DEPARTAMENTO DE INTELIGENCIA ARTIFICIAL

Escuela Técnica Superior de Ingenieros Informáticos
Universidad Politécnica de Madrid

PhD THESIS

Multidimensional clustering with Bayesian networks

Author

Fernando Rodríguez Sánchez
MSc Artificial Intelligence

PhD supervisors

Pedro Larrañaga
PhD Computer Science

Concha Bielza
PhD Computer Science

2021

Thesis Committee

President: Serafín Moral Callejón

Member: Antonio Salmerón Cerdán

Member: Mª Carmen Rodríguez Blázquez

Member: Bojan Mihaljevic

Secretary: Antonio Fernández del Pozo

Acknowledgements

First and foremost, I want to thank my supervisors, Pedro Larrañaga and Concha Bielza, for their dedicated time, support, and motivation. Their corrections and suggestions have been essential to conduct this research and I am grateful for all the acquired knowledge. I want to also express my gratitude to Pablo Martínez, Carmen Rodríguez, Daniel Weintraub, Anette Schrag, Kim Ray Chaudhuri, and Alexandra Rizos for all of their insights and comments on the Parkinson's disease study.

I appreciate the financial support of the following projects and institutions: "Multi-view clustering with Bayesian networks" project funded by Fundación BBVA grants to Scientific Research Teams in Big Data 2016; Cajal Blue Brain Project (C080020-09), TIN2016-79684-P, and PID2019-109247GB-I00 projects funded by the Spanish Ministry of Education, Culture and Sport; S2013/ICE-2845-CASI-CAM-CM project funded by the Regional Government of Madrid; Human Brain Project SGA3 funded by the European Union's Horizon 2020 Framework Programme for Research and Innovation under the Specific Grant Agreement No. 945539; and "Score-based nonstationary temporal Bayesian networks. Applications in climate and neuroscience" project funded by Fundación BBVA grants to Scientific Research Teams in Big Data 2019.

I am very grateful to my colleagues and friends at the Computational Intelligence Group. Their help and support over the past few years have proven to be invaluable: David Atienza, Irene, Gabriel, David Quesada, Carlos, Nikolas, Sergio, Marco, Bojan, Gherardo, Luis, Alberto, Iñaki, Laura Antón, Vicente, Santiago, Mario, Javier, Pablo, and Laura González.

Finally, my greatest of gratitudes goes to my family. I am grateful for their unconditional love and encouragement.

Abstract

The evolution of communication and a continued globalization process have resulted in bigger quantities of data being storaged. However, data has not only increased in volume but also in complexity. Nowadays, more and more data is collected from different measurement methods. In this context, traditional clustering algorithms are unable to comprehensively describe all of the contained information. That is why new clustering techniques that consider multiple dimensions of data are more necessary than ever. One of these techniques is multidimensional clustering, which extends model-based clustering by learning mixture models with multiple categorical latent variables. Each latent variable identifies a dimension along which data are partitioned into clusters. Each dimension is conformed of a different subset of domain variables.

Bayesian networks are useful in multidimensional clustering for several reasons. First, their graphical structure allows for an easier interpretation, showing which variables are relevant for each clustering. Second, their conditional independences result in more compact models that are easier to learn. Finally, Bayesian networks support probabilistic inference, which is useful for making predictions, diagnoses and explanations.

In this dissertation we explore the problem of learning Bayesian network models for multidimensional clustering. Although there is an extensive literature on multidimensional clustering methods for categorical data and for continuous data, there is a lack of work for mixed data (i.e., data that is composed of both categorical and continuous variables). For this reason, we propose approaches that are able to efficiently deal with mixed data by exploiting the Bayesian network factorization and the variational Bayes framework. More specifically, we make the following contributions.

First, we present an incremental algorithm for learning conditional linear Gaussian Bayesian networks with categorical latent variables whose structures are restricted to forests. The learning process is divided in two phases. In the first phase, the forest structure is expanded with a new arc or latent variable. In the second phase, the cardinalities of latent variables are estimated. Furthermore, we devise a variant of this algorithm that only considers a subset of the possible structures and demonstrate the effectiveness of the approach.

Second, we develop a greedy algorithm for learning conditional linear Gaussian Bayesian networks with categorical latent variables that are not restricted to tree-like structures. To this purpose, the proposed method hill-climbs the space of models using a series of latent operators and a variational Bayesian version of the structural expectation-maximization algorithm.

Finally, we present a multidimensional clustering study with Parkinson's disease data where

we apply the proposed methodology. We consider data from a large, multi-center, international, and well-characterized cohort of patients. As a result, eight sets of motor and non-motor symptoms are identified. Each of them provides a different way to group patients: impulse control issues, overall non-motor symptoms, presence of dyskinesias and psychosis, fatigue, axial symptoms and motor fluctuations, autonomic dysfunction, depression, and excessive sweating.

Resumen

La evolución de la comunicación y un proceso de globalización continuado han dado lugar a que se almacenen mayores cantidades de datos. Sin embargo, los datos no solo han aumentado en volumen sino también en complejidad. Hoy en día, cada vez se recopilan más datos de diferentes métodos de medición. En este contexto, los algoritmos de *clustering* tradicionales no pueden describir de manera integral toda la información contenida. Es por esto que son necesarias nuevas técnicas de *clustering* que consideren múltiples dimensiones de los datos. Una de estas técnicas es el *clustering* multidimensional, el cual extiende el *clustering* basado en modelos al aprender modelos de mixturas con múltiples variables latentes categóricas. Cada una de estas variables latentes identifica una dimensión en la cual los datos son divididos en grupos. Cada dimensión se conforma de un conjunto diferente de variables observadas.

Las redes Bayesianas son interesantes en el *clustering* multidimensional por varias razones. En primer lugar, su estructura gráfica permite una interpretación más sencilla, mostrando qué variables son relevantes para cada *clustering*. En segundo lugar, sus independencias condicionales dan como resultado modelos más compactos y fáciles de aprender. Finalmente, las redes Bayesianas ofrecen la posibilidad de emplear inferencia probabilística, la cual es útil para realizar predicciones, diagnósticos y explicaciones.

En esta tesis, exploramos el problema de aprender modelos de redes Bayesianas para *clustering* multidimensional. Si bien existe una extensa literatura sobre métodos de *clustering* multidimensional con datos categóricos y con datos continuos, no se ha estudiado en profundidad el caso de datos mixtos (i.e., aquellos que se encuentran formados tanto de variables categóricas como de variables continuas). Por este motivo, proponemos varios enfoques que pueden tratar de manera eficiente con datos mixtos mediante la explotación de la factorización de la red Bayesiana y del *framework* variacional Bayesiano. Más concretamente, realizamos las siguientes aportaciones.

En primer lugar, presentamos un algoritmo incremental para el aprendizaje de redes Bayesianas Gaussianas lineales condicionales con variables latentes categóricas cuyas estructuras se encuentran restringidas a bosques. El proceso de aprendizaje se divide en dos fases. En la primera fase, la estructura del bosque se expande con un nuevo arco o variable latente. En la segunda fase, se estiman las cardinalidades de las variables latentes. Además, diseñamos una variante de este algoritmo que solo considera un subconjunto de las posibles estructuras y demostramos la efectividad de este método.

En segundo lugar, desarrollamos un algoritmo voraz para el aprendizaje de redes Bayesianas Gaussianas lineales condicionales con variables latentes categóricas cuyas estructuras no se restringen a bosques. Nuestro método explora el espacio de modelos mediante una serie de operadores latentes y una versión variacional Bayesiana del algoritmo estructural de esperanza-

maximización.

Finalmente, presentamos un estudio de *clustering* multidimensional con datos de la enfermedad de Parkinson, donde aplicamos la metodología propuesta a lo largo de la tesis. En este estudio, consideramos datos de una cohorte de pacientes grande, multicéntrica, internacional, y bien caracterizada. Como resultado, identificamos ocho conjuntos de síntomas motores y no motores. Cada uno de ellos proporciona una forma diferente de agrupar a los pacientes: problemas de control de impulsos, síntomas generales no motores, presencia de discinesias y psicosis, fatiga, síntomas axiales y fluctuaciones motoras, disfunción autonómica, depresión, y sudoración excesiva.

Contents

List of Figures

List of Tables

Acronyms

AI Arc introduction

AIC Akaike information criterion

AE Arc elimination

AR Arc reversal

BI Bridged islands

BIC Bayesian information criterion

BN Bayesian network

CAMI Clustering for alternatives with mutual information

COALA Constrained orthogonal average link algorithm

CD Cardinality decrease

CE Conjugate-exponential

CFHLC Construction of forests of hierarchical latent class models

CI Cardinality increase

CIL Constrained incremental learner

CLI Conditional latent variable introduction

CPD Conditional probability distribution

CVPLL Cross-validated predictive log-likelihood

DAG Directed acyclic graph

DHC Double hill-climbing

EAST Expansion, adjustment, and simplification until termination

ELBO Evidence lower bound

EM Expectation-maximization

GEAST Gaussian expansion, adjustment, and simplification until termination

GLFM General latent feature model

HC Hill-climbing

HCL Hierarchical clustering learning

HI-VAE Heterogeneous incomplete variational autoencoder

HLCM Hierarchical latent class model

HLTA Hierarchical latent tree analysis

H&Y Hoehn & Yahr

ICRDs Impulse control and related disorders

IL Incremental learner

I-map Independence map

IPW Inverse probability weighting

JPD Joint probability distribution

KDE Kernel density estimator

KL Kullback-Leibler

LCM Latent class model

LDA Latent Dirichlet allocation

LI Latent variable introduction

LE Latent variable elimination

LL Log-likelihood

LTC Latent tree classifier

LTM Latent tree model

LVM Latent variable model

MAP Maximum a posteriori

MB Markov blanket

MDS-NMS Movement Disorder Society non-motor rating scale

MDS-UPDRS Movement Disorder Society unified Parkinson's disease rating scale

MI Mutual information

MICE Multiple imputation by chained equations

MLE Maximum likelihood estimation

MoP Mixture of polynomials

MoTBF Mixture of truncated basis functions

MSPN Mixed sum-product networks

MTE Mixture of truncated exponentials

NACI Non-linear alternative clustering with information theory

NMS Non-motor rating scale

p-ELBO Penalized evidence lower bound

PIGD Postural instability gait difficulty

PD Parkinson's disease

PGM Probabilistic graphical model

RG Recursive grouping

SD Standard deviation

SEM Structural expectation-maximization

SHC Single hill-climbing

uk-DB Unsupervised k-dependence Bayesian classifier

VB Variational Bayes

VB-EM Variational Bayesian expectation-maximization

VB-SEM Variational Bayesian structural expectation-maximization

VMP Variational message passing

Notations

X, Y, Z, ...	random variables
$\mathbf{X}$, $\mathbf{Y}$, $\mathbf{Z}$, ...	sets of random variables
x, y, z, ...	assignments to random variables
$\mathbf{x}$, $\mathbf{y}$, $\mathbf{z}$, ...	joint assignments to sets of random variables
$\Omega_{\mathbf{X}}$	set of all possible joint assignments to $\mathbf{X}$
$\mathbf{X} \perp \mathbf{Y}$	independence of $\mathbf{X}$ and $\mathbf{Y}$
$\mathbf{X} \perp \mathbf{Y} \vert \mathbf{Z}$	conditional independence of $\mathbf{X}$ and $\mathbf{Y}$ given $\mathbf{Z}$
$P(\mathbf{X})$	probability distribution of $\mathbf{X}$
$P(\mathbf{x})$	shorthand for $P(\mathbf{X} = \mathbf{x})$
$P(\mathbf{X}, \mathbf{Y})$	joint probability distribution of $\mathbf{X}$ and $\mathbf{Y}$
$P(\mathbf{X} \vert \mathbf{Y})$	conditional probability distribution of $\mathbf{X}$ given $\mathbf{Y}$
$Q(\mathbf{X})$	variational probability distribution of $\mathbf{X}$
$\mathcal{B}$	a Bayesian network
$\mathcal{G}$	the graph structure of a Bayesian network
$\mathcal{I}_l(\mathcal{G})$	set of local Markov independences encoded by $\mathcal{G}$
$\boldsymbol{\theta}$	parameters of a Bayesian network
$\hat{\boldsymbol{\theta}}$	maximum likelihood estimate of the parameters of a Bayesian network
$\boldsymbol{\theta}_{X_i \vert \mathbf{Pa}_i^{\mathcal{G}}}$	parameters of the CPD of X_i given its parents
$\boldsymbol{\varphi}$	parameters of a variational probability distribution Q
$\mathbf{Pa}_i^{\mathcal{G}}$	parent variables of variable X_i in $\mathcal{G}$
$\mathbf{Ch}_i^{\mathcal{G}}$	children variables of the variable X_i in $\mathcal{G}$
$\mathbf{Adj}_i^{\mathcal{G}}$	adjacent variables to X_i in $\mathcal{G}$ (its parents and children)
$\mathbf{MB}_i^{\mathcal{G}}$	Markov blanket of X_i in $\mathcal{G}$
$\mathcal{D}$	the training dataset
N	number of data instances in $\mathcal{D}$
m	number of variables in $\mathcal{D}$
$\mathbf{H}[n]$	unobserved variables in the nth data instance
$\mathbf{h}[n]$	an assignment to $\mathbf{H}$ in the nth data instance
$\mathcal{H}$	set of all possible assignments to all unobserved values in $\mathcal{D}$ (i.e., $\cup_n \mathbf{h}[n]$)
$\mathcal{R}_A$	arc restrictions in a structure learning method
$\text{score}(\mathcal{B} : \mathcal{D})$	score of $\mathcal{B}$ given a dataset $\mathcal{D}$
$\text{ELBO}(Q_{\mathcal{B}} : \mathcal{D})$	ELBO score of the variational distribution of $\mathcal{B}$ given $\mathcal{D}$

Part I

INTRODUCTION

Chapter 1

Introduction

Clustering is a fundamental tool for data exploration that finds interesting patterns by grouping objects based on some notion of similarity. Traditional clustering algorithms aim at finding the "best" solution at partitioning the data. However, the assumption of a single clustering solution is often untrue for real-world datasets, where several meaningful sets of clusters may exist, each of them associated to a specific subset of domain variables. For instance, one can imagine that different subsets of disease symptoms (e.g., muscular disorders, psychological issues, organ dysfunctions, etc.) that describe a cohort of patients can provide different ways of grouping those patients. In this setting, multidimensional clustering is a popular tool due to its probabilistic foundations and its flexibility. Multidimensional clustering [Zhang, 2004; Chen et al., 2012] extends model-based clustering [McLachlan and Peel, 2004; Melnykov and Maitra, 2010] by assuming the existence of a finite mixture model with multiple categorical latent variables, where each latent variable defines a clustering according to a different subset of domain variables.

However, working with a multidimensional clustering model is cumbersome because the number of model parameters increases quadratically with respect to the number of continuous variables, and exponentially with respect to the number of categorical (both observed and latent) variables. It is therefore necessary to exploit the conditional independences that may be present in the data. Conditional independence is a central concept to Bayesian networks (BNs) [Pearl, 1988; Koller and Friedman, 2009; Maathuis et al., 2018]. A BN is a probabilistic graphical model (PGM) that represents a joint probability distribution (JPD) as a product of conditional probability distributions (CPDs). It is a compact graphical representation in the form of a directed acyclic graph (DAG), where nodes correspond to random variables and arcs correspond to probabilistic dependences between them.

Learning a BN from data is typically performed by a search method that approaches the learning process from a model selection perspective. Search methods define a hypothesis space of potential models, a set of operators to navigate this space, and a scoring function that measures how well the model fits the observed data [Scanagatta et al., 2019]. When all the variables in the network are observed, the decomposability property of some BN scores allows for efficient learning. However, in the presence of latent variables it is not feasible to efficiently

learn a BN because scoring functions do not decompose. Friedman's structural expectation-maximization (SEM) [Friedman, 1997] is the most widely used algorithm for learning a BN in the presence of latent variables. It generalizes the expectation-maximization (EM) algorithm [Dempster et al., 1977; McLachlan and Krishnan, 2008] to the problem of BN learning.

When a latent variable is known to exist, we can introduce it into the BN model and apply SEM. However, when performing multidimensional clustering, we do not know either the appropriate number of latent variables, nor their respective cardinalities. We can address this problem by developing a search method that explores the space of multidimensional clustering models using a series of latent operators that introduce latent variables, remove latent variables, and change the cardinality of latent variables. Each application of these latent operators could be accompanied by a run of SEM, which introduces, removes, or reverses BN arcs.

There is an extensive literature on the identification of latent variables using BNs with categorical and continuous data. However, there still exists a lack of work when dealing with mixed data. Therefore, we intend to contribute to the multidimensional clustering research by proposing new approaches that can work with categorical, continuous, and mixed data. To this purpose, we develop several methods for learning conditional linear Gaussian BNs [Lauritzen and Wermuth, 1989] with categorical latent variables. Since learning this type of models is computationally unfeasible using exact inference procedures, we base our approaches on the variational Bayes (VB) framework [Attias, 2000]. We evaluate the performance of our approaches by solving both density estimation and missing data imputation tasks. In addition, we apply our proposal to the problem of identifying Parkinson's disease (PD) subtypes via multidimensional clustering.

Chapter outline

In Section 1.1, we present the research hypotheses and the main objectives of this dissertation. Then, in Section 1.2, we provide the document organization.

1.1 Hypotheses and objectives

The research hypotheses of this dissertation can be stated as the following two main points:

- The use of a score-based method that searches in the space of directed acyclic graphs in combination with the variational Bayes framework will allow us to learn conditional linear Gaussian Bayesian networks with categorical latent variables that accurately fit the data.
- Conditional linear Gaussian Bayesian networks with multiple categorical latent variables can be used to identify multiple Parkinson's disease subtypes via multidimensional clustering.

Based on these hypotheses, the main objectives of this dissertation are:

- To develop a new methodology for learning conditional linear Gaussian BNs with categorical latent variables where the structure is restricted to a forest.
- To develop a VB version of the SEM algorithm that is capable of dealing with incomplete mixed data.
- To develop a new methodology for learning conditional linear Gaussian BNs with categorical latent variables with an unrestricted structure.
- To identify multiple PD subtypes and associations between those subtypes via multidimensional clustering with conditional linear Gaussian BNs.

1.2 Document organization

The dissertation includes five parts and eight chapters that are organized as follows:

Part I. Introduction

This part introduces the dissertation.

- Chapter 1 presents the hypotheses and objectives of this work as well as the document organization.

Part II. Background

This part consists of three chapters that introduce the background of the topics addressed in this dissertation.

- Chapter 2 introduces BNs as an efficient framework for statistical modelling under uncertainty. In this chapter, we formally introduce the BN model with its parametrizations, we present the mechanism of probabilistic inference, and we address the problem of learning BNs.
- Chapter 3 provides an overview of latent tree models, a special type of BN with a rooted tree structure that contains multiple latent variables. In this chapter, we start with a formal definition of the latent tree model. Then, we present its learning process and applications.
- Chapter 4 presents an overview of model-based clustering with BNs. We start this chapter by discussing traditional model-based clustering, where a single clustering variable is considered. Then, we expand on this topic by considering multiple clustering variables.

Part III. Contributions

This part includes three chapters that present the main contributions of this dissertation.

- Chapter 5 addresses the problem of learning forests of tree-structured conditional linear Gaussian BNs with categorical latent variables. In this chapter, we propose two incremental algorithms and demonstrate their effectiveness in density estimation experiments using categorical, continuous, and mixed real-world data.
- Chapter 6 provides a method for learning conditional linear Gaussian BNs with categorical latent variables from incomplete mixed data. Our proposal greedily explores the space of models using a series of latent operators and a VB version of the SEM algorithm. We explore the performance of our method for the tasks of density estimation and missing data imputation with categorical, continuous, and mixed real-world data.
- Chapter 7 presents a multidimensional clustering study with PD data. In this chapter, we apply the methodology introduced in Chapter 6 to identify multiple subtypes of PD. Furthermore, we study the associations between these subtypes and discuss the clinical implications of the results.

Part IV. Conclusions

This part concludes the dissertation.

- Chapter 8 summarises the conclusions of this work, suggests future research lines, and provides the list of publications developed as a result of this research.

Part V. Appendices

This part provides supplementary information about the research.

- Appendix A provides the specification of the motor variables considered in the PD study of Chapter 7.

Part II

BACKGROUND

Chapter 2

Bayesian networks

2.1 Introduction

Most real-world problems imply dealing with uncertainty. This is a consequence of several factors such as partial information, stochastic environments, and measurement errors. Probability theory provides a well-established foundation for managing uncertainty, therefore it is natural to use it for reasoning under these conditions. However, if we naïvely apply probability to complex problems, we will soon be deterred by the resulting computational complexity [Sucar, 2015].

PGMs provide a computationally efficient framework for managing uncertainty that combines probability theory with graph theory. The main idea is to consider only those conditional probabilistic independence relations that are held in the problem data and encode them in a graph. By using a PGM, we can compactly and clearly represent the JPD of the data, graphically representing relevant relationships between the problem variables. We distinguish two components in a PGM: the graphical structure and the probabilistic content. In the graphical structure, nodes correspond to random variables and arcs correspond to probabilistic dependences between them. The probabilistic content quantifies these dependences using CPDs. The JPD is factorized according to the graphical structure as a product of those CPDs, which reduces the total number of model parameters and facilitates the computation of probabilistic queries.

Depending on the type of graphical structure, we can distinguish three main families of PGMs: (i) Markov networks [Kidermann and Snell, 1980], which assume an undirected graph to represent relations between the problem variables, (ii) BNs [Pearl, 1988], which assume a directed acyclic graph, and (iii) chain graphs [Lauritzen and Wermuth, 1989], which admit both directed and undirected arcs, but without any directed cycles. In this dissertation, we mainly work with BNs, as they are the most widely used PGM for reasoning with uncertainty. They are commonly used in fields as diverse as medical diagnosis [Pourret et al., 2008], maintenance planning [Jones et al., 2010], risk assessment [Fenton and Neil, 2012], neuroscience [Bielza and Larrañaga, 2014] and many more [Koller and Friedman, 2009].

Chapter outline

In Section 2.2, we start with a brief overview of how a graph can encode conditional independence relations and then formally introduce the BN model with its parametrizations. In Section 2.3, we present the mechanism of probabilistic inference. Finally, in Section 2.4, we overview the problem of learning BNs, where we put especial attention in explaining the differences between the complete and incomplete data case.

2.2 The Bayesian network model

A BN is a compact representation of a JPD over a set of random variables. We can distinguish two components in a BN: (i) a directed acyclic graphical structure that encodes conditional independence relations between the different variables, and (ii) a set of parameters that, together with the graphical structure, define a unique distribution. We start with a brief overview of how a graph encodes the conditional independence relations and then formally define the BN model with its parametrizations.

2.2.1 Encoding independences

At the core of BNs is the concept of conditional independence. Let P be a probability distribution with $\mathbf{X}_A$, $\mathbf{X}_B$ and $\mathbf{X}_C$ as three disjoint sets of variables. We say that $\mathbf{X}_A$ and $\mathbf{X}_B$ are conditionally independent given $\mathbf{X}_C$ if

$$P(\mathbf{X}_A, \mathbf{X}_B|\mathbf{X}_C) = P(\mathbf{X}_A|\mathbf{X}_C)P(\mathbf{X}_B|\mathbf{X}_C)$$

and we denote this statement by $(\mathbf{X}_A \perp \mathbf{X}_B|\mathbf{X}_C)$. As an example, the BN structure depicted in Figure 2.1 implies several conditional independence statements over its set of variables $\{X_1, X_2, X_3, X_4, X_5\}$: $(X_1 \perp X_2)$, $(X_3 \perp \{X_2, X_4\}|X_1)$, $(X_4 \perp \{X_3, X_5\}|\{X_1, X_2\})$ and $(X_5 \perp \{X_1, X_2, X_4\}|X_3)$. We see a pattern emerging from the independence statements of this example: the parents of a variable "shield" it from the probabilistic influence of other variables that are not its descendants. We can formalize this intuition.

Let $\mathcal{G}$ be a DAG whose nodes represent a set of random variables $\mathbf{X}$. Let $\mathbf{Pa}_i^{\mathcal{G}}$ denote the parents of X_i in $\mathcal{G}$, and $\text{NonDescendants}_{X_i}$ denote the variables in the graph that are nondescendants of X_i. Then $\mathcal{G}$ encodes the following set of conditional independence statements, called the local Markov independences, and denoted by $\mathcal{I}_l(\mathcal{G})$:

$$\forall X_i : \ (X_i \perp \text{NonDescendants}_{X_i}|\mathbf{Pa}_i^{\mathcal{G}}) \ .$$

The ability to infer conditional independences from a graph allows us to characterize the notion of Markov blanket (MB). The MB of any variable $X_i \in \mathbf{X}$ in $\mathcal{G}$ is the set of variables composed of the parents of X_i, its children and the parents of its children.

In general there are multiple conditional independence statements that can be derived from $\mathcal{I}_l(\mathcal{G})$. The notion of d-separation can be used to determine whether a specific statement

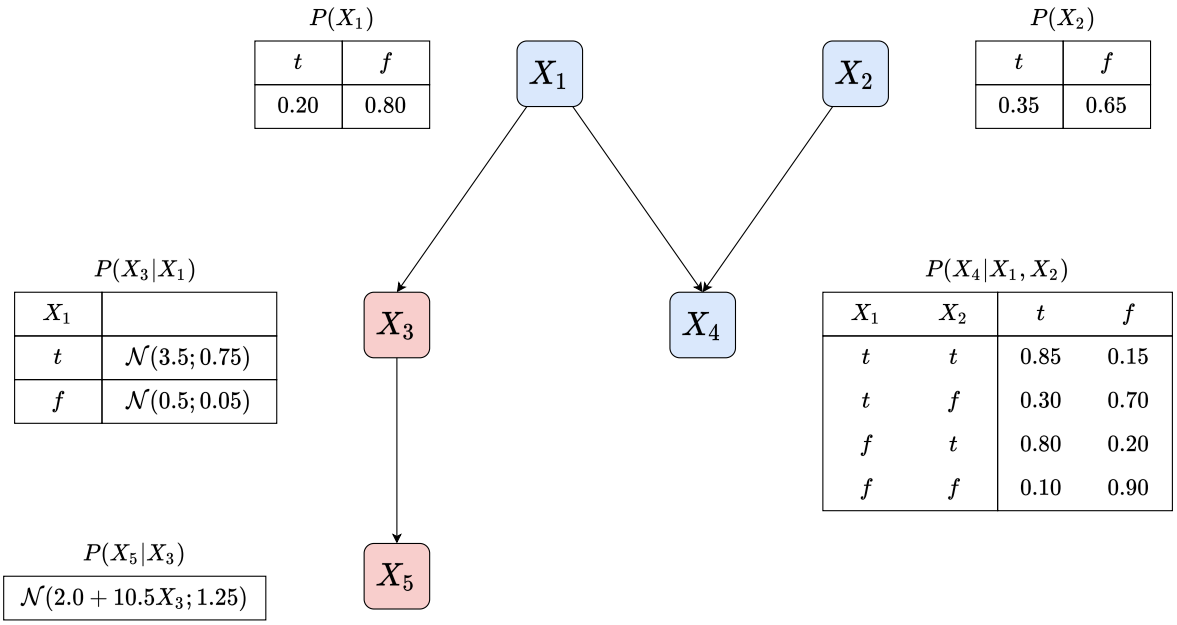


Figure 2.1: Example of a conditional linear Gaussian BN. Blue nodes represent categorical variables and red nodes represent continuous variables.

$(\mathbf{X}_A \perp \mathbf{X}_B|\mathbf{X}_C)$ holds. Let $\mathbf{X}_A$, $\mathbf{X}_B$ and $\mathbf{X}_C$ be three disjoint sets of nodes in a DAG $\mathcal{G}$. Let $\mathbf{T}$ be the set of possible trials from any node $X_a \in \mathbf{X}_A$ to any node $X_b \in \mathbf{X}_B$, where a trial in the graph is a succession of arcs, independent of their directions. Then $\mathbf{X}_C$ blocks a trial $T_I \in \mathbf{T}$ if one of the following holds:

1. T_I contains a chain $T_{i-1} \to T_i \to T_{i+1}$ such that $T_i \in \mathbf{X}_C$.
2. T_I contains a fork $T_{i-1} \leftarrow T_i \to T_{i+1}$ such that $T_i \in \mathbf{X}_C$.
3. T_I contains a collider $T_{i-1} \to T_i \leftarrow T_{i+1}$ such that T_i and any of its descendants do not belong to $\mathbf{X}_C$.

If all the trials in $\mathbf{T}$ are blocked by $\mathbf{X}_C$, then $\mathbf{X}_C$ d-separates $\mathbf{X}_A$ and $\mathbf{X}_B$. It is important to note that d-separation does not rule out additional conditional independences that may hold in the distribution and are not encoded by the DAG. For example, if $\mathbf{X}_A$ and $\mathbf{X}_B$ are d-separated given $\mathbf{X}_C$, then $(\mathbf{X}_A \perp \mathbf{X}_B|\mathbf{X}_C)$ holds in the distribution P. However, a lack of d-separation does not necessarily indicate that $(\mathbf{X}_A \perp \mathbf{X}_B|\mathbf{X}_C)$ does not hold in P. Finally, d-separation can be efficiently computed in time that is linear in the number of nodes in the graph [Geiger et al., 1990].

Since we are interested in DAGs that encode a JPD, we now define the relation between the topological properties of a DAG and the conditional independences of its underlying probability distribution. We say that a DAG $\mathcal{G}$ is an independence map (I-map) of the distribution P if all the conditional independences that can be derived from $\mathcal{I}_l(\mathcal{G})$ are satisfied by P. We can now formally define the BN model.

2.2.2 Model definition

A BN $\mathcal{B} = (\mathcal{G}, \boldsymbol{\theta})$ is a representation of a JPD over a set of random variables $\mathbf{X}$. It consists of two components: (i) a DAG $\mathcal{G}$ that encodes the conditional independences $\mathcal{I}_l(\mathcal{G})$, and (ii) a set of parameters $\boldsymbol{\theta}$ that describes the CPD $P(X_i|\mathbf{Pa}_i^{\mathcal{G}})$ of each variable $X_i \in \mathbf{X}$ given its parents in the graph. In addition, we require that $\mathcal{G}$ is an I-map of the JPD P represented by the BN. This property is key for allowing the BN to compactly factorize the JPD as a product of CPDs. It is formally expressed by the following theorem:

Theorem 2.1 [Pearl, 1988] *Let $\mathcal{G}$ be a BN graph over the set of variables $\mathbf{X}$. We say that $\mathcal{G}$ is an I-map of P if and only if P can be expressed as*

$$P(\mathbf{X}) = \prod_i P(X_i|\mathbf{Pa}_i^{\mathcal{G}}) \ . \tag{2.1}$$

Using this theorem, a BN is able to define a unique probability distribution that can be written as in Equation (2.1). This is called the chain rule for Bayesian networks. As an example, consider the distribution $P(X_1, X_2, X_3, X_4, X_5)$ of Figure 2.1. By taking into account the conditional independences represented by the graph, we can write:

$$P(X_1, X_2, X_3, X_4, X_5) = P(X_1)P(X_2)P(X_3|X_1)P(X_4|X_1, X_2)P(X_5|X_3) \ .$$

2.2.3 Parametrization

Depending on the nature of variables that are present in a BN, we usually distinguish between categorical, continuous, and hybrid (i.e., mixed) BNs. The latter is composed of both categorical and continuous variables. Each of these is presented in the following sections.

2.2.3.1 Categorical Bayesian networks

Categorical BNs are strictly composed of categorical variables (i.e., qualitative variables with a finite number of values). Examples include variables such as sex, eye color, blood type, and many others. However, discrete variables with a finite number of values can also be used (e.g., number of children in a family, number of tests in an experiment, etc.). The terms "categorical" and "discrete" are usually used indiscriminately in the BN literature. For simplicity, we are going to refer to both types of variables as categorical variables. The natural choice for modelling categorical variables is the categorical distribution. The categorical distribution is a generalization of the Bernoulli distribution that allows more than two possible values. This distribution provides several advantages to categorical BNs. First, all of the CPDs are categorical distributions, which simplifies computations. Second, each CPD can be represented in tabular form, where every assignment of a variable and its parents has a designated probability. Third, they are easily interpretable due to a direct representation of the BN parameters as probabilities. All these benefits make categorical BNs the most common parametrization in the literature [Darwiche, 2009].

As an example, consider the CPD $P(X_4|X_1, X_2)$ of Figure 2.1. In this example, X_4 takes two possible values $\{t, f\}$, and their respective probabilities depend on the assignment of parent variables X_1 and X_2. One problem, however, arises from using this representation: the number of parameters in a categorical CPD increases exponentially with respect to the number of parents, which limits the complexity of the model.

2.2.3.2 Continuous Bayesian networks

Continuous BNs are strictly composed of continuous variables (i.e., quantitative variables with an uncountable number of values). Examples include variables such as age, weight, blood pressure and many others. In this section, we describe how continuous variables can be integrated into the BN framework. We first describe the discretization approach, which simply transforms continuous variables into categorical variables. Then, we present the linear Gaussian BN, which is composed of linear Gaussian CPDs and it is the most commonly used approach. Finally, we comment several semiparametric and nonparametric approaches that are not based on the Gaussian distribution.

Discretization

Given the desirable properties of categorical BNs, it is natural to bin continuous variables into a finite set of intervals. Two types of discretization techniques can be distinguished [Fu, 2005]:

- Prediscretization methods, which discretize the data prior to the definition or learning of the BN. Due to the pertinence of these methods to most machine learning algorithms, a great deal of research has focused on this area [Liu et al., 2002].

- Integrated methods, which employ a greedy iterative search that alternates between BN structure learning and discretization. They are more computationally expensive than prediscretization methods, but they have also shown better results at modelling the data [Friedman and Goldszmidt, 1996; Monti, 1999].

Discretization is still an active research topic and different strategies can be applied to different types of data [Nojavan et al., 2017; Beuzen et al., 2018]. The main drawback of discretizing continuous variables is that it unavoidably results in loss of information. All variation within the created intervals is discarded. As a consequence, certain dependences between variables may become unnoticeable [Friedman et al., 2000].

Linear Gaussian Bayesian networks

The most commonly used parametric form for continuous density functions is the Gaussian distribution. The Gaussian distribution is a member of the exponential family [Wainwright and Jordan, 2008] that makes very strong assumptions, such as the exponential decay of the distribution away from its mean, and the linearity of interactions between its variables. While

these assumptions are often invalid, the Gaussian distribution has proven to be a surprisingly good approximation for many real-world distributions [Kotz et al., 2004].

In a linear Gaussian BN [Shachter and Kenley, 1989], all of its variables are continuous and all of its CPDs are linear Gaussians. Let X be a continuous variable with continuous parents $\mathbf{C} = \{C_1, \ldots, C_k\}$. We say that X has a linear Gaussian CPD if there are parameters $\beta_0, \ldots, \beta_k$ and σ^2 such that

$$P(X|\mathbf{C}) = \mathcal{N}(\beta_0 + \sum_{i=1}^{k} \beta_k C_i; \sigma^2) \ .$$

From this formulation we can see that X follows a Gaussian distribution with a mean that is linear in the values of its parent variables $\mathbf{C}$ and with a variance σ^2. An important result of this formulation is that linear Gaussian BNs are an alternative representation for the class of multivariate Gaussian distributions and vice versa. As an example, consider the conditional distribution $P(X_5|X_3)$ of Figure 2.1. In this example, X_5 has a continuous parent, X_3, and follows a linear Gaussian distribution with parameters $\beta_0 = 2.0$, $\beta_1 = 10.5$ and $\sigma^2 = 1.25$. Linear Gaussian BNs are widely used in the continuous domain due to their support of efficient inference [Koller and Friedman, 2009] and learning [Geiger and Heckerman, 1994].

Other approaches

Some researchers have investigated the use of richer (semiparametric or nonparametric) models for representing nonlinear dependencies in BNs with continuous variables. These include kernel estimators [Hofmann and Tresp, 1995], neural networks [Monti and Cooper, 1997; Choi and Darwiche, 2018], and Gaussian processes [Friedman and Nachman, 2000]. While these approaches may result in a better representation of the underlying probability distribution, they have several disadvantages: (i) they are harder to interpret due to their use of models that are opaque to humans, and (ii) they do not usually allow efficient inference or learning.

2.2.3.3 Hybrid Bayesian networks

Hybrid BNs are composed of both categorical and continuous variables. So far we have discussed the relations between variables of the same type, categorical or continuous. We have shown that in the case of categorical variables, the categorical distribution is the natural CPD. In addition, for continuous variables, we have shown the advantages of the linear Gaussian CPD. We can extrapolate this knowledge to hybrid BNs when both the parents and the children are categorical or continuous. However, there are two new types of relations that we need to address: (i) a continuous variable with continuous and categorical parents, and (ii) a categorical variable with continuous and categorical parents.

A continuous variable with continuous and categorical parents

The simplest way of making a continuous variable depend on a categorical variable is to define a different set of parameters for every value of the categorical parent. While there is no restriction about the parametric form of the continuous variable, we have previously shown the advantages of the linear Gaussian distribution. If we restrict our attention to linear Gaussians, we get a class of hybrid BNs called the conditional linear Gaussian BN [Lauritzen and Wermuth, 1989], where every categorical variable has only categorical parents and every continuous variable has a conditional linear Gaussian CPD. Let X be a continuous variable, where $\mathbf{C} = \{C_1, \ldots, C_k\}$ are its continuous parents and $\mathbf{D} = \{D_1, \ldots, D_l\}$ are its categorical parents. We say that X has a conditional linear Gaussian CPD if for every $\mathbf{d} \in \Omega_{\mathbf{D}}$ (where $\Omega_{\mathbf{D}}$ represents all the joint assignments to $\mathbf{D}$) there are parameters $\beta_{\mathbf{d},0}, \ldots, \beta_{\mathbf{d},k}$ and $\sigma^2_{\mathbf{d}}$, such that

$$P(X|\mathbf{C}, \mathbf{d}) = \mathcal{N}(\beta_{\mathbf{d},0} + \sum_{i=1}^{k} \beta_{\mathbf{d},i} C_i; \sigma^2_{\mathbf{d}}) \ .$$

This CPD can be represented in tabular form, where every assignment of the categorical parent variables has an associated linear Gaussian. As an example, consider the conditional distribution $P(X_3|X_1)$ of Figure 2.1. In this example, X_3 has categorical parent (i.e., X_1) and follows a conditional linear Gaussian distribution with two sets of parameters. When $X_1 = t$, then X_3 follows $\mathcal{N}(3.5; 0.75)$, and when $X_1 = f$, then X_3 follows $\mathcal{N}(0.5; 0.05)$. In addition, we can see that a linear Gaussian distribution is simply a conditional linear Gaussian distribution with no categorical parents and a single set of parameters. This is the case of $P(X_5|X_3)$. It follows that the JPD represented by a conditional linear Gaussian BN is a mixture of Gaussians where every mixture component corresponds to an instantiation of the categorical variables.

A categorical variable with continuous and categorical parents

Note that the definition of conditional linear Gaussian does not allow for categorical variables to have continuous parents. In order to overcome this restriction, Koller et al. [1999] proposed to model the CPDs of categorical variables with continuous parents using the soft-max function. This model is usually referred to as the augmented conditional linear Gaussian BN. Unfortunately there is no exact inference for this type of BN [Lerner and Parr, 2001], but we can always resort to the use of approximate inference [Murphy, 1999].

A completely different approach to the hybrid BN representation problem considers a family of relateds models that include mixtures of truncated exponentials (MTEs) [Moral et al., 2001], mixtures of polynomials (MoPs) [Shenoy and West, 2011], and mixtures of truncated basis functions (MoTBFs) [Langseth et al., 2012]. A MoTBF can be seen as a generalized form of discretization, but instead of using a constant as approximation within each interval, we use a linear combination of basis functions. MTEs and MoPs are particular scenarios of MoTBFs when using exponential or polynomial functions, respectively, as basis functions. These models are interesting because they do not impose any restrictions on

the interactions between variables (discrete nodes with continuous parents are allowed), and because they allow exact probabilistic inference. However, their learning capabilities are currently limited, requiring the structure of a hybrid BN to be supplied in order to learn the MoTBF model [López-Cruz et al., 2013, 2014; Langseth et al., 2014].

2.3 Inference

Probabilistic inference is one of the most fundamental mechanisms for reasoning under uncertainty. Given a BN $\mathcal{B}$ over a set of random variables $\mathbf{X}$, probabilistic inference allows us to answer general queries of the form $P(\mathbf{i}|\mathbf{e})$ where $\mathbf{e}$ are the values of the evidence variables $\mathbf{E} \subset \mathbf{X}$ and $\mathbf{i}$ are the values of the variables of interest $\mathbf{I} \subseteq \{\mathbf{X} \setminus \mathbf{E}\}$, whose values we do not know. For example, in the medical domain, we can query the probability of a certain disease given the observed symptoms. The goal of inference can be formulated as follows:

$$P(\mathbf{i}|\mathbf{e}) = \frac{P(\mathbf{i}, \mathbf{e})}{P(\mathbf{e})} \ .$$

The general problem of probabilistic inference in BNs was first tackled by Kim and Pearl [1983]. On a very high level, inference algorithms can be divided into two main classes: exact inference methods and approximate inference methods. We address each of these classes in the following sections. For additional information on this topic, Salmeron et al. [2018] provide a recent review of the literature.

2.3.1 Exact inference methods

Exact inference algorithms are designed to give an exact answer to the probabilistic query. While exact methods were originally designed for categorical BNs, they can be easily adapted for linear Gaussian BNs using the canonical form representation [Lauritzen, 1992]. There are three main strategies of exact inference in BNs:

- Variable elimination. As the name suggests, the idea behind variable elimination is to successively remove variables from a BN while maintaining its ability to answer the query of interest. It was first formalized by Zhang and Poole [1994], although its origins go back to Shachter et al. [1990].

- Jointrees. The jointree algorithm is a variation on the variable elimination algorithm that can be understood in terms of factor elimination. This algorithm improves on the complexity of variable elimination when answering multiple queries. There are two main approaches to the junction tree algorithm: (i) the Shenoy-Shafer architecture [Shenoy and Shafer, 1990], and (ii) the Hugin architecture [Jensen et al., 1990]. Both of them are based on the work of Lauritzen and Spiegelhalter [1988], which introduced the first jointree algorithm.

- Conditioning. The first incarnation of the conditioning algorithm was presented by Pearl [1986] in the context of cutset conditioning, where the conditioning variables cut

all loops in the network, forming a polytree[1]. The general algorithm, under the name of global conditioning, was presented by Shachter et al. [1994], which demonstrated the relation between conditioning and variable elimination. Finally recursive conditioning was developed by Darwiche [2001].

Although it is NP-hard to perform exact inference in general BNs [Cooper, 1990], it becomes tractable in BNs with bounded treewidth [Kwisthout et al., 2010]. The notion of treewidth was introduced by Robertson and Seymour [1986] and can be understood as a measure of similarity between a graph and a tree (e.g., trees have a treewidth ≤ 1). However, this result only applies to categorical BNs and linear Gaussian BNs. Inference in conditional linear Gaussian BNs is much harder. Even in simple models like polytrees, exact inference is NP-hard [Lerner and Parr, 2001].

2.3.2 Approximate inference methods

Approximate inference algorithms are designed to give an approximate answer to the probabilistic query, with the understanding that giving the exact probability is not crucial. Unfortunately, identical to the exact case, performing approximate inference has also been demonstrated to be NP-hard in general BNs [Dagum and Luby, 1993; Lerner and Parr, 2001]. Despite these discouraging results, one can try to produce useful approximate algorithms with a computational performance that is in many cases far more manageable than that of exact algorithms. There are two main strategies of approximate inference in BNs:

- Sampling. The idea behind sampling-based algorithms is to randomly pick assignments of the random variables, called samples, and then estimate the JPD of the query. It was first introduced for inference in BNs by Henrion [1986], who proposed a method based on probabilistic logic sampling. Several improvements have been done since then, such as likelihood weighting [Shachter and Peot, 1989] and Markov chain Monte Carlo [Pearl, 1987; Hrycej, 1990; York, 1992].

- Optimization. The idea behind optimization-based algorithms is to construct an approximation to the target distribution, and then optimize a similarity function. This approach is usually referred to as variational inference [Blei et al., 2016]. Methods in this class fall into three main categories: belief propagation [Pearl, 1988; Yedidia et al., 2000], expectation propagation [Minka, 2001] and mean-field variational inference [Saul et al., 1996]. In this dissertation, we focus on the mean-field approach. It is discussed in detail in Section 2.4.3 since it is a key part of the learning framework that is used throughout this dissertation.

[1] A polytree is a DAG whose underlying undirected graph is both connected and acyclic (i.e., a tree).

