# Hierarchical estimation of parameters in Bayesian networks[☆]

Laura Azzimonti[**], Giorgio Corani[*], Marco Zaffalon[*]

*IDSIA - SUPSI/USI, Manno, Switzerland*

**Abstract**

A novel approach for parameter estimation in Bayesian networks is presented. The main idea is to introduce a hyper-prior in the Multinomial-Dirichlet model, traditionally used for conditional distribution estimation in Bayesian networks. The resulting hierarchical model jointly estimates different conditional distributions belonging to the same conditional probability table, thus *borrowing statistical strength* from each other. An analytical study of the dependence structure a priori induced by the hierarchical model is performed and an *ad hoc* variational algorithm for fast and accurate inference is derived. The proposed hierarchical model yields a major performance improvement in classification with Bayesian networks compared to traditional models. The proposed variational algorithm reduces by two orders of magnitude the computational time, with the same accuracy in parameter estimation, compared to traditional MCMC methods. Moreover, motivated by a real case study, the hierarchical model is applied to the estimation of Bayesian networks parameters by *borrowing strength* from related domains.

*Keywords:* Hierarchical Bayesian modelling, Bayesian networks, variational inference, multi-domain classification

[☆]Preliminary version of this work was presented in the IEEE International Conference on Data Mining (ICDM), New Orleans 2017.

[*]Corresponding author

[**]Principal corresponding author

*Email addresses:* `laura@idsia.ch` (Laura Azzimonti ), `giorgio@idsia.ch` (Giorgio Corani), `zaffalon@idsia.ch` (Marco Zaffalon)

## 1. Introduction

Bayesian networks (BNs) are probabilistic models that have been successfully applied in a large variety of domains, as reviewed for instance by Darwiche (2010). Almost always, BNs model the relations between discrete variables. Indeed, for discrete data many powerful *inference* algorithms have been developed (Darwiche, 2009, Chap. 6–8) . In the BN terminology, *inference* is the task of computing the posterior distribution of a set of variables, given the observation of another set of variables.

*Learning* a Bayesian network is accomplished in two main steps: *structure learning*, which consists in the identification of the most probable structure, defined by means of a Direct Acyclic Graph (DAG), and *parameter estimation*, which corresponds to the estimation of the conditional distributions in each node of the DAG. In particular, given a DAG, we need to estimate the *conditional probability table* (CPT) of each variable (a node of the DAG), containing the conditional probability distributions of the variable given each possible configuration of its parents. A single conditional distribution is also referred to as a *column* of the CPT.

While there have been many works on *structure learning* in recent years (Campos and Ji, 2011; Yuan et al., 2011; Bartlett and Cussens, 2015; Scanagatta et al., 2016), *parameter estimation* is usually addressed adopting the established approach (Koller and Friedman, 2009, Sec. 17) of performing Bayesian estimation, assuming Multinomial likelihood and Dirichlet prior over the parameters. Each conditional distribution is estimated independently of the others. Moreover, all the states of the conditional distribution are usually assumed to be equally probable *a priori*; hence the maximum likelihood estimates are smoothed towards the uniform distribution.

Under this model, parameter estimation is very efficient, thanks to the conjugacy of Multinomial and Dirichlet distributions. However, parameter estimation is not accurate in sparse data scenarios, which arise frequently even in presence of moderate number of variables due to the exponential number of conditional distributions to be estimated at a node. In these scenarios, instead of smoothing the conditional distributions towards the uniform distribution, it would be more reasonable to learn from all the available data an optimal distribution towards which the estimates should be smoothed. It has already been conjectured (Koller and Friedman, 2009, Sec. 17.5.4) that the estimates could be improved by estimating the different columns jointly rather than independently, since different conditional distributions belonging to the same CPT are expected to be similar to each other.

Yet, parameter estimation becomes in this way much harder, as it is no longer possible to express the posterior distribution of the parameters in closed form.

We propose a novel approach for improving parameter estimation in Bayesian networks, based on hierarchical Bayesian modelling. We thus adopt an approach similar to the one proposed in Kim et al. (2013), based on finite version of the Hierarchical Dirichlet Process (Teh et al., 2006). The proposed generative model assumes that the parameters of different conditional distributions belonging to the same CPT are drawn from a common higher-level distribution, constituted by a mixture of Dirichlet distributions. On the one hand, mixtures of Dirichlet distributions improve the estimation of Multinomial parameters (Casella and Moreno, 2009) with respect to simple Dirichlet distributions. On the other hand, the mixing parameter is a random vector $\boldsymbol{\alpha}$, whose posterior distribution is informative about the distribution towards which the estimates should be smoothed. Such distribution is determined by the values of $\boldsymbol{\alpha}$ that are most probable *a posteriori*. Moreover, the hierarchical model jointly estimates the different conditional distributions belonging to the same CPT, thus *borrowing statistical strength* from each other (Murphy, 2012, Sec. 6.3.3.2). As a result, the hierarchical approach yields a major improvement in parameter estimation, compared to the traditional approach.

The idea of estimating a set of conditional distributions with a hierarchical approach has already brought interesting results in the context of Gaussian Graphical Models (Leday et al., 2017), dynamic Bayesian networks (Grzegorczyk and Husmeier, 2012) and specific Bayesian network classifiers (Petitjean et al., 2018). Yet, an important shortcoming of the hierarchical approach is its computational complexity. The posterior distribution of its parameters does not have a closed form and it should be numerically sampled; this can be time consuming when dealing with Bayesian networks containing a large number of variables, even adopting state-of-the-art Monte Carlo Markov Chain (MCMC) samplers, such as those implemented in Stan (Carpenter et al., 2017). Variational Inference (VI) (Jaakkola and Jordan, 2000; Blei et al., 2017) is instead a technique which efficiently approximates the posterior distribution.

We derive an original VI algorithm for the proposed hierarchical model, which computes the posterior distributions with a negligible decrease of accuracy with respect to MCMC, while reducing the computational time by two orders of magnitude. We also show that, being specifically tailored for our model, it compares favourably to recently proposed algorithms for automatic variational inference (Kucukelbir et al., 2017).

We then extensively assess the impact of the novel parameter estimates based on hierarchical modelling when performing classification with Bayesian networks. We consider a large number of data sets, and we show a consistent improvement of classification accuracy when the hierarchical model is adopted for parameter estimation. Remarkably, the improvement is found even if we favor the traditional Multinomial-Dirichlet by tuning its equivalent sample size, while we run the hierarchical model without any tuning. The advantage over the traditional estimation approach is huge when dealing with very small sample sizes, while it decreases as the sample size increases, as predicted by the theoretical analysis.

We then consider the problem of learning parameters from related data sets; this task is called *meta-analysis* in the statistical literature (Gelman et al., 2014, Chap. 5.6) and *transfer learning* (Pan and Yang, 2010) in the machine learning literature. There is currently no established method for learning parameters of Bayesian networks from related data sets; we show how this task can be easily accomplished thanks to the hierarchical model, by borrowing strength across related data sets. To the best of our knowledge, this is the first principled and scalable approach for learning parameters of Bayesian networks from related data sets. As an application, we consider the case study of a mobile phone application that should classify the user behaviour by means of a small amount of data available for each user. The proposed Bayesian network classifiers, whose parameters are estimated by borrowing strength across users, are much more accurate than their counterparts trained independently on the data set of each user. They are also more accurate than random forest classifiers trained independently on the data set of each user, despite random forests being extremely effective classifiers (Fernández Delgado et al., 2014; Wainberg et al., 2016; Bagnall and Cawley, 2017). Variational inference is fundamental in this application, as it performs parameters estimations borrowing strength across 100 users in minutes rather than in days, as it would be required by MCMC.

The paper is organised as follows. Section 2 introduces both the traditional and the hierarchical models for parameter estimation in Bayesian networks; moreover, it derives analytical properties and exact inference for the proposed hierarchical model. Section 3 describes a variational inference algorithm for approximating the posterior distributions under the hierarchical model and reports some simulation studies for assessing the accuracy of the proposed method. Section 4 describes the parameter estimation setting in Bayesian networks and reports some simulation studies for comparing the performance of hierarchical estimators with respect to traditional estimators in a classification setting. Section 5 shows how to apply the hierarchical model to the problem of parameter learning in Bayesian networks from data coming from related data sets and shows the application to a real case study. Section 6 outlines future research directions. The appendix details exact inference for the proposed hierarchical model and reports all the proofs.

## 2. Multinomial-Dirichlet models

We describe in this section both the traditional Multinomial-Dirichlet (MD) model and the novel hierarchical model for parameter estimation in Bayesian networks. We present both models assuming a discrete children variable $X$ (with values in the set $\mathcal{X}$) and a discrete parent variable $Y$ (with values in the set $\mathcal{Y}$). Even if we present the models in a Bayesian network framework, they are quite general and they could be applied to the estimation of any set of related conditional distributions.

The conditional distribution of $X$ given $Y = y$ is characterised by the parameter vector $\boldsymbol{\theta}_{X|y} \in \mathbf{R}^{|\mathcal{X}|}$, whose generic element $\theta_{x|y}$ represents the probability of $X$ being in state $x$, when its parent variable is in state $y$, i.e., $\theta_{x|y} = P(X = x|Y = y) > 0$. The collection of the conditional probability distributions of $X$ is $\boldsymbol{\theta}_{X|Y} = \left\{\boldsymbol{\theta}_{X|y_1}, \boldsymbol{\theta}_{X|y_2}, \ldots\right\}$, which constitutes the *conditional probability table* (CPT) of $X$. Each conditional distribution $\boldsymbol{\theta}_{X|y}$, with $y \in \mathcal{Y}$, is also referred to as a *column* of the CPT.

Given a data set $D$, the sufficient statistics for the estimation of $\theta_{x|y}$ are the counts $n_{xy}$, i.e., the number of observations with $X = x$ and $Y = y$, where $x \in \mathcal{X}$ and $y \in \mathcal{Y}$. We denote by $n_y = \sum_{x \in \mathcal{X}} n_{xy}$ the counts for the variable $Y$.

### 2.1. Traditional Multinomial-Dirichlet model

The traditional parameter estimation approach in Bayesian networks (Koller and Friedman, 2009, Sec. 17) assumes the prior over each parameter vector $\boldsymbol{\theta}_{X|y}$ to be a Dirichlet distribution. Thus, the generative model is:

$$\begin{aligned} \boldsymbol{\theta}_{X|y}|\boldsymbol{\alpha} &\sim \text{Dirichlet}(\boldsymbol{\alpha}) & y \in \mathcal{Y}, \\ X|Y = y, \boldsymbol{\theta}_{X|y} &\sim \text{Categorical}(\boldsymbol{\theta}_{X|y}) & y \in \mathcal{Y}, \end{aligned} \tag{1}$$

where $\boldsymbol{\alpha} \in \mathbf{R}^{|\mathcal{X}|}$ is a parameter vector such that $\sum_{x \in \mathcal{X}} \alpha_x = s$, where $s \in \mathbb{R}^+$ denotes the prior strength, also called *equivalent sample size*. The most common choice is to set $\alpha_x = (|\mathcal{Y}||\mathcal{X}|)^{-1}$, which is called BDeu (Bayesian Dirichlet equivalent uniform) prior (Koller and Friedman, 2009, Sec. 17). A graphical representation of model (1) is given in Figure 1 (top panel).

Given a data set $D$ of $n$ i.i.d. observations $(x_k, y_k)$ for $k = 1, \dots, n$ drawn from the MD model (1), the posterior distribution of $\boldsymbol{\theta}_{X|y}$ is still a Dirichlet distribution (Gelman et al., 2014). In the following we denote by $\mathbb{E}^D[\cdot]$ the posterior average and by $\mathbb{C}\text{ov}^D(\cdot, \cdot)$ the posterior covariance.

The estimator of $\theta_{x|y}$ is the posterior expectation:

$$\mathbb{E}^D[\theta_{x|y}] = \frac{n_{xy} + \alpha_x}{n_y + s}, \tag{2}$$

which shrinks the maximum likelihood estimates towards the fixed vector $\frac{1}{s}\boldsymbol{\alpha}$, usually set to the uniform distribution. The value of $\boldsymbol{\alpha}$ has a large impact on parameter estimates; however, no prior knowledge is usually available in real problems to set it in an informed way.

The posterior covariance between $\theta_{x|y}$ and $\theta_{x'|y}$ is

$$\mathbb{C}\text{ov}^D(\theta_{x|y}, \theta_{x'|y}) = \frac{\mathbb{E}^D[\theta_{x|y}]\left(\delta_{x,x'} - \mathbb{E}^D[\theta_{x'|y}]\right)}{n_y + s + 1}, \tag{3}$$

where $\delta_{x,x'}$ is a Kronecker delta. The posterior covariance between elements coming from different columns of the CPT is instead zero, i.e., if $y \neq y'$

$$\mathbb{C}\text{ov}^D(\theta_{x|y}, \theta_{x'|y'}) = 0, \tag{4}$$

since different columns of the same CPT are independently estimated.

## *2.2. Hierarchical Multinomial-Dirichlet model*

We generalise the model (1) to a hierarchical Multinomial-Dirichlet (hierarchical MD) model by assuming $\boldsymbol{\alpha}$ to be a latent random vector, with Dirichlet prior:

$$\begin{aligned} \boldsymbol{\alpha}|s, \boldsymbol{\alpha}_0 &\sim s \cdot \text{Dirichlet}(\boldsymbol{\alpha}_0), \\ \boldsymbol{\theta}_{X|y}|\boldsymbol{\alpha} &\sim \text{Dirichlet}(\boldsymbol{\alpha}) & y \in \mathcal{Y}, \\ X|Y = y, \boldsymbol{\theta}_{X|y} &\sim \text{Categorical}(\boldsymbol{\theta}_{X|y}) & y \in \mathcal{Y}, \end{aligned} \tag{5}$$

where the hyper parameters of the model are the equivalent sample size $s \in \mathbb{R}^+$ and the parameter vector $\boldsymbol{\alpha}_0 \in \mathbb{R}^{|\mathcal{X}|}$, whose elements sum to $s_0 \in \mathbb{R}^+$, i.e., $\sum_{x \in \mathcal{X}} [\boldsymbol{\alpha}_0]_x = s_0$. In the following we do not write explicitly the conditioning on the fixed parameters $s$ and $\boldsymbol{\alpha}_0$. The hierarchical model (5) is represented as a probabilistic graphical model in Figure 1 (bottom panel).

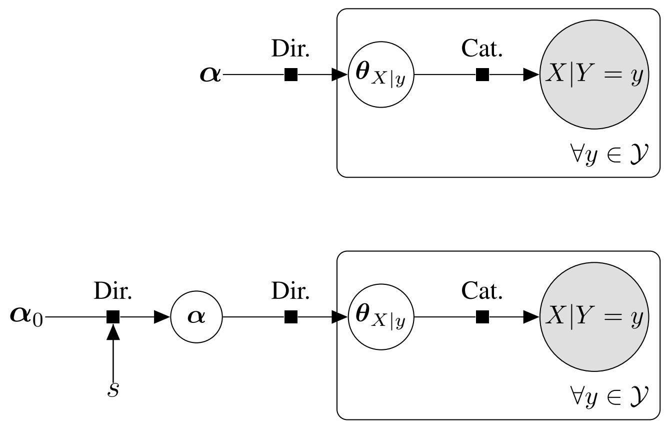

Dir.
Cat.
$\boldsymbol{\alpha}$
$\boldsymbol{\theta}_{X|y}$
$X|Y=y$
$\forall y \in \mathcal{Y}$
Dir.
Dir.
Cat.
$\boldsymbol{\alpha}_0$
$\boldsymbol{\alpha}$
$s$
$\boldsymbol{\theta}_{X|y}$
$X|Y=y$
$\forall y \in \mathcal{Y}$

Figure 1: Directed factor graphs for traditional (top panel) and hierarchical (bottom panel) Multinomial-Dirichlet model. Cat. and Dir. represent respectively Categorical and Dirichlet distributions.

The proposed hierarchical model corresponds to the finite version of the Hierarchical Dirichlet Process (HDP), proposed by Teh et al. (2006). Similar finite HDP models have been introduced in natural language processing to extend the Latent Dirichlet Allocation model by Kim et al. (2013) and in Tree-Augmented Naive (TAN) classifiers by Petitjean et al. (2018). Although less general than HDP, the hierarchical model (5) overcomes the computational efficiency limitations of HDP. We also point out that a simpler model with Binomial likelihood, Beta prior and a non-informative hyper prior on the parameters of the Beta distribution, has been proposed by Gelman et al. (2014). However, the adopted non-informative prior can lead to improper posterior distributions.

According to (5) the prior of $\boldsymbol{\theta}_{X|Y}$ is a *mixture* of Dirichlet distributions:

$$p(\boldsymbol{\theta}_{X|Y}) = \int p(\boldsymbol{\theta}_{X|Y}, \boldsymbol{\alpha}) d\boldsymbol{\alpha} = \int \prod_{y \in \mathcal{Y}} p(\boldsymbol{\theta}_{X|y} | \boldsymbol{\alpha}) p(\boldsymbol{\alpha}) d\boldsymbol{\alpha}.$$

This type of prior is known (Casella and Moreno, 2009) to improve the estimates of the Categorical distribution, compared to the adoption of simple Dirichlet distributions as prior. Notice that the above prior is not factorised with respect to different values of $y$ and for this reason the parameters of different conditional distributions are no longer independent *a priori*.

We now analyse in detail $\mathbb{C}\mathrm{or}\left(\theta_{x|y}, \theta_{x|y'}\right)$ in order to understand the correlations induced by the hierarchical model between the different columns of the same

CPT. The analysis is performed *a priori*.

**Lemma 1.** *A priori, the correlation between $\theta_{x|y}$ and $\theta_{x|y'}$ is*

$$\mathbb{C}or\left(\theta_{x|y},\theta_{x|y'}\right)=\frac{s}{s+s_0+1}. \tag{6}$$

Notice that $s_0$ corresponds to the equivalent sample size of the higher-level prior. Large values of $s_0$ make the hyper-prior sharply peaked around its mean value, thus producing similar values of $\boldsymbol{\alpha}$. In the limit of $s_0$ going to infinity, the random vector $\boldsymbol{\alpha}$ becomes deterministic. Hence the hierarchical model reduces to the traditional MD model (1), and there is no correlation between the parameters of different columns of the same CPT. The hierarchical model is thus more expressive than the traditional one, since it includes the latter as special case.

Instead, the correlation between $\theta_{x|y}$ and $\theta_{x|y'}$ increases as $s$ increases. Indeed, large values of $s$ make the distribution Dirichlet$(\boldsymbol{\alpha})$ sharply peaked around its mean; hence the parameter vectors $\boldsymbol{\theta}_{X|y_1}$, $\boldsymbol{\theta}_{X|y_2}$, ... sampled from such distribution are similar to each other. In the limit of $s$ going to infinity, the same parameter vector is sampled for all the columns of the CPT, implying a correlation of 1 between the parameters of the different columns.

