algorithm, starting from the previous values of the variational and the model parameters  $\tau_{iq}^{(h-1)}$  and  $\theta^{(h-1)}$ , respectively, we iterate over some index  $u$  the computation given by (4.8) and obtain a sequence of values  $\tau_{iq}^{(t,u)}$ . We stop these iterations whenever we reach a maximum number of fixed point iterations ( $u > U_{\max}$ ) or the variational parameters converged ( $\max_{iq} |\tau_{iq}^{(h,u-1)} - \tau_{iq}^{(h,u)}| \leq \varepsilon$ ). Although in all the situations we experienced, the algorithm converged in a reasonable number of iterations, we have no guarantee about existence nor uniqueness of a solution to (4.8).

#### 4.4.3.3 Algorithm initialization and stopping criteria

As it typically happens, the proposed VEM algorithm requires a set of starting values to initialize the first iteration. We choose to start the algorithm with its  $M$ -step, hence providing an initial value for  $\tau$  instead of  $\theta$ . This way, we take advantage of smart initialization strategies based on a preliminary clustering of the nodes. Specifically, the following different approaches were tested and compared (see Appendix E for a more detailed explanation of the involved methods):

1. *random*: this naive method simply draws each  $(\tau_{iq})_{1 \leq q \leq Q}$  uniformly in  $(0, 1)$  for every node  $i$  and normalize the vector  $\tau_i$ ; a similar approach is to draw  $\tau_i$  from a Dirichlet

distribution with parameter vector  $(1, 1, \dots, 1)$ , to obtain a uniform distribution on the simplex.

2. *spectral clustering*: it is one of the most popular modern clustering algorithm. It computes a hypergraph Laplacian matrix and constructs the column matrix  $X$  of its leading  $Q$  orthonormal eigenvectors. The rows of  $X$  are normalized to have unit norm, and the  $k$ -means algorithm (Forgy, 1965; MacQueen, 1967) is performed on these normalized rows. The starting value for  $\tau$  is then defined by taking  $\tau_{iq} = 1$  if spectral clustering assigns node  $i$  to group  $q$ , and  $\tau_{iq} = 0$  otherwise.
3. *“soft” spectral clustering*: this approach performs spectral clustering as described at the previous point, but a soft  $k$ -means algorithm (Suganya and Shanthi, 2012) is applied on the normalized rows of  $X$ . Then  $\tau_{iq}$  is defined as the posterior probability for node  $i$  to belong to cluster  $q$ .
4. *graph-component absolute spectral clustering*: we restrict our attention to edges in the hypergraph ( $m = 2$ ) and the corresponding adjacency matrix. We then perform the absolute spectral clustering (Rohe et al., 2011) on this adjacency matrix. This initialization does not use the whole information from the hypergraph (hyperedges of size  $m \geq 3$  are not used). Nonetheless, absolute spectral clustering is believed to be superior to spectral clustering as it captures disassortative groups.

An other important aspect is how to check for the algorithm convergence. Here we consider two of the most common criteria, based on the relative difference in terms of the ELBO  $\mathcal{J}$  of two consecutive steps and on the difference between the corresponding parameter vectors. More precisely, let us denote by  $\boldsymbol{\theta}^{(h)} = (\boldsymbol{\theta}_s^{(h)})_s$  the sequence of model parameter vectors estimated at the  $t$ -th iteration of the M-Step. Then, according to the two criteria, the algorithm is stopped when

$$\frac{|\mathcal{J}(\boldsymbol{\theta}^{(h-1)}) - \mathcal{J}(\boldsymbol{\theta}^{(h)})|}{|\mathcal{J}(\boldsymbol{\theta}^{(h)})|} \leq \varepsilon$$

and when

$$\max_s |\boldsymbol{\theta}_s^{(h-1)} - \boldsymbol{\theta}_s^{(h)}| \leq \varepsilon,$$

respectively. Here  $\varepsilon$  is a suitable tolerance level.

However, experimenting with the above conditions, the algorithm sometimes stops when the VE-Step still requires a few iterations to reach a fixed point. In these cases, carrying on with the VEM iterations generally leads to higher values of the ELBO function, and hence

to better estimates. Therefore we impose that the fixed point in the VE-Step is reached at its first iteration:

$$\max_{i,q} |\tau_{iq}^{(t,0)} - \tau_{iq}^{(t,1)}| \leq \varepsilon$$

This prevents possible convergence to some local maxima of  $\mathcal{J}$ .

Obviously, a proper check on convergence needs to rely on all conditions above. Finally, when at least one of these conditions is not fulfilled, we stop the algorithm if a maximum number of iterations has been reached:  $t > T_{\max}$ .

#### 4.4.4 Model selection

Model selection on the number of latent groups  $Q$  relies on the Integrated Classification Likelihood (ICL, [Biernacki et al., 2000](#)) criterion. Let  $\hat{\boldsymbol{\theta}}$  and  $(\hat{\tau}_i)_i$  denote the estimated parameters obtained at the end of the VEM algorithm and let  $\hat{U}_i = \arg \max_q \hat{\tau}_{iq}$  denote the estimated group for node  $i$ . Then, for any number  $Q \geq 1$ , the ICL is defined for the full model and for **(Aff-m)**, **(Aff)** submodels as

$$\text{ICL}_{\text{full}}(Q) = \log \mathbb{P}_{\hat{\boldsymbol{\theta}}}(\mathbf{Y}, \hat{\mathbf{U}}) - \frac{1}{2}(Q-1) \log n - \frac{1}{2} \sum_{m=2}^M \binom{Q+m-1}{m} \log \binom{n}{m} \quad (4.9)$$

$$\text{ICL}_{\text{aff-m}}(Q) = \log \mathbb{P}_{\hat{\boldsymbol{\theta}}}(\mathbf{Y}, \hat{\mathbf{U}}) - \frac{1}{2}(Q-1) \log n - 2(M-1) \log \binom{n}{m} \quad (4.10)$$

$$\text{ICL}_{\text{aff}}(Q) = \log \mathbb{P}_{\hat{\boldsymbol{\theta}}}(\mathbf{Y}, \hat{\mathbf{U}}) - \frac{1}{2}(Q-1) \log n - \log \binom{n}{m}, \quad (4.11)$$

respectively. In each expression the first penalization term accounts for the prior probabilities  $(\pi_q)_q$  and for the  $n$  latent variables  $U_1, \dots, U_n$ ; the second penalization term, instead, refers to  $(B_{q_1, \dots, q_m}^{(m)})_{m, q_1, \dots, q_m}$  and to the  $\binom{n}{m}$  different indicator variables  $Y_{i_1, \dots, i_m}$ . The number of latent groups is then selected with  $\hat{Q} = \arg \max_Q \text{ICL}(Q)$ . Note that [Ghoshdastidar and Dukkipati \(2017\)](#) propose to select the number of groups by looking for the spectral gap.

## 4.5 Simulation study

In this Section we conduct a simulation study to assess the performance of the proposed VEM algorithm for the HSB model. In the following, we illustrate the simulation scheme and summarize the main results.

### 4.5.1 Clustering performances

Hypergraphs are simulated from the HSB model, considering  $Q = 2$  latent groups with prior probabilities equal to 0.6 and 0.4, respectively. The largest size  $M$  of hyperedges is set to 3, and 4 different values are examined for the number of nodes:  $n = 50, 100, 150,$  and  $200$ . A simplified latent structure, according to the (**Aff**) submodel is assumed, and various scenarios, corresponding to different possible real-world situations, are analyzed:

- A. *Communities*: in this scenario, we focus on community detection and consider the case of high intra-groups and low inter-groups connection probabilities. We thus set  $\alpha = 0.7 > \beta = 0.3$ ;
- B. *Disassortative*: in this scenario, we focus on disassortative behavior and consider the case of low intra-groups and high inter-groups connection probabilities. We thus set  $\alpha = 0.3 < \beta = 0.7$ ;
- C. *Erdős-Rényi-like*: in this scenario, we focus on the difficult case of very similar intra-groups and inter-groups connection probabilities. We thus set  $\alpha = 0.25$  very close to  $\beta = 0.35$ .

For each scenario and each value  $n$  of the number of nodes, 10 different datasets are simulated. We consider estimation under the full HSB model formulation with our VEM algorithm and rely on soft spectral clustering initialization only. The performance of the proposed VEM algorithm is assessed in terms of both recovery of the correct clustering and estimation of the original parameters.

For the correct classification, the Adjusted Rand Index (ARI, [Hubert and Arabie, 1985](#)) is considered, measuring the similarity between the correct node clustering and the estimated one. This index is always smaller than or equal to 1 (two identical clusterings have an ARI exactly equal to 1), and it can assume negative values when the agreement between the two clusterings is less than what is expected from a random result. Table 4.3 reports,

| $n$ | Scenario A | Scenario B | Scenario C |
|-----|------------|------------|------------|
| 50  | 1.00       | 1.00       | 0.50       |
| 100 | 1.00       | 1.00       | 0.90       |
| 150 | 1.00       | 1.00       | 1.00       |
| 200 | 1.00       | 1.00       | 1.00       |

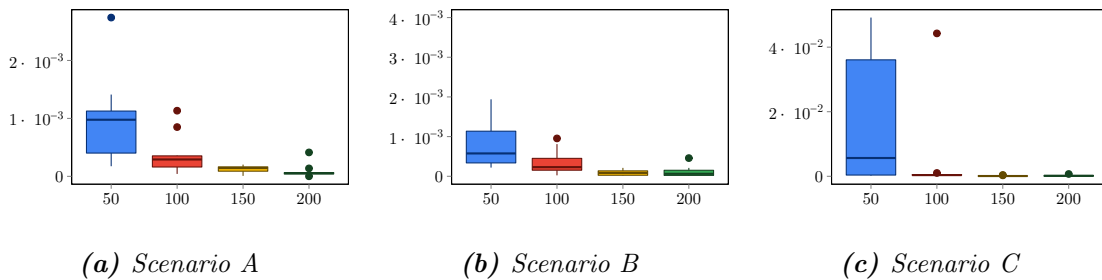
**Table 4.3:** Adjusted Rand Index for different scenarios and number of nodes. Each value is obtained as the average over 10 simulated datasets

for each setting, the average value of the ARI over the 10 simulated datasets. Considering scenarios A and B, the results are highly satisfactory, all values being equal to 1. The VEM algorithm perfectly recovers the correct clusters in all cases, hence showing an optimal performance in detecting communities as well as disassortative behaviors. Scenario C proves to be a more complex setting for clustering, especially when combined with a small number of nodes. Considering this setting, the proposed approach sometimes fails to recover the optimal clustering. This behavior is particularly evident in the case with  $n = 50$  nodes, where the average ARI is rather low (0.5): the correct clusters are obtained for only half the hypergraphs. In that scenario, the performance improves with the increase of the number of nodes.

We also inspect the estimation of model parameters by computing the Mean Squared Error (MSE) between the true parameters and the estimated ones, for both the prior probabilities  $\pi_q$ , and the probabilities of hyperedge occurrence  $B_{q_1, \dots, q_m}^{(m)}$ . More specifically, we computed an aggregated MSE over all the components of  $\theta$ , defined as

$$MSE = \frac{1}{10} \sum_{i=1}^{10} \left\{ (\hat{\pi}_1^i - \pi_1)^2 + \sum_{m=2}^M \sum_{q_1, \dots, q_m} (\hat{B}_{q_1, \dots, q_m}^{(m), i} - B_{q_1, \dots, q_m}^{(m)})^2 \right\}.$$

where  $\hat{\theta}^i = (\hat{\pi}_1^i, \{\hat{B}_{q_1, \dots, q_m}^{(m), i}\}_{m, q_1, \dots, q_m})$  is the parameter estimated on the  $i$ -th dataset by the full model.



**Figure 4.3:** Mean Squared Error between true and estimated model parameters for different scenarios and number of nodes

The corresponding results are summarized through the boxplots in Figure 4.3. All values are rather small, showing that the model parameters are generally estimated with a high degree of accuracy. In particular, scenarios A and B provide the best results, with values of the MSE that are always lower than 0.5%. On the other hand, scenario C confirms to be the most difficult from the estimation perspective, showing the highest MSE for each

value of  $n$  (up to 8%). This analysis also allows us to better outline the behavior of the VEM algorithm for different values of  $n$ ; in particular, in each scenario, the parameters estimation becomes more accurate as the number of nodes increases. Estimates obtained assuming the submodel (**Aff**) formulation do not present any significant difference.