2.4 Learning

In order to introduce the task of probabilistic inference, we have assumed that the structure and parameters of the BN were already given. There are two possible ways to construct a BN: (i) to manually define the model, usually with the help of an expert [Beinlich et al., 1992; van der Gaag et al., 2002], and (ii) to learn it from a dataset $\mathcal{D}$. In this dissertation we focus on learning BNs from data.

Learning BNs is a very broad topic that depends on the specific goal that we set [Daly et al., 2011]. We may want to learn only the model parameters for a fixed structure, or some or all of the structure of a model. In some cases, we may be interested to produce not a single model but rather a probability distribution over models. In the following sections we address the problem of learning the BN parameters and structure when data is complete. In addition, we also address both of these tasks in the presence of incomplete data and discuss the complications that arise in that scenario.

2.4.1 Parameter estimation

In this section, we present the problem of estimating the parameters of a BN when data is complete. That is, we assume that the network structure $\mathcal{G}$ is fixed and our data $\mathcal{D} = \{\mathbf{x}[1], \ldots, \mathbf{x}[N]\}$ consists of N fully observed instances of the network variables $\mathbf{X}$. As we will see in Section 2.4.3, this is not always the case.

There are two main approaches to fit the parameters of a BN: (i) maximum likelihood estimation, and (ii) Bayesian estimation. We address each of these approaches in the following sections.

2.4.1.1 Maximum likelihood estimation

Maximum likelihood estimation (MLE) is the most common method for parameter estimation in BNs. At its core is the idea that a good model is one that fits the data well. This goodness of fit is measured by the likelihood function, which is the probability that a model with a set of parameters $\boldsymbol{\theta}$ assigns to the data $\mathcal{D}$:

$$L(\boldsymbol{\theta} : \mathcal{D}) = P(\mathcal{D}|\boldsymbol{\theta}) = \prod_{n=1}^{N} P(\mathbf{x}[n]|\boldsymbol{\theta}) \ , \tag{2.2}$$

where $P(\mathbf{x}[n]|\boldsymbol{\theta})$ represents the probability of the nth data instance given the model parameters. In practice, it is often convenient to work with the natural logarithm of the likelihood function, called the log-likelihood (LL):

$$\ell(\boldsymbol{\theta} : \mathcal{D}) = \sum_{n=1}^{N} \log P(\mathbf{x}[n]|\boldsymbol{\theta}) \ .$$

In the MLE approach, we want to find the set of parameters $\hat{\boldsymbol{\theta}}$ that maximizes the LL of

the data:

$$\hat{\boldsymbol{\theta}} = \max_{\boldsymbol{\theta}} \ell(\boldsymbol{\theta} : \mathcal{D}) \ . \tag{2.3}$$

This poses a high-dimensional optimization problem, even for BNs with a low number of variables, since we need to optimize over all the CPDs in the network. Fortunately, we can use the factorization property of Equation (2.1) to write

$$\ell(\boldsymbol{\theta} : \mathcal{D}) = \sum_i \ell_i(\boldsymbol{\theta}_{X_i|\mathbf{Pa}_i^{\mathcal{G}}} : \mathcal{D}) \ , \tag{2.4}$$

where $\boldsymbol{\theta}_{X_i|\mathbf{Pa}_i^{\mathcal{G}}}$ are the parameters that encode the CPD of X_i given its parents $\mathbf{Pa}_i^{\mathcal{G}}$ and

$$\ell_i(\boldsymbol{\theta}_{X_i|\mathbf{Pa}_i^{\mathcal{G}}} : \mathcal{D}) = \sum_{n=1}^{N} \log P(x_i[n]|\mathbf{pa}_i^{\mathcal{G}}[n], \boldsymbol{\theta}_{X_i|\mathbf{Pa}_i^{\mathcal{G}}})$$

is the local LL function of X_i. Identically to the LL function, we can easily induce the local function of the likelihood function, i.e., $L_i(\boldsymbol{\theta}_{X_i|\mathbf{Pa}_i^{\mathcal{G}}} : \mathcal{D})$. The exact form of these functions depends on the form of the CPD, see Koller and Friedman [2009] for a detailed explanation. The optimization problem of Equation (2.3) is decomposed into a summation of independent terms, one for each CPD in the network. We can then combine these individual solutions to get the MLE.

2.4.1.2 Bayesian estimation

By using a point estimate of the parameters, such as the MLE, there is no measure of uncertainty, and no prior knowledge can be incorporated into the learning process. In Bayesian statistics, prior knowledge is introduced via a prior distribution over the parameters, and uncertainty is reflected in its posterior distribution. The posterior distribution encodes updated beliefs once prior knowledge and data have been taken into consideration. For a fixed structure $\mathcal{G}$, the posterior distribution of the parameters $\boldsymbol{\theta}$ given the observed data $\mathcal{D}$ is defined as

$$P(\boldsymbol{\theta}|\mathcal{D}) = \frac{P(\mathcal{D}|\boldsymbol{\theta})P(\boldsymbol{\theta})}{P(\mathcal{D})} \ .$$

The term $P(\boldsymbol{\theta})$ denotes the prior distribution of the parameters, $P(\mathcal{D}|\boldsymbol{\theta})$ is the probability of the data given the set of parameters, which is simply the likelihood function. Finally, $P(\mathcal{D})$ acts as a normalizing constant for the posterior distribution.

In the case of MLE, we have seen in Equation (2.4) that the likelihood function decomposes according to the network structure. This decomposition allows us to individually estimate the parameters $\boldsymbol{\theta}_{X_i|\mathbf{Pa}_i^{\mathcal{G}}}$ of each CPD. For Bayesian estimation, we introduce the assumption of global parameter independence, which leads to a similar decomposition of the prior distribution [Spiegelhalter and Lauritzen, 1990]:

$$P(\boldsymbol{\theta}) = \prod_i P(\boldsymbol{\theta}_{X_i|\mathbf{Pa}_i^{\mathcal{G}}}) \ .$$

The decomposability properties of the likelihood function and the prior distribution produce that the posterior distribution can also be decomposed as a product of local terms:

$$P(\boldsymbol{\theta}|\mathcal{D}) = \frac{1}{P(\mathcal{D})} \prod_i \left[L_i(\boldsymbol{\theta}_{X_i|\mathbf{Pa}_i^{\mathcal{G}}} : \mathcal{D}) P(\boldsymbol{\theta}_{X_i|\mathbf{Pa}_i^{\mathcal{G}}}) \right]$$
$$= \prod_i P(\boldsymbol{\theta}_{X_i|\mathbf{Pa}_i^{\mathcal{G}}}|\mathcal{D}) \ .$$

Finally, we need to address the issue of choosing a convenient prior. The form of the prior depends on the form of the CPD. When estimating the parameters of a categorical CPD, the common choice is to use Dirichlet priors. The Dirichlet distribution has the appealing property of being a conjugate prior to the categorical distribution. That is, the posterior distribution has the same functional form as the prior distribution. For linear Gaussian CPDs, the conjugate Gaussian-inverse-Gamma prior plays a similar role (or in the case that we use the multivariate Gaussian representation [Geiger and Heckerman, 1994], the Gaussian-inverse-Wishart). We can combine the above to learn the CPDs of conditional linear Gaussian BNs. For more information on this topic, see Bishop [2016].

2.4.2 Structure learning

In this section, we consider the problem of learning the structure of a BN when data is complete. We can distinguish three main approaches: (i) scored-based structure learning, (ii) constraint-based structure learning, and (iii) hybrid structure learning. We address each of these approaches in the following sections.

2.4.2.1 Score-based structure learning

Score-based methods approach the learning process from a model selection perspective. These methods define a hypothesis space of potential models, a set of operators to navigate this space, and a scoring function that measures how well the model fits the data. Then, the learning task is to find the highest-scoring BN structure. While this problem has proven to be NP-hard [Chickering, 1996; Chickering et al., 2004], various techniques have been developed to render the structure search tractable. Depending on the nature of their search space, they are usually divided into two main groups:

- Space of orders. These algorithms assume an initial topological order for the variables, and use this prior ordering to reduce the complexity of the search space. Given a BN with m nodes (where there is a maximum number of k parents per node) and an ordering of the BN nodes, the highest-scoring BN structure can be learned in $O(m^k)$ time [Cooper and Herskovits, 1992]. The main disadvantage of order-based search is that, without restrictions, the complexity of finding the true ordering is $O(m!)$. Despite this discouraging result, several algorithms have successfully approached this problem using both greedy search [Teyssier and Koller, 2005; Scanagatta et al., 2017] and metaheuristics [Larrañaga et al., 1996a; Hsu et al., 2002; Faulkner, 2007].

Algorithm 2.1: Hill-climbing (HC)

Input : $\mathcal{D}$, $\mathcal{G}_0$, $\mathcal{R}_A$

```
1  G ← G_0
2  while True do
3      G_AI ← AI(D, G, R_A)
4      G_AE ← AE(D, G, R_A)
5      G_AR ← AR(D, G, R_A)
6      G' ← highest scoring structure in {G_AI, G_AE, G_AR}
7      if score(G' : D) > score(G : D) then
8          G ← G'
9      else
10         break /* Stop the loop */
```

Output: The resulting BN structure $\mathcal{G}$

- Space of structures. These algorithms start with some initial structure $\mathcal{G}_0$, usually an empty graph, which automatically becomes the currently best structure $\mathcal{G}$. Then, they get all the neighbor structures of $\mathcal{G}$ by applying local structure modification operators such as arc introduction (AI), arc elimination (AE), and arc reversal (AR). Certain arc restrictions $\mathcal{R}_A$ may limit the structures produced by these operators. Finally, $\mathcal{G}$ is replaced with the highest-scoring structure. This process is repeated until there are no changes in the structure that can improve the score. One of the initial works of this approach is the hill-climbing (HC) method proposed by Chickering et al. [1995], which is depicted in Algorithm 2.1. Several researchers have presented variants of this work that seek to make the search process faster and more accurate. These include techniques based on reducing the search space [Hwang et al., 2002], branch and bound [Suzuki, 1999, 2018], and metaheuristics, usually with a different representation such as a connectivity matrix [Larrañaga et al., 1996b; Wong et al., 1999; Blanco et al., 2003; Kim et al., 2013].

Evaluating a structure from the search space requires computing its score. As expected, one of the most important decisions in this type of learning is the choice of the scoring function. There are two main groups of scores: (i) those based on the likelihood function, and (ii) those based on the marginal likelihood. We address each of these groups of scores in the following sections. For more information, see Carvalho [2009].

Likelihood scores

A natural choice for the scoring function is the LL, which tries to find a model that would make the data as probable as possible. However, the LL tends to favour complete BN structures and it does not provide a useful representation of the independence assumptions of the learned BN. This lack of generalization (over-fitting) is usually avoided by using a

penalized version of the LL score:

$$\text{score}_L(\mathcal{G} : \mathcal{D}) = \ell(\hat{\boldsymbol{\theta}}_{\mathcal{G}} : \mathcal{D}) - \text{pen}(\mathcal{G}, \mathcal{D}) \ , \tag{2.5}$$

where $\ell(\hat{\boldsymbol{\theta}}_{\mathcal{G}} : \mathcal{D})$ is the log-likelihood function, $\hat{\boldsymbol{\theta}}_{\mathcal{G}}$ are the MLE parameters of the considered structure $\mathcal{G}$, and $\text{pen}(\mathcal{G}, \mathcal{D})$ is a penalty function that can depend on $\mathcal{G}$ and $\mathcal{D}$. In the Akaike information criterion (AIC) [Akaike, 1974], $\text{pen}(\mathcal{G}, \mathcal{D}) = \dim(\mathcal{G})$, where $\dim(\mathcal{G})$ is the model dimension, or the number of independent parameters in $\mathcal{G}$. The Bayesian infromation criterion (BIC) [Schwarz, 1978], extends this penalization by including the length of $\mathcal{D}$:

$$BIC(\mathcal{G} : \mathcal{D}) = \ell(\hat{\boldsymbol{\theta}}_{\mathcal{G}} : \mathcal{D}) - \frac{\dim(\mathcal{G})}{2} \log(N) \ ,$$

where N is the number of instances in $\mathcal{D}$. Note that the BIC coincides with the minimum description length (MDL) [Lam and Bacchus, 1994] when scoring BN structures.

The decomposability property of the LL allows an efficient computation of these scores. With a decomposable score, a local change in the structure (such as adding an arc) does not change the score of other parts of the structure that were unaffected.

Bayesian scores

In the Bayesian learning problem, we assume that the learner has a prior distribution $P(\mathcal{G})$ over the set of possible structures. In addition, we assume that, once a structure is considered, the learner has a prior distribution $P(\boldsymbol{\theta}_{\mathcal{G}}|\mathcal{G})$ over its set of parameters $\boldsymbol{\theta}_{\mathcal{G}}$. By Bayes rule, we have that

$$P(\mathcal{G}|\mathcal{D}) = \frac{P(\mathcal{D}|\mathcal{G})P(\mathcal{G})}{P(\mathcal{D})} \ ,$$

where $P(\mathcal{D})$ acts as a normalizing factor that does not help distinguish between different structures. Thus, we define the Bayesian score as:

$$\text{score}_B(\mathcal{G} : \mathcal{D}) = \log P(\mathcal{D}|\mathcal{G}) + \log P(\mathcal{G}) \ .$$

Although the structure prior $P(\mathcal{G})$ gives us a way of preferring some structures over others, it does not play an important role in the asymptotic analysis of the Bayesian score because it is not related to the data. For this reason, we often assume a uniform prior over the structures and ignore this term of the score. The term $P(\mathcal{D}|\mathcal{G})$ is the marginal likelihood of the data given the structure, since it marginalizes out the unknown parameters:

$$P(\mathcal{D}|\mathcal{G}) = \int P(\mathcal{D}|\boldsymbol{\theta}_{\mathcal{G}}, \mathcal{G}) P(\boldsymbol{\theta}_{\mathcal{G}}|\mathcal{G}) d\boldsymbol{\theta}_{\mathcal{G}} \ ,$$

where $P(\mathcal{D}|\boldsymbol{\theta}_{\mathcal{G}}, \mathcal{G})$ is the likelihood of the data given a structure $\mathcal{G}$ with parameters $\boldsymbol{\theta}_{\mathcal{G}}$, and $P(\boldsymbol{\theta}_{\mathcal{G}}|\mathcal{G})$ is our prior distribution over the set of parameters for this structure. The integration over all possible sets of parameters protects us from over-fitting: models with more parameters do not necessarily have higher marginal likelihood. This is called the Bayesian Occam's razor

effect [MacKay, 1991].

The form of the prior distribution over the set of parameters determines the form and properties of the Bayesian score. If the prior $P(\boldsymbol{\theta}_{\mathcal{G}}|\mathcal{G})$ satisfies both the global parameter independence and the parameter modularity assumptions, then the Bayesian score is decomposable. Bayesian scores that are decomposable include the K2 [Cooper and Herskovits, 1992] and BDe [Heckerman et al., 1995] scores for categorical BNs, the BGe for linear Gaussian BNs [Geiger and Heckerman, 1994], and the combination of BDe and BGe for conditional linear Gaussian BNs [Heckerman and Geiger, 1995].

2.4.2.2 Constraint-based structure learning

Constraint-based methods approach the learning process from a statistical independence point of view. These methods perform conditional independence tests and select the equivalence class of BNs that best explains them. An equivalence class of BNs is defined by all the BN structures that represent the same JPD. Some of the key algorithms of the constraint-based approach are: inductive-causation [Verma and Pearl, 1990], PC [Spirtes et al., 2000], grow-shrink [Margaritis, 2003], and incremental association Markov blanket [Tsamardinos et al., 2003].

All constraint-based methods share a common two-phase structure inherited from the inductive-causation algorithm. First, they learn the skeleton of the DAG by checking through conditional independence tests if there is a set of variables that separates a particular pair of variables. If that set is empty, then there must be an arc. Second, they try to assign directions to the arcs as in Meek [1995]. The common limitation of these methods is their sensitiveness to failures in the conditional independence tests. A single mistake in one of these tests suffices to mislead the learning process. Another drawback is the amount of data required by these algorithms. The size of the data hugely increases with respect to the number of conditional variables in the independence test.

2.4.2.3 Hybrid structure learning

Both score-based methods and constraint-based methods have their advantages. For example, score-based methods usually perform better with less data and constraint-based methods are usually faster. For this reason, researchers have tried to combine the advantages of both approaches in the generation of hybrid methods. Some representative hybrid methods include the works of Singh and Valtorta [1993, 1995], Dash and Druzdzel [1999], de Campos et al. [2003], and Tsamardinos et al. [2006].

2.4.3 Learning with incomplete data

The assumption of complete data is often unrealistic. In some cases, certain data values are missing by accident (e.g., they could have been omitted in the collection process). In other cases, their absence is intentional (e.g., in a medical setting, the presence or absence of a biopsy result is clearly not random and it is related to the result of a preliminary blood test).

In addition, data may be also incomplete because some variables may be hidden or latent (e.g., in a clustering problem, cluster assignments are not observed; in factor analysis, factors also not observed). The distinction between these cases has been thoroughly studied in the literature, see Little and Rubin [1987] for more information on this topic.

In this section we consider the problem of learning when missing values and hidden variables are present. We first address the parameter estimation problem and then discuss the even more challenging structure learning problem. We first need to expand the notation previously introduced in Section 2.4.1. For an incomplete dataset $\mathcal{D}$ with N instances, we denote the missing (or latent) variables and their possible assignments in the nth instance by $\mathbf{H}[n]$ and $\mathbf{h}[n]$, respectively. In addition, we use $\mathcal{H} = \cup_n \mathbf{h}[n]$ to denote the set of all possible assignments to all unobserved values in the dataset. Thus, the pair $(\mathcal{D}, \mathcal{H})$ defines an assignment to all variables in all data instances.

2.4.3.1 Parameter estimation

Most of the key properties that helped make parameter estimation feasible with complete data vanish in the incomplete data setting. The learning task requires to globally optimize over a high-dimensional space, with an objective that is highly susceptible to local optima. As a consequence, we need to adjust the MLE and Bayesian estimation approaches.

Maximum likelihood estimation

In the presence of incomplete data, the likelihood function of Equation (2.2) has to consider an exponential number of assignments to the missing values in the dataset:

$$L(\boldsymbol{\theta} : \mathcal{D}) = P(\mathcal{D}|\boldsymbol{\theta}) = \sum_{\mathcal{H}} P(\mathcal{D}, \mathcal{H}|\boldsymbol{\theta}) \ .$$

As a result, although each of the terms $P(\mathcal{D}, \mathcal{H}|\boldsymbol{\theta})$ is a unimodal distribution, the sum can have, in the worst case, an exponential number of modes. However, unimodality is not the only property that we lose. In the presence of incomplete data, the likelihood function can be written as

$$L(\boldsymbol{\theta} : \mathcal{D}) = \prod_{n=1}^{N} P(\mathbf{x}[n]|\boldsymbol{\theta}) = \prod_{n=1}^{N} \sum_{\mathbf{h}[n]} P(\mathbf{o}[n], \mathbf{h}[n]|\boldsymbol{\theta}) \ , \tag{2.6}$$

where $\mathbf{o}[n]$ refers to the observed values in the nth data instance. Equation (2.6) shows that, to compute the likelihood function, we need to perform inference for each instance. As we discussed in Section 2.3, this problem can be intractable. Thus, even the task of evaluating the likelihood function becomes a difficult computational problem.

Therefore, in the presence of incomplete data, we lose all of the important properties of the likelihood function: its unimodality, its closed-form representation, and its decomposition as a product of local likelihoods. Without these properties, the parameter estimation task becomes a substantially more complex optimization problem.

The most straightforward approach to this optimization is to apply a gradient ascent procedure. Binder et al. [1997] derive the gradient form for BNs and show that it can be efficiently computed by applying probabilistic inference. Unfortunately, gradient ascent procedures are guaranteed to achieve only a local maximum of the likelihood function. In order to increase our chances of finding a global maximum, or at least a better local maximum, we have to consider multiple starting points or apply random perturbations.

An alternative algorithm for optimizing the likelihood function is the EM algorithm [Dempster et al., 1977; McLachlan and Krishnan, 2008]. Each iteration t of this algorithm is divided into two steps: (i) the expectation step, in which the estimated parameters $\hat{\boldsymbol{\theta}}_t$ are used to infer a posterior distribution $P(\mathcal{H}|\mathcal{D}, \hat{\boldsymbol{\theta}}_t)$ of the missing values $\mathcal{H}$ given the observations $\mathcal{D}$, and (ii) the maximization step, in which a new point estimate of the parameters $\hat{\boldsymbol{\theta}}_{t+1}$ is computed. It can be shown that the EM algorithm is guaranteed to monotonically improve the likelihood of the observed data until convergence to a (typically) local maximum. In order to diminish the problem of local maxima and improve the performance of the EM algorithm, we can run EM from multiple starting points or apply the pyramid scheme proposed by Chickering and Heckerman [1997].

Bayesian estimation

Since the posterior distribution is a product of the prior distribution and the likelihood function, it follows that the useful properties shown by the posterior distribution when data is complete are lost in the incomplete data setting. The posterior $P(\mathcal{H}, \boldsymbol{\theta}|\mathcal{D})$ becomes a highly complex and multimodal distribution that can no longer be represented as a product of local posterior distributions. In order to estimate this complex distribution, we need to resort to approximate inference methods. In theory, we can apply any approximate inference procedure to this problem. Thus, all the methods we discussed in Section 2.3.2 can potentially be used for performing Bayesian estimation with incomplete data. However, due to its speed and simplicity, we have chosen the mean-field variational inference method. When applied to the Bayesian learning problem, it is usually referred to as the VB approach [Attias, 2000].

The goal of variational inference is to find an approximate distribution $Q(\mathcal{H}, \boldsymbol{\theta})$ from some tractable family $\mathcal{Q}$ that closely approximates the true posterior distribution $P(\mathcal{H}, \boldsymbol{\theta}|\mathcal{D})$. For simplicity, we denote these distributions Q and P, respectively. The key principle of variational inference is to solve this problem via optimization, in which a set of variational parameters $\boldsymbol{\varphi}$ that makes Q closest to P is identified. The usual cost function for this optimization problem is the reverse Kullback-Leibler (KL) divergence:

$$\begin{aligned} \mathrm{KL}(Q||P) &= \int\int Q(\mathcal{H}, \boldsymbol{\theta}) \log \frac{Q(\mathcal{H}, \boldsymbol{\theta})}{P(\mathcal{H}, \boldsymbol{\theta}|\mathcal{D})} d\mathcal{H} d\boldsymbol{\theta} \\ &= \mathbb{E}_Q \left[\log \frac{Q(\mathcal{H}, \boldsymbol{\theta})}{P(\mathcal{H}, \boldsymbol{\theta}|\mathcal{D})} \right] \\ &= \mathbb{E}_Q \left[\log Q(\mathcal{H}, \boldsymbol{\theta}) \right] - \mathbb{E}_Q \left[\log P(\mathcal{H}, \boldsymbol{\theta}, \mathcal{D}) \right] + \log p(\mathcal{D}) \ . \end{aligned} \tag{2.7}$$

However, we cannot minimize this function because it requires computing the intractable $\log p(\mathcal{D})$. Instead, we can maximize an alternative function that is equivalent to the reverse KL divergence up to this constant (the marginal likelihood). This function is called the lower bound of the marginal likelihood or the evidence lower bound (ELBO):

$$\text{ELBO}(Q : \mathcal{D}) = \mathbb{E}_Q \left[\log P(\mathcal{H}, \boldsymbol{\theta}, \mathcal{D})\right] - \mathbb{E}_Q \left[\log Q(\mathcal{H}, \boldsymbol{\theta})\right] \ . \tag{2.8}$$

We can infer two things from Equations (2.7) and (2.8). First, we can see that maximizing the ELBO is equivalent to minimizing $\text{KL}(Q||P)$. Second, we can see that, as its name suggests, the ELBO is a lower bound of the marginal likelihood (usually called the evidence in the Bayesian literature), which follows from the derivation through Jensen's inequality and the fact that $\text{KL}(\cdot) \geq 0$ [Jordan et al., 1999]. The complexity of maximizing the ELBO is determined by the complexity of the variational family $\mathcal{Q}$. In this dissertation, we use the VB framework, which assumes a factorization of the variational posterior that is based on the mean-field approximation:

$$Q(\mathcal{H}, \boldsymbol{\theta}) = \prod_{n=1}^{N} Q(\mathbf{h}[n]) Q(\boldsymbol{\theta}) \ .$$

The VB framework iteratively maximizes the ELBO with respect to $Q(\mathcal{H})$ and $Q(\boldsymbol{\theta})$. This results in an iterative algorithm that is directly analogous to the EM (called the VB-EM algorithm), which is guaranteed to monotonically increase the ELBO. Its expectation and maximization steps have the following form:

- **Expectation**. Update the variational posterior distribution of the unobserved values:

$$Q_{t+1}(\mathcal{H}) \propto \exp\left[\int Q_t(\boldsymbol{\theta}) \log P(\mathcal{D}, \mathcal{H}|\boldsymbol{\theta}) d\boldsymbol{\theta}\right] \ ,$$

- **Maximization**. Update the variational posterior distribution of the parameters:

$$Q_{t+1}(\boldsymbol{\theta}) \propto \exp\left[\int Q_{t+1}(\mathcal{H}) \log P(\mathcal{D}, \mathcal{H}|\boldsymbol{\theta}) d\mathcal{H}\right] P(\boldsymbol{\theta}) \ .$$

The exact forms of the variational expectation and maximization equations depend on the functional forms of the CPDs in the model (e.g. for categorical BNs, see [Beal and Ghahramani, 2006]). However, deriving a set of specific update equations for each type of conditional distribution is an arduous task. Fortunately, the variational message passing (VMP) framework [Winn and Bishop, 2005] provides a set of general purpose update equations that work for any BN for which all parent distributions are conjugate. A model in which both of these constraints hold is known as a conjugate-exponential (CE) model. In this dissertation, we combine the VMP framework with the VB-EM algorithm to estimate the parameters of a conditional linear Gaussian BN when data is incomplete. Fortunately, a conditional linear Gaussian BN is a CE model. Finally, even though we focus on the VB framework, we can

Algorithm 2.2: Structural EM (SEM)

Input : $\mathcal{D}$, $\mathcal{B}_0$, $\mathcal{R}_A$

for $t = 0, 1, \ldots$ *until convergence* **do**

`/* Expectation step */`

$\mathcal{D}_t^* \leftarrow$ Complete data using exact inference with $\mathcal{B}_t$

`/* Maximization step */`

$\mathcal{G}_{t+1} \leftarrow$ Structure-Learn$(\mathcal{D}_t^*, \mathcal{R}_A)$

$\boldsymbol{\theta}_{t+1} \leftarrow$ MLE$(\mathcal{D}_t^*, \mathcal{G}_{t+1})$

$\mathcal{B}_{t+1} \leftarrow \{\mathcal{G}_{t+1}, \boldsymbol{\theta}_{t+1}\}$

Output: The resulting model $\mathcal{B}_t$

add more flexibility to the parameter estimation process by using more flexible variational families [Bishop et al., 1998; Barber and Wiegerinck, 1999] and nonconjugate priors [Wang and Blei, 2013], but at the cost of a more difficult optimization problem.

2.4.3.2 Structure learning

We can distinguish two possible scenarios when learning a BN from incomplete data: (i) data is partially observed, i.e., no variable in the data has all its values missing, and (ii) there are latent variables in the data, i.e., one or many variables in the data have all its values missing. We address each of these scenarios in the following sections.

Structure learning with partially observed data

We need data to be complete in order to apply any of the approaches that we discussed in Section 2.4.2. Constraint-based methods require data completeness to compute conditional independence tests, and score-based methods require data completeness for its scores to be decomposable. A simple approach is to exclude those instances with missing values. However, estimates obtained from this approach may be biased if the excluded instances are systematically different from those included. Inverse probability weighting (IPW) [Horvitz and Thompson, 1952; Robins et al., 1994] is one of several methods that can reduce this bias. In IPW, complete instances are weighted by the inverse of their probability of being a complete instance. This idea has been applied to the domain of constraint-based structure learning. Gain and Shpitser [2018] propose a variant of the PC algorithm that utilizes IPW for each conditional independence test.

Friedman [1997] proposes a completely different approach by generalizing the EM algorithm to the problem of score-based structure learning. This method, called SEM, is described in Algorithm 2.2. Similar to EM, it also iterates over a pair of steps. At each iteration t, in the expectation step, it uses the current model to generate a complete dataset D_t^*. Then, in the maximization step, it estimates not only the parameters of the new model, but also its structure. Any of the score-based procedures we described in Section 2.4.2.1 can be used for this purpose. However, the scoring function to be maximized must be a penalized version of

the LL, as in Equation (2.5).

It is important to note the benefits of SEM compared to a brute-force approach: rather than re-estimating the model parameters after each structure change, the output of a single expectation step is used to perform many structure changes. At each iteration t, SEM selects the model $\mathcal{B}_{t+1}$ with the highest expected score. The expected score is usually referred to as $\text{score}(\mathcal{D}_t^*, \mathcal{B}_{t+1})$, and its use is motivated by the following inequality:

$$\text{score}(\mathcal{B}_{t+1} : \mathcal{D}_t^*) - \text{score}(\mathcal{B}_t : \mathcal{D}_t^*) \leq \text{score}(\mathcal{B}_{t+1} : \mathcal{D}) - \text{score}(\mathcal{B}_t : \mathcal{D}) \ . \tag{2.9}$$

Equation (2.9) states that an improvement in the observed score (i.e., using the observed data $\mathcal{D}$) of network $\mathcal{B}_{t+1}$, relative to the network $\mathcal{B}_t$ that was used to generate $\mathcal{D}_t^*$, is at least as large as the improvement of the expected score using the completed data $\mathcal{D}_t^*$. This guarantees that SEM converges without the need of using probabilistic inference in each structure change. However, despite the computational savings provided by SEM, it is still a highly demanding algorithm due to the need of applying probabilistic inference in its expectation step. For this reason, works like Scanagatta et al. [2018] and Benjumeda et al. [2019] have proposed adaptations of the SEM algorithm that bound the inference complexity in the expectation step.

Finally, the idea of SEM can also be applied to Bayesian learning. However, in order to compute the expectation step we have to rely either on a maximum a posteriori (MAP) solution or on an approximate Bayesian inference solution. The Bayesian SEM algorithm proposed by Friedman [1998] advocates for the first solution. In this dissertation, we advocate for the second solution, where we combine SEM with the VB framework.

Structure learning with latent variables

Latent variables, as opposed to observed variables, are variables that are not directly measured but rather inferred from the observed variables through the statistical model. When a latent variable is known to exist, we can introduce it into the BN model and apply methods such as the SEM algorithm to perform structure learning with incomplete data. However, we cannot simply place a latent variable arbitrarily in the model and expect our learning procedure to produce a reasonable model. In fact, if a latent variable is placed where it does not improve the model score, there is a good chance it will end up being disconnected from the rest of variables [Koller and Friedman, 2009]. Thus, we need a mechanism to introduce latent variables in approximately the right location in the BN structure.

There are many approaches that can be used to introduce a latent variable. One approach is based on finding structural signatures that the latent variable might leave, such as densely connected variables [Elidan et al., 2000]. Unfortunately, this technique does not perform too well, since structure learning algorithms are usually biased against fitting models with densely connected variables, especially with limited data. As a result, Elidan and Friedman [2005] propose instead to consider information signatures, which are identified using the information bottleneck method [Tishby et al., 1999; Friedman et al., 2001]. A completely

different approach is to use a search method to iteratively explore a space of models. This idea was first proposed by Zhang [2004], which considered the space of latent tree models. Since then, it has been greatly improved with several methods and different strategies. We discuss this approach with more detail in Chapters 3, 5, and 6.

Chapter 3

Latent tree models

3.1 Introduction

In the analysis of real-world data, it is useful to learn a latent variable model (LVM) that faithfully represents the data generation process. Applications of LVMs are numerous and cover many scientific fields. This is typically the case for domains such as economics [Aigner et al., 1984], psychology [Bollen, 2002] and medicine [Rabe-Hesketh and Skrondal, 2008], to cite some examples. Such fields usually need complex constructs that cannot be directly observed. For instance, abstract concepts such as human personality in psychology and social class in socio-economics [Everitt, 1984].

An LVM that has received considerable attention is the latent tree model (LTM). An LTM is a special type of BN with a rooted tree structure. In this BN, leaf nodes are observed variables and internal nodes can be either observed or latent variables. Subclasses of LTMs have been studied for decades. For instance, phylogenetic trees [Durbin et al., 1998] are binary LTMs (each internal node has two children) whose variables are categorical and have all the same cardinality. Another subclass of the LTM is the latent class model (LCM) [Lazarsfeld and Henry, 1968]. In an LCM, all observed variables are conditionally independent given a single latent variable.

LTMs were first identified as a class of potentially useful models by Pearl [1988]. In his work, Pearl noticed the ability of these models to capture complex relationships with a simple structure that could be easily interpretable. Subsequently, Zhang [2004] expanded Pearl's work by doing the first systematic study about LTMs. In his work, Zhang proposed the hierarchical latent class model (HLCM), a generalization of the categorical LCM that allows multiple latent variables. Since then, LTMs have been extensively studied, proving their value in many areas of machine learning, such as density estimation, topic detection, clustering and classification. In this chapter we give a general overview of LTMs and their applications. For more information on this topic, see the works of Mourad et al. [2013], Zhang and Poon [2017] and Zwiernik [2018].

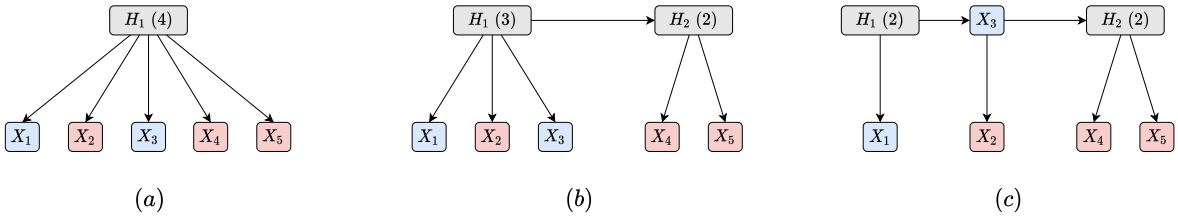


Figure 3.1: Example structures of (a) an LCM, (b) an HLCM and (c) an LTM with observed internal nodes. Categorical observed variables are colored blue while continuous observed variables are colored red. Categorical latent variables are colored grey, where the number between parentheses corresponds to their respective cardinalities.

Chapter outline

In Section 3.2, we start with a formal definition of the LTM. In Section 3.3, we present the learning process of LTMs and address each of the existing approaches. Finally, in Section 3.4, we discuss several applications of LTMs.

3.2 Model definition

An LTM is a tree-structured BN that contains a set of observed (or partially observed) variables $\mathbf{X}$, and a set of latent (fully unobserved) variables $\mathbf{H}$. Following the notations of BNs, we denote an LTM as a pair $\mathcal{B} = (\mathcal{G}, \boldsymbol{\theta})$. The second component $\boldsymbol{\theta}$ is the same as in a BN. It is the collection of parameters that describe the CPD of each variable given its parents in the tree. However, the first element slightly differs from the previous definition. In addition to represent the arcs between the variables in the model, we also consider elements of the structure as the number of latent variables in the model and the cardinality of categorical latent variables.

Identical to BNs, we can distinguish between categorical, continuous and hybrid (i.e., mixed) LTMs. In this dissertation we are especially interested in the hybrid case, more specifically, in the conditional linear Gaussian case, where categorical variables can have categorical and continuous children but cannot have continuous parents. Figure 3.1 shows several example structures of conditional linear Gaussian LTMs. These include: (a) an LCM, (b) an HLCM, and (c) an LTM with observed internal nodes. In this dissertation, we restrict ourselves to the common case of LTMs with categorical latent variables. However, LTMs can also include continuous latent variables (e.g., see Choi et al. [2011]).

3.3 Learning

In order to learn the structure of an LTM, the following information must be determined: (i) the number of latent variables, (ii) the cardinality of categorical latent variables, and (iii) the arcs between variables. Akin to BNs, we distinguish three main structure learning

approaches: score-based learning, constraint-based learning and variable clustering. The first two approaches are very similar to their BN counterparts. However, while the variable clustering approach is a hybrid approach, it combines ideas from score-based and constraint-based learning in a way that is specific to LTMs. We address each of these approaches in the following sections. With respect to the parameter estimation process, the majority of the following methods use the EM algorithm (with few noted exceptions, such as the work of Huang et al. [2020]).

3.3.1 Score-based methods

Score-based methods construct LTMs one local move at a time, such as by introducing a latent variable, removing an arc, or increasing the cardinality of a categorical latent variable. The selection of the best move on each iteration is done with the scoring function. Zhang [2004] was the first to propose an approach of this kind, called double hill-climbing (DHC) algorithm. DHC starts with an LCM and then explores the space of categorical HLCMs using the BIC score. At each step of the search, it employs a hill-climbing procedure to identify the best HLCM that can be produced by introducing a single arc or latent variable. Once this model is found, DHC optimizes the cardinalities of its latent variables by employing a second hill-climbing procedure (hence the name of the algorithm).

Using two hill-climbing procedures results in a very computationally expensive process. For this reason, a less computationally intensive alternative called the single hill-climbing (SHC) algorithm was later proposed by Zhang and Kočka [2004]. In this work, the authors take inspiration on the SEM algorithm to reduce the computational cost of evaluating each operator. In addition, instead of using the BIC score for model selection, they proposed to use an improvement of BIC score per unit of complexity.

The SHC algorithm was later improved by Chen et al. [2012], who proposed to divide each search step into three stages: expansion, adjustment and simplification. Each stage uses its own set of operators (e.g., during the expansion stage, the HLCM is modified by introducing a new arc or a new latent variable). For this reason, this algorithm is called EAST (expansion, adjustment, and simplification until termination).

The aforementioned algorithms were all designed to work with categorical data. An extension of EAST for continuous data was later proposed by Poon et al. [2013]. This algorithm produces HLCMs with categorical latent variables and continuous observed variables. Additionally, Galimberti et al. [2018] proposed a greedy method for learning forests of unconnected LCMs. All of these methods assume that each observed variable in the model follows a Gaussian distribution.

Score-based methods do not have any theoretical guarantees for learning LTMs. However, if they are not trapped by local maxima, the maximized score provides an assurance of the model quality.

3.3.2 Constraint-based methods

Constraint-based methods identify conditional independence relations as constraints when learning BNs from complete data. These constraints are sufficient when variables are fixed. However, variables are not fixed when learning LTMs, and thus other constraints are required. For LTMs, some form of sibling test is usually employed. Siblings refer to variables in the LTM that have the same parent, and a sibling cluster refers to a set of children variables that have the same parent. The sibling test not only allows the connections among variables to be determined, it may also suggest that a latent variable can be introduced as the parent of a sibling cluster.

Phylogenetic tree reconstruction led to the development of constraint-based methods that use information distances between variables to determine sibling clusters [Saitou and Nei, 1987; Durbin et al., 1998]. A nice property of information distances is that they are additive Lake [1994]. That is, the information distance between two nodes in an LTM is given by the sum of the distances along the path that connects them. Due to this property, Choi et al. [2011] proposed the recursive grouping (RG) method, an algorithm that iteratively finds parent-child and sibling relationships among variables. RG is structurally consistent. That is, the LTM structure can be correctly recovered provided that the number of data instances is large enough. In addition, unlike current score-based methods, RG is not limited to learn HLCMs. It can learn LTMs with observed variables as internal nodes. However, RG also has several drawbacks. First, when applied to categorical data, all variables must have the same cardinality. Second, in the presence of continuous data, all variables must be continuous. These restrictions make it impossible to mix continuous and categorical variables. The work of Huang et al. [2020] follows on the idea of RG and circumvents these issues by defining a new distance metric and a new parameter estimation method that is based on moments and tensor decompositions [Anandkumar et al., 2014].

Another popular class of constraint-based methods that is not based on information distances is the family of quartet-based methods. The idea is to first construct a set of quartets for all subsets of four observed variables and, subsequently, combine these quartets to form an LTM. A quartet is simply an unrooted tree with four observed nodes, where each latent node has a minimum of two children. As an example, Figure 3.2 illustrates the four possible quartets of a set of observed variables $\{X_1, X_2, X_3, X_4\}$. However, it is known that determining an LTM that agrees with the maximum number of quartets is NP-hard [Steel, 1992]. For this reason, several heuristics have been proposed [Pearl, 1988; Chen and Zhang, 2006; Anandkumar et al., 2011].

While constraint-based methods usually give strong theoretical guarantees, they rely on two assumptions that may not be realistic. First, they assume that the information distances and quartet tests are accurate. However, this is only true when they are estimated from infinite data. In reality they have to be estimated from a finite set of data instances, which lowers the accuracy of the identified relationships between variables. Second, they assume that data is sampled from an LTM. If the samples are generated from a different model, there is no guarantee on the performance of the models resulting from these algorithms.

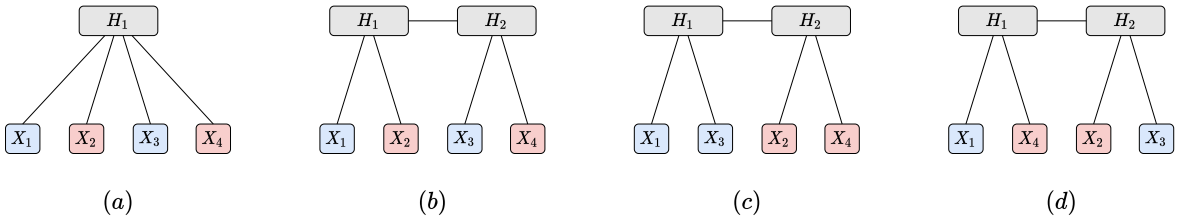


Figure 3.2: The four possible quartets of the set of observed variables $\{X_1, X_2, X_3, X_4\}$. Variable colors and parentheses have the same meaning as in Figure 3.1.

3.3.3 Variable clustering methods

Variable clustering methods arise from the observation that sibling variables in an LTM are usually more similar to each other than to the rest of variables. From this observation, variable clustering methods use correlation or information-theoretic metrics such as the mutual information (MI) [Cover and Thomas, 1991] to cluster similar variables. Once variables have been grouped, a latent variable is added as the parent of each variable cluster. Finally, latent variables are connected so an LTM can be obtained.

The first approach of this kind was proposed by Wang et al. [2008]. In their work, the authors developed an algorithm called HCL (a shorthand for hierarchical clustering learning of LTMs) that constructs a categorical HLCM in the same way as how a dendrogram is built. Specifically, in each step it finds the pair of variables that have the highest MI (for latent variables, it is estimated using the highest MI of the observed children). Once a pair is selected, a new categorical latent variable of predefined cardinality is added as the parent of the pair and takes their place in the next step. This process is repeated until there is only one remaining variable, yielding a binary HLCM. Finally, the algorithm applies a regularization process that may result in a non-binary HLCM.

Harmeling and Williams [2010] proposed two closely related algorithms for learning forests of binary HLCMs, namely BIN-G and BIN-A. Similar to HCL, both algorithms follow a hierarchical process based on MI. However, unlike HCL, both algorithms estimate the cardinality of the resulting categorical latent variables. The main differences between BIN-G and BIN-A lie in the MI and cardinalities estimation. In BIN-G, the MI between a latent variable and other variables is estimated using the completed data. In addition, the cardinality of a new latent variable is locally estimated at the moment of creation. In BIN-A, the MI between latent variables is estimated using the average MI of their respective observed children. In addition, the cardinalities of the latent variables are recursively estimated in a bottom-up fashion once the structure of the model has been defined.

All of the above methods are limited to cluster pairs of variables. To allow a higher number of variables in the cluster, it is necessary to define a criterion for the size of the cluster. Wang [2009] proposed such a criterion. A unidimensionality test that determines whether some variables should belong to the same cluster or different clusters. Given a subset of variables, the test works by learning an LTM with a score-based algorithm such

as EAST. If the resulting LTM has more than one variable, it indicates that the subset of variables belongs to the same cluster. This unidimensionality test was later incorporated into the bridged islands (BI) algorithm, proposed by Liu et al. [2015]. This algorithm first partitions the set of all observed variables into sibling clusters. Then, it creates an LCM with a categorical latent variable for each sibling cluster. After that, it imputes the values of the latent variables and connects them using Chow-Liu's algorithm [Chow and Liu, 1968], resulting in an HLCM.

When the number of variables in the dataset is large (e.g., more than 30 variables), it might be more reasonable to learn a forest rather than a tree because many variables might not be significantly dependent of each other. From the above methods, only BIN-A and BIN-G are able to learn forests of LTMs. However, as we previously commented, they are limited to forests of binary HLCMs. A method named CFHLC, which stands for construction of forests of hierarchical latent class models, was proposed by Mourad et al. [2011] to learn non-binary forests. CFHLC works by grouping similar variables following an information decay criterion [Ben-Dor et al., 1999]. Once groups have been determined, the cardinality of the introduced latent variable is given by its number of children variables.