### *2.2.1. Posterior distribution of the parameters*

Given a data set $D$ of $n$ i.i.d. observations $(x_k, y_k)$ for $k=1,\ldots,n$, the posterior distribution of the parameters is a mixture of Dirichlet distributions:

$$\begin{aligned}p(\boldsymbol{\theta}_{X|Y}|D) &= \int\prod_{y\in\mathcal{Y}}p(\boldsymbol{\theta}_{X|y}|\boldsymbol{\alpha},D)p(\boldsymbol{\alpha}|D)d\boldsymbol{\alpha} \qquad (7)\\ &\propto \int\frac{\Gamma(s)}{\prod_{x\in\mathcal{X}}\Gamma(\alpha_x)}\prod_{\substack{x\in\mathcal{X}\\ y\in\mathcal{Y}}}\theta_{x|y}^{n_{xy}+\alpha_x-1}\left(\frac{\alpha_x}{s}\right)^{[\boldsymbol{\alpha}_0]_x-1}\frac{d\boldsymbol{\alpha}}{s},\end{aligned}$$

where the mixing is controlled by the posterior distribution of $\boldsymbol{\alpha}$. The values of $\boldsymbol{\alpha}$ that are more probable *a posteriori* have a higher weight in determining the posterior distribution of $\boldsymbol{\theta}_{X|Y}$. Although the posterior distribution (7) is not analytically tractable, we can compactly express its posterior expectation and covariance. These quantities are defined in terms of posterior expectation and covariance of $\boldsymbol{\alpha}$, respectively denoted as $\mathbb{E}^D[\boldsymbol{\alpha}]$ and $\mathbb{C}\mathrm{ov}^D(\alpha_x,\alpha_{x'})$.

**Lemma 2.** *The posterior average of $\theta_{x|y}$ is:*

$$\mathbb{E}^D\left[\theta_{x|y}\right] = \frac{n_{xy} + \mathbb{E}^D\left[\alpha_x\right]}{n_y + s}, \tag{8}$$

*while the posterior covariance between $\theta_{x|y}$ and $\theta_{x'|y}$ is:*

$$\mathbb{C}ov^D\big(\theta_{x|y}, \theta_{x'|y}\big) = \frac{\mathbb{C}ov^D\left(\alpha_x, \alpha_{x'}\right)}{(n_y + s)(n_y + s + 1)} + \frac{\mathbb{E}^D\left[\theta_{x|y}\right]\left(\delta_{xx'} - \mathbb{E}^D\left[\theta_{x'|y}\right]\right)}{n_y + s + 1}, \tag{9}$$

*and the posterior covariance between $\theta_{x|y}$ and $\theta_{x'|y'}$, with $y \neq y'$, is:*

$$\mathbb{C}ov^D\big(\theta_{x|y}, \theta_{x'|y'}\big) = \frac{\mathbb{C}ov^D\left(\alpha_x, \alpha_{x'}\right)}{(n_y + s)(n_{y'} + s)}. \tag{10}$$

Notice that the posterior expectation $\mathbb{E}^D\left[\theta_{x|y}\right]$ smooths the maximum likelihood estimates towards the posterior expectation of $\frac{1}{s}\alpha_x$. The posterior expectation of $\alpha_x$ depends on all the available data (not only on those conditioned on $Y = y$) and thus it can be reliably estimated by sharing information between different columns of the same CPT, i.e., *borrowing strength* between columns of the CPT. The posterior expectation shrinks thus the maximum likelihood estimates towards a reliable distribution learnt from data, rather than towards a fixed *a priori* distribution, as in the traditional model. Moreover, the covariances (9) and (10) contain an additional term that models the relation between different columns of the CPT, compared to the covariances of the MD model (3) and (4).

It is clear from (7) that the posterior distribution of the parameters is not factorised over different values of $y$; hence the conditional distributions $\boldsymbol{\theta}_{X|y}$ referring to different values of $y$ are jointly estimated, rather than independently estimated as usual. Considering in (10) the specific case of $x = x'$, we obtain the covariance between elements in the same row of the CPT, i.e.,

$$\mathbb{C}\mathrm{ov}^D\left(\theta_{x|y}, \theta_{x|y'}\right) = \frac{\mathbb{V}\mathrm{ar}^D(\alpha_x)}{(n_y + s)(n_{y'} + s)}.$$

It is clear that $\mathbb{C}\mathrm{ov}^D\left(\theta_{x|y}, \theta_{x|y'}\right)$ is non-negative, thus there is always a positive association between $\theta_{x|y'}$ and $\theta_{x|y}$. As the number of observations goes to infinity, the covariance vanishes: asymptotically the parameter estimates for the different conditional distributions are independent, as the effect of the prior is overwhelmed by the data.

### 2.2.2. *Posterior moments of $\boldsymbol{\alpha}$*

The posterior moments of $\boldsymbol{\alpha}$ cannot be computed analytically since the associated posterior distribution is

$$p(\boldsymbol{\alpha}|D) \propto \prod_{y\in\mathcal{Y}} \left( \frac{\Gamma(s) \prod_{x\in\mathcal{X}} \Gamma(\alpha_x + n_{xy})}{\Gamma(s+n_y) \prod_{x\in\mathcal{X}} \Gamma(\alpha_x)} \frac{\alpha_x^{[\boldsymbol{\alpha}_0]_x - 1}}{s^{[\boldsymbol{\alpha}_0]_x}} \right).$$

However, the following lemma states a general result for computing the first two moments of $\boldsymbol{\alpha}$. The generalisation to any posterior moment of $\boldsymbol{\alpha}$ is stated in Appendix A.3.

**Lemma 3.** *Under the assumptions of model* (5) *and setting for simplicity* $[\boldsymbol{\alpha}_0]_x = 1\ \forall x \in \mathcal{X}$*, the element* $x'$ *of the posterior average vector* $\mathbb{E}^D[\boldsymbol{\alpha}]$ *is*

$$\mathbb{E}^D[\alpha_{x'}] = \gamma \int \prod_{\substack{x\in\mathcal{X}\\ y\in\mathcal{Y}\\ s.t.\ n_{xy}>0}} \prod_{m=1}^{n_{xy}} (\alpha_x + m - 1) \left(\frac{\alpha_x}{s}\right)^{\delta_{x,x'}} \frac{d\boldsymbol{\alpha}}{s},$$

*where* $\delta_{x,x'}$ *is a Kronecker delta and* $\gamma$ *is a proportionality constant such that*

$$\gamma^{-1} = \int \prod_{\substack{x\in\mathcal{X}\\ y\in\mathcal{Y}\\ s.t.\ n_{xy}>0}} \prod_{m=1}^{n_{xy}} (\alpha_x + m - 1) \frac{d\boldsymbol{\alpha}}{s}.$$

*The element* $(x', x'')$ *of* $\mathbb{E}^D[\alpha_{x'}\alpha_{x''}]$ *is*

$$\mathbb{E}^D[\alpha_{x'}\alpha_{x''}] = \gamma \int \prod_{\substack{x\in\mathcal{X}\\ y\in\mathcal{Y}\\ s.t.\ n_{xy}>0}} \prod_{m=1}^{n_{xy}} (\alpha_x + m - 1) \left(\frac{\alpha_x}{s}\right)^{\delta_{x,x'}+\delta_{x,x''}} \frac{d\boldsymbol{\alpha}}{s}.$$

All the integrals in Lemma 3 are multiple integrals computed with respect to the $|\mathcal{X}|$ elements of vector $\frac{1}{s}\boldsymbol{\alpha}$, such that $\sum_{x\in\mathcal{X}} \frac{1}{s}\alpha_x = 1$. The space of integration is thus the standard $|\mathcal{X}|$-simplex.

These integrals cannot be computed analytically, unless the integration space is low dimensional and the polynomial degree is very small. In a general case, it is possible to resort to symbolic calculus to compute them. We adopt such exact solution in order to assess the accuracy of the estimates yielded by variational inference on some low-dimensional examples. Specifically we make use of

Python's symbolic calculus library `sympy`. The numerical computation, detailed in Appendix A.3, consists in an iterative algorithm that goes through all the layers of the multiple integral, integrating a polynomial function in one variable at a time. This approach allows to compute exactly the posterior moments of $\boldsymbol{\alpha}$; however, it has some computational limitations. As the number of observations increases, the numerical computation of the integral becomes more expensive due to the increase in polynomial degree of the integrand. The integration is very expensive also in the case of high number of states for the $X$ variable, due to the increase in dimension of the integration space. For this reason, in the following we propose a variational inference algorithm in order to compute a fast approximation of the posterior distributions.

## 3. Variational inference

Since the posterior distribution for $\boldsymbol{\theta}_{X|Y}$ is not analytically tractable, a possible solution consists in approximating it numerically. A traditional approximation is obtained by means of Markov-Chain Monte Carlo (MCMC) methods. Yet, the MCMC approximations can be too time-consuming when dealing with large models, as we will show in Section 3.2. We thus adopt Variational Inference (VI) (Jordan et al., 1999; Wainwright and Jordan, 2008; Blei et al., 2017) in order to efficiently approximate the joint posterior distribution of the parameters.

Specifically, we approximate $p(\boldsymbol{\theta}_{X|Y}, \boldsymbol{\alpha}|D)$ by means of the factorised distribution

$$q(\boldsymbol{\theta}_{X|Y}, \boldsymbol{\alpha}|\boldsymbol{\nu}, \tau, \boldsymbol{\kappa}) = \prod_{y\in\mathcal{Y}} q(\boldsymbol{\theta}_{X|y}|\boldsymbol{\nu}_y) q(\boldsymbol{\alpha}|\tau, \boldsymbol{\kappa}), \tag{11}$$

which treats $\boldsymbol{\theta}_{X|y}$ and $\boldsymbol{\alpha}$ as independent random variables. $\boldsymbol{\nu}_y \in \mathbb{R}^{|\mathcal{X}|}$ for $y \in \mathcal{Y}$, $\tau \in \mathbb{R}^+$ and $\boldsymbol{\kappa} \in \mathbb{R}^{|\mathcal{X}|}$, such that $\sum_{x\in\mathcal{X}} \kappa_x = 1$, are the parameters of the following variational model:

$$\begin{aligned} \boldsymbol{\alpha}|s, \tau, \boldsymbol{\kappa} &\sim s \cdot \text{Dirichlet}(\tau\boldsymbol{\kappa}), & \\ \boldsymbol{\theta}_{X|y}|\boldsymbol{\nu}_y &\sim \text{Dirichlet}(\boldsymbol{\nu}_y) & y \in \mathcal{Y}, \\ X|Y = y, \boldsymbol{\theta}_{X|y} &\sim \text{Categorical}(\boldsymbol{\theta}_{X|y}) & y \in \mathcal{Y}. \end{aligned} \tag{12}$$

The variational model (12) is similar to the original hierarchical MD model (5), but it removes the dependence between $\boldsymbol{\theta}_{X|y}$ and $\boldsymbol{\alpha}$. The dependence between $\boldsymbol{\theta}_{X|y}$ and $\boldsymbol{\alpha}$ is later recovered by making the variational parameters $\boldsymbol{\nu}_y$, $\tau$ and $\boldsymbol{\kappa}$ dependent. Notice that we used for convenience a non-standard notation for the

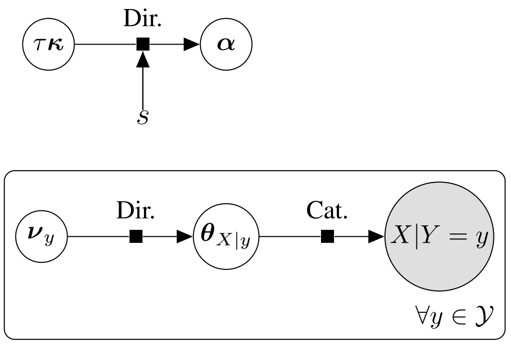

Dir.
$\tau\boldsymbol{\kappa}$
$\boldsymbol{\alpha}$
$s$
Dir.
Cat.
$\boldsymbol{\nu}_y$
$\boldsymbol{\theta}_{X|y}$
$X|Y=y$
$\forall y \in \mathcal{Y}$

Figure 2: Directed factor graphs for variational model (12). Cat. and Dir. represent respectively Categorical and Dirichlet distributions.

Dirichlet distribution of the parameter vector $\boldsymbol{\alpha}$. The proposed variational model is shown in Figure 2.

For ease of notation, in the following we drop the dependence of $q$ on the variational parameters, i.e., $q(\boldsymbol{\theta}_{X|y}) = q(\boldsymbol{\theta}_{X|y}|\boldsymbol{\nu}_y)$ and $q(\boldsymbol{\alpha}) = q(\boldsymbol{\alpha}|\tau, \boldsymbol{\kappa})$. The joint variational distribution $q(\boldsymbol{\theta}_{X|Y}, \boldsymbol{\alpha})$ is a function of the variational parameters $\tau$, $\boldsymbol{\kappa}$ and $\boldsymbol{\nu}_y$ for $y \in \mathcal{Y}$. These parameters are estimated from the available data $D$ in order to minimise the Kullback-Leibler (KL) divergence between the exact posterior distribution $p(\boldsymbol{\theta}_{X|Y}, \boldsymbol{\alpha}|D)$ and the variational approximation $q(\boldsymbol{\theta}_{X|Y}, \boldsymbol{\alpha})$.

It is well known that minimising the KL divergence is equivalent to maximising the evidence lower bound (ELBO) $\mathcal{L}$ for the marginal likelihood $\log(p(D))$ (Blei et al., 2017):

$$\mathcal{L} = \mathbb{E}_q \left[\log\left(p(D, \boldsymbol{\theta}_{X|Y}, \boldsymbol{\alpha})\right)\right] - \mathbb{E}_q \left[\log\left(q(\boldsymbol{\theta}_{X|Y}, \boldsymbol{\alpha})\right)\right], \tag{13}$$

where $\mathbb{E}_q[\cdot]$ represents the mean with respect to the variational distribution $q$. Indeed, the KL divergence between the variational and the real posterior distributions corresponds to the difference between $\log(p(D))$ and $\mathcal{L}$, i.e.,

$$\log(p(D)) - \mathcal{L} = \mathrm{KL}(q(\boldsymbol{\theta}_{X|Y}, \boldsymbol{\alpha}) \,||\, p(\boldsymbol{\theta}_{X|Y}, \boldsymbol{\alpha}|D)).$$

Thus, the maximisation of $\mathcal{L}$ corresponds to the minimisation of the KL divergence, since the KL divergence is a positive quantity. On the other hand, the minimisation of the KL divergence corresponds to the approximation of $\log(p(D))$ by means of a tight lower bound.

## *3.1. Variational inference algorithm*

In the specific case of the hierarchical MD model (5) with approximating model (12) it is not possible to write explicitly the ELBO $\mathcal{L}$ as an analytical function of the variational parameters. Thus, we resort to a tight lower approximation of $\mathcal{L}$, named $\tilde{\mathcal{L}}$, which can be written as an analytical function of the parameters. The quantity $\tilde{\mathcal{L}}$, which satisfies

$$\log(p(D)) \geq \mathcal{L} \geq \tilde{\mathcal{L}},$$

can be written as

$$\begin{aligned}\tilde{\mathcal{L}} = & \sum_{\substack{y\in\mathcal{Y}\\x\in\mathcal{X}}} n_{xy}(\psi(\nu_{xy}) - \psi(\nu_{\cdot y})) + \sum_{\substack{y\in\mathcal{Y}\\x\in\mathcal{X}}} (s\kappa_x - 1)\Big(\psi(\nu_{xy}) - \psi(\nu_{\cdot y})\Big) + \\ & + \sum_{\substack{y\in\mathcal{Y}\\x\in\mathcal{X}}} \log\Gamma(\nu_{xy}) - \sum_{\substack{y\in\mathcal{Y}\\x\in\mathcal{X}}} (\nu_{xy} - 1)(\psi(\nu_{xy}) - \psi(\nu_{\cdot y})) - \sum_{y\in\mathcal{Y}} \log\Gamma(\nu_{\cdot y}) + \\ & - |\mathcal{Y}| \sum_{x\in\mathcal{X}} \log\Gamma(s\kappa_x) + |\mathcal{Y}| \sum_{x\in\mathcal{X}} (s\kappa_x - 1)(\log(\kappa_x) - \psi(\tau\kappa_x) + \psi(\tau)) + \\ & - \sum_{x\in\mathcal{X}} \log\Gamma([\boldsymbol{\alpha}_0]_x) + \sum_{x\in\mathcal{X}} \log\Gamma(\tau\kappa_x) + \sum_{x\in\mathcal{X}} ([\boldsymbol{\alpha}_0]_x - \tau\kappa_x)(\psi(\tau\kappa_x) - \psi(\tau)) + \\ & + |\mathcal{Y}| \log\Gamma(s) - \frac{s}{\tau}|\mathcal{Y}|\left(|\mathcal{X}| - 1\right) + \log\Gamma\left(s_0\right) - \log\Gamma(\tau),\end{aligned}$$

where $\psi(\cdot)$ is the digamma function, derivative of the log Gamma function. The derivation of the lower bound $\tilde{\mathcal{L}}$ is in Appendix B.1.

In order to estimate the variational parameters $\boldsymbol{\nu}_y$, $\tau$ and $\boldsymbol{\kappa}$, we then maximise $\tilde{\mathcal{L}}$. The proposed optimisation method rests on an iterative fixed-point method, which alternates between an analytical optimisation with respect to $\boldsymbol{\nu}_y$ and a numerical optimisation with respect to $\tau$ and $\boldsymbol{\kappa}$. It is similar to the variational inference proposed by Kim et al. (2013).

The iterative structure of the variational inference method is detailed in the following algorithm.

