Learning Bayesian Network Structure from Highly Missing Data without Imputation

Melanie Piot[1], **Frederic Bertrand**[1], **Myriam Maumy**[1]
[1]University of Technology of Troyes
firstname.lastname@utt.fr

Abstract

In some application scopes, the data produced involve a significant proportion of missing data. Depending on the type of missing data, they can be ignored or imputed. However, ignoring missing data may lead to insufficient volume of available data to ensure good modeling, and imputation is not always desired. This work presents a Bayesian Network structure learning algorithm that can handle highly missing categorical data without performing any data imputation before or during learning. The learning process is based on a set of local bootstrap learnings performed on complete sub-datasets which are then aggregated and locally optimized. This learning method presents competitive results compared to other structure learning algorithms, whatever the type of missing data.

1 Introduction

Bayesian Networks (BN) (Pearl 1988) are models that describe dependencies relationships between variables (features) in a dataset using a *directed acyclic graph* (DAG) and *conditional probability distributions* (CPD). They are able to consider prior knowledge and to deal with uncertainty in the modeling, which makes them very powerful tools. Because of their graphical component, making them more intuitive to non-Machine Learning users, they are used in many multidisciplinary application fields, such as health (McLachlan et al. 2020) or social sciences (Smith et al. 2006).

In these scopes, it is not uncommon for data sets produced to have a higher proportion of missing data than those produced by industry (involving systematic and automatic data collection). Reasons for missing data can be manifold, such as a failure of the data entry software, a non-legitimacy of a question in specific situations, a data entry error, a participant dropping out of a study, or items being censored.
When the missing data are considered, they can be classified into three categories:

- Missing Completely At Random (MCAR) data: when the missingness is not related to the data itself but due to randomness. For example, a failure of the data entry software.
- Missing At Random (MAR) data: when the missingness can be explained (and thus deduced) by the observed data. For example, men not answering the question "Are you pregnant?" in a survey (assuming that the gender is known).
- Missing Not At Random (MNAR) data: when the missingness has unknown causes or can be explained by unobserved data. For example, when the lack of response to a question on alcohol consumption depends on the participant's alcohol consumption, which they are not ready to disclose.

The MCAR data can be ignored, as long as removing the samples from the dataset is equivalent to random sampling and does not introduce any bias in the final data set. However, in some cases the proportion of missing data is such that removing the examples results in a decrease in the volume of data that is too large to be still modeled.
When data are MAR, the preferred approach is imputation (single or multiple (Azur et al. 2011)) since ignoring samples with missing data would bias the final data set. Nevertheless as MAR data are hard to differentiate from MNAR data in most of the case, data imputation may lead to the introduction of biases, which is why some expert can be against this type of approach (Jakobsen et al. 2017).
If data are MNAR, complete case analysis and imputation may both lead to bias introduction in the final dataset.

The aim of this work is to provide an algorithm that can learn BN structure when missing data are too frequent to perform complete case analysis or when imputation is not wanted. It is based on the combination of data clustering, multiple local learning and graph fusion in order to exploit all the variables available in a dataset.

2 Bayesian Networks

Definition

For the rest of this paper, all variables will be assumed to be categorical.
A BN is the combination of a DAG, $\mathcal{G}$, and a set of CPDs, $\mathcal{P}$, that encodes the joint probability distribution of a set of random variables $\mathcal{X} = \{X_1, ..., X_M\}$ so that:

$$P(\mathcal{X}) = P(X_1, ..., X_M) = \prod_{i=1}^{M} P(X_i \mid Pa_{X_i}^{\mathcal{G}}) \qquad (1)$$

where $Pa^{\mathcal{G}}_{X_i}$ is the *parents* of the variable X_i in the DAG $\mathcal{G}$, i.e. the variables that point to X_i in the DAG. It is then clear that the fitness of the joint probability distribution to the data depends strongly on the fitness of the DAG.

The DAG $\mathcal{G}$ is composed of nodes, $\mathcal{V}$, representing the variables in $\mathcal{X}$ and arcs, $\mathcal{E}$, representing direct dependencies relations between them. Typically, if the relationship $X_i \rightarrow X_j$ exists, then X_i and X_j are dependent. X_i is called a *parent* of X_j, and X_j is called a *child* of X_i.
From this, it is possible to define the *Markov Blanket (MB)* of a node $v \in \mathcal{V}$, denoted $MB_{\mathcal{G}}(v)$, as the smallest subset of nodes, $S \subset \mathcal{V}$, such that *v* is conditionally independent of all other nodes given *S*. The subset *S* always contains the parents of *v*, its children, and the parents of its children.
There are three different ways to represent and interpret relationships between variables. Let X_i, X_j, X_k be three variables of $\mathcal{X}$ not independent from each other. They can be presented as follows :

$$\begin{cases} X_i \rightarrow X_j \rightarrow X_k \implies X_i \perp\!\!\!\perp X_k \mid X_j \\ X_i \leftarrow X_j \rightarrow X_k \implies X_i \perp\!\!\!\perp X_k \mid X_j \\ X_i \rightarrow X_j \leftarrow X_k \implies X_i \not\perp\!\!\!\perp X_k \mid X_j \end{cases} \tag{2}$$

Where $X_i \perp\!\!\!\perp X_k \mid X_j$ means that X_i is independent of X_k given X_j. This same reasoning can be applied to distinct subsets of variables.

Bayesian Networks Structure Learning

Structure learning is an NP-Hard problem (Chickering 1996), meaning that the space of possible DAGs increases super-exponentially with the number of variables. Structure learning algorithms are most often divided into three categories: i) constraint-based methods, ii) score-based methods and iii) hybrid methods, which consist of a sequential use of methods i and ii.