Methods based on variable clustering do not provide any guarantee on their performance. However, given that their parameters are usually estimated using the EM algorithm, they return a score that, as in score-based methods, provides an assurance of the model quality while usually being much faster.

3.4 Machine learning applications

LTMs have proven to be valuable in many machine learning applications such as density estimation, classification, clustering, and topic detection. We address each of these areas in the following sections. However, they are not the only known applications of LTMs. Other interesting uses include those of computer vision [Kaltwang et al., 2015], spatial analysis [Yu et al., 2016], and deep probabilistic modelling [Chen et al., 2017b].

3.4.1 Density estimation

Density estimation is one of the most common uses of LTMs. The idea is to construct an estimate of the probability distribution underlying the observed data. The problem of density estimation has been long studied in the literature, using both parametric [McLachlan and Peel, 2004] and nonparametric [Scott, 1992] approaches. LTMs offer a parametric perspective that is not only flexible but also interpretable thanks to their graphical representation. They serve as an exploratory tool, where the structure of the model reveals how the observed variables are directly or indirectly (due to latent variables) related to each other.

An example of density estimation with LTMs can be found in Zhang et al. [2008a]. In this work, the authors learn an LTM from the Coil challenge 2000 data [van der Putten and van Someren, 2004] using the SHC algorithm. The Coil data has 42 variables. Three variables show socio-demographic information of the customers, while the others show the ownership of

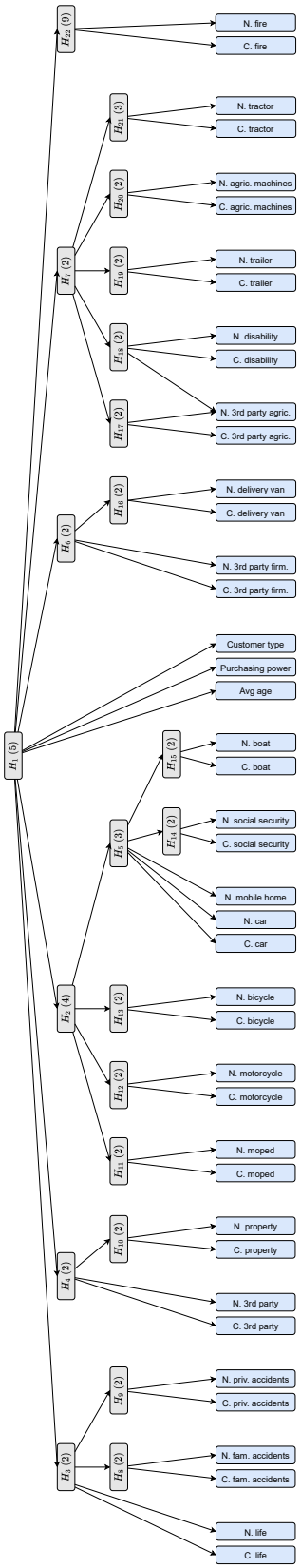


Figure 3.3: LTM structure returned by EAST when applied to the Coil challenge 2000 data. Variable colors and parentheses have the same meaning as in Figure 3.1.

various insurance products by the customers. Figure 3.3 shows the resulting model structure, which groups similar insurance products such as those related to agriculture or to vehicles. Other examples of density estimation with LTMs can be found in the areas of traditional Chinese medicine [Zhang et al., 2008b] and finance [Choi et al., 2011].

Wang et al. [2008] suggest the use of categorical LTMs for approximate inference. This application is closely related to density estimation. Their motivation is that probabilistic inference may be intractable for some complex BNs. For this reason, it would be interesting to learn an LTM with a similar probability distribution that would allow us to efficiently compute inference queries. To carry out this idea, the authors propose to first generate samples from the original BN and then learn an LTM from them. In their original work, this approach shows competitive results compared to other approximate inference approaches such as loopy belif propagation [Murphy et al., 1999]. However, although inference becomes faster, the training time of the LTM should also be considered.

3.4.2 Classification

Classification is a supervised learning problem where a class label is predicted for a given data instance. LTMs also find their use in this domain of application. A straightforward approach is to hierarchically introduce latent variables in a naïve Bayes classifier to relax its conditional independence assumption. The resulting model is almost identical to an LTM, except the class variable is the root node of this tree. Zhang et al. [2004] named this classifier the hierarchical naïve Bayes model. In order to learn it, the authors proposed a generative approach that used the DHC algorithm with the BIC score. Langseth and Nielsen [2009] improved this idea with a discriminative approach which, instead of using the BIC score to guide the search, employed the classification accuracy. In addition, the search algorithm was modified to improve the computational complexity.

Wang et al. [2013] proposed a different approach for using LTMs in classification. A new type of model called the latent tree classifier (LTC) was presented. An LTC can be considered a mixture of LTMs, where the class variable is used as the mixture variable. Specifically, every class label in an LTC has an associated LTM.

In the multi-label setting, Nimmagadda and Anandkumar [2015] took advantage of the hierarchical structure of LTMs to group various object categories in the problem of multi-object classification. In this work, the authors propose to recover the LTM structure using a variant of the RG algorithm, and estimate model parameters using a feed-forward artificial neural network.

3.4.3 Clustering

Clustering is an unsupervised learning problem where data instances are assigned to several groups (called clusters) so that instances belonging to the same group are similar in some sense. In a mixture model [McLachlan and Peel, 2004], the categorical latent variable can be used for clustering. Similarly, categorical latent variables in an LTM can be used for this

purpose and, with multiple latent variables, produce multiple clustering solutions. Zhang [2004] was the first to apply LTMs to this purpose. This idea was further developed by Chen et al. [2012], who coined the term "multidimensional clustering". In an LTM, each categorical latent variable represents a dimension along which data are grouped into clusters.

In addition to multidimensional clustering, LTMs can also be applied to the case of unidimensional clustering, where there is a single clustering variable [Poon et al., 2018]. In this case, the objective of introducing several categorical latent variables is to model local dependence in an LCM so as to improve clustering quality. We discuss both unidimensional and multidimensional clustering in Chapter 4.

3.4.4 Topic detection

Topic detection allows us to automatically extract meaning from texts by identifying recurrent themes or topics. Similar to clustering, it is an unsupervised learning problem. Researchers have proposed many methods for topic detection [Daud et al., 2010; Jelodar et al., 2019]. The most commonly used method is latent Dirichlet allocation (LDA) [Blei et al., 2003]. LDA is a generative probabilistic model of a corpus. The basic idea is that documents are represented as random mixtures over latent topics, where a topic is characterized by a distribution over words.

Liu et al. [2014] proposed the first use of LTMs for topic detection. The key idea is to build a hierarhical model where each latent variable represents a soft partition of the documents and its states can be interpreted as topics. This way, a document can 100% belong to multiple topics. The proposed method for topic detection is therefore called hierarchical latent tree analysis (HLTA). Figure 3.4 shows the model structure that results from applying HLTA to several magazine texts. In this example, we can see that the LTM groups words that are semantically similar. One of these groups is the one governed by the latent variable H_9, which is formed by {*baseball, games, hockey, league, nhl*}. An extension of this algorithm was later proposed by Chen et al. [2017a], which introduced a faster alternative to the EM algorithm called progressive EM.

There are three fundamental differences between LDA-based models and LTMs for topic detection. First, unlike LDA-based models, LTMs do not refer to a document generation process. Latent variables in an LTM are considered unobserved attributes of the documents. Second, observed variables in LDA-based models are token variables, where each token variable stands for a location in a document, and its possible values are the words in a vocabulary. In contrast, each observed variable in an LTM stands for a word and has two values representing the presence/absence of the word in a document. Finally, the third difference lies in the definition and characterization of topics. Topics in LDA-based models are probabilistic distributions over a vocabulary. In contrast, topics in LTMs are probabilistic clusters of documents.

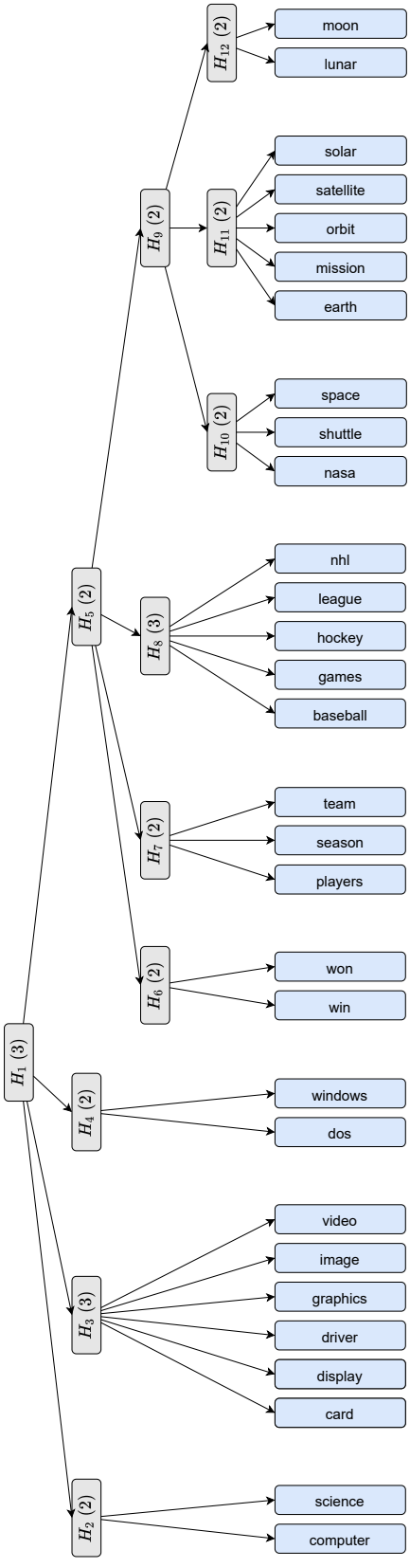


Figure 3.4: LTM structure returned by HLTA when applied to a simple topic detection dataset. Variable colors and parentheses have the same meaning as in Figure 3.1.

Chapter 4

Model-based clustering with Bayesian networks

4.1 Introduction

Clustering is an unsupervised learning problem where data instances are grouped based on similar characteristics. The resulting groups are called clusters. Ideally, data instances in the same cluster are more similar to each other than to those in other clusters. Clustering is also referred to as unsupervised classification because data instances are classified without any class labels given beforehand. Since it does not need any previous labelling, it is a useful technique for exploratory data analysis.

Based on their principle, clustering algorithms can be classified as distance-based or model-based. In distance-based methods, a distance measure is used to determine the similarity or dissimilarity between data instances. Examples of distance-based methods include hierarchical clustering [Gordon, 1987], K-means [MacQueen, 1967], K-modes [Chaturvedi et al., 2001], affinity propagation [Frey and Dueck, 2007], and many others [Xu and Wunsch, 2008]. In contrast, model-based methods [McLachlan and Peel, 2004; Melnykov and Maitra, 2010] assume that data has been originated by a finite mixture model (whose components correspond with the clusters) and try to estimate the probability distribution that underlies each cluster.

Based on their cluster assignments, clustering algorithms can be classified as hard or soft. Hard clustering methods assign each data instance to a single cluster. The majority of clustering algorithms follow this approach. In contrast, soft clustering methods assign each data instance to all the clusters but with different degrees of membership. This approach is mainly followed by fuzzy clustering methods [Höppner et al., 1999] and model-based methods.

Compared to non-probabilistic approaches, model-based clustering offers several advantages. First, it provides a measure of uncertainty in its cluster assignments. Second, it provides a generative model (e.g., a BN), which allows inference and model selection. Finally, it allows to simultaneously define multiple clusterings on different subsets of observed

variables within a single model.

Chapter outline

In Section 4.2, we start with an overview of traditional model-based clustering, also known as unidimensional clustering, where a single clustering variable is considered. Then, in Section 4.3, we expand on this topic by considering multiple clustering variables. This model-based clustering approach is usually referred to as multidimensional clustering.

4.2 Unidimensional clustering

Traditional model-based clustering algorithms assume that data has been generated by a probability distribution $P(\mathbf{X})$ that can be expressed as a finite mixture of k cluster-specific CPDs. In this model, each mixture component $P(\mathbf{X}|H)$ represents the CPD of the observed variables given the clustering variable H, whose probability distribution is referred to as $P(H)$. That is,

$$P(\mathbf{X}) = \sum_{H} P(H)P(\mathbf{X}|H) \ .$$

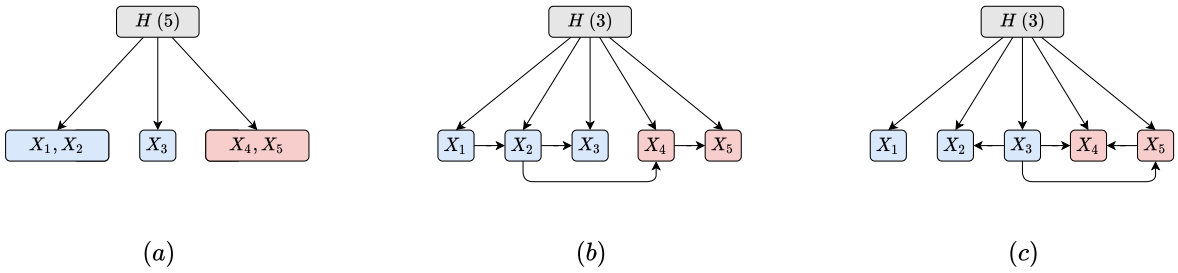
Estimating the parameters of this model is not trivial given the hidden nature of clustering variables (i.e., a clustering variable is simply a categorical latent variable whose cardinality corresponds to the number of clusters). For this reason, its learning process requires a method that is able to handle incomplete data such as the EM algorithm or the VB-EM algorithm. Once the model is learned, we can use it for clustering by computing the posterior probability $P(H|\mathbf{X})$:

$$P(H|\mathbf{X}) \propto P(H) \ P(\mathbf{X}|H) \ . \tag{4.1}$$

However, working with $P(H|\mathbf{X})$ may be cumbersome due to the exponential increase in model parameters with respect to the number of categorical variables. It is therefore necessary to exploit the conditional independences that are present in the data. The application of the BN factorization allows us to simplify Equation (4.1):

$$P(H|\boldsymbol{X}) \propto P(H) \ \prod_{i} P(X_i|\mathbf{Pa}_i^{\mathcal{G}}) \ ,$$

where $\mathbf{Pa}_i^{\mathcal{G}} \subseteq \{H, \mathbf{X}\}$. Hence, by using a BN we need to learn not only the model parameters but also its network structure. Several methods have been developed for clustering with BNs. The majority of them use Bayesian classifier structures such as: (i) the unsupervised naïve Bayes (i.e., the LCM) [Cheeseman et al., 1988; Barash and Friedman, 2002; Santafé et al., 2006a], (ii) the unsupervised semi-naïve Bayes [Peña et al., 1999], (iii) the unsupervised tree-augmented naïve Bayes [Santafé et al., 2006b; Pham and Ruz, 2009], and (iv) the unsupervised k-dependence Bayesian classifier (uk-DB) [Peña et al., 2000; Pham and Ruz, 2009]. Other proposals using less common BN structures consider recursive Bayesian multinets [Peña et al., 2002] and LTMs [Poon et al., 2018]. Figure 4.1 presents some examples of these models.



(a) (b) (c)

Figure 4.1: Examples of (a) an unsupervised semi-naïve Bayes model structure, (b) an unsupervised tree-augmented naïve Bayes model structure and (c) an uk-DB with $k = 2$ model structure. Variable colors and parentheses have the same meaning as in Figure 3.1.

4.3 Multidimensional clustering

A single clustering variable implies the assumption that there exists a single way to partition the data. However, this assumption is hardly true except maybe for very simple datasets. In a real-world dataset, there might be multiple ways to partition the data. Following this idea, multidimensional clustering algorithms assume that data has been generated by a finite mixture model with multiple clustering variables $\mathbf{H} = \{H_1, \ldots, H_s\}$, where each clustering variable H_i uniquely partitions the data with a different number k_i of clusters:

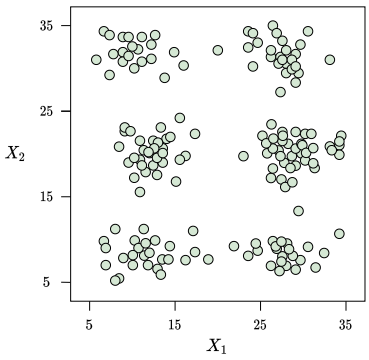
$$P(\mathbf{X}) = \sum_{H_1} \cdots \sum_{H_s} P(\mathbf{H}) P(\mathbf{X}|\mathbf{H}) \ .$$

Suppose we want to cluster the data shown in Figure 4.2 (a), which has two continuous variables, X_1 and X_2. The traditional approach is to learn a unidimensional clustering model such as an LCM (Figure 4.2 (b)) with 6 clusters. However, as shown by Figures 4.2 (e) and (f), this is not the only possible solution. We can obtain a meaningful clustering from either one of the variables. If we consider a model with two clustering variables (Figure 4.2 (d)), we get a 2-cluster solution on X_1 and a 3-cluster solution on X_2. This approach is called multidimensional clustering because multiple clusterings are obtained along different dimensions of data. Each dimension corresponds to a subset of observed variables, which gives meaning to the clustering.

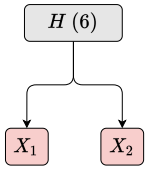
In the example of Figure 4.2, we see one appealing trait of multidimensional clustering: compactness. The bidimensional clustering model of this example is able to represent the data with not only fewer clusters than the unidimensional clustering model (5 versus 6), but also with fewer parameters (18[1] versus 30[2]). Model compactness becomes more appealing as the dimensionality of data increases. When the number of observed variables is large, each cardinality increase of a unidimensional clustering model results in a large increase of model

[1] H_1 and H_2 follow categorical distributions with 2 and 6 parameters, respectively. X_1 and X_2 follow conditional linear Gaussian distributions with 4 and 6 parameters, respectively.

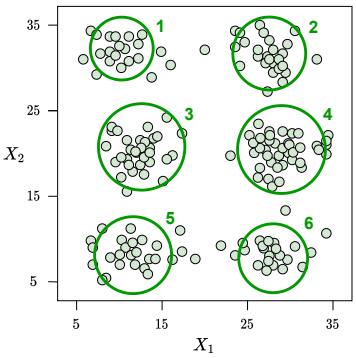
[2] H follows a categorical distribution with 6 parameters. X_1 and X_2 follow conditional linear Gaussian distributions with 12 parameters each.



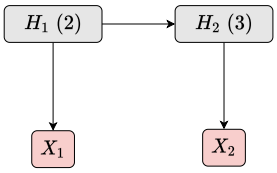
(*a*) Data



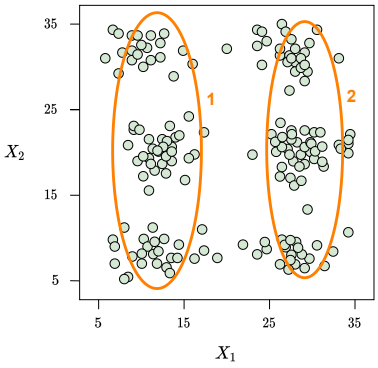
(*b*) Unidimensional model



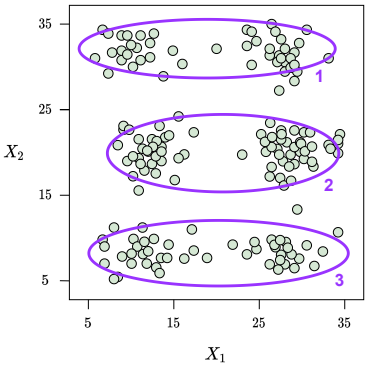
(*c*) Unidimensional clustering on X_1 and X_2



(*d*) Bidimensional model



(*e*) Bidimensional clustering on X_1



(*f*) Bidimensional clustering on X_2

Figure 4.2: Multidimensional clustering with two observed variables. (a) Data. (b) Unidimensional clustering model structure. Variable colors and parentheses have the same meaning as in Figure 3.1. (c) Unidimensional clustering on both X_1 and X_2, resulting in 6 clusters. (d) Bidimensional clustering model structure. (e) Bidimensional clustering on X_1, resulting in 2 clusters. (f) Bidimensional clustering on X_2, resulting in 3 clusters.

parameters. Alternatively, each cardinality increase of a multidimensional clustering model only affects a subset of the model parameters, since each clustering variable is only related to a subset of observed variables.

Identically to the unidimensional case, we are interested in the probability distribution of the clustering variables $\mathbf{H}$ given the observed variables $\mathbf{X}$. When combined with the BN factorization, this probability distribution, $P(\mathbf{H}|\boldsymbol{X})$, can be expressed as follows:

$$P(\mathbf{H}|\boldsymbol{X}) \propto \prod_i P(H_i|\mathbf{Pa}_i^{\mathcal{G}}) \prod_j P(X_j|\mathbf{Pa}_j^{\mathcal{G}}) \ ,$$

where $\mathbf{Pa}_i^{\mathcal{G}} \subseteq \mathbf{H}$ and $\mathbf{Pa}_j^{\mathcal{G}} \subseteq \{\mathbf{H}, \mathbf{X}\}$. Learning an LVM for multidimensional clustering is much more computationally expensive than for unidimensional clustering. In contrast to the unidimensional case, a multidimensional clustering algorithm has to determine: (i) the number of clustering variables, (ii) the number of clusters in each clustering, (iii) the observed variables that are relevant for each clustering, (iv) the relationships between clustering variables, and (v) the relationships between observed variables. For this reason, the majority of multidimensional clustering algorithms have considered restricted models such as LTMs [Zhang, 2004; Chen et al., 2012; Poon et al., 2013; Liu et al., 2015; Li et al., 2019], latent forests [Harmeling and Williams, 2010; Mourad et al., 2011; Galimberti et al., 2018], and DAGs where the observed variables are conditionally independent given the clustering variables [Asbeh and Lerner, 2016a,b].

4.3.1 Related work

In addition to multidimensional clustering, there are other lines of work that produce multiple clusterings. They can be classified according to the way new clusterings are found: sequentially or simultaneously.

4.3.1.1 Sequential clustering algorithms

Sequential clustering algorithms (also referred to as alternative clustering algorithms) retrieve new data partitions that are distinct to the previously generated ones. Bae and Bailey [2006] were the first to propose an algorithm of this kind, named COALA (constrained orthogonal average link algorithm). Given a clustering, COALA applies an agglomerative clustering process in combination with a series of pairwise cannot-link constraints to generate a new alternative clustering. These constraints are imposed to data instances that, in the previous clustering, belonged to the same cluster. Subsequently, Gondek and Hofmann [2007] presented a method for finding an alternative clustering using an extension of the conditional information bottleneck [Gondek and Hofmann, 2003]. The idea is to generate a new clustering solution by maximizing the pairwise MI between the new clustering variable and the observed variables, conditioned on the previous clustering. Dang and Bailey [2010a] posteriorly presented NACI (non-linear alternative clustering with information theory), a hierarchical technique that uses information theory to discover an alternative clustering for

which: (i) the MI between its cluster labels and data instances is maximized, and (ii) the MI between the alternative clustering and the original clustering is minimized.

These three methods are only able to produce a single alternative clustering. However, there may exist more than two plausible groupings. Following this line of work, Cui et al. [2010] developed two methods that iteratively search for alternative clustering solutions in subspaces that are orthogonal to the previously found ones. Davidson and Qi [2008] proposed a different approach where new clustering solutions are generated by transforming the previously used data using a distance function that has been learned according to a set of must-link and cannot-link constraints. This idea was subsequently improved by Qi and Davidson [2009], who proposed to minimize the KL divergence between the probability distributions of the original and the transformed data. Alternatively, Niu et al. [2014] devised a spectral clustering algorithm that generates alternative clusterings by minimizing the Hilbert-Schmidt independence criterion. Finally, Yang and Zhang [2017] have recently proposed to use nonnegative matrix factorization to find multiple alternative clusterings.

4.3.1.2 Simultaneous clustering algorithms

Simultaneous clustering algorithms generate new data partitions without taking into consideration any of the previous ones. Caruana et al. [2006] first formulated an approach that is able to generate a set of potentially interesting clustering solutions by either randomly initializing the clustering algorithm or by using random feature weights. This collection of solutions is subsequently grouped using agglomerative clustering based on their similarity. Jain et al. [2008] then proposed two clustering algorithm: (i) a K-means variant that returns uncorrelated clusterings based on the notion of orthogonality, and (ii) a sum of parts approach, which models the clustering problem as one of learning the component distributions when data has been sampled from a convolution of mixture distributions. A model-based approach called CAMI (clustering for alternatives with mutual information) was later proposed by Dang and Bailey [2010b]. CAMI is essentially a regularized version of the EM algorithm that maximizes the likelihood of each clustering, while at the same time, minimizes the MI between them. Both Guan et al. [2010] and Niu et al. [2012] introduced non-parametric Bayesian models that are able to discover multiple clustering solutions and the feature subsets that are relevant to each of them. Alternatively, Ye et al. [2016] proposed to identify several arbitrarily oriented subspaces in which clusterings may exist. This work has been recently updated by using independence subspace analysis [Mautz et al., 2020]. Finally, while not explicitly included in this section, multidimensional clustering is considered a type of simultaneous clustering.

4.3.1.3 Other related approaches

A completely different strategy consists of generating multiple intermediate clustering solutions and combine them into a single clustering solution. Two approaches that follow this strategy are those of ensemble and multi-view clustering [Kriegel and Zimek, 2010].

- Ensemble clustering methods create a series of diverse base clusterings and then com-

bine them to produce a unified solution. We can distinguish two components in every ensemble method: the generation function, and the consensus function. The generation function is tasked with the introduction of diversity into the ensembles. The main sources of diversity are: (i) different subsets of observed variables, (ii) different subsets of data instances, and (iii) different clustering algorithms. Alternatively, the consensus function is tasked with the combination of intermediate clustering solutions. For more information on this topic, see Strehl and Ghosh [2003] and Vega-Pons and Ruiz-Shulcloper [2011].

- Multi-view clustering methods search for clusterings in different subspaces and then combine them into a single solution. Three main approaches can be distinguished: (i) co-training [Bickel and Scheffer, 2004], (ii) subspace clustering [Kriegel et al., 2009; Zhang et al., 2018], and (iii) multiple kernel clustering [Zhao et al., 2009]. See Yang and Wang [2018] for a recent review on multi-view clustering.

Part III

CONTRIBUTIONS

Chapter 5

Incremental learning of latent forests from mixed data

5.1 Introduction

Forests of LTMs are appealing owing to their ability to separate groups of similar variables in different trees. Although several methods for learning latent forests have been studied [Harmeling and Williams, 2010; Mourad et al., 2011; Galimberti et al., 2018], they have two main limitations. First, current methods rely on MLE of model parameters, which does not provide a measure of uncertainty, and does not allow to incorporate prior knowledge into the learning process. Second, these methods are currently limited to work with either categorical or continuous data.

In this chapter, we propose two incremental algorithms for learning forests of conditional linear Gaussian LTMs. Unlike current methods, the proposed algorithms are based on the VB framework, which allows them to work with mixed data and to introduce uncertainty into the learning process. Our first algorithm, which we refer to as incremental learner, hill-climbs the space of latent forests in a two-phase iterative process. In its first phase, the forest structure is expanded with a new arc or latent variable. In its second phase, the cardinalities of latent variables are estimated. Our second algorithm, which we name constrained incremental learner, modifies the previous procedure by considering only a subset of candidate structures in each step of the search. We demonstrate the effectiveness of our proposals by comparing them with existing methods. To this purpose, we conduct density estimation experiments using categorical, continuous, and mixed real-world data.

This chapter includes the content of Rodriguez-Sanchez et al. [2020]. In addition, all code, data, and results are available at https://github.com/ferjorosa/incremental-latent-forests.

Chapter outline

In Section 5.2, we present our approaches for incrementally learning forests of conditional linear Gaussian LTMs. Then, in Section 5.3, we explore the performance of our methods with respect to the state of the art for various experimental settings: (i) categorical data, (ii) continuous data, and (iii) mixed data. Finally, we report our conclusions and propose future research directions in Section 5.4.

5.2 Incremental learning

In this section, we propose a search-based method that hill-climbs the space of latent forests using the VB framework. Instead of exploring this space directly, we approach this search as an iterative process with two phases:

- **Structure phase**. The forest structure is incrementally built, in which variables are connected via a new arc or latent variable.
- **Cardinalities phase**. The cardinalities of previously involved latent variables are estimated.

Our search method uses two search operators (latent variable introduction and arc introduction) to modify the forest structure, and another two operators (cardinality increase and cardinality decrease) to estimate the cardinalities of latent variables. We describe these operators in Section 5.2.1 and present our search procedure in Section 5.2.2. An alternative search method that considers fewer candidate models is presented in Section 5.2.3. In Section 5.2.4, we describe an efficient method for evaluating candidate models. Finally, in Section 5.2.5, we address the problem of specifying prior distributions.

5.2.1 Search operators

- **Latent variable introduction** (LI) generates a new model by introducing a new categorical latent variable H as the parent of two variables (observed or latent) that currently have no parent. The cardinality of H is set to two. We only consider pairs of variables to reduce the computational complexity of this operator. This restriction is compensated by the following operator.
- **Arc introduction** (AI) creates a new model by introducing an arc from one variable to another. In this operator, both directions are considered. Since we focus on conditional linear Gaussian LTMs, arcs from continuous variables to categorical variables are ignored by the operator.
- **Cardinality increase** (CI) creates a new model by increasing the cardinality of a categorical latent variable H by one. This operator is not applicable if the variable has a cardinality equal to k_{max}, which can be specified by the user.

- **Cardinality decrease** (CD) produces a new model by decreasing the cardinality of a categorical latent variable H by one. This operator is not applicable if H already has a cardinality of two.

5.2.2 Search procedure

The search starts with an unconnected forest $\mathcal{B} = (\mathcal{G}, \boldsymbol{\theta})$ in which no latent variables $\mathbf{H}$ are present and whose observed variables $\mathbf{X}$ are independent. These observed variables form the working set $\mathbf{W}$. Then, the parameters of this model are estimated via the VB-EM algorithm and its corresponding score is stored as a baseline for comparison. Once the initial model has been estimated, the search method explores the space of latent forests by alternating between the structure and cardinalities phases. The search continues until the model score ceases to increase or there is only one remaining variable in $\mathbf{W}$. We call this method incremental learner (IL) and it is formally described in Algorithm 5.1. Its two phases are as follows:

- In the structure phase, the LI and AI operators are applied to each pair of variables in $\mathbf{W}$, resulting in the $\mathbf{B}_{LI}$ and $\mathbf{B}_{AI}$ sets of candidate models. Each candidate model is then evaluated by estimating its parameters and storing its score. When this process is completed, the highest scoring model $\mathcal{B}_S$ is selected, and all of its latent variables that were involved in the selected operator are stored in a new set of variables $\mathbf{V}$. We refer to this phase as the `structure` subroutine and it is formally described in Algorithm 5.2.

- In the cardinalities phase, the CI and CD operators are applied to the latent variables in $\mathbf{V}$, resulting in the $\mathbf{B}_{CI}$ and $\mathbf{B}_{CD}$ sets of candidate models. In contrast to the structure phase, this is done iteratively. It starts with $\mathcal{B}_C$ and then, at each iteration, it applies the CI and CD operators, selecting the best candidate model $\mathcal{B}'_C$. If its score is greater than that of $\mathcal{B}_C$, the candidate takes its place and the process continues. Once the score ceases to improve, the resulting model is returned as $\mathcal{B}_C$. We refer to this phase as the `cardinalities` subroutine and it is formally described in Algorithm 5.3.

Fig. 5.1 provides an example execution of the IL algorithm. It starts with an unconnected latent forest with sets of two categorical and two continuous observed variables, $\{X_1, X_2\}$ and $\{X_3, X_4\}$, respectively. In its first iteration, the AI operator is selected and X_3 becomes the new parent of X_4, resulting in the removal of X_4 from $\mathbf{W}$. In its second iteration, the LI operator is selected and a new latent variable H_1 is included as the parent of X_2 and X_3, thus removing both X_2 and X_3 from $\mathbf{W}$. The cardinality of H_1 is estimated and it remains at its initial value (i.e., two). In the third and last iteration, the AI operator is selected again, and H_1 is set as the parent of X_1. Unlike the previous iteration, the cardinality of H_1 is increased to four. Given that only one variable remains in $\mathbf{W}$, the process stops, and if the score of the model is greater than that of the previous iteration, then $\mathcal{B}$ is updated and returned.

Algorithm 5.1: Incremental learner (IL)

Input : $\mathcal{D}$, k_{max}

$\mathbf{W} \leftarrow \mathbf{X}$
Let $\mathcal{B}$ be an unconnected latent forest with nodes $\mathbf{X}$
while $|\mathbf{W}| \neq 1$ **do**
    $\mathcal{B}_S, \mathbf{V} \leftarrow$ `structure`($\mathcal{D}$, $\mathcal{B}$, $\mathbf{W}$)
    $\mathcal{B}_C \leftarrow$ `cardinalities`($\mathcal{D}$, $\mathcal{B}_S$, $\mathbf{V}$, k_{max})
    if score($\mathcal{B}_C : \mathcal{D}$) > score($\mathcal{B} : \mathcal{D}$) **then**
        $\mathcal{B} \leftarrow \mathcal{B}_C$
        Update $\mathbf{W}$ with $\mathbf{V}$
    else
        break /* Stop the loop */
Model refinement (see Section 5.2.4 for details)

Output: The resulting latent forest $\mathcal{B}$

Algorithm 5.2: `structure`

Input : $\mathcal{D}$, $\mathcal{B}$, $\mathbf{W}$

$\mathbf{B}_{LI} \leftarrow$ LI($\mathcal{D}$ $\mathcal{B}$, $\mathbf{W}$)
$\mathbf{B}_{AI} \leftarrow$ AI($\mathcal{D}$, $\mathcal{B}$, $\mathbf{W}$)
$\mathcal{B}_S \leftarrow$ highest scoring model in $\{\mathbf{B}_{LI}, \mathbf{B}_{AI}\}$
$\mathbf{V} \leftarrow$ latent variables involved in the selected operator

Output: The resulting latent forest $\mathcal{B}_S$ and $\mathbf{V}$

Algorithm 5.3: `cardinalities`

Input : $\mathcal{D}$, $\mathcal{B}$, $\mathbf{V}$, k_{max}

$\mathcal{B}_C \leftarrow \mathcal{B}$
while *True* **do**
    $\mathbf{B}_{CI} \leftarrow$ CI($\mathcal{D}$, $\mathcal{B}_C$, $\mathbf{V}$, k_{max})
    $\mathbf{B}_{CD} \leftarrow$ CD($\mathcal{D}$, $\mathcal{B}_C$, $\mathbf{V}$)
    $\mathcal{B}'_C \leftarrow$ highest scoring model in $\{\mathbf{B}_{CI}, \mathbf{B}_{CD}\}$
    if score($\mathcal{B}'_C : \mathcal{D}$) > score($\mathcal{B}_C : \mathcal{D}$) **then**
        $\mathcal{B}_C \leftarrow \mathcal{B}'_C$
    else
        break /* Stop the loop */

Output: The resulting latent forest $\mathcal{B}_C$

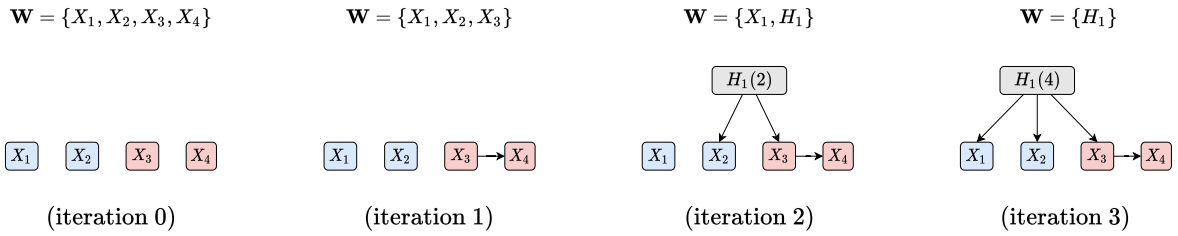


Figure 5.1: Example execution of the IL algorithm with three iterations. It introduces an arc in its first iteration, a latent variable in its second iteration, and another arc on its third (and last) iteration. Variable colors and parentheses have the same meaning as in Figure 3.1.

Model evaluation

Every application of a search operator produces a set of candidate models. Each of these models is then evaluated by estimating its parameters via the VB-EM algorithm and storing the resulting score. The scoring function that is traditionally used within the VB framework is the ELBO (see Equation 2.8). However, when applied to models with categorical latent variables, an adjustment must be made to the ELBO in order to account for the parameters' lack of identifiability. To better understand this problem, consider the CPD of a variable whose parent is a categorical latent variable with k labels. There are $k!$ equivalent settings of parameters for this CPD that merely differ by permuting the parent labels. Two common approaches exist for addressing this redundancy: (i) using asymmetric priors, and (ii) introducing a small penalty into the ELBO score. We use the second approach when there is a lack of expert knowledge in the learning process. In this context, a simple penalty involves subtracting the term $\log k!$ from the ELBO [Bishop, 2016]. We can generalize this idea for a model with multiple latent variables the following way. Let $\mathcal{B}$ be a latent variable model with s categorical latent variables, each latent variable H_i with a cardinality k_i. We define the scoring function, $\text{score}(\mathcal{B} : \mathcal{D})$, as a penalized version of the ELBO (p-ELBO), where $Q_{\mathcal{B}}$ refers to the variational distribution of $\mathcal{B}$:

$$\text{score}(\mathcal{B} : \mathcal{D}) = \text{p-ELBO}(Q_{\mathcal{B}} : \mathcal{D}) = \text{ELBO}(Q_{\mathcal{B}} : \mathcal{D}) - \sum_{i=1}^{s} \log k_i! \ . \tag{5.1}$$

Complexity analysis

The following auxiliary variables are considered for the complexity analysis of IL:

- $\mathbf{W} \rightarrow$ current set of variables with no parents (i.e., the working set).
- $k_{max} \rightarrow$ maximum latent cardinality.

In the structure phase, the LI and AI operators evaluate a total of $O(3(|\mathbf{W}|^2 - |\mathbf{W}|)/2)$ candidate models. In the cardinalities phase, the number of evaluations depend on the

previously selected operator. The maximum number of evaluations occurs when a latent variable is the new parent of two other latent variables, resulting in $O(3(k_{max}-2))$ evaluations for CI and CD. Therefore, the total number of candidate models evaluated at each iteration is $O(3(|\mathbf{W}|^2-|\mathbf{W}|)/2+3(k_{max}-2))$. Note that each evaluation requires running the VB-EM algorithm.

As can bee seen, IL can be very time-consuming due to two factors: (i) the evaluation of a large number of models, and (ii) the VB-EM algorithm. To address these challenges, we propose an alternative search procedure that considers fewer candidate models in Section 5.2.3, and in Section 5.2.4, we propose a local version of the VB-EM algorithm that speeds up model evaluation.

5.2.3 Constrained search

A straightforward approach for increasing the speed of IL is to introduce restrictions into the search process [Tsamardinos et al., 2006; Gámez et al., 2011]. We achieve this by only evaluating certain structures. More specifically, in the structure phase, instead of considering all pairs of variables in the working set $\mathbf{W}$, we select the pair of variables $\{W_i, W_j\}$ with the highest MI. The rationale for this approach stems from the following observation [Harmeling and Williams, 2010]. Let $P(\mathbf{W})$ be the probability distribution over the set of variables in $\mathbf{W}$. Knowing that all variables in $\mathbf{W}$ are considered to be independent of each other, we can approximate $P(\mathbf{W})$ with another distribution, $\hat{P}(\mathbf{W})$, that models the joint distribution $P(W_i, W_j)$ of the selected pair and the marginal distributions of the remaining variables:

$$\hat{P}(\mathbf{W}) = P(W_i, W_j) \prod_{k \neq i,j} P(W_k) = \frac{P(W_i, W_j)}{P(W_i)P(W_j)} \prod_k P(W_k) \ .$$

$P(W_i, W_j)$ represents the connection (via an arc or a latent variable) between W_i and W_j. Then, our objective is to select the pair of variables whose connection makes $\hat{P}(\mathbf{W})$ as close to $P(\mathbf{W})$ as possible. Evaluating this separation using the KL divergence, we have:

$$\begin{aligned}
\mathrm{KL}(P||\hat{P}) &= \int P(\mathbf{W}) \ log \frac{P(\mathbf{W})}{\hat{P}(\mathbf{W})} d\mathbf{W} \\
&= \int P(\mathbf{W}) \ log \frac{P(\mathbf{W})P(W_i)P(W_j)}{P(W_i, W_j)\prod_k P(W_k)} d\mathbf{W} \\
&= \int P(\mathbf{W}) \ log \frac{P(\mathbf{W})}{\prod_k P(W_k)} d\mathbf{W} - I(W_i, W_j) \ ,
\end{aligned}$$

where $I(W_i, W_j)$ is the MI between W_i and W_j and it is defined as

$$I(W_i, W_j) = \int \int P(W_i, W_j) \log \frac{P(W_i, W_j)}{P(W_i)P(W_j)} dW_i dW_j \ . \tag{5.2}$$

By selecting the pair of variables with the highest MI, we minimize the KL divergence between $P(\mathbf{W})$ and $\hat{P}(\mathbf{W})$. However, $I(W_i, W_j)$ is often intractable to compute given that W_i

Algorithm 5.4: Constrained incremental learner (CIL)

Input : $\mathcal{D}$, k_{max}, α

1 $\mathbf{W} \leftarrow \mathbf{X}$
2 Estimate the MI matrix with each pair in $\mathbf{W}$
3 Let $\mathcal{B}$ be an unconnected latent forest with nodes $\mathbf{X}$
4 **while** $|\mathbf{W}| \neq 1$ **do**
5 $\mathbf{U} \leftarrow \alpha$ pairs of variables in $\mathbf{W}$ with the highest MI
6 $\mathcal{B}_S, \mathbf{V} \leftarrow$ `structure`($\mathcal{D}$, $\mathcal{B}$, $\mathbf{U}$)
7 $\mathcal{B}_C \leftarrow$ `cardinalities`($\mathcal{D}$, $\mathcal{B}_S$, $\mathbf{V}$, k_{max})
8 **if** score($\mathcal{B}_C : \mathcal{D}$) > score($\mathcal{B} : \mathcal{D}$) **then**
9 $\mathcal{B} \leftarrow \mathcal{B}_C$
10 Update $\mathbf{W}$ with $\mathbf{V}$
11 Update the MI matrix with respect to $\mathbf{W}$
12 **else**
13 **break** /* Stop the loop */
14 Model refinement (see Section 5.2.4 for details)

Output: The resulting latent forest $\mathcal{B}$

and W_j can be either categorical or continuous. For this reason, approximate MI estimators, such as binning, k-nearest neighbors [Kraskov et al., 2004], or kernel density estimation [Moon et al., 1995], are required. By approximating the MI, we cannot ensure that the selected pair of variables results in the truly minimal $\mathrm{KL}(P||\hat{P})$. A simple way to alleviate this problem is to consider the α pairs of variables with the highest MI.

We apply this idea to IL, resulting in the constrained incremental learner (CIL), which is formally described in Algorithm 5.4. CIL is very similar to IL. The main execution differences occur: (i) at the beginning of the algorithm (i.e., line 2), where CIL estimates the MI matrix with each pair of variables in $\mathbf{W}$, (ii) in the structure phase (i.e., line 6), where CIL only considers the α pairs of variables in $\mathbf{W}$ with the highest MI, and (iii) at the end of each iteration (i.e., line 12), where CIL updates the MI matrix. The update removes the selected pair of variables and estimates the MI of those variables that are new in $\mathbf{W}$ (i.e., the MI of a newly created latent variable and the rest of variables in $\mathbf{W}$). In this method, α represents a compromise between model quality and computational cost.

Complexity analysis

CIL generates fewer candidate models than IL at each step of the search. While its cardinalities phase is identical to that of IL, its structure phase depends only on α, which results in a total of $O(2\alpha + 3(k_{max} - 2))$ evaluations at each step of the search.

5.2.4 Efficient model evaluation

Repeatedly evaluating candidate models can become prohibitive when each evaluation involves running the VB-EM algorithm. Previously, we addressed this problem by reducing the number of candidates. However, when the number of variables is high, this reduction may be insufficient. One solution that has been successfully applied in the phylogenetic [Guindon and Gascuel, 2003] and LTM [Chen et al., 2012; Harmeling and Williams, 2010] literature involves optimizing some parameters of the model while the rest of parameters are kept unchanged.

Following this idea, we propose to replace the VB-EM algorithm with a more efficient procedure that is able to estimate the variational posterior distributions of a subset of variables while the variational posterior distributions of the rest of variables are kept unchanged. We refer to this method as local VB-EM. Thus, when evaluating a candidate model using local VB-EM, we estimate the variational posteriors of: (i) variables involved in the search operator, and (ii) variables belonging to the MBs of the variables in (i).