**Algorithm 1.** *Variational algorithm for inference under the hierarchical Multinomial-Dirichlet model.*
*Fix starting values for the parameters $\boldsymbol{\nu}_y$, for $y \in \mathcal{Y}$, $\tau$, $\boldsymbol{\kappa}$, i.e., set $\hat{\boldsymbol{\nu}}_y^0$, $\hat{\tau}^0$, $\hat{\boldsymbol{\kappa}}^0$;*
*while $K <$ maxiter$_1$ or tolerance $>$ tol$_1$:*

1. *update the estimate of $\boldsymbol{\nu}_y$ $\forall y \in \mathcal{Y}$, i.e., compute $\hat{\boldsymbol{\nu}}_y^{K+1}$ with $\boldsymbol{\kappa} = \hat{\boldsymbol{\kappa}}^K$ and $\tau = \hat{\tau}^K$;*

```
2. fix starting values for inner optimisation, i.e., $\hat{\tau}_0^{K+1}$=$\hat{\tau}^K$, $\hat{\boldsymbol{\kappa}}_0^{K+1} = \hat{\boldsymbol{\kappa}}^K$;
3. while $k <$ maxiter$_2$ or tolerance $>$ tol$_2$:
    (a) update the estimate of $\tau$, i.e., compute $\hat{\tau}_{k+1}^{K+1}$ with $\boldsymbol{\kappa} = \hat{\boldsymbol{\kappa}}_k^{K+1}$ and $\boldsymbol{\nu}_y = \hat{\boldsymbol{\nu}}_y^{K+1}$;
    (b) update the estimate of $\boldsymbol{\kappa}$, i.e., compute $\hat{\boldsymbol{\kappa}}_{k+1}^{K+1}$ with $\tau = \hat{\tau}_{k+1}^{K+1}$ and $\boldsymbol{\nu}_y = \hat{\boldsymbol{\nu}}_y^{K+1}$;
    (c) increase the iterator $k$;
4. define $\hat{\tau}^{K+1}$ and $\hat{\boldsymbol{\kappa}}^{K+1}$ as the estimates for $\tau$ and $\boldsymbol{\kappa}$ obtained in the inner loop;
5. increase the iterator $K$;
define $\hat{\boldsymbol{\nu}}_y$ for $y \in \mathcal{Y}$, $\hat{\tau}$ and $\hat{\boldsymbol{\kappa}}$ as the estimates for $\boldsymbol{\nu}_y$, $\tau$ and $\boldsymbol{\kappa}$ obtained in the outer loop.
```

The optimisation with respect to $\boldsymbol{\nu}_y$ is derived analytically. It provides, given a value for the parameter vector $\boldsymbol{\kappa}$, an estimate $\hat{\nu}_{xy}$ for the parameter $\nu_{xy}$, element $x$ of parameter vector $\boldsymbol{\nu}_y$:

$$\hat{\nu}_{xy} = n_{xy} + s\kappa_x. \tag{14}$$

Instead, the optimisation with respect to the parameters $\tau$ and $\boldsymbol{\kappa}$ is performed by means of a fixed-point method, which alternates the optimisation of $\tilde{\mathcal{L}}$ with respect to $\tau$ and the optimisation of the same quantity with respect to $\boldsymbol{\kappa}$. Since an analytical solution for the optimisation problems is not available, we propose solving the two optimisation problems by means of a Newton algorithm.

The analytical optimisation of $\tilde{\mathcal{L}}$ with respect to $\boldsymbol{\nu}_y$ is derived in Appendix B.2, while the numerical optimisations with respect to $\tau$ and $\boldsymbol{\kappa}$ are detailed in Appendix B.3.

Notice that the estimated variational parameters are functions of the observations $D$. Thus, the variational distribution varies as well as a function of the observations. Moreover, the estimated variational parameters $\hat{\boldsymbol{\nu}}_y$, $\hat{\tau}$ and $\hat{\boldsymbol{\kappa}}$ are mutually dependent. This dependence partially recovers the dependence between $\boldsymbol{\theta}_{X|y}$ and $\boldsymbol{\alpha}$. These dependences are particularly clear in the closed form update for the parameter $\nu_{xy}$. Indeed, the update formula (14) corresponds exactly to the numerator of the posterior expectation of $\theta_{x|y}$ in (8), where $\mathbb{E}_q\left[\alpha_x\right]$ is approximated by means of $s\hat{\kappa}_x$. The posterior expectation of $\theta_{x|y}$ is thus approximated by means of the average of a Dirichlet distribution with parameter $\hat{\boldsymbol{\nu}}_y$, i.e.,

$$\mathbb{E}_q\left[\theta_{x|y}\right] = \frac{\hat{\nu}_{xy}}{\sum_{x\in\mathcal{X}}\hat{\nu}_{xy}} = \frac{n_{xy} + s\hat{\kappa}_x}{n_y + s},$$

while the posterior expectation of $\alpha_x$ is approximated by means of the average of a Dirichlet distribution with parameter $\hat{\tau}\hat{\boldsymbol{\kappa}}$, i.e., $\mathbb{E}_q[\alpha_x] = s\hat{\kappa}_x$.

## *3.2. Experimental comparison of variational inference and MCMC*

We perform a simulation study to assess the performance of the proposed variational inference algorithm with respect to Markov Chain Monte Carlo. The comparison is performed in terms of accuracy in parameter estimation and computational time. Both the methods provide parameter estimates based on the proposed hierarchical model. The variational inference algorithm of Section 3.1 is implemented in R and is available at `https://ipg.idsia.ch/software.php?id=139`. The MCMC hierarchical model is implemented in `stan` (Carpenter et al., 2017) in order to have a state-of-the-art sampler as a term of comparison. Specifically, a Hamiltonian Monte Carlo algorithm is used in `stan` to perform MCMC. We run all experiments in R, using the `rstan` package (Carpenter et al., 2017) as interface between `R` and `stan`, with `rstan` default options. We expect VI to be faster and computationally more efficient than MCMC. However, we are interested in understanding if the computational efficiency of VI causes any major loss of accuracy. For this reason in the following experiments we run both VI and MCMC till convergence and we analyse the accuracy of the estimates. The stopping criteria for VI are: number of iterations greater than 1000 and tolerance lower than $10^{-6}$ . The stopping criteria for MCMC are the `rstan` default options.

We generate observations by sampling from the hierarchical model (5), in different settings. We consider all the possible combinations of the number of states $|\mathcal{X}|$ and the number of conditioning states $|\mathcal{Y}|$, with $|\mathcal{X}| \in \{2, 4, 6, 8\}$ and $|\mathcal{Y}| \in \{2, 4, 6, 8\}$. For each combination of $|\mathcal{X}|$ and $|\mathcal{Y}|$ we generate data sets with size $n \in \{20, 40, 80, 160, 320, 640\}$.

In the following we denote with a tilde the sampled parameters, which correspond to the true underlying parameters of the model. In every repetition of the experiment we sample the data as follows:

1) we sample $\tilde{\boldsymbol{\alpha}}$ as $s = |\mathcal{X}|$ times a Dirichlet distribution with parameter $\boldsymbol{\alpha}_0 = \mathbf{1}_{1\times|\mathcal{X}|}$,
2) we sample $\tilde{\boldsymbol{\theta}}_{X|y}$, for each $y \in \mathcal{Y}$, from a Dirichlet distribution with parameter $\tilde{\boldsymbol{\alpha}}$,
3) we sample the observations from a Categorical distribution with parameters $\tilde{\boldsymbol{\theta}}_{X|Y}$.

We repeat the data sampling and the estimation procedure 10 times for each combination of $|\mathcal{X}|$, $|\mathcal{Y}|$ and $n$. For the hierarchical model we set $s = |\mathcal{X}|$ and $\boldsymbol{\alpha}_0 = \mathbf{1}_{1\times|\mathcal{X}|}$; we then use the posterior expectation of $\boldsymbol{\theta}_{X|Y}$ and $\boldsymbol{\alpha}$ as parameter estimates.

We consider the following measures of accuracy to assess the goodness of an estimator $\hat{\boldsymbol{\theta}}_{X|Y}$:

- the mean squared error (MSE), computed as

$$\mathrm{MSE}(\hat{\boldsymbol{\theta}}_{X|Y}) = \frac{1}{|\mathcal{X}||\mathcal{Y}|}\sum_{\substack{x\in\mathcal{X}\\ y\in\mathcal{Y}}}\left(\hat{\theta}^i_{x|y} - \tilde{\theta}_{x|y}\right)^2, \tag{15}$$

- the Kullback-Leibler divergence (KL), computed as

$$\mathrm{KL}(\hat{\boldsymbol{\theta}}_{X|Y}) = \frac{1}{|\mathcal{Y}|}\mathrm{KL}(\tilde{\boldsymbol{\theta}}_{X|y}, \hat{\boldsymbol{\theta}}_{X|y}) = \frac{1}{|\mathcal{Y}|}\sum_{\substack{x\in\mathcal{X}\\ y\in\mathcal{Y}}}\tilde{\theta}^i_{x|y}\log\left(\tilde{\theta}^i_{x|y}/\hat{\theta}_{x|y}\right),$$

- the Hellinger distance (H), computed as

$$\mathrm{H}(\hat{\boldsymbol{\theta}}_{X|Y}) = \mathrm{H}(\hat{\boldsymbol{\theta}}_{X|y}, \tilde{\boldsymbol{\theta}}_{X|y}) = \frac{1}{\sqrt{2}}\sqrt{\sum_{\substack{x\in\mathcal{X}\\ y\in\mathcal{Y}}}\left(\sqrt{\hat{\theta}^i_{x|y}} - \sqrt{\tilde{\theta}_{x|y}}\right)^2},$$

where $\hat{\theta}^i_{x|y}$ represents the estimate of $\theta_{x|y}$ in the $i$-th repetition of the experiment and $\tilde{\theta}_{x|y}$ represents the true underlying parameter.

Figure 3 shows that the MSE, KL and H obtained by means of VI and MCMC are equivalent in every setting. There is therefore no loss of accuracy due to the adoption of the proposed VI method instead of MCMC. As expected, for both algorithms MSE, KL and H decrease with the number of observations and increases with $|\mathcal{Y}|$. The same qualitative behaviour has been observed for different values of $|\mathcal{X}|$.

The accuracy being equal, the speedup allowed by VI is instead of paramount importance. The VI algorithm converges in tens of iterations in all the performed simulations: the estimated lower bound reaches a plateau in few iterations (less than 5) and only minor adjustments are performed in the following iterations. The computational times decrease of about two orders of magnitude with respect to

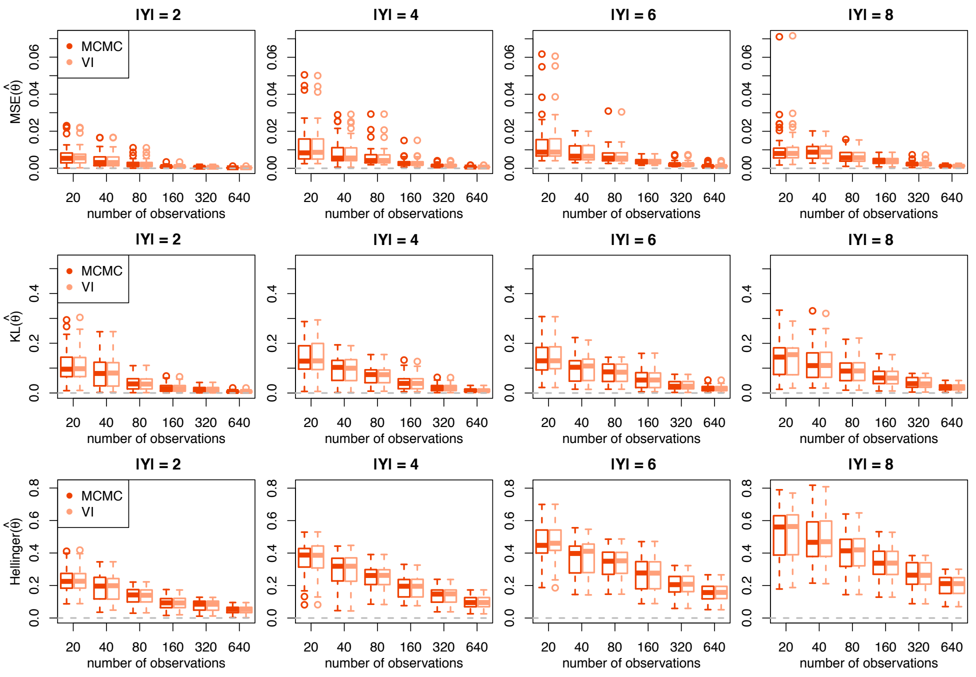


Figure 3: Boxplots of MSE (top panel), KL divergence (central panel) and Hellinger distance (bottom panel) for MCMC (red) and VI (orange) with different dimension of the conditioning set ($|\mathcal{Y}| = 2, 4, 6, 8$). In each plot all the results obtained with different numbers of states for the child variable are summarised. In general, there is no noticeable difference between MCMC and VI.

MCMC, as shown in Figure 4. For instance the average time required by MCMC is between 10 and 60 seconds, respectively for $|\mathcal{X}| = 2$ and $|\mathcal{X}| = 8$; it drops respectively to 0.04 and 0.5 seconds for VI. The dimension of the conditioning set $|\mathcal{Y}|$ does not affect the computational times. Such efficiency allows to deal with a large number of variables and conditioning states, as for instance in the application presented in Section 5.

For small values of $n$ (20, 40, 80, 160), we compare the parameter estimates obtained by means of VI also with respect to the posterior expectation computed by means of polynomial integration, as described in Appendix A.3. The obtained mean squared errors are on average equal to $10^{-5}$. Comparing to the automatic variational inference (Kucukelbir et al., 2017) implemented in Stan, we obtain on average a relative improvement of about 35% in terms of MSE (corresponding to an absolute improvement of $3 \cdot 10^{-6}$), 20% in terms of KL , 8% in terms of H and 30% in terms of computing time.

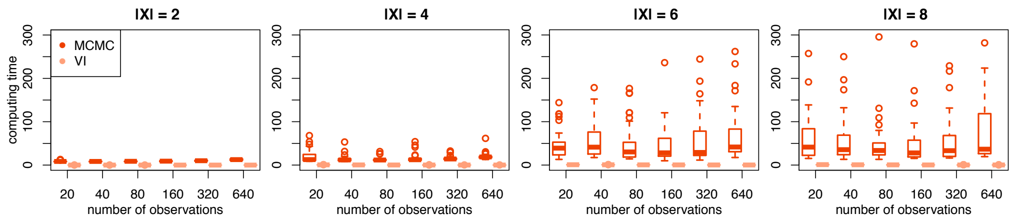


Figure 4: Boxplots of computing time (seconds) for MCMC (red) and VI (orange) with different number of states for the child variable ($|\mathcal{X}| = 2, 4, 6, 8$). In each plot all the results obtained with different dimensions of the conditioning set are summarised. VI provides a major reduction of computing times especially for large number of observations and/or large values of $|\mathcal{X}|$.

We thus conclude that the proposed VI method provides accurate and reliable parameter estimates.

## 4. Learning parameters in Bayesian networks

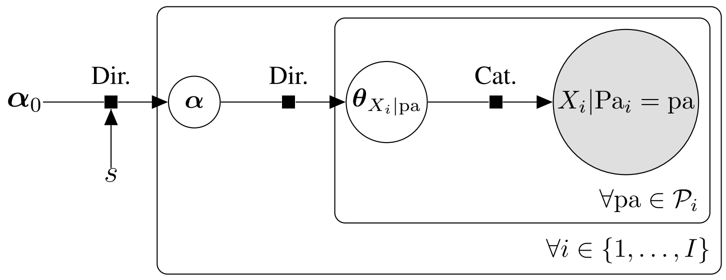
We use the hierarchical model to estimate the conditional probability tables of a Bayesian network over a set of discrete random variables $\boldsymbol{X} = \{X_1, \dots, X_I\}$. We assume that the Bayesian network structure is known and we apply the hierarchical model separately on each single node $X_i \in \boldsymbol{X}$. We choose $X = X_i$ for $i \in 1, \dots, I$ and $Y$ to be a discrete variable representing the joint states of $\mathrm{Pa}_i$, the parent set of $X_i$. The parameter vector $\boldsymbol{\theta}_{X|y}$, for $y \in \mathcal{Y}$, corresponds to the columns of the CPT associated to the variables $X_i$ and $\mathrm{Pa}_i$, i.e., $\boldsymbol{\theta}_{X|y} = \boldsymbol{\theta}_{X_i|\mathrm{pa}}$, for $i = 1, \dots, I$ and $\mathrm{pa}$ in the set of the joint states of $\mathrm{Pa}_i$. A graphical representation of the model is shown in Figure 5.

We adopt the posterior expectation of $\boldsymbol{\theta}_{X|Y}$ as estimator, i.e., $\hat{\theta}_{x|y} = \mathbb{E}^D\left[\theta_{x|y}\right]$. The estimator is hence given by (2) for the traditional inference under BDeu and by (8) for inference under the hierarchical model.

The hierarchical model is suitable for a general Bayesian network; in the following we focus however on parameter estimation for Bayesian network classifiers, in order to assess the quality of parameter estimation by means of simple measures, like e.g., accuracy and area under the ROC curve. The aim of the following experiments is thus to show the parameter estimation improvement yielded by the proposed hierarchical method with respect to traditional approaches.

### *4.1. Experiments in classification with Bayesian networks*

In the following we consider a classification problem. Without loss of generality, we split the set of random variables $\boldsymbol{X}$ into $k = I - 1$ feature variables



$\boldsymbol{\alpha}_0$
Dir.
$s$
$\boldsymbol{\alpha}$
Dir.
$\boldsymbol{\theta}_{X_i|\mathrm{pa}}$
Cat.
$X_i|\mathrm{Pa}_i = \mathrm{pa}$
$\forall \mathrm{pa} \in \mathcal{P}_i$
$\forall i \in \{1, \ldots, I\}$

Figure 5: Hierarchical model applied to parameter learning in Bayesian networks represented by means of a directed factor graph. $\mathcal{P}_i$ represents the set of the joint states of $\mathrm{Pa}_i$. Cat. and Dir. represent respectively Categorical and Dirichlet distributions.

$\{X_1, \ldots, X_k\}$ and a class variable $C = X_I$.

One of the simplest Bayesian network classifiers is the naive Bayes, which assumes the stochastic independence of the features given the class. This assumption is represented by a DAG whose arcs connect the class to each feature, as shown in Figure 6 (left panel). Naive Bayes is a good model if the goal is simple classification, without any need of accurate probability estimates, i.e., under 0-1 loss (Friedman, 1997; Domingos and Pazzani, 1997).

More accurate posterior probabilities can be estimated by relaxing the assumption of conditional independence. This can be done for instance by adopting a Tree-Augmented Naive classifier (TAN), whose graph contains a tree that links the feature variables, in addition to the edges already present in the naive Bayes graph, as shown in Figure 6 (right panel). The algorithm for learning the TAN structure has been presented by Friedman et al. (1997).

In the following experimental studies we compare the performance of the TAN classifier when its parameters are estimated by means of the hierarchical model or by the traditional ones. We consider 57 data sets from UCI Machine Learning Repository[1] and the Weka data sets page[2]; they are listed, together with information on the number of instances, attributes, states of the class variable and missing values, in Table B.1. We impute missing values with the median (for continuous variables) and the mode (for discrete variables). We then discretise the numerical variables into five equal-frequency bins.