#### 4.5.2 Performance of model selection

In this Section we assess the performance of ICL as a model selection criterion. To this aim we simulate 50 hypergraphs from the HSB model with  $Q = 3$  latent states and assuming the simplified (**Aff**) formulation for the latent structure. Two different values are tested for the number of nodes,  $n = 100$  and  $n = 200$ , while the largest size  $M$  of hyperedges is set equal to 3 in both cases. The simulated data is then fitted with the HSB model with a number of latent states ranging from 1 to 5.

| $Q$ | $n = 100$  |                  | $n = 200$  |                  |
|-----|------------|------------------|------------|------------------|
|     | Percentage | ARI for 3 groups | Percentage | ARI for 3 groups |
| 2   | 0%         | -                | 2%         | 0.55             |
| 3   | 68%        | 1.00             | 90%        | 1.00             |
| 4   | 22%        | 0.57             | 6%         | 0.60             |
| 5   | 10%        | 0.58             | 2%         | 0.61             |

**Table 4.4:** Frequency of the selected number of groups and average Adjusted Rand Index of the classification obtained with  $Q = 3$  depending on the selected number of groups. Model selection is carried out by means of the ICL criterion. Results are computed over 50 sample for each value of  $n$

In Table 4.4 we show the frequency of the selected number of groups. Results are highly satisfactory: the correct model is selected in 68% of cases for  $n = 100$  and in 90% of cases for  $n = 200$ . We also compute the value of ARI of the classification obtained with 3 clusters depending on the selected number of latent groups. This value is always equal to 1 when the correct model is recovered, thus confirming the optimal behavior of our HSB model stated in Section 4.5.1. On the contrary, in cases where an incorrect number of groups is selected, values of ARI are quite low (around or smaller than 0.60). This behavior clarifies that the estimation through the VEM algorithm is responsible for the bad recovery more than the selection criterion. It is again confirmed that better results are obtained for higher values of  $n$ .

## 4.6 Analysis of a co-authorship dataset

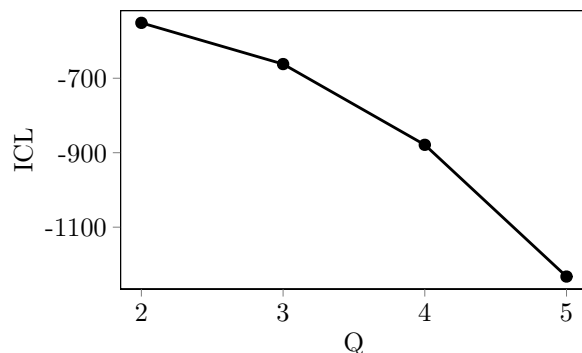
We analyze a co-authorship dataset available at the following link: <http://vlado.fmf.uni-lj.si/pub/networks/data/2mode/Sandi/Sandi.htm>. The dataset is extracted from the bibliography of a book (“Product Graphs: Structure and recognition” by Imrich and Klavžar) and is given as a bipartite author/paper graph.

### 4.6.1 Dataset description

Following (Estrada and Rodríguez-Velázquez, 2006), we construct the hypergraph in which nodes are authors and hyperedges link the authors of a same paper. The original dataset has 274 papers and 314 authors, with 1 paper having 6 authors and 1 paper having 5 authors. We decide to consider  $M = 4$ , discarding these 2 papers with 5 or more authors. Thereafter, we look at the largest connected component of the resulting graph; it results in 79 authors (nodes) and 76 papers (hyperedges, 68.5% of which have size 2, while 29% have size 3 and 2.5% have size 4).

### 4.6.2 Analysis with the HyperSBM package

We perform an analysis of this dataset with our HyperSBM package, estimating the HSB model with  $Q$  ranging from 2 to 5, and with two different (random and soft spectral clustering) initializations. The results are robust to different tries. As shown in Figure 4.4, the ICL criterion selects  $Q = 2$  latent groups.



**Figure 4.4:** Integrated Classification Likelihood index resulting from fitting the HSB model to the co-authorship dataset with number of latent groups ranging from 2 to 5

We obtain a small group with only 8 authors (the remaining 71 authors being in the second group). Table 4.5 presents the distribution of the number of distinct co-authors per

author. Among the 8 authors of the first group, 6 of them have the highest number of distinct co-authors (and the remaining 2 have 4 distinct co-authors each).

| Number of distinct co-authors | 1  | 2  | 3  | 4 | 5 | 6 | 7 | 8 | 10 | 11 | 12 |
|-------------------------------|----|----|----|---|---|---|---|---|----|----|----|
| Count                         | 23 | 27 | 13 | 6 | 2 | 2 | 1 | 1 | 2  | 1  | 1  |

**Table 4.5:** *Distribution of the number of distinct co-authors per author. The first group contains the 6 authors having the largest number of distinct co-authors (between 7 and 12) plus 2 authors with 4 co-authors each*

We also look at the degree distribution of the authors, given in Table 4.6. This corresponds to the distribution of the number of co-authored published papers per author. We observe that 5 of the 8 authors from our first group are the ones that co-published the most, the three others having also high degree (one of degree 5 and two of degree 4). Thus, our first group is made of authors (among) the most collaborative ones, which are also (among) the most prolific ones.

| Author degree | 1  | 2  | 3 | 4 | 5 | 6 | 7 | 8 | 10 | 13 |
|---------------|----|----|---|---|---|---|---|---|----|----|
| Count         | 44 | 14 | 6 | 6 | 4 | 1 | 1 | 1 | 1  | 1  |

**Table 4.6:** *Degree distribution of authors in the bipartite graph. Our first group contains the 5 most collaborating authors, one of the sixth, plus 2 authors with degree equal to 4*

Inspecting more closely the variational parameters  $\tau_{iq}$  for all the nodes, we find that a total of 4 nodes could be considered as ambiguously classified, while all other nodes had posterior probabilities to belong to one of the two groups larger than 0.8. More precisely, in the first small group, 2 nodes have posterior probabilities to belong to that group equal to 0.54 and 0.63, respectively; while in the second large group, 2 nodes have posterior probabilities to belong to that group equal to 0.56 and 0.72, respectively. We notice that the 2 authors in the first group that have the smallest number of co-authors (namely 4) and the smallest number of degrees (also 4) are the ones that are ambiguously clustered in this group. While the 2 other authors ambiguously clustered in the second large group have a number of co-authors of 6 and 4, respectively, and both a degree of 4. This reinforces the conclusion that on this dataset, **HyperSBM** has grouped apart the authors which are both the most collaborative and the most prolific ones.

Neither the first nor the second group inferred by the **HyperSBM** are communities. Indeed, we obtain the following estimated values from the size-2 hyperedges (expressed as percentages):  $\hat{B}_{11}^{(2)} \simeq 4.2\%$  is of the same order as  $\hat{B}_{12}^{(2)} \simeq 5.1\%$  while  $\hat{B}_{22}^{(2)} \simeq 0.8\%$  is around five times smaller. This means that the first group contains authors that have published



with authors from the two groups, while the second group is made of authors who have less co-authored papers with people of their own group.

Looking now at size-3 hyperedges, we get that  $\hat{B}_{111}^{(3)} \simeq 2 \cdot 10^{-4}$ ,  $\hat{B}_{112}^{(3)} \simeq 18 \cdot 10^{-4}$ ,  $\hat{B}_{122}^{(3)} \simeq 7 \cdot 10^{-4}$ , and  $\hat{B}_{222}^{(3)} \simeq 0.6 \cdot 10^{-4}$ . The most important estimated frequency is  $\hat{B}_{112}^{(3)}$  that concerns 2 authors of the small first group co-authoring a paper with one author of the large second group. The second most important estimated frequency is  $\hat{B}_{122}^{(3)}$  and is obtained for one author from small first group co-authoring a paper with two authors of the large second group. The remaining frequencies of size-3 hyperedges are negligible. This characterizes further the first groups as being composed by authors that do co-author with their own group as well as with authors from the second one.

Finally, looking now at size-4 hyperedges, the only non negligible estimated frequency is obtained for  $\hat{B}_{1222}^{(4)} \simeq 4 \cdot 10^{-6}$ . We note here that the quantities  $B^{(3)}$ 's and  $B^{(4)}$ 's are intrinsically on different scales, as are the quantities  $B^{(2)}$ 's and  $B^{(3)}$ 's. So again, authors from group one co-authored with the others authors. (Note that the first group is not large enough for a  $B^{(4)}$  frequency with at least 2 authors in that group 1 to be non negligible).

### 4.6.3 Comparison with Hypergraph Spectral Clustering

We first compare our approach with the spectral clustering algorithm proposed in [Ghoshdastidar and Dukkipati \(2017\)](#). Let us recall that spectral clustering does not come with a statistical criterion to select the number of groups; we look at the spectral gap, that indicate the presence of 15 groups, but the result is not clear.

Looking at the partition obtained imposing  $Q = 2$ , spectral clustering outputs two groups with sizes 24 and 55, respectively. The smaller group contains the only author with 12 co-authors and the remaining authors have a number of co-authors ranging from 1 to 4. The larger group has a distribution of the number of co-authors ranging from 1 to 11. Moreover, the smaller group contains authors with small degree in the bipartite graph, i.e having few co-published papers (all but one author have degrees less 4 and a last author has degree 7), while the second large group contains the 3 authors with largest degree, the rest of the authors having degrees ranging from 1 to 6. Thus, these groups are neither characterized by the number of co-authors nor by their degrees in the bipartite graph.

Indeed, in our case the best clusters are not communities and their sizes are very different, while we recall that spectral clustering tends to: *(i)* extract communities; *(ii)* favor groups of similar size.

#### 4.6.4 Comparison with a bipartite SB model

Finally, we also analyze the same dataset as a bipartite graph of authors/papers with the R package `SBM` through the function `estimateBipartiteSBM` (Chiquet et al., 2022). This method infers a latent block model (that in fact corresponds to a SB model for bipartite graphs) and automatically selects a number of groups on both parts (authors and papers). Let us underline here that while the bipartite stochastic block model can be written as a particular case of a HSB model, the converse is not true (See Appendix G).

The bipartite SB model also selects two groups of authors (and one group of papers). There is a small group with 4 authors, entirely contained in our first small group obtained with the `HyperSBM` package; it corresponds to authors that have the highest degree in the bipartite graph and the highest number of co-authors. So, the bipartite implementation outputs a very small group of the most prolific and the most collaborative authors in this dataset.

Here, two nodes may be considered as ambiguously classified: one node from the first small group has posterior probability to belong to that group equal to 0.73, while one node from the second large group has posterior belonging probability equal to 0.67. These two nodes are not ambiguously classified by the `HyperSBM` package and both appears in our first small group.

It is interesting to compare the situation of three particular authors here. Author with index 48 has 7 co-authors (sixth highest value) and 6 co-authored papers (fifth highest value). It is outside the small group with the bipartite approach model (posterior probability  $1 - 0.67 = 0.33$  to belong to that group); on the contrary `HyperSBM` clusters it unambiguously in the first small group. Similarly, author with index 27 has 12 coauthors (highest value) and only 7 co-authored papers (fourth highest value). This node was ambiguously classified by the bipartite method in the first small group (posterior probability 0.73 only); while `HyperSBM` clusters it unambiguously in the first small group. Now, conversely, author with index 35 has 8 co-authors (the 6th highest) and 5 co-authored papers (also the 5th highest). This author is unambiguously clustered from the two methods; but while `HyperSBM` puts it in the first small graph, the bipartite approach excludes it from that group. The examination of these 3 particular tangent cases seems to show that on this dataset, the bipartite-based method was more sensible to authors's degrees in the bipartite graph while `HyperSBM` paid more attention to the sizes of the hyperedges (*i.e.* number of co-authors) an author was involved in.

Finally, we also compute the parameters values  $B_{q_1, \dots, q_m}^{(m)}$  obtained with the groups estimated by the bipartite SB model. Considering  $m = 2$ , we obtain  $\hat{B}_{11}^{(2)} \simeq 16,6\%$  (to

be compared with 4.2% in `HyperSBM`), while  $\hat{B}_{12}^{(2)} \simeq 7\%$  and  $\hat{B}_{22}^{(2)} \simeq 1\%$  (more similar to the results of `HyperSBM`, which are 5.1% and 0.8%, respectively). In this case, the first group of authors behaves differently with respect to intra-group connections compared to outer-group connections.

As a conclusion, we see that while the outputs of BSB and HSB models may seem close on this specific dataset, they are nonetheless different. On the other hand, and still on this specific dataset, the spectral clustering approach outputs results that are completely different from those of `HyperSBM`.

## 4.7 Conclusions

Despite the broad variety of models developed for networks, modern applications in many fields highlight the need to account for high-order interactions, in order to include the information deriving from groups of more than two nodes. In this chapter, the notions of hypergraphs and hyperedges are reviewed, generalizing the concepts of graphs and edges, respectively, and providing the most accurate formalization for high-order interactions. In particular, it is emphasized the difference between “simple” hypergraphs, where hyperedges are subsets of distinct nodes taking part in an interaction, and “multisets” hypergraphs, where repeated nodes are allowed in the same hyperedge. A proper choice has to rely on the specificity of each dataset.

In the present chapter, focusing on simple hypergraphs where literature is quite scarce and computational challenges increase, a stochastic block model is proposed to perform model-based clustering, capturing the information deriving from higher-order interactions. A discrete latent variable with  $k$  support points is associated with each node, identifying the latent states in the population. The model parameters are the weight of each latent state, and the occurrence probability of an hyperedge given the belonging latent states of its nodes. The formulation of the model is sufficiently flexible to account for possible simplified latent structures. Maximum likelihood estimation of model parameters is performed through a variational expectation-maximization algorithm by maximizing a lower bound of the log-likelihood function. Model selection is explored using the Integrated Classification Likelihood criterion. The algorithm is implemented in C++ and employing parallel computation for efficiency.