To illustrate the behavior of local VB-EM, let us examine the rightmost example in Figure 5.1 (iteration 3), which results from applying the AI operator on H_1 and X_1. Evaluating this model requires the estimation of the variational posteriors of X_1 and H_1 as well as X_2 and X_3, which belong to the MB of H_1. Incidentally, we do not require to update X_4 because it does not belong to the MB of either H_1 or X_1.

One iteration of local VB-EM is computationally much cheaper than one iteration of VB-EM because it updates fewer variational parameters. This also implies that a run of local VB-EM requires fewer steps to converge than a run of VB-EM. However, parameter estimates provided by local VB-EM may deviate from those provided by VB-EM. To prevent this from affecting the quality of the IL and CIL results, we can perform a run of VB-EM after several search steps or before returning the model. We advocate for the second approach, performing a run of VB-EM before returning the model.

Finally, like VB-EM, local VB-EM may get trapped at local maxima. To avoid this, we use a variant of the multiple-restart approach proposed by Chickering and Heckerman [1997] and adapt it for the variational case. First, we sample c initial configurations of the variational parameters $\boldsymbol{\varphi}$ being considered. Next, we perform one local VB-EM step and retain $c/2$ of the configurations that lead to the largest score values. Then, we perform two local VB-EM steps and retain $c/4$ configurations. We continue this procedure, doubling the number of local VB-EM steps at each iteration until only one configuration of the parameters remains.

5.2.5 Prior specification

A key aspect in Bayesian learning is the specification of prior distributions, which includes their form and parameters. Since IL and CIL are designed to work with CE models, the forms of prior distributions are already established by the VMP algorithm (see Winn and Bishop [2005]), and it is only necessary to focus on their parametrization.

Tuning prior parameters is largely dependent on the availability of expert information.

This information can be obtained from a person or from other directly related studies (see Bayesian meta-analysis [Gelman et al., 2013]). When expert information is available, prior parameter values can be selected to best reflect the expert knowledge. When this information is unavailable, we propose using the following strategy, which varies for observed and latent variables. First, for observed variables, we use an empirical Bayes [Casella, 1985] approach and assign maximum likelihood estimates to their prior parameters. Second, for categorical latent variables, we assume a symmetric Dirichlet prior with a total concentration of 1.

5.3 Experiments

In this section, we evaluate the performances of IL and CIL in terms of data fitting and computational complexity. For CIL, we used α values of 1 and 10. The goal of using these values was to evaluate whether an increase in α produced better results or simply an increase in computational complexity. To this end, we conducted comparative density estimation studies using categorical (Section 5.3.1), continuous (Section 5.3.2) and mixed (Section 5.3.3) real-world data. In these experiments, IL and CIL were compared to several LTM methods of the state of the art. However, given that most of these algorithms are specifically designed to work in either the categorical or continuous domain, we complemented these comparisons with other approaches that are not based on LTMs but can deal with mixed data. The following approaches were considered:

- **EAST**. Expansion, adjustment and simplification until termination algorithm [Chen et al., 2012]. Score-based method that constructs categorical HLCMs following a greedy process with five operators. We used the Java implementation provided as an executable at http://www.cse.ust.hk/∽lzhang/ltm/softwares/EAST.zip.

- **GEAST**. Gaussian version of the EAST algorithm for continuous data [Poon et al., 2013]. It constructs HLCMs with categorical latent variables and continuous observed variables. We used the Java implementation provided at https://github.com/kmpoon/pltm-east.

- **BI**. Bridged islands algorithm [Liu et al., 2015]. Variable clustering method that constructs categorical HLCMs. It first generates a series of categorical LCMs and then connects them using the Chow-Liu method [Chow and Liu, 1968]. We used the Java implementation provided as an executable at http://www.cse.ust.hk/∽lzhang/ltm/softwares/BI.zip.

- **KDE**. Kernel density estimator that can deal with mixed data, and uses Silverman's rule of thumb for bandwidth selection [Li and Racine, 2003]. We used the multivariate implementation provided by the Python library Statsmodels (https://www.statsmodels.org/).

- **MSPN**. Mixed sum-product networks [Molina et al., 2018], which combine sum-product networks [Poon and Domingos, 2011] with non-parametric estimation to learn hierarchically structured latent variable models that do not require the specification of variables'

Dataset	m	N	Origin
HIV-test	4	428	[Zhang, 2004]
Hayes-roth	5	160	[Dua and Graff, 2019]
Balance-scale	5	625	[Dua and Graff, 2019]
Car-evaluation	7	1728	[Dua and Graff, 2019]
Nursery	9	12960	[Dua and Graff, 2019]
Breast-cancer	10	277	[Dua and Graff, 2019]
Web-phishing	10	1353	[Dua and Graff, 2019]
Solar-flare	13	1389	[Dua and Graff, 2019]
Zoo	17	101	[Dua and Graff, 2019]
Vote	17	232	[Dua and Graff, 2019]
Spect-heart	23	267	[Dua and Graff, 2019]
Alarm	36	1000	[Liu et al., 2015]
Coil-42	42	5822	[Zhang et al., 2008a]
News-100	100	16242	[Liu et al., 2015]
Webkb-336	336	1038	[Liu et al., 2015]

m: total number of variables.
N: sample size.

Table 5.1: Basic properties of the categorical datasets. The datasets are sorted in ascending order according to their dimensionality.

parametric forms. We used the implementation provided by the Python library SPFlow (https://github.com/SPFlow/SPFlow).

- **GLFM**. General latent feature model [Valera et al., 2020], which handles both categorical and continuous observed variables using a Bayesian nonparametric model with Markov chain Monte Carlo inference. We ran 1000 iterations of the sampler using the Python implementation available at https://github.com/ivaleraM/GLFM.

As a performance measure, we used the 10-fold cross-validated predictive log-likelihood (CVPLL). That is, we divided each dataset into 10 equal-sized folds, trained a model on 9 of them, and computed the predictive log-likelihood on the remaining fold. Both IL and CIL were implemented in Java 8 using the AMIDST toolbox [Masegosa et al., 2019]. Experiments were conducted on a computer with an Intel Core i7-6700K CPU at 4.00 GHz with 64GB RAM, running Windows 10 Enterprise.

5.3.1 Comparative study on categorical data

To conduct the study, we used 15 real-world categorical datasets of various dimensionalities (m) and sample sizes (N). The majority of these datasets were retrieved from the UCI repository [Dua and Graff, 2019]. We also added several datasets that had been previously used in the literature. Table 5.1 shows the basic properties of each dataset. None of these datasets provided expert knowledge to set Bayesian priors. For this reason, we set the prior parameters of the IL and CIL models using an empirical Bayes approach.

Tables 5.2 and 5.3 display the average (and standard deviation) results for the CVPLL and execution time, respectively. The maximum time allowed per fold was one week. The winner in each row is highlighted in blue. For a better comparison between IL and CIL, we

	EAST		BI		MSPN		GLFM		IL		CIL [α =1]		CIL [α =10]	
HIV-test	−1485.89	(4 484.39)	−85.61	(59.44)	−104.84	(100.70)	−93.68	(147.46)	−84.22	(51.96)	−84.48	(52.02)	−84.22	(51.96)
Hayes-roth	−89.72	(5.91)	−89.98	(5.68)	−88.89	(7.87)	−110.67	(12.99)	−89.09	(5.71)	−89.09	(5.71)	−89.09	(5.71)
Balance-scale	−492.48	(16.81)	−491.67	(16.57)	−493.10	(45.40)	−505.65	(34.91)	−487.16	(15.78)	−487.16	(15.78)	−487.16	(15.78)
Car-evaluation	−12 095.26	(9 825.73)	−1610.46	(251.41)	−1690.90	(241.26)	−1728.79	(128.60)	−1503.81	(185.32)	−1721.09	(178.65)	−1503.81	(185.32)
Nursery	−264 750.76	(228 444.63)	−20 307.37	(2 348.33)	−15 288.31	(1 017.49)	−14 277.25	(956.67)	−13 626.52	(592.82)	−14 153.90	(1 601.28)	−13 649.92	(576.75)
Breast-cancer	−321.55	(91.26)	−258.56	(24.74)	−310.73	(39.61)	−415.29	(66.56)	−261.36	(24.95)	−262.44	(25.04)	−261.36	(24.95)
Web-phishing	−1018.54	(19.76)	−1010.19	(20.93)	−1013.53	(18.21)	−1758.30	(126.83)	−1074.22	(20.75)	−1083.07	(25.23)	−1074.22	(20.75)
Solar-flare	−973.54	(187.98)	−949.32	(131.62)	−1106.10	(131.89)	−1433.10	(147.16)	−1026.94	(131.25)	−1033.62	(131.97)	−1026.94	(131.25)
Zoo	−122.54	(190.95)	−71.86	(20.60)	−108.94	(21.31)	−214.64	(50.84)	−93.05	(20.94)	−92.31	(19.95)	−93.05	(20.94)
Vote	−216.01	(79.54)	−176.49	(21.45)	−174.27	(19.84)	−440.29	(99.99)	−193.73	(24.38)	−194.44	(25.47)	−193.73	(24.38)
Spect-heart	−288.35	(46.97)	−289.78	(46.94)	−316.32	(61.76)	−459.13	(95.03)	−299.42	(41.52)	−299.42	(41.52)	−299.42	(41.52)
Alarm	−1179.96	(536.68)	−1352.71	(555.29)	−4142.37	(1 027.10)	−5685.95	(744.38)	−1461.45	(564.23)	−1430.75	(499.24)	−1452.13	(535.00)
Coil-42	−30 597.77	(45 573.34)	−6978.14	(1 501.16)	−6879.65	(860.33)	−8766.60	(1 107.85)	−6269.04	(918.50)	−6316.36	(943.68)	−6269.04	(918.50)
News-100	*time*		−25 343.09	(6 113.94)	−26 768.92	(6 971.29)	−27 434.12	(7 383.22)	*time*		−26 301.65	(6 531.46)	−26 325.91	(6 576.45)
Webkb-336	*time*		−8430.17	(1 681.82)	−9211.57	(1 733.69)	−10 721.51	(2 309.92)	*time*		−9440.12	(1 990.32)	−9439.57	(1 990.87)

The winner in each row is highlighted in blue. Results from IL and CIL that outperformed those from existing methods without ranking first place are highlighted in yellow. Numbers between parentheses correspond to standard deviations.
time: the algorithm was unable to complete a fold in a week.

Table 5.2: 10-fold cross-validated predictive log-likelihood for various categorical datasets (rows) and algorithms (columns).

	EAST		BI		MSPN		GLFM		IL		CIL [α =1]		CIL [α =10]	
HIV-test	0.66	(0.13)	0.57	(0.14)	0.30	(0.11)	8.57	(0.14)	1.22	(0.02)	0.51	(0.05)	1.00	(0.05)
Hayes-roth	2.15	(0.07)	1.71	(0.09)	1.50	(0.11)	6.05	(0.15)	1.45	(0.07)	0.41	(0.02)	0.99	(0.09)
Balance-scale	7.94	(0.12)	5.63	(0.12)	5.28	(1.66)	32.05	(1.12)	3.51	(0.09)	2.23	(0.38)	4.19	(0.18)
Car-evaluation	243.66	(52.14)	38.83	(21.99)	4.04	(1.12)	178.14	(10.80)	32.09	(0.23)	7.29	(1.43)	27.85	(0.62)
Nursery	6977.54	(2 471.52)	496.93	(280.72)	4.12	(0.70)	1238.84	(153.49)	800.37	(20.26)	44.87	(1.94)	449.10	(21.87)
Breast-cancer	44.63	(11.41)	8.50	(1.28)	3.86	(0.51)	31.06	(0.44)	30.86	(1.98)	9.05	(1.86)	23.25	(2.03)
Web-phishing	327.40	(67.68)	46.00	(6.01)	9.00	(1.51)	183.10	(39.08)	125.66	(5.47)	12.68	(2.17)	54.12	(1.40)
Solar-flare	349.84	(59.14)	50.62	(29.19)	8.67	(0.78)	125.47	(4.63)	335.51	(30.90)	11.53	(1.95)	99.80	(1.68)
Zoo	50.90	(9.53)	9.54	(1.06)	4.29	(0.65)	31.44	(12.18)	49.32	(1.70)	1.45	(0.09)	8.32	(0.44)
Vote	96.00	(15.19)	13.77	(3.40)	3.61	(0.38)	30.76	(2.76)	64.31	(1.21)	2.42	(0.08)	12.71	(0.25)
Spect-heart	328.52	(50.97)	12.84	(2.08)	8.46	(1.18)	42.00	(5.02)	237.04	(4.15)	4.66	(0.05)	27.80	(0.44)
Alarm	4592.03	(746.44)	67.42	(3.72)	75.98	(9.73)	482.41	(170.77)	6209.85	(55.82)	40.67	(4.67)	293.53	(15.44)
Coil-42	27 531.16	(3 032.85)	254.00	(14.54)	33.55	(3.00)	1360.87	(7.63)	118 844.19	(2 165.96)	261.85	(36.21)	2446.87	(116.52)
News-100	*time*		2537.71	(150.19)	142.22	(6.91)	9893.96	(1 685.00)	*time*		4809.26	(408.17)	38 156.47	(2 320.58)
Webkb-336	*time*		1244.90	(36.40)	2139.86	(344.65)	2846.41	(13.53)	*time*		2266.56	(59.96)	22 448.31	(926.31)

The winner in each row is highlighted in blue. Results from IL and CIL that outperformed those from existing methods without ranking first place are highlighted in yellow. Numbers between parentheses correspond to standard deviations.
time: the algorithm was unable to complete a fold in a week.

Table 5.3: 10-fold cross-validated execution times (in seconds) for various categorical datasets (rows) and algorithms (columns).

highlighted in yellow the score results from CIL that were in second place behind IL. Our objective was to evaluate whether CIL was able to produce top results in less time than IL. In this study, CIL and IL were compared with EAST, BI, GLFM and MSPN. Although KDE was initially considered, its results were not included due to space limitations. In addition, KDE was unable to outperform any of the considered methods and returned an error in three of the datasets.

Results indicate that IL and CIL are competitive in terms of both model quality and execution time. Table 5.2 shows that IL and CIL had the best score performance in 5 out 15 datasets, where CIL [$\alpha = 10$] was second behind IL in one of the datasets. The best scoring method was BI, being first in 6 out 15 datasets. EAST and MSPN each obtained the highest score in 2 out of 15 datasets. GLFM was unable to reach first position in any of the considered datasets.

Table 5.3 shows that, in terms of speed, MSPN was the fastest of the seven methods evaluated, having the best performance in 8 out 15 datasets. It was followed by CIL [$\alpha = 1$], which had the best performance in 6 out 15 datasets. EAST and IL appeared to be the slowest methods, being unable to complete their executions in high-dimensional datasets due to their respective computational complexities.

Finally, while CIL was considerably faster than IL, we did not observe substantial differences between their respective scores. This observation was independent of the selected α value. We attribute this score similarity to the exact estimation of MI with categorical data (see Section 5.2.3). In fact, CIL [$\alpha = 1$] returned almost identical results to those of CIL [$\alpha = 10$] and IL in all the datasets (expect Car-evaluation)

5.3.2 Comparative study on continuous data

As with the comparative study on categorical data, we ran experiments on 15 real-world continuous datasets that were taken from the UCI repository and the current literature. Table 5.4 shows the basic properties of each dataset. We also used CVPLL as the evaluation measure and allowed a maximum execution time of one week per fold. In addition, given the absence of expert knowledge, we used an empirical Bayes approach to set the prior parameters of the IL and CIL models. CVPLLs and execution times of each method are displayed in Tables 5.5 and 5.6, respectively. In this study, CIL and IL were compared with GEAST, KDE, MSPN and GLFM.

Table 5.5 indicates that, in terms of model quality, GEAST was the best performing method, returning the highest CVPLL score in 7 out of 15 datasets. GEAST was followed by IL, which had the best performance in 4 out 15 datasets. It should be noted that even though CIL [$\alpha = 10$] had the best performance in only 3 out 15 datasets, it returned competitive results that outperformed all the considered methods (except IL) in 6 out of 15 datasets. KDE and MSPN each obtained the highest score in 1 out of 15 datasets. Similar to the categorical comparative study, GLFM was unable to outperform the rest of methods in any of the considered datasets.

Table 5.6 shows that, even though GEAST was the best performing method in terms

Dataset	m	N	Origin
Real-state	5	414	[Dua and Graff, 2019]
Buddymove	6	249	[Dua and Graff, 2019]
Qsar-fish	7	908	[Dua and Graff, 2019]
Qsar-aqua	9	545	[Dua and Graff, 2019]
ILPD	9	579	[Dua and Graff, 2019]
Alcohol	10	125	[Dua and Graff, 2019]
Travel-reviews	10	980	[Dua and Graff, 2019]
Wine-quality	12	4898	[Dua and Graff, 2019]
Wine	13	178	[Dua and Graff, 2019]
Leaf	14	340	[Dua and Graff, 2019]
NBA	18	441	[Poon et al., 2013]
WDBC	30	569	[Dua and Graff, 2019]
Waveform	40	5000	[Dua and Graff, 2019]
100-plants	64	1600	[Dua and Graff, 2019]
Geo-music	71	2856	[Dua and Graff, 2019]

m: total number of variables.
N: sample size.

Table 5.4: Basic properties of the continuous datasets. The datasets are sorted in ascending order according to their dimensionality.

of CVPLL, it still had the same computational issues with high-dimensional datasets as its categorical version (i.e., EAST). GEAST was unable to complete its execution in 3 of the experiments. Similarly, IL was unable to complete its execution in 1 of the experiments. In terms of speed, KDE was the fastest method in all of the considered datasets.

Although there were no substantial differences between the CVPLL scores of IL and CIL with categorical data, we observed the opposite behavior with continuous data. More specifically, there was no dataset in which CIL [$\alpha = 1$] resulted in the same score as IL. This coincides with our intuition from Section 5.2.3. When handling continuous or mixed data, we must rely on approximate MI methods, which may result in suboptimal solutions (i.e., the k-nearest neighbors approach proposed by Kraskov et al. [2004]). For these cases, increasing the number of candidate models considered on each step of the algorithm improves the quality of the results. This is illustrated by Table 5.5, which shows that CIL [$\alpha = 10$] performs considerably better than CIL [$\alpha = 1$].

5.3.3 Comparative study on mixed data

As with previous comparative studies, we ran experiments on 15 real-world mixed datasets taken from the UCI repository and the current literature. Table 5.7 shows the basic properties of each dataset. We also used CVPLL as the evaluation measure and allowed a maximum execution time of one week per fold. In addition, given the absence of expert knowledge, we used an empirical Bayes approach to set the prior parameters of the IL and CIL models. CVPLLs and execution times of each method are displayed in Tables 5.8 and 5.9, respectively. In this study, CIL and IL were compared with KDE, MSPN and GLFM.

Table 5.8 shows that, in terms of model quality, IL was the best performing method, returning the highest CVPLL score in 7 out of 15 datasets. It should be noted that even

	GEAST		KDE		MSPN		GLFM		IL		CIL [$\alpha = 1$]		CIL [$\alpha = 10$]	
Real-state	−246.58	(18.87)	−295.02	(11.79)	−539.61	(465.71)	−872.50	(92.10)	−380.16	(15.29)	−377.46	(23.88)	−380.16	(15.29)
Buddymove	−671.54	(95.00)	−663.69	(85.19)	−1258.76	(410.91)	−726.64	(97.73)	−662.24	(103.50)	−675.27	(99.13)	−662.24	(103.50)
Qsar-fish	−410.86	(38.37)	−561.57	(30.05)	−1223.66	(750.94)	−1885.14	(869.58)	−467.64	(42.30)	−666.64	(108.99)	−377.10	(52.04)
Qsar-aqua	−937.12	(111.26)	−1060.84	(101.30)	−718.65	(119.72)	−2106.64	(799.14)	−905.96	(222.15)	−970.60	(224.46)	−879.76	(231.69)
ILPD	−1567.17	(165.57)	−1834.17	(280.75)	−2125.03	(739.63)	−2024.70	(177.32)	−1443.60	(231.19)	−1772.20	(225.15)	−1448.62	(233.46)
Alcohol	−553.53	(75.94)	−611.08	(78.82)	−1544.25	(466.68)	−1483.47	(386.36)	−493.29	(63.69)	−527.98	(66.94)	−498.38	(67.62)
Travel-reviews	311.82	(32.71)	154.67	(22.73)	−953.40	(1 055.57)	−3595.92	(1 125.14)	81.12	(32.24)	67.12	(33.84)	63.92	(32.10)
Wine-quality	−1321.11	(277.53)	−2211.00	(196.45)	−8758.61	(2 567.19)	−16 505.01	(2 596.44)	−3013.58	(230.65)	−3378.24	(234.58)	−3083.70	(257.49)
Wine	−360.01	(68.19)	−365.86	(58.82)	−1093.08	(293.06)	−704.77	(379.01)	−400.20	(105.59)	−406.13	(93.15)	−398.06	(92.73)
Leaf	902.82	(143.46)	694.85	(42.35)	−2412.34	(687.03)	−1519.57	(1 518.66)	603.59	(161.66)	484.33	(163.57)	619.96	(142.93)
NBA	−399.21	(30.68)	−694.60	(16.54)	−2537.31	(243.05)	−3163.04	(281.89)	−660.75	(27.87)	−762.68	(55.17)	−681.09	(28.92)
WDBC	1168.41	(180.00)	360.47	(67.67)	−4383.39	(1 999.32)	−9966.35	(1 194.90)	721.22	(219.05)	353.83	(209.84)	615.81	(215.84)
Waveform	*time*		−32 643.86	(56.89)	−30 816.14	(207.60)	−40 771.48	(563.01)	−30 370.28	(59.91)	−31 238.63	(280.32)	−30 580.09	(152.53)
100-plants	*time*		31 175.92	(777.24)	−13 220.36	(13 736.14)	−15 808.74	(7 583.75)	28 239.13	(2 198.29)	28 193.23	(2 202.93)	28 197.25	(2 206.23)
Geo-music	*time*		−10 029.05	(294.76)	*time*		−29 926.40	(1 059.27)	*time*		−10 749.96	(374.64)	−9287.31	(365.14)

The winner in each row is highlighted in blue. Results from IL and CIL that outperformed those from existing methods without ranking first place are highlighted in yellow. Numbers between parentheses correspond to standard deviations.
time: the algorithm was unable to complete a fold in a week.

Table 5.5: 10-fold cross-validated predictive log-likelihood for various continuous datasets (rows) and algorithms (columns).

	GEAST		KDE		MSPN		GLFM		IL		CIL [$\alpha = 1$]		CIL [$\alpha = 10$]	
Real-state	302.12	(129.36)	0.01	(0.00)	4.66	(0.86)	29.88	(2.74)	2.56	(0.33)	1.11	(1.08)	2.18	(0.29)
Buddymove	213.78	(112.87)	0.00	(0.00)	4.48	(1.03)	0.03	(0.01)	2.80	(0.09)	0.58	(0.15)	2.54	(0.19)
Qsar-fish	1911.91	(651.32)	0.04	(0.01)	13.36	(3.03)	124.79	(5.31)	28.92	(2.05)	7.40	(2.90)	29.11	(5.97)
Qsar-aqua	2292.40	(515.20)	0.02	(0.00)	11.21	(1.97)	87.92	(2.49)	39.53	(6.29)	19.23	(10.59)	34.96	(7.88)
ILPD	1757.13	(490.96)	0.02	(0.00)	15.89	(2.50)	0.09	(0.02)	43.16	(2.12)	2.49	(0.79)	29.00	(3.09)
Alcohol	316.98	(68.85)	0.00	(0.00)	5.36	(1.43)	11.37	(0.82)	6.46	(1.16)	1.07	(0.36)	3.13	(0.34)
Travel-reviews	4180.70	(362.00)	0.08	(0.02)	20.19	(4.47)	214.64	(45.86)	133.72	(12.18)	21.57	(6.18)	51.54	(9.17)
Wine-quality	169 021.63	(34 671.99)	1.68	(0.27)	91.95	(9.55)	1160.05	(67.81)	1366.39	(120.05)	25.38	(0.53)	318.31	(143.96)
Wine	795.98	(145.86)	0.01	(0.00)	13.89	(2.21)	29.55	(1.36)	21.59	(1.60)	0.94	(0.39)	4.40	(0.52)
Leaf	3941.66	(420.50)	0.02	(0.00)	9.76	(0.74)	85.35	(12.50)	61.19	(4.15)	0.59	(0.17)	12.77	(2.44)
NBA	13 608.33	(835.63)	0.03	(0.00)	59.87	(13.16)	210.85	(0.79)	257.69	(9.89)	5.61	(2.50)	49.60	(12.67)
WDBC	50 784.86	(3 315.22)	0.08	(0.02)	52.60	(1.43)	776.45	(10.12)	2436.78	(26.62)	8.90	(1.61)	57.90	(4.24)
Waveform	*time*		3.31	(0.14)	17.72	(3.90)	1662.30	(186.18)	45 831.88	(1 964.74)	383.15	(115.27)	1314.81	(244.66)
100-plants	*time*		0.68	(0.02)	604.52	(19.70)	8848.11	(335.22)	6936.40	(1 244.92)	61.20	(6.49)	58.20	(7.41)
Geo-music	*time*		0.26	(0.01)	*time*		7465.81	(3 575.05)	*time*		64.99	(8.96)	1849.98	(751.94)

The winner in each row is highlighted in blue. Results from IL and CIL that outperformed those from existing methods without ranking first place are highlighted in yellow. Numbers between parentheses correspond to standard deviations.
time: the algorithm was unable to complete a fold in a week.

Table 5.6: 10-fold cross-validated execution times (in seconds) for various continuous datasets (rows) and algorithms (columns).

Dataset	m	#Categorical	#Continuous	N	
Haberman	4	1	3	306	[Dua and Graff, 2019]
Iris	5	1	4	150	[Dua and Graff, 2019]
User-knowledge	6	1	5	258	[Dua and Graff, 2019]
Vertebral	7	1	6	310	[Dua and Graff, 2019]
Ecoli	7	1	6	336	[Dua and Graff, 2019]
Planning-relax	12	1	12	4898	[Dua and Graff, 2019]
Thoracic-surgery	14	11	3	470	[Dua and Graff, 2019]
Vehicle	19	1	18	846	[Dua and Graff, 2019]
Thyroid	21	16	5	3103	[Dua and Graff, 2019]
Parkinsons	23	1	22	195	[Dua and Graff, 2019]
Autos	24	10	14	193	[Dua and Graff, 2019]
Ionosphere	34	1	33	351	[Dua and Graff, 2019]
Qsar-biodeg	41	1	40	1055	[Dua and Graff, 2019]
Housing-prices	64	33	34	1338	[Dua and Graff, 2019]
Census-india	144	1	143	1908	[Dua and Graff, 2019]

m: total number of variables.
#Categorical: number of categorical variables.
#Continuous: number of continuous variables.
N: sample size.

Table 5.7: Basic properties of the mixed datasets. The datasets are sorted in ascending order according to their dimensionality.

though CIL [$\alpha = 10$] was second behind IL with 6 out 15 datasets, it returned competitive results that outperformed all the considered methods (except IL) in 9 out of 15 datasets. KDE and MSPN each obtained the highest score in 2 out of 15 datasets. Similar to previous comparative studies, GLFM was unable to outperform the rest of methods in any of the considered datasets.

Table 5.6 illustrates that, similar to the continuous case, KDE was the fastest method, ranking first in 12 out 15 datasets. It was followed by CIL[$\alpha = 1$], which ranked first in the three datasets in which KDE returned an error. IL had computational issues with high-dimensional datasets and was unable to complete its execution in two of the experiments.

Similar to the continuous case, we observed considerable CVPLL differences between IL, CIL [$\alpha = 1$], and CIL [$\alpha = 10$]. This coincides with our previously discussed intuition (Section 5.2.3) about approximate MI estimation. Although CIL obtained slightly worse results (in terms of model quality) than IL, it was able to work with high-dimensional data, which makes it a reasonable alternative to IL when the number of variables is high or there is a computational limitation.

5.4 Conclusions and future work

In this chapter, we proposed an incremental method that, in combination with the VB framework, is able to learn forests of conditional linear Gaussian LTMs from data. Considering that directly searching this space requires the evaluation of a large number of candidate models, a constrained variant that only evaluates the most prominent α candidates of each iteration was also developed. As demonstrated by the experimental results, the constrained approach

	KDE		MSPN		GLFM		IL		CIL [$\alpha = 1$]		CIL [$\alpha = 10$]	
Haberman	−319.16	(38.41)	−1012.93	(379.19)	−376.13	(44.48)	−278.13	(35.51)	−325.79	(43.90)	−278.13	(35.51)
Iris	−69.51	(6.97)	−194.20	(124.36)	−406.22	(339.04)	−51.10	(12.05)	−46.31	(11.63)	−51.11	(12.05)
User-knowledge	−29.03	(20.15)	−194.36	(188.05)	−292.22	(125.60)	−20.15	(22.66)	−20.15	(22.66)	−20.15	(22.66)
Vertebral	−798.78	(72.53)	−610.13	(237.69)	−1192.51	(444.36)	−763.81	(68.16)	−779.00	(61.80)	−763.84	(68.14)
Ecoli	*error*		−180.73	(190.28)	−592.32	(1 205.38)	12.26	(125.52)	−4.43	(113.00)	12.24	(125.59)
Planning-relax	−94.31	(14.73)	−1087.97	(307.52)	−617.46	(194.49)	−44.72	(22.26)	−44.72	(22.26)	−44.72	(22.26)
Thoracic-surgery	−750.57	(5.73)	−671.51	(52.22)	−1726.67	(702.16)	−553.39	(14.57)	−566.58	(17.40)	−566.58	(17.40)
Vehicle	−5469.10	(122.74)	−8801.35	(3 295.42)	−20 567.24	(1 500.78)	−5245.73	(135.66)	−5662.02	(107.25)	−5309.80	(150.54)
Thyroid	−7151.07	(97.26)	−5943.21	(119.81)	−20 475.09	(2 517.29)	−6047.79	(192.81)	−6864.09	(294.07)	−6680.76	(301.05)
Parkinsons	562.01	(66.83)	−981.89	(933.66)	−946.90	(1 218.76)	458.41	(186.02)	407.76	(193.57)	429.48	(184.91)
Autos	*error*		−2136.42	(386.91)	−2528.45	(391.36)	−1478.48	(582.03)	−1369.79	(109.24)	−1308.21	(111.15)
Ionosphere	−682.32	(202.73)	−6633.65	(3 123.35)	−5846.25	(1 474.37)	−722.87	(284.33)	−811.49	(247.98)	−731.52	(279.00)
Qsar-biodeg	−5206.23	(212.84)	−2806.25	(357.13)	−18 775.94	(6 052.60)	−1272.40	(685.67)	−4397.80	(583.59)	−2128.14	(1 400.07)
Housing-prices	*error*		−17 357.98	(5 216.93)	−41 389.21	(2 638.65)	*time*		−20 679.28	(1 958.25)	−15 429.34	(1 350.31)
Census-india	−82 609.07	(2 276.73)	*error*		−212 839.72	(42 269.43)	*time*		−77 398.50	(10 989.94)	−68 548.53	(21 427.16)

The winner in each row is highlighted in blue. Results from IL and CIL that outperformed those from existing methods without ranking first place are highlighted in yellow. Numbers between parentheses correspond to standard deviations.
time: the algorithm was unable to complete a fold in a week.
error: the algorithm returned an error.

Table 5.8: 10-fold cross-validated predictive log-likelihood for various mixed datasets (rows) and algorithms (columns).

	KDE		MSPN		GLFM		IL		CIL [$\alpha = 1$]		CIL [$\alpha = 10$]	
Haberman	0.00	(0.00)	4.34	(1.49)	9.07	(0.52)	1.34	(0.03)	0.16	(0.02)	1.63	(0.07)
Iris	0.00	(0.00)	1.32	(0.27)	5.45	(0.35)	2.65	(1.16)	0.96	(0.08)	3.03	(0.43)
User-knowledge	0.00	(0.00)	1.99	(0.69)	16.56	(1.86)	1.64	(0.08)	0.71	(0.30)	1.16	(0.09)
Vertebral	0.01	(0.00)	2.85	(0.66)	35.12	(6.94)	14.68	(5.99)	10.30	(3.38)	52.43	(103.66)
Ecoli	*error*		11.07	(1.61)	30.65	(4.07)	3.92	(0.11)	0.34	(0.02)	1.87	(0.07)
Planning-relax	0.01	(0.00)	28.67	(5.80)	64.44	(2.71)	15.81	(0.17)	1.06	(0.03)	3.31	(0.08)
Thoracic-surgery	0.03	(0.00)	3.43	(1.01)	111.99	(5.13)	118.50	(1.56)	2.64	(0.69)	4.44	(0.63)
Vehicle	0.08	(0.04)	85.12	(6.24)	553.84	(144.17)	986.61	(15.43)	38.62	(9.83)	293.17	(170.78)
Thyroid	0.84	(0.07)	11.22	(0.99)	2409.59	(153.39)	5418.17	(626.46)	63.82	(38.75)	457.04	(116.14)
Parkinsons	0.01	(0.00)	77.63	(11.10)	123.75	(18.29)	98.30	(8.55)	0.44	(0.04)	1.70	(0.04)
Autos	*error*		22.87	(1.56)	72.73	(12.43)	221.05	(3.57)	0.51	(0.15)	4.53	(0.90)
Ionosphere	0.02	(0.00)	222.49	(18.85)	517.23	(45.77)	2880.54	(150.23)	67.45	(68.79)	226.98	(71.53)
Qsar-biodeg	0.17	(0.00)	491.29	(37.76)	3500.91	(118.78)	25 229.09	(1 098.92)	26.23	(8.98)	1108.75	(282.47)
Housing-prices	*error*		750.93	(99.15)	10 106.29	(1 525.88)	*time*		94.33	(32.56)	1149.53	(706.70)
Census-india	1.47	(0.04)	*error*		43 458.55	(2 819.02)	*time*		2948.52	(1 838.42)	23 575.06	(4 266.94)

The winner in each row is highlighted in blue. Results from IL and CIL that outperformed those from existing methods without ranking first place are highlighted in yellow. Numbers between parentheses correspond to standard deviations.
time: the algorithm was unable to complete a fold in a week.
error: the algorithm returned an error.

Table 5.9: 10-fold cross-validated execution times (in seconds) for various mixed datasets (rows) and algorithms (columns).

is a valid alternative to the incremental approach. It returns almost identical results for categorical data experiments, and displays only slightly worse performance for continuous and mixed data experiments. We believe that these differences are caused by the use of approximate MI estimation, and demonstrate that they can be reduced by increasing α.

Although restricting the structure search to an incremental process limits the space of possible models, our experiments demonstrate that this restriction still leads to competitive results. Furthermore, due to this restriction, our proposed method is able to work effectively with both low- and high-dimensional datasets. Other methods (e.g., GEAST) may perform better by considering a larger number of candidate models at each iteration, but their computational complexity makes them unfeasible when the number of variables is large. Additionally, and in contrast to current LTM methods, by taking advantage of the VB framework, we are able to provide a means of incorporating prior knowledge and produce superior evaluation of the generalization properties of a model given data.

There are various future research directions. First, inspired by factor analysis, our methods can be modified to work with Gaussian latent variables, where the number of factors can be analogously estimated to the cardinality of categorical latent variables. Second, we can extend parameter estimation by using more flexible variational families [Barber and Wiegerinck, 1999; Bishop et al., 1998] and nonconjugate priors [Wang and Blei, 2013], but at the cost of a more difficult variational optimization problem. Third, we can add more flexibility to the structure by replacing the arc addition operator with a VB version of SEM. This can allow us to efficiently add and remove arcs without relying on an incremental process (see Chapter 6). Finally, selecting the most probable latent forest may not be suitable when we are interested in quantifying our confidence in the forest structure or when we have a low number of data instances. For these cases, Bayesian estimation can also be introduced into the search process by averaging over the set of possible latent forests (i.e., by using Bayesian model averaging [Hoeting et al., 1999; Fragoso et al., 2018]). For this, we can define a Markov chain over the space of possible latent forests (given our search operators) and then do a random walk in this Markov chain.

Chapter 6

Learning Bayesian networks from incomplete mixed data

6.1 Introduction

Real-world data is often complex, large, and incomplete. In this setting, learning BNs that accurately capture the underlying probability distribution may enable us to better understand the data, estimate missing or corrupted values, and make predictions, diagnoses and explanations. To this end, BNs with latent variables have demonstrated their effectiveness in capturing the data generation process, as they are able to represent underlying data concepts by grouping similar observed variables.

There is an extensive literature on the identification of latent variables using BNs with homogeneous data, where all the domain variables are of the same type, that is, categorical or continuous. These works have considered from limited structures such as trees (see Chapter 3) and polytrees [Rodriguez-Sanchez et al., 2018], to general structures such as DAGs [Elidan and Friedman, 2005; Asbeh and Lerner, 2016a,b]. However, there still exists a lack of work when dealing with heterogeneous (i.e., mixed) data, which in fact is common in real-world applications. In this chapter, we propose a new approach for learning BNs with categorical latent variables that is suitable for mixed datasets. The proposed method hill-climbs the space of conditional linear Gaussian BNs with categorical latent variables using a series of latent operators and a VB version of the SEM algorithm. We demonstrate the effectiveness of our approach by solving unsupervised tasks, such as density estimation and missing data imputation.

This chapter includes the content of Rodriguez-Sanchez et al. [2021a]. In addition, all code, data, and results are available at https://github.com/ferjorosa/GLSL.

Chapter outline

In Section 6.2, we describe the VB-SEM algorithm. In Section 6.3, we present our method for learning conditional linear Gaussian BNs with categorical latent variables. Then, in

Algorithm 6.1: VB-SEM

Input : $\mathcal{D}$, $\mathcal{B}_0$, $\mathcal{R}_A$

1 **for** $t = 0, 1, \ldots$ **do**

  `/* Expectation step */`

2 $\mathcal{D}_t^* \leftarrow$ Complete latent data using variational inference with $\mathcal{B}_t$

  `/* Maximization step */`

3 $\mathcal{G}_{t+1} \leftarrow \text{GS}(\mathcal{D}_t^*, \mathcal{R}_A)$

4 $\boldsymbol{\theta}_{t+1} \leftarrow \text{VB-EM}(\mathcal{D}, \mathcal{G}_{t+1})$

5 $\mathcal{B}_{t+1} \leftarrow \{\mathcal{G}_{t+1}, \boldsymbol{\theta}_{t+1}\}$

6 **if** $score(\mathcal{B}_{t+1} : \mathcal{D}) \leq score(\mathcal{B}_t : \mathcal{D})$ **then**

7 **break** `/* Stop the loop */`

Output: The resulting model $\mathcal{B}_t$

Section 6.4, we explore the performance of our method for the tasks of density estimation and missing data imputation. Finally, we draw our conclusions and propose future research lines in Section 6.5.

6.2 VB-SEM

In this section, we introduce a variant of SEM that it is framed within the VB framework to ensure its applicability to mixed data. We refer to this method as VB-SEM. The introduction of the VB framework arises from the need to apply probabilistic inference in the expectation step of SEM. Since exact inference is NP-hard for conditional linear Gaussian BNs even when restricting the BN structure to a polytree [Lerner and Parr, 2001], approximate inference is required. We chose VI, and more specifically the VB framework, due to its speed and easy integration with BNs through the VMP framework [Winn and Bishop, 2005].

Algorithm 6.1 describes VB-SEM, which performs as follows. It receives a dataset $\mathcal{D}$, an initial model $\mathcal{B}_0$, and a set of arc restrictions $\mathcal{R}_A$ that may limit the structures produced by the algorithm (e.g., we may not be interested in adding/removing certain graph arcs). Line 1 is the main loop of the algorithm. Line 2 describes the expectation step, in which the current model $\mathcal{B}_t$ estimates the values of its latent variables and generates a completed dataset $\mathcal{D}_t^*$. Lines 3-7 describe the maximization step. In line 3, the completed dataset is used to learn the structure $\mathcal{G}_{t+1}$ of the new model $\mathcal{B}_{t+1}$. For simplicity, we use an HC method with AI, AE, and AR operators. Once the structure has been learned, the parameters $\boldsymbol{\theta}_{t+1}$ of the new model are estimated using the observed data $\mathcal{D}$ (line 4). To this purpose, the VB-EM algorithm is employed. Finally, the resulting model $\mathcal{B}_{t+1}$ is compared with the current best model $\mathcal{B}_t$, and the model with greater score with respect to $\mathcal{D}$ is selected.

VB-SEM estimates the parameters and score of the new model with respect to the observed data (i.e., the observed score) rather than with the completed data (i.e., the expected score) to improve its model fitting with respect to the observed data. Despite the desirable properties of SEM, Equation (2.9) offers no guarantee that the model selected at the end of the

search is near the optimum of the observed score [Koller and Friedman, 2009]. For example, if an HC method is used inside SEM, only the first structure change (e.g., the introduction of a new arc) guarantees an improvement with respect to the observed score. Benjumeda et al. [2019] address this problem by estimating the observed score after each structure change at the expense of the score decomposition. Their approach applies to categorical data, where exact inference can be made tractable by bounding the treewidth of the BN. We instead consider the approximation of estimating the expected score when learning the BN structure, and the observed score when estimating the parameters. The scoring function that is traditionally used within the VB framework is the ELBO. However, when applied to models with multiple categorical variables, it is more appropriate to use a penalized version such as the p-ELBO (see Equation 5.1).

Finally, a key aspect for the performance of VB-SEM is prior specification. When expert information is available, prior parameters can be selected to best reflect the expert knowledge. When this information is unavailable, we propose to use the following strategy: (i) observed variables assume empirical Bayes [Casella, 1985] priors (with maximum likelihood estimates as the values of prior parameters), and (ii) categorical latent variables assume a symmetric Dirichlet prior with a total concentration of 1.

6.3 Greedy latent structure learner

In this section, we propose a greedy search method for learning conditional linear Gaussian BNs with categorical latent variables. Our method iteratively explores this space of models using five latent operators and the VB-SEM algorithm. Latent operators are tasked with introducing latent variables, removing latent variables, and changing the cardinality of latent variables. Each application of the latent operators produces a candidate model whose structure is subsequently refined using a local run of VB-SEM, which introduces, removes, or reverses BN arcs. Both aspects of the structure search (i.e., latent variables and BN arcs) are separated to reduce the computational cost of the algorithm.

We start with a description of the latent operators in Section 6.3.1. Then, in Section 6.3.2, we present a local version of the VB-SEM algorithm. Finally, in Section 6.3.3, we discuss the search procedure.

6.3.1 Latent operators

- **Latent variable introduction** (LI) generates a new model by introducing a new categorical latent variable H as the parent of two variables (observed or latent) that currently have no parents. The cardinality of H is set to two. We only consider pairs of variables to reduce the computational complexity of this operator. This restriction will be compensated by the local VB-SEM algorithm.

- **Conditional latent variable introduction** (CLI) produces a new model by introducing a new latent variable H' as the parent of two variables (observed or latent) that

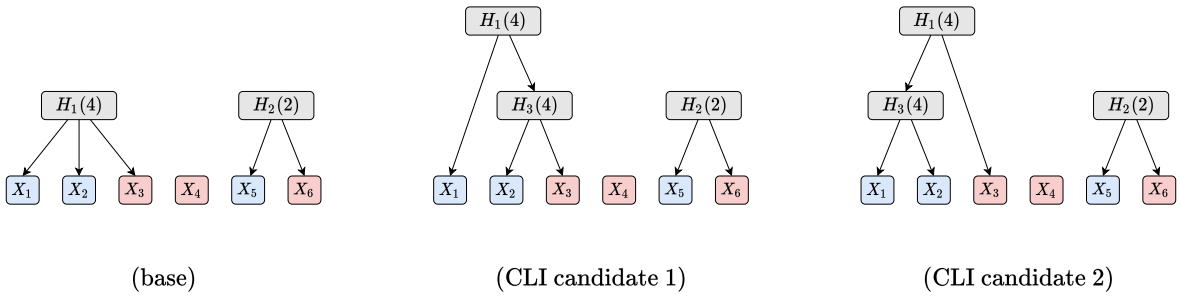


Figure 6.1: Example application of the CLI operator. The base model contains two latent variables (i.e., H_1 and H_2). However only H_1 has more than two children. As a result, two candidate models are generated by introducing a new latent variable H_3 as the child of H_1. Variable colors and parentheses have the same meaning as in Figure 3.1.

currently have a latent variable H as their parent. H' becomes the new child of H. The cardinality of H' is set equal to the cardinality of H. This operator is not applicable if H only has two children. Figure 6.1 provides an example application of this operator.