[1] `https://archive.ics.uci.edu/ml/index.php`

[2] `https://www.cs.waikato.ac.nz/ml/weka/datasets.html`

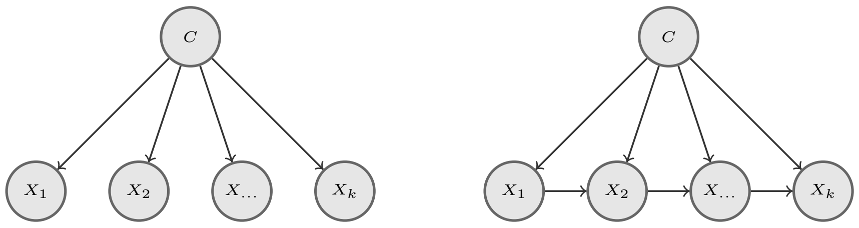


Figure 6: Structure of naive Bayes (left panel) and TAN (right panel). The TAN structure contains the same arcs of naive Bayes, with an additional tree that connects the features.

For each data set we first learn the maximum-likelihood TAN structure using all the available samples under the BDeu assumption; we adopt the `R` package `bnlearn` (Scutari, 2010), which learns the TAN structure according to the algorithm of Friedman et al. (1997). We then keep fixed this structure for all the experiments performed with the same data set and we estimate the parameters of TAN using different methods.

Then, for each data set and for each $n \in \{20, 40, 80, 160, 320, 640\}$, we repeat 10 times the procedure: 1) create a training set by randomly sampling $n$ observations from the original data set, 2) estimate the parameters of TAN from the training set, 3) create a test set containing either 1000 instances randomly sampled from all instances not included in the training set or, if the data set is not large enough, all the instances not included in the training set, 4) classify the instances of the test set. We perform parameter estimation using the traditional BDeu prior, $\alpha_x = \frac{\texttt{iss}}{|\mathcal{Y}||\mathcal{X}|}$, with two values of equivalent sample size `iss=1`, the most common choice, and `iss=10`, the default value proposed by `bnlearn`. For the hierarchical model we set $s = |\mathcal{X}|$ and $\boldsymbol{\alpha}_0 = \mathbf{1}_{1\times|\mathcal{X}|}$, without any further tuning of the equivalent sample size.

We measure the classification performance through classification accuracy, which corresponds to the percentage of correctly classified instances, and the area under the ROC curve (ROC AUC). In a binary classification problem the ROC AUC represents the probability that a randomly chosen positive instance and a randomly chosen negative instance are correctly ranked based on the posterior probability of being positive (Bradley, 1997). In multi-class problems, the ROC AUC is averaged over the different classes, by considering in turn each class as the positive one.

As shown in Figure 7 and 8, the hierarchical model improves both accuracy and ROC AUC of the TAN classifier on the large majority of the 57 data sets. The improvements are huge when the training sets are small, i.e, $n = 20$, $n =$

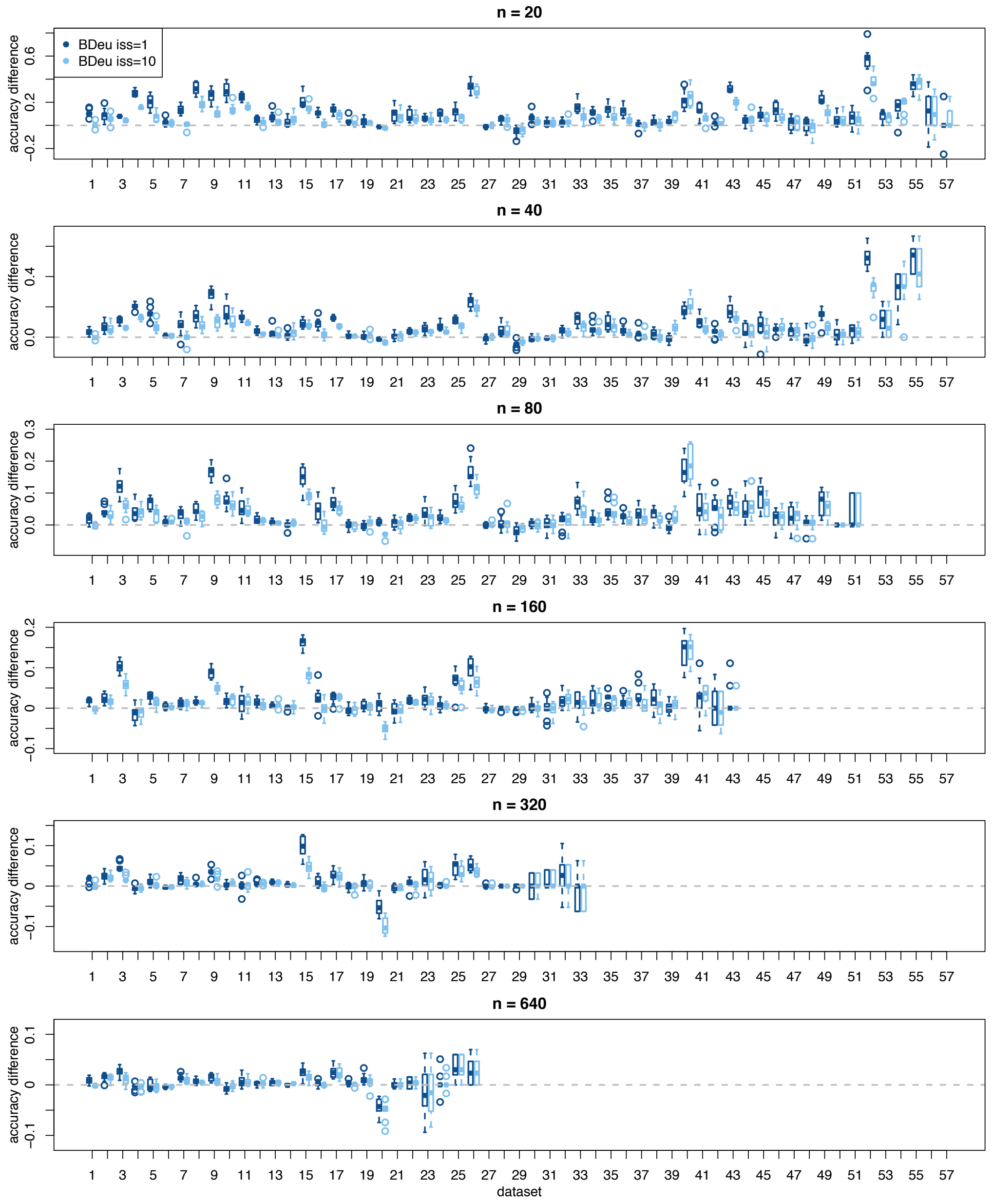


Figure 7: Improvement of accuracy in classification, obtained by estimating the TAN parameters through the hierarchical model rather than the BDeu prior with $s = 1$ (blue) and $s = 10$ (light blue). The experiments refer to 57 data sets and to different sizes of the training set ($n = 20, 40, 80, 160, 320, 640$). The data sets are ordered according to the number of instances, as in Table B.1. Positive values imply that the hierarchical model yields a higher ROC AUC. The y-axes have a different scale in different panels in order to highlight the gain attained by the hierarchical model even for large values of $n$.

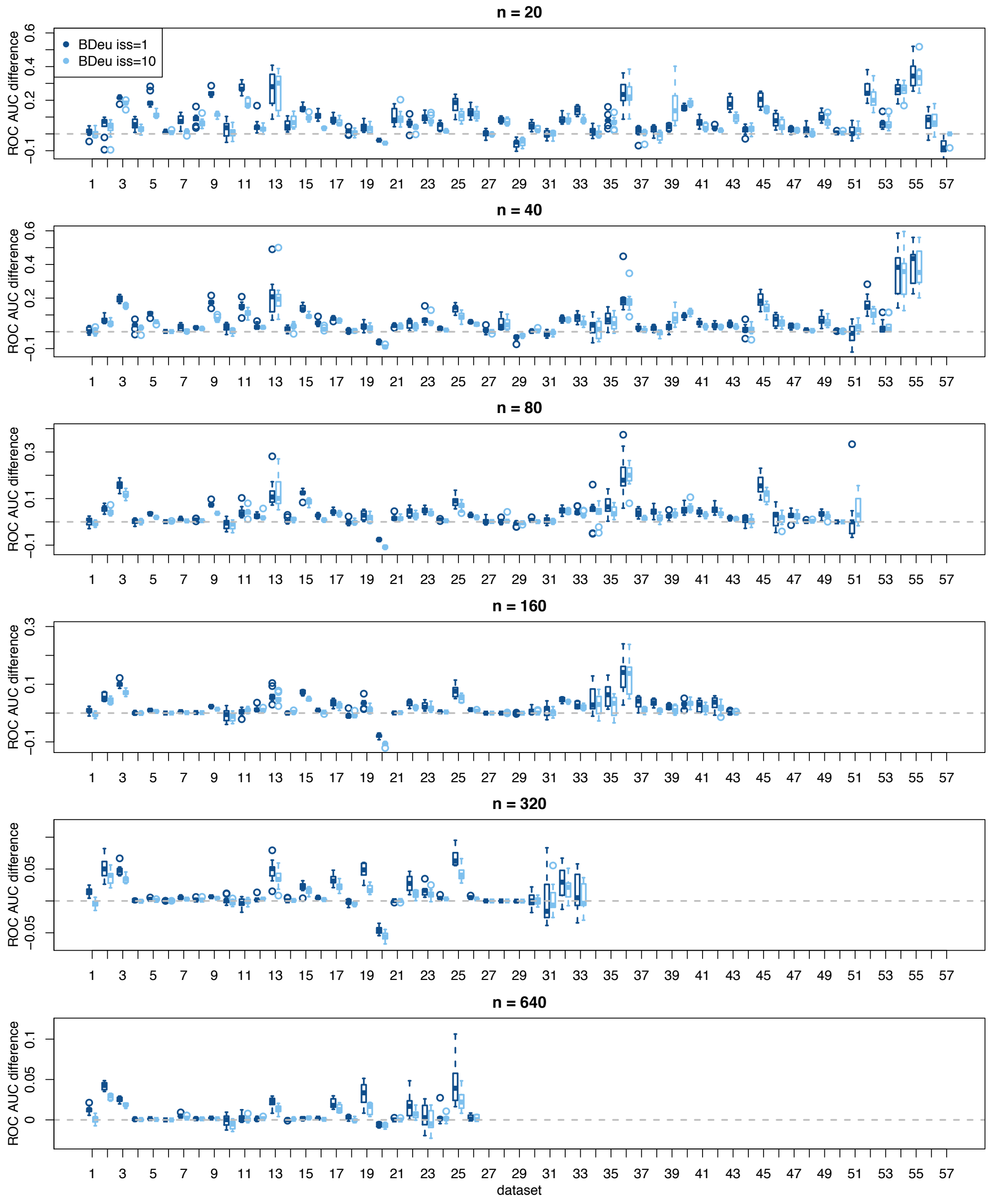


Figure 8: Improvement of the area under the ROC in classification, obtained by estimating the TAN parameters through the hierarchical model rather than the BDeu prior with $s = 1$ (blue) and $s = 10$ (light blue). The experiments refer to 57 data sets and to different sizes of the training set ($n = 20, 40, 80, 160, 320, 640$). The data sets are ordered according to the number of instances, as in Table B.1. Positive values correspond to a higher ROC AUC yielded by the hierarchical model. The y-axes have a different scale in different panels in order to highlight the gain attained by the hierarchical model even for large values of $n$.

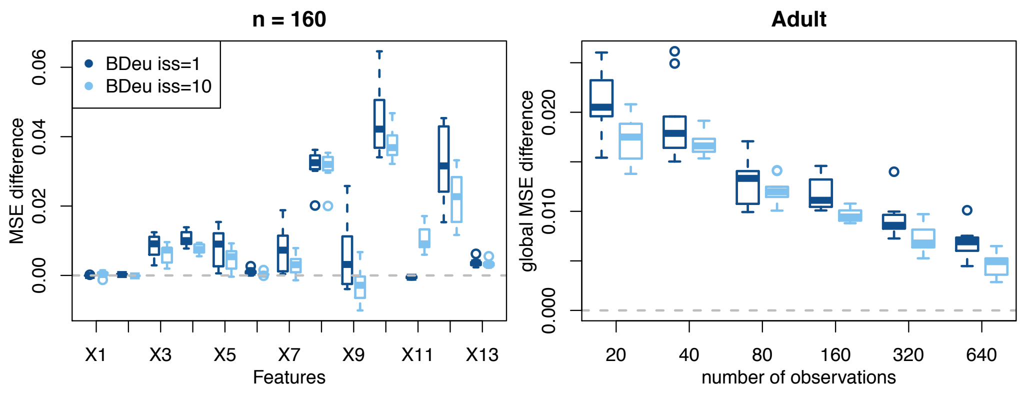


Figure 9: Difference in MSE between the BDeu and the hierarchical estimator on the *Adult* data set. Positive values favour the hierarchical model. Left panel: MSE difference of each CPT, for a selected sample size. Right panel: MSE difference summing over all the CPTs of the TAN structure, as a function of the sample size.

40, (the maximum achievable value of both indicators is 1). However, consistent improvements, albeit smaller, are observed also on the largest sample sizes.

These considerations remain valid regardless whether traditional estimation is carried out using `iss=1` or `iss=10`, although in general the latter setting performs better.

### *4.1.1. Further insights*

We further analyse the parameter estimates by considering the *Adult* data set, which contains 45,000 instances. We estimate the TAN structure and its parameters on the whole data set. Given the large sample size, we then consider such estimates to be the ground truth against which we compare the estimates obtained with subsampled versions of the data set. We then measure the MSE of the hierarchical model and the one of the traditional approach, adopting the same simulation procedures of the previous section. The MSE for each CPT is computed by means of (15).

In Figure 9 (right panel) for each CPT we represent the MSE difference between the traditional model and the hierarchical one, for a given sample size ($n$=160). Positive values favour the hierarchical model. The larger advantages are obtained for the features whose parents have a large number of joint states, namely: $X_3$, $X_4$, $X_5$, $X_8$, $X_{10}$, $X_{12}$. Indeed, the number of the joint states of their parents is respectively 28, 28, 10, 82, 14 and 28.

In Figure 9 (left panel) we instead represent the difference between traditional and hierarchical model in terms of global MSE, which is defined as the average MSE over different CPTs in the TAN. The difference in global MSE is represented as a function of the number of observations. The hierarchical model achieves lower MSE for each sample size then the traditional parameter estimation (which we carry out both with equivalent sample size `iss=1` and `iss=10`). As expected, the difference between the hierarchical estimators and the traditional estimator decreases with $n$; yet even for $n = 640$, the difference in MSE is positive in favor of the hierarchical model in all the simulations.

### *4.1.2. Computational considerations*

Using MCMC sampling, the time required for estimating the parameters of a single CPT on the *Adult* data set ranges between 6 seconds and 1 hour. Using VI, such a time is comprised between 0.01 and 18 seconds when our VI algorithm is used. Learning the parameters of the whole TAN model on the *Adult* data set requires on average 20 seconds using the proposed VI algorithm; the computational time required on average by MCMC is instead about one hour and ten minutes. Another practical advantage of our variational inference implementation is that it always converged successfully in all the experiments, while we experienced a significant number of crashes (around 10% of the CPTs to be estimated in the *Adult* data set) with Stan automatic variational inference, especially when dealing with the largest examples. We have also considered the HDP method presented in Petitjean et al. (2018) for a comparison on the *Adult* data set with 2500 training samples. Compared to HDP, the proposed method provides faster inference (one order of magnitude faster, i.e., 4 seconds against 86 seconds reported in Petitjean et al. (2018)), with similar accuracy (10% higher accuracy with respect to random forests, against which Petitjean et al. (2018) compare the performance of HDP). However, this comparison is only illustrative since the computing times depend on the machine on which the experiments are run.

## 5. Learning parameters from related data sets

A typical application of hierarchical models is the estimation of parameters from related data sets (Gelman et al., 2014, Chap. 5). Nowadays, related data sets arise commonly for instance in the contexts of multi-user platforms, patient specific treatments or multi-center clinical studies. In all these settings, some traits of the user/patient/center behaviour should be modeled and typically only few data for each user/patient/center are available. For convenience, in the following we refer to each related data set as *domain*. However, the method can be applied equally well to multi-user, multi-patient, multi-center problems.

At the moment, there is not an established method for learning the parameters of Bayesian networks from related domains. There are instead some methods for learning the *structure* of related Bayesian networks (Niculescu-Mizil and Caruana, 2007; Oates et al., 2016), which encourage the inferred graphs to be similar. However, both works recognise the problem of estimating the parameters from related domains as an open challenge. Oates et al. (2016) states: "At present this [referring to *information sharing for parameters*] appears to be challenging to include within our framework and represent an area of future research". Attempts of learning parameters in BNs from related domains include the work of Demichelis et al. (2006) for the Gaussian case and Malovini et al. (2012) for the discrete case, which however estimates the parameters of the hyper-prior in a point-wise fashion via maximum likelihood. The reported results deal only with the simple naive Bayes structure.

The hierarchical model proposed here can be easily customised to learn parameters of Bayesian networks from related domains. To the best of our knowledge, this is the first principled and scalable approach suitable to this end. For simplicity, we assume that there is a shared structure $\mathcal{G}$ common to all different domains. We then introduce the auxiliary discrete variable $F$ with values in $\mathcal{F}$, which represents the domain. We assume in the following that the domain variable is always observed. We will hence estimate the parameters of the $|\mathcal{F}|$ related Bayesian networks, characterised by the same structure $\mathcal{G}$. We apply the hierarchical model independently to each node $X_i$ of $\mathcal{G}$. For each $X_i$, we introduce the variable $X_i^{'} = X_i \times \mathrm{Pa}_i$, whose states are constituted by the joint states of $X_i$ and its parent set $\mathrm{Pa}_i$. We apply the hierarchical model by considering $X = X_i^{'}$ and the parent variable $Y = F$. Using the hierarchical model we simultaneously infer the $|\mathcal{F}|$ joint distributions $\boldsymbol{\theta}_{X_i,\mathrm{Pa}_i|f}$, one for each domain $f \in \mathcal{F}$. In this way, the joint distributions of $X_i$ and $\mathrm{Pa}_i$ for different domains are estimated by borrowing strength between the domains.