The model is applied to both synthetic and real data, and the performance of the proposal is assessed in terms of parameter estimation and ability to recover the clusters (through the Adjusted Rand Index). The proposed algorithm shows an appealing ability to recover the correct clusters in many different scenarios, ensuring an optimal performance in

detecting communities as well as disassortative behaviors. In each scenario the estimation becomes more accurate as the number of the nodes increases.

Future work may consider the issue of extending and generalizing the proposed model to a broader range of cases and situations. For example, our model could allow for self-loops without any important changes. Following a different research direction, the conditional Bernoulli distribution of the hyperedges could be replaced by any parametric distribution to handle weighted hypergraphs (with or without self-loops); an obvious example is a (zero-inflated or deflated) Poisson law. The case of multisets-hypergraphs could also be handled and would result in a fastest algorithm (though requiring a distinct implementation, which is not provided in our R package). Finally, likewise other discrete latent variable models, the proposed hypergraph stochastic block model is possibly beset by the problem of convergence to local maxima, and would benefit from the implementation of a tempered or evolutionary version of the estimation algorithm, as introduced in the previous chapters.

## Appendices

### A Proofs of theoretical results

*Proof of Lemma 1.*

Notation: for each pair of real numbers  $a, b \in \mathbb{R}$ , we denote by  $\llbracket a, b \rrbracket$  the set of integer values between  $a$  and  $b$ . Note that if  $a, b \in \mathbb{N}$ , then the cardinality of  $\llbracket a, b \rrbracket$  is equal to  $b - a + 1$ .

We consider a fixed value of  $m \geq 2$  and  $Q \geq 1$ ; let us recall that  $B^{(m)}$  is a fully symmetric tensor (4.1), so the number of free parameters in  $B^{(m)}$  is equal to the number of ordered sequences  $q_1 \leq \dots \leq q_m$  of elements in  $\llbracket 1, Q \rrbracket$ . We denote by  $\mathcal{Q}^+$  this set:

$$\mathcal{Q}^+ = \{\underline{q} = (q_1, \dots, q_m) : q_1 \leq \dots \leq q_m, q_i \in \llbracket 1, Q \rrbracket\}.$$

Moreover we denote by  $\mathcal{L}^+$  the set

$$\mathcal{L}^+ = \{\underline{l} = (l_1, \dots, l_m) : q_1 < \dots < q_m, q_i \in \llbracket 1, Q + m - 1 \rrbracket\}.$$

Then we define a function  $f$  which, to any such sequence  $\underline{q} \in \mathcal{Q}^+$ , associates  $f(\underline{q}) = (q_1, q_2 + 1, q_3 + 2, \dots, q_m + m - 1)$ . Conversely, we define a function  $g$  such that, to any sequence  $\underline{l} \in \mathcal{L}^+$  associates  $g(\underline{l}) = (l_1, l_2 - 1, l_3 - 2, \dots, l_m - m + 1)$ . It is easy to see that  $(q_1, q_2 + 1, q_3 + 2, \dots, q_m + m - 1) \in \mathcal{L}^+$  and  $(l_1, l_2 - 1, l_3 - 2, \dots, l_m - m + 1) \in \mathcal{Q}^+$ . Thus, for any  $\underline{q} \in \mathcal{Q}^+$  we get that  $f(\underline{q}) \in \mathcal{L}^+$  and for any  $\underline{l} \in \mathcal{L}^+$  we get that  $g(\underline{l}) \in \mathcal{Q}^+$ .

As a consequence, the functions  $f$  and  $g$  are such that their composition is the identity function:  $f \circ g = g \circ f = Id$ . These are one-to-one functions mapping  $\mathcal{Q}^+$  to  $\mathcal{L}^+$  and conversely. This implies that the cardinalities of these two sets are equal. But an element in  $\mathcal{L}^+$  is exactly a subset of size  $m$  of  $\llbracket 1, Q + m - 1 \rrbracket$  so that the cardinality of  $\mathcal{L}^+$  is the number of subsets of size  $m$  of  $\llbracket 1, Q + m - 1 \rrbracket$ . This concludes the proof of the lemma.  $\square$

*Proof of Proposition 2.*

We want to maximize  $\mathcal{J}(\boldsymbol{\theta}, \boldsymbol{\tau})$  with respect to  $\tau_{iq}$  under the constraint  $\sum_{q=1}^Q \tau_{iq} = 1$  for all  $i$ . Using the method of Lagrange multipliers, this is equivalent to maximizing with respect

to  $\tau_{iq}$  the Lagrangian function

$$\begin{aligned}
 \Lambda(\boldsymbol{\theta}, \boldsymbol{\tau}, \boldsymbol{\lambda}) &= \sum_{i=1}^n \lambda_i \left( \sum_{q=1}^Q \tau_{iq} - 1 \right) + \mathcal{J}(\boldsymbol{\theta}, \boldsymbol{\tau}) \\
 &= \sum_{i=1}^n \lambda_i \left( \sum_{q=1}^Q \tau_{iq} - 1 \right) + \sum_{q=1}^Q \sum_{i=1}^n \tau_{iq} \log \frac{\pi_q}{\tau_{iq}} \\
 &\quad + \sum_{i=1}^n \sum_{q=1}^Q \sum_{m=1}^{M-1} \sum_{\mathcal{Q}^{(m)}} \sum_{\mathcal{V}^{(m)} \not\ni i} \tau_{iq} \tau_{i_1 q_1} \cdots \tau_{i_m q_m} \left[ Y_{ii_1 \dots i_m} \log(B_{qq_1 \dots q_m}^{(m)}) \right. \\
 &\quad \left. + (1 - Y_{ii_1 \dots i_m}) \log(1 - B_{qq_1 \dots q_m}^{(m)}) \right].
 \end{aligned}$$

Computing the partial derivative of  $\Lambda(\boldsymbol{\theta}, \boldsymbol{\tau}, \boldsymbol{\lambda})$  with respect to  $\tau_{iq}$ , we obtain the following expression

$$\begin{aligned}
 \frac{\partial \Lambda}{\partial \tau_{iq}} &= \lambda_i + \log \frac{\pi_q}{\tau_{iq}} - 1 \\
 &\quad + \sum_{m=1}^{M-1} \sum_{\mathcal{Q}^{(m)}} \sum_{\mathcal{V}^{(m)} \not\ni i} \tau_{i_1 q_1} \cdots \tau_{i_m q_m} \left[ Y_{ii_1 \dots i_m} \log(B_{qq_1 \dots q_m}^{(m)}) + (1 - Y_{ii_1 \dots i_m}) \log(1 - B_{qq_1 \dots q_m}^{(m)}) \right] \\
 &= \lambda_i + \log \pi_q - \log \tau_{iq} - 1 \\
 &\quad + \log \prod_{m=1}^{M-1} \prod_{\mathcal{Q}^{(m)}} \prod_{\mathcal{V}^{(m)} \not\ni i} \left[ (B_{qq_1 \dots q_m}^{(m)})^{Y_{ii_1 \dots i_m}} \cdot (1 - B_{qq_1 \dots q_m}^{(m)})^{1 - Y_{ii_1 \dots i_m}} \right]^{\tau_{i_1 q_1} \cdots \tau_{i_m q_m}},
 \end{aligned}$$

which is equal to 0 if

$$\tau_{iq} = e^{\lambda_i - 1} \pi_q \prod_{m=1}^{M-1} \prod_{\mathcal{Q}^{(m)}} \prod_{\mathcal{V}^{(m)} \not\ni i} \left[ (B_{qq_1 \dots q_m}^{(m)})^{Y_{ii_1 \dots i_m}} \cdot (1 - B_{qq_1 \dots q_m}^{(m)})^{1 - Y_{ii_1 \dots i_m}} \right]^{\tau_{i_1 q_1} \cdots \tau_{i_m q_m}}.$$

The term  $e^{\lambda_i - 1} = \frac{1}{\sum_{q=1}^Q \tau_{iq}}$  is the normalizing constant such that  $\sum_{q=1}^Q \tau_{iq} = 1$  for each  $i$ . Finally, let us remark that the Lagrangian function  $\Lambda$  is concave with respect to each  $\tau_{iq}$ , being the sum of a concave term ( $\tau_{iq} \log(\pi_q / \tau_{iq})$ ) and linear terms. Then the critical point is a maximum.  $\square$

*Proof of Proposition 3.*

For the prior probabilities  $\pi_q$ , we want to maximize  $\mathcal{J}(\boldsymbol{\theta}, \boldsymbol{\tau})$  with respect to  $\pi_q$  subject to the

constraint  $\sum_{q=1}^Q \pi_q = 1$ . Using again Lagrange multipliers, this is equivalent to maximizing

$$\Lambda(\boldsymbol{\theta}, \tau, \lambda) = \lambda \left( \sum_{q=1}^Q \pi_q - 1 \right) + \mathcal{J}(\boldsymbol{\theta}, \tau)$$

Noting that the second term of  $\mathcal{J}(\boldsymbol{\theta}, \tau)$  does not depend on  $\pi_q$ , the computation of the partial derivative of  $\Lambda(\boldsymbol{\theta}, \tau, \lambda)$  reduces to

$$\frac{\partial}{\partial \pi_q} \left[ \lambda \left( \sum_{q=1}^Q \pi_q - 1 \right) + \sum_{q=1}^Q \sum_{i=1}^n \tau_{iq} \log \frac{\pi_q}{\tau_{iq}} \right] = \lambda + \sum_{i=1}^n \frac{\tau_{iq}}{\pi_q}.$$

This quantity is equal to 0 if

$$\pi_q = -\frac{1}{\lambda} \sum_{i=1}^n \tau_{iq},$$

where  $\lambda = -n$  is the normalizing constant in order to satisfy  $\sum_{q=1}^Q \pi_q = 1$ .

Note the Lagrangian function  $\Lambda$  is concave with respect to each  $\pi_q$ , being the sum of a concave term ( $\log(\pi_q/\tau_{iq})$ ), of a linear term ( $\lambda \sum_{q=1}^Q \pi_q$ ) and of a constant. The critical point is then a maximum.

Finally, the partial derivative *w.r.t.*  $B_{q_1, \dots, q_m}^{(m)}$  is

$$\frac{\partial \mathcal{J}}{\partial B_{q_1, \dots, q_m}^{(m)}} = \sum_{\mathcal{V}^{(m)}} \tau_{i_1 q_1} \cdots \tau_{i_m q_m} \left[ Y_{i_1 \dots i_m} \frac{1}{B_{q_1 \dots q_m}^{(m)}} - (1 - Y_{i_1 \dots i_m}) \frac{1}{1 - B_{q_1 \dots q_m}^{(m)}} \right].$$

Through some basic algebraic manipulations, this quantity results equal to 0 if

$$B_{q_1, \dots, q_m}^{(m)} = \frac{\sum_{\mathcal{V}^{(m)}} \tau_{i_1 q_1} \cdots \tau_{i_m q_m} Y_{i_1 \dots i_m}}{\sum_{\mathcal{V}^{(m)}} \tau_{i_1 q_1} \cdots \tau_{i_m q_m}}.$$

Again, the Lagrangian function is the sum of a concave term ( $\log(B_{q_1, \dots, q_m}^{(m)})$ ) and of some constant terms, thus being a concave function. The critical point is then a maximum.  $\square$

*Proof of Proposition 4.*

Let us define the following two subsets of  $\mathcal{Q}^{(m)}$ :

$$\begin{aligned} \mathcal{Q}^{(m,1)} &= \{ \{q_1, \dots, q_m\} \in \mathcal{Q}^{(m)} : q_1 = \dots = q_m \} \subset \mathcal{Q}^{(m)}, \\ \mathcal{Q}^{(m,2)} &= \{ \{q_1, \dots, q_m\} \in \mathcal{Q}^{(m)} : |\{q_1, \dots, q_m\}| \geq 2 \} \subset \mathcal{Q}^{(m)}. \end{aligned}$$

It is straightforward to prove that  $\mathcal{Q}^{(m,1)} \sqcup \mathcal{Q}^{(m,2)} = \mathcal{Q}^{(m)}$  (here  $\sqcup$  denotes the disjoint

union). Moreover, note that the summation  $\sum_{\mathcal{Q}^{(m,2)}}$  is equivalent to  $\sum_{q=1}^Q$ . Then the following decomposition of  $\mathcal{J}(\boldsymbol{\theta}, \boldsymbol{\tau})$  naturally holds:

$$\begin{aligned} \mathcal{J}(\boldsymbol{\theta}, \boldsymbol{\tau}) &= \sum_{q=1}^Q \sum_{i=1}^n \tau_{iq} \log \frac{\pi_q}{\tau_{iq}} \\ &+ \sum_{m=2}^M \sum_{q=1}^Q \sum_{\mathcal{V}^{(m)}} \tau_{i_1 q} \cdots \tau_{i_m q} \left[ Y_{i_1, \dots, i_m} \log \alpha^{(m)} + (1 - Y_{i_1, \dots, i_m}) \log(1 - \alpha^{(m)}) \right] \\ &+ \sum_{m=2}^M \sum_{\mathcal{Q}^{(m,2)}} \sum_{\mathcal{V}^{(m)}} \tau_{i_1 q_1} \cdots \tau_{i_m q_m} \left[ Y_{i_1, \dots, i_m} \log \beta^{(m)} + (1 - Y_{i_1, \dots, i_m}) \log(1 - \beta^{(m)}) \right] \end{aligned}$$