- **Latent variable elimination** (LE) generates a new model by removing a latent variable and its associated arcs.
- **Cardinality increase** (CI) creates a new model by increasing the cardinality of a categorical latent variable H by one. This operator is not applicable if the variable has a cardinality equal to k_{max}, which is established by the user.
- **Cardinality decrease** (CD) produces a new model by decreasing the cardinality of a categorical latent variable H by one. This operator is not applicable if H already has a cardinality of two.

6.3.2 Local VB-SEM

Repeatedly evaluating candidate models produced by the latent operators can become prohibitive when each evaluation involves a full execution of VB-SEM. Therefore, we propose replacing VB-SEM with a local version of this method, which only considers the section of the model that is affected by the latent operator while the rest of the model is kept unchanged. We refer to this method as local VB-SEM.

In the local VB-SEM algorithm, we allow adding, removing, or reversing arcs that contain any of the variables considered by the latent operator. Once the structure has been learned, the local VB-EM algorithm (see Section 5.2.4) estimates the parameters of those variables belonging to the MBs of: (i) variables initially selected by the latent operator, and (ii) variables affected by a structure change (e.g., a new incoming arc). For any variable in a BN, its MB consists of the set of all its parents, children, and spouses (parents of children) in the network. Therefore, the pseudocode of local VB-SEM is identical to the pseudocode of VB-SEM (Algorithm 6.1), except that the structure learning process is always restrained

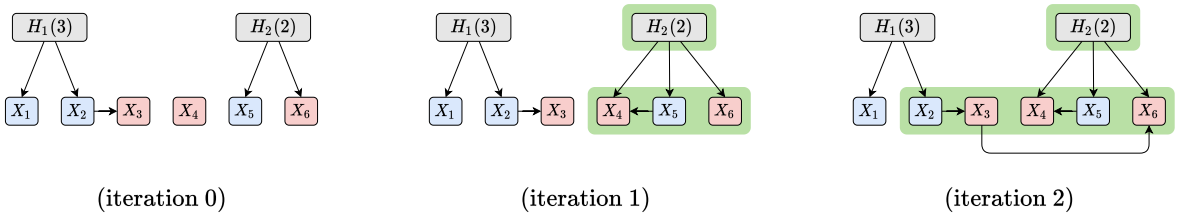


Figure 6.2: Example execution of the local VB-SEM algorithm with two iterations. The initial model is a candidate of the latent introduction operator, which has introduced H_2. It introduces two arcs in its first iteration ($H_2 \rightarrow X_4$ and $X_5 \rightarrow X_4$) and one arc in its second iteration ($X_3 \rightarrow X_6$). Variables highlighted in green have their parameters estimated by the local VB-EM algorithm. Variable colors and parentheses have the same meaning as in Figure 3.1.

by a specific set of arc restrictions, and the parameter estimation process is done with the local VB-EM algorithm.

One iteration of local VB-SEM is computationally much cheaper than one iteration of VB-SEM because it considers fewer structure changes and it updates fewer model parameters. This also implies that a run of local VB-SEM usually requires fewer steps to converge than a run of VB-SEM. The computational complexity of local VB-SEM is upper-bounded by the computational complexity of VB-SEM.

Figure 6.2 provides an example execution of the local VB-SEM algorithm. The initial model is a candidate of the LI operator, which has introduced a new latent variable H_2 as the parent of the observed variables X_5 and X_6. In its first iteration, local VB-SEM introduces two new arcs: $H_2 \rightarrow X_4$ and $X_5 \rightarrow X_4$. After the structure learning process, the local VB-EM algorithm estimates the parameters of H_2, X_4, X_5, and X_6 (highlighted in green in the figure), all of which belong to the MBs of the variables initially selected by the latent operator. In its second and last iteration, local VB-SEM introduces a new arc from X_3 to X_6 and estimates the parameters of H_2, X_2, X_3, X_4, X_5, and X_6. While X_2 does not belong to the MB of the variables initially considered by the latent operator, it does belong to the MB of a variable affected by a structure change (i.e., X_3). This is not the case of H_1 and X_1, whose parameters are kept unchanged.

6.3.3 Search procedure

The search starts with an initial model $\mathcal{B}_0$ established by the user. Given this model, let $\mathbf{W}$ denote the set of variables that are considered by the LI operator (i.e., the set of variables that currently have no parents) and let $\mathbf{H}_C$ denote the set of latent variables that are considered by the CLI operator (i.e., the set of latent variables that currently have three or more children). Once the initial model has been established, the search method traverses the space of models by iteratively applying the latent operators. Each application of a latent operator produces a set of candidate models that are evaluated using the local VB-SEM algorithm with p-ELBO as the scoring function (see Equation (5.1)). Once all candidate models have been evaluated, the highest scoring model $\mathcal{B}'$ is selected. If its score is higher than that of $\mathcal{B}$, the new model

Algorithm 6.2: Greedy latent structure learner (GLSL)

Input : $\mathcal{D}$, $\mathcal{B}_0$, k_{max}, $\mathcal{R}_A$

$\mathcal{B} \leftarrow \mathcal{B}_0$
Let $\mathbf{W}$ be the set of variables that currently have no parent in $\mathcal{B}$
Let $\mathbf{H}_C$ be the set of latent variables with 3 or more children in $\mathcal{B}$
while *True* **do**
    $\mathbf{B}_{LI} \leftarrow \text{LI}(\mathcal{D}, \mathcal{B}, \mathbf{W}, \mathcal{R}_A)$
    $\mathbf{B}_{CLI} \leftarrow \text{CLI}(\mathcal{D}, \mathcal{B}, \mathbf{H}_C, \mathcal{R}_A)$
    $\mathbf{B}_{LE} \leftarrow \text{LE}(\mathcal{D}, \mathcal{B}, \mathcal{R}_A)$
    $\mathbf{B}_{CI} \leftarrow \text{CI}(\mathcal{D}, \mathcal{B}, k_{max}, \mathcal{R}_A)$
    $\mathbf{B}_{CD} \leftarrow \text{CD}(\mathcal{D}, \mathcal{B}, \mathcal{R}_A)$
    $\mathcal{B}' \leftarrow$ highest scoring model in $\{\mathbf{B}_{LI}, \mathbf{B}_{CLI}, \mathbf{B}_{LE}, \mathbf{B}_{CI}, \mathbf{B}_{CD}\}$
    if $score(\mathcal{B}' : \mathcal{D}) > score(\mathcal{B} : \mathcal{D})$ **then**
        $\mathcal{B} \leftarrow \mathcal{B}'$
        Update $\mathbf{W}$ and $\mathbf{H}_C$ with respect to $\mathcal{B}$
    else
        break `/* Stop the loop */`
`/* Model refinement */`
$\mathcal{B}' \rightarrow \text{VB-SEM}(\mathcal{D}, \mathcal{B}, \mathcal{R}_A)$

Output: The resulting model $\mathcal{B}'$

takes its place and the process is repeated. Once the search stops, the resulting model is refined using a full run of VB-SEM. This search procedure is formally defined in Algorithm 6.2.

Figure 6.3 provides an example execution of the GLSL algorithm. It starts with an empty graph with three categorical variables $\{X_1, X_2, X_5\}$ and three continuous variables $\{X_3, X_4, X_6\}$. In its first iteration, the LI operator is selected and a new latent variable H_1 is introduced as the parent of X_3 and X_4. In addition, the local run of VB-SEM results in other three arcs: $H_1 \rightarrow X_1$, $H_1 \rightarrow X_2$, and $X_5 \rightarrow X_4$. In its second iteration, the CI operator is selected, and the cardinality of H_1 is increased. No structure changes are introduced by the local VB-SEM. In its third iteration, the CLI operator is selected. A new latent variable H_2 is introduced as the child of H_1, and as the parent of X_3 and X_4. In the fourth and last iteration, the CD operator is selected, and the cardinality of H_2 is decreased. In addition, the local run of VB-SEM introduces a new arc from X_1 to X_3. Finally, the refinement run of VB-SEM introduces a new arc from X_5 to X_6 and the model is returned.

6.3.3.1 Complexity analysis

The following auxiliary variables are considered for the complexity analysis of GLSL:

- $\mathbf{W} \rightarrow$ current set of variables with no parents.
- $\mathbf{H} \rightarrow$ current set of latent variables.

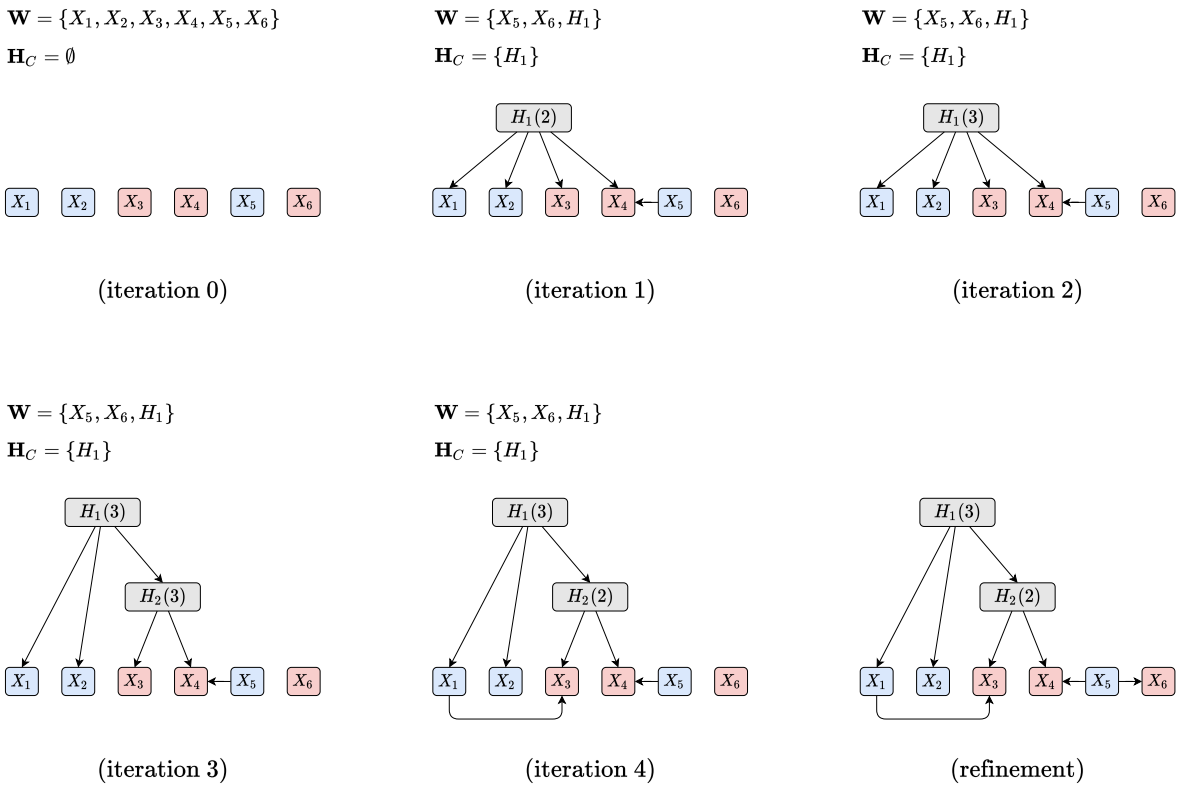


Figure 6.3: Example execution of the GLSL algorithm with four iterations. It introduces a latent variable H_1 in the first iteration, it increases the cardinality of H_1 in the second iteration, it introduces a new conditional latent variable H_2 in the third iteration, and it decreases the cardinality of H_2 in the fourth iteration. Each iteration is followed by a local run of the VB-SEM algorithm, which is tasked with introducing, removing, or reversing BN arcs. Once the iteration process of GLSL finishes, a full run of VB-SEM refines the structure, introducing a new arc from X_5 to X_6. Variable colors and parentheses have the same meaning as in Figure 3.1.

- $\mathbf{H}_C \rightarrow$ current set of latent variables with three or more children.
- $\mathbf{Ch}_{Ci} \rightarrow$ current set of children variables of the latent variable $H_i \in \mathbf{H}_C$.

At each iteration of the GLSL algorithm, the CI, CD, and LE operators evaluate a total of $O(|\mathbf{H}|)$ candidate models each. In turn, the LI operator evaluates a total of $O((|\mathbf{W}|^2 - |\mathbf{W}|)/2)$ candidate models. Estimating the number of candidate models that are evaluated by the CLI operator is more complex than for LI. This is because CLI depends on the current number of latent variables and their respective number of children variables. The CLI operator evaluates a total of $O(\sum_i(|\mathbf{Ch}_{Ci}|^2 - |\mathbf{Ch}_{Ci}|)/2)$ candidate models. Therefore, the total number of candidate models evaluated at each iteration of GLSL is $O(3|\mathbf{H}|+(|\mathbf{W}|^2-|\mathbf{W}|)/2+\sum_i(|\mathbf{Ch}_{Ci}|^2-|\mathbf{Ch}_{Ci}|)/2)$. Note that each evaluation requires running the local VB-SEM algorithm.

6.4 Experiments

In this section, we evaluate the performance of GLSL at solving two unsupervised tasks. First, we evaluate GLSL at learning an LVM that is able to generalize and fit the data well (Section 6.4.1). Second, we evaluate GLSL at imputing missing data (Section 6.4.2). We considered an empty graph as the initial structure for GLSL in all of the experiments. In addition, GLSL did not consider any additional arc restrictions except for those intrinsic to conditional linear Gaussian BNs. GLSL was implemented in Java 8 using the AMIDST toolbox [Masegosa et al., 2019]. Experiments were conducted on a computer with an Intel Core i7-6700K CPU at 4.00 GHz with 64GB RAM, running Windows 10 Enterprise.

6.4.1 Density estimation

In this section, we evaluate the performance of GLSL in terms of data fitting and computational complexity. To this end, we conducted comparative studies using categorical (Section 6.4.1.1), continuous (Section 6.4.1.2) and mixed (Section 6.4.1.3) real-world data. In these experiments GLSL was compared to the following methods:

- **EAST**. Expansion, adjustment and simplification until termination algorithm [Chen et al., 2012].
- **GEAST**. Gaussian version of the EAST algorithm [Poon et al., 2013].
- **BI**. Bridged islands algorithm [Liu et al., 2015].
- **KDE**. Multivariate kernel density estimator [Li and Racine, 2003].
- **MSPN**. Mixed sum-product networks [Molina et al., 2018].
- **GLFM**. Gaussian latent feature model [Valera et al., 2020].
- **IL**. Incremental learner (see Section 5.2.2).
- **CIL**. Constrained incremental learner with $\alpha = 10$ (see Section 5.2.3).

As a performance measure, we used the 10-fold CVPLL. That is, we divided each dataset into 10 equal-sized folds, trained a model on 9 of them, and computed the predictive LL on the remaining fold.

6.4.1.1 Comparative study on categorical data

To conduct the study, we used 15 real-world datasets of various dimensionalities (m) and sample sizes (N) that were taken from the UCI repository and the current literature. Table 5.1 shows the basic properties of each dataset. None of these datasets provided expert knowledge to set Bayesian priors. For this reason, we set the prior parameters of the IL, CIL, and GLSL models using an empirical Bayes approach.

Tables 6.1 and 6.2 display the average (and standard deviation) results for the CVPLL and execution time, respectively. The maximum time allowed per fold was one week. The winner in each row is highlighted in blue. In this study, GLSL was compared with EAST, BI, MSPN, GLFM, IL, and CIL [$\alpha = 10$]. Although KDE was also initially considered, its results were not included due to space limitations. In addition, KDE was unable to outperform any of the considered methods and returned an error in three of the datasets.

Table 6.1 indicates that GLSL returned the best results in terms of model quality, achieving the highest CVPLL score in 6 out of 15 datasets. IL and CIL were close behind, achieving the highest score in 5 and 4 datasets, respectively. BI returned competitive results, and while it only achieved the highest score in 3 out of 15 datasets, it was the best perfoming method for high-dimensional data (i.e., News-100 and Webkb-336).

Table 6.2 shows that, even though GLSL was the best performing method in terms of CVPLL, it presented computational issues when dealing with high-dimensional data. GLSL was unable to complete its execution in three of the experiments. Similarly, both IL and EAST were unable to complete its execution in one of the experiments. In terms of speed, MSPN was the fastest method in all of the considered datasets.

6.4.1.2 Comparative study on continuous data

As with the previous comparative study, we considered 15 real-world continuous datasets from the UCI repository and the literature. Table 5.4 shows the basic properties of each dataset. We also used CVPLL as the evaluation measure and allowed a maximum execution time of one week per fold. In addition, given the absence of expert knowledge, we used an empirical Bayes approach to set the prior parameters of the IL, CIL, and GLSL models. CVPLLs and execution times of each method are displayed in Tables 6.3 and 6.4, respectively. In this study, GLSL was compared with GEAST, KDE, MSPN, GLFM, IL, and CIL [$\alpha = 10$].

Table 6.3 indicates that, in terms of model quality, GEAST was the best performing method, returning the highest CVPLL score in 6 out of 15 datasets. GEAST was followed by GLSL, which achieved top performance in 5 out of 15 datasets. GLSL considerably improved the results of IL and CIL, achieving a better performance in all of the datasets in which it was able to complete its execution. Each of the KDE, MSPN, IL, and CIL methods obtained the highest score in 1 out of 15 datasets. Similar to the categorical comparative study, GLFM was unable to outperform the rest of methods in any of the considered datasets.

Table 6.4 shows that, even though GEAST and GLSL were the best performing methods in terms of CVPLL, they both showed computational issues with high-dimensional datasets. Both of them were unable to complete their executions in 3 of the experiments. Similarly, IL was unable to complete its execution in one of the experiments. In terms of speed, KDE was the fastest method in all of the considered datasets.

	EAST		BI		MSPN		GLFM		IL		CIL [$\alpha = 10$]		GLSL	
HIV-test	−1485.89	(4 484.39)	−85.61	(59.44)	−104.84	(100.70)	−93.68	(147.46)	−84.22	(51.96)	−84.22	(51.96)	−84.48	(52.02)
Hayes-roth	−89.72	(5.91)	−89.98	(5.68)	−88.89	(7.87)	−110.67	(12.99)	−89.09	(5.71)	−89.09	(5.71)	−87.16	(7.67)
Balance-scale	−492.48	(16.81)	−491.67	(16.57)	−493.10	(45.40)	−505.65	(34.91)	−487.16	(15.78)	−487.16	(15.78)	−487.16	(15.78)
Car-evaluation	−12 095.26	(9 825.73)	−1610.46	(251.41)	−1690.90	(241.26)	−1728.79	(128.60)	−1503.81	(185.32)	−1503.81	(185.32)	−1693.70	(82.27)
Nursery	−264 750.76	(228 444.63)	−20 307.37	(2 348.33)	−15 288.31	(1 017.49)	−14 277.25	(956.67)	−13 626.52	(592.82)	−13 649.92	(576.75)	−18 286.73	(961.62)
Breast-cancer	−321.55	(91.26)	−258.56	(24.74)	−310.73	(39.61)	−415.29	(66.56)	−261.36	(24.95)	−261.36	(24.95)	−259.66	(23.86)
Web-phishing	−1018.54	(19.76)	−1010.19	(20.93)	−1013.53	(18.21)	−1758.30	(126.83)	−1074.22	(20.75)	−1074.22	(20.75)	−992.44	(22.47)
Solar-flare	−973.54	(187.98)	−949.32	(131.62)	−1106.10	(131.89)	−1433.10	(147.16)	−1026.94	(131.25)	−1026.94	(131.25)	−941.57	(126.16)
Zoo	−122.54	(190.95)	−71.86	(20.60)	−108.94	(21.31)	−214.64	(50.84)	−93.05	(20.94)	−93.05	(20.94)	−71.69	(21.47)
Vote	−216.01	(79.54)	−176.49	(21.45)	−174.27	(19.84)	−440.29	(99.99)	−193.73	(24.38)	−193.73	(24.38)	−177.23	(21.28)
Spect-heart	−288.35	(46.97)	−289.78	(46.94)	−316.32	(61.76)	−459.13	(95.03)	−299.42	(41.52)	−299.42	(41.52)	−291.22	(45.81)
Alarm	−1179.96	(536.68)	−1352.71	(555.29)	−4142.37	(1 027.10)	−5685.95	(744.38)	−1461.45	(564.23)	−1452.13	(535.00)	−1131.08	(348.30)
Coil-42	−30 597.77	(45 573.34)	−6978.14	(1 501.16)	−6879.65	(860.33)	−8766.60	(1 107.85)	−6269.04	(918.50)	−6269.04	(918.50)	*time*	
News-100	*time*		−25 343.09	(6 113.94)	−26 768.92	(6 971.29)	−27 434.12	(7 383.22)	*time*		−26 325.91	(6 576.45)	*time*	
Webkb-336	*time*		−8430.17	(1 681.82)	−9211.57	(1 733.69)	−10 721.51	(2 309.92)	*time*		−9439.57	(1 990.87)	*time*	

The winner in each row is highlighted in blue. Numbers between parentheses correspond to standard deviations.
time: the algorithm was unable to complete a fold in a week.

Table 6.1: 10-fold cross-validated predictive log-likelihood for various categorical datasets (rows) and algorithms (columns).

	EAST		BI		MSPN		GLFM		IL		CIL [$\alpha = 10$]		GLSL	
HIV-test	0.66	(0.13)	0.57	(0.14)	0.30	(0.11)	8.57	(0.14)	1.22	(0.02)	1.00	(0.05)	12.96	(1.82)
Hayes-roth	2.15	(0.07)	1.71	(0.09)	1.50	(0.11)	6.05	(0.15)	1.45	(0.07)	0.99	(0.09)	108.32	(28.91)
Balance-scale	7.94	(0.12)	5.63	(0.12)	5.28	(1.66)	32.05	(1.12)	3.51	(0.09)	4.19	(0.18)	144.21	(21.03)
Car-evaluation	243.66	(52.14)	38.83	(21.99)	4.04	(1.12)	178.14	(10.80)	32.09	(0.23)	27.85	(0.62)	6554.26	(3 002.44)
Nursery	6977.54	(2 471.52)	496.93	(280.72)	4.12	(0.70)	1238.84	(153.49)	800.37	(20.26)	449.10	(21.87)	49 516.87	(12 955.71)
Breast-cancer	44.63	(11.41)	8.50	(1.28)	3.86	(0.51)	31.06	(0.44)	30.86	(1.98)	23.25	(2.03)	1518.69	(394.47)
Web-phishing	327.40	(67.68)	46.00	(6.01)	9.00	(1.51)	183.10	(39.08)	125.66	(5.47)	54.12	(1.40)	7775.28	(682.10)
Solar-flare	349.84	(59.14)	50.62	(29.19)	8.67	(0.78)	125.47	(4.63)	335.51	(30.90)	99.80	(1.68)	87 113.11	(31 967.18)
Zoo	50.90	(9.53)	9.54	(1.06)	4.29	(0.65)	31.44	(12.18)	49.32	(1.70)	8.32	(0.44)	3160.18	(754.81)
Vote	96.00	(15.19)	13.77	(3.40)	3.61	(0.38)	30.76	(2.76)	64.31	(1.21)	12.71	(0.25)	2014.72	(172.72)
Spect-heart	328.52	(50.97)	12.84	(2.08)	8.46	(1.18)	42.00	(5.02)	237.04	(4.15)	27.80	(0.44)	3883.40	(293.50)
Alarm	4592.03	(746.44)	67.42	(3.72)	75.98	(9.73)	482.41	(170.77)	6209.85	(55.82)	293.53	(15.44)	393 983.04	(20 122.14)
Coil-42	27 531.16	(3 032.85)	254.00	(14.54)	33.55	(3.00)	1360.87	(7.63)	118 844.19	(2 165.96)	2446.87	(116.52)	*time*	
News-100	*time*		2537.71	(150.19)	142.22	(6.91)	9893.96	(1 685.00)	*time*		38 156.47	(2 320.58)	*time*	
Webkb-336	*time*		1244.90	(36.40)	2139.86	(344.65)	2846.41	(13.53)	*time*		22 448.31	(926.31)	*time*	

The winner in each row is highlighted in blue. Numbers between parentheses correspond to standard deviations.
time: the algorithm was unable to complete a fold in a week.

Table 6.2: 10-fold cross-validated execution times (in seconds) for various categorical datasets (rows) and algorithms (columns).

	GEAST		KDE		MSPN		GLFM		IL		CIL [$\alpha = 10$]		GLSL	
Real-state	−246.58	(18.87)	−295.02	(11.79)	−539.61	(465.71)	−872.50	(92.10)	−380.16	(15.29)	−380.16	(15.29)	−265.34	(19.82)
Buddymove	−671.54	(95.00)	−663.69	(85.19)	−1258.76	(410.91)	−726.64	(97.73)	−662.24	(103.50)	−662.24	(103.50)	−651.45	(105.12)
Qsar-fish	−410.86	(38.37)	−561.57	(30.05)	−1223.66	(750.94)	−1885.14	(869.58)	−467.64	(42.30)	−377.10	(52.04)	−260.94	(115.68)
Qsar-aqua	−937.12	(111.26)	−1060.84	(101.30)	−718.65	(119.72)	−2106.64	(799.14)	−905.96	(222.15)	−879.76	(231.69)	−798.64	(162.11)
ILPD	−1567.17	(165.57)	−1834.17	(280.75)	−2125.03	(739.63)	−2024.70	(177.32)	−1443.60	(231.19)	−1448.62	(233.46)	−1412.63	(327.39)
Alcohol	−553.53	(75.94)	−611.08	(78.82)	−1544.25	(466.68)	−1483.47	(386.36)	−493.29	(63.69)	−498.38	(67.62)	−457.51	(64.58)
Travel-reviews	311.82	(32.71)	154.67	(22.73)	−953.40	(1 055.57)	−3595.92	(1 125.14)	81.12	(32.24)	63.92	(32.10)	216.93	(71.17)
Wine-quality	−1321.11	(277.53)	−2211.00	(196.45)	−8758.61	(2 567.19)	−16 505.01	(2 596.44)	−3013.58	(230.65)	−3083.70	(257.49)	−2381.29	(238.44)
Wine	−360.01	(68.19)	−365.86	(58.82)	−1093.08	(293.06)	−704.77	(379.01)	−400.20	(105.59)	−398.06	(92.73)	−373.29	(90.59)
Leaf	902.82	(143.46)	694.85	(42.35)	−2412.34	(687.03)	−1519.57	(1 518.66)	603.59	(161.66)	619.96	(142.93)	704.04	(141.71)
NBA	−399.21	(30.68)	−694.60	(16.54)	−2537.31	(243.05)	−3163.04	(281.89)	−660.75	(27.87)	−681.09	(28.92)	398.71	(56.83)
WDBC	1168.41	(180.00)	360.47	(67.67)	−4383.39	(1 999.32)	−9966.35	(1 194.90)	721.22	(219.05)	615.81	(215.84)	321.25	(250.42)
Waveform	*time*		−32 643.86	(56.89)	−30 816.14	(207.60)	−40 771.48	(563.01)	−30 370.28	(59.91)	−30 580.09	(152.53)	*time*	
100-plants	*time*		31 175.92	(777.24)	−13 220.36	(13 736.14)	−15 808.74	(7 583.75)	28 239.13	(2 198.29)	28 197.25	(2 206.23)	*time*	
Geo-music	*time*		−10 029.05	(294.76)	*time*		−29 926.40	(1 059.27)	*time*		−9287.31	(365.14)	*time*	

The winner in each row is highlighted in blue.
time: the algorithm was unable to complete a fold in a week.

Table 6.3: 10-fold cross-validated predictive log-likelihood for various continuous datasets (rows) and algorithms (columns).

	GEAST		KDE		MSPN		GLFM		IL		CIL [$\alpha = 10$]		GLSL	
Real-state	302.12	(129.36)	0.01	(0.00)	4.66	(0.86)	29.88	(2.74)	2.56	(0.33)	2.18	(0.29)	4922.89	(4 432.58)
Buddymove	213.78	(112.87)	0.00	(0.00)	4.48	(1.03)	0.03	(0.01)	2.80	(0.09)	2.54	(0.19)	87.85	(10.94)
Qsar-fish	1911.91	(651.32)	0.04	(0.01)	13.36	(3.03)	124.79	(5.31)	28.92	(2.05)	29.11	(5.97)	3169.77	(3 160.07)
Qsar-aqua	2292.40	(515.20)	0.02	(0.00)	11.21	(1.97)	87.92	(2.49)	39.53	(6.29)	34.96	(7.88)	3116.11	(1 189.20)
ILPD	1757.13	(490.96)	0.02	(0.00)	15.89	(2.50)	0.09	(0.02)	43.16	(2.12)	29.00	(3.09)	33 301.14	(18 769.68)
Alcohol	316.98	(68.85)	0.00	(0.00)	5.36	(1.43)	11.37	(0.82)	6.46	(1.16)	3.13	(0.34)	646.10	(63.37)
Travel-reviews	4180.70	(362.00)	0.08	(0.02)	20.19	(4.47)	214.64	(45.86)	133.72	(12.18)	51.54	(9.17)	6215.27	(2 923.33)
Wine-quality	169 021.63	(34 671.99)	1.68	(0.27)	91.95	(9.55)	1160.05	(67.81)	1366.39	(120.05)	318.31	(143.96)	220 342.97	(107 568.86)
Wine	795.98	(145.86)	0.01	(0.00)	13.89	(2.21)	29.55	(1.36)	21.59	(1.60)	4.40	(0.52)	3597.89	(2 670.71)
Leaf	3941.66	(420.50)	0.02	(0.00)	9.76	(0.74)	85.35	(12.50)	61.19	(4.15)	12.77	(2.44)	1393.00	(176.48)
NBA	13 608.33	(835.63)	0.03	(0.00)	59.87	(13.16)	210.85	(0.79)	257.69	(9.89)	49.60	(12.67)	49 550.76	(30 447.24)
WDBC	50 784.86	(3 315.22)	0.08	(0.02)	52.60	(1.43)	776.45	(10.12)	2436.78	(26.62)	57.90	(4.24)	152 898.31	(36 975.97)
Waveform	*time*		3.31	(0.14)	17.72	(3.90)	1662.30	(186.18)	45 831.88	(1 964.74)	1314.81	(244.66)	*time*	
100-plants	*time*		0.68	(0.02)	604.52	(19.70)	8848.11	(335.22)	6936.40	(1 244.92)	58.20	(7.41)	*time*	
Geo-music	*time*		0.26	(0.01)	*time*		7465.81	(3 575.05)	*time*		1849.98	(751.94)	*time*	

The winner in each row is highlighted in blue.
time: the algorithm was unable to complete a fold in a week.

Table 6.4: 10-fold cross-validated execution times (in seconds) for various continuous datasets (rows) and algorithms (columns).

6.4.1.3 Comparative study on mixed data

As with previous comparative studies, we ran experiments on 15 real-world datasets taken from the UCI repository and the current literature. Table 5.7 shows the basic properties of each dataset. We also used CVPLL as the evaluation measure and allowed a maximum execution time of one week per fold. In addition, given the absence of expert knowledge, we used an empirical Bayes approach to set the prior parameters of the IL, CIL, and GLSL models. CVPLLs and execution times of each method are displayed in Tables 6.5 and 6.6, respectively. In this study, GLSL was compared with KDE, MSPN and GLFM, IL, and CIL $[\alpha = 10]$.

Table 6.5 illustrates that, in terms of model quality, GLSL was the best performing method, returning the highest CVPLL score in 10 out of 15 datasets. IL and CIL were behind with 2 and 3 datasets, respectively. The rest of methods were unable to achieve the highest score in any of the datasets, except KDE, which outperformed all of the methods in 1 dataset.

Table 6.6 illustrates that, similar to the continuous case, KDE was the fastest method, ranking first in 12 out 15 datasets. It was followed by CIL, which ranked first in 2 of the datasets, and MSPN, which ranked first in 1 of the datasets. Both IL and GLSL had computational issues with high-dimensional datasets. However, GLSL was considerably much slower than IL.

6.4.2 Missing data imputation

In this section, we evaluate the performance of GLSL at imputing missing data, where we assume that the data is missing completely at random. To this end, we conducted comparative studies with categorical (Section 6.4.2.1), continuous (Section 6.4.2.2) and mixed (Section 6.4.2.3) real-world data. For each dataset, we generated 5 different incomplete datasets, removing completely at random a percentage of the data ranging from a 10% deletion to a 50% deletion. In these experiments, GLSL was compared to the following methods (all of which can handle categorical, continuous, and mixed data):

- **Mean imputation**. Baseline algorithm that imputes the mean of each continuous variable and the mode of each categorical variable that are missing in the data.
- **MICE**. Multiple imputation by chained equations [Azur et al., 2011], which is an iterative method that performs a series of supervised regression models, in which missing data is modelled conditionally on the other variables in the data. We used the implementation provided by the Python library scikit-learn (https://scikit-learn.org/).
- **GLFM**. Gaussian latent feature model [Valera et al., 2020].
- **HI-VAE**. Heterogeneous incomplete variational autoencoder [Nazabal et al., 2020], which provides a general framework for variational autoencoders that effectively incorporates incomplete data and handles both categorical and continuous variables. We

	KDE		MSPN		GLFM		IL		CIL [$\alpha = 10$]		GLSL	
Haberman	−319.16	(38.41)	−1012.93	(379.19)	−376.13	(44.48)	−278.13	(35.51)	−278.13	(35.51)	−299.88	(38.67)
Iris	−69.51	(6.97)	−194.20	(124.36)	−406.22	(339.04)	−51.10	(12.05)	−51.11	(12.05)	−33.60	(13.85)
User-knowledge	−29.03	(20.15)	−194.36	(188.05)	−292.22	(125.60)	−20.15	(22.66)	−20.15	(22.66)	−1.71	(18.00)
Vertebral	−798.78	(72.53)	−610.13	(237.69)	−1192.51	(444.36)	−763.81	(68.16)	−763.84	(68.14)	−606.32	(82.31)
Ecoli	*error*		−180.73	(190.28)	−592.32	(1 205.38)	12.26	(125.52)	12.24	(125.59)	169.63	(70.39)
Planning-relax	−94.31	(14.73)	−1087.97	(307.52)	−617.46	(194.49)	−44.72	(22.26)	−44.72	(22.26)	28.54	(28.49)
Thoracic-surgery	−750.57	(5.73)	−671.51	(52.22)	−1726.67	(702.16)	−553.39	(14.57)	−566.58	(17.40)	−513.54	(18.00)
Vehicle	−5469.10	(122.74)	−8801.35	(3 295.42)	−20 567.24	(1 500.78)	−5245.73	(135.66)	−5309.80	(150.54)	−4682.75	(91.48)
Thyroid	−7151.07	(97.26)	−5943.21	(119.81)	−20 475.09	(2 517.29)	−6047.79	(192.81)	−6680.76	(301.05)	−5786.55	(89.43)
Parkinsons	562.01	(66.83)	−981.89	(933.66)	−946.90	(1 218.76)	458.41	(186.02)	429.48	(184.91)	525.70	(171.17)
Autos	*error*		−2136.42	(386.91)	−2528.45	(391.36)	−1478.48	(582.03)	−1308.21	(111.15)	−1211.34	(470.75)
Ionosphere	−682.32	(202.73)	−6633.65	(3 123.35)	−5846.25	(1 474.37)	−722.87	(284.33)	−731.52	(279.00)	−92.51	(410.13)
Qsar-biodeg	−5206.23	(212.84)	−2806.25	(357.13)	−18 775.94	(6 052.60)	−1272.40	(685.67)	−2128.14	(1 400.07)	*time*	
Housing-prices	*error*		−17 357.98	(5 216.93)	−41 389.21	(2 638.65)	*time*		−15 429.34	(1 350.31)	*time*	
Census-india	−82 609.07	(2 276.73)	*error*		−212 839.72	(42 269.43)	*time*		−68 548.53	(21 427.16)	*time*	

The winner in each row is highlighted in blue. Numbers between parentheses correspond to standard deviations.
time: the algorithm was unable to complete a fold in a week.
error: the algorithm returned an error.

Table 6.5: 10-fold cross-validated predictive log-likelihood for various mixed datasets (rows) and algorithms (columns).

	KDE		MSPN		GLFM		IL		CIL [$\alpha = 10$]		GLSL	
Haberman	0.00	(0.00)	4.34	(1.49)	9.07	(0.52)	1.34	(0.03)	1.63	(0.07)	15.57	(5.44)
Iris	0.00	(0.00)	1.32	(0.27)	5.45	(0.35)	2.65	(1.16)	3.03	(0.43)	148.42	(4.17)
User-knowledge	0.00	(0.00)	1.99	(0.69)	16.56	(1.86)	1.64	(0.08)	1.16	(0.09)	354.11	(263.90)
Vertebral	0.01	(0.00)	2.85	(0.66)	35.12	(6.94)	14.68	(5.99)	52.43	(103.66)	1213.54	(199.05)
Ecoli	*error*		11.07	(1.61)	30.65	(4.07)	3.92	(0.11)	1.87	(0.07)	8303.88	(1 133.25)
Planning-relax	0.01	(0.00)	28.67	(5.80)	64.44	(2.71)	15.81	(0.17)	3.31	(0.08)	576.06	(28.00)
Thoracic-surgery	0.03	(0.00)	3.43	(1.01)	111.99	(5.13)	118.50	(1.56)	4.44	(0.63)	10 327.42	(7 035.55)
Vehicle	0.08	(0.04)	85.12	(6.24)	553.84	(144.17)	986.61	(15.43)	293.17	(170.78)	104 936.72	(25 311.87)
Thyroid	0.84	(0.07)	11.22	(0.99)	2409.59	(153.39)	5418.17	(626.46)	457.04	(116.14)	586 680.44	(250 315.16)
Parkinsons	0.01	(0.00)	77.63	(11.10)	123.75	(18.29)	98.30	(8.55)	1.70	(0.04)	10 756.41	(3 770.89)
Autos	*error*		22.87	(1.56)	72.73	(12.43)	221.05	(3.57)	4.53	(0.90)	42 478.46	(3 593.67)
Ionosphere	0.02	(0.00)	222.49	(18.85)	517.23	(45.77)	2880.54	(150.23)	226.98	(71.53)	133 196.56	(32 968.60)
Qsar-biodeg	0.17	(0.00)	491.29	(37.76)	3500.91	(118.78)	25 229.09	(1 098.92)	1108.75	(282.47)	*time*	
Housing-prices	*error*		750.93	(99.15)	10 106.29	(1 525.88)	*time*		1149.53	(706.70)	*time*	
Census-india	1.47	(0.04)	*error*		43 458.55	(2 819.02)	*time*		23 575.06	(4 266.94)	*time*	

The winner in each row is highlighted in blue. Numbers between parentheses correspond to standard deviations.
time: the algorithm was unable to complete a fold in a week.
error: the algorithm returned an error.

Table 6.6: 10-fold cross-validated execution times (in seconds) for various mixed datasets (rows) and algorithms (columns).

use the same neural network architecture as in the article and set the dimensionality of parameters z, y and s to 10, 5 and 10, respectively. We trained the model with mini-batches of 1000 samples and 2000 epochs, using the Python implementation available at https://github.com/probabilistic-learning/HI-VAE.

As a performance measure, we estimated the average imputation error. Given a dataset with N instances and m variables, the average imputation error is computed as follows:

$$AvgErr = \frac{1}{m}\sum_i err(i) \ ,$$

where we use the following error metrics for each variable X_i, since the computation of the errors depends on the type of variable we are considering:

- Normalized root mean square error for continuous variables, i.e.,

$$err(i) = \frac{\sqrt{1/N\sum_n (x_i[n] - \hat{x}_i[n])^2}}{max(X_i) - min(X_i)}.$$

- Accuracy error for categorical variables, i.e.,

$$err(i) = \frac{1}{N}\sum_n (x_i[n] \neq \hat{x}_i[n]).$$

Imputations were done using MAP estimates, where $\hat{x}_i[n]$ refers to the imputed value of X_i in the data instance n.

6.4.2.1 Comparative study on categorical data

To conduct the study, we considered 15 real-world categorical datasets from the UCI repository and the literature. Table 5.1 shows the basic properties of each dataset. None of these datasets provided expert knowledge to set Bayesian priors. For this reason, we set the prior parameters of GLSL models using an empirical Bayes approach.

Figure 6.4 illustrates the average imputation error for each method and dataset as we varied the percentage of missing data. While there existed some disparities between datasets, GLSL and HI-VAE were the best-performing methods, achieving top performance in several datasets and consistently outperforming mean imputation. Both MICE and GLFM returned worse results than GLSL and HI-VAE in the majority of datasets. In addition, there were several datasets (i.e., Breast cancer, Solar flare, and Webkb-336) in which both MICE and GLFM returned even worse results than mean imputation. As in the density estimation experiments, GLSL was unable to finish execution in the 3 largest datasets (i.e., Coil-42, News-100, and Webkb-336). In order to better compare the performance of the considered methods, Figure 6.5 presents a summary of the average imputation error for each categorical dataset (for which GLSL completed its execution) at each missing percentage.

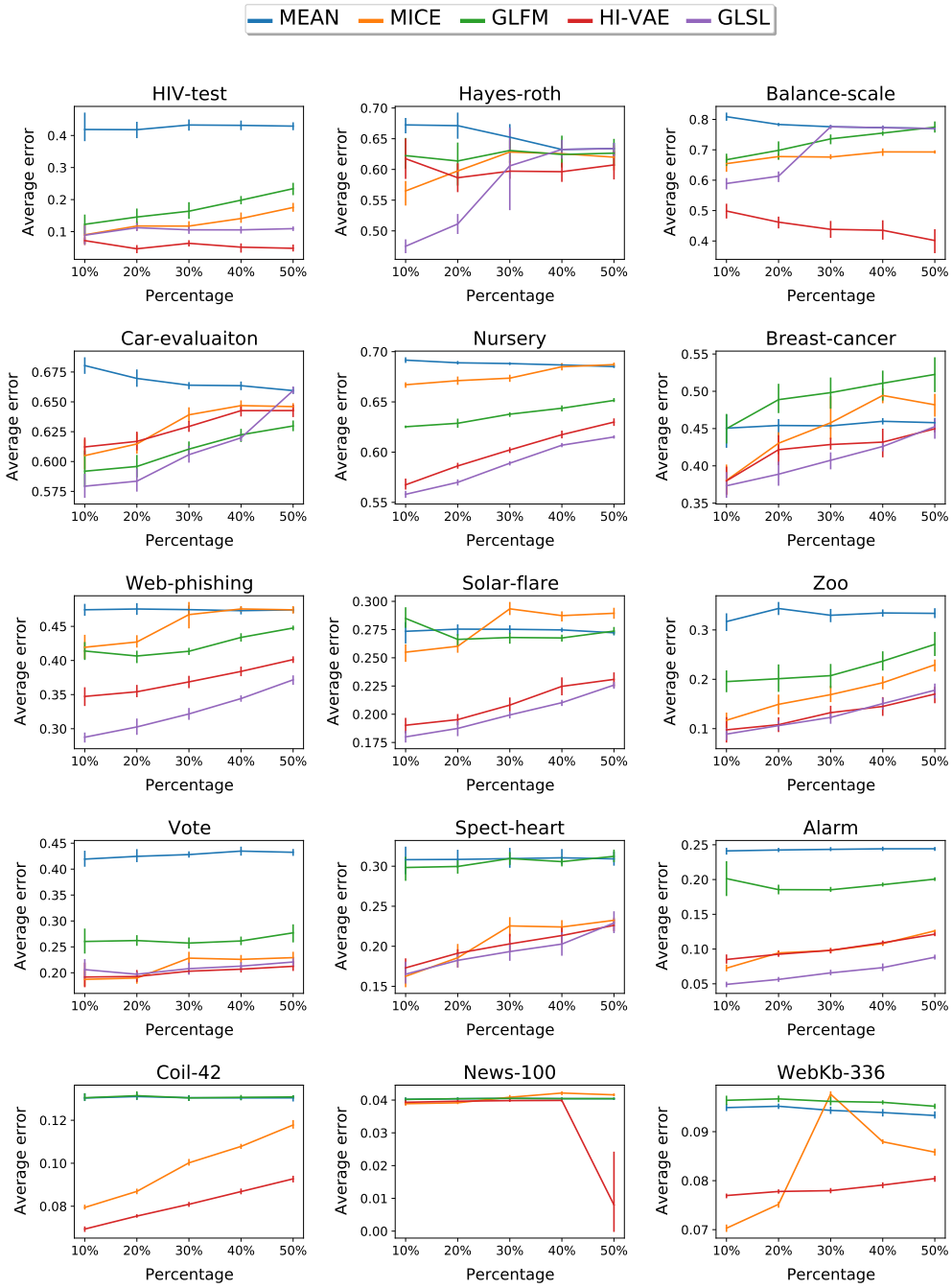


Figure 6.4: Average imputation error for each method and categorical dataset as the percentage of missing data varies.