The parameter vector $\boldsymbol{\theta}_{X|y}$, for $y \in \mathcal{Y}$, corresponds now to the joint distribution associated to a variable $X_i$ and its parents $\mathrm{Pa}_i$, for a specific domain $f$, i.e., $\boldsymbol{\theta}_{X|y} = \boldsymbol{\theta}_{X_i,\mathrm{Pa}_i|f}$, for $i = 1, \ldots, I$, $\mathrm{pa} \in \mathcal{P}_i$ and $f \in \mathcal{F}$. In order to obtain the associated CPT, the joint distribution $\boldsymbol{\theta}_{X_i,\mathrm{Pa}_i|f}$ for a given $f \in \mathcal{F}$ is normalised with respect to the marginal distribution $\boldsymbol{\theta}_{\mathrm{Pa}_i|f}$, obtained summing by column $\boldsymbol{\theta}_{X_i,\mathrm{Pa}_i|f}$. Figure 10 shows a directed graphical model representing the hierarchical model applied to parameter learning from related domains.

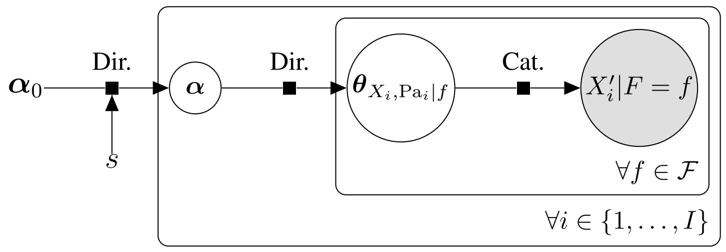

$\boldsymbol{\alpha}_0$
Dir.
$s$
$\boldsymbol{\alpha}$
Dir.
$\boldsymbol{\theta}_{X_i,\mathrm{Pa}_i|f}$
Cat.
$X_i'|F=f$
$\forall f \in \mathcal{F}$
$\forall i \in \{1,\ldots,I\}$

Figure 10: Hierarchical model applied to parameter learning in Bayesian networks from related domains represented by means of a directed factor graph. $X_i^{'} = X_i \times \mathrm{Pa}_i$ represents a new variable whose states are constituted by the joint states of $X_i$ and its parent set $\mathrm{Pa}_i$. Cat. and Dir. represent respectively Categorical and Dirichlet distributions.

### *5.1. Application to multi-domain classification*

We analyse the multi-user data set collected within by the *GoEco!*[3] (Bucher et al., 2016; Cellina et al., 2016) project, which tracks the travels of the users in order to suggest alternative mobility options. During the first phase of the project, each user logs via a mobile phone application the means of transport adopted for each displacement. This phase lasts about one week; such displacements, whose actual means of transport is known, constitute the training set of each user. After this phase, the application has to autonomously identify the means of transport adopted by the user in their next displacements.

There are thirteen means of transport to be recognised: {foot, bike, e-bike, kick scooter, e-car, car, motorbike, scooter, bus, train, tram, plane, ship}. The classification is based on sixteen features, which include average speed, total traveled distance, maximum distance between track-points, average heading change between track-points, differences between actual travel information (duration, starting and ending points); see the work of Bucher et al. (2016) for more details. As a first step, we discretise the numerical features into five equal-frequency bins.

As a first approach we train a TAN classifier independently for each user; we learn its parameters using the BDeu prior, with equivalent sample sizes `iss=1` and `iss=10`. As a further term of comparison, we learn an independent Random Forest (RF) classifier for each user. The random forest algorithm has been recognised as one of the best performing classifiers in extensive classification studies (Fernández Delgado et al., 2014), and the `R` implementation provided by

[3]`http://goeco-project.ch/index.php/en/`

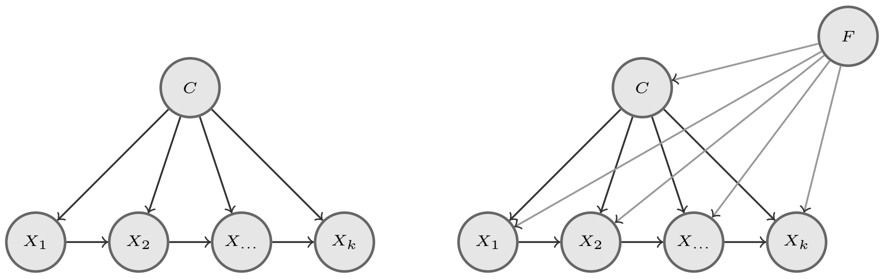


Figure 11: A TAN structure (left panel) and the artificial TAN structure associated to the problem of learning parameters from related domains (right panel). The class variable and the features have the domain variable $F$ as an extra parent.

`randomForest` package is empirically more accurate than other implementations (Bagnall and Cawley, 2017). We thus train a different RF for each user, using the default settings of the `randomForest` package.

We then implement the proposed hierarchical approach to learn a classifier from related data sets as follows. We first learn a TAN structure by pooling the training data of different users; this yields an estimated structure $\mathcal{G}$, shared across domains. We then estimate the parameters of $\mathcal{G}$ one node at a time; for each node we jointly estimate the parameters of different users by means of the hierarchical model. Hence each user has his own parameters, but the parameters of different users are shrunk towards each other. The real TAN structure and the artificial TAN structure associated to the problem of parameter learning from related domains are represented respectively in left and right panels of Figure 11.

We consider 72570 travels of 351 users collected during one month. We select from the data set 100 users with more than 200 labeled records. We adopt this threshold since we need a reasonably sized test set in order to obtain reliable empirical results; as already shown, the advantage of the hierarchical model would further increase when considering smaller training sets. For each of the $K$ users and for each $n \in \{100, 250, 500\}$, we repeat 10 times the procedure of 1) sampling $n$ observations as training set and 100 observations from the remaining data as test set, 2) learning the classifiers and 3) classifying the instances of the test set. The analyses differ not only in terms of number of observations per user, i.e., $n \in \{100, 250, 500\}$, but also in terms of number of users involved in the analysis, i.e., $K \in \{100, 69, 6\}$. Only 6 users recorded more than 600 observations.

We then classify each instance of the test set. For each of the $K$ users, the test set contains 100 instances sampled uniformly from all the instances not included

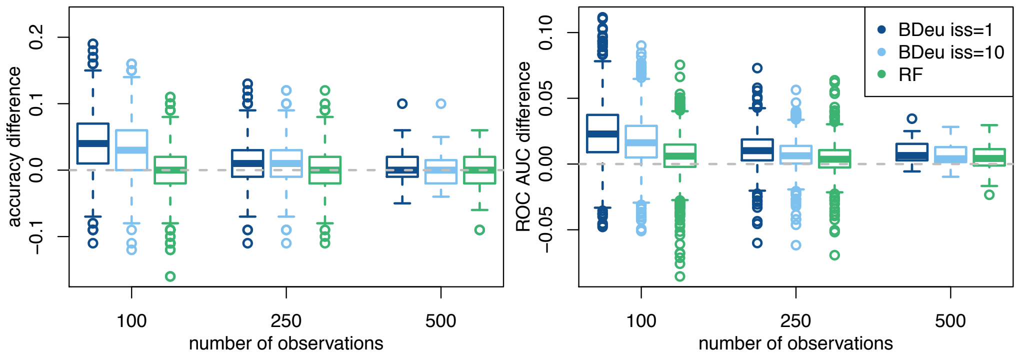


Figure 12: Boxplots of accuracy gain (left panel) and area under the ROC gain (right panel) obtained comparing the hierarchical method with respect to user-specific traditional methods (BDeu with $s = 1$ in blue, BDeu with $s = 10$ in light blue and Random Forest in green) for three scenarios ($n = 100, 250, 500$) in the multi-user problem. Positive values favour the hierarchical model. The boxplots for $n = 100, 250, 500$ collect respectively a number of users $K = 100, 69, 6$.

in the training set. In Figure 12 we report the difference in accuracy (left panel) and ROC AUC (right panel) between the multi-user TAN and the user-specific classifiers. The boxplots collect the results obtained for 100 instances of the test set, $K$ users and 10 repetitions of the experiment. Notice that the hierarchical BN produces a large gain in both accuracy and ROC AUC with respect to both BDeu approaches. Moreover, it produces comparable results with respect to RF. In Figure 13 and 14 we instead report respectively the difference in accuracy and ROC AUC between the hierarchical BN and the traditional classifiers for each of the $K = 100$ users, when $n = 100$. Notice that the hierarchical approach provides a gain in accuracy and ROC AUC with respect to both BDeu for approximately the 95% of users. The comparison with respect to RF is instead more balanced. There is not a clear difference between the two methods.

According to Figures 12, 13 and 14, the hierarchical approach yields consistently better results in multi-user classification with respect to both the BDeu classifiers. Qualitatively similar results are obtained if we consider as competitors the TAN models whose structure is learned on the pooled data set and whose parameters are learned independently for each user using the BDeu prior. Moreover, the hierarchical BN makes BN a competitive method with respect to RF in multi-user problems.

Variational inference makes the introduction of hierarchical models in Bayesian networks possible. Indeed, the computational time of VI for parameter estimation

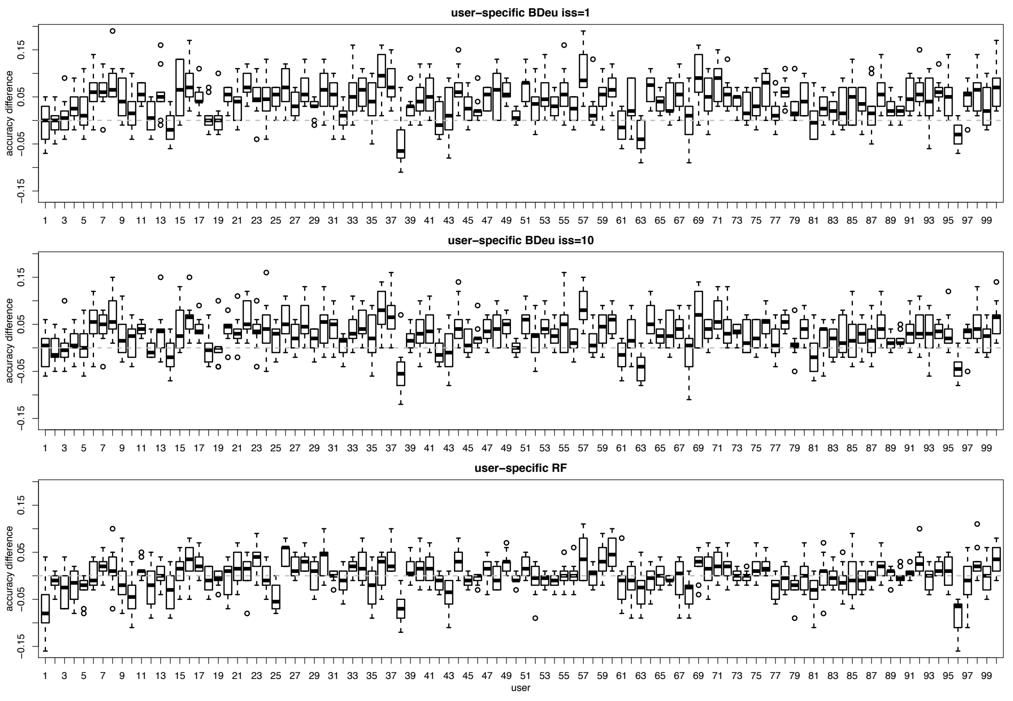


Figure 13: Improvements of accuracy yielded by the multi-domain approach over the user-specific classifiers: TAN with BDeu prior and $s = 1$ (top panel), TAN with BDeu prior and $s = 10$ (central panel) and Random Forest (bottom panel). Positive values imply a higher accuracy of the hierarchical model. The results refer to 100 users with a user training set of 100 observations each.

in a single CPT with 100 users ranges between 2 seconds and 7 minutes. The same estimation performed by means of MCMC takes more than 30 hours for a single CPT estimation. If the number of users, i.e., the number of conditioning states, diminishes, the computing time diminishes as well. With 6 users, the computational time of VI ranges between 0.6 and 12 seconds, while the computing time of MCMC ranges between 23 minutes and 20.5 hours. In both settings the advantage of introducing VI instead of MCMC in parameter estimation procedures is clear. The proposed VI method always converged successfully in all the experiments, while Stan automatic variational inference experienced crashes in about 20% of the cases, when borrowing strength among 100 users.

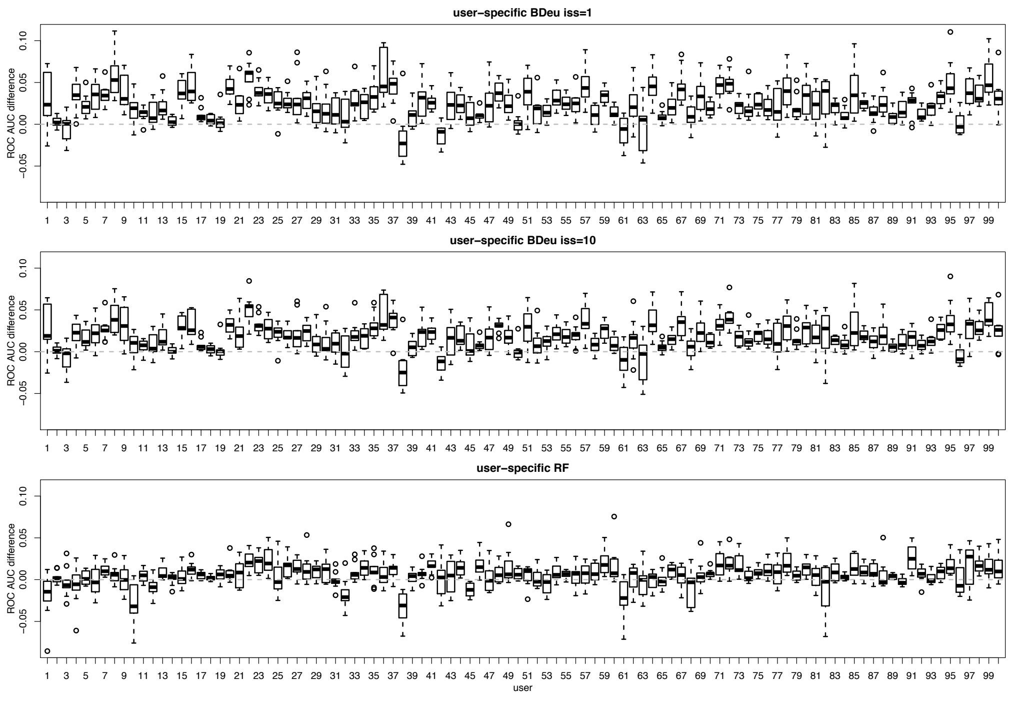


Figure 14: Improvements of area under the ROC yielded by the multi-domain approach over the user-specific classifiers: TAN with BDeu prior and $s = 1$ (top panel), TAN with BDeu prior and $s = 10$ (central panel) and Random Forest (bottom panel). Positive values imply a higher ROC AUC of the hierarchical model. The results refer to 100 users with a user training set of 100 observations each.

## 6. Conclusions and future work

In this work, we have introduced an innovative method for parameter estimation in Bayesian networks by relaxing the local independence assumption. The proposed variational method provides faster inference with respect to MCMC methods and more accurate inference with respect to Stan automatic variational inference. Moreover it improves classification performance of a BN classifier with respect to traditional Bayesian estimation under parameter independence. The proposed hierarchical method can be also applied to the problem of learning Bayesian networks parameters from related domains. The hierarchical model allows to improve parameter estimation by *borrowing statistical strength* from related domains, while the variational approximation makes inference feasible even when the number of related domain is large.

Although focused on BN applications, the proposed approach can be applied whenever a set of related conditional distributions should be estimated.

One interesting development consists in the application of the proposed method to other classifiers. Indeed, further advances in classification can be obtained by considering ensembles of TAN classifiers, such as AODE (Webb et al., 2005). Another interesting development consists in the adoption of a hierarchical approach also in *structural learning*. Coupling a hierarchical approach in structural learning with the proposed hierarchical parameter learning would further increase the performance of the Bayesian networks classifiers. A further research direction consists in modelling also the hyper parameters $s$ and $\boldsymbol{\alpha}_0$ as random variables, by adopting full Bayesian or empirical Bayesian approaches.

*Acknowledgements*

We are grateful to Francesca Cellina and Francesca Mangili for sharing with us the application presented in Section 5.1 and the data collected within the *GoEco!* project and to the anonymous reviewers for the insightful comments. The research in this paper has been partially supported by the Swiss NSF grants ns. IZKSZ2 162188, NRP 71 "Managing energy consumption" and by the Commission for Technology and Innovation (CTI) within the Swiss Competence Center for Energy Research (SCCER) Mobility.

## Appendix

## Appendix A. Posterior distributions under the hierarchical model

### *Appendix A.1. Prior correlation*

PROOF (LEMMA 1). The correlation between $\theta_{x|y}$ and $\theta_{x|y'}$ is

$$\mathbb{C}\text{or}\left(\theta_{x|y}, \theta_{x|y'}\right) = \frac{\mathbb{C}\text{ov}\left(\theta_{x|y}, \theta_{x|y'}\right)}{\sqrt{\mathbb{V}\text{ar}\left(\theta_{x|y}\right)}\sqrt{\mathbb{V}\text{ar}\left(\theta_{x|y'}\right)}}.$$

In order to derive $\mathbb{C}\text{or}\left(\theta_{x|y}, \theta_{x|y'}\right)$ we thus need to derive $\mathbb{V}\text{ar}\left(\theta_{x|y}\right)$ and $\mathbb{C}\text{ov}\left(\theta_{x|y}, \theta_{x|y'}\right)$.

The prior average of $\theta_{x|y}$ is obtained by means of the law of total expectation as

$$\mathbb{E}\left[\theta_{x|y}\right] = \mathbb{E}\left[\mathbb{E}^{\boldsymbol{\alpha}}\left[\theta_{x|y}\right]\right] = \frac{1}{s}\mathbb{E}\left[\alpha_x\right],$$

where $\mathbb{E}^{\boldsymbol{\alpha}}\left[\cdot\right] = \mathbb{E}\left[\cdot\,|\boldsymbol{\alpha}\right]$.