The partial derivative *w.r.t.*  $\alpha^{(m)}$  is

$$\frac{\partial \mathcal{J}}{\partial \alpha^{(m)}} = \sum_{q=1}^Q \sum_{\mathcal{V}^{(m)}} \tau_{i_1 q} \cdots \tau_{i_m q} \left[ Y_{i_1 \dots i_m} \frac{1}{\alpha^{(m)}} - (1 - Y_{i_1 \dots i_m}) \frac{1}{1 - \alpha^{(m)}} \right],$$

hence it follows that:

$$\hat{\alpha}^{(m)} = \frac{\sum_{q=1}^Q \sum_{\mathcal{V}^{(m)}} \tau_{i_1 q} \cdots \tau_{i_m q} Y_{i_1 \dots i_m}}{\sum_{q=1}^Q \sum_{\mathcal{V}^{(m)}} \tau_{i_1 q} \cdots \tau_{i_m q}}.$$

Analogously, the partial derivative *w.r.t.*  $\beta^{(m)}$  is

$$\frac{\partial \mathcal{J}}{\partial \beta^{(m)}} = \sum_{\mathcal{Q}^{(m,2)}} \sum_{\mathcal{V}^{(m)}} \tau_{i_1 q_1} \cdots \tau_{i_m q_m} \left[ Y_{i_1 \dots i_m} \frac{1}{\beta^{(m)}} - (1 - Y_{i_1 \dots i_m}) \frac{1}{1 - \beta^{(m)}} \right],$$

and

$$\hat{\beta}^{(m)} = \frac{\sum_{\{q_1, \dots, q_m\} \in \mathcal{Q}^{(m,2)}} \sum_{\mathcal{V}^{(m)}} \tau_{i_1 q_1} \cdots \tau_{i_m q_m} Y_{i_1 \dots i_m}}{\sum_{\{q_1, \dots, q_m\} \in \mathcal{Q}^{(m,2)}} \sum_{\mathcal{V}^{(m)}} \tau_{i_1 q_1} \cdots \tau_{i_m q_m}}.$$

This concludes the proof for the formulas under assumption **(Aff-m)**. In the same way the expressions for  $\alpha$  and  $\beta$  under assumption **(Aff)** are computed.  $\square$



## B Complete proof of the identifiability

For the sake of completeness, we provide here the complete proofs of Theorem 1 and Corollary 2. These mostly reproduce the proof of Theorem 2 in [Allman et al. \(2011\)](#).

### B.1 Proof of Theorem 1

**The strategy relying on Kruskal’s result.** The proof strongly relies on an algebraic result from [Kruskal \(1977\)](#) that appeared to be a powerful tool to establish identifiability results in various models whose common feature is the presence of discrete latent groups and at least three conditionally independent random variables. We first rephrase Kruskal’s result in a statistical context. Consider a latent random variable  $V$  with state space  $\{1, \dots, r\}$  and distribution given by the column vector  $\mathbf{v} = (v_1, \dots, v_r)$ . Assume that there are three observable random variables  $U_j$  for  $j = 1, 2, 3$ , each with finite state space  $\{1, \dots, \kappa_j\}$ . The  $U_j$ s are moreover assumed to be independent conditional on  $V$ . Let  $M_j$ ,  $j = 1, 2, 3$  be the stochastic matrix of size  $r \times \kappa_j$  whose  $i$ th row is  $\mathbf{m}_i^j = \mathbb{P}(U_j = \cdot \mid V = i)$ . Then consider the 3-dimensional array (or tensor) with dimensions  $\kappa_1 \times \kappa_2 \times \kappa_3$  denoted  $[\mathbf{v}; M_1, M_2, M_3]$  and whose  $(s, t, u)$  entry (for any  $1 \leq s \leq \kappa_1, 1 \leq t \leq \kappa_2, 1 \leq u \leq \kappa_3$ ) is defined by

$$\begin{aligned} [\mathbf{v}; M_1, M_2, M_3]_{s,t,u} &= \sum_{i=1}^r v_i m_i^1(s) m_i^2(t) m_i^3(u) \\ &= \sum_{i=1}^r \mathbb{P}(V = i) \mathbb{P}(U_1 = s \mid V = i) \mathbb{P}(U_2 = t \mid V = i) \mathbb{P}(U_3 = u \mid V = i) \\ &= \mathbb{P}(U_1 = s, U_2 = t, U_3 = u). \end{aligned}$$

Note that  $[\mathbf{v}; M_1, M_2, M_3]$  is left unchanged by simultaneously permuting the rows of all the  $M_j$  and the entries of  $\mathbf{v}$ , as this corresponds to permuting the labels of the latent classes. Knowledge of the distribution of  $(U_1, U_2, U_3)$  is equivalent to knowledge of the tensor  $[\mathbf{v}; M_1, M_2, M_3]$ .

The *Kruskal rank* of a matrix  $M$ , denoted  $\text{rank}_K M$ , is the largest number  $I$  such that *every* set of  $I$  rows of  $M$  are independent. Note that for any matrix  $M$ , its Kruskal rank is necessarily less than its rank, namely  $\text{rank}_K M \leq \text{rank } M$ , and equality of rank and Kruskal rank does not hold in general. However, in the particular case when a matrix  $M$  of size  $p \times q$  has rank  $p$ , it also has Kruskal rank  $p$ . Now, let  $I_j = \text{rank}_K M_j$ . [Kruskal \(1977\)](#) established the following result. If

$$I_1 + I_2 + I_3 \geq 2r + 2, \tag{4.12}$$

then the tensor  $[\mathbf{v}; M_1, M_2, M_3]$  uniquely determines  $\mathbf{v}$  and the  $M_j$ , up to simultaneous permutation of the rows. In other words, the set of parameters  $\{(\mathbf{v}, \mathbb{P}(U_j = \cdot | V))\}$  is uniquely identified, up to label switching on the latent groups, from the distribution of the random variables  $(U_1, U_2, U_3)$ .

To obtain generic identifiability, it is sufficient to exhibit a single parameter value for which (4.12) is satisfied. Indeed, the set of parameter values for which  $\text{rank}_K M_j$  is fixed can be expressed through a Boolean combination of polynomial inequalities ( $\neq$ , or rather non-equalities) involving matrix minors in those parameters. In the same way, the converse condition of (4.12), namely inequality  $I_1 + I_2 + I_3 \leq 2r + 1$  is the finite Boolean combination of polynomial non-equalities on the model parameters. This means that this set of parameters is an algebraic variety. But an algebraic variety can only be either the whole parameter space (in which case exhibiting a single value where (4.12) is satisfied would not be possible) or a proper subvariety, thus a subspace of dimension strictly lower than that of the whole parameter space.

The strategy of the proof for showing identifiability of certain discrete latent class models developed in [Allman et al. \(2011\)](#) and other papers by the same authors is to embed these models in the context of Kruskal's result just described. Applying Kruskal's result to the embedded model, the authors derive partial identifiability results on the embedded model, and then, using details of the embedding, relate these to the original model.

**Embedding the HSBM into Kruskal's setup.** For some number of nodes  $n$  (to be specified later), we let  $V = (Z_1, Z_2, \dots, Z_n)$  be the latent random variable, with state space  $\{1, \dots, Q\}^n$  and denote by  $\mathbf{v}$  the corresponding vector of its probability distribution. The entries of  $\mathbf{v}$  are of the form  $\pi_1^{n_1} \dots \pi_Q^{n_Q}$  for some integers  $n_q \geq 0$  and such that  $\sum_q n_q = n$ . We fix  $m \geq 2$  and consider simple  $m$ -uniform hypergraphs on the set of nodes  $\mathcal{V} = \{1, \dots, n\}$ . Recall that  $\mathcal{V}^{(m)}$  is the set of all distinct  $m$ -tuples of nodes in  $\mathcal{V}$  and  $\{Y_{i_1, \dots, i_m}; \{i_1, \dots, i_m\} \in \mathcal{V}^{(m)}\}$  the set of all indicator variables corresponding to possible (simple) hyperedges of a  $m$ -uniform hypergraph over  $\mathcal{V}$ . We will construct below subsets  $H_1, H_2, H_3 \subset \mathcal{V}^{(m)}$  of distinct  $m$ -tuples of nodes such that  $H_i \cap H_j = \emptyset$  for any  $i \neq j$ . Then, we choose the 3 observed variables  $U_j$  ( $1 \leq j \leq 3$ ) as the vectors of indicator variables  $U_j = (Y_{i_1, \dots, i_m})_{\{i_1, \dots, i_m\} \in H_j}$ . This induces that  $\kappa_j = 2^{|H_j|}$  (where  $|H_j|$  is the cardinality of  $H_j$ ). As the subsets  $H_1, H_2, H_3$  do not share any  $m$ -tuple of nodes, the random variables  $U_j$  are conditionally independent given  $V$ . We are in the statistical context of Kruskal's result.

The goal is now to construct the 3 subsets  $H_j$  of  $m$ -tuples such that their pairwise intersections are empty and such that condition (4.12) is satisfied (for at least one param-

eter value of the embedded model and thus generically for this embedded model). This construction of the  $H_j$ 's proceeds in two steps: the base case and an extension step.

Starting with a small set  $\mathcal{V}_0 = \{1, \dots, n_0\}$  of nodes, we define a matrix  $A$  of dimension  $Q^{n_0} \times 2^{\binom{n_0}{m}}$ . Its rows are indexed by latent configurations  $v \in \{1, \dots, Q\}^{n_0}$  of the nodes in  $\mathcal{V}_0$ , its columns by the set of all possible states of the vector of indicator variables  $(Y_{i_1, \dots, i_m})_{\{i_1, \dots, i_m\} \in \mathcal{V}_0}$ , and the entries of  $A$  give the probability of observing the specified states of the vector of indicator variables, conditioned on the latent configurations  $v$ . Thus each column index corresponds to a different simple  $m$ -uniform hypergraph on  $\mathcal{V}_0$ . The base case consists in exhibiting a value of  $n_0$  such that this matrix  $A$  generically has full row rank. Then, in an extension step, relying on  $n = n_0^2$  nodes, we construct the subsets  $H_1, H_2, H_3$  with the desired properties (namely their pairwise intersections are empty and (4.12) is generically satisfied).

From Kruskal's theorem, we obtain that the vector  $\mathbf{v}$  and the matrices  $M_1, M_2, M_3$  are generically uniquely determined, up to simultaneous permutation of the rows from the distribution of a simple  $m$ -uniform HSBM.

With these embedded parameters  $\mathbf{v}, M_1, M_2, M_3$  in hand, it is still necessary to recover the initial parameters of the simple  $m$ -uniform HSBM: the group proportions  $\pi_q$  and the connectivity matrix  $B^{(m)} = (B_{q_1, \dots, q_m}^{(m)})_{1 \leq q_1 \leq \dots \leq q_m \leq Q}$ . This will be done in the conclusion.

**Base case.** In the following, we drop the exponent  $(m)$  in the notation for the connection probabilities  $B$  and simply let  $B_{q_1, \dots, q_m} = \mathbb{P}(Y_{i_1, \dots, i_m} = 1 \mid Z_{i_1} = q_1, \dots, Z_{i_m} = q_m) = 1 - \bar{B}_{q_1, \dots, q_m}$ . The initial step consists in finding a value of  $n_0$  such that the matrix  $A$  of size  $Q^{n_0} \times 2^{\binom{n_0}{m}}$  containing the probabilities of any simple  $m$ -uniform hypergraph over these  $n_0$  nodes, conditional on the hidden node states, generically has full row rank.

The condition of having full row rank can be expressed as the non-vanishing of at least one  $Q^{n_0} \times Q^{n_0}$  minor of  $A$ . Composing the map sending the parameters  $\{B_{q_1, \dots, q_m}\} \rightarrow A$  with this collection of minors gives polynomials in the parameters of the model. To see that these polynomials are not identically zero, and thus are non-zero for generic parameters, it is enough to exhibit a single choice of the  $\{B_{q_1, \dots, q_m}\}$  for which the corresponding matrix  $A$  has full row rank. We choose to consider parameters  $\{B_{q_1, \dots, q_m}\}$  of the form

$$B_{q_1, \dots, q_m} = \frac{s_{q_1} s_{q_2} \dots s_{q_m}}{s_{q_1} s_{q_2} \dots s_{q_m} + t_{q_1} t_{q_2} \dots t_{q_m}}, \text{ so } \bar{B}_{q_1, \dots, q_m} = \frac{t_{q_1} t_{q_2} \dots t_{q_m}}{s_{q_1} s_{q_2} \dots s_{q_m} + t_{q_1} t_{q_2} \dots t_{q_m}},$$

with  $s_q, t_l > 0$  to be chosen later. However, since the property of having full row rank is unchanged under non-zero rescaling of the rows of the matrix  $A$ , and all entries of  $A$  are monomials with total degree  $\binom{n_0}{m}$  in  $B_{q_1, \dots, q_m}, \bar{B}_{q_1, \dots, q_m}$ , we may simplify the entries of  $A$

by removing denominators, and consider the matrix (also called  $A$ ) with entries in terms of  $B_{q_1, \dots, q_m} = s_{q_1} s_{q_2} \dots s_{q_m}$  and  $\bar{B}_{q_1, \dots, q_m} = t_{q_1} t_{q_2} \dots t_{q_m}$ .