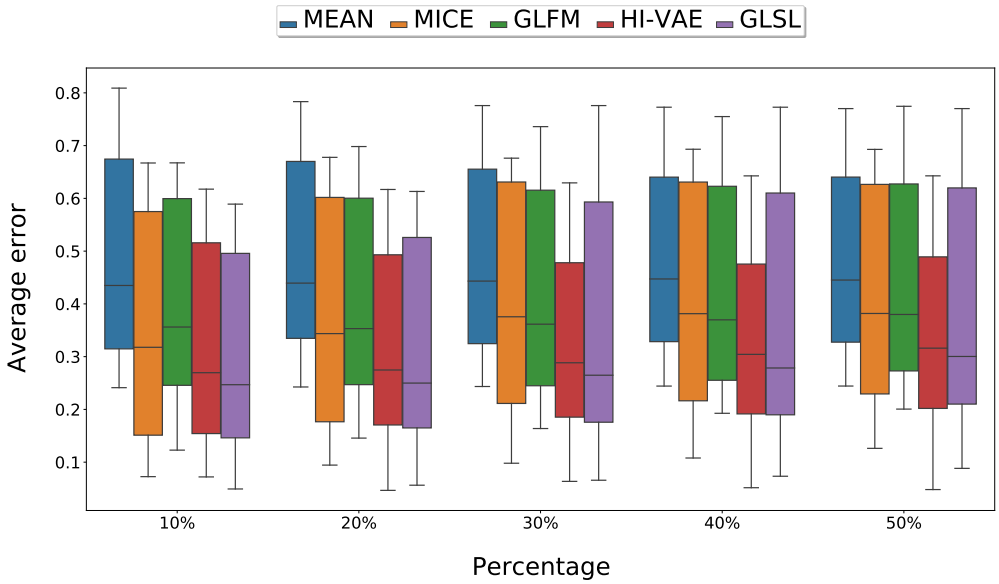


Figure 6.5: Summary of the average imputation error of each method for different percentages of categorical missing data.

	MICE		GLFM		HI-VAE		GLSL	
HIV-test	0.04	(0.01)	9.62	(0.31)	30.04	(4.82)	54.10	(11.80)
Hayes-roth	0.05	(0.01)	6.38	(0.32)	34.15	(3.97)	33.26	(21.68)
Balance-scale	0.05	(0.01)	27.50	(0.65)	38.49	(1.46)	136.46	(115.71)
Car-evaluation	0.09	(0.02)	122.24	(3.58)	61.36	(2.70)	1622.92	(716.22)
Nursery	0.47	(0.11)	1373.71	(136.66)	725.40	(72.46)	71 558.75	(38 145.17)
Breast-cancer	0.10	(0.02)	29.83	(5.19)	53.48	(3.63)	3231.11	(1 983.94)
Web-phishing	0.15	(0.03)	185.68	(33.65)	84.75	(4.84)	7901.43	(3 595.63)
Solar-flare	0.26	(0.09)	128.02	(2.17)	100.48	(7.08)	75 919.43	(47 139.73)
Zoo	0.21	(0.05)	22.71	(3.40)	99.18	(10.23)	8691.76	(3 493.95)
Vote	0.20	(0.05)	35.99	(1.81)	105.45	(10.39)	7361.50	(2 627.54)
Spect-heart	0.32	(0.04)	45.56	(2.49)	102.38	(4.28)	13 022.11	(1 959.14)
Alarm	2.58	(1.18)	340.33	(41.92)	296.01	(10.84)	692 848.89	(50 811.14)
Coil-42	11.81	(2.90)	1596.19	(263.70)	1046.58	(41.46)	*time*	
News-100	187.62	(54.31)	13 637.22	(668.41)	9927.24	(716.26)	*time*	
Webkb-336	819.96	(87.68)	4422.09	(60.27)	3233.39	(545.97)	*time*	

The winner in each row is highlighted in blue. Numbers between parentheses correspond to standard deviations.
time: the algorithm was unable to complete a run in a week.

Table 6.7: Missing data imputation times (in seconds) for various categorical datasets (rows) and algorithms (columns).

Table 6.7 presents the average execution time of each method for each dataset. We grouped the results of all the considered missing percentages since we did not observed much difference between them in terms of execution times. From the results, MICE was the fastest method, ranking first in all of the considered datasets. GLSL was the slowest method, being unable to complete its execution in 3 out of 15 datasets.

6.4.2.2 Comparative study on continuous data

As with the previous comparative study, we considered 15 real-world continuous datasets from the UCI repository and the current literature. Table 5.4 shows the basic properties of each dataset. None of these datasets provided expert knowledge to set Bayesian priors. For this reason, we set the prior parameters of GLSL models using an empirical Bayes approach.

Figure 6.6 illustrates the average imputation error for each method and dataset as we varied the percentage of missing data. In addition, Figure 6.7 presents a summary of the average imputation error for each continuous dataset (for which GLSL completed its execution) at each missing percentage. Contrary to the categorical case, MICE was the best-performing method, achieving top performance in the majority of datasets, especially when the percentage of missing data was low (i.e., 10% and 20%). HI-VAE displayed the second best performance. From the considered methods, GLSL and GLFM showed the worst performances. GLFM behaved slightly better when the percentage of data was low (i.e., 10% and 20%) and GLSL performed slightly better when the percentage of data was high (i.e., 40% and 50%). Finally, although GLSL did not show a great performance, its error results did not degrade as much as other methods (i.e., GLFM and MICE) with respect to the percentage of missing data.

Identically to the categorical case, Table 6.8 illustrates that MICE was the fastest method, ranking first in all of the considered datasets. Additionally GLSL was the slowest method, being unable to complete its execution in 3 out of 15 datasets.

6.4.2.3 Comparative study on mixed data

As with previous comparative studies, we considered 15 real-world mixed datasets from the UCI repository and the current literature. Table 5.7 shows the basic properties of each dataset. None of these datasets provided expert knowledge to set Bayesian priors. For this reason, we set the prior parameters of GLSL models using an empirical Bayes approach.

Figure 6.8 shows the average imputation error for each method and dataset as we varied the percentage of missing data. In addition, Figure 6.9 exhibits a summary of the average imputation error for each mixed dataset (for which GLSL completed its execution) at each missing percentage. Similar to the categorical case, GLSL and HI-VAE were the best-performing methods. Of the two, GLSL performed better when the percentage of missing data was high (i.e., 40% and 50%), and HI-VAE performed better when the percentage of data was around 20% and 30%. They showed a similar performance when the missing percentage was very low (i.e., 10% of missing data). Both MICE and GLFM performed worse than GLSL

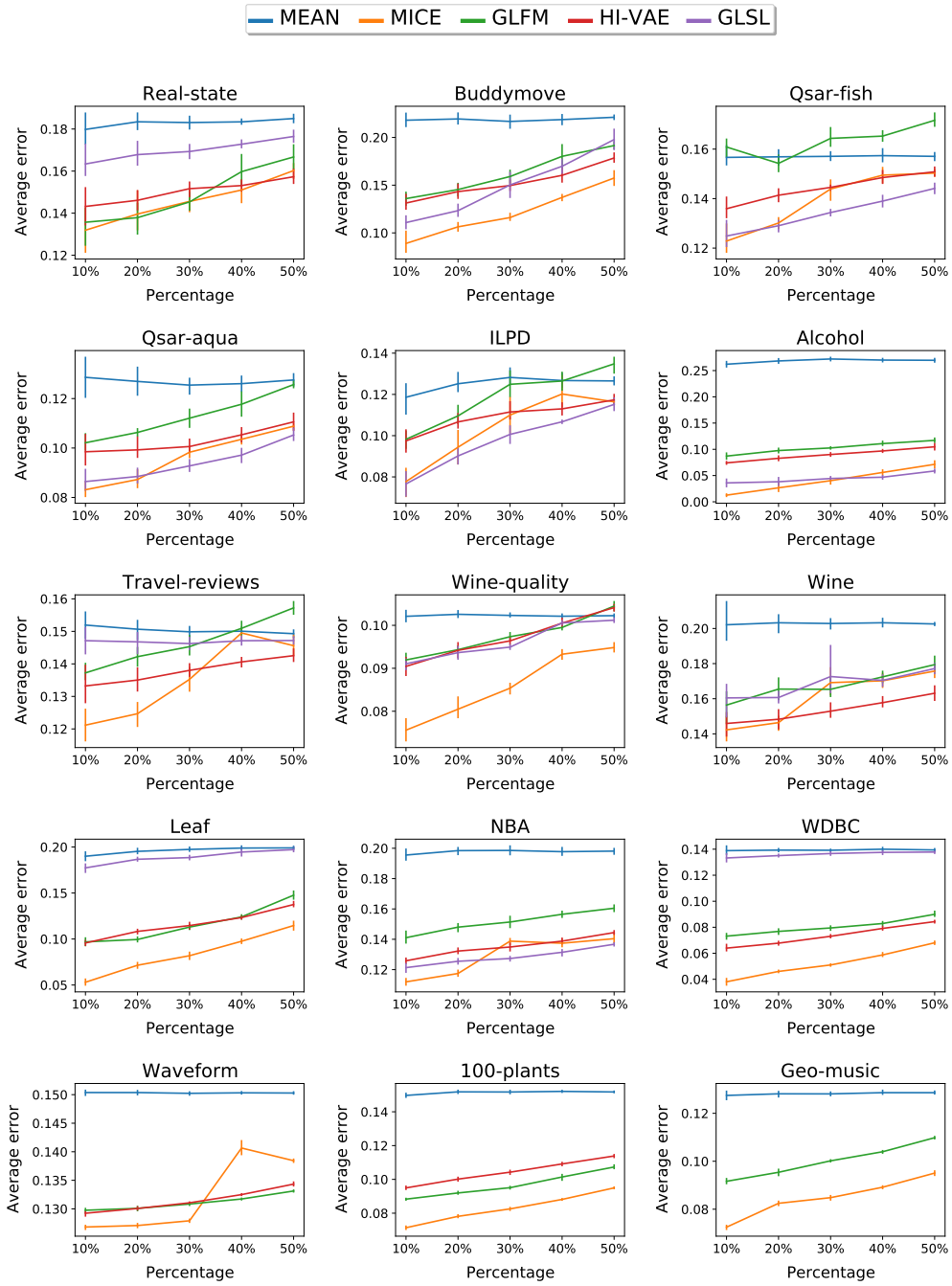


Figure 6.6: Average imputation error for each method and continuous dataset as the percentage of missing data varies.

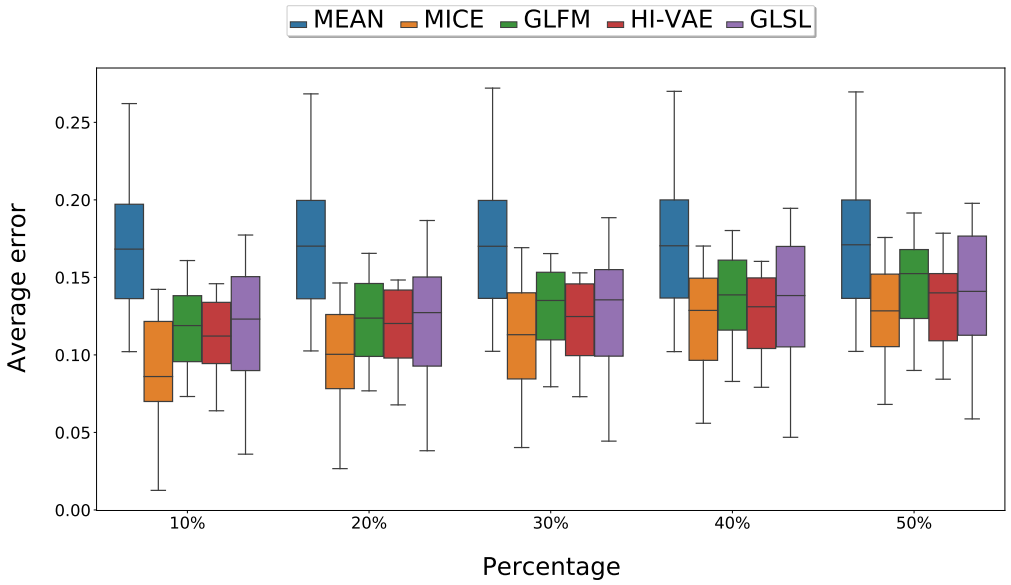


Figure 6.7: Summary of the average imputation error of each method for different percentages of continuous missing data.

	MICE		GLFM		HI-VAE		GLSL	
Real-state	0.04	(0.01)	15.57	(0.29)	59.55	(2.36)	1651.05	(1 306.12)
Buddymove	0.07	(0.01)	14.14	(0.40)	69.40	(5.79)	518.42	(273.74)
Qsar-fish	0.09	(0.01)	73.21	(3.87)	90.79	(13.79)	3245.26	(1 405.43)
Qsar-aqua	0.10	(0.03)	70.45	(0.99)	103.17	(5.45)	6483.70	(4 925.20)
ILPD	0.14	(0.02)	99.03	(4.97)	184.03	(7.25)	11 009.09	(3 816.93)
Alcohol	0.11	(0.04)	16.47	(1.90)	186.32	(7.89)	1064.20	(1 142.59)
Travel-reviews	0.15	(0.02)	156.85	(0.88)	115.99	(2.86)	14 084.96	(7 381.36)
Wine-quality	0.54	(0.09)	1158.77	(8.64)	369.92	(18.55)	722 189.04	(445 835.68)
Wine	0.16	(0.07)	44.86	(7.26)	144.38	(42.75)	4867.45	(2 822.69)
Leaf	0.18	(0.06)	100.78	(7.44)	144.20	(10.08)	8211.79	(4 116.24)
NBA	0.23	(0.14)	343.20	(92.55)	259.49	(89.40)	81 962.45	(30 689.33)
WDBC	0.61	(0.29)	1053.38	(94.75)	295.10	(77.39)	359 282.92	(145 634.42)
Waveform	8.67	(3.21)	1922.38	(49.72)	1361.52	(270.29)	*time*	
100-plants	2.02	(1.06)	8902.59	(587.20)	1073.49	(86.73)	*time*	
Geo-music	14.93	(4.49)	5967.75	(691.06)	*error*		*time*	

The winner in each row is highlighted in blue. Numbers between parentheses correspond to standard deviations.
time: the algorithm was unable to complete a run in a week.

Table 6.8: Missing data imputation times (in seconds) for various continuous datasets (rows) and algorithms (columns).

and HI-VAE. Of the two, GLFM was the worst performing method, even achieving slightly worse average performance than mean imputation when there was a 50% of missing data.

Identically to both the categorical and continuous cases, MICE was the fastest method (see Table 6.9), ranking first in all of the considered datasets. Additionally GLSL was the slowest method, being unable to complete its execution in the 3 largest datasets.

6.5 Conclusions and future work

In this chapter, we developed a new method, named GLSL, for learning conditional linear Gaussian BNs with multiple categorical latent variables. Since searching directly in the space of models would be computationally unfeasible, GLSL follows instead an iterative process with five latent operators and a VB version of the SEM algorithm. The main idea is to produce multiple candidate models that introduce a new latent variable, remove an existing latent variable, or change the cardinality of a latent variable. Then, learn the local structure of the proposed candidates and select the model with the highest score.

We demonstrated the applicability of GLSL in density estimation and missing data imputation tasks. In these experiments, we considered categorical, continuous, and mixed real-world data. GLSL demonstrated its good performance against several parametric and nonparametric methods from the state of the art. However, its implementation could be improved to obtain faster results.

There are various future research directions. First, we can extend VB-SEM by using more flexible variational families (instead of VB) and prior distributions [Barber and Wiegerinck, 1999; Bishop et al., 1998; Wang and Blei, 2013], but at the expense of a more difficult variational optimization problem. Second, we can speed up GLSL by using parallelization (e.g., by simultaneously evaluating multiple latent operators). Finally, we can investigate a variant of variational autoencoders where there is a latent superstructure with multiple categorical latent variables [Li et al., 2019; Falck et al., 2021] that is learned using GLSL.

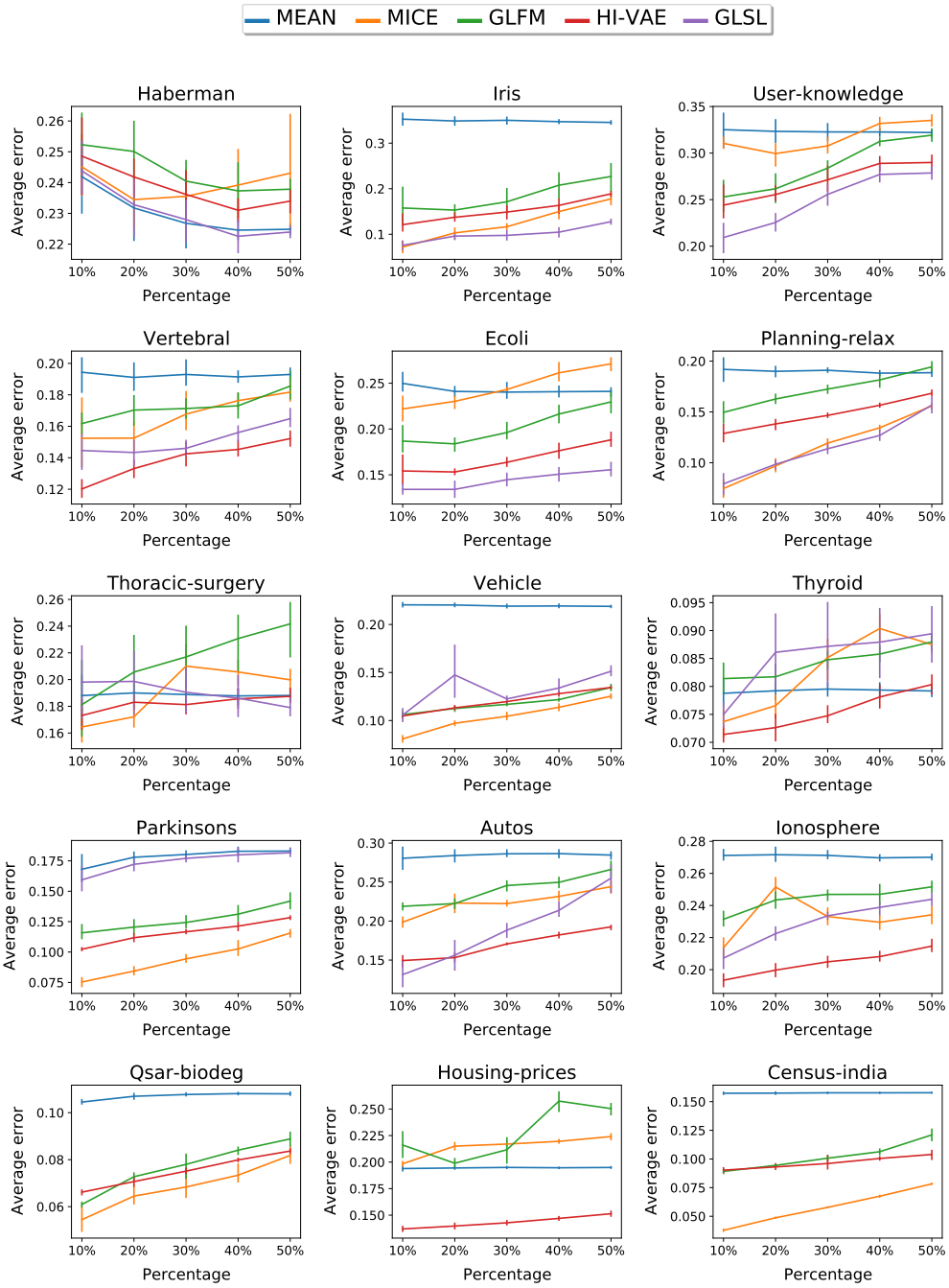


Figure 6.8: Average imputation error for each method and mixed dataset as the percentage of missing data varies.

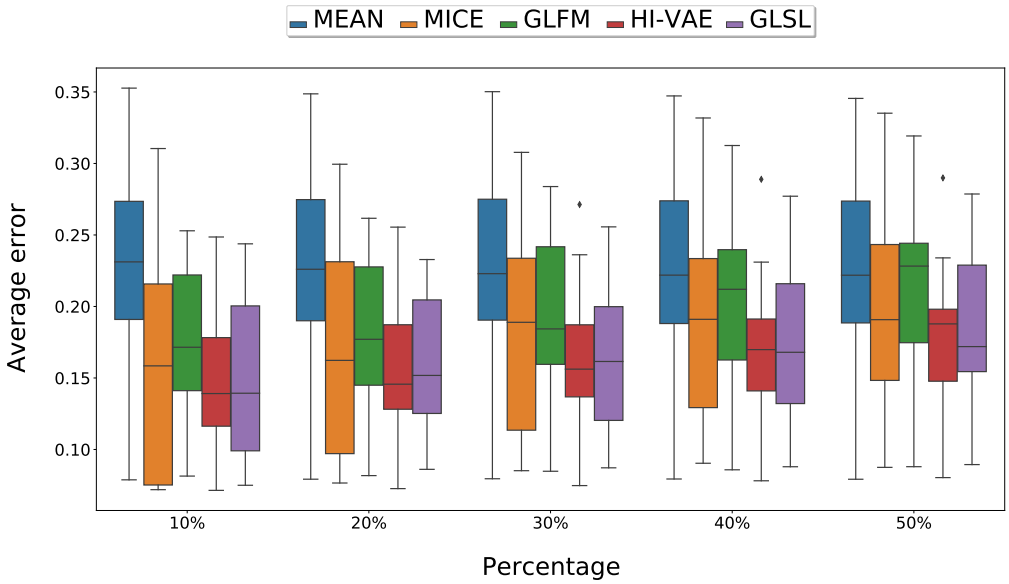


Figure 6.9: Summary of the average imputation error of each method for different percentages of mixed missing data.

	MICE		GLFM		HI-VAE		GLSL	
Haberman	0.03	(0.01)	8.47	(0.16)	45.35	(9.80)	34.94	(16.02)
Iris	0.06	(0.00)	8.65	(0.67)	49.41	(7.27)	114.92	(47.38)
User-knowledge	0.07	(0.01)	19.57	(0.38)	73.28	(4.00)	86.27	(14.10)
Vertebral	0.08	(0.02)	38.31	(2.03)	74.09	(22.83)	804.44	(380.33)
Ecoli	0.09	(0.01)	31.87	(1.37)	70.97	(16.28)	1997.56	(1 252.17)
Planning-relax	0.15	(0.04)	54.41	(13.55)	108.18	(17.92)	888.81	(436.57)
Thoracic-surgery	0.17	(0.06)	139.67	(6.72)	171.18	(11.85)	4060.89	(1 679.43)
Vehicle	0.25	(0.10)	562.70	(34.00)	302.30	(27.32)	310 741.70	(208 581.18)
Thyroid	0.57	(0.25)	2722.98	(173.77)	520.57	(10.99)	654 996.82	(482 907.46)
Parkinsons	0.37	(0.08)	134.35	(11.78)	193.93	(26.80)	47 831.03	(38 792.18)
Autos	0.34	(0.12)	151.87	(63.79)	176.70	(35.57)	288 169.36	(268 149.54)
Ionosphere	0.66	(0.13)	470.76	(92.36)	400.10	(22.51)	105 642.05	(12 690.94)
Qsar-biodeg	1.54	(1.25)	3209.04	(184.74)	562.45	(15.39)	*time*	
Housing-prices	5.37	(3.39)	13 403.68	(1 608.05)	771.79	(123.27)	*time*	
Census-india	25.87	(9.55)	47 979.61	(7 153.93)	1748.12	(543.79)	*time*	

The winner in each row is highlighted in blue. Numbers between parentheses correspond to standard deviations.
time: the algorithm was unable to complete a run in a week.

Table 6.9: Missing data imputation times (in seconds) for various mixed datasets (rows) and algorithms (columns).

Chapter 7

Identifying Parkinson's disease subtypes via multidimensional clustering

7.1 Introduction

PD is a progressive neurodegenerative disease that is clinically characterized by a broad spectrum of motor and non-motor manifestations [Greenland et al., 2019]. There is, however, considerable clinical phenotypic and natural history related variability between PD patients, which may indicate the existence of disease subtypes. Identification of PD subtypes may help understand the underlying disease mechanisms, since homogeneous groups of patients may be more likely to share pathological and genetic features. In addition, identification of PD subtypes may ultimately lead to more precise treatment strategies (i.e., precision medicine) [Chase, 2015; Marras et al., 2020].

Data-driven techniques such as clustering may be suitable for establishing PD subtypes. In clustering, patients are assigned to several groups (i.e., clusters) so that patients belonging to the same group share similarities. Each of these groups is usually then considered a subtype of the disease. Previous clustering studies have already identified PD subtypes with motor and non-motor symptoms [Mavandadi et al., 2009; van Rooden et al., 2011; Erro et al., 2013; Pont-Sunyer et al., 2015; Ma et al., 2015; Erro et al., 2016; Fereshtehnejad and Postuma, 2017; Mu et al., 2017; Martinez-Martin et al., 2020]. However, to the best of our knowledge, all of them have used method such as K-means [MacQueen, 1967], latent class analysis [Lazarsfeld and Henry, 1968], and agglomerative hierarchical clustering [Gordon, 1987], that assumed the existence of a single true clustering in a dataset. As a result, each patient was assigned to a single subtype that considered all the symptoms.

Recently, several issues have been raised about data-driven PD subtypes, such as the low number in the samples, their lack of internal homogeneity, and their difficulty to reproduce meaningful data in real life and external validity [Qian and Huang, 2019; Espay and Marras,

2019; Fearon et al., 2021; Mestre et al., 2021]. We believe that these issues may be a consequence of considering a single clustering solution. The assumption that each patient should be assigned to a single generic subtype does not hold for PD, which is usually multifaceted and can be meaningfully partitioned in multiple ways [Thenganatt and Jankovic, 2014; Marras, 2015]. For this reason, we advocate for multidimensional clustering (see Chapter 4). The idea is to use statistical principles to find sets of related symptoms where patients are divided into a number of distinct groups. Each set of symptoms defines a different clustering of patients. As a result, each patient is assigned to one subtype for each clustering. The analysis of these subtypes and their associations may provide more accurate insights about the considered symptoms, as well as their relationship with socio-demographic and clinical information of the patients.

Based on the above, the objectives of this chapter were: (i) to identify PD subtypes using multidimensional clustering, and (ii) to analyze the associations between the resulting subtypes. To this end, we used data from a large, multi-center, international, and well-characterized cohort of patients, where both motor and non-motor symptoms were recorded.

This chapter includes the content of Rodriguez-Sanchez et al. [2021b]. In addition, all code, data, and results are available at https://github.com/ferjorosa/parkinson-subtypes.

Chapter outline

In Section 7.2, we introduce the data and methods that were used in this study. Then, in Section 7.3, we present the results of applying these methods to the data. Finally, in Section 7.4, we discuss the clinical implications of our results, exploring their impact on current research and proposing future research directions.

7.2 Methods

7.2.1 Study design

The analysis was carried out on data gathered from the first validation study of the Movement Disorder Society non-motor rating scale (MDS-NMS), an international, multi-center, cross-sectional study that included 402 PD English-speaking patients from England and the United States [Chaudhuri et al., 2019]. The study was approved by the institutional review boards or ethics committees of the participating centers. All patients gave their written informed consent to participate in the study. Institutional review boards or ethics committees that approved the study: (1) National Research Ethics Service Committee East Midlands - Northampton, England; (2) Institutional Review Board at the Perelman School of Medicine at the University of Pennsylvania, United States. In addition, the study was conducted according to good clinical practice and all research was performed in accordance with relevant guidelines and regulations.

7.2.2 Assessments

For all patients, socio-demographic information and basic clinical variables (i.e., sex, age, age of onset, and disease duration) were recorded and the following assessments were applied:

1. The Movement Disorder Society unified Parkinson's disease rating scale (MDS-UPDRS) [Goetz et al., 2008], which is composed of 65 items divided across 4 parts, namely, Part I: Non-motor Experiences of Daily Living (13 items); Part II: Motor Experiences of Daily Living (13 items); Part III: Motor Examination (33 items); and Part IV: Motor Complications (6 items). Each item has 5 options of response, running from 0 (normal) to 4 (maximum intensity). The total score of each part is obtained by summing the respective item scores.

2. The MDS-NMS [Chaudhuri et al., 2019], which is composed of 52 items grouped into 13 domains: depression, anxiety, apathy, psychosis, impulse control and related disorders (ICRDs), cognition, orthostatic hypotension, urinary, sexual, gastrointestinal, sleep and wakefulness, pain, and other. Each item is scored for both frequency and severity, where both scores have 5 options of response, ranging from 0 (normal) to 4 (maximum intensity). Each item score is generated by multiplying frequency and severity. The score of each domain is obtained by summing the respective item scores. The MDS-NMS also includes a subscale for non-motor fluctuations, composed of 8 items, which was not considered in this study.

3. The Hoehn & Yahr (H&Y) staging system [Hoehn and Yahr, 1967], which ranges from 1 to 5.

7.2.3 Data analysis

Items from the MDS-NMS were grouped into their respective domains, with the exception of the items from the "other" domain (unintentional weight loss, decrease in sense of smell, physical fatigue, mental fatigue, and excessive sweating). These items were individually considered due to their individual and unique status. This resulted in 17 non-motor variables. Additionally, motor items from the MDS-UPDRS were classed as 5 motor cardinal signs: tremor, rigidity, bradykinesia, dyskinesias and motor fluctuations; and 2 motor subtypes: axial symptoms and postural instability gait difficulty (PIGD) [Stebbins et al., 2013]. This resulted in 7 motor variables. The specific MDS-UPDRS items that constitute each motor variable are provided as supporting material (see Appendix A). Finally, with the objective of improving the interpretability of the results, both motor and non-motor variables were normalized to the [0, 1] range using min-max scaling.

7.2.3.1 Multidimensional clustering

To identify subtypes (i.e., groups of individuals) with specific patterns in the motor and non-motor domains, multidimensional clustering was performed. A BN was used to represent the

multidimensional clustering model. More specifically, a conditional linear Gaussian BN. To learn this model, GLSL (see Section 6.3) was employed. In order to make the resulting model easier to interpret, we established the restriction that each observed variable could only have one parent. This way, every latent variable acted as a clustering variable that could be easily interpreted by looking at its observed descendants. In addition, the model would have a hierarchical structure. This restriction did not affect clustering variables, which could have multiple parents, allowing multiple clusterings to be interconnected.

By using a conditional linear Gaussian BN, each subtype was defined by a linear Gaussian distribution whose dimensions corresponded to the clustering symptoms. The symbol μ was used to denote the mean of this subtype for a specific symptom and the symbol σ was used to denote the standard deviation. In addition, to improve the readability of these subtypes, we devised a simple scale that considered the quartiles of the normalized [0, 1] range to refer to the mean symptom severity: (i) slight [0.01, 0.25]; (ii) mild [0.26, 0.50]; (iii) moderate [0.51, 0.75]; and (iv) severe [0.76, 1]. Note that this scale differs from the MDS-UPDRS and MDS-NMS ratings.

7.2.3.2 Comparative subtype analysis

To explore the relationship between socio-demographic information, basic clinical variables, H&Y stage and the identified subtypes, hypothesis tests were performed. Each pair of subtypes in a clustering were compared. For continuous variables such as age, age of PD onset, and PD duration, a Mann-Whitney U-test [Mann and Whitney, 1947] (implemented in the Python library SciPy version 1.5.2) was used. When three or more subtypes were present in a clustering, a Kruskal-Wallis test [Kruskal and Wallis, 1952] (implemented in SciPy) was performed instead, followed by a post-hoc analysis using Tukey's range test [Tukey, 1949] (implemented in the Python library Statsmodels version 0.11.1). For categorical variables such as the sex of the patient and discrete variables such as the H&Y stage, χ^2 tests (implemented in Scipy) [Pearson, 1900] were performed. Statistical significance was defined as p-value $p < 0.01$.

7.2.3.3 Probabilistic inference

To analyze the multidimensional clustering model and extract useful associations between subtypes (and their relevant symptoms), probabilistic inference was employed (see Section 2.3). For example, consider two hypothetical clustering variables, A and B, which are connected by an arc in the multidimensional clustering model. Clustering A defines two subtypes $\{A1, A2\}$ according to a set of symptoms. Clustering B defines three subtypes $\{B1, B2, B3\}$ according to a different set of symptoms. We are interested in estimating the difference between the probability distributions $P(B)$ and $P(B|A = A1)$, but also the difference between $P(B)$ and $P(B|A = A2)$. That is, how belonging to a subtype in A affects the probability distribution of B. The inverse probabilistic queries are also relevant (i.e., how belonging to a subtype in B affects the probability distribution of A). Since each subtype in a clustering

is characterized by a set of symptoms with a certain severity, we are incidentally studying the relationships between their respective symptoms (i.e., how an increase or decrease of the severity of certain symptoms affect the probability of suffering the other symptoms with more or less severity) when we analyze the relationships between subtypes of different clusterings. In this study, probabilistic inferences were carried out using Monte Carlo sampling in the tool GeNIe (https://www.bayesfusion.com/genie/).

7.3 Results

A total of 402 patients were considered for this study. Average onset age was 59 $\pm$ 11 years, 62% were male and average PD duration was 8 $\pm$ 6 years. 13% of the patients were in H&Y stage 1; 54% in H&Y stage 2; 28% in H&Y stage 3; and 5% in H&Y stage 4. No patients in this study were in H&Y stage 5.

With respect to missing information, 64 values ($<$ 1% of the total) were missing, mostly in the Sexual domain of the MDS-NMS. As our multidimensional clustering method was able to work with missing information, no patients were excluded from the analysis. For more information about the data, see Table 7.1.

7.3.1 Model selection

In this section, we evaluate the quality of the multidimensional clustering model returned by GLSL. Due to the absence of prior information, we considered empirical Bayes priors for GLSL. We compared GLSL with several model-based clustering methods from the literature that allow continuous data. We considered three unidimensional clustering methods and two multidimensional clustering methods:

- **LCM** [Lazarsfeld and Henry, 1968], which learns an LVM where where all the observed variables are conditionally independent given a single categorical latent variable. The cardinality of the latent variable is iteratively estimated by maximizing the BIC score. The implementation is provided in our public repository.

- **uk-DB** [Pham and Ruz, 2009], which learns an LVM with a single categorical latent variable that is the parent of all the observed variables. Observed variables may also be the children of other $k-1$ observed variables. The cardinality of the latent variable is iteratively estimated using a scoring function, and the arcs between observed variables are usually estimated using SEM. We used the p-ELBO and the VB-SEM algorithm for the structure learning process. We also considered a value of k up to 5. The implementation is provided in our public repository.

- **Gaussian mixture model** (GMM) [McLachlan and Peel, 2004], which learns an LVM where all the observed variables follow a joint Gaussian distribution that is conditioned on a categorical latent variable. The cardinality of the latent variable is iteratively

	Variables	*N*	Mean (SD)	Median	Min	Max
Clinical	Age	402	67.42 (9.96)	68	35	93
	Age of onset	402	59.23 (10.67)	59	26	93
	Disease duration	402	8.19 (5.93)	7	0	35
	H&Y	402	2.25 (0.74)	2	1	4
MDS-NMS	Depression	401	0.07 (0.13)	0.01	0.00	0.90
	Anxiety	402	0.09 (0.13)	0.03	0.00	0.84
	Apathy	402	0.08 (0.15)	0.00	0.00	0.75
	Psychosis	402	0.03 (0.06)	0.00	0.00	0.56
	ICRDs	401	0.02 (0.05)	0.00	0.00	0.39
	Cognition	402	0.10 (0.12)	0.05	0.00	0.69
	Orthostatic hypotension	402	0.07 (0.13)	0.00	0.00	0.75
	Urinary	402	0.16 (0.19)	0.08	0.00	1.00
	Sexual	375	0.14 (0.25)	0.00	0.00	1.00
	Gastrointestinal	401	0.10 (0.12)	0.06	0.00	0.73
	Sleep and wakefulness	401	0.12 (0.12)	0.08	0.00	0.79
	Pain	402	0.13 (0.15)	0.08	0.00	0.83
	Unintentional weight loss	402	0.06 (0.18)	0.00	0.00	1.00
	Decrease in sense of smell	402	0.39 (0.40)	0.25	0.00	1.00
	Physical fatigue	402	0.21 (0.27)	0.06	0.00	1.00
	Mental fatigue	402	0.09 (0.19)	0.00	0.00	1.00
	Excessive sweating	402	0.07 (0.18)	0.00	0.00	1.00
MDS-UPDRS	Tremor	391	0.13 (0.12)	0.11	0.00	0.57
	Rigidity	398	0.19 (0.16)	0.15	0.00	1.00
	Dyskinesias	402	0.07 (0.15)	0.00	0.00	1.00
	Motor fluctuations	401	0.16 (0.19)	0.06	0.00	0.81
	Bradykinesia	394	0.29 (0.17)	0.25	0.00	0.89
	Axial symptoms	402	0.23 (0.15)	0.21	0.00	0.86
	PIGD	393	0.22 (0.19)	0.15	0.00	0.85

Items from both the MDS-NMS and MDS-UPDRS scales were normalized to the [0,1] range using min-max scaling.
N: sample size without missing values.
SD: standard deviation.
Min and Max: minimum and maximum recorded values.
MDS-NMS: Movement Disorder Society Non-Motor Rating Scale.
MDS-UPDRS: Movement Disorder Society Unified Parkinson's Disease Rating Scale.
H&Y: Hoehn & Yahr stage.
ICRDs: impulse control and related disorders.
PIGD: postural instability gait difficulty.

Table 7.1: Descriptive statistics of the data.

estimated by maximizing the BIC score. The implementation is provided in our public repository.

- **GEAST**. Gaussian version of the EAST algorithm [Poon et al., 2013].
- **IL**. Incremental learner (see Section 5.2.2).
- **Multi-partition mixture model** (MPMM) [Galimberti et al., 2018], which learns a forests of unconnected LCMs by iteratively grouping similar (according to MI) observed variables. The cardinality of each LCM is independently estimated by maximizing the BIC score. The implementation is provided in our public repository.

Method	Log-Likelihood[1]	BIC[1]	Learning time (s)	#Clusterings	#Subtypes	#Parameters
LCM	8808.38	8076.81	40.34	1	5	245
GMM	8795.14	6849.29	79.26	1	2	1202
uk-DB	7717.87	7277.13	115.75	1	2	147
GEAST	11150.21	10397.66	23650.34	18	55	301
IL	5496.33	5298.45	192.80	3	6	66
MPMM	5104.10	4912.21	1216.89	1	2	64
GLSL	11028.08	10635.31	7764.83	9	19	131

[1] The higher the score the better.
LL: log-likelihood.
BIC: Bayesian information criterion.
s: seconds.
#Clusterings: total number of clusterings.
#Subtypes: total number of subtypes across all clusterings.
#Parameters: number of model parameters.

Table 7.2: Models considered in this study.

From the considered methods, GEAST and MPMM were the only ones inherently incapable to work with missing values. For this reason, their models were learned using an imputed version of the dataset. Missing data imputation was carried out by the MICE algorithm [Azur et al., 2011]. We used the implementation provided in the Python library Scikit-learn version 0.24.2.

In order to evaluate the quality of these models, we estimated their LL and BIC scores. As indicated by Table 7.2, the scores of the unidimensional clustering models (i.e., LCM, GMM and uk-DB) were similar between them, but far from the majority of the multidimensional clustering models (i.e., GEAST and GLSL). IL and MPMM were the exception, returning the worst models even though they were multidimensional. In terms of score, GLSL returned the best model, showing the highest BIC score. Although GEAST obtained a better LL, the BIC difference indicated that GEAST probably suffered from overfitting. This intuition was corroborated by the fact that the GEAST model had 18 clusterings and 301 parameters, while the GLSL result had 9 clusterings and 131 parameters.

In terms of clustering quality, we observed remarkable differences between unidimensional clustering models, and even greater differences between unidimensional and multidimensional clustering models. First, we observed that both uk-DB (see Figure 7.1) and GMM (see Figure 7.2) were only able to find two subtypes. These two subtypes could be easily associated with the general severity of all the disease symptoms. One subtype was formed by patients with low (mean of 0.06) symptom severity and the other subtype was formed by patients with a slightly higher (mean of 0.18) symptom severity.

The other unidimensional clustering model, i.e., the LCM (see Figure 7.3), was able to find five subtypes: (i) patients with almost no severity in both motor and non-motor symptoms (mean of 0.05); (ii) patients with slight severity in both motor and non-motor (mean of 0.11), (iii) patients that also had slight severity in both motor and non-motor (mean of 0.12), but with some differences, such as the presence of sweating and the absence of weight loss problems; (iv) patients with slightly higher symptom severity in both motor and non-motor

symptoms (mean of 0.17), especially in mental problems such as depression, apathy and anxiety; (v) patients with mild severity in all symptoms (mean of 0.25).

While LCM was able to find more subtypes than both GMM and uk-DB, its subtypes also lacked specificity. The absence of feature selection or multiple clusterings led to every subtype being dependent on all the motor and non-motor symptoms. This resulted in insubstantial variables (i.e., tremor and rigidity, whose severity did not change much between subtypes) and general patterns (i.e., there were no remarkable differences between subtypes apart from subtypes 1 and 5, which were practically opposite). The main problem of unidimensional clustering models lies on the dimensionality of each subtype. Given that each subtype depends on all of the observed variables, increasing the number of subtypes to model a specific pattern results in a large number of unnecessary parameters (overfitting). For this reason, and to avoid overfitting, unidimensional clustering models usually end up identifying only a few subtypes.

Multidimensional clustering models returned by GEAST (see Figure 7.4) and GLSL (see Figure 7.5) were able to find a higher number of subtypes that were only related to certain symptoms. These subtypes were not only far more specific, but they also appeared to be far more faithful to the data (much higher scores). Between the two algorithms, GLSL was able to find the highest scoring model. While GEAST was able to find a good-fitting model, its intrinsic restriction to learning tree structures resulted in a model with 18 clusterings (55 subtypes) that was difficult to interpret. Alternatively, the model found by GLSL had 9 clusterings, with a total of 19 subtypes. In addition, it was able to exclude the tremor and rigidity symptoms from the clustering, which, as previously commented, appeared to be independent of the rest of symptoms. Compared to the GEAST result, GLSL returned a model much easier to interpret. Finally, IL and MPMM were unable to successfully group related symptoms. IL returned a model with three clustering variables and 6 subtypes (see Figure 7.6). MPMM returned a model with a single clustering variable and 2 subtypes (see Figure 7.7). In both models, the majority of the observed variables ended up being independent.

7.3.2 Multidimensional clustering and subtype analysis

The multidimensional clustering model that resulted from applying GLSL is protrayed in Figure 7.5. It consisted of 9 (alphabetically-named) clustering variables. Each clustering variable provided a different way to group PD patients according to a specific set of motor and non-motor variables. For example, in clustering A, patients were divided into two subtypes according to the severity of their ICRDs and PIGD. There was, however, one clustering variable that differed from the rest by not being directly related to any motor or non-motor variables. Instead, it acted as an auxiliary latent variable that connected clusterings G (weight loss - depression) and H (excessive sweating - anxiety). This variable was I, and to simplify the analysis, its relevant information was condensed in those of G and H. As a result, 8 clustering variables were considered, each with a different number of subtypes. The sex, age, age of onset, disease duration and H&Y stage of each subtype is provided in Table 7.3.

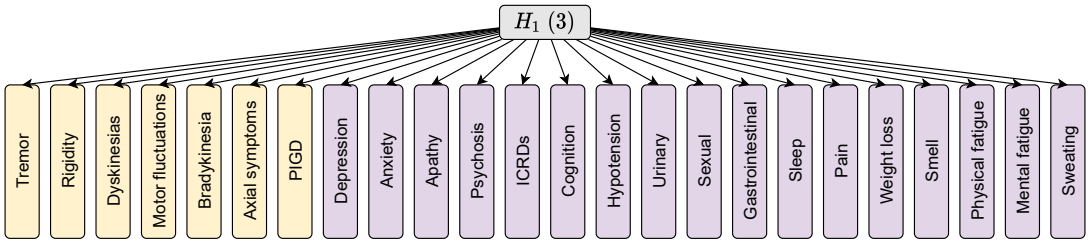


Figure 7.1: Unidimensional clustering model structure returned by the uk-DB algorithm. Motor variables are colored yellow while non-motor variables are colored purple. Clustering variables are colored grey, where the number in parentheses corresponds to the number of subtypes.

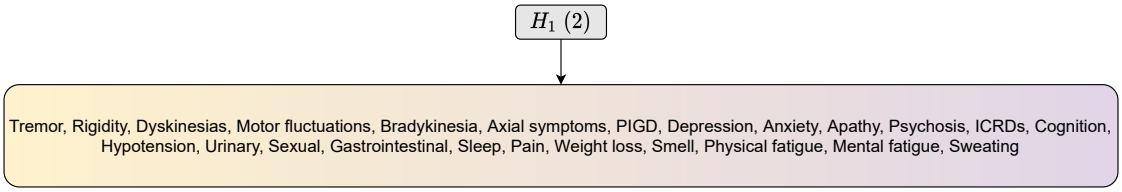


Figure 7.2: Unidimensional clustering model structure returned by the GMM algorithm. Colors and parentheses have the same meaning as in Figure 7.1. We followed the style of Poon et al. [2013] to represent joint nodes (i.e., nodes with a set of observed variables).