Analogously, the posterior covariance between $\theta_{x|y}$ and $\theta_{x'|y'}$ is obtained by means of the law of total covariance, i.e.,

$$\mathbb{C}\text{ov}\big(\theta_{x|y}, \theta_{x'|y'}\big) = \mathbb{C}\text{ov}\left(\mathbb{E}^{\boldsymbol{\alpha}}\left[\theta_{x|y}\right], \mathbb{E}^{\boldsymbol{\alpha}}\left[\theta_{x'|y'}\right]\right) + \mathbb{E}\left[\mathbb{C}\text{ov}^{\boldsymbol{\alpha}}\left(\theta_{x|y}, \theta_{x'|y'}\right)\right].$$

The first quantity is:

$$\mathbb{C}\text{ov}\left(\mathbb{E}^{\boldsymbol{\alpha}}\left[\theta_{x|y}\right], \mathbb{E}^{\boldsymbol{\alpha}}\left[\theta_{x'|y'}\right]\right) = \frac{1}{s^2}\mathbb{C}\text{ov}\left(\alpha_x, \alpha_{x'}\right).$$

If $y' = y$, the second quantity is:

$$\begin{aligned}\mathbb{E}\left[\mathbb{C}\text{ov}^{\boldsymbol{\alpha}}\left(\theta_{x|y}, \theta_{x'|y}\right)\right] &= \mathbb{E}\left[\frac{\alpha_x(s\delta_{xx'} - \alpha_{x'})}{s^2(s+1)}\right] \\ &= \frac{\mathbb{E}\left[\alpha_x\right]\left(s\delta_{xx'} - \mathbb{E}\left[\alpha_{x'}\right]\right)}{s^2(s+1)} - \frac{\left(\mathbb{E}\left[\alpha_x\alpha_{x'}\right] - \mathbb{E}\left[\alpha_x\right]\mathbb{E}\left[\alpha_{x'}\right]\right)}{s^2(s+1)} \\ &= \frac{(s_0+1)\mathbb{C}\text{ov}\left(\alpha_x, \alpha_{x'}\right)}{s^2(s+1)} - \frac{\mathbb{C}\text{ov}\left(\alpha_x, \alpha_{x'}\right)}{s^2(s+1)} \\ &= \frac{s_0}{s^2(s+1)}\mathbb{C}\text{ov}\left(\alpha_x, \alpha_{x'}\right),\end{aligned}$$

since $\boldsymbol{\alpha}$ is distributed as a rescaled Dirichlet with parameter vector $\boldsymbol{\alpha}_0$.
Otherwise, if $y' \neq y$, $\mathbb{E}\left[\mathbb{C}\text{ov}^{\boldsymbol{\alpha}}\left(\theta_{x|y}, \theta_{x'|y'}\right)\right] = 0$, since $\boldsymbol{\theta}_{X|y} \perp\!\!\!\perp \boldsymbol{\theta}_{X|y'}$ given $\boldsymbol{\alpha}$.

Exploiting the law of total covariance, we obtain

$$\mathbb{C}\text{ov}\big(\theta_{x|y}, \theta_{x'|y'}\big) = \frac{\mathbb{C}\text{ov}\left(\alpha_x, \alpha_{x'}\right)}{s^2} + \frac{\delta_{yy'}s_0\mathbb{C}\text{ov}\left(\alpha_x, \alpha_{x'}\right)}{s^2(s+1)}.$$

The covariance between $\theta_{x|y}$ and $\theta_{x|y'}$ and the variance of $\theta_{x|y}$ are obtained as special cases of $\mathbb{C}\text{ov}\big(\theta_{x|y}, \theta_{x'|y'}\big)$ as

$$\mathbb{C}\text{ov}\left(\theta_{x|y}, \theta_{x|y'}\right) = \frac{1}{s^2}\mathbb{V}\text{ar}(\alpha_x),$$

and

$$\mathbb{V}\text{ar}\left(\theta_{x|y}\right) = \mathbb{V}\text{ar}\left(\theta_{x|y'}\right) = \frac{s+s_0+1}{s^2(s+1)}\mathbb{V}\text{ar}(\alpha_x).$$

Thus,

$$\mathbb{C}\text{or}\left(\theta_{x|y}, \theta_{x|y'}\right) = \frac{\frac{1}{s^2}\mathbb{V}\text{ar}(\alpha_x)}{\frac{s+s_0+1}{s^2(s+1)}\mathbb{V}\text{ar}(\alpha_x)} = \frac{s+1}{s+s_0+1}.$$

*Appendix A.2. Posterior distribution of $\theta_{X|Y}$*

PROOF (LEMMA 2). Assuming $\boldsymbol{\alpha}$ to be known, the marginal posterior density for $\boldsymbol{\theta}_{X|y}$ is a Dirichlet distribution with parameters $\boldsymbol{\alpha}+\boldsymbol{n}_y$, where $\boldsymbol{n}_y = (n_{x_1 y}, n_{x_2 y}, \ldots)'$. It is thus easy to compute

$$\mathbb{E}\left[\theta_{x|y}\middle|\, \boldsymbol{\alpha}, D\right] = \mathbb{E}^{\boldsymbol{\alpha},D}\left[\theta_{x|y}\right] = \frac{n_{xy}+\alpha_x}{n_y+s},$$

where $\mathbb{E}^{\boldsymbol{\alpha},D}[\cdot] = \mathbb{E}[\cdot\,|\boldsymbol{\alpha}, D]$.

The posterior expectation of $\theta_{x|y}$ can thus be computed by means of the law of total expectation as

$$\mathbb{E}^{D}\left[\theta_{x|y}\right] = \mathbb{E}^{D}\left[\mathbb{E}^{\boldsymbol{\alpha},D}\left[\theta_{x|y}\right]\right] = \frac{n_{xy}+\mathbb{E}^{D}\left[\alpha_x\right]}{n_y+s}.$$

In order to compute the posterior covariance between $\theta_{x|y}$ and $\theta_{x'|y'}$ we can use the law of total covariance, i.e.,

$$\mathbb{C}\text{ov}^{D}\left(\theta_{x|y}, \theta_{x'|y'}\right) = \mathbb{C}\text{ov}^{D}\left(\mathbb{E}^{\boldsymbol{\alpha},D}\left[\theta_{x|y}\right], \mathbb{E}^{\boldsymbol{\alpha},D}\left[\theta_{x'|y'}\right]\right) + \mathbb{E}^{D}\left[\mathbb{C}\text{ov}^{\boldsymbol{\alpha},D}\left(\theta_{x|y}, \theta_{x'|y'}\right)\right].$$

The first quantity is:

$$\begin{aligned}\mathbb{C}\text{ov}^{D}\left(\mathbb{E}^{\boldsymbol{\alpha},D}\left[\theta_{x|y}\right], \mathbb{E}^{\boldsymbol{\alpha},D}\left[\theta_{x'|y'}\right]\right) &= \mathbb{C}\text{ov}^{D}\left(\frac{n_{xy}+\alpha_x}{n_y+s}, \frac{n_{x'y'}+\alpha_{x'}}{n_{y'}+s}\right)\\ &= \frac{\mathbb{C}\text{ov}^{D}\left(\alpha_x, \alpha_{x'}\right)}{\left(n_y+s\right)\left(n_{y'}+s\right)}.\end{aligned}$$

If $y' = y$, the second quantity is:

$$\begin{aligned}\mathbb{E}^{D}\left[\mathbb{C}\text{ov}^{\boldsymbol{\alpha},D}\left(\theta_{x|y}, \theta_{x'|y}\right)\right] &= \mathbb{E}^{D}\left[\frac{(n_{xy}+\alpha_x)((n_y+s)\delta_{xx'} - (n_{x'y}+\alpha_{x'}))}{(n_y+s)^2(n_y+s+1)}\right]\\ &= \frac{(n_{xy}+\mathbb{E}^{D}\left[\alpha_x\right])\delta_{xx'}}{(n_y+s)(n_y+s+1)} - \frac{\mathbb{E}^{D}\left[(n_{xy}+\alpha_x)(n_{x'y}+\alpha_{x'})\right]}{(n_y+s)^2(n_y+s+1)}\\ &= \frac{\mathbb{E}^{D}\left[\theta_{x|y}\right]\left(\delta_{xx'} - \mathbb{E}^{D}\left[\theta_{x'|y'}\right]\right)}{n_y+s+1} - \frac{\mathbb{C}\text{ov}^{D}\left(\alpha_x, \alpha_{x'}\right)}{(n_y+s)^2(n_y+s+1)}.\end{aligned}$$

Otherwise, if $y' \neq y$, $\mathbb{E}^{D}\left[\mathbb{C}\text{ov}^{\boldsymbol{\alpha},D}\left(\theta_{x|y}, \theta_{x'|y'}\right)\right] = 0$, since $\boldsymbol{\theta}_{X|y} \perp\!\!\!\perp \boldsymbol{\theta}_{X|y'}$ given $\boldsymbol{\alpha}$.

Exploiting the law of total covariance, we obtain

$$\begin{aligned}\mathbb{C}\text{ov}^{D}\left(\theta_{x|y}, \theta_{x'|y'}\right) =& \frac{\mathbb{C}\text{ov}^{D}\left(\alpha_x, \alpha_{x'}\right)}{\left(n_y+s\right)\left(n_{y'}+s\right)} + \delta_{yy'}\frac{\mathbb{E}^{D}\left[\theta_{x|y}\right]\delta_{xx'} - \mathbb{E}^{D}\left[\theta_{x|y}\right]\mathbb{E}^{D}\left[\theta_{x'|y}\right]}{n_y+s+1} +\\ &- \delta_{yy'}\frac{\mathbb{C}\text{ov}^{D}\left(\alpha_x, \alpha_{x'}\right)}{(n_y+s)^2(n_y+s+1)}.\end{aligned}$$

*Appendix A.3. Posterior moments of $\boldsymbol{\alpha}$*

The following lemma states a general result for computing any posterior moment of $\boldsymbol{\alpha}$, whose general expression is $\mathbb{E}^D\left[\prod_{x\in\mathcal{X}} \alpha_x^{k_x}\right]$, where $k_x \in \mathbb{N}$ represents the power of element $\alpha_x$. Thanks to this lemma it is possible to easily prove Lemma 3.

**Lemma 4.** *Under the assumptions of model* (5)*, the posterior average of the quantity* $\prod_{x\in\mathcal{X}} \alpha_x^{k_x}$*, with* $k_x \in \mathbb{N}$, $\forall x \in \mathcal{X}$*, is*

$$\mathbb{E}^D\left[\prod_{x\in\mathcal{X}} \alpha_x^{k_x}\right] = \gamma \int \prod_{\substack{x\in\mathcal{X}\\ y\in\mathcal{Y}\\ s.t.\ n_{xy}>0}} \prod_{m=1}^{n_{xy}} (\alpha_x + m - 1)\left(\frac{\alpha_x}{s}\right)^{\tilde{k}_x} \frac{d\boldsymbol{\alpha}}{s}, \tag{A.1}$$

*where* $\tilde{k}_x = [\boldsymbol{\alpha}_0]_x + k_x - 1$ *and* $\gamma$ *is a proportionality constant such that*

$$\gamma^{-1} = \int \prod_{\substack{x\in\mathcal{X}\\ y\in\mathcal{Y}\\ s.t.\ n_{xy}>0}} \prod_{m=1}^{n_{xy}} (\alpha_x + m - 1)\left(\frac{\alpha_x}{s}\right)^{[\boldsymbol{\alpha}_0]_x - 1} \frac{d\boldsymbol{\alpha}}{s}. \tag{A.2}$$

PROOF. Under the assumptions of model (5), the joint posterior density of $\boldsymbol{\alpha}, \boldsymbol{\theta}_{X|Y}$ is

$$p(\boldsymbol{\alpha}, \boldsymbol{\theta}_{X|Y}|D) \propto \frac{\Gamma(s)}{\prod_{x\in\mathcal{X}}\Gamma(\alpha_x)} \prod_{y\in\mathcal{Y}}\prod_{x\in\mathcal{X}} \theta_{x|y}^{n_{xy}+\alpha_x-1}\left(\frac{\alpha_x}{s}\right)^{[\boldsymbol{\alpha}_0]_x - 1}\frac{1}{s}.$$

Marginalising $p(\boldsymbol{\alpha}, \boldsymbol{\theta}_{X|Y}|D)$ with respect to $\boldsymbol{\theta}_{X|Y}$, we obtain

$$p(\boldsymbol{\alpha}|D) \propto \frac{\Gamma(s)}{\prod_{x\in\mathcal{X}}\Gamma(\alpha_x)} \prod_{y\in\mathcal{Y}} \frac{\prod_{x\in\mathcal{X}}\Gamma(\alpha_x + n_{xy})}{\Gamma(s+n_y)} \prod_{x\in\mathcal{X}}\left(\frac{\alpha_x}{s}\right)^{[\boldsymbol{\alpha}_0]_x - 1}\frac{1}{s}. \tag{A.3}$$

Thanks to the well-known property of the Gamma function

$$\Gamma(\alpha + M) = \prod_{m=1}^{M}(\alpha + m - 1)\cdot\Gamma(\alpha), \quad \text{for } M \geq 1$$

we can write the posterior marginal density (A.3) as

$$p(\boldsymbol{\alpha}|D) \propto \prod_{\substack{x\in\mathcal{X}\\ y\in\mathcal{Y}\\ \text{s.t. } n_{xy}>0}} \prod_{m=1}^{n_{xy}} (\alpha_x + m - 1)\left(\frac{\alpha_x}{s}\right)^{[\boldsymbol{\alpha}_0]_x - 1}\frac{1}{s}. \tag{A.4}$$

The proportionality constant of the posterior marginal density is obtained by integrating the right term in (A.4) with respect to the $|\mathcal{X}|$ elements of $\boldsymbol{\alpha}$, such that $\sum_{x\in\mathcal{X}} \alpha_x = s$. The resulting proportionality constant is thus

$$\gamma = \left( \int \prod_{\substack{x\in\mathcal{X} \\ y\in\mathcal{Y} \\ \text{s.t. } n_{xy}>0}} \prod_{m=1}^{n_{xy}} (s\alpha_x + m - 1) \left(\frac{\alpha_x}{s}\right)^{[\boldsymbol{\alpha}_0]_x - 1} \frac{d\boldsymbol{\alpha}}{s} \right)^{-1} .$$

The posterior average for the quantity $\prod_{x\in\mathcal{X}} \alpha_x^{k_x}$ can be derived directly from the posterior marginal density of $\boldsymbol{\alpha}$ as

$$\mathbb{E}^D \left[ \prod_{x\in\mathcal{X}} \alpha_x^{k_x} \right] = \gamma \int \prod_{\substack{x\in\mathcal{X} \\ y\in\mathcal{Y} \\ \text{s.t. } n_{xy}>0}} \prod_{m=1}^{n_{xy}} (\alpha_x + m - 1) \left(\frac{\alpha_x}{s}\right)^{\tilde{k}_x} d\boldsymbol{\alpha},$$

where $\tilde{k}_x = [\boldsymbol{\alpha}_0]_x + k_x - 1$. In the special case of $[\boldsymbol{\alpha}_0]_x = 1, \forall x \in \mathcal{X}$, we have $\tilde{k}_x = k_x$.

The proof of Lemma 3 derives directly from Lemma 4.

PROOF (LEMMA 3). Both the posterior expectation for $\alpha_{x'}$ and the posterior expectation for the product of $\alpha_{x'}$ and $\alpha_{x''}$ are obtained directly from (A.1), by choosing respectively $k_x = \delta_{xx'}$, i.e., $k_x = 1$ for $x = x'$ and $k_x = 0$ for $\forall x \neq x'$, and $k_x = \delta_{xx'} + \delta_{xx''}$, i.e., $k_x = 1$ for $x \in \{x', x''\}$ and $k_x = 0$ for $\forall x \in \{x', x''\}$.

The numerical computation of integrals in Lemma 4 consists in an iterative algorithm that goes through all the layers of the multiple integral, integrating a polynomial function in one variable at each time. In the following the first argument of a function represents the integration variable, while the second one represents a fixed parameter.

**Algorithm 2.** *Numerical computation of moments.*
*Fix starting value* $I(\alpha_{x_{|\mathcal{X}|}}; \beta_{x_{|\mathcal{X}|}}) = 1$*, where* $\beta_{x_{|\mathcal{X}|}} = s - \alpha_{x_{|\mathcal{X}|}}$.
*For* $i$ *in* $|\mathcal{X}| : 1$*:*

1. *compute the integrand function* $f(\alpha_{x_i}; \beta_{x_i})$ *as the product of the binomials in* (A.1) *or* (A.2) *and* $I(\alpha_{x_i}; \beta_{x_i})$*;*

2. *compute the primitive function $F(\alpha_{x_i};\beta_{x_i})$ associated to the integrand $f(\alpha_{x_i};\beta_{x_i})$;*
3. *evaluate the primitive in $\beta_{x_i}$ , i.e., $F(\beta_{x_i}) = F(\beta_{x_i};\beta_{x_i})$;*
4. *set $\beta_{x_{i-1}} = \beta_{x_i} - \alpha_{x_i}$;*
5. *compute the inner integral $I(\alpha_{x_{i-1}};\beta_{x_{i-1}})$ by evaluating $F$ in $\beta_{x_{i-1}} - \alpha_{x_{i-1}}$.*

*The four steps can be computed by means of symbolic calculus in Python.*

The computation starts from the innermost integral, which corresponds, e.g., to the last index of $\boldsymbol{\alpha}$. We always define the quantity $\beta_x = s - \sum_{x'<x} \alpha_{x'}$. At each step of the algorithm the integrand function $f(\alpha_x;\beta_x)$ is a polynomial function of $\alpha_x$, whose coefficients depend on the quantity $\beta_x$. The integrand function $f(\alpha_x;\beta_x)$ is computed by multiplying the integral computed in the previous step $I(\alpha_x;\beta_x)$ by all the binomials $(\alpha_x + m - 1)$ in (A.1) or (A.2). In the first step $I(\alpha_x;\beta_x) = 1$. The primitive $F(\alpha_x;\beta_x)$ associated to $f(\alpha_x;\beta_x)$ is computed by means of symbolic calculus with *sympy*. $F(\alpha_x;\beta_x)$ is still a polynomial function in $\alpha_x$ with degree increased by one with respect to the integrand. The integral is then computed by evaluating the primitive function in $\beta_x$ and in 0, i.e. $F(\beta_x;\beta_x) - F(0;\beta_x)$. $F(\beta_x) = F(\beta_x;\beta_x)$ is a polynomial function in $\beta_x$, while $F(0;\beta_x) = 0$, since there are no constant terms in the primitive function. The quantity $\beta_x$ is now written as $\beta_{x'} - \alpha_{x'}$, with $x' < x$, thus by means of symbolic calculus we set $I(\alpha_{x'};\beta_{x'}) = F(\beta_{x'} - \alpha_{x'})$. The polynomial function $I(\alpha_{x'};\beta_{x'})$ represents the inner integral as a function of the new integration variable $\alpha_{x'}$, with coefficients depending on $\beta_{x'}$. All these steps are repeated through all the layers of the multiple integral. When the outer integral has been reached, the final result is obtained as $I(s;s)$.