The rows of  $A$  are indexed by the composite node states  $v \in \{1, \dots, Q\}^{n_0}$ , while its columns are indexed by the  $m$ -uniform hypergraphs  $\mathcal{H} = (y_{i_1, \dots, i_m})_{\{i_1, \dots, i_m\} \in \mathcal{V}_0} \in \{0, 1\}^{\binom{n_0}{m}}$ . For any composite hidden state  $v \in \{1, \dots, Q\}^{n_0}$  and any node  $i \in \{1, \dots, n_0\}$ , let  $v(i) \in \{1, \dots, Q\}$  denote the state of node  $i$  in the composite state  $v$ . With our particular choice of the parameters  $B_{q_1, \dots, q_m}$ , the  $(v, \mathcal{H})$ -entry of  $A$  is given by

$$\prod_{\{i_1 \dots i_m\} \in \mathcal{V}_0^{(m)}} B_{v(i_1), \dots, v(i_m)}^{y_{i_1, \dots, i_m}} \bar{B}_{v(i_1), \dots, v(i_m)}^{1-y_{i_1, \dots, i_m}} = \prod_{i=1}^{n_0} s_{v(i)}^{d_i} t_{v(i)}^{n_0-1-d_i},$$

where

$$d_i = \sum_{\substack{\{i_1 \dots i_m\} \in \mathcal{V}_0^{(m)} \\ i \in \{i_1 \dots i_m\}}} y_{i_1, \dots, i_m}$$

is the degree of node  $i$  in the hypergraph  $\mathcal{H} = (y_{i_1, \dots, i_m})_{\{i_1, \dots, i_m\} \in \mathcal{V}_0}$ . With this choice of parameters  $\{B_{q_1, \dots, q_m}\}$ , the entries in a column of  $A$  are entirely determined by the degree sequence  $\mathbf{d} = (d_i)_{1 \leq i \leq n_0}$  of the hypergraph under consideration. Two different hypergraphs may result in the same degree sequence, thus the same values in the two columns of  $A$ . For any degree sequence  $\mathbf{d} = (d_i)_{1 \leq i \leq n_0}$  arising from a simple  $m$ -uniform hypergraph on  $n_0$  nodes, let  $A_{\mathbf{d}}$  denote a corresponding column of  $A$ . In order to prove that the matrix  $A$  has full row rank, it is enough to exhibit  $Q^{n_0}$  independent columns of  $A$ . To this aim, we introduce polynomial functions whose independence is equivalent to that of corresponding columns.

For each node  $i \in \{1, \dots, n_0\}$  and each latent group  $q \in \{1, \dots, Q\}$ , introduce an indeterminate  $X_{i,q}$  and a  $Q^{n_0}$ -size row vector  $\mathbf{X} = (\prod_{1 \leq i \leq n_0} X_{i,v(i)})_{v \in \{1, \dots, Q\}^{n_0}}$ . For each degree sequence  $\mathbf{d}$ , we have

$$\begin{aligned} \mathbf{X} A_{\mathbf{d}} &= \sum_{v \in \{1, \dots, Q\}^{n_0}} \prod_{1 \leq i \leq n_0} s_{v(i)}^{d_i} t_{v(i)}^{n_0-1-d_i} X_{i,v(i)} \\ &= \prod_{1 \leq i \leq n_0} \left( s_1^{d_i} t_1^{n_0-1-d_i} X_{i,1} + \dots + s_Q^{d_i} t_Q^{n_0-1-d_i} X_{i,Q} \right). \end{aligned}$$

Now, independence of a set of columns  $\{A_{\mathbf{d}}\}$  is equivalent to the independence of the corresponding set of polynomial functions  $\{\mathbf{X} A_{\mathbf{d}}\}$  in the indeterminates  $\{X_{i,q}\}$ . For a set  $\mathcal{D}$  of degree sequences, to prove that the polynomials  $\{\mathbf{X} A_{\mathbf{d}}\}_{\mathbf{d} \in \mathcal{D}}$  are independent, we

assume that there exist scalars  $a_{\mathbf{d}}$  such that

$$\sum_{\mathbf{d} \in \mathcal{D}} a_{\mathbf{d}} \mathbf{X} A_{\mathbf{d}} \equiv 0, \quad (4.13)$$

and show that necessarily all  $a_{\mathbf{d}} = 0$ . This will be given by the following lemma from [Allman et al. \(2011\)](#). This lemma is originally formulated for a set  $\mathcal{D}$  of degree sequences. However it is not specific to degree sequences; it applies for any sets  $\mathcal{D}$  of sequences of integers indexed by  $\{1, \dots, n_0\}$  and thus we phrase it in this way. We refer to [Allman et al. \(2011\)](#) for its proof.

**Lemma 2.** *(Lemma 18 in [Allman et al. \(2011\)](#).) Assume  $n_0 \geq Q$ . Let  $\mathcal{D}$  be a set of  $n_0$ -length integer sequences such that for each  $i \in \{1, \dots, n_0\}$ , the set of  $i$ -th coordinates  $\{d_i \mid \mathbf{d} \in \mathcal{D}\}$  has cardinality at most  $Q$ . Then for generic values of  $s_q, t_l$ , for each  $i$  and each  $d^* \in \{d_i \mid \mathbf{d} \in \mathcal{D}\}$  there exist values of the indeterminates  $\{X_{i,q}\}_{1 \leq q \leq Q}$  that annihilate all the polynomials  $\mathbf{X} A_{\mathbf{d}}$  for  $\mathbf{d} \in \mathcal{D}$  except those for which  $d_i = d^*$ .*

The next step is to construct a set  $\mathcal{D}$  of  $n_0$ -length integer sequences that satisfies

- for each  $i \in \{1, \dots, n_0\}$ , the set of  $i$ -th coordinates  $\{d_i \mid \mathbf{d} \in \mathcal{D}\}$  has cardinality at most  $Q$  (condition in Lemma 2);
- any  $\mathbf{d} \in \mathcal{D}$  may be the degree sequence of a simple  $m$ -uniform hypergraph;
- $|\mathcal{D}| \geq Q^{n_0}$ .

With such a set at hand, by choosing one column of  $A$  associated to each degree sequence in  $\mathcal{D}$ , we obtain a collection of  $|\mathcal{D}| \geq Q^{n_0}$  different columns of  $A$ . These columns are independent since for each sequence  $\mathbf{d}^* \in \mathcal{D}$ , by Lemma 2 we can choose values of the indeterminates  $\{X_{i,q}\}_{1 \leq i \leq n_0, 1 \leq q \leq Q}$  such that all polynomials  $\mathbf{X} A_{\mathbf{d}}$  vanish, except  $\mathbf{X} A_{\mathbf{d}^*}$ , leading to  $a_{\mathbf{d}^*} = 0$  in equation (4.13). Thus, exhibiting such a set  $\mathcal{D}$  is the last step to prove that  $A$  has generically full row rank.

In particular, let us consider the following set of integer-valued sequences:

$$\mathcal{D} = \left\{ \mathbf{d} = (d_1, \dots, d_{n_0}) \mid \text{for } 1 \leq i \leq n_0, d_i \in \{m, 2m, 3m, \dots, Qm\} \right\}.$$

**Lemma 3.** *The set  $\mathcal{D}$  of  $n_0$ -length integer sequences satisfies*

- (i) *for each  $i \in \{1, \dots, n_0\}$ , the set of  $i$ -th coordinates  $\{d_i \mid \mathbf{d} \in \mathcal{D}\}$  has cardinality at most  $Q$ ;*

- (ii) For large enough  $n_0$  (depending on  $Q, m$ ), any  $\mathbf{d} \in \mathcal{D}$  is the degree sequence of a simple  $m$ -uniform hypergraph over  $n_0$  nodes;
- (iii)  $|\mathcal{D}| \geq Q^{n_0}$ .

Note that conditions (i), (iii) imply that  $\{d_i \mid \mathbf{d} \in \mathcal{D}\}$  should have cardinality exactly  $Q$  and that  $|\mathcal{D}| = Q^{n_0}$ .

*Proof of Lemma 3.* Points (i), (iii) are a consequence of the definition of  $\mathcal{D}$ . For any integer sequence  $\mathbf{d}$ , a necessary condition for  $\mathbf{d}$  to be a degree sequence of a simple  $m$ -uniform hypergraph over  $n_0$  nodes is that  $m$  divides  $\sum_i d_i$ . Here, we rather need sufficient conditions in order to prove (ii). We rely on Corollary 2.2 in Behrens et al. (2013).

**Corollary 2.2 in Behrens et al. (2013).** *Let  $\mathbf{d}$  be an integer-valued sequence with maximum term  $\Delta$  and let  $p$  be an integer such that  $\Delta \leq \binom{p-1}{m-1}$ . If  $m$  divides  $\sum_i d_i$  and  $\sum_i d_i \geq (\Delta - 1)p + 1$  then  $\mathbf{d}$  is the degree sequence of a simple  $m$ -uniform hypergraph.*

Fix some  $\mathbf{d} \in \mathcal{D}$ . Note that by construction,  $m$  divides  $\sum_i d_i$ . Let  $\Delta$  be the maximum value of this sequence and note that  $\Delta \leq Qm$ . Thus we choose  $p$  an integer such that  $Qm \leq \binom{p-1}{m-1}$ . Moreover,  $\sum_i d_i \geq mn_0$  and  $(\Delta - 1)p + 1 \leq \Delta p \leq Qmp$ . Then by choosing  $n_0 \geq Qp$ , we obtain the desired result.  $\square$

This concludes the proof of the base case.

**The extension step.** The extension step builds on the base case, in order to construct a larger set of  $n = n_0^2$  nodes and subsets  $H_1, H_2, H_3 \subset \mathcal{V}^{(m)}$  of distinct  $m$ -tuples of nodes in  $\mathcal{V} = \{1, \dots, n\}$  with the desired properties. This step was first stated as Lemma 16 in Allman et al. (2009) in the context of simple graphs SBM and we extend it below to our case.

Let us recall that we want to construct  $H_1, H_2, H_3 \subset \mathcal{V}^{(m)}$  that are pairwise disjoint. Then, with notation from above, we choose the 3 observed variables  $U_j$  ( $1 \leq j \leq 3$ ) as the vectors of indicator variables  $U_j = (Y_{i_1, \dots, i_m})_{\{i_1, \dots, i_m\} \in H_j}$ . As the subsets  $H_1, H_2, H_3$  do not share any  $m$ -tuple of nodes, the random variables  $U_j$  are conditionally independent given  $V = (Z_1, \dots, Z_n)$ . We let  $M_j$  denote the  $Q^n \times 2^{|H_j|}$  matrix of conditional probabilities of  $U_j$  given  $Z$ .

**Lemma 4.** *Suppose that for some number of nodes  $n_0$ , the matrix  $A$  of size  $Q^{n_0} \times 2^{\binom{n_0}{m}}$  defined above has generically full row rank. Then with  $n = n_0^2$  there exist pairwise disjoint subsets  $H_1, H_2, H_3 \subset \mathcal{V}^{(m)}$  of  $m$ -tuples of nodes in  $\mathcal{V} = \{1, \dots, n\}$  such that for each  $j$  the  $Q^n \times 2^{|H_j|}$  matrix  $M_j$  has generically full row rank ( $Q^n$ ).*

*Proof of Lemma 4.* Let us describe the construction of  $H_j$ . We will partition the  $n_0^2$  nodes into  $n_0$  groups of size  $n_0$  in three different ways, each way leading to one  $H_j$ . Then each  $H_j$  will be the union of the  $n_0$  sets of all  $m$ -tuples made of some  $n_0$  nodes. Thus each  $H_j$  has cardinality  $n_0 \binom{n_0}{m}$ .