On the one hand, constraint-based algorithms (Margaritis 2003; Yaramakala and Margaritis 2005; Colombo and Maathuis 2014) use statistical tests to link together nodes that are not independent. These algorithms are always organized in at least two phases: i) neighborhood learning and ii) arc orientation. In phase i, the parents and children of each variable X_i (i.e., direct dependencies) are learned using a test of independence. For categorical variables, this test is most often the G^2 test (an asymptotic χ^2 mutual information test). Equivalently, for each pair X_i,X_j, $i \neq j$, a set $S_{X_iX_j} \subset \mathcal{V}$ is searched such that $X_i \perp\!\!\!\perp X_j \mid S_{X_iX_j}$. This phase can be simplified if the Markov Blanket of each variable is passed to the algorithm. Phase ii then orients the arcs found by first identifying the v-structures by analyzing pairs of non-adjacent variables, then orients the rest of the graph using graph engineering, as described in (Scutari 2015). Although very efficient on datasets with a large number of variables, they remain dependent on the choice of the significance threshold of the statistical test (specific to the application scope). Moreover, each test produces type-I and type-II error rates. Finally, they require the existence of an acyclic directed graph that satisfies the learned constraints.

On the other hand, score-based algorithms address the structure learning problem as a heuristic search. The best-known and most widely used algorithm is the greedy algorithm Hill-Climbing (HC). It usually starts from an empty structure and explores the space of possible DAGs using three operations: adding, deleting or reversing an arc. At each step, the fitness of the generated DAG to the data is evaluated using a fitness score (BIC (Schwarz 1978), AIC (Akaike 1974), BDe(u) (Heckerman, Geiger, and Chickering 1995), or others (Scutari 2016; Silander et al. 2018)). In the end, the DAG that is selected is the one that maximizes the fitness score to the data $\mathcal{D}$:

$$\underset{\mathcal{G} \in \boldsymbol{\mathcal{G}}}{\operatorname{argmax}}\, score(\mathcal{G}, \mathcal{D}) \tag{3}$$

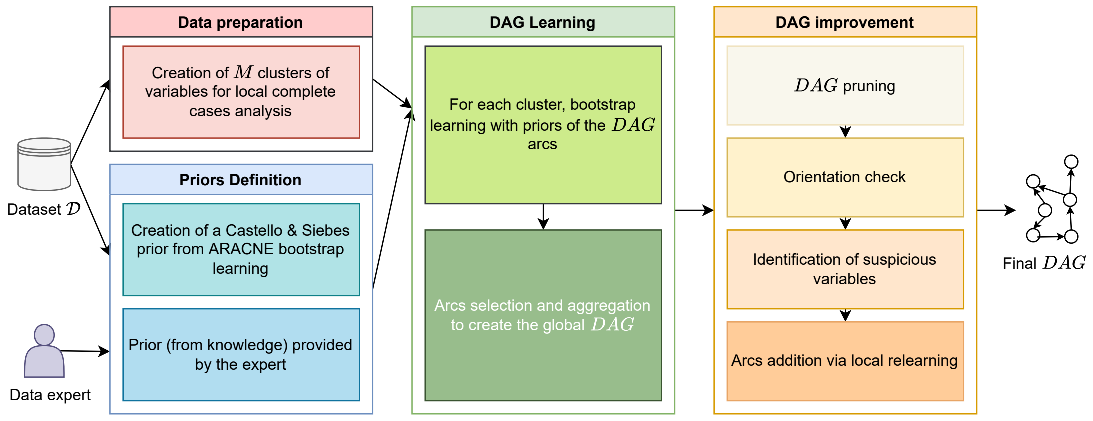
where $\boldsymbol{\mathcal{G}}$ is the set of generated DAGs.
Score-based methods are not exact and very often return the DAG resulting from a local maximum of the score function. To avoid getting stuck in the same local maximum during the learning process, different parameters are possible to add such as setting up a tabu list, performing random restarts, or disturbing the data in order to improve the generalization and to get closer to the global maximum. Moreover, parameterization (*imaginary sample size* (*iss*) value, prior distribution,...) of score-based algorithms can be tricky to set as they strongly depend on the data and the constraints of the application scope.

Bayesian Networks Structure Learning from Missing Data

Bayesian scores theoretically make it possible for score-based algorithms to take missing data into account. However, this implies a redefinition of the score as a function of both the observed and missing data (with all possible configurations of missing data). In case of high proportions of missing data this task becomes computationally infeasible (Bodewes and Scutari 2021).

Constraint-based algorithms can handle missing data as they perform conditional independence tests on a small subset of variables at a time to handle locally complete observations (i.e. local complete cases analysis). Nevertheless, when the proportion of missing data is too large, the number of complete observations becomes too small and both the type-I and type-II error rates of the conditional independence test will increase. As a result, the graph can be only partially oriented or inaccurate.
Another possibility is the "pairwise approach" that evaluates the mutual information between the variables two by two on a subset of locally complete observations to generate an undirected graph. This is the principle behind algorithms like ARACNE (Margolin et al. 2006). However, because the mutual information is symmetric, the graph cannot be oriented.
It is important to note that the use of these two approaches on missing MAR or MNAR data may lead to the introduction of biases in the resulting DAG.