## Appendix B. Variational inference

### *Appendix B.1. ELBO derivation*

The general expression (13) for the ELBO and its relation with the KL divergence are well known in the machine learning literature. Blei et al. (2017) derived them in some general cases. Exploiting Jensen's inequality, the ELBO bound of the hierarchical MD model is:

$$\begin{aligned}\log(p(D)) &= \log\left(\int\int p(D,\boldsymbol{\theta}_{X|Y},\boldsymbol{\alpha})d\boldsymbol{\theta}_{X|Y}d\boldsymbol{\alpha}\right)\\ &= \log\left(\int\int q(\boldsymbol{\theta}_{X|Y},\boldsymbol{\alpha})\frac{p(D,\boldsymbol{\theta}_{X|Y},\boldsymbol{\alpha})}{q(\boldsymbol{\theta}_{X|Y},\boldsymbol{\alpha})}d\boldsymbol{\theta}_{X|Y}d\boldsymbol{\alpha}\right)\\ &\geq \int\int q(\boldsymbol{\theta}_{X|Y},\boldsymbol{\alpha})\log\left(\frac{p(D,\boldsymbol{\theta}_{X|Y},\boldsymbol{\alpha})}{q(\boldsymbol{\theta}_{X|Y},\boldsymbol{\alpha})}\right)d\boldsymbol{\theta}_{X|Y}d\boldsymbol{\alpha}\\ &= \int\int q(\boldsymbol{\theta}_{X|Y},\boldsymbol{\alpha})\log\left(p(D,\boldsymbol{\theta}_{X|Y},\boldsymbol{\alpha})\right)d\boldsymbol{\theta}_{X|Y}d\boldsymbol{\alpha}\\ &- \int\int q(\boldsymbol{\theta}_{X|Y},\boldsymbol{\alpha})\log\left(q(\boldsymbol{\theta}_{X|Y},\boldsymbol{\alpha})\right)d\boldsymbol{\theta}_{X|Y}d\boldsymbol{\alpha}\\ &= \mathbb{E}_q\left[\log\left(p(D,\boldsymbol{\theta}_{X|Y},\boldsymbol{\alpha})\right)\right] - \mathbb{E}_q\left[\log\left(q(\boldsymbol{\theta}_{X|Y},\boldsymbol{\alpha})\right)\right] = \mathcal{L}.\end{aligned}$$

The KL divergence between the variational and the real posterior distributions corresponds to the difference between $\log(p(D))$ and $\mathcal{L}$, i.e.,

$$\log(p(D)) - \mathcal{L} = \text{KL}(q(\boldsymbol{\theta}_{X|Y},\boldsymbol{\alpha}) \,||\, p(\boldsymbol{\theta}_{X|Y},\boldsymbol{\alpha}|D)).$$

Indeed,

$$\begin{aligned}\mathcal{L} &= \int\int q(\boldsymbol{\theta}_{X|Y},\boldsymbol{\alpha})\log\left(\frac{p(D,\boldsymbol{\theta}_{X|Y},\boldsymbol{\alpha})}{q(\boldsymbol{\theta}_{X|Y},\boldsymbol{\alpha})}\right)d\boldsymbol{\theta}_{X|Y}d\boldsymbol{\alpha}\\ &= \int\int q(\boldsymbol{\theta}_{X|Y},\boldsymbol{\alpha})\log\left(\frac{p(\boldsymbol{\theta}_{X|Y},\boldsymbol{\alpha}|D)p(D)}{q(\boldsymbol{\theta}_{X|Y},\boldsymbol{\alpha})}\right)d\boldsymbol{\theta}_{X|Y}d\boldsymbol{\alpha}\\ &= \text{KL}(q(\boldsymbol{\theta}_{X|Y},\boldsymbol{\alpha}) \,||\, p(\boldsymbol{\theta}_{X|Y},\boldsymbol{\alpha}|D)) + \log(p(D)).\end{aligned}$$

Since the KL divergence is a positive quantity, the maximisation of $\mathcal{L}$ corresponds to the minimisation of the KL divergence. On the other hand, the minimisation of the KL divergence corresponds to the approximation of $\log(p(D))$ by means of a tight lower bound.

In order to derive the ELBO $\tilde{\mathcal{L}}$ we first derive the bound (13) in the specific case of the hierarchical MD model (5) with approximating model (12), i.e.,

$$\begin{aligned}\mathcal{L} =& \sum_{y\in\mathcal{Y}}\mathbb{E}_q\left[\log\left(p(D|\boldsymbol{\theta}_{X|y})\right)\right] + \sum_{y\in\mathcal{Y}}\mathbb{E}_q\left[\log\left(p(\boldsymbol{\theta}_{X|y}|\boldsymbol{\alpha})\right)\right]\\ &+ \mathbb{E}_q\left[\log\left(p(\boldsymbol{\alpha})\right)\right] - \sum_{y\in\mathcal{Y}}\mathbb{E}_q\left[\log\left(q(\boldsymbol{\theta}_{X|y})\right)\right] - \mathbb{E}_q\left[\log\left(q(\boldsymbol{\alpha})\right)\right]. \qquad \text{(B.1)}\end{aligned}$$

Then, the five terms of equation (B.1) are expanded as functions of the variational parameters. To this end some results reported in Blei et al. (2003) and Kim et al. (2013) have been used.

**Lemma 5 (Blei et al. (2003)).** *Let $\boldsymbol{\eta}$ be a parameter vector whose distribution is Dirichlet with parameter $\boldsymbol{\beta}$, then the expected value of the logarithm of $\eta_x$ is*

$$\mathbb{E}[\log(\eta_x)] = \psi(\beta_x) - \psi\left(\sum_{x'\in\mathcal{X}} \beta_{x'}\right), \tag{B.2}$$

*where $\psi(\cdot)$ is the digamma function, derivative of the log Gamma function.*

**Lemma 6 (Kim et al. (2013)).** *Let $\boldsymbol{\eta}$ be a parameter vector whose distribution is Dirichlet with parameter $\boldsymbol{\beta}$ and $\gamma \in \mathbb{R}^+$, then*

$$\begin{aligned}\mathbb{E}[\log\Gamma(\gamma\eta_x)] \leq \log\Gamma(\gamma\mathbb{E}[\eta_x]) + \frac{\gamma(1-\mathbb{E}[\eta_x])}{\sum_{x'\in\mathcal{X}}\beta_{x'}} + (1-\gamma\mathbb{E}[\eta_x])(\log(\mathbb{E}[\eta_x]) \\ + \psi\left(\sum_{x'\in\mathcal{X}}\beta_{x'}\right) - \psi(\beta_x))\end{aligned} \tag{B.3}$$

*and the bound is tight.*

Thanks to (B.2), we can expand the terms of (B.1) as:

$$\mathbb{E}_q\left[\log\left(p(D|\boldsymbol{\theta}_{X|y})\right)\right] = \sum_{x\in\mathcal{X}} n_{xy}(\psi(\nu_{xy}) - \psi(\nu_{\cdot y})),$$

where $\nu_{\cdot y} = \sum_{x\in\mathcal{X}} \nu_{xy}$,

$$\begin{aligned}
\mathbb{E}_q\left[\log\left(p(\boldsymbol{\theta}_{X|y}|\boldsymbol{\alpha})\right)\right] &= \log\Gamma(s) - \sum_{x\in\mathcal{X}}\mathbb{E}_q[\log\Gamma(\alpha_x)] - \sum_{x\in\mathcal{X}}(s\kappa_x - 1)(\psi(\tau\kappa_x) - \psi(\tau)),\\
\mathbb{E}_q\left[\log\left(p(\boldsymbol{\alpha})\right)\right] &= \sum_{x\in\mathcal{X}}([\boldsymbol{\alpha}_0]_x - 1)(\psi(\tau\kappa_x) - \psi(\tau)) + \log\Gamma\left(s_0\right) - \sum_{x\in\mathcal{X}}\log\Gamma([\boldsymbol{\alpha}_0]_x),\\
\mathbb{E}_q\left[\log\left(q(\boldsymbol{\theta}_{X|y}|D)\right)\right] &= \log\Gamma(\nu_{\cdot y}) - \sum_{x\in\mathcal{X}}\log\Gamma(\nu_{xy}) + \sum_{x\in\mathcal{X}}(\nu_{xy} - 1)(\psi(\nu_{xy}) - \psi(\nu_{\cdot y})),\\
\mathbb{E}_q\left[\log\left(q(\boldsymbol{\alpha})|D\right)\right)] &= \log\Gamma(\tau) - \sum_{x\in\mathcal{X}}\log\Gamma(\tau\kappa_x) + \sum_{x\in\mathcal{X}}(\tau\kappa_x - 1)(\psi(\tau\kappa_x) - \psi(\tau)).
\end{aligned}$$

Since we cannot compute analytically the quantity $\mathbb{E}_q[\log \Gamma(\alpha_x)]$, we bound the second term of $\mathcal{L}$ by means of (B.3):

$$\mathbb{E}_q\left[\log\left(p(\boldsymbol{\theta}_{X|y}|\boldsymbol{\alpha})\right)\right] \geq \log\Gamma(s) - \sum_{x\in\mathcal{X}} \log\Gamma(s\kappa_x) - \frac{s}{\tau}\left(|\mathcal{X}|-1\right) + \\ + \sum_{x\in\mathcal{X}}(s\kappa_x - 1)(\log(\kappa_x) - \psi(\tau\kappa_x) + \psi(\tau) + \psi(\nu_{xy}) - \psi(\nu_{\cdot y})).$$

The ELBO can be thus lower bounded by means of the quantity

$$\begin{aligned}\tilde{\mathcal{L}} &= \sum_{y\in\mathcal{Y}}\sum_{x\in\mathcal{X}} n_{xy}(\psi(\nu_{xy}) - \psi(\nu_{\cdot y})) + |\mathcal{Y}|\log\Gamma(s) - |\mathcal{Y}|\sum_{x\in\mathcal{X}}\log\Gamma(s\kappa_x) + \\ &- \frac{s}{\tau}|\mathcal{Y}|\left(|\mathcal{X}|-1\right) + |\mathcal{Y}|\sum_{x\in\mathcal{X}}(s\kappa_x - 1)(\log(\kappa_x) - \psi(\tau\kappa_x) + \psi(\tau)) + \\ &+ \sum_{y\in\mathcal{Y}}\sum_{x\in\mathcal{X}}(s\kappa_x - 1)\Big(\psi(\nu_{xy}) - \psi(\nu_{\cdot y})\Big) + \log\Gamma\left(s_0\right) - \sum_{x\in\mathcal{X}}\log\Gamma([\boldsymbol{\alpha}_0]_x) + \\ &+ \sum_{x\in\mathcal{X}}([\boldsymbol{\alpha}_0]_x - 1)(\psi(\tau\kappa_x) - \psi(\tau)) - \sum_{y\in\mathcal{Y}}\log\Gamma(\nu_{\cdot y}) + \sum_{y\in\mathcal{Y}}\sum_{x\in\mathcal{X}}\log\Gamma(\nu_{xy}) + \\ &- \sum_{y\in\mathcal{Y}}\sum_{x\in\mathcal{X}}(\nu_{xy} - 1)(\psi(\nu_{xy}) - \psi(\nu_{\cdot y})) - \log\Gamma(\tau) + \sum_{x\in\mathcal{X}}\log\Gamma(\tau\kappa_x) + \\ &- \sum_{x\in\mathcal{X}}(\tau\kappa_x - 1)(\psi(\tau\kappa_x) - \psi(\tau)).\end{aligned}$$

### *Appendix B.2. Parameter estimation:* $\boldsymbol{\nu}_y$

In order to maximise $\tilde{\mathcal{L}}$ with respect to $\nu_{x'y'}$ we compute the partial derivative of $\tilde{\mathcal{L}}$ with respect to $\nu_{x'y'}$ and we set it to zero.

The partial derivative with respect to $\nu_{x'y'}$ is

$$\frac{\partial\tilde{\mathcal{L}}_{[\boldsymbol{\nu}_y]}}{\partial\nu_{x'y'}} = (\psi'(\nu_{x'y'}) - \psi'(\nu_{\cdot y'}))(n_{x'y'} + s\kappa_{x'} - \nu_{x'y'}).$$

Assuming $\boldsymbol{\kappa}$ to be fixed to a known value, the estimate for $\nu_{xy}$ is obtained by setting the partial derivative of $\tilde{\mathcal{L}}$ with respect to $\nu_{xy}$ to zero, i.e.,

$$\hat{\nu}_{xy} = n_{xy} + s\kappa_x.$$

*Appendix B.3. Parameter estimation: $\tau$ and $\boldsymbol{\kappa}$*

Given a value for $\boldsymbol{\nu} = \left(\boldsymbol{\nu}_1, \dots, \boldsymbol{\nu}_{|\mathcal{Y}|}\right)$, we propose to estimate parameters $\tau$ and $\boldsymbol{\kappa}$ by means of a fixed-point method, which alternates the optimisation of $\tilde{\mathcal{L}}$ with respect to $\tau$ and the optimisation of the same quantity with respect to $\boldsymbol{\kappa}$. Since we cannot find analytical solutions for the two optimisation problems, we propose for both of them a numerical optimisation performed by means of a Newton algorithm.

The Newton update for the parameter $\tau$ is obtained by considering the parameter vector $\boldsymbol{\kappa}$ fixed to a known value. If we define $g_\tau(\tau, \boldsymbol{\kappa}) = \partial\tilde{\mathcal{L}}/\partial\tau$ and $h_\tau(\tau, \boldsymbol{\kappa}) = \partial^2\tilde{\mathcal{L}}/\partial\tau^2$, the Newton update for the parameter $\tau$ becomes

$$\hat{\tau}_{k+1} = \hat{\tau}_k \exp\left(-\frac{g_\tau(\hat{\tau}_k, \boldsymbol{\kappa})}{h_\tau(\hat{\tau}_k, \boldsymbol{\kappa})\hat{\tau}_k + g_\tau(\hat{\tau}_k, \boldsymbol{\kappa})}\right), \tag{B.4}$$

where $\hat{\tau}_k$ represents the estimate for the parameter $\tau$ obtained in the previous iteration of the Newton algorithm.

The Newton update for the parameter vector $\boldsymbol{\kappa}$ is obtained by considering the parameters $\tau$ and $\boldsymbol{\nu}$ fixed to a known value. If we define $g_{\kappa_x}(\boldsymbol{\kappa}, \tau, \boldsymbol{\nu}) = \partial\mathcal{L}/\partial\kappa_x$ and $h_{\kappa_x}(\boldsymbol{\kappa}, \tau, \boldsymbol{\nu}) = \partial^2\tilde{\mathcal{L}}/\partial\kappa_x^2$ we can write the Newton update for the element $x$ of the parameter vector as

$$\kappa_x^{k+1} = \kappa_x^k + \frac{\displaystyle\sum_{x'\in\mathcal{X}} \frac{g_{k_{x'}}(\boldsymbol{\kappa}^k, \tau, \boldsymbol{\nu})}{h_{k_{x'}}(\boldsymbol{\kappa}^k, \tau, \boldsymbol{\nu})}}{\displaystyle\sum_{x'\in\mathcal{X}} \frac{h_{\kappa_x}(\boldsymbol{\kappa}^k, \tau, \boldsymbol{\nu})}{h_{k_{x'}}(\boldsymbol{\kappa}^k, \tau, \boldsymbol{\nu})}} - \frac{g_{k_{x'}}(\boldsymbol{\kappa}^k, \tau, \boldsymbol{\nu})}{h_{\kappa_x}(\boldsymbol{\kappa}^k, \tau, \boldsymbol{\nu})}, \tag{B.5}$$

where $\hat{\boldsymbol{\kappa}}_k$ represents the estimate for the parameter vector $\boldsymbol{\kappa}$ obtained in the previous iteration of the Newton algorithm.

*Appendix B.3.1. Newton step for $\tau$*

In order to derive a Newton algorithm to maximise $\tilde{\mathcal{L}}$ with respect to $\tau$, we need to compute the first and second partial derivative of $\tilde{\mathcal{L}}$.

The partial derivative of $\tilde{\mathcal{L}}$ with respect to $\tau$ is

$$\begin{aligned}\frac{\partial \tilde{\mathcal{L}}_{[\tau]}}{\partial \tau} =&\sum_{x\in\mathcal{X}}(\psi'(\tau\kappa_x)\kappa_x - \psi'(\tau))([\boldsymbol{\alpha}_0]_x - \tau\kappa_x - |\mathcal{Y}|(s\kappa_x - 1)) - \psi(\tau) +\\ &+ \sum_{x\in\mathcal{X}}\psi(\tau\kappa_x)\kappa_x - \sum_{x\in\mathcal{X}}(\psi(\tau\kappa_x) - \psi(\tau))\kappa_x + \frac{s}{\tau^2}|\mathcal{Y}|\left(|\mathcal{X}| - 1\right)\\ =&\sum_{x\in\mathcal{X}}(\psi'(\tau\kappa_x)\kappa_x - \psi'(\tau))([\boldsymbol{\alpha}_0]_x - \tau\kappa_x - |\mathcal{Y}|(s\kappa_x - 1)) + \frac{s}{\tau^2}|\mathcal{Y}|\left(|\mathcal{X}| - 1\right),\end{aligned}$$

while the second partial derivative with respect to $\tau$ is

$$\begin{aligned}\frac{\partial^2 \tilde{\mathcal{L}}_{[\tau]}}{\partial \tau^2} =&\sum_{x\in\mathcal{X}}(\psi''(\tau\kappa_x)\kappa_x^2 - \psi''(\tau))([\boldsymbol{\alpha}_0]_x - \tau\kappa_x - |\mathcal{Y}|(s\kappa_x - 1)) +\\ &- \sum_{x\in\mathcal{X}}(\psi'(\tau\kappa_x)\kappa_x - \psi'(\tau))\kappa_x - \frac{2s}{\tau^3}|\mathcal{Y}|\left(|\mathcal{X}| - 1\right)\\ =&\sum_{x\in\mathcal{X}}(\psi''(\tau\kappa_x)\kappa_x^2 - \psi''(\tau))([\boldsymbol{\alpha}_0]_x - \tau\kappa_x - |\mathcal{Y}|(s\kappa_x - 1)) +\\ &- \sum_{x\in\mathcal{X}}\psi'(\tau\kappa_x)\kappa_x^2 + \psi'(\tau) - \frac{2s}{\tau^3}|\mathcal{Y}|\left(|\mathcal{X}| - 1\right).\end{aligned}$$

Since the parameter $\tau$ is always positive, we derive a Newton algorithm for the update of $\log(\tau)$. If we define $g_\tau(\tau,\boldsymbol{\kappa}) = \partial\tilde{\mathcal{L}}_{[\tau]}/\partial\tau$ and $h_\tau(\tau,\boldsymbol{\kappa}) = \partial^2\tilde{\mathcal{L}}_{[\tau]}/\partial\tau^2$, we have that

$$\begin{aligned}\frac{\partial \tilde{\mathcal{L}}_{[\tau]}}{\partial \log(\tau)} &= \frac{\partial \tilde{\mathcal{L}}_{[\tau]}}{\partial \tau}\left(\frac{\partial \log(\tau)}{\partial \tau}\right)^{-1} = g_\tau(\tau,\boldsymbol{\kappa})\tau;\\ \frac{\partial^2 \tilde{\mathcal{L}}_{[\tau]}}{\partial (\log(\tau))^2} &= \frac{\partial}{\partial \log(\tau)}\left(\frac{\partial \tilde{\mathcal{L}}_{[\tau]}}{\partial \log(\tau)}\right) = \frac{\partial(g_\tau(\tau,\boldsymbol{\kappa})\tau)}{\partial\tau}\left(\frac{\partial \log(\tau)}{\partial \tau}\right)^{-1}\\ &= (h_\tau(\tau,\boldsymbol{\kappa})\tau + g_\tau(\tau,\boldsymbol{\kappa}))\tau.\end{aligned}$$

Assuming the parameter vector $\boldsymbol{\kappa}$ to be fixed to a known value, the Newton update at step $k$ is

$$\Delta\log(\tau^k) = -\frac{\partial\tilde{\mathcal{L}}_{[\tau]}/\partial\log(\tau)}{\partial^2\tilde{\mathcal{L}}_{[\tau]}/\partial(\log(\tau))^2} = -\frac{g_\tau(\tau^k,\boldsymbol{\kappa})}{h_\tau(\tau^k,\boldsymbol{\kappa})\tau + g_\tau(\tau^k,\boldsymbol{\kappa})}$$

and the updated parameter $\tau^{k+1}$ can be obtained as

$$\tau^{k+1} = \tau^k\exp(\Delta\log(\tau^k)).$$

*Appendix B.3.2. Newton step for $\boldsymbol{\kappa}$*

In order to derive a Newton algorithm to maximise $\tilde{\mathcal{L}}$ with respect to $\boldsymbol{\kappa}$, we need to compute the first and second partial derivative of $\tilde{\mathcal{L}}$ with respect to $\kappa_x$.