Labeling the nodes by  $(u, v) \in \{1, \dots, n_0\} \times \{1, \dots, n_0\}$ , we picture the nodes as lattice points in a square grid. We take as the partition leading to  $H_1$  the rows of the grid, as the partition leading to  $H_2$  the columns of the grid, and as the partition leading to  $H_3$  the diagonals. In other words,  $H_1$  is the union over  $n_0$  rows of all  $m$ -tuples of nodes within each row. The same with columns and diagonals. Explicitly, we define two functions  $u, v$  that associate to any  $i \in \{1, \dots, n_0\}$  its coordinates  $(u(i), v(i))$  on the  $n_0 \times n_0$  grid. Then, the  $H_j$  are  $m$ -tuple of nodes defined as

$$\begin{aligned} H_1 &= \cup_{u=1}^{n_0} H_1(u) = \cup_{u=1}^{n_0} \{ \{i_1, \dots, i_m\} \in \mathcal{V}^{(m)} \mid \forall k, u(i_k) = u, v(i_k) \in \{1, \dots, n_0\} \}, \\ H_2 &= \cup_{v=1}^{n_0} H_2(v) = \cup_{v=1}^{n_0} \{ \{i_1, \dots, i_m\} \in \mathcal{V}^{(m)} \mid \forall k, v(i_k) = v, u(i_k) \in \{1, \dots, n_0\} \}, \\ H_3 &= \cup_{s=1}^{n_0} H_3(s) \\ &= \cup_{s=1}^{n_0} \{ \{i_1, \dots, i_m\} \in \mathcal{V}^{(m)} \mid \forall k, u(i_k) = s, v(i_k) = s + t \bmod n_0 \text{ for } t \in \{1, \dots, n_0\} \}. \end{aligned}$$

The  $H_j$  are pairwise disjoint as required.

The matrix  $M_j$  of conditional probabilities of  $U_j$  given  $Z$  has  $Q^n$  rows indexed by composite states of all  $n = n_0^2$  nodes, and  $2^{n_0 \binom{n_0}{m}}$  columns indexed by  $m$ -tuples in  $H_j$ .

Observe that with an appropriate ordering of the rows and columns (which is dependent on  $j$ ),  $M_j$  has a block structure given by

$$M_j = A \otimes A \otimes \dots \otimes A \text{ (} n_0 \text{ factors)}. \quad (4.14)$$

(Note that since  $A$  is  $Q^{n_0} \times 2^{\binom{n_0}{m}}$ , the tensor product on the right is  $(Q^{n_0})^{n_0} \times \left(2^{\binom{n_0}{m}}\right)^{n_0}$  which is  $Q^{n_0^2} \times 2^{n_0 \binom{n_0}{m}}$ , the size of  $M_j$ .) That  $M_j$  is this tensor product is most easily seen by noting the partitioning of the  $n_0^2$  nodes into  $n_0$  disjoint sets (rows, columns and diagonals of the grid) gives rise to  $n_0$  copies of the matrix  $A$ , one for each set of all simple  $m$ -uniform hypergraphs over  $n_0$  nodes. The row indices of  $M_j$  are obtained by choosing an assignment of states to the nodes in  $H_j(u)$  for each  $u$  independently, and the column indices by the union of independently-chosen simple  $m$ -uniform hypergraphs subgraphs on  $H_j(u)$  for each  $u$ . This independence in both rows and columns leads to the tensor decomposition of  $M_j$ .

Now since  $A$  has generically full row rank ( $Q^{n_0}$ ), equation (4.14) implies that  $M_j$  does as well (*i.e.* has row rank  $Q^{n_0^2} = Q^n$ ).  $\square$

Next, with  $\mathbf{v}, M_1, M_2, M_3$  defined by the embedding given in the previous paragraphs, we apply Kruskal's Theorem to the table  $[\mathbf{v}; M_1, M_2, M_3]$ . By construction of the  $M_j$ , condition (4.12) is generically satisfied since  $3Q^n \geq 2Q^n + 2$ . Thus the vector  $\mathbf{v}$  and the matrices  $M_1, M_2, M_3$  are generically uniquely determined, up to simultaneous permutation of the rows from the distribution of a simple  $m$ -uniform HSBM.

It now remains to recover the original parameters of the simple  $m$ -uniform HSBM: the group proportions  $\pi_q$  and the connectivity matrix  $(B_{q_1, \dots, q_m}^{(m)})_{1 \leq q_1 \leq \dots \leq q_m \leq Q}$ .

**Conclusion for the original model.** The entries of  $\mathbf{v}$  are of the form  $\pi_1^{n_1} \dots \pi_Q^{n_Q}$  with  $\sum n_q = n$ , while the entries of the  $M_j$  contain information on the  $B_{q_1, \dots, q_m}^{(m)}$ . Although the ordering of the rows of the  $M_j$  is arbitrary, crucially we do know how the rows of  $M_j$  are paired with the entries of  $\mathbf{v}$ .

By focusing on one of the matrices, say  $M_1$ , and adding appropriate columns of it, we can obtain marginal conditional probabilities of single hyperedge variables, namely a column vector with values  $(\mathbb{P}_{\boldsymbol{\theta}}(Y_{i_1, \dots, i_m} = 1 | (Z_1, \dots, Z_n) = v))_v$  for any  $m$ -tuple  $\{i_1, \dots, i_m\}$ . Indeed, this vector is obtained by summing all the columns of  $M_1$  corresponding to simple  $m$ -uniform hypergraphs with  $Y_{i_1, \dots, i_m} = 1$ . Thus, we recover the set  $\{B_{q_1, \dots, q_m}^{(m)}\}_{1 \leq q_1 \leq \dots \leq q_m \leq Q}$ , but without order. Namely, we still do not know the  $B_{q_1, \dots, q_m}^{(m)}$  up to a permutation on  $\{1, \dots, Q\}$  only, but rather up to a permutation on  $\{1, \dots, Q\}^n$ .

In the following, we assume without loss of generality, as it is a generic condition, that all  $\{B_{q_1, \dots, q_m}^{(m)}\}_{1 \leq q_1 \leq \dots \leq q_m \leq Q}$  are distinct.

We look at the first  $(m + 1)$  nodes  $\mathcal{V}_1 = \{1, \dots, m, m + 1\}$  and consider the  $m + 1$  different  $m$ -tuples  $\{i_1, \dots, i_m\} \in \mathcal{V}_1^{(m)}$  that can be made from these nodes ( $i_k \in \mathcal{V}_1$ ). Again, for each of these  $m$ -tuples, adding appropriate columns of  $M_1$ , we can jointly obtain the vectors of conditional marginal probabilities  $(\mathbb{P}_{\boldsymbol{\theta}}(Y_{\{i_1, \dots, i_m\}} = 1 | (Z_1, \dots, Z_n) = v))_v$ . Jointly means that all those vectors share the same ordering over the index  $v \in \{1, \dots, Q\}^n$ . In other words, we recover the sets of values

$$\forall v \in \{1, \dots, Q\}^n, \quad R_v = \{B_{v_{i_1}, \dots, v_{i_m}}^{(m)}; \{i_1, \dots, i_m\} \in \mathcal{V}_1^{(m)}\}.$$

Now, we assumed the  $B$ 's are all distinct so the cardinalities of the sets  $R_v$  will help us discriminate the different parameters (up to a permutation on  $\{1, \dots, Q\}$  only). Indeed, there are exactly  $Q$  sets  $R_v$  with cardinality exactly one. These corresponds to the cases where  $v = (q, q, \dots, q)$  for some  $1 \leq q \leq Q$ . From this, we can distinguish the parameters of the form  $\{B_{q, \dots, q}^{(m)}; 1 \leq q \leq Q\}$  from the complete set of parameters. Note that the corresponding entries of  $v$  are given by  $\pi_q^m$ . So we also recover the paired values



$\{(\pi_q, B_{q,\dots,q}^{(m)}); 1 \leq q \leq Q\}$ . Then, we continue with the sets  $R_v$  with cardinality two: these are of the form  $\{B_{q,\dots,q}^{(m)}; B_{q,\dots,q,l}^{(m)}\}$  for some  $1 \leq q \neq l \leq Q$ . As we already identified the parameters  $\{B_{q,\dots,q}^{(m)}; 1 \leq q \leq Q\}$  and all  $B$ 's are distinct, this enables us to identify the set of parameters  $\{B_{q,\dots,q,l}^{(m)}; 1 \leq q \neq l \leq Q\}$ . By induction, we recover the set of parameters  $\{B_{q,\dots,q,l,l'}^{(m)}; 1 \leq q, l, l' \leq Q \text{ and } q, l, l' \text{ distinct}\}$  *et caetera*, ending with the set of parameters  $\{B_{q_1,\dots,q_m}^{(m)}; 1 \leq q_1 < q_2 < \dots < q_m \leq Q\}$ . This means that we finally have obtained the parameters  $\{\pi_q, B_{q_1,\dots,q_m}^{(m)}\}_{1 \leq q \leq Q; 1 \leq q_1 < \dots < q_m \leq Q}$  up to a permutation over  $\{1, \dots, Q\}$ .

Finally, note that all generic aspects of this argument, in the base case and the requirement that the parameters  $B_{q_1,\dots,q_m}^{(m)}$  be distinct, concern only the  $B_{q_1,\dots,q_m}^{(m)}$ . Thus if the group proportions  $\pi_q$  are fixed to any specific values, the theorem remains valid.

*Remark 1.* The requirement on large enough  $n$  is more precisely given as  $n \geq Q^2 p^2$  where  $p$  is the smallest integer such that  $\binom{p-1}{m-1} \geq Qm$ . A rough approximation gives that  $p$  is of the order  $(Qm)^{1/(m-1)}$  which gives that  $n$  should be larger than  $Q^2(Qm)^{2/(m-1)}$ .

## B.2 Proof of Corollary 2

From the probability distribution  $\mathbb{P}_\theta$  over simple hypergraphs  $\mathcal{H}$  on a set of  $n$  nodes and hyperedges with largest size  $M$ , we automatically obtain all the probability distributions  $\mathbb{P}_\theta$  restricted to simple  $m$ -uniform hypergraphs  $\mathcal{H}_m$  on the same set of nodes. Applying the result of Theorem 1 for all values  $m$  is sufficient to obtain the desired result. Indeed, as  $M$  is finite, the union of the finite number of lower-dimensional subspaces where identifiability for fixed  $m$  may not be satisfied gives a lower-dimensional subspace, ensuring generic identifiability. Moreover, for each value of  $m$ , we recover the parameter  $\theta^{(m)}$  up to a permutation on  $\{1, \dots, Q\}$ . Now, for any  $m \neq m'$  it remains to be able to jointly order the parameters  $\theta^{(m)}$  and  $\theta^{(m')}$  up to a permutation on  $\{1, \dots, Q\}$ . If all the  $\pi_q$ 's are different, which is a generic condition, this can be easily done because  $\theta^{(m)}$  and  $\theta^{(m')}$  share the same distinct  $\pi_q$ 's.

## C Artifacts induced by bipartite graph models

We give here additional considerations to the issues from Section 4.3.1.

First, in order to view a bipartite graph as a hypergraph, one first needs to select the top and bottom parts. Swapping the role of the two parts will in general give another hypergraph. Most statistical models of bipartite graphs handle the two parts symmetrically and do not differentiate between a top and a bottom part. They are thus inadequate for modeling hypergraphs.

One may also note that most random bipartite graphs models are designed for fixed parts sizes, which induces, on top of a fixed number of nodes, a fixed number of hyperedges in the corresponding hypergraph model, an artifact which is not always desirable. For instance the random uniform hypergraphs model allows for any possible density on the hyperedges.

A last example of inadequacy is given by configuration models on bipartite graphs that induce configuration models on hypergraphs. In these models, the degree distributions in each part are kept fixed. When projected in the hypergraphs space, that means that the degrees of the nodes and the sizes of the hyperedges are kept fixed. Then, relying on shuffling algorithms to explore the space of this configuration model, one will lose the labels on the bottom part (the hyperedges part) as these are automatically induced by the new edges of the bipartite graph and the labeling of the top part (the nodes part). As a consequence, if a specific node tends to take part in large size hyperedges, this information is lost in the configuration model issued from bipartite graphs.

To our knowledge, there is no configuration model on hypergraphs that only keeps the nodes degrees sequence fixed. We mention that Section 4 from [Chodrow \(2020\)](#) provides a discussion about the limitations of the embedding approach in terms of the types of hypergraph null models from which we can conveniently sample. In particular, [Chodrow \(2020\)](#) establishes that there is no obvious route for vertex-label sampling in hypergraphs through bipartite random graphs.

## D Computational details

In order to provide an efficient implementation, the whole estimation algorithm is implemented in C++ language using the `Armadillo` library for linear algebra. Moreover the implementation is freely available by means of the R packages `Rcpp` (Eddelbuettel and François, 2011; Eddelbuettel, 2013) and `RcppArmadillo` (Eddelbuettel and Sanderson, 2014). In the following, we consider some of the most relevant computational details.