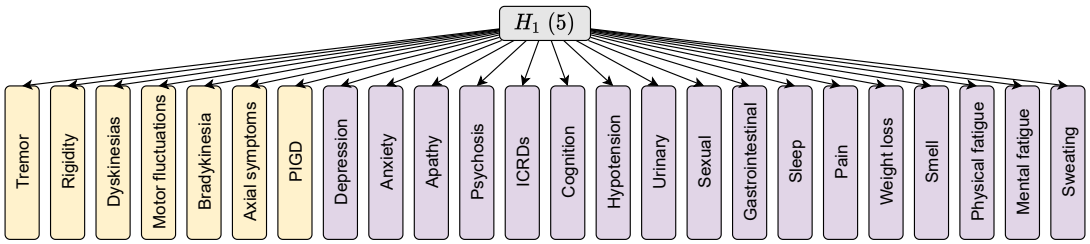


Figure 7.3: Unidimensional clustering model structure returned by the LCM algorithm. Colors and parentheses have the same meaning as in Figure 7.1.

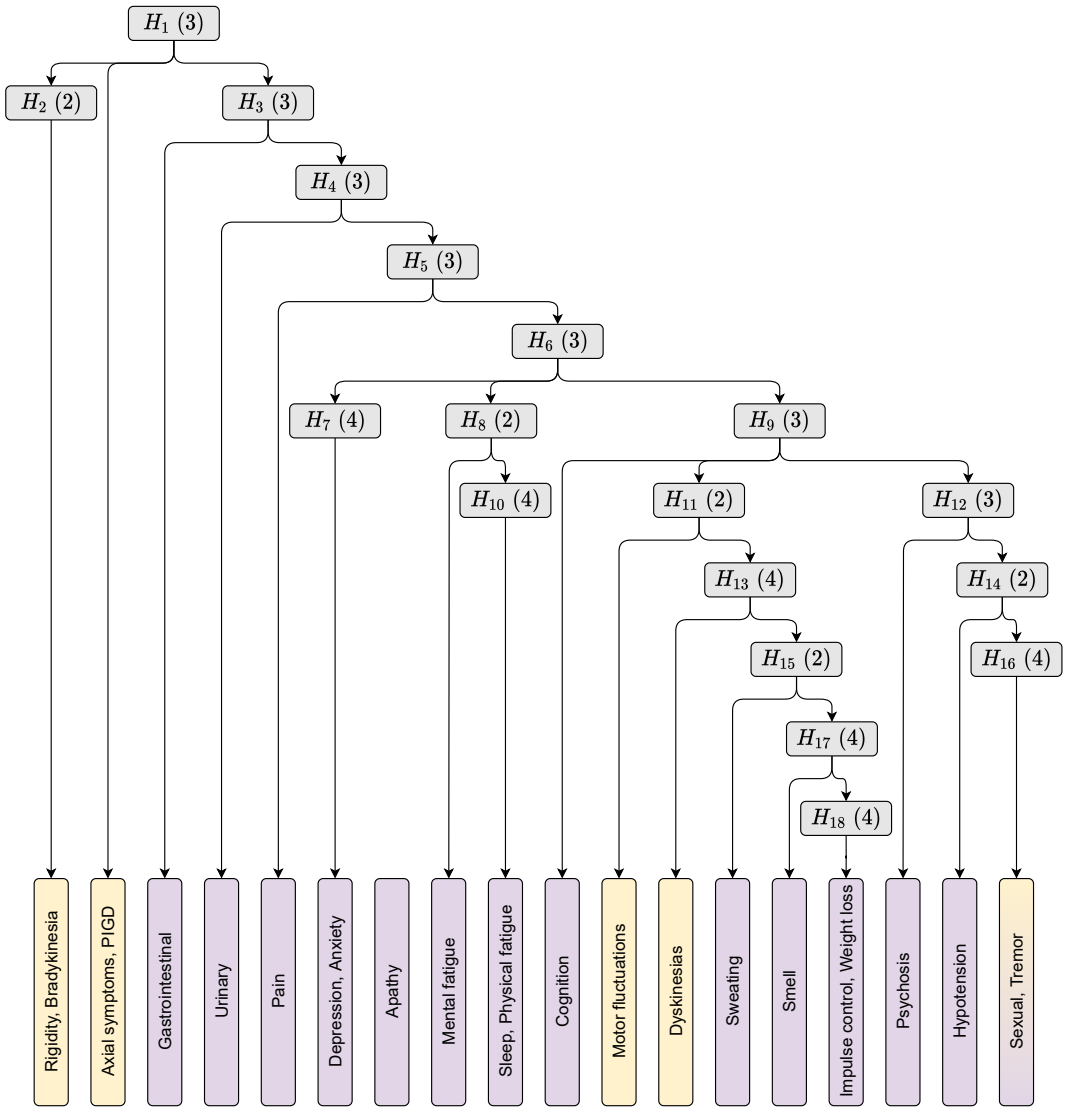


Figure 7.4: Multidimensional clustering model structure returned by the GEAST algorithm. Colors and parentheses have the same meaning as in Figure 7.1. We followed the style of [Poon et al., 2013] to represent joint nodes (i.e., nodes with a set of observed variables).

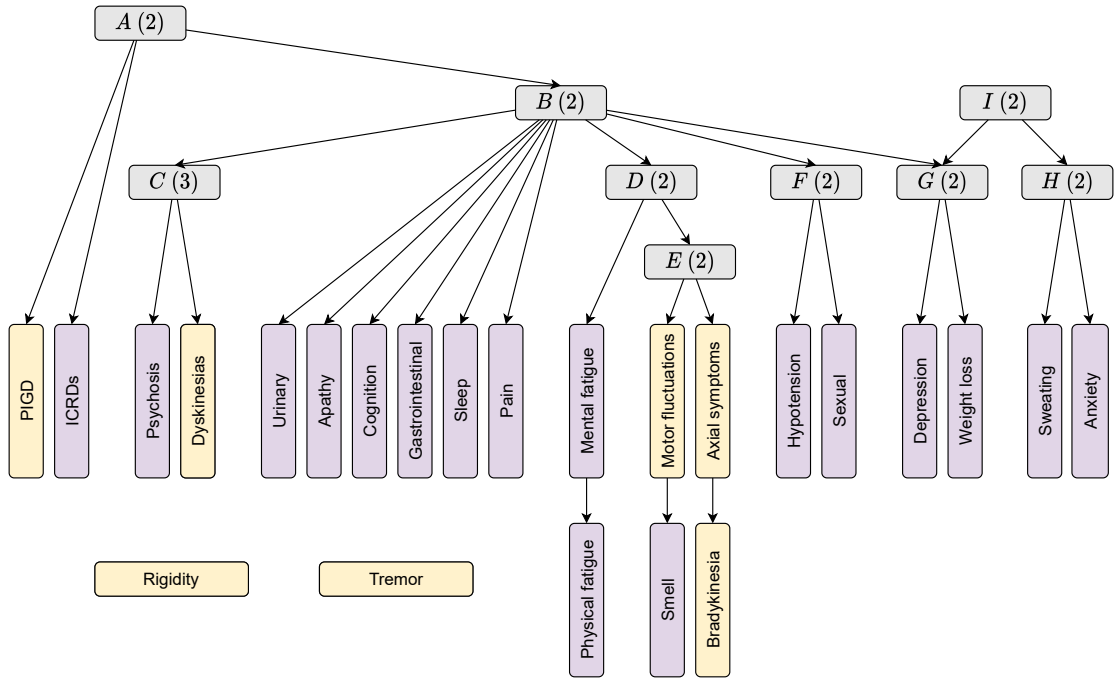


Figure 7.5: Multidimensional clustering model structure returned by the GLSL algorithm. Colors and parentheses have the same meaning as in Figure 7.1.

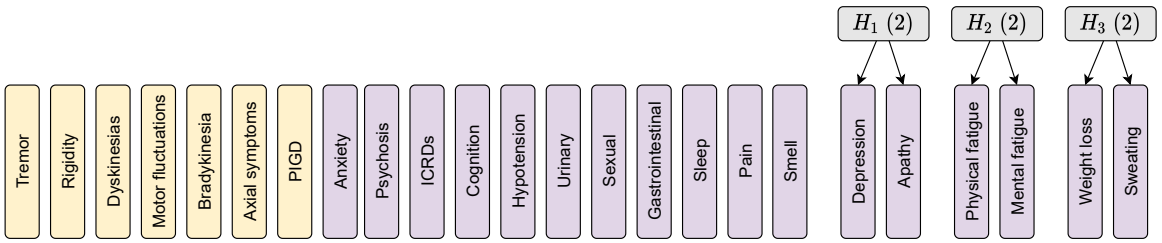


Figure 7.6: Multidimensional clustering model structure returned by the IL algorithm. Colors and parentheses have the same meaning as in Figure 7.1.

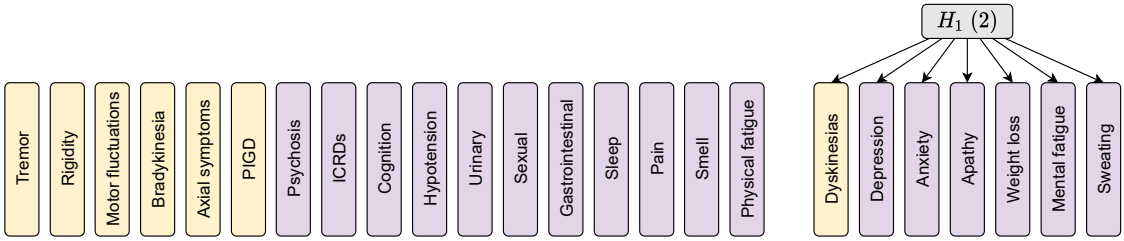


Figure 7.7: Multidimensional clustering model structure returned by the MPMM algorithm. Colors and parentheses have the same meaning as in Figure 7.1.

	Subtype	Sex (% male)	Age	Age of onset	Duration	H&Y
Clustering A	*A*1	62.39	68.02 (9.85)^{a}	60.00 (10.82)^{a}	8.02 (5.95)	2.27 (0.76)
	*A*2	61.19	64.45 (10.02)^{a}	55.37 (9.06)^{a}	9.07 (5.79)	2.16 (0.62)
Clustering B	*B*1	60.80	66.07 (10.18)^{b}	59.18 (10.79)	6.89 (5.15)^{b}	2.05 (0.71)^{b}
	*B*2	63.55	68.75 (9.58)^{b}	59.28 (10.59)	9.48 (6.36)^{b}	2.45 (0.72)^{b}
Clustering C	*C*1	60.74	67.42 (10.19)	60.72 (10.70)^{c_1,c_2}	6.70 (5.35)^{c_1,c_2}	2.10 (0.73)^{c_1,c_2}
	*C*2	64.29	67.10 (9.83)	56.98 (11.00)^{c_1}	10.12 (6.19)^{c_1}	2.46 (0.70)^{c_1}
	*C*3	64.58	68.19 (9.22)	56.92 (8.37)^{c_2}	11.27 (5.73)^{c_2}	2.54 (0.71)^{c_2}
Clustering D	*D*1	61.76	68.17 (10.21)	60.48 (10.92)^{d}	7.68 (5.65)^{d}	2.17 (0.74)
	*D*2	63.08	65.87 (9.27)	56.60 (9.66)^{d}	9.27 (6.36)^{d}	2.42 (0.72)
Clustering E	*E*1	63.32	68.61 (9.75)^{e}	62.25 (10.26)^{e}	6.36 (5.18)^{e}	2.13 (0.74)^{e}
	*E*2	61.08	66.26 (10.05)^{e}	56.27 (10.26)^{e}	10.00 (6.08)^{e}	2.37 (0.72)^{e}
Clustering F	*F*1	54.67^{f}	66.64 (10.49)	59.29 (11.04)	7.35 (5.29)^{f}	2.11 (0.75)^{f}
	*F*2	70.74^{f}	68.31 (9.26)	59.15 (10.27)	9.16 (6.46)^{f}	2.41 (0.69)^{f}
Clustering G	*G*1	63.09	67.37 (10.16)	59.31 (10.86)	8.05 (5.93)	2.20 (0.74)
	*G*2	58.82	67.74 (9.24)	58.91 (10.02)	8.73 (5.93)	2.46 (0.72)
Clustering H	*H*1	63.55	68.25 (9.67)^{h}	60.38 (10.31)^{h}	7.87 (5.99)^{h}	2.23 (0.76)
	*H*2	58.25	65.02 (10.44)^{h}	55.87 (11.06)^{h}	9.15 (5.67)^{h}	2.31 (0.69)

The numbers between parentheses correspond to standard deviations.
H&Y: Hoehn & Yahr stage.
^{a} Significant differences ($p < 0.01$) between *A*1 and *A*2.
^{b} Significant differences ($p < 0.01$) between *B*1 and *B*2.
^{c_1} Significant differences ($p < 0.01$) between *C*1 and *C*2.
^{c_2} Significant differences ($p < 0.01$) between *C*1 and *C*3.
^{c_3} Significant differences ($p < 0.01$) between *C*2 and *C*3.
^{d} Significant differences ($p < 0.01$) between *D*1 and *D*2.
^{e} Significant differences ($p < 0.01$) between *E*1 and *E*2.
^{f} Significant differences ($p < 0.01$) between *F*1 and *F*2.
^{g} Significant differences ($p < 0.01$) between *G*1 and *G*2.
^{h} Significant differences ($p < 0.01$) between *H*1 and *H*2.

Table 7.3: Sex, age, age of onset, disease duration, and H&Y stage of each clustering.

Clustering A (ICRDs - PIGD)

- Subtype $A1$ (84%) was characterized by patients that did not show problems to control their impulses ($\mu = 0.00$, $\sigma = 0.00$), but did show slight PIGD ($\mu = 0.22$, $\sigma = 0.19$).
- Subtype $A2$ (16%) was characterized by patients that showed slight problems to control their impulses ($\mu = 0.09$, $\sigma = 0.08$), and also presented slight PIGD ($\mu = 0.20$, $\sigma = 0.16$).

Clustering B (apathy - cognitive - pain - gastrointestinal - sleep - urinary)

- Subtype $B1$ (49%) was formed of patients that showed no apathy ($\mu = 0.00$, $\sigma = 0.00$), slight cognitive changes ($\mu = 0.03$, $\sigma = 0.04$), slight pain ($\mu = 0.06$, $\sigma = 0.07$), slight gastrointestinal problems ($\mu = 0.04$, $\sigma = 0.05$), slight sleep disorders ($\mu = 0.06$, $\sigma = 0.07$), and slight urinary issues ($\mu = 0.06$, $\sigma = 0.09$).
- Subtype $B2$ (51%) was formed of patients that showed slight apathy ($\mu = 0.16$, $\sigma = 0.18$), slight cognitive changes ($\mu = 0.17$, $\sigma = 0.14$), slight pain ($\mu = 0.19$, $\sigma = 0.17$), slight gastrointestinal problems ($\mu = 0.15$, $\sigma = 0.14$), slight sleep disorders ($\mu = 0.17$, $\sigma = 0.13$), and slight urinary issues ($\mu = 0.25$, $\sigma = 0.22$).

Clustering C (dyskinesias - psychosis)

- Subtype $C1$ (59%) was composed of patients that showed no dyskinesias ($\mu = 0.00$, $\sigma = 0.01$) or psychosis ($\mu = 0.00$, $\sigma = 0.00$).
- Subtype $C2$ (27%) was composed of patients that showed slight dyskinesias ($\mu = 0.18$, $\sigma = 0.18$) and slight psychosis ($\mu = 0.02$, $\sigma = 0.03$).
- Subtype $C3$ (14%) was composed of patients that showed slight dyskinesias ($\mu = 0.15$, $\sigma = 0.23$) and slight psychosis ($\mu = 0.14$, $\sigma = 0.11$).

Clustering D (mental fatigue - physical fatigue)

- Subtype $D1$ (66%) consisted of patients that showed no mental fatigue ($\mu = 0.00$, $\sigma = 0.01$) and slight physical fatigue ($\mu = 0.14$, $\sigma = 0.06$).
- Subtype $D2$ (34%) consisted of patients that showed mild mental fatigue ($\mu = 0.28$, $\sigma = 0.23$) and mild physical fatigue ($\mu = 0.35$, $\sigma = 0.06$).

Clustering E (axial symptoms - bradykinesia - loss of smell - motor fluctuations)

- Subtype $E1$ (47%) was constituted by patients that showed slight axial symptoms ($\mu = 0.19$, $\sigma = 0.15$), slight bradykinesia ($\mu = 0.21$, $\sigma = 0.02$), mild loss of smell ($\mu = 0.29$, $\sigma = 0.15$), but no motor fluctuations ($\mu = 0.00$, $\sigma = 0.02$).

- Subtype $E2$ (53%) was constituted by patients that showed mild axial symptoms ($\mu = 0.26$, $\sigma = 0.15$), mild bradykinesia ($\mu = 0.30$, $\sigma = 0.02$), moderate loss of smell ($\mu = 0.51$, $\sigma = 0.15$), and mild motor fluctuations ($\mu = 0.30$, $\sigma = 0.16$).

Clustering F (orthostatic hypotension - sexual problems)

- Subtype $F1$ (55%) was composed of patients that showed no orthostatic hypotension ($\mu = 0.00$, $\sigma = 0.02$) and slight sexual problems ($\mu = 0.01$, $\sigma = 0.03$).
- Subtype $F2$ (45%) was composed of patients that showed slight orthostatic hypotension ($\mu = 0.15$, $\sigma = 0.17$) and mild sexual problems ($\mu = 0.29$, $\sigma = 0.30$).

Clustering G (weight loss - depression)

- Subtype $G1$ (80%) was characterized by patients that showed no weight loss ($\mu = 0.00$, $\sigma = 0.01$) and slight depression ($\mu = 0.03$, $\sigma = 0.04$).
- Subtype $G2$ (20%) was characterized by patients that showed mild weight loss ($\mu = 0.26$, $\sigma = 0.32$) and slight depression ($\mu = 0.24$, $\sigma = 0.20$).

Clustering H (excessive sweating - anxiety)

- Subtype $H1$ (74%) consisted of patients that showed no degree of excessive sweating ($\mu = 0.00$, $\sigma = 0.01$) and slight anxiety ($\mu = 0.06$, $\sigma = 0.07$).
- Subtype $H2$ (26%) consisted of patients that showed mild degree of excessive sweating ($\mu = 0.27$, $\sigma = 0.28$) and slight anxiety ($\mu = 0.19$, $\sigma = 0.19$).

7.3.3 Probabilistic inference

A total of 29 probabilistic queries were performed to analyze the connections between the 8 clustering variables. Their results are portrayed in Table 7.4.

7.4 Discussion

Clustering variables were underpinned by a reasonable spread of contributory PD symptoms, thus bridging a statistical and clinical divide. Tremor and rigidity were the exceptions, appearing to be independent of the rest of variables in the model (see Figure 7.5). Weak correlation between rigidity, tremor, and non-motor symptoms is not uncommon and has also been observed in a recent study that considered a similar population [Berganzo et al., 2016].

In clustering A, patients were divided into two subtypes according to the severity of their ICRDs and PIGD. Although the mean PIGD of the subtypes did not differ by much, subtype $A2$ was characterized by a higher severity of ICRDs, a younger age and a younger age of onset. A relationship between young age, early PD onset and more severe ICRDs has been

Probabilistic query	Probability increase	Result
$P(B\|A = A1)$	$B1$: +0.05	0.49 → 0.54
$P(B\|A = A2)$	$B2$: +0.24	0.51 → 0.75
$P(A\|B = B1)$	$A1$: +0.08	0.84 → 0.92
$P(A\|B = B2)$	$A2$: +0.08	0.16 → 0.24
$P(C\|B = B1)$	$C1$: +0.19	0.59 → 0.78
$P(C\|B = B2)$	$C2$: +0.10	0.27 → 0.37
	$C3$: +0.09	0.14 → 0.23
$P(B\|C = C1)$	$B1$: +0.16	0.49 → 0.65
$P(B\|C = C2)$	$B2$: +0.16	0.51 → 0.67
$P(B\|C = C3)$	$B2$: +0.37	0.51 → 0.88
$P(D\|B = B1)$	$D1$: +0.20	0.66 → 0.86
$P(D\|B = B2)$	$D2$: +0.20	0.34 → 0.53
$P(B\|D = D1)$	$B1$: +0.14	0.49 → 0.63
$P(B\|D = D2)$	$B2$: +0.28	0.51 → 0.79
$P(F\|B = B1)$	$F1$: +0.22	0.55 → 0.77
$P(F\|B = B2)$	$F2$: +0.22	0.45 → 0.67
$P(B\|F = F1)$	$B1$: +0.19	0.49 → 0.68
$P(B\|F = F2)$	$B2$: +0.23	0.51 → 0.74
$P(G\|B = B1)$	$G1$: +0.11	0.80 → 0.91
$P(G\|B = B2)$	$G2$: +0.11	0.20 → 0.30
$P(B\|G = G1)$	$F1$: +0.06	0.49 → 0.55
$P(B\|G = G2)$	$F2$: +0.29	0.51 → 0.80
$P(E\|D = D1)$	$E1$: +0.10	0.47 → 0.57
$P(E\|D = D2)$	$E2$: +0.20	0.53 → 0.73
$P(D\|E = E1)$	$D1$: +0.14	0.66 → 0.80
$P(D\|E = E2)$	$D2$: +0.13	0.34 → 0.47
$P(H\|G = G1)$	$H1$: +0.05	0.74 → 0.79
$P(H\|G = G2)$	$H2$: +0.18	0.26 → 0.44
$P(G\|H = I1)$	$G1$: +0.05	0.80 → 0.85
$P(G\|H = I2)$	$G2$: +0.13	0.34 → 0.47

Table 7.4: Probabilistic queries that were performed to analyze the GLSL result.

previously observed [Weintraub et al., 2015]. Both socio-demographic aspects are known risk factors for ICRDs along with motor complications, a pre-PD history of ICRDs, and a dopamine agonist treatment [Gatto and Aldinio, 2019].

Apathy, cognition, pain, gastrointestinal, sleep and urinary symptoms were associated in clustering B. Two subtypes were identified, where the subtype $B2$ was characterized by a higher severity of these symptoms. This subtype is consistent with the Parkinson's apathy subtype [Dujardin et al., 2014; Sauerbier et al., 2015], which has been described to be formed by older patients that showed cognitive impairment, sleep disorders, and relatively severe motor symptoms. The relationship between sleep disorders and urinary problems may indicate the presence of nocturia [Batla et al., 2016]. In addition, a recent study has also identified a relationship between constipation and cognitive dysfunction in two independent cohorts of patients [Leta et al., 2021].

Clustering C distinguished three subtypes that differed according to the severity of dyskinesias and psychosis. Subtypes $C2$ and $C3$ consisted of patients with a longer duration of the disease and a younger age of onset. These subtypes coincided with the observation that dyskinesias and psychosis are usually present in late stages of PD and may be associated with

higher dopaminergic medication doses [Espay et al., 2018; Dave et al., 2020].

Fatigue is considered a common and complex non-motor symptom of PD, prevalent from the prodromal to the palliative stage. It is usually present from early stages of the disease and may often persist or even worsen over time [Friedman et al., 2016]. While fatigue is usually regarded as an independent symptom, it has been moderately associated with apathy, sleep disorders, depression, and motor problems [Stocchi et al., 2014; Siciliano et al., 2018]. Our model was able to capture this duality by identifying a specific clustering for fatigue symptoms (i.e., D), and connecting it with clusterings B (apathy, sleep and depression) and E (motor problems). In addition, patients that suffered from more severe fatigue showed a longer duration of the disease and a younger age of PD onset.

Bradykinesia, axial symptoms, and motor fluctuations were associated in clustering E with a decrease in sense of smell (i.e., hyposmia). Patients were divided into two subtypes. While both subtypes presented motor issues, subtype $E2$ was characterized by a higher severity of motor symptoms, hyposmia, and the presence of motor fluctuations. Anosmia/hyposmia is considered a preclinical marker of PD with relatively static severity. However, while it has not been associated to any particular PD phenotype [Rossi et al., 2016; Sui et al., 2019], a recent study has noted that normosmic PD patients usually display better motor function than hyposmic PD patients [Lee et al., 2015].

In clustering F, patients were divided into two subtypes based on orthostatic hypotension and sexual problems. While the rest of clusterings were independent of the sex of the patient, 71% of patients in $F2$ were male, showing significant differences in the sex of the patients belonging to $F1$ and $F2$. We also observed significant differences in the H&Y stage and PD duration of these patients, reflecting the later occurrence of the autonomic features of orthostatic hypotension and sexual dysfunction [Özcan et al., 2016; Palma and Kaufmann, 2018; Hiorth et al., 2019].

Weight loss and depression were associated in clustering G. Two subtypes were identified, where $G2$ was characterized by mild weight loss and depression. Loss of appetite due to depression is a known weight loss factor [Ma et al., 2018]. There were no significant differences in sex, age, age of onset, or H&Y stage of the patients belonging to $G1$ and $G2$.

Regarding clustering H, anxiety was associated with excessive sweating. Anxiety was present in both its defined subtypes, but patients in $H2$ showed a higher severity of this symptom along with the presence of a mild degree of excessive sweating. This result is consistent with a recent study [van Wamelen et al., 2019] in which anxiety and depression were more prominent in PD patients with hyperhidrosis.

It is important to note that none of the discovered clusterings were fully independent of each other. Belonging to a specific subtype in a clustering influenced the subtype probabilities in the rest of clusterings. By using probabilistic inference, we were able to study the effect of these associations on their respective symptoms. Some interesting patterns that we observed included: (i) patients with ICRDs ($A2$ subtype) had a 0.75 probability of presenting the symptoms of $B2$. This result is consistent with a recent study that has challenged the traditional concept of apathy and ICRDs as opposite symptoms [Scott et al., 2020]; (ii) patients

that suffered psychosis ($C3$ subtype) had an 0.88 probability of suffering the symptoms of $B2$. The presence of visual hallucinations has been linked to sleep deprivation, cognitive impairment and depression [Ffytche et al., 2017; Lenka et al., 2019]; and (iii) patients with mild mental and physical fatigue ($D2$ subtype) had a 0.79 probability of suffering the symptoms of $B2$ and a 0.73 probability of suffering the symptoms of $E2$. As previously discussed, fatigue has been related to the presence of apathetic symptoms, sleep disturbances, and higher H&Y stages [Stocchi et al., 2014; Siciliano et al., 2018].

The majority of clusterings were directly or indirectly influenced by B, which acted as a pivotal latent variable in the model. This aligns with the current observation that sleep disorders, depression, constipation, and other non-motor symptoms appear across the spectrum of patients with PD [Schapira et al., 2017].

7.4.1 Limitations

This study has some limitations. Concerning the population of the study, patients were not specifically selected for this analysis, but rather for the validation of the MDS-NMS. Nonetheless, the large sample size and the high quality of the collected data will allow these results to be contrasted and compared with the results of future studies. The sample was comparatively younger than the average population of patients with PD. It is therefore possible that the results differ in those with an older age where higher rates of comorbidities exist. In addition, we did not report a control group, although our intention was not to describe the symptoms as discriminant from normal subjects. Finally, we did not consider PD genetic biomarkers, which could provide more information about the identified subtypes.

7.4.2 Conclusions

Dividing PD patients into groups with common symptoms may help understand their underlying pathologies. In this study, we used multidimensional clustering to categorize patients according to 8 different sets of motor and/or non-motor symptoms. By using probabilistic inference, we were able to explore the resulting subtypes and extract useful patterns. Independent confirmation of these results could allow for more precise PD treatments. In the future, it would be interesting to research how the evolution of PD throughout the years would affect these subtypes, and to which extent they could be markers of PD progression.

Part IV

CONCLUSIONS

Chapter 8

Conclusions and future work

In this chapter, we summarize the most important contributions of this dissertation and describe future research directions. The chapter also includes the list of publications and submissions produced during this research.

Chapter outline

In Section 8.1, we summarize the main contributions of this dissertation. Then, in Section 8.2, we provide the list of publications and submissions produced during this research. Finally, in Section 8.3, we discuss future work and open issues.

8.1 Summary of contributions

Chapters 5 - 7 describe the original contributions of this dissertation.

- Chapter 5 proposes two incremental algorithms for learning forests of conditional linear Gaussian LTMs. Unlike current methods, the proposed algorithms are based on the VB framework, which allows them to work with mixed data and to introduce uncertainty into the learning process. The first method, which we refer to as incremental learner, hill-climbs the space of latent forests in a two-phase iterative process. In its first phase, the forest structure is expanded with a new arc or latent variable. In its second phase, the cardinalities of latent variables are estimated. Considering that directly searching this space requires the evaluation of a large number of candidate models, a constrained variant that only evaluates the most prominent α candidates of each iteration is also developed. As demonstrated by the density estimation experiments, the constrained approach is a valid alternative to the incremental learner method. It returns almost identical results for categorical data experiments, and displays only slightly worse performance for continuous and mixed data experiments. We believe that these differences are caused by the use of approximate MI estimation, and demonstrate that they can be reduced by increasing α.

- Chapter 6 presents a greedy algorithm for learning conditional linear Gaussian BNs with categorical latent variables. While it takes inspiration from the approaches presented in Chapter 5, the resulting models are not limited to tree-like structures. This is accomplished by introducing a VB version of the SEM algorithm. The main idea is to produce multiple candidate models that introduce a new latent variable, remove an existing latent variable, or change the cardinality of a latent variable. Then, learn the local structure of the proposed candidates using the VB-SEM algorithm and select the model with the highest score. The experiments demonstrate the effectiveness of the approach by solving both density estimation and missing data imputation tasks. In those experiments, our method is compared with several parametric and nonparametric methods from the state of the art. However, our implementation could be improved to obtain faster results.

- Chapter 7 provides a multidimensional clustering study with PD data. The data of 402 PD patients from an international, multi-center, and cross-sectional study is used to learn a conditional linear Gaussian BN with several categorical latent variables. Each latent variable represents a different clustering of PD patients that considers a unique subset of motor and/or non-motor symptoms. As a result, eight sets of related symptoms are identified. Each of them provides a different way to group patients: impulse control issues, overall non-motor symptoms, presence of dyskinesias and psychosis, fatigue, axial symptoms and motor fluctuations, autonomic dysfunction, depression, and excessive sweating. Each of these groups can be seen as a subtype of the disease. Significant differences between subtypes ($p < 0.01$) are found in sex, age, age of onset, disease duration, and H&Y stage.

8.2 List of publications

Peer-reviewed JCR journals

- F. Rodriguez-Sanchez, P. Larrañaga, and C. Bielza. Incremental learning of latent forests. *IEEE Access*, 8:22420-224432, 2020.

- F. Rodriguez-Sanchez, C. Bielza, and P. Larrañaga. Multi-partition clustering of mixed data with Bayesian networks. *Submitted*, 2021.

- F. Rodriguez-Sanchez, C. Rodriguez-Blazquez, C. Bielza, P. Larrañaga, D. Weintraub, P. Martinez-Martin, A. Rizos, A. Schrag, and K. R. Chaudhuri. Identifiying Parkinson's disease subtypes with motor and non-motor symptoms via model-based multi-partition clustering. *Submitted*, 2021.

Peer-reviewed conferences

- F. Rodriguez-Sanchez, P. Larrañaga, and C. Bielza. Discrete model-based clustering with overlapping subsets of attributes. *Proceedings of the 9th International Conference*

on Probabilistic Graphical Models, pages 392-403, 2018.

Technical reports

- F. Rodriguez-Sanchez, P. Larrañaga, C. Bielza. Multi-facet determination for clustering with Bayesian networks. TR:UPM-ETSIINF/DIA/2017-1: Universidad Politécnica de Madrid, 2017.

8.3 Future work

This section points out future research lines to the topics covered in this dissertation.

- Our proposals from Chapters 5 and 6 aim at learning BNs with categorical latent variables. Inspired by factor analysis, these works can be expanded to work with continuous latent variables, where the number of factors can be analogously estimated to the cardinality of categorical latent variables.

- We can extend the parameter estimation of Chapters 5 and 6 by considering more flexible variational families [Barber and Wiegerinck, 1999; Bishop et al., 1998] and nonconjugate priors [Wang and Blei, 2013], but at the cost of a more difficult variational optimization problem.

- In Chapters 5 and 6 we focus on selecting the most probable model by maximizing the p-ELBO. However, this approach may not be suitable when we are interested in quantifying our confidence in the structure of the result or when we have a low number of data instances. For these cases, Bayesian estimation can be introduced into the structure learning process by averaging over the set of possible BN structures (i.e., by using Bayesian model averaging [Hoeting et al., 1999; Fragoso et al., 2018]).

- We intend to parallelize our proposals from Chapters 5 and 6. The idea would be to simultaneously evaluate multiple latent operators.

- There is a growing interest in combining complex LVMs with unsupervised deep learning models [Li et al., 2019; Falck et al., 2021]. In this context, we can investigate a variant of variational autoencoders whose latent superstructure is learned using our method from Chapter 6.

- It would be interesting to consider data from more PD patients to analyze the consistency of the subtypes identified in Chapter 7. It would be especially interesting to access longitudinal data to study how the evolution of PD throughout the years would affect these subtypes, and to which extent they could be markers of PD progression.

Part V

APPENDICES

Appendix A

Motor variables specification

Cardinal signs

- **Tremor** (MDS-UPDRS items 2.10, 3.15 - 3.18)
- **Rigidity** (MDS-UPDRS items 3.3a-e)
- **Dyskinesias** (MDS-UPDRS items 4.1, 4.2)
- **Motor fluctuations** (MDS-UPDRS items 4.3 - 4.6)
- **Bradykinesia** (MDS-UPDRS items 3.4 - 3.8, 3.14)

Motor subtypes

- **Axial symptoms** (MDS-UPDRS items 2.1 - 2.3, 3.1 - 3.2, 3.9, 3.13)
- **PIGD** (MDS-UPDRS items 2.12, 2.13, 3.10, 3.11, 3.12)

Bibliography

D. J. Aigner, C. Hsiao, A. Kapteyn, and T. Wansbeek. *Handbook of Econometrics*, chapter 23: Latent variable models in econometrics, pages 1321–1393. Elsevier, 1984.

H. Akaike. A new look athe statistical model identification. *IEEE Transactions on Automatic Control*, 19(6):716–723, 1974.

A. Anandkumar, K. Chaudhuri, D. Hsu, S. M. Kakade, L. Song, and T. Zhang. Spectral methods for learning multivariate latent tree structure. In *Proceedings of the 25th Conference on Neural Information Processing Systems*, pages 2025–2033, 2011.

A. Anandkumar, R. G. D. Hsu, S. M. Kakade, and M. Telgarsky. Tensor decompositions for learning latent variable models. *Journal of Machine Learning Research*, 15:2773–2832, 2014.

N. Asbeh and B. Lerner. Learning latent variable models by pairwise cluster comparison: Part I - theory and overview. *Journal of Machine Learning Research*, 17(223):1–52, 2016a.

N. Asbeh and B. Lerner. Learning latent variable models by pairwise cluster comparison: Part II - algorithm and evaluation. *Journal of Machine Learning Research*, 17(230):1–45, 2016b.

H. Attias. A variational Bayesian framework for graphical models. In *Proceedings of the 14th Conference on Neural Information Processing Systems*, pages 209–215, 2000.

M. J. Azur, E. A. Stuart, C. Frangakis, and P. J. Leaf. Multiple imputation by chained equations: What is it and how does it work? *International Journal of Methods in Psychiatric Research*, 20(1):40–49, 2011.

E. Bae and J. Bailey. COALA: A novel approach for the extraction of an alternate clustering of high quality and high dissimilarity. In *Proceedings of the 6th IEEE International Conference on Data Mining*, pages 53–62, 2006.

Y. Barash and N. Friedman. Context-specific Bayesian clustering for gene expression data. *Journal of Computational Biology*, 9(2):169–191, 2002.

D. Barber and W. Wiegerinck. Tractable variational structures for approximating graphical models. In *Proceedings of the 13th Conference on Neural Information Processing Systems*, pages 183–189, 1999.

A. Batla, V. Phé, L. D. Min, and J. N. Panicker. Nocturia in Parkinson's disease: Why does it occur and how to manage? *Movement Disorders Clinical Practice*, 3(5):443–451, 2016.

M. J. Beal and Z. Ghahramani. Variational Bayesian learning of directed graphical models with hidden variables. *Bayesian Analysis*, 1(4):793–831, 2006.

I. Beinlich, H. Suermondt, R. Chavez, and G. Cooper. The ALARM monitoring system. In *Proceedings of the 2nd European Conference on Artificial Intelligence in Medicine*, pages 689–693, 1992.

A. Ben-Dor, R. Shamir, and Z. Yakhini. Clustering gene expression patterns. *Journal of Computational Biology*, 6(3-4):281–297, 1999.

M. Benjumeda, S. Luengo-Sanchez, P. Larrañaga, and C. Bielza. Tractable learning of Bayesian networks from partially observed data. *Pattern Recognition*, 91:190–199, 2019.

K. Berganzo, B. Tijero, A. González-Eizaguirre, J. Somme, E. Lezcano, I. Gabilondo, M. Fernandez, J. J. Zarranz, and J. C. Gómez-Esteban. Motor and non-motor symptoms of Parkinson's disease and their impact on quality of life and on different clinical subgroups. *Neurología*, 31(9):585–591, 2016.

T. Beuzen, L. Marshall, and K. D. Splinter. A comparison of methods for discretizing continuous variables in Bayesian networks. *Environmental Modelling & Software*, 108:61–66, 2018.

S. Bickel and T. Scheffer. Multi-view clustering. In *Proceedings of the 4th IEEE International Conference on Data Mining*, pages 19–26, 2004.

C. Bielza and P. Larrañaga. Bayesian networks in neuroscience: A survey. *Frontiers in Computational Neuroscience*, 8:131, 2014.

J. Binder, D. Koller, S. Russell, and K. Kanazawa. Adaptive probabilistic networks with hidden variables. *Machine Learning*, 29:213–244, 1997.

C. M. Bishop. *Pattern Recognition and Machine Learning*. Springer, 2016.

C. M. Bishop, N. D. Lawrence, T. Jaakola, and M. I. Jordan. Approximating posterior distributions in belief networks using mixtures. In *Proceedings of the 12th Conference on Neural Information Processing Systems*, pages 416–422, 1998.

R. Blanco, I. Inza, and P. Larrañaga. Learning Bayesian networks in the space of structures by estimation of distribution algorithms. *International Journal of Intelligent Systems*, 18 (2):205–220, 2003.

D. M. Blei, A. Y. Ng, and M. I. Jordan. Latent Dirichlet allocation. *Journal of Machine Learning Research*, 3:993–1022, 2003.

D. M. Blei, A. Kucukelbir, and J. D. McAuliffe. Variational inference: A review for statisticians. *Journal of the American Statistical Association*, 112(518):859–877, 2016.

K. A. Bollen. Latent variables in psychology and the social sciences. *Annual Review of Psychology*, 53(1):605–634, 2002.

R. Caruana, M. Elhawary, N. Nguyen, and C. Smith. Meta clustering. In *Proceedings of the 6th IEEE International Conference on Data Mining*, pages 107–118, 2006.

A. M. Carvalho. Scoring functions for learning Bayesian networks. Technical report, University of Lisbon, 2009.

G. Casella. An introduction to empirical Bayes data analysis. *The American Statistician*, 39 (2):83–87, 1985.

A. Chase. Treatment needs vary between Parkinson disease subtypes. *Nature Reviews Neurology*, 11(3):123–123, 2015.

A. Chaturvedi, P. E. Green, and J. D. Caroll. K-modes clustering. *Journal of Classification*, 18(1):35–55, 2001.

K. R. Chaudhuri, A. Schrag, D. Weintraub, A. Rizos, C. Rodriguez-Blazquez, E. Mamikonyan, and P. Martinez-Martin. The Movement Disorder Society Non-Motor Rating Scale (MDS-NMS): Initial validation study. *Movement Disorders*, 35(1):116–133, 2019.

P. Cheeseman, M. Self, J. Kelly, and J. Stutz. Bayesian classification. In *Proceedings of the 7th AAAI Conference on Artificial Intelligence*, pages 607–611, 1988.

P. Chen, N. L. Zhang, T. Liu, L. K. Poon, Z. Chen, and F. Khawar. Latent tree models for hierarchical topic detection. *Artificial Intelligence*, 250:105–124, 2017a.

T. Chen and N. L. Zhang. Quartet-based learning of hierarchical latent class models: Discovery of shallow latent variables. In *9th International Symposium of Artificial Intelligence and Mathematics*, 2006.

T. Chen, N. L. Zhang, and Y. Wang. Model-based multidimensional clustering of categorical data. *Artificial Intelligence*, 176(1):2246–2269, 2012.

Z. Chen, N. L. Zhang, D. Y. Yeung, and P. Chen. Sparse Boltzmann machines with structure learning. In *Proceedings of the 31st AAAI Conference on Artificial Intelligence*, pages 1805–1811, 2017b.

D. Chickering, D. Geiger, and D. Heckerman. Learning Bayesian networks: Search methods and experimental results. In *Proceedings of the 5th International Workshop on Artificial Intelligence and Statistics*, pages 112–128, 1995.

D. M. Chickering. *Learning from Data*, chapter 12: Learning Bayesian networks is NP-complete, pages 121–130. Springer, 1996.

D. M. Chickering and D. Heckerman. Efficient approximations for the marginal likelihood of Bayesian networks with hidden variables. *Machine Learning*, 29(2-3):181–212, 1997.

D. M. Chickering, D. Heckerman, and C. Meek. Large-sample learning of Bayesian networks is NP-hard. *Journal of Machine Learning Research*, 5:1287–1330, 2004.

A. Choi and A. Darwiche. On the relative expressiveness of Bayesian and neural networks. In *Proceedings of the 9th International Conference on Probabilistic Graphical Models*, pages 157–168, 2018.

M. J. Choi, V. Y. Tan, A. Anandkumar, and A. S. Willsky. Learning latent tree graphical models. *Journal of Machine Learning Research*, 12:1771–1812, 2011.

C. Chow and C. Liu. Approximating discrete probability distributions with dependence trees. *IEEE Transactions on Information Theory*, 14(3):462–467, 1968.

G. Cooper. Probabilistic inference using belief networks is NP-hard. *Artificial Intelligence*, 42:393–405, 1990.

G. F. Cooper and E. Herskovits. A Bayesian method for the induction of probabilistic networks from data. *Machine Learning*, 9(4):309–347, 1992.

T. M. Cover and J. A. Thomas. *Elements of Information Theory*. John Wiley & Sons, 1991.

Y. Cui, X. Z. Fern, and J. G. Dy. Learning multiple nonredundant clusterings. *ACM Transactions on Knowledge Discovery from Data*, 4(3):1–32, 2010.

P. Dagum and M. Luby. Approximating probabilistic inference in Bayesian belief networks is NP-hard. *Artificial Intelligence*, 60(1):141–153, 1993.

R. Daly, Q. Shen, and S. Aitken. Learning Bayesian networks: Approaches and issues. *The Knowledge Engineering Review*, 26(2):99–157, 2011.

X. H. Dang and J. Bailey. A hierarchical information theoretic technique for the discovery of non-linear alternative clusterings. In *Proceedings of the 16th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 118–129, 2010a.

X. H. Dang and J. Bailey. Generation of alternative clusterings using the CAMI apporach. In *Proceedings of the 2010 SIAM International Conference on Data Mining*, pages 118–129, 2010b.

A. Darwiche. Recursive conditioning. *Artificial Intelligence*, 126(1-2):5–41, 2001.

A. Darwiche. *Modeling and Reasoning with Bayesian Networks*. Cambridge University Press, 2009.

D. Dash and M. J. Druzdzel. A hybrid anytime algorithm for the construction of causal models from sparse data. In *Proceedings of the 15th Conference on Uncertainty in Artificial Intelligence*, pages 142–149, 1999.

A. Daud, J. Li, L. Zhou, and F. Muhammad. Knowledge discovery through directed probabilistic models: A survey. *Frontiers of Computer Science in China*, 4(2):280–301, 2010.

S. Dave, D. Weintraub, D. Aarsland, and D. H. Ffytche. Drug and disease effects in Parkinson's psychosis: Revisiting the role of dopamine. *Movement Disorders Clinical Practice*, 7 (1):32–36, 2020.

I. Davidson and Z. Qi. Finding alternative clusterings using constraints. In *8th IEEE International Conference on Data Mining*, pages 773–778, 2008.

L. M. de Campos, J. M. Fernández-Luna, and J. M. Puerta. An iterated local search algorithm for learning Bayesian networks with restarts based on conditional independence tests. *International Journal of Intelligent Systems*, 18(2):221–235, 2003.