Most algorithms that handle structural learning from (MAR) missing data involve an imputation phase before or



Data preparation
Creation of M clusters of variables for local complete cases analysis
Priors Definition
Creation of a Castello & Siebes prior from ARACNE bootstrap learning
Prior (from knowledge) provided by the expert
Dataset $\mathcal{D}$
Data expert
DAG Learning
For each cluster, bootstrap learning with priors of the DAG arcs
Arcs selection and aggregation to create the global DAG
DAG improvement
DAG pruning
Orientation check
Identification of suspicious variables
Arcs addition via local relearning
Final DAG

Figure 1: The CBSL algorithm and its steps

during learning (Friedman 1998; Fernández, Nielsen, and Salmerón 2010; Adel and de Campos 2017; Scanagatta et al. 2018). The most famous and used one (still today) is the Structural EM (Friedman 1998) algorithm (SEM). It repeats until convergence an E-step ,that completes the data from a BN, and an M-step, that learns the BN induced by the newly completed data. However, this algorithm is mainly intended for structure learning in case of latent (unobserved) variables and its performance decreases drastically beyond 20% missing data (Friedman 1997).

3 CBSL Algorithm

The algorithm described in this section is designed to be applied on datasets from various areas such as healthcare, which present missing data in unequal but potentially high proportions (up to 99% for some variables). It presents an interesting alternative to handle such datasets for DAG learning when analysis by global complete cases leads to an important reduction of the volume of available data and when the imputation of the data is not desired.

The Clustered Bootstrap Structural Learning algorithm is an algorithm for BN structural learning. It has been entirely realized in R (R Core Team 2021) using the bnlearn package (Scutari 2010). It allows local complete cases learning with any type of algorithm implemented in bnlearn (score-based, constraint-based or hybrid) and takes into account the inclusion of expert priors if available. It is structured in one step of data preparation, on step of structural priors definition and two learning phases. They are described below and showed in Figure 1.

Data preparation

In order to proceed to local complete cases learning, it is necessary to cluster the dataset to create completely observed sub-datasets. To ensure that each variable will be sampled at least once, as many sub-datasets are created as there are variables (M) in the initial dataset, with in each of them the presence of a variable that is forced. These sub-datasets, denoted clusters $\mathcal{C}_m, m \in \{1, ..., M\}$, are the subset of observed variables in cluster m. The creation of the set of variables $\mathcal{X}_{\mathcal{C}_m}$ of the cluster $\mathcal{C}_m$ was done by iteratively adding variables with the strongest Mutual Information with X_m as long as the number of complete samples of the sub-dataset $\mathcal{D}_{\mathcal{C}_m}$ is larger than n_{min} (by default set to 100). Thus, the more complete a sample is, the more clusters it will be present in. Similarly, the more often a variable is observed, the more likely it is to be present in many clusters.

Priors Definition

Structural priors definition is a key step that will counteract the introduction of biases during learning as much as possible. The more informative the expert prior, the better the final DAG. By default, the CBSL algorithm considers that no expert prior is available except a partial ordering of the variables. By default, this partial order separates the roots variables, the leaves variables and the intermediates variables. This partial order can also refer, for example, to a chronology of variables (the variables relating to a treatment will be known before those relating to its side effects). It is provided by a tier list (converted as blacklist) which, once provided to the algorithm during training, will prevent certain directions of arcs and reduce the search space of possible DAGs. This partial order needs to be provided by the user.
This is why a second prior is artificially defined from the data. The Castello & Siebes' prior (CS prior) (Castelo and Siebes 2000) provides the learning algorithm with a probability for the presence or absence of an arc between two variables, such that $\overrightarrow{p_{ij}} + \overleftarrow{p_{ij}} + \widehat{p_{ij}} = 1$, where $\overrightarrow{p_{ij}}$ is the probability of an arc $X_i \to X_j$, $\overleftarrow{p_{ij}}$ is the probability of an arc $X_j \to X_i$ and $\widehat{p_{ij}}$ is the probability of no arc between X_i and X_j. The stated probabilities are estimated from performing non-parametric bootstrap learning (100 replicates) of an undirected graph using the ARACNE algorithm. If the resulting probabilities are 0 or 1, they had an α argument added (respectively subtracted) in order to preserve some uncertainty on the arcs[1]. Thus $\overrightarrow{p_{ij}} = \overleftarrow{p_{ij}} = \overleftrightarrow{p_{ij}}/2$. It is important to note that this prior is only compatible with score-based algorithms using a Bayesian score (BDe, BDs (Scutari

[1]The value of this parameter is to be chosen by the user. Its default value is 0.1.

2016)[2], ...) as it replaces the conditional probability distribution of the score.
The CS prior probabilities can be replaced by expert knowledge if necessary.

Phase 1: DAG Learning

There are two steps in the DAG learning phase: local learning of the arcs then selection and aggregation of the arcs.
The first step consists in a non-parametric bootstrap learning (100 replicates) of a local DAG for each of the *M* clusters. By default, this learning is performed with the HC algorithm implemented in bnlearn, parameterized with a BDs score (with imaginary sample size (*iss*) set to 1), the partial order (passed as blacklist), and the CS prior. The result provided is, for each $\mathcal{C}_m$, the *strength*, $P(\overrightarrow{p_{ij}} + \overleftarrow{p_{ij}})$, and the *direction*, $P(\overrightarrow{p_{ij}} \mid \overrightarrow{p_{ij}} + \overleftarrow{p_{ij}})$, for each arc. A first selection is made so that only arcs with $direction > 0.5$ are kept.

The second step is to identify the relevant arcs that will compose the global DAG. The achievement of this task was based on the assertion made by Friedman in (Friedman, Goldszmidt, and Wyner 1999), stating that when using bootstrap learning, the prediction of a (Markov) neighborhood of two variables was more robust than the learning of the edges of a DAG. Thus, this step aim to recover the possible neighborhood of each variable. For each variable, the arcs related to the variable of interest are selected. A confidence threshold calculated from the L1 norm (Scutari and Nagarajan 2013) is then applied on their probabilities. Arcs whose probability is greater than this confidence threshold are kept to form the DAG. The acyclicity is guaranteed thanks to a feature of bnlearn, which, in case of cycle, will remove the arc with the smallest *strength*.

Phase 2: DAG improvement

As all variables are not simultaneously observed and sample size of each sub-dataset is limited due to clustering, part of the arcs found during local bootstrap learning might come from the noise induced in the data. The objective of this part is to perform different local operations on the DAG resulting from the previous phase to remove noisy arcs and add new ones when necessary. Due to the presence of missing data in the dataset, the DAG can only be analyzed locally. Therefore, all operations detailed in the rest of this section will be performed on a subgraph (and a sub-dataset) composed of the variables at the endpoints of the arc and their MB. The four improvement steps in Figure 1 will occur iteratively.

The first step compares, for each arc of the DAG, the scores of a subgraph on a sub-dataset with and without the arc and keeps the arc only if its presence improves the score. The same reasoning is valid for the second step, which evaluates the orientation of the DAG arcs.

In order to know if arcs are missing and to identify potential concerned variables, the third step will compare the status of the DAG variables with their theoretical status induced by the tier list (see Section 3) to identify "suspicious" variables. As a reminder, by default each variable can be a

[2]Score used by default in Phase 1 of learning

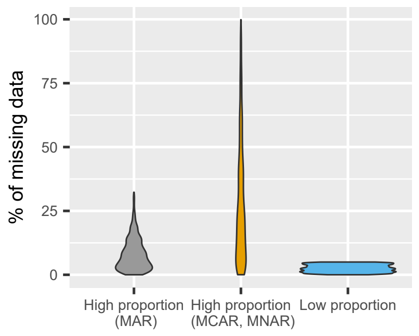


Figure 2: Distribution of the proportions of missing data generated for each variable in each replicate.

leaf (without children), a root (without parents) or intermediate (with parents and children). A variable is considered suspicious if its status in the DAG does not correspond to its theoretical status.
Once the suspicious variables are selected, possible arcs to add are identified among those found during local learning but left aside because they were not related to the variable of interest and among the undirected arcs learned by ARACNE during the prior definition phase. For each of these possible arcs, a local learning[3] is performed in order to prevent cycles from appearing in the graph and to minimize orientation errors. The arcs of these new graphs are stored if the subgraph maximizes the score on the sub-dataset. The inclusion of each stored arc is then evaluated, in the same way as for steps 1 and 2, in order to guarantee their correct integration in the final DAG. When adding the new arc to the final DAG, again, acyclicity is guaranteed thanks to a bnlearn feature, which, in case of cycling, will remove the arc with the smallest force.

4 Experiments & Results

Prior Creation

Two different types of prior knowledge have been generated and provided to the learning algorithms: a whitelist and a CS' prior. The first allows the integration of ground truth knowledge of the domain (supported by the literature), the second allows the integration of expert knowledge while preserving a reasonable uncertainty level (α). For the CS' prior, in addition to the experts of the application scope, the information gathered by the data scientists in charge of the data can also be useful (identification of the mechanism of missing data, categorization of a variable, correlations, etc.). These knowledge will be used to guide the learning to overcome missing data. The different priors and their use are summarized in the table 1.

Whitelist: The whitelist forces some arcs to be present in the graph. It allows to restrict the search space of the DAG to converge more quickly and efficiently to the optimal DAG.

[3]By default the HC algorithm is used, parameterized with a qNML score (Silander et al. 2018) and partial order passed as blacklist

	Generation of priors						Use of priors			
Mechanism	MCAR		MAR		MNAR		Algorithms			
Missing data prop.	Low	High	Low	High	Low	High	Complete	Inter IAMB	SEM	CBSL
Whitelist	No	Yes	No	Yes	No	Yes	Yes			
Expert prior	Yes						Yes	No	Yes	Yes
Data scientist prior	No	No	Yes	Yes	Yes	Yes	Yes	No	Yes	Yes

Table 1: Generation and use of priors conditionally to missing data mechanism and learning algorithm

It is important to include only arcs for which domain experts are absolutely sure, to limit the risk of including false positives and distorting DAG search.
This type of prior was introduced only for highly missing variables (≥50% for the MCAR and MNAR mechanisms and ≥25% for the MAR mechanism) since such a high proportion of missing data would have necessarily led the data scientist to turn to domain experts to understand the causes of such deficiencies. For each of these highly missing variables, one incoming or outgoing arc was drawn at random and whitelisted.
The resulting whitelist is then provided all the learning algorithms.

CS' prior: Castello & Siebes' prior (CS prior) consist of a list of probabilities that each arc exists or not. These probabilities are used as prior for the Bayesian score when learning the structure. CS' priors from domain experts and from data scientists were generated and combined. The resulting probabilities are then provided to the SEM and CBSL learning algorithms (as inter IAMB is not compatible with this type of prior, only the whitelist is provided). For CBSL algorithm, the generated probabilities replace those computed by the bootstrap learning with ARACNE.

Expert prior: The expert prior consists of the experts' intuition about the existence or not of certain arcs. It can be any kind of expertise for which uncertainty remains or which is not reported in the literature. This prior was simulated by randomly drawing 10% of the arcs of the real graph and assigning them the following probabilities:

$$\begin{cases} \overrightarrow{p_{ij}} = \frac{1}{2}(1-\alpha) \\ \overleftarrow{p_{ij}} = \frac{1}{2}(1-\alpha) \\ \widehat{p_{ij}} = \alpha \end{cases} \tag{4}$$

where α represents the uncertainty (by default, alpha is 0.1 for all simulations). If some of the drawn arcs are already in the whitelist, they are kept in the CS' prior with α and the wrong orientation set to 0.

Data scientist prior: The purpose of this second type of prior is to represent any additional knowledge that data scientists could acquire from the analysis of the data before their modeling (covariances, missingness mechanism, etc.). It was simulated by trying to recover the missing data mechanism and considering for each variable a possible arc ($\alpha = 0.3$) with the most observed covariate when the variable is missing. Since the MCAR mechanism can be identified by a test (Little 1988), this prior was generated only for the MAR and MNAR mechanisms. Since it is impossible to distinguish MAR from MNAR, the default assumption is that the mechanism is MAR. Therefore, this prior may introduce many false positives, especially for MNAR data.

Experiments

Experiments have been performed to compare the performance of the CBSL algorithm to algorithms well known and widely used, performing local learning (constraint-based algorithm Inter-IAMB (Yaramakala and Margaritis 2005) as it is known to avoid false positives in the Markov blanket detection phase (Scutari 2010)), or imputation (SEM). They are both implemented in *bnlearn* R package. The performances of the three algorithms were compared to those of a structure learning on complete data (without missing data).
Five toy datasets were chosen for the experiments. : ASIA (8 nodes, 8 arcs)(Lauritzen and Spiegelhalter 1988), CHILD (20 nodes, 25 arcs)(Spiegelhalter et al. 1993), ALARM (37 nodes, 46 arcs)(Beinlich et al. 1989), INSURANCE (27 nodes, 52 arcs)(Binder et al. 1997) and HAILFINDER (56 nodes, 66 arcs)(Abramson et al. 1996). For each dataset, 20 replicates of 3000 samples were randomly drawn. On each replicates, high and low proportion of MCAR, MAR and MNAR missing data (Fig. 2) were generated. Each of these six combinations was learned with and without additional prior, generated as described in Section 4. Only the results from learnings on high proportions of missing data will be discussed in the results, as learnings on low proportions leads to similar conclusions.
The blacklist encoding the partial order of the variables was systematically provided to all algorithms for DAG learning. The algorithm for complete learning (Complete Learning) was parameterized with the HC algorithm and BDs score ($iss = 1$). The SEM algorithm was parameterized with the HC algorithm, BDs score ($iss = 1$), and maximum likelihood estimator for the maximisation step and with the *'bayes-lw'* method (averaging likelihood weighting simulations performed using all the available nodes as evidence) for the expectation step. The inter IAMB algorithm was parameterized with $\alpha = 0.05$ for statistical tests. Finally, the CBSL algorithm was parameterized with default parameters described in Section 3.

Results

Figures 3, 4 and 5 show the results of learnings with and without prior knowledge on the different types of missing data in high proportions. True Positive (TP) arcs are arcs in the learned DAG that are in the true DAG, False Positive

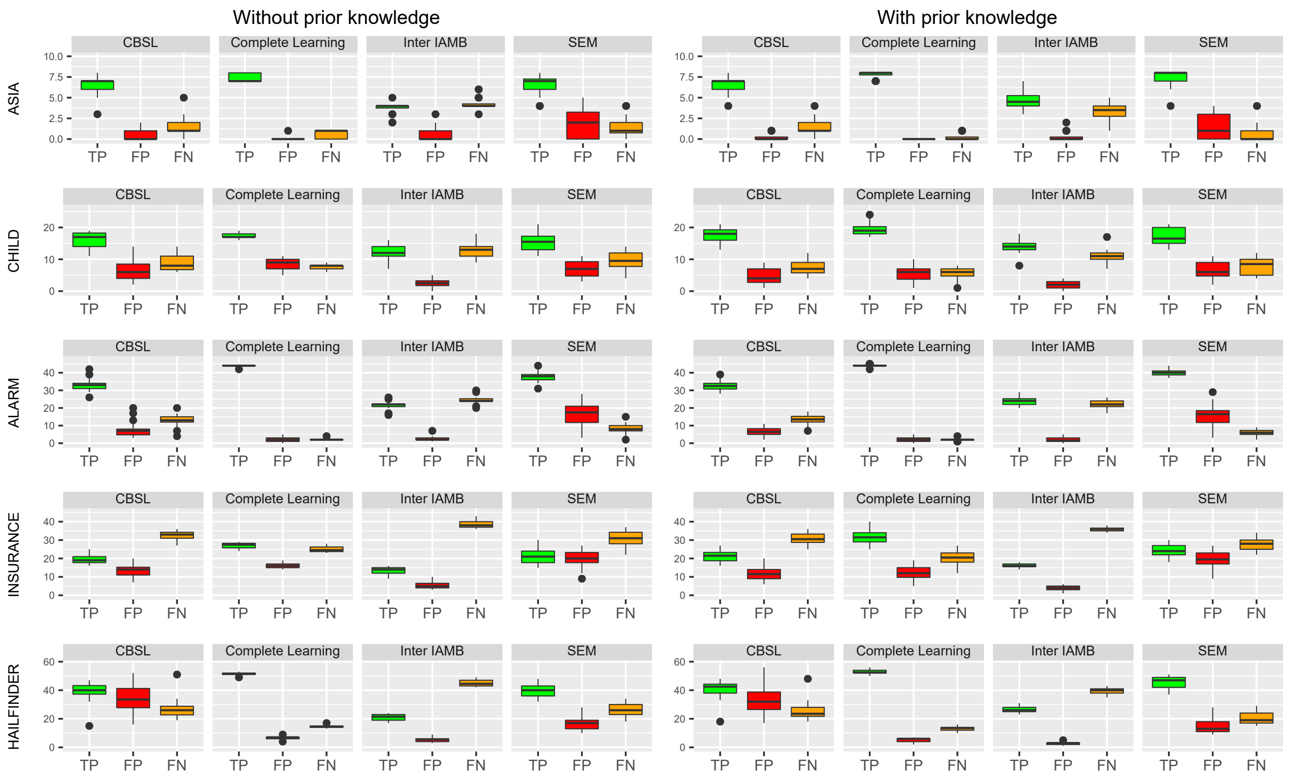


Figure 3: Results of learnings with and without prior knowledge on high proportion of MCAR data

(FP) arcs are arcs in the learned DAG that shouldn't have been include according to the true DAG (including arcs with wrong orientation) and False Negative (FN) arcs are arcs not present in the learned DAG that should have been included. The theoretical maximums of the values for each dataset are given below in the form (TP, FP). The FN are not given since they have the same maximum as the TP: ASIA (8,48), CHILD (25, 355), ALARM (46, 1286), INSURANCE (52, 650), HAINFINDER (66, 3014). The log likelihood of the DAGs resulting from learning on each of the 20 replicates is also recovered, in order to evaluate the fitness to the data.

Whatever the mechanism and the proportion of missing data, Inter IAMB is nearly always the one with the most FNs the least FPs. The large number of FNs tends to show that the presence of missing data coupled with a limited initial volume (3k samples) may distort the results of the statistical tests performed that prevent the algorithm from correctly identifying the *MB* of each variable. Furthermore, it must be noticed that the algorithm failed to orient the arcs of the graph and therefore fails to produce a fully oriented graph for 10% of the learnings.

The SEM algorithm is the algorithm most affected (positively) by the addition of prior knowledge. Nevertheless, although this algorithm often seems to be the one with the most TP (median and 3rd quartile), this comes at the cost of many additional FPs, regardless of the mechanism of missing data. This is a direct consequence of imputation, which introduces noise in the data, especially when the a priori knowledge contains FPs (as explained in Section 4). Here again, the algorithm sometimes failed to produce a DAG due to an inability to generate weights needed for data imputation (for 1% of the learnings).

Unlike the two previous algorithms, CBSL allows a good compromise between TP and FP. It often presents less spread FPs and with a lower median than SEM without sacrificing too much TPs. Figure 6 The figure displays the average number of changes made by phase 2 of the algorithm. For example, when learning with a prior knowledge on MAR missing data for the HAILFINDER dataset, this phase added an average of 13 TPs to the DAG (respectively removed 13 FNs), without introducing any new FPs. It clearly shows the impact of phase 2 on the decrease of the number of FPs and FNs and on the increase of the number of TPs in most of the cases. It is thanks to this that the algorithm seems to be more robust to noise (introduced in the data by bootstrapping and inexact prior knowledge) than SEM. Figure 7 displays the average number of samples used for each of the two learning phases of the CBSL algorithm. It shows that the algorithm almost always uses all the samples in phase 2, explaining the performance gain offered by this phase. Figure 8 also shows that CBSL resulting DAGs has a better log likelihood than Inter IAMB and very close to SEM, without doing any imputation.

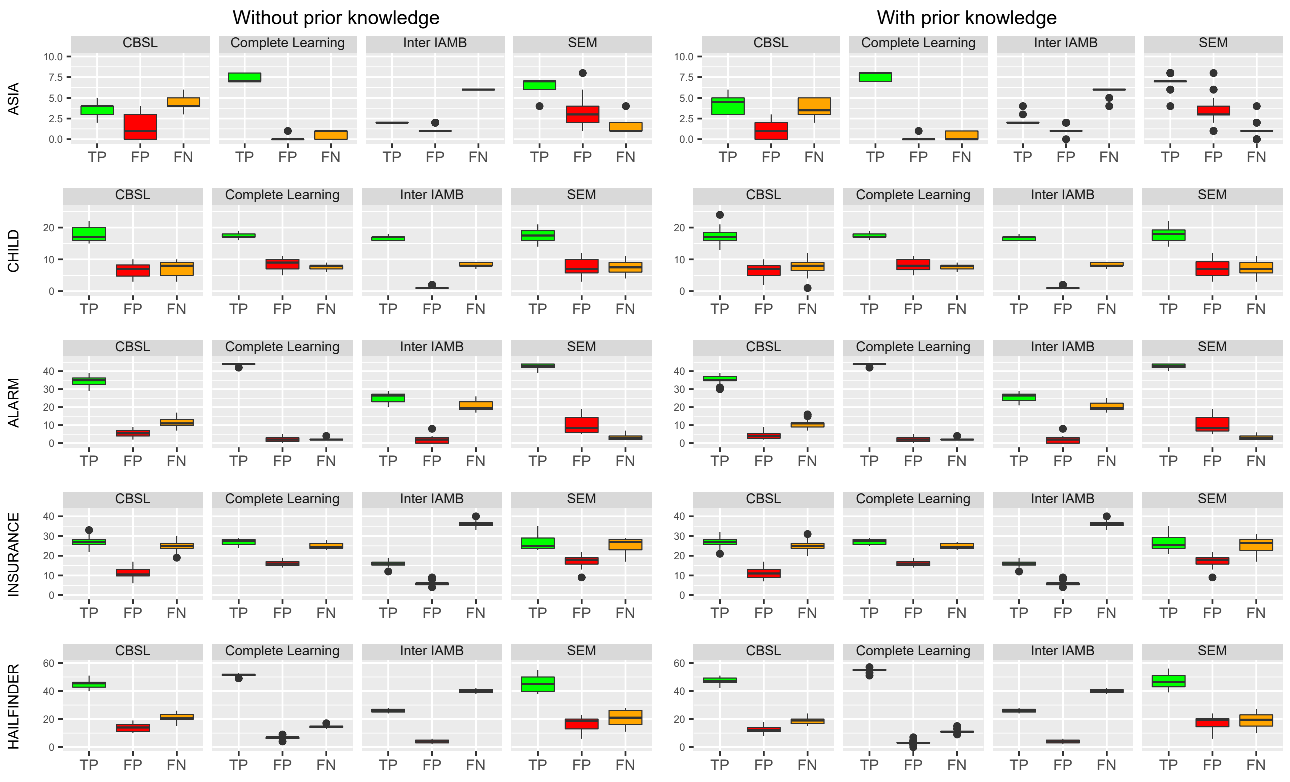


Figure 4: Results of learnings with and without prior knowledge on high proportion of MAR data

5 Discussion

The first phase of CBSL allows to limit the risks of FP linked to a sampling bias resulting from both the missing data and the clustering of the variables performed beforehand. This is done by means of two assets: bootstrap learning and arc selection.

The first one allows to artificially introduce diversity in the data in order to maximize the chances of finding TP arcs. Nevertheless, the efficiency of the bootstrap depends strongly on the representativity of the initial samples. In some cases (in particular for MAR data) this set of samples may be too small and unrepresentative. In these cases, the sampling bias may be too strong to be compensated by the bootstrap. Conversely, note that a variable with few missing data will tend to be over-represented by the clusters (i.e. will be present in most clusters), thus increasing the risk of noisy arcs related to this variable. Thus, the way the sub-datasets are created has a big impact on the results of phase 1. Future investigations may implement other methods for creating sub-datasets and study their impact on learning.

The second one tries to identify the significant arcs resulting from the learning. Identifying significant arcs is a complex problem. The objective is to find a confidence threshold allowing to separate arcs coming from the noise contained in the data from those representing the *true* relations between the variables. This confidence threshold is an unknown function that depends on both the data and the structure learning algorithm. Nevertheless, the problem is even more complex here, as all the variables are never simultaneously observed, and their sample size is limited, which can increase the risk of false positive arcs. For example, the case where three variables X_i, X_j, X_k are connected as $X_i \to X_j \to X_k$ could appear as $X_i \to X_k$ with high probability if X_j does not belong to the same cluster; since $X_i \perp\!\!\!\perp X_k \mid X_j$, they become dependent when X_j is not observed. As a result, it becomes difficult to distinguish good from bad arcs based on their *strength*, since each cluster does not encode the same set of dependencies between variables. The selection of a confidence neighborhood, as performed by CBSL, instead of the set of significant arcs allows to limit the introduction of FP in this kind of situations.

One way to improve the results of this first phase of the algorithm would be to improve the probabilities of appearance of each arc so that they are easier to select later. In other words, the DAGs learned by each cluster should be made closer to the true DAG. Opting for a hybrid algorithm, providing a non-empty starting structure to the algorithm (established from prior knowledge), or performing random restarts during learning to avoid getting stuck on a local maximum of the score function, are all possible solutions, to be preferred according to the constraints of the scope.

The second phase of CBSL depends very strongly on the chosen score. Although only one score has been chosen here for all the steps of the phase, it is possible to specify a dif-

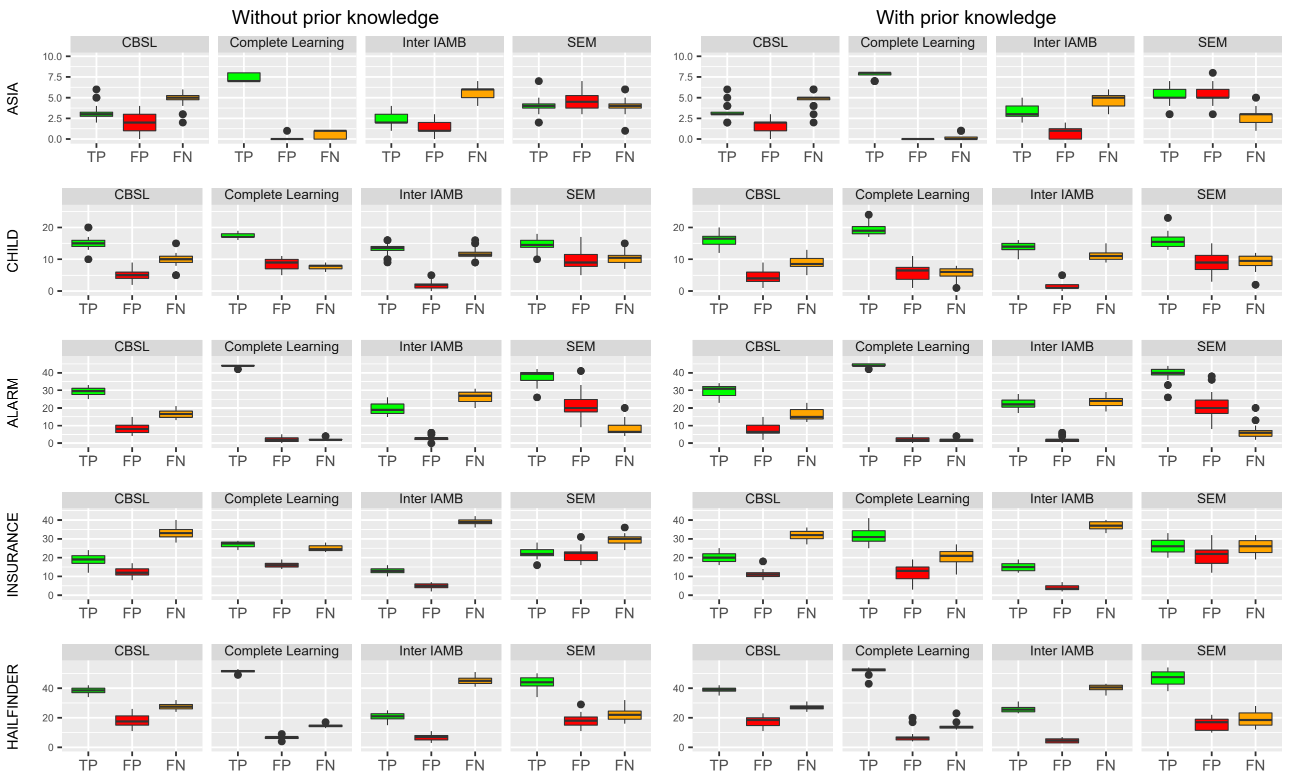


Figure 5: Results of learnings with and without prior knowledge on high proportion of MNAR data

ferent score each time, depending on the constraints to be applied on the DAG. For example, using the BIC score for the last step will further penalize new arcs to be added to the DAG, thus decreasing the risk of FP. This example could be indicated for medical domains.
Also, the last step (adding arcs) relies only on the previous step of identifying suspicious variables. The latter has been designed to work in the case where no prior knowledge would be available for the studied data. Nevertheless, when expertise is available, it would be good to be able to take it into account during this step. It could be possible, for example, to ask the user to identify suspicious variables based on the DAG resulting from the orientation checking step.

Also, since CBSL is based on a set of local learning, the learning of the DAG is not unified. As a consequence, it is possible that a succession of local improvements do not benefit to a global improvement of the DAG, as it seems to be shown by the learning on the HAILFINDER dataset with MCAR data (figures 3 and 6). This could be partly explained by figure 7. Indeed, the efficiency of phase 2 seems to depend on its ratio in number of samples to phase 1. The higher the ratio, the more the phase 2 enriches the DAG. Here, the ratio was not high enough for phase 2 to properly enrich the DAG.

Moreover, the analysis of data missingness mechanisms before learning can act on the data clustering method in addition to provide as prior knowledge, in order to limit the sampling bias. This hypothesis will be the subject of a future work.

6 Conclusion

It can be deduced from Figures 3 to 5 that Inter IAMB often has the best Positive Predictive Value (PPV). In situations where the quality of the arcs discovered is preferred to the quantity, it is a good choice as long as that the volume of available data is sufficient to not distort the tests.

In the same way, SEM has often a lower PPV because the imputation induces many FPs. In situations where the quantity of arcs discovered is more important than the quality, it is a legitimate choice as long as there are enough knowledge priors to guide imputation process.

CBSL offers an interesting trade-off since it learns DAGs with a log likelihood better than Inter IAMB and very close to SEM, without performing any imputation. It allows to perform structure learning even when the prior knowledge is limited. The clustered bootstrap learning phase performed by CBSL allows to limit the effect of the sampling bias, and its local improvement phase allows to reduce efficiently the FPs. These properties make it more robust to noisy (inaccurate) prior knowledge and its results are less dependent on the mechanism of missing data. Therefore CBSL is a relevant addition to the existing Bayesian Networks structure learning dataset with missing values.

References

Abramson, B.; Brown, J.; Edwards, W.; Murphy, A.; and Winkler, R. L. 1996. Hailfinder: A Bayesian System for Forecasting Severe Weather. *International Journal of Forecasting*, 12(1): 57–71.

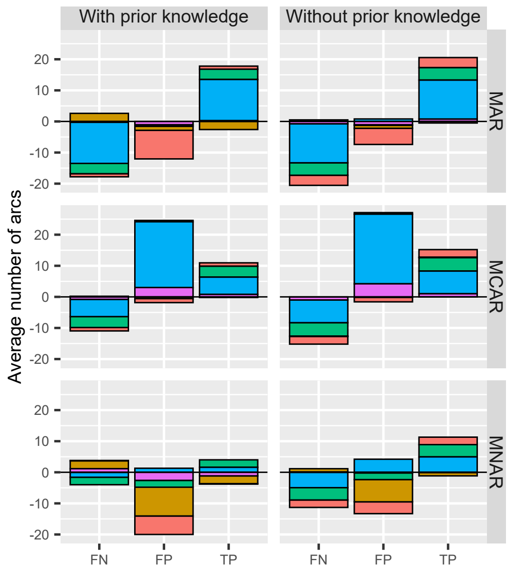


Figure 6: Average number of changes made by phase 2 of the CBSL algorithm relative to the number of arcs in phase 1 for each missing data mechanism and learning type. Each dataset is represented by a color (red: ALARM, yellow: ASIA, green: CHILD, blue: HAILFINDER, violet: INSURANCE).