**Heavy computational cost** Dealing with very large data structures, the main drawback of the proposed algorithm is the intensive computational effort, in terms of both execution time needed to converge and required memory space. The most outstanding example regards the computation of the products  $\tau_{i_1 q_1} \cdots \tau_{i_m q_m}$ , required both in the VE-Step (see Proposition 2, for  $\tau_{iq}$ ) and in the M-Step (see Proposition 3, for  $B_{q_1, \dots, q_m}^{(m)}$ ). The huge computational cost of this calculation derives from the large number of potential unordered node tuples even for rather small values of  $n$  and  $m$ ; indeed it is straightforward to show that  $|\mathcal{V}^{(m)}| = \binom{n}{m}$ . A first possibility is to compute all the products  $\tau_{i_1 q_1} \cdots \tau_{i_m q_m}$  in a recursive manner at the beginning of each VEM iteration and to store them in a matrix. Although this is actually very beneficial for the computational time, the resulting matrix is huge, having number of rows and columns equal to  $\binom{n}{m}$  and  $\binom{Q+m-1}{m}$  respectively. The result is a structure that is intractable except for very small values of  $n$ ,  $Q$ , and (especially)  $m$ . Taking into account that every element requires 8 bytes, we report some examples in Table 4.7, in order to better clarify the magnitude of the quantity to store. Stated the impossibility to store a matrix of such sizes, the computation of the required products  $\tau_{i_1 q_1} \cdots \tau_{i_m q_m}$  is implemented directly inside the VE- and M-Steps through nested loops; this process involves an important increase in the computing times, but on the other hand requires a minimal amount of memory. To handle the slowness of the computation, both the VE-Step and the M-Step are efficiently implemented in parallel through the `RcppParallel` package (Allaire et al., 2022).

**Floating point underflow.** Another crucial aspect is the possible occurrence of numerical instability deriving from the multiplication of many small values in the computation of  $\hat{\tau}_{iq}$ . A simple remedy is provided by the calculation of  $\log \hat{\tau}_{iq}$  instead of  $\hat{\tau}_{iq}$ . So, denoting  $b_{iq} = \log(\hat{\tau}_{iq} - c_i)$ , we compute  $\hat{\tau}_{iq}$  relying on

$$\hat{\tau}_{iq} = \frac{\exp(b_{iq} - b_{\text{MAX},i})}{\sum_{p=1}^Q \exp(b_{ip} - b_{\text{MAX},i})},$$

| $n$ | $m$ | $Q$ | Memory size        | $n$ | $m$ | $Q$ | Memory size          |
|-----|-----|-----|--------------------|-----|-----|-----|----------------------|
| 100 | 3   | 2   | $\approx 5.2 MB$   | 500 | 3   | 2   | $\approx 0.6 GB$     |
| 100 | 3   | 4   | $\approx 25.9 MB$  | 500 | 3   | 4   | $\approx 3.3 GB$     |
| 100 | 6   | 2   | $\approx 66.8 GB$  | 500 | 6   | 2   | $\approx 1179.2 TB$  |
| 100 | 6   | 4   | $\approx 801.1 GB$ | 500 | 6   | 4   | $\approx 14150.8 TB$ |

**Table 4.7:** Memory size of the matrix containing the products  $\tau_{i_1q_1} \cdots \tau_{i_mq_m}$  for given values of  $n$  (number of nodes),  $Q$  (number of latent groups) and  $m$  (hyperedge size)

where  $b_{\text{MAX},i} = \max_{q=1\dots Q} b_{iq}$  prevents the denominator to grow excessively large, thus avoiding new potential numerical issues related to the floating point underflow.

## E Spectral clustering

Spectral clustering is one of the most popular modern clustering algorithm. It is widely used for partitioning of data with complex topological clustering. Several different spectral clustering algorithms have been proposed for graphs and, more recently, for hypergraphs. In the following we will not give a review of the whole literature on spectral clustering (this would be impossible due to the overwhelming amount of literature on this subject, and it would fall outside the purposes of this work). Instead we will go through the specific algorithms and aspects useful. The following notation, introduced in Section 4.2.3, will be used for both graphs and hypergraphs:  $H$  denotes the incidence matrix,  $A$  the adjacency matrix,  $D$  and  $\Delta$  two diagonal matrices such that  $d_{ii} = \sum_e h_{ie}$  and  $\delta_{ee} = \sum_i h_{ie}$ .

**Absolute spectral clustering for graphs** Following [Rohe et al. \(2011\)](#), we enumerate here the required steps:

1. Compute the graph Laplacian matrix as  $L = I_n - D^{1/2}AD^{-1/2}$ .
2. Find the eigenvectors of  $L$  corresponding to the  $Q$  eigenvalues of  $L$  that are largest in absolute value. Define  $X \in \mathbb{R}^{n \times Q}$  as the matrix containing these eigenvectors as columns.
3. Treating each row of  $X$  as a point in  $\mathbb{R}^Q$ , run the  $k$ -means algorithm ([Forgy, 1965](#); [MacQueen, 1967](#)) with  $Q$  clusters.

**Spectral clustering for hypergraphs** Following [Ghoshdastidar and Dukkipati \(2017, Algorithm 1\)](#), we enumerate here the required steps:

1. Compute the hypergraph Laplacian matrix  $L$  as  $L = I_n - D^{-1/2}H\Delta^{-1}H'D^{-1/2}$  (see Section 4.2.3 for more details).
2. Compute the  $Q$  leading eigenvectors of  $L$  (*i.e.* the eigenvectors that corresponds to the  $k$  smallest eigenvalues of  $L$ ), and denote by  $X \in \mathbb{R}^{n \times Q}$  the matrix containing these vectors as columns.
3. Normalize rows of  $X$  to have unit norm and call this matrix  $\bar{X}$ .
4. Cluster the rows of  $\bar{X}$  with the  $k$ -means algorithm ([Forgy, 1965](#); [MacQueen, 1967](#)) into  $Q$  clusters.

## F More on the simulation study

### F.1 Hyperparameters settings

All the simulations in Sections 4.5 and 4.6 were performed with the following hyperparameters. Concerning the spectral clustering initializations, the  $k$ -means algorithm (on the rows of the column leading eigenvectors matrix) is run with 100 different initializations. The tolerance threshold  $\epsilon$  used to stop the fixed point and the VEM algorithm is set to  $10^{-6}$ . The maximum numbers of iterations for the fixed point and the VEM algorithm were set to  $U_{\max} = 50$  and  $T_{\max} = 50$ , respectively.

### F.2 Comparison of different initialization strategies

In the following we compare the performance of the proposed VEM algorithm under different initialization strategies introduced in Section 4.4.3.3. In particular, we consider standard spectral clustering, “soft” spectral clustering, and graph-component absolute spectral clustering initializations. For each strategy, we compute the average ARI over the same 10 samples used in the simulation study in Section 4.5.1. Results are summarized in Table 4.8.

| $n$ | Scenario A |      |      | Scenario B |      |      | Scenario C |      |      |
|-----|------------|------|------|------------|------|------|------------|------|------|
|     | SC         | SSC  | ASC  | SC         | SSC  | ASC  | SC         | SSC  | ASC  |
| 50  | 0.70       | 1.00 | 0.70 | 0.50       | 1.00 | 0.50 | 0.10       | 0.50 | 0.19 |
| 100 | 1.00       | 1.00 | 0.60 | 0.40       | 1.00 | 0.20 | 0.20       | 0.90 | 0.20 |
| 150 | 1.00       | 1.00 | 0.70 | 0.30       | 1.00 | 0.40 | 0.30       | 1.00 | 0.10 |
| 200 | 1.00       | 1.00 | 0.50 | 0.30       | 1.00 | 0.40 | 0.30       | 1.00 | 0.20 |

**Table 4.8:** Performance of the proposed VEM algorithm under different initialization strategies: spectral clustering (SC), “soft” spectral clustering (SSC) and graph-component absolute spectral clustering (ASC). Each value is the average Adjusted Rand Index over 10 simulated datasets

It is clear that the “soft” spectral clustering initialization provides the best performance in each scenario, showing much higher values than those of the other two strategies; this behavior is especially evident for settings B and C, while in scenario A the advantage is generally less pronounced. We conjecture that a “hard” initialization prevents the VEM algorithm to accurately explore the parameter space, thus leading the estimation algorithm to fail when the starting value is not close enough to the correct solution. The result is convergence to a local maximum. On the contrary, a “soft” initial value is able to explore more thoroughly the solution space, thus reaching the global maximum and recovering the

correct clustering in the overwhelming majority of samples. This analysis justifies the use of “soft” spectral clustering initialization in the simulation study in Section 4.5.

### F.3 Influence of the initial value on the VEM algorithm

As a final note, we also assess the influence of starting values on the behavior of the VEM algorithm. To this aim, we preliminary analyze the performance of spectral clustering algorithm, relying again on the ARI. Results are reported in Table 4.9 and clearly show the opposite behavior of this clustering method in detecting communities (scenario A) and disassortative behaviors (scenarios B and C): taking into account community detection, spectral clustering algorithm perfectly recovers the true clusters, apart from a few cases in which  $n$  is very small. On the contrary, considering disassortative behaviors, spectral clustering algorithm completely fails in determining the correct clusters, all values of the ARI being extremely close to 0. Hence, “soft” spectral clustering proves to be a very smart initialization strategy for scenario A, while for scenarios B and C, it behaves analogously to a random starting value. The optimal performance of VEM algorithm throughout all scenarios, therefore, highlights a very weak influence of the starting value on the behavior of the algorithm: a random initialization usually ensures a proper convergence and a correct clustering; instead, the real advantage deriving from the adoption of a smart initial value is a reduction in computing time (data not shown).

| $n$ | Scenario A | Scenario B            | Scenario C            |
|-----|------------|-----------------------|-----------------------|
| 50  | 0.69       | $-0.43 \cdot 10^{-2}$ | $-3.57 \cdot 10^{-3}$ |
| 100 | 0.99       | $0.43 \cdot 10^{-2}$  | $-7.01 \cdot 10^{-3}$ |
| 150 | 1.00       | $-0.12 \cdot 10^{-2}$ | $4.60 \cdot 10^{-5}$  |
| 200 | 1.00       | $-0.42 \cdot 10^{-2}$ | $-4.51 \cdot 10^{-3}$ |

**Table 4.9:** Adjusted Rand Index for different scenarios and number of nodes with respect to the soft spectral clustering initialization. Each value is obtained as the average over 10 simulated datasets

### F.4 Computational time

In this Section we assess the performance of the proposed VEM algorithm in terms of computation al time. To this aim we rely again on the samples used in the simulation study in Section 4.5.1, and we consider “soft” spectral clustering initialization. The estimation is performed by employing an Intel(R) Core(TM) i7-8700T CPU @ 2.40GHz Windows desktop with 8 GB of RAM. Table 4.10 reports the average time in seconds for each scenario.

| $n$ | Scenario A | Scenario B | Scenario C |
|-----|------------|------------|------------|
| 50  | 13.96      | 18.00      | 32.61      |
| 100 | 98.11      | 311.00     | 570.34     |
| 150 | 1,054.75   | 3,113.51   | 3,269.17   |
| 200 | 5,755.97   | 18,629.58  | 19,181.36  |

**Table 4.10:** Computational time in seconds of the Variational Expectation-Maximization algorithms for each setting, computed as the mean over 10 simulated samples as presented in Section 4.5.1



## G The Hypergraph SB model is not a bipartite SB model

In this Section, we briefly outline that (i) while the bipartite stochastic block model can be seen as a particular case of the HSB model, (ii) the converse is not true in general.

To see point (i), let us consider a bipartite SB model on a graph  $\mathcal{G}$  with nodes divided in 2 parts, say  $\mathcal{V} = \{1, \dots, n\}$  and  $\mathcal{U} = \{1, \dots, e\}$ . The model has  $Q$  groups (respectively,  $R$  groups) on the subset of nodes  $\mathcal{V}$  (respectively,  $\mathcal{U}$ ), with group proportions  $\pi$  (respectively,  $\eta$ ). We let  $Z_1, \dots, Z_n$  (respectively,  $W_1, \dots, W_e$ ) denote the latent groups of the node set  $\mathcal{V}$  (respectively,  $\mathcal{U}$ ).

The model is also endowed with a connectivity matrix  $M$  of size  $Q \times R$  whose entries  $M_{qr}$  are the conditional probabilities that a node in  $\mathcal{V}$  from group  $q$  connects a node in  $\mathcal{U}$  from group  $r$ . In other words  $M_{qr} = \mathbb{P}(X_{iu} = 1 | Z_i = q, W_u = r)$  where  $X = (X_{iu})$  is the  $n \times e$  incidence matrix of  $G$ .

Now consider the hypergraph  $\mathcal{H}$  constructed on the set of nodes  $\mathcal{V}$  and whose hyperedges are obtained by looking at the set of nodes in  $\mathcal{V}$  connected to a same node in  $\mathcal{U}$ . (A similar construction could be made with swapping the roles of  $\mathcal{V}$  and  $\mathcal{U}$ ). Then, the probability distribution of  $\mathcal{H}$  under the induced bipartite SB model is exactly a HSB model with  $Q$  groups, with group proportions  $\pi$  and parameters

$$\begin{aligned} B_{q_1, \dots, q_m}^{(m)} &= \mathbb{P}(Y_{i_1, \dots, i_m} = 1 | Z_{i_1} = q_1, \dots, Z_{i_m} = q_m) \\ &= \mathbb{P}(X_{i_1, u} = 1, \dots, X_{i_m, u} = 1 | Z_{i_1} = q_1, \dots, Z_{i_m} = q_m) \\ &= \sum_{r=1}^R \mathbb{P}(X_{i_1, u} = 1, \dots, X_{i_m, u} = 1, W_u = r | Z_{i_1} = q_1, \dots, Z_{i_m} = q_m) \\ &= \sum_{r=1}^R \eta_r \prod_{q=q_1}^{q_m} M_{qr}, \end{aligned}$$

where  $u$  is the node that connects  $\{i_1, \dots, i_m\}$  into a hyperedge. So, we see that the bipartite SB model induces a HSB model with constrained connection probabilities.

Let us now explain why (ii) the converse is not true in general. We start from a HSB model with  $Q$  groups and parameters  $((\pi_q)_q, (B_{q_1, \dots, q_m}^{(m)})_{q_1, \dots, q_m})_{2 \leq m \leq M}$  on a hypergraph  $\mathcal{H}$  with set of nodes  $\mathcal{V}$ . Considering  $\mathcal{U} = \{1, \dots, e\}$  where  $e$  is the number of hyperedges in  $H$ , we construct a bipartite graph  $\mathcal{G}$  with nodes  $\mathcal{V} \times \mathcal{U}$  and links between any  $i \in \mathcal{V}$  and any  $u \in \mathcal{U}$  whenever node  $i$  belongs to hyperedge  $u$  in the hypergraph  $\mathcal{H}$ . Now, if there is a bipartite SB model on  $\mathcal{G}$  with the same distribution of  $\mathcal{H}$ , then necessarily it has  $Q$  groups on  $\mathcal{V}$ , with group proportions given by  $\pi$ . We let  $R$  denote the number of groups of such a model on  $\mathcal{U}$ , together with  $\eta$  the corresponding group proportions, and  $M$  the  $Q \times R$  matrix

of connection probabilities. Then we observe that  $\eta$  and  $M$  should satisfy the relations

$$\forall 2 \leq m \leq M, \forall q_1, \dots, q_m \in \{1, \dots, Q\}^m, \quad B_{q_1, \dots, q_m}^{(m)} = \sum_{r=1}^R \eta_r \prod_{q=q_1}^{q_m} M_{qr}. \quad (4.15)$$

Here, we first remark that the bipartite SB model fit on the co-authorship dataset (from Section 4.6) selects  $R = 1$ , thus inducing hyperedges connectivity parameters with a product form

$$B_{q_1, \dots, q_m}^{(m)} = \prod_{q=q_1}^{q_m} M_{q1}.$$

Our HSB model on this same dataset did not result in hyperedges connectivity parameters with a product form, which establishes that the models are clearly different.

Now, more generally, we could ask whether for given parameters  $(B_{q_1, \dots, q_m}^{(m)})_{2 \leq m \leq M}$ , there exist some values of  $R, \eta$  and  $M$  such that (4.15) is satisfied. The answer is: not always. To see this, consider for instance  $Q = 2$  and remark the relation between the two quantities

$$B_{11}^{(2)} = \sum_{r=1}^R \eta_r M_{1r}^2,$$

$$B_{111}^{(3)} = \sum_{r=1}^R \eta_r M_{1r}^3,$$

so that  $B_{11}^{(2)}$  and  $B_{111}^{(3)}$  cannot be chosen independently.

## Bibliography

- AHN, K., LEE, K., AND SUH, C. (2018). Hypergraph spectral clustering in the weighted stochastic block model. *IEEE J. Sel. Top. Signal Process.*, **12**, 959–974.
- ALLAIRE, J., FRANCOIS, R., USHEY, K., VANDENBROUCK, G., M., G., AND INTEL (2022). *RcppParallel: Parallel Programming Tools for Rcpp*. R package version 5.1.5.
- ALLMAN, E., MATIAS, C., AND RHODES, J. (2009). Identifiability of parameters in latent structure models with many observed variables. *Ann. Statist.*, **37**, 3099–3132.
- ALLMAN, E., MATIAS, C., AND RHODES, J. (2011). Parameters identifiability in a class of random graph mixture models. *J. Stat. Plan. Inference*, **141**, 1719–1736.
- BALASUBRAMANIAN, K. (2021). Nonparametric modeling of higher-order interactions via hypergraphons. *J. Mach. Learn. Res.*, **22**, 1–35.
- BATTISTON, F., CENCETTI, G., IACOPINI, I., LATORA, V., LUCAS, M., PATANIA, A., YOUNG, J.-G., AND PETRI, G. (2020). Networks beyond pairwise interactions: Structure and dynamics. *Phys. Rep.*, **874**, 1–92.
- BEHRENS, S., ERBES, C., FERRARA, M., HARTKE, S., REINIGER, B., SPINOZA, H., AND TOMLINSON, C. (2013). New results on degree sequences of uniform hypergraphs. *Electron. J. Comb.*, **20**, 14–18.
- BICK, C., GROSS, E., HARRINGTON, H., AND SCHAUB, M. (2021). What are higher-order networks? Technical report, arXiv:2104.11329.
- BIERNACKI, C., CELEUX, G., AND GOVAERT, G. (2000). Assessing a mixture model for clustering with the Integrated Completed Likelihood. *IEEE Trans. Pattern Anal. Mach. Intell.*, **22**, 719–725.
- BOLLA, M. (1993). Spectra, euclidean representations and clusterings of hypergraphs. *Discrete Math.*, **117**, 19–39.
- CHELARU, M., EAGLEMAN, S., ANDREI, A., MILTON, R., KHARAS, N., AND DRAGOI, V. (2021). High-order correlations explain the collective behavior of cortical populations in executive, but not sensory areas. *Neuron*, **109**, 3954–3961.
- CHIEN, I., LIN, C., AND WANG, I. (2019). On the minimax misclassification ratio of hypergraph community detection. *IEEE Trans. Inf. Theory*, **65**, 8095–8118.

- CHIQUET, J., DONNET, S., GROSSBM TEAM, AND BARBILLON, P. (2022). *sbm: Stochastic Blockmodels*. R package version 0.4.4.
- CHODROW, P. (2020). Configuration models of random hypergraphs. *J. Complex Netw.*, **8**, 1–26.
- CHODROW, P., VELDT, N., AND BENSON, A. (2021). Generative hypergraph clustering: From blockmodels to modularity. *Sci. Adv.*, **7**, 1–13.
- COLE, S. AND ZHU, Y. (2020). Exact recovery in the hypergraph stochastic block model: A spectral algorithm. *Linear Algebra Appl.*, **593**, 45–73.
- DEMPSTER, A., LAIRD, N., AND RUBIN, D. (1977). Maximum likelihood from incomplete data via the em algorithm (with discussion). *J. R. Stat. Soc. Ser. B*, **39**, 1–38.
- EDDELBUETTEL, D. (2013). *Seamless R and C++ Integration with Rcpp*. Springer, New York. ISBN 978-1-4614-6867-7.
- EDDELBUETTEL, D. AND FRANÇOIS, R. (2011). Rcpp: Seamless R and C++ integration. *Journal of Statistical Software*, **40**, 1–18.
- EDDELBUETTEL, D. AND SANDERSON, C. (2014). RcppArmadillo: Accelerating R with high-performance C++ linear algebra. *Computational Statistics and Data Analysis*, **71**, 1054–1063.
- ESTRADA, E. AND RODRÍGUEZ-VELÁZQUEZ, J. (2006). Subgraph centrality and clustering in complex hyper-networks. *Physica A*, **364**, 581–594.
- FLAMM, C., STADLER, B., AND STADLER, P. (2015). Generalized topologies: Hypergraphs, chemical reactions, and biological evolution. In *Advances in Mathematical Chemistry and Applications*, pages 300–328. Bentham Science Publishers.
- FORGY, E. (1965). Cluster analysis of multivariate data: Efficiency versus interpretability of classifications. *Biometrics*, **21**, 768–780.
- FRANK, O. AND HARARY, F. (1982). Cluster inference by using transitivity indices in empirical graphs. *J. Amer. Statist. Assoc.*, **77**, 835–840.
- GHOSHAL, G., ZLATIC, V., CALDARELLI, G., AND NEWMAN, M. (2009). Random hypergraphs and their applications. *Phys. Rev. E*, **79**, 1–10.

- GHOSHDASTIDAR, D. AND DUKKIPATI, A. (2014). Consistency of spectral partitioning of uniform hypergraphs under planted partition model. In *Advances in Neural Information Processing Systems*, volume 27, pages 1–9. Curran Associates, Inc., New York.
- GHOSHDASTIDAR, D. AND DUKKIPATI, A. (2017). Consistency of spectral hypergraph partitioning under planted partition model. *Ann. Stat.*, **45**, 289–315.
- GUILLAUME, J. AND LATAPY, M. (2004). Bipartite structure of all complex networks. *Inf. Process. Lett.*, **90**, 215–221.
- HATCHER, A. (2002). *Algebraic Topology*. Cambridge University Press, Cambridge.
- HOLLAND, P., LASKEY, K., AND LEINHARDT, S. (1983). Stochastic blockmodels: some first steps. *Soc. networks*, **5**, 109–137.
- HUBERT, L. AND ARABIE, P. (1985). Comparing partitions. *J. Classif.*, **2**, 193–218.
- JORDAN, M., GHAHRAMANI, Z., JAAKKOLA, T., AND SAUL, L. (1999). An introduction to variational methods for graphical models. *Mach. Learn.*, **37**, 183–233.
- KAMIŃSKI, B., POULIN, V., PRAŁAT, P., SZUFEL, P., AND THÉBERGE, F. (2019). Clustering via hypergraph modularity. *PLoS One*, **14**, 1–15.
- KE, Z., SHI, F., AND XIA, D. (2020). Community detection for hypergraph networks via regularized tensor power iteration. Technical report, arXiv:1909.06503.
- KRUSKAL, J. (1977). Three-way arrays: rank and uniqueness of trilinear decompositions, with application to arithmetic complexity and statistics. *Linear Algebra Appl.*, **18**, 95–138.
- MACQUEEN, J. (1967). Some methods for classification and analysis of multivariate observations. In *In 5-th Berkeley Symposium on Mathematical Statistics and Probability*, pages 281–297.
- MATIAS, C. AND ROBIN, S. (2014). Modeling heterogeneity in random graphs through latent space models: a selective review. *Esaim Proc. & Surveys*, **47**, 55–74.
- MUYINDA, N., DE BAETS, B., AND RAO, S. (2020). Non-king elimination, intransitive triad interactions, and species coexistence in ecological competition networks. *Theor. Ecol.*, **13**, 385–397.
- NEWMAN, M. (2010). *Networks: An Introduction*. Oxford University Press, New York.

- NEWMAN, M. (2016). Equivalence between modularity optimization and maximum likelihood methods for community detection. *Phys. Rev. E*, **94**, 1–8.
- NG, T. AND MURPHY, T. (2021). Model-based clustering for random hypergraphs. *Adv. Data Anal. Classif.*, **16**, 691–723.
- RODRÍGUEZ, J. (2002). On the laplacian eigenvalues and metric parameters of hypergraphs. *Linear Multilinear Algebra*, **50**, 1–14.
- ROHE, K., CHATTERJEE, S., AND YU, B. (2011). Spectral clustering and the high-dimensional stochastic blockmodel. *Ann. Stat.*, **39**, 1878–1915.
- ROY, S. AND RAVINDRAN, B. (2015). Measuring network centrality using hypergraphs. In *Proceedings of the Second ACM IKDD Conference on Data Sciences*, CoDS '15, page 59–68.
- SHANNON, C. (1948). A mathematical theory of communication. *Bell Labs. Tech. J.*, **27**, 379–423.
- SIMMEL, G. (1950). *The sociology of Georg Simmel*. The free press, New York.
- SINGH, P. AND BARUAH, G. (2021). Higher order interactions and species coexistence. *Theor. Ecol.*, **14**, 71–83.
- SUGANYA, R. AND SHANTHI, R. (2012). Fuzzy c-means algorithm - a review. *Int. J. Sci. Res. Publ.*, **2**, 440–442.
- SWAN, M. AND ZHAN, J. (2021). Clustering hypergraphs via the mapequtation. *IEEE Access*, **9**, 72377–72386.
- TORRES, L., BLEVINS, A., BASSETT, D., AND ELIASSI-RAD, T. (2021). The why, how, and when of representations for complex systems. *SIAM Rev. Soc. Ind. Appl. Math.*, **63**, 435–485.
- TURNBULL, K., LUNAGÓMEZ, S., NEMETH, C., AND AIROLDI, E. (2021). Latent space modelling of hypergraph data. Technical report, arXiv:1909.00472.
- VAZQUEZ, A. (2009). Finding hypergraph communities: a bayesian approach and variational solution. *J. Stat. Mech. Theory Exp.*, **2009**, 1–16.
- ZHOU, D., HUANG, J., AND SCHÖLKOPF, B. (2006). Learning with hypergraphs: Clustering, classification, and embedding. In *Advances in Neural Information Processing Systems*, volume 19, pages 1601–1608. MIT Press.