A. P. Dempster, N. M. Laird, and D. B. Rubin. Maximum likelihood from incomplete data via the EM algorithm. *Journal of the Royal Statistical Scoeity: Series B*, 39(1):1–38, 1977.

D. Dua and C. Graff. *UCI Machine Learning Repository*. http://archive.ics.uci.edu/ml, 2019.

K. Dujardin, C. Langlois, L. Plomhause, A. S. Carette, M. Delliaux, A. Duhamel, and L. Defebvre. Apathy in untreated early-stage Parkinson disease: Relationship with other nonmotor symptoms. *Movement Disorders*, 29(14):1796–1801, 2014.

R. Durbin, S. R. Eddy, A. Krogh, and G. Mitchison. *Biological Sequence Analysis: Probabilistic Models of Proteins and Nucleic Acids*. Cambridge University Press, 1998.

G. Elidan and N. Friedman. Learning hidden variable networks: The information bottleneck approach. *Journal of Machine Learning Research*, 6:81–127, 2005.

G. Elidan, N. Lotner, N. Friedman, and D. Koller. Discovering hidden variables: A structure-based approach. In *Proceedings of the 14th Conference on Neural Information Processing Systems*, pages 479–485, 2000.

R. Erro, C. Vitale, M. Amboni, M. Picillo, M. Moccia, K. Longo, G. Santangelo, A. D. Rosa, R. Allocca, F. Giordano, G. Orefice, G. D. Michele, L. Santoro, M. T. Pellecchia, and P. Barone. The heterogeneity of early Parkinson's disease: A cluster analysis on newly diagnosed untreated patients. *PLoS One*, 8:e70244, 2013.

R. Erro, M. Picillo, C. Vitale, R. Palladino, M. Amboni, M. Moccia, M. T. Pellecchia, and P. Barone. Clinical clusters and dopaminergic dysfunction in de-novo Parkinson disease. *Parkinsonism & Related Disorders*, 28:137–140, 2016.

A. J. Espay and C. Marras. Clinical Parkinson disease subtyping does not predict pathology. *Nature Reviews Neurology*, 15(4):189–190, 2019.

A. J. Espay, F. Morgante, A. Merloa, A. Fasano, L. Marsili, S. H. Fox, E. Bezard, and B. Pi. Levodopa-induced dyskinesia in Parkinson disease: Current and evolving concepts. *Annals of Neurology*, 84(6):797–811, 2018.

B. S. Everitt. *An Introduction to Latent Variable Models*. Chapman and Hall, 1984.

F. Falck, H. Zhang, M. Willetts, G. Nicholson, C. Yau, and C. C. Holmes. Multi-facet clustering variational autoencoders. *arXiv:2106.05241*, 2021.

E. Faulkner. K2GA: Heuristically guided evolution of Bayesian network structures from data. In *Proceedings of the 2007 IEEE Symposium on Computational Intelligence and Data Mining*, pages 18–25, 2007.

C. Fearon, A. E. Lang, and A. J. Espay. The logic and pitfalls of Parkinson's disease as "brain-first" versus "body-first" subtypes. *Movement Disorders*, 36(3):594–598, 2021.

N. Fenton and M. Neil. *Risk Assesment and Decision Analysis with Bayesian Networks*. CRC Press, 2012.

S. M. Fereshtehnejad and R. B. Postuma. Subtypes of Parkinson's disease: What do they tell us about disease progression? *Current Neurology and Neuroscience Reports*, 17(4):34, 2017.

D. H. Ffytche, B. Creese, M. Politis, K. R. Chaudhuri, D. Weintraub, C. Ballard, and D. Aarsland. The psychosis spectrum in Parkinson disease. *Nature Reviews Neurology*, 13(2):81–95, 2017.

T. M. Fragoso, W. Bertoli, and F. Louzada. Bayesian model averaging: A systematic review and conceptual classification. *International Statistical Review*, 86(1):1–28, 2018.

B. J. Frey and D. Dueck. Clustering by passing messages between data points. *Science*, 315 (5814):972–976, 2007.

J. H. Friedman, J. C. Beck, K. L. Chou, G. Clark, C. P. Fagundes, C. G. Goetz, K. Herlofson, B. Kluger, L. B. Krupp, A. E. Lang, J. Lou, L. Marsh, A. Newbould, and D. Weintraub. Fatigue in Parkinson's disease: Report from a multidisciplinary symposium. *NPJ Parkinson's Disease*, 2(1):1–6, 2016.

N. Friedman. Learning belief networks in the presence of missing values and hidden variables. In *Proceedings of the 14th International Conference on Machine Learning*, pages 125–133, 1997.

N. Friedman. The Bayesian structural EM algorithm. In *Proceedings of the 14th Conference on Uncertainty in Artificial Intelligence*, pages 129–138, 1998.

N. Friedman and M. Goldszmidt. Discretizing continuous attributes. In *Proceedings of the 13th International Conference on Machine Learning*, pages 157–165, 1996.

N. Friedman and I. Nachman. Gaussian process networks. In *Proceedings of the 16th Conference on Uncertainty in Artificial Intelligence*, pages 211–219, 2000.

N. Friedman, M. Linial, I. Nachman, and D. Pe'er. Using Bayesian networks to analyze expression data. *Journal of Computational Biology*, 7(3-4):601–620, 2000.

N. Friedman, O. Mosenzon, N. Slonim, and N. Tishby. Multivariate information bottleneck. In *Proceedings of the 17th Conference on Uncertainty in Artificial Intelligence*, pages 152–161, 2001.

L. D. Fu. *A Comparison of State-of-the-Art Algorithms for Learning Bayesian Network Structure from Continuous Data*. Master's thesis, Vanderbilt University, 2005.

A. Gain and I. Shpitser. Structure learning under missing data. In *Proceedings of the 9th International Conference on Probabilistic Graphical Models*, pages 121–132, 2018.

G. Galimberti, A. Manisi, and G. Soffritti. Modelling the role of variables in model-based cluster analysis. *Statistics and Computing*, 28(1):145–169, 2018.

E. M. Gatto and V. Aldinio. Impulse control disorders in Parkinson's disease. A brief and comprehensive review. *Frontiers in Neurology*, 10:351, 2019.

D. Geiger and D. Heckerman. Learning Gaussian networks. In *Proceedings of the 10th Conference on Uncertainty in Artificial Intelligence*, pages 235–243, 1994.

D. Geiger, T. Verma, and J. Pearl. Identifying independence in Bayesian networks. *Networks*, 20(5):507–534, 1990.

A. Gelman, J. B. Carlin, H. S. Stern, D. B. Dunson, A. Vehtari, and D. B. Rubin. *Bayesian Data Analysis*. CRC Press, 2013.

J. A. Gámez, J. L. Mateo, and J. M. Puerta. Learning Bayesian networks by hill climbing: Efficient methods based on progressive restriction of the neighborhood. *Data Mining and Knowledge Discovery*, 22(1):106–148, 2011.

C. G. Goetz, B. C. Tilley, S. R. Shaftman, G. T. Stebbins, S. Fhan, P. Martinez-Martin, W. Poewe, C. Sampaio, M. B. Stern, R. Dodel, B. Dubois, R. Holloway, J. Jankovic, J. Kulisevsky, A. E. Lang, A. Lees, S. Leurgans, P. A. LeWitt, D. Nuenhuis, C. W. Olanow, O. Rascol, A. Schrag, J. A. Teresi, and J. J. an Hilten an N. LaPelle. Movement Disorder Society-sponsored revision of the Unified Parkinson's Disease Rating Scale (MDS-UPDRS): Scale presentation and clinimetric testing results. *Movement Disorders*, 23(15):2129–2170, 2008.

D. Gondek and T. Hofmann. Conditional information bottleneck clustering. In *Workshop Proceedings of the 3rd IEEE International Conference on Data Mining*, pages 36–42, 2003.

D. Gondek and T. Hofmann. Non-redundant data clustering. *Knowledge and Information Systems*, 12(1):1–24, 2007.

A. D. Gordon. A review of hierarchical classification. *Journal of the Royal Statistical Society: Series A*, 150(2):119–137, 1987.

J. C. Greenland, C. H. Williams-Gray, and R. A. Barker. The clinical heterogeneity of Parkinson's disease and its therapeutic implications. *European Journal of Neuroscience*, 49(3):328–338, 2019.

Y. Guan, J. G. Dy, and D. Ghahramani. Variational inference for nonparametric multiple clusterings. In *Proceedings of the 1st International Workshop on Discovering, Summarizing and Using Multiple Clusterings*, pages 1–8, 2010.

S. Guindon and O. Gascuel. A simple, fast, and accurate algorithm to estimate large phylogenies by maximum likelihood. *Systematic Biology*, 52(5):696–704, 2003.

S. Harmeling and C. K. Williams. Greedy learning of binary latent trees. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 33(6):1087–1097, 2010.

D. Heckerman and D. Geiger. Learning Bayesian networks: A unification for discrete and Gaussian domains. In *Proceedings of the 11th Conference on Uncertainty in Artificial Intelligence*, pages 274–284, 1995.

D. Heckerman, D. Geiger, and D. M. Chickering. Learning Bayesian networks: The combination of knowledge and statistical data. *Machine Learning*, 20(3):197–243, 1995.

M. Henrion. Propagating uncertainty in Bayesian networks by probabilistic logic sampling. In *Proceedings of the 2nd Conference on Uncertainty in Artificial Intelligence*, pages 149–163, 1986.

Y. H. Hiorth, K. F. Pedersen, I. Dalen, O. B. Tysnes, and G. Alves. Orthostatic hypotension in Parkinson disease: A 7-year prospective population-based study. *Neurology*, 93(16): 1526–1534, 2019.

M. M. Hoehn and M. D. Yahr. Parkinsonism: Onset, progression and mortality. *Neurology*, 17:427–442, 1967.

J. A. Hoeting, D. Madigan, A. E. Raftery, and C. T. Volinsky. Bayesian model averaging: A tutorial. *Statistical Science*, pages 382–401, 1999.

R. Hofmann and V. Tresp. Discovering structure in continuous variables using Bayesian networks. In *Proceedings of the 9th Conference on Neural Information Processing Systems*, pages 500–506, 1995.

D. G. Horvitz and D. J. Thompson. A generalization of sampling without replacement from a finite universe. *Journal of the American Statistical Association*, 47(260):663–685, 1952.

F. Höppner, F. Klawonn, R. Druse, and T. Runkler. *Fuzzy Cluster Analysis: Methods for Classification, Data Analysis and Image Recognition.* John Wiley & Sons, 1999.

T. Hrycej. Gibbs sampling in Bayesian networks. *Artificial Intelligence*, 46:351–363, 1990.

W. H. Hsu, H. Guo, B. B. Perry, and J. A. Stilson. A permutation genetic algorithm for variable ordering in learning Bayesian networks from data. In *Proceedings of the 4th Annual Conference on Genetic and Evolutionary Computation*, pages 383–390, 2002.

F. Huang, I. Perros, R. Chen, J. Sun, and A. Anandkumar. Guaranteed scalable learning of latent tree models. In *Proceedings of the 35th Conference on Uncertainty in Artificial Intelligence*, pages 883–893, 2020.

K. B. Hwang, J. W. Lee, S. W. Chung, and B. T. Zhang. Construction of large-scale Bayesian networks by local to global search. In *Proceedings of the 7th Pacific Rim International Conference on Artificial Intelligence*, pages 375–384, 2002.

P. Jain, R. Meka, and I. S. Dhillon. Simultaneous unsupervised learning of disparate clusterings. *Statistical Analysis and Data Mining*, 1(3):195–210, 2008.

H. Jelodar, Y. Wang, C. Yuan, X. Feng, X. Jiang, Y. Li, and L. Zhao. Latent Dirichlet allocation (LDA) and topic modeling: Models, applications, a survey. *Multimedia Tools and Applications*, 78(11):15169–15211, 2019.

F. V. Jensen, S. L. Lauritzen, and K. G. Olesen. Bayesian updating in recursive graphical models. *Computational Statistics Quarterly*, 4:269–282, 1990.

B. Jones, I. Jenkinson, Z. Yang, and J. Wang. The use of Bayesian network modelling for maintenance planning in a manufacturing industry. *Reliability Engineering & System Safety*, 95(3):267–277, 2010.

M. I. Jordan, Z. Ghahramani, T. Jaakola, and L. Saul. Introduction to variational methods for graphical models. *Machine Learning*, 37:183–223, 1999.

S. Kaltwang, S. Todorovic, and M. Pantic. Latent trees for estimating intensity of facial action units. In *Proceedings of the 2015 IEEE Conference on Computer Vision and Pattern Recognition*, pages 296–304, 2015.

R. Kidermann and J. L. Snell. *Markov Random Fields and their Applications.* American Mathematical Society, 1980.

D. W. Kim, S. Ko, and B. Y. Kang. Structure learning of Bayesian networks by estimation of distribution algorithms with transpose mutation. *Journal of Applied Research and Technology*, 11(4):586–596, 2013.

J. Kim and J. Pearl. A computational model for causal and diagnostic reasoning in inference systems. In *Proceedings of the 7th International Joint Conference on Artificial Intelligence*, pages 190–193, 1983.

D. Koller and N. Friedman. *Probabilistic Graphical Models: Principles and Techniques.* The MIT Press, 2009.

D. Koller, U. Lerner, and D. Angelov. A general algorithm for approximate inference an its application to hybrid Bayes nets. In *Proceedings of the 15th Conference on Uncertainty in Artificial Intelligence*, pages 324–333, 1999.

S. Kotz, N. Balakrishnan, and N. L. Johnson. *Continuous Multivariate Distributions, Models and Applications*. John Wiley & Sons, 2004.

A. Kraskov, H. Stögbauer, and P. Grassberger. Estimating mutual information. *Physics Review E*, 69(6):066138, 2004.

H. P. Kriegel and A. Zimek. Ensemble clustering, alternative clustering, multi-view clustering: What can we learn from each other. In *Proceedings of the 1st International Workshop on Discovering, Summarizing and Using Multiple Clusterings*, pages 9–14, 2010.

H. P. Kriegel, P. Kröger, and A. Zimek. Clustering high-dimensional data: A survey on subspace clustering, pattern-based clustering, and correlation clustering. *ACM Transactions on Knowledge Discovery from Data*, 3(1):1–58, 2009.

W. H. Kruskal and W. A. Wallis. Use of ranks in one-criterion variance analysis. *Journal of the American Statistical Association*, 47(260):583–621, 1952.

J. Kwisthout, H. L. Bodlaender, and L. C. van der Gaag. The necessity of bounded treewidth for efficient inference in Bayesian networks. In *Proceedings of the 19th European Conference on Artificial Intelligence*, pages 237–242, 2010.

J. A. Lake. Reconstructing evolutionary trees from DNA and protein sequences: Paralinear distances. *Proceedings of the National Academy of Sciences*, 91(4):1455–1459, 1994.

W. Lam and F. Bacchus. Learning Bayesian belief networks: An approach based on the MDL principle. *Computational Intelligence*, 10(3):269–293, 1994.

H. Langseth and T. D. Nielsen. Latent classification models for binary data. *Pattern Classification*, 42(11):2724–2736, 2009.

H. Langseth, T. D. Nielsen, R. Rumí, and A. Salmerón. Mixtures of truncated basis functions. *International Journal of Approximate Reasoning*, 53:212–227, 2012.

H. Langseth, T. D. Nielsen, I. Pérez-Bernabé, and A. Salmerón. Learning mixtures of truncated basesi functions from data. *International Journal of Approximate Reasoning*, 55(4): 940–956, 2014.

P. Larrañaga, C. M. H. Kuijpers, R. H. Murga, and Y. Yurramendi. Learning Bayesian network structures by searching for the best ordering with genetic algorithms. *IEEE Transactions on Systems, Man, and Cybernetics, Part A: Systems and Humans*, 26(4):487–493, 1996a.

P. Larrañaga, R. Murga, M. Poza, and C. Kuijpers. *Learning from Data*, chapter 16: Structure learning of Bayesian networks by hybrid genetic algorithms, pages 165–174. Springer, 1996b.

S. L. Lauritzen. Propagation of probabilities, means, and variances in mixed graphical association models. *Journal of the American Statistical Association*, 87(420):1098–1108, 1992.

S. L. Lauritzen and D. J. Spiegelhalter. Local computations with probabilities on graphical structures and their application to expert systems. *Journal of Royal Statistical Society, Series B*, 50(2):157–224, 1988.

S. L. Lauritzen and N. Wermuth. Graphical models for associations between variables, some of which are qualitatitve and some quantitative. *The Annals of Statistics*, pages 31–57, 1989.

P. F. Lazarsfeld and N. W. Henry. *Latent Structure Analysis*. Houghton Mifflin, 1968.

D. H. Lee, J. S. Oh, J. H. Ham, J. J. Lee, I. Lee, P. H. Lee, J. S. Kim, and Y. H. Sohn. Is normosmic Parkinson disease a unique clinical phenotype? *Neurology*, 85(15):1270–1275, 2015.

A. Lenka, J. Pagonabarraga, P. K. Pal, H. Bejr-Kasem, and J. Kulisvesky. Minor hallucinations in Parkinson disease: A subtle symptom with major clinical implications. *Neurology*, 93(6):259–266, 2019.

U. Lerner and R. Parr. Inference in hybrid networks: Theoretical limits and practical algorithms. In *Proceedings of the 17th Conference on Uncertainty in Artificial Intelligence*, pages 310–318, 2001.

V. Leta, D. Urso, L. Batzu, D. Weintraub, N. Titova, D. Aarsland, P. Martinez-Martin, P. Borghammer, D. J. van Wamelen, T. Yousaf, A. Rizos, C. Rodriguez-Blazquez, G. Chung-Faye, and K. R. Chaudhuri. Constipation is associated with development of cognitive impairment in de novo Parkinson's disease: A longitudinal analysis of two international cohorts. *Journal of Parkinson's Disease*, 11(3):1209–1219, 2021.

Q. Li and J. Racine. Nonparametric estimation of distributions with categorical and continuous data. *Journal of Multivariate Analysis*, 86(2):266–292, 2003.

X. Li, Z. Chen, L. K. Poon, and N. L. Zhang. Learning latent superstructures in variational autoencoders for deep multidimensional clustering. In *7th International Conference on Learning Representations*, 2019.

R. J. A. Little and D. B. Rubin. *Statistical Analysis with Missing Data*. John Wiley & Sons, 1987.

H. Liu, F. Hussain, C. L. Tan, and M. Dash. Discretization: An enabling technique. *Data Mining and Knowledge Discovery*, 6(4):393–432, 2002.

T. Liu, N. L. Zhang, and P. Chen. Hierarchical latent tree analysis for topic detection. In *Proceedings of the 7th Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pages 256–272, 2014.

T. F. Liu, N. L. Zhang, P. Chen, A. H. L. L. K. Poon, and Y. Wang. Greedy learning of latent tree models for multidimensional clustering. *Machine Learning*, 98(1-2):301–330, 2015.

P. L. López-Cruz, T. D. Nielsen, C. Bielza, and P. Larrañaga. Learning mixtures of polynomials of conditional densities from data. In *Proceedings of the 15th MultiConference of the Spanish Association for Artificial Intelligence*, pages 363–372, 2013.

P. L. López-Cruz, C. Bielza, and P. Larrañaga. Learning mixtures of polynomials from data using B-spline interpolation. *International Journal of Approximate Reasoning*, 55(4): 989–1010, 2014.

K. Ma, N. Xiong, Y. Shen, C. Han, L. Liu, G. Zhang, L. Wang, S. Guo, X. Guo, Y. Xia, F. Wan, J. Huang, Z. Lin, and T. Wang. Weight loss and malnutrition in patients with Parkinson's disease: Current knowledge and future prospects. *Frontiers in Aging Neuroscience*, 10:1, 2018.

L. Y. Ma, P. Chan, Z. Q. Gu, F. F. Li, and T. Feng. Heterogeneity among patients with Parkinson's disease: Cluster analysis and genetic association. *Journal of the Neurological Sciences*, 351(1-2):41–45, 2015.

M. Maathuis, M. Drton, S. Lauritzen, and M. Wainwright. *Handbook of Graphical Models*. CRC Press, 2018.

D. J. C. MacKay. *Bayesian Methods for Adaptive Models*. PhD thesis, California Institute of Technology, 1991.

J. MacQueen. Some methods for classification and analysis of multivariate observations. In *Proceedings of the 5th Berkeley Symposium on Mathematical Statistics and Probability*, pages 281–297, 1967.

H. B. Mann and D. R. Whitney. On a test of whether one of two random variables is stochastically larger than the other. *Annals of Mathematical Statistics*, 18(1):50–60, 1947.

D. Margaritis. *Learning Bayesian Network Model Structure from Data*. PhD thesis, Carnegie-Mellon University, 2003.

C. Marras. Subtypes of Parkinson's disease: State of the field and future directions. *Current Opinion in Neurology*, 28(4):382–386, 2015.

C. Marras, K. R. Chaudhuri, N. Titova, and T. A. Mestre. Therapy of Parkinson's disease subtypes. *Neurotherapeutics*, 17:1366–1377, 2020.

P. Martinez-Martin, J. M. Rojo-Abuín, D. Weintraub, K. R. Chaudhuri, C. Rodríguez-Blázquez, A. Rizos, and A. Schrag. Factor analysis and clustering of the movement disorder society non-motor rating scale. *Movement Disorders*, 35(6):969–975, 2020.

A. R. Masegosa, A. Martínez, D. Ramos-López, R. Cabañas, A. Salmerón, H. Langseth, T. D. Nielsen, and A. L. Madsen. AMIDST: A Java toolbox for scalable probabilistic machine learning. *Knowledge-Based Systems*, 163:595–597, 2019.

D. Mautz, W. Ye, C. Plant, and C. Böhm. Non-redundant subspace clusterings with Nr-Kmeans and Nr-DipMeans. *ACM Transactions on Knowledge Discovery from Data*, 14(5): 1–24, 2020.

S. Mavandadi, S. Nazem, T. R. T. Have, A. D. Siderowf, J. E. Duda, M. B. Stern, and D. Weintraub. Use of latent variable modeling to delineate psychiatric and cognitive profiles in Parkinson disease. *The American Journal of Geriatric Psychiatry*, 17(11):986–995, 2009.

G. McLachlan and T. Krishnan. *The EM Algorithm and Extensions*. John Wiley & Sons, 2008.

G. McLachlan and D. Peel. *Finite Mixture Models*. John Wiley & Sons, 2004.

C. Meek. Causal inference and causal explanation with background knowledge. In *Proceedings of the 11th Conference on Uncertainty in Artificial Intelligence*, pages 403–410, 1995.

V. Melnykov and R. Maitra. Finite mixture models and model-based clustering. *Statistics Surveys*, 4:80–116, 2010.

T. A. Mestre, S. M. Fereshtehnejad, D. Berg, N. I. Bohen, K. Dujardin, R. Erro, A. J. Espay, G. Halliday, J. J. van Hilten, M. T. Hu, B. Jeon, C. Klein, A. F. G. Leentjens, J. Marinus, B. Mollenhauer, R. Postuma, R. Rajalingam, M. Rdríguez-Violante, T. Simuni, D. J. Surmeier, D. Weintraub, M. P. McDermott, M. Lawton, and C. Marras. Parkinson's disease subtypes: Critical appraisal and recommendations. *Journal of Parkinson's Disease*, 11(2):395–404, 2021.

T. P. Minka. Expectation propagation for approximate Bayesian inference. In *Proceedings of the 17th Conference on Uncertainty in Artificial Intelligence*, pages 362–369, 2001.

A. Molina, A. Vergari, N. D. Mauro, S. Natarajan, F. Esposito, and K. Kersting. Mixed sum-product networks: A deep architecture for hybrid domains. In *Proceedings of the 32nd AAAI Conference on Artificial Intelligence*, pages 3828–3835, 2018.

S. Monti. *Learning Hybrid Bayesian Networks from Data*. PhD thesis, University of Pittsburgh, 1999.

S. Monti and G. F. Cooper. Learning Bayesian belief neworks with neural network estimators. In *Proceedings of the 11th Conference on Neural Information Processing Systems*, pages 578–584, 1997.

Y. I. Moon, B. Rajagopalan, and U. Lall. Estimation of mutual information using kernel density estimators. *Physics Review E*, 52(3):2318–2321, 1995.

S. Moral, R. Rumí, and A. Salmerón. Mixtures of truncated exponentials in hybrid Bayesian networks. In *Proceedings of the 6th European Conference on Symbolic and Quantitative Approaches to Reasoning and Uncertainty*, pages 156–167, 2001.

R. Mourad, C. Sinoquet, and P. Leray. A hierarchical Bayesian network approach for linkage disequilibrium modeling and data-dimensionality. *BMC Bioinformatics*, 12(16):1–20, 2011.

R. Mourad, C. Sinoquet, N. L. Zhang, T. Liu, and P. Leray. A survey on latent tree models and applications. *Journal of Artificial Intelligence Research*, 47:157–203, 2013.

J. Mu, K. R. Chaudhuri, C. Bielza, J. Pedro-Cuesta, P. Larrañaga, and P. Martinez-Martin. Parkinson's disease subtypes identified from cluster analysis of motor and non-motor symptoms. *Frontiers in Aging Neuroscience*, 9:301, 2017.

K. Murphy. A variational approximation for Bayesian networks with discrete and continuous latent variables. In *Proceedings of the 15th Conference on Uncertainty in Artificial Intelligence*, pages 457–466, 1999.

K. P. Murphy, Y. Weiss, and M. I. Jordan. Loopy belief propagation for approximate inference: An empirical study. In *Proceedings of the 15th Conference on Uncertainty in Artificial Intelligence*, pages 467–475, 1999.

A. Nazabal, P. M. Olmos, Z. Ghahramani, and I. Valera. Handling incomplete heterogeneous data using VAES. *Pattern Recognition*, 107:107501, 2020.

T. Nimmagadda and A. Anandkumar. Multi-object classification and unsupervised scene understanding using deep learning features and latent tree probabilistic models. *arXiv:1505.00308*, 2015.

D. Niu, J. Dy, and Z. Ghahramani. A nonparametric Bayesian model for multiple clusterings with overlapping feature views. In *Proceedings of the 15th International Conference on Artificial Intelligence and Statistics*, pages 814–822, 2012.

D. Niu, J. G. Dy, and M. I. Jordan. Iterative discovery of multiple alternative clustering views. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 36(7):1340–1353, 2014.

F. Nojavan, S. S. Qian, and C. A. Stow. Comparative analysis of discretization methods in Bayesian networks. *Environmental Modelling & Software*, 87:64–71, 2017.

J. A. Palma and H. Kaufmann. Treatment of autonomic dysfunction in Parkinson disease and other synuclinopathies. *Movement Disorders*, 33(3):372–390, 2018.

J. M. Peña, J. A. Lozano, and P. Larrañaga. Learning Bayesian networks for clustering by means of constructive induction. *Pattern Recognition Letters*, 20(11):1219–1230, 1999.

J. M. Peña, J. A. Lozano, and P. Larrañaga. An improved Bayesian structural EM algorithm for learning Bayesian networks for clustering. *Pattern Recognition Letters*, 21(8):779–786, 2000.

J. M. Peña, J. A. Lozano, and P. Larrañaga. Learning recursive Bayesian multinets for data clustering by means of constructive induction. *Machine Learning*, 47(1):63–89, 2002.

J. Pearl. A constraint-propagation approach to probabilistic reasoning. In *Proceedings of the 2nd Conference on Uncertainty in Artificial Intelligence*, pages 357–370, 1986.

J. Pearl. Evidential reasoning using stochastic simulation of causal models. *Artificial Intelligence*, 32:245–257, 1987.

J. Pearl. *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference*. Morgan Kaufmann, 1988.

K. Pearson. On the criterion that a given system of deviations from the probable in the case of a correlated system of variables is such that it can be reasonably supposed to have arisen from random sampling. *The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science*, 50(302):157–175, 1900.

D. T. Pham and G. A. Ruz. Unsupervised training of Bayesian networks for data clustering. *Proceedings of the Royal Society A: Mathematical, Physical and Engineering Sciences*, 465 (2019):2927–2948, 2009.

C. Pont-Sunyer, A. Hotter, C. Gaig, K. Seppi, Y. Compta, R. Ktzenschlager, N. Mas, D. Hofeneder, T. Brücke, A. Bayés, K. Wenzel, J. Infante, H. Zach, W. Pirker, P. I. J, R. Álvarez, L. Ispierto, O. D. Fàbregues, A. Callén, A. Palasí, M. Aguilar, M. J. Martí, F. Valldeoriola, M. Salamero, W. Poewe, and E. Tolosa. The onset of nonmotor symptoms in Parkinson's disease The ONSET PD study. *Movement Disorders*, 30(2):229–237, 2015.

H. Poon and P. Domingos. Sum-product networks: A new deep architecture. In *Proceedings of the 27th Conference on Uncertainty in Artificial Intelligence*, pages 337–346, 2011.

L. K. Poon, N. L. Zhang, and A. H. Liu. Model-based clustering of high-dimensional data: Variable selection versus facet determination. *International Journal of Approximate Reasoning*, 54(1):196–215, 2013.

L. K. Poon, A. H. Liu, and N. L. Zhang. UC-LTM: Unidimensional clustering using latent tree models for discrete data. *Journal of Approximate Reasoning*, 92:392–409, 2018.

O. Pourret, P. Naïm, and B. Marcot. *Bayesian Networks: A Practical Guide to Applications*. John Wiley & Sons, 2008.

Z. Qi and I. Davidson. A principled and flexible framework for finding alternative clusterings. In *Proceedings of the 15th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 717–726, 2009.

E. Qian and Y. Huang. Subtyping of Parkinson's disease - Where are we up to? *Aging and Disease*, 10(5):1130, 2019.

S. Rabe-Hesketh and A. Skrondal. Classical latent variable models for medical research. *Statistical Methods in Medical Research*, 17(1):5–32, 2008.

N. Robertson and P. Seymour. Graph minors. II. Algorithmic aspects of tree-width. *Journal of Algorithms*, 7(3):309–322, 1986.

J. M. Robins, A. Rotnitzky, and L. P. Zhao. Estimation of regression coefficients when some regressors are not always observed. *Journal of the American Statistical Association*, 89 (427):846–866, 1994.

F. Rodriguez-Sanchez, P. Larrañaga, and C. Bielza. Discrete model-based clustering with overlapping subsets of attributes. In *Proceedings of the 9th International Conference on Probabilistic Graphical Models*, pages 392–403, 2018.

F. Rodriguez-Sanchez, P. Larrañaga, and C. Bielza. Incremental learning of latent forests. *IEEE Access*, 8:224420–224432, 2020.

F. Rodriguez-Sanchez, C. Bielza, and P. Larrañaga. Multi-partition clustering of mixed data with Bayesian networks. Submitted, 2021a.

F. Rodriguez-Sanchez, C. Rodriguez-Blazquez, C. Bielza, P. Larrañaga, D. Weintraub, P. Martinez-Martin, A. Rizos, A. Schrag, and K. R. Chaudhuri. Identifying Parkinson's disease subtypes with motor and non-motor symptoms via model-based multi-partition clustering. Submitted, 2021b.

M. Rossi, A. M. Escobar, A. Bril, P. M. Vernetti, J. I. D. Palo, D. Cerquetti, and M. Merello. Motor features in Parkinson's disease with normal olfactory function. *Movement Disorders*, 31(9):1414–1417, 2016.

N. Saitou and M. Nei. The neighbor-joining method: A new method for reconstructing phylogenetic trees. *Molecular Biology and Evolution*, 4(4):406–425, 1987.

A. Salmeron, R. Rumí, H. Langseth, T. D. Nielsen, and A. L. Madsen. A review of inference algorithms for hybrid Bayesian networks. *Journal of Artificial Intelligence Research*, 62: 799–828, 2018.

G. Santafé, J. A. Lozano, and P. Larrañaga. Bayesian model averaging of naïve Bayes for clustering. *IEEE Transactions on Systems, Man, and Cybertetics, Part B (Cybernetics)*, 36(5):1149–1161, 2006a.

G. Santafé, J. A. Lozano, and P. Larrañaga. Bayesian model averaging of TAN models for clustering. In *Proceedings of the 3rd European Workshop on Probabilistic Graphical Models*, pages 271–278, 2006b.

A. Sauerbier, P. Jenner, A. Todorova, and K. R. Chaudhuri. Non-motor subtypes and Parkinson's disease. *Parkinsonism & Related disorders*, 22:41–46, 2015.

L. K. Saul, T. Jaakkola, and M. I. Jordan. Mean field theory for sigmoid belief networks. *Journal of Artificial Intelligence Research*, 4:61–76, 1996.

M. Scanagatta, G. Corani, and M. Zaffalon. Improved local search in Bayesian networks structure learning. In *Proceedings of the 3rd International Workshop on Advanced Methodologies for Bayesian Networks*, pages 45–56, 2017.

M. Scanagatta, G. Corani, M. Zaffalon, J. Yoo, and U. Kang. Efficient learning of bounded-treewith Bayesian networks from complete and incomplete data sets. *Journal of Approximate Reasoning*, 95:152–166, 2018.

M. Scanagatta, A. Salmerón, and F. Stella. A survey on Bayesian network structure learning from data. *Progress in Artificial Intelligence*, 8(4):425–439, 2019.

A. H. V. Schapira, K. R. Chaudhuri, and P. Jenner. Non-motor features of Parkinson disease. *Nature Reviews Neuroscience*, 18(7):435–450, 2017.

G. Schwarz. Estimating the dimension of a model. *The Annals of Statistics*, 6(2):461–464, 1978.

B. M. Scott, R. S. Eisinger, M. R. Burns, J. Lopes, M. S. Okun, A. Gunduz, and D. Bowers. Co-occurrence of apathy and impulse control disorders in Parkinson disease. *Neurology*, 95 (20):2769–2780, 2020.

D. Scott. *Multivariate Density Estimation: Theory, Practice and Visualization*. John Wiley & Sons, 1992.

R. Shachter, B. D. D'Ambrosio, and B. del Favero. Symbolic probabilistic inference in belief networks. In *Proceedings of the 6th Conference on Uncertainty in Artificial Intelligence*, pages 126–131, 1990.

R. Shachter, S. K. Andersen, and P. Szolovits. Global conditioning for probabilistic inference in belief networks. In *Proceedings of the 10th Conference on Uncertainty in Artificial Intelligence*, pages 514–522, 1994.

R. D. Shachter and C. R. Kenley. Gaussian influence diagrams. *Management Science*, 35(5): 527–550, 1989.

R. D. Shachter and M. A. Peot. Simulation approaches to general probabilistic inference on belief networks. *Machine Intelligence and Pattern Recognition*, 10:221–231, 1989.

P. P. Shenoy and G. Shafer. Axioms for probability and belief-function propagation. In *Proceedings of the 6th Conference on Uncertainty in Artificial Intelligence*, pages 169–198, 1990.

P. P. Shenoy and J. C. West. Inference in hybrid Bayesian networks using mixtures of polynomials. *International Journal of Approximate Reasoning*, 52(5):641–657, 2011.

M. Siciliano, L. Trojano, G. Santagelo, R. D. Micco, G. Tedeshi, and A. Tessitore. Fatigue in Parkinson's disease: A systematic review and meta-analysis. *Movement Disorders*, 33 (11):1712–1723, 2018.

M. Singh and M. Valtorta. An algorithm for the construction of Bayesian network structures from data. In *Proceedings of the 9th Conference on Uncertainty in Artificial Intelligence*, pages 259–265, 1993.

M. Singh and M. Valtorta. Construction of Bayesian network structures from data: A brief survey and an efficient algorithm. *International Journal of Approximate Reasoning*, 12(2): 111–131, 1995.

D. J. Spiegelhalter and S. L. Lauritzen. Sequential updating of conditional probabilities on directed graphical structures. *Networks*, 20(5):579–605, 1990.

P. Spirtes, C. Glymour, R. Scheines, D. Heckerman, C. Meek, G. Cooper, and T. Richardson. *Causation, Prediction and Search*. The MIT Press, 2000.

G. T. Stebbins, C. G. Goetz, D. J. . Burn, J. Jankovic, T. K. Khoo, and B. C. Tilley. How to identify tremor dominant and postural instability/gait difficulty groups with the movement disorder society unified Parkinson's disease rating scale: comparison with the unified Parkinson's disease rating scale. *Movement Disorders*, 28(5):668–670, 2013.

M. Steel. The complexity of reconstructing trees from qualitative characters and subtrees. *Journal of Classification*, 9:91–116, 1992.

F. Stocchi, G. Abbruzzese, R. Ceravolo, P. Cortelli, M. D'Amelio, M. F. D. Pandis, G. Fabbrini, C. Pacchetti, G. Pezzoli, A. Tessitore, M. Cansei, C. Iannacone, and M. Zappia. Prevalence of fatigue in Parkinson disease and its clinical correlates. *Neurology*, 83(3): 215–220, 2014.

A. Strehl and J. Ghosh. Cluster ensembles - A knowledge reuse framework for combining multiple partitions. *Journal of Machine Learning Research*, 3:583–617, 2003.

L. E. Sucar. *Probabilistic Graphical Models: Principles and Applications*. Springer, 2015.

X. Sui, C. Zhou, J. Li, L. Chen, X. Yang, and F. Li. Hyposmia as a predictive marker of Parkinson's disease: A systematic review and meta-analysis. *BioMed Research International*, 19:1–9, 2019.

J. Suzuki. Learning Bayesian belief networks based on the MDL principle: An efficient algorithm using the branch and bound technique. *IEICE Transactions on Information and Systems*, 82(2):356–367, 1999.

J. Suzuki. Branch and bound for continuous Bayesian network structure learning. In *Workshop Proceedings of the 9th International Conference on Probabilistic Graphical Models*, pages 49–60, 2018.

M. Teyssier and D. Koller. Ordering-based search: A simple and effective algorithm for learning Bayesian networks. In *Proceedings of the 21st Conference on Uncertainty in Artificial Intelligence*, pages 584–590, 2005.

M. A. Thenganatt and J. Jankovic. Parkinson disease subtypes. *JAMA Neurology*, 71(4): 499–504, 2014.

N. Tishby, F. Pereira, and W. Bialek. The information bottleneck method. In *Proceedings of the 37th Allerton Conference on Communication, Control and Computation*, pages 368–377, 1999.

I. Tsamardinos, C. F. Aliferis, and A. Statnikov. Algorithms for large scale Markov blanket discovery. In *Proceedings of the 16th International Florida Artificial Intelligence Reasearch Society Conference*, pages 376–381, 2003.

I. Tsamardinos, L. E. Brown, and C. F. Aliferis. The max-min hill-climbing Bayesian network structure learning algorithm. *Machine Learning*, 65(1):31–78, 2006.

J. Tukey. Comparing individual means in the analysis of variance. *Biometrics*, 5(2):99–114, 1949.

I. Valera, M. F. Pradier, M. Lomeli, and Z. Ghahramani. General latent feature models for heterogeneous datasets. *Journal of Machine Learning Research*, 21(100):1–49, 2020.

L. C. van der Gaag, S. Renooji, C. L. Witteman, and B. Aleman. Probabilities for a probabilistic network: A case study in oesophageal cancer. *Artificial Intelligence in Medicine*, 25(2):123–148, 2002.

P. van der Putten and M. van Someren. A bias-variance analysis of a real world learning problem: The CoIL challenge 2000. *Machine Learning*, 57(1-2):177–195, 2004.

S. M. van Rooden, F. Colas, P. Martinez-Martin, M. Visser, D. Verbaan, J. Marinus, K. R. Chaudhuri, J. N. Kok, and J. J. van Hilten. Clinical subtypes of Parkinson's disease. *Movement Disorders*, 26(1):51–58, 2011.

D. J. van Wamelen, V. Leta, A. M. Podlewska, Y. M. Wan, K. Krbot, E. Jaakola, P. Martinez-Martin, A. Rizos, M. Parry, V. Metta, and K. R. Chaudhuri. Exploring hyperhidrosis and related thermoregulatory symptoms as a possible clinical identifier for the dysautonomic subtype of Parkinson's disease. *Journal of Neurology*, 266(7):1736–1742, 2019.

S. Vega-Pons and J. Ruiz-Shulcloper. A survey of clustering ensemble algorithms. *International Journal of Pattern Recognition and Artificial Intelligence*, 25(3):337–372, 2011.

T. Verma and J. Pearl. Equivalence and synthesis of causal models. In *Proceedings of the 6th Conference on Uncertainty in Artificial Intelligence*, pages 255–269, 1990.

M. J. Wainwright and M. I. Jordan. *Graphical Models, Exponential Families and Variational Inference*. Now Publishers Inc., 2008.

C. Wang and D. M. Blei. Variational inference in nonconjugate models. *Journal of Machine Learning Research*, 14:1005–1031, 2013.

Y. Wang. *Latent Tree Models for Multivariate Density Estimation Algorithms and Applications*. PhD thesis, Hong Kong University of Science and Technology, 2009.

Y. Wang, N. L. Zhang, and T. Chen. Latent tree models and approximate inference in Bayesian networks. *Journal of Artificial Intelligence Research*, 32:879–900, 2008.

Y. Wang, N. L. Zhang, T. Chen, and L. K. Poon. LTC: A latent tree approach to classification. *International Journal of Approximate Reasoning*, 54(4):560–572, 2013.

D. Weintraub, A. S. David, A. H. Evans, J. E. Grant, and M. Stacy. Clinical spectrum of impulse control disorders in Parkinson's disease. *Movement Disorders*, 30(2):121–127, 2015.

J. Winn and C. M. Bishop. Variational message passing. *Journal of Machine Learning Research*, 6:661–694, 2005.

M. L. Wong, W. Law, and K. S. Leung. Using evolutionary programming and minimum description length principle for data mining of Bayesian networks. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 21(2):174–178, 1999.

R. Xu and D. Wunsch. *Clustering*. John Wiley & Sons, 2008.

S. Yang and L. Zhang. Non-redundant multiple clustering by nonegative matrix factorization. *Machine Learning*, 106(5):695–712, 2017.

Y. Yang and H. Wang. Multi-view clustering: A survey. *Big Data Mining and Analytics*, 1 (2):83–107, 2018.

W. Ye, S. Maurus, N. Hubig, and C. Plant. Generalized independent subspace clustering. In *Proceedings of the 16th IEEE International Conference on Data Mining*, pages 569–578, 2016.

J. S. Yedidia, W. T. Freeman, and Y. Weiss. Generalized belief propagation. In *Proceedings of the 14th Conference on Neural Information Processing Systems*, pages 689–695, 2000.

J. York. Use of the Gibbs sampler in expert systems. *Artificial Intelligence*, 56:115–130, 1992.

H. Yu, J. Huang, and J. Dauwels. Latent tree ensemble of pairwise copulas for spatial extremes analysis. In *Proceedings of the 2016 IEEE International Symposium on Information Theory*, pages 1730–1734, 2016.

T. Özcan, E. Benli, F. Özer, E. Y. Demir, Y. Kaya, and A. Ayyildiz. The association between symptoms of sexual dysfunction and age at onset in Parkinson's disease. *Clinical Autonomic Research*, 26(3):205–209, 2016.

C. Zhang, H. Fu, Q. Hu, X. Cao, Y. Xie, D. Tao, and D. Xu. Generalized latent multi-view subspace clustering. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 42 (1):86–99, 2018.

N. L. Zhang. Hierarchical latent class models for cluster analysis. *Journal of Machine Learning Research*, 5:697–723, 2004.

N. L. Zhang and T. Kočka. Efficient learning of hierarchical latent class models. In *Proceedings of the 16th IEEE International Conference on Tools with Artificial Intelligence*, pages 585–593, 2004.

N. L. Zhang and D. Poole. A simple approach to Bayesian network computations. In *Proceedings of the 10th Canadian Conference on Artificial Intelligence*, pages 171–178, 1994.

N. L. Zhang and L. K. Poon. Latent tree analysis. In *Proceedings of the 31st AAAI Conference on Artificial Intelligence*, pages 4891–4897, 2017.

N. L. Zhang, T. D. Nielsen, and F. V. Jensen. Latent variable discovery in classification models. *Artificial Intelligence in Medicine*, 30:283–299, 2004.

N. L. Zhang, Y. Wang, and T. Chen. Discovery of latent structures: Experience with the CoIL challenge 2000 data set. *Journal of Systems Science and Complexity*, 21:172–183, 2008a.

N. L. Zhang, S. Yuan, T. Chen, and Y. Wang. Latent tree models and diagnosis in traditional Chinese medicine. *Artificial Intelligence in Medicine*, 42:229–245, 2008b.

B. Zhao, J. T. Kwok, and C. Zhang. Multiple kernel clustering. In *Proceedings of the 2009 SIAM International Conference on Data Mining*, pages 638–649, 2009.

P. Zwiernik. *Handbook of Graphical Models*, chapter 11: Latent tree models, pages 267–290. CRC Press, 2018.