The partial derivative of $\tilde{\mathcal{L}}_{[\boldsymbol{\kappa}]}$ with respect to $\kappa_{x'}$ is

$$\begin{aligned}\frac{\partial \tilde{\mathcal{L}}_{[\boldsymbol{\kappa}]}}{\partial \kappa_{x'}} =&\tau\psi'(\tau\kappa_{x'})([\boldsymbol{\alpha}_0]_{x'} - \tau\kappa_{x'} - |\mathcal{Y}|(s\kappa_{x'}-1)) - \psi(\tau\kappa_{x'})(\tau + |\mathcal{Y}|s) +\\ &- s|\mathcal{Y}|\psi(s\kappa_{x'}) + |\mathcal{Y}|s\log(\kappa_{x'}) + |\mathcal{Y}|\frac{(s\kappa_{x'}-1)}{\kappa_{x'}} + \tau\psi(\tau\kappa_{x'}) + \sum_{y\in\mathcal{Y}} s\psi(\nu_{x'y})\\ =&\tau\psi'(\tau\kappa_{x'})([\boldsymbol{\alpha}_0]_{x'} - \tau\kappa_{x'} - |\mathcal{Y}|(s\kappa_{x'}-1)) - s|\mathcal{Y}|(\psi(\tau\kappa_{x'}) + \psi(s\kappa_{x'}) +\\ &- \log(\kappa_{x'}) - 1) - \frac{|\mathcal{Y}|}{\kappa_{x'}} + \sum_{y\in\mathcal{Y}} s\psi(\nu_{x'y}).\end{aligned}$$

The second partial derivative with respect to $\kappa_{x'}$ is

$$\begin{aligned}\frac{\partial^2 \tilde{\mathcal{L}}_{[\boldsymbol{\kappa}]}}{\partial \kappa_{x'}^2} =&\tau^2\psi''(\tau\kappa_{x'})([\boldsymbol{\alpha}_0]_{x'} - \tau\kappa_{x'} - |\mathcal{Y}|(s\kappa_{x'}-1)) - \tau\psi'(\tau\kappa_{x'})(\tau + s|\mathcal{Y}|) +\\ &- s|\mathcal{Y}|\Big(\tau\psi'(\tau\kappa_{x'}) + s\psi'(s\kappa_{x'}) - \frac{1}{\kappa_{x'}}\Big) + \frac{|\mathcal{Y}|}{\kappa_{x'}^2}\\ =&\tau^2\psi''(\tau\kappa_{x'})([\boldsymbol{\alpha}_0]_{x'} - \tau\kappa_{x'} - |\mathcal{Y}|(s\kappa_{x'}-1)) - \tau\psi'(\tau\kappa_{x'})(\tau + 2s|\mathcal{Y}|) +\\ &- s^2|\mathcal{Y}|\psi'(s\kappa_{x'}) + \frac{s|\mathcal{Y}|}{\kappa_{x'}} + \frac{|\mathcal{Y}|}{\kappa_{x'}^2},\end{aligned}$$

while all the mixed second order derivatives $\frac{\partial^2 \tilde{\mathcal{L}}_{[\boldsymbol{\kappa}]}}{\partial \kappa_{x'}\partial \kappa_{x''}}$ are equal to zero.

Since $\sum_{x\in\mathcal{X}} \kappa_x = 1$, we need to use a constrained Newton method to optimise $\tilde{\mathcal{L}}$. Specifically, at step $k$ we need to obtain a Newton update $\Delta\boldsymbol{\kappa}^k$ such that $\sum_{x\in\mathcal{X}} \Delta\kappa_x^k = 0$. As a consequence, the system to be solved is:

$$\begin{bmatrix} H^k & \mathbf{1} \\ \mathbf{1}' & \mathbf{0} \end{bmatrix} \begin{bmatrix} \Delta\boldsymbol{\kappa}^k \\ \boldsymbol{u} \end{bmatrix} = \begin{bmatrix} -\boldsymbol{g}^k \\ \mathbf{0} \end{bmatrix},$$

where the elements of $\boldsymbol{u}$ are the dual variables for the constrain, $H^k$ is the Hessian matrix at step $k$, $\mathbf{1}$ is the identity matrix and $\boldsymbol{g}^k$ is the gradient vector at step $k$. We define $g_{\kappa_x}(\boldsymbol{\kappa}, \tau, \boldsymbol{\nu}) = \partial\tilde{\mathcal{L}}_{[\boldsymbol{\kappa}]}/\partial\kappa_x$ and $h_{\kappa_x}(\boldsymbol{\kappa}, \tau, \boldsymbol{\nu}) = \partial^2\tilde{\mathcal{L}}_{[\boldsymbol{\kappa}]}/\partial\kappa_x^2$. Given a value for the parameters $\tau$ and $\boldsymbol{\nu}$, we can write the element $x$ of the gradient vector as $[\boldsymbol{g}^k]_x = g_{\kappa_x}(\boldsymbol{\kappa}^k, \tau, \boldsymbol{\nu})$. In the specific case under exam the Hessian matrix is diagonal with elements $[\boldsymbol{h}^k]_x = h_{\kappa_x}(\boldsymbol{\kappa}^k, \tau, \boldsymbol{\nu})$, i.e., $H = \mathrm{diag}(\boldsymbol{h}^k)$. The constrained Newton step for the element $x$ of the vector is thus

$$\Delta\kappa_x^k = \frac{1}{h_x^k}\left(\frac{\sum_{x'\in\mathcal{X}} g_{x'}^k / h_{x'}^k}{\sum_{x'\in\mathcal{X}} 1/h_{x'}^k} - g_x^k\right).$$

See, e.g., Kim et al. (2013) for a similar optimisation method.

## References

Bagnall, A., Cawley, G. C., 2017. On the use of default parameter settings in the empirical evaluation of classification algorithms. arXiv preprint arXiv:1703.06777.

Bartlett, M., Cussens, J., 2015. Integer linear programming for the Bayesian network structure learning problem. Artificial Intelligence.

Blei, D. M., Kucukelbir, A., McAuliffe, J. D., 2017. Variational inference: A review for statisticians. Journal of the American Statistical Association 112 (518), 859–877.

Blei, D. M., Ng, A. Y., Jordan, M. I., 2003. Latent dirichlet allocation. J. Mach. Learn. Res. 3, 993–1022.

Bradley, A. P., 1997. The use of the area under the ROC curve in the evaluation of machine learning algorithms. Pattern recognition 30 (7), 1145–1159.

Bucher, D., Cellina, F., Mangili, F., Raubal, M., Rudel, R., Rizzoli, A. E., Elabed, O., 2016. Exploiting Fitness Apps for Sustainable Mobility - Challenges Deploying the GoEco! App. 4th Int. Conf. ICT Sustain. (ICT4S 2016) (September), 89–98.

Campos, C. P. d., Ji, Q., 2011. Efficient structure learning of Bayesian networks using constraints. Journal of Machine Learning Research 12 (Mar), 663–689.

Carpenter, B., Gelman, A., Hoffman, M., Lee, D., Goodrich, B., Betancourt, M., Brubaker, M., Guo, J., Li, P., Riddell, A., 2017. Stan: A probabilistic programming language. Journal of Statistical Software 76 (1), 1–32.

Casella, G., Moreno, E., 2009. Assessing robustness of intrinsic tests of independence in two-way contingency tables. J. Am. Stat. Assoc. 104 (487), 1261–1271.

Table B.1: Data sets for classification analysis sorted in terms of number of instances. Marked data sets (*) are part of Weka agricultural data sets *agridatasets*. All the other data sets are part of the *UCI Machine Learning* repository.

| # | Name | instances | attributes | classes | missing |
|---|---|---|---|---|---|
| 1 | Connect-4 | 67557 | 42 | 3 | - |
| 2 | Adult | 45222 | 14 | 2 | replaced |
| 3 | Letter | 20000 | 16 | 26 | - |
| 4 | Nursery | 12960 | 8 | 5 | - |
| 5 | Pendigits | 10992 | 16 | 10 | - |
| 6 | Mushroom | 8124 | 22 | 2 | replaced |
| 7 | Satimage | 6435 | 36 | 6 | - |
| 8 | Shuttle | 5800 | 9 | 7 | - |
| 9 | Optdigits | 5620 | 64 | 10 | - |
| 10 | Page blocks | 5473 | 10 | 5 | - |
| 11 | Waveform | 5000 | 21 | 3 | - |
| 12 | Spambase | 4601 | 57 | 2 | replaced |
| 13 | Hypothiroid | 3772 | 29 | 4 | replaced |
| 14 | Chess | 3196 | 36 | 2 | - |
| 15 | Splice | 3190 | 60 | 3 | - |
| 16 | Segmentation | 2310 | 19 | 7 | - |
| 17 | Yeast | 1484 | 8 | 10 | - |
| 18 | Cmc | 1473 | 9 | 3 | - |
| 19 | German credit | 1000 | 20 | 2 | - |
| 20 | Vowel | 990 | 12 | 11 | - |
| 21 | Anneal | 898 | 38 | 6 | - |
| 22 | Diabetes | 768 | 8 | 2 | - |
| 23 | Eucalyptus* | 736 | 19 | 5 | replaced |
| 24 | Breast cancer | 699 | 9 | 2 | - |
| 25 | Credit approval | 690 | 15 | 2 | replaced |
| 26 | Soybean | 683 | 35 | 19 | - |
| 27 | Monk's 2 | 601 | 6 | 2 | - |
| 28 | Monk's 1 | 556 | 6 | 2 | - |
| 29 | Monk's 3 | 554 | 6 | 2 | - |
| 30 | Ionosphere | 351 | 34 | 2 | - |
| 31 | Liver disorders | 345 | 6 | 2 | - |
| 32 | Primary tumor | 339 | 17 | 22 | replaced |

| # | Name | instances | attributes | classes | missing |
|---|---|---|---|---|---|
| 33 | Ecoli | 336 | 7 | 8 | - |
| 34 | Solar flare C | 323 | 10 | 3 | - |
| 35 | Solar flare M | 323 | 10 | 4 | - |
| 36 | Solar flare X | 323 | 10 | 2 | - |
| 37 | Cleveland-heart | 303 | 13 | 2 | replaced |
| 38 | Hungary-heart | 303 | 13 | 2 | replaced |
| 39 | Spect | 267 | 22 | 2 | - |
| 40 | Audiology | 226 | 69 | 24 | replaced |
| 41 | Glass | 214 | 9 | 7 | - |
| 42 | Sonar | 208 | 60 | 2 | - |
| 43 | Wine | 178 | 13 | 3 | - |
| 44 | Hayes-Roth | 160 | 4 | 3 | - |
| 45 | Grub damage* | 155 | 8 | 4 | - |
| 46 | Hepatitis | 155 | 19 | 2 | replaced |
| 47 | Tae | 152 | 5 | 3 | - |
| 48 | Iris | 150 | 4 | 3 | - |
| 49 | Lymphography | 148 | 18 | 4 | - |
| 50 | Zoo | 101 | 16 | 7 | - |
| 51 | Post-operative | 90 | 8 | 3 | - |
| 52 | White clover* | 63 | 31 | 4 | - |
| 53 | Labor | 57 | 16 | 2 | replaced |
| 54 | Squash stored* | 52 | 24 | 3 | replaced |
| 55 | Squash unstored* | 52 | 23 | 3 | replaced |
| 56 | Pasture* | 36 | 22 | 3 | - |
| 57 | Lenses | 24 | 4 | 3 | - |

Cellina, F., Bucher, D., Rudel, R., Raubal, M., Andrea, E., 2016. Promoting Sustainable Mobility Styles Using Eco- Feedback and Gamification Elements : Introducing the GoEco! Living Lab Experiment. 4th Eur. Conf. Behav. Energy Effic. (Behave 2016) (September), 8–9.

Darwiche, A., 2009. Modeling and reasoning with Bayesian networks. Cambridge University Press.

Darwiche, A., 2010. Bayesian networks. Communications of the ACM 53 (12), 80–90.

Demichelis, F., Magni, P., Piergiorgi, P., Rubin, M. A., Bellazzi, R., 2006. A hierarchical naive Bayes model for handling sample heterogeneity in classification problems: an application to tissue microarrays. BMC bioinformatics 7 (1), 514.

Domingos, P., Pazzani, M., 1997. On the optimality of the simple Bayesian classifier under zero-one loss. Machine learning 29 (2), 103–130.

Fernández Delgado, M., Cernadas, E., Barro, S., Amorim, D., 2014. Do we need hundreds of classifiers to solve real world classification problems. Journal of Machine Learning Research 15 (1), 3133–3181.

Friedman, J., 1997. On bias, variance, 0/1 - loss, and the curse-of-dimensionality. Data Mining and Knowledge Discovery 1, 55–77.

Friedman, N., Geiger, D., Goldszmidt, M., 1997. Bayesian networks classifiers. Machine Learning 29 (2/3), 131–163.

Gelman, A., Carlin, J. B., Stern, H. S., Dunson, D. B., Vehtari, A., Rubin, D. B., 2014. Bayesian data analysis. Vol. 2. CRC press.

Grzegorczyk, M., Husmeier, D., 2012. A non-homogeneous dynamic bayesian network with sequentially coupled interaction parameters for applications in systems and synthetic biology. Statistical applications in genetics and molecular biology 11 (4).

Jaakkola, T. S., Jordan, M. I., 2000. Bayesian parameter estimation via variational methods. Statistics and Computing 10 (1), 25–37.

Jordan, M. I., Ghahramani, Z., Jaakkola, T. S., Saul, L. K., Nov 1999. An introduction to variational methods for graphical models. Machine Learning 37 (2), 183–233.

Kim, D.-k., Voelker, G., Saul, L., 2013. A variational approximation for topic modeling of hierarchical corpora. In: Proceedings of the 30th International Conference on Machine Learning. Vol. 28. PMLR, pp. 55–63.

Koller, D., Friedman, N., 2009. Probabilistic graphical models: principles and techniques. MIT press.

Kucukelbir, A., Tran, D., Ranganath, R., Gelman, A., Blei, D. M., 2017. Automatic differentiation variational inference. The Journal of Machine Learning Research 18 (1), 430–474.

Leday, G. G., de Gunst, M. C., Kpogbezan, G. B., Van der Vaart, A. W., Van Wieringen, W. N., Van de Wiel, M. A., 2017. Gene network reconstruction using global-local shrinkage priors. The Annals of Applied Statistics 11 (1), 41.

Malovini, A., Barbarini, N., Bellazzi, R., De Michelis, F., 2012. Hierarchical naive Bayes for genetic association studies. BMC bioinformatics 13 (14), S6.

Murphy, K. P., 2012. Machine learning: a probabilistic perspective. MIT press.

Niculescu-Mizil, A., Caruana, R., 2007. Inductive transfer for Bayesian network structure learning. In: Proc. Artificial Intelligence and Statistics. pp. 339–346.

Oates, C. J., Smith, J. Q., Mukherjee, S., Cussens, J., Jul 2016. Exact estimation of multiple directed acyclic graphs. Statistics and Computing 26 (4), 797–811.

Pan, S. J., Yang, Q., 2010. A survey on transfer learning. IEEE Transactions on knowledge and data engineering 22 (10), 1345–1359.

Petitjean, F., Buntine, W., Webb, G. I., Zaidi, N., May 2018. Accurate parameter estimation for bayesian network classifiers using hierarchical dirichlet processes. Machine Learning.

Scanagatta, M., Corani, G., de Campos, C., Zaffalon, M., 2016. Learning treewidth-bounded Bayesian networks with thousands of variables. In: Lee, D. D., Sugiyama, M., Luxburg, U. V., Guyon, I., Garnett, R. (Eds.), NIPS 2016: Advances in Neural Information Processing Systems 29.

Scutari, M., 2010. Learning Bayesian networks with the bnlearn R package. Journal of Statistical Software 35 (3).

Teh, Y. W., Jordan, M. I., Beal, M. J., Blei, D. M., 2006. Hierarchical dirichlet processes. Journal of the American Statistical Association 101 (476), 1566–1581.

Wainberg, M., Alipanahi, B., Frey, B. J., 2016. Are random forests truly the best classifiers? The Journal of Machine Learning Research 17 (1), 3837–3841.

Wainwright, M. J., Jordan, M. I., Jan. 2008. Graphical models, exponential families, and variational inference. Found. Trends Mach. Learn. 1 (1-2), 1–305.

Webb, G. I., Boughton, J. R., Wang, Z., 2005. Not so naive Bayes: aggregating one-dependence estimators. Machine learning 58 (1), 5–24.

Yuan, C., Malone, B., Wu, X., 2011. Learning optimal Bayesian networks using A* search. In: Proc. IJCAI - international joint conference on artificial intelligence. Vol. 22. p. 2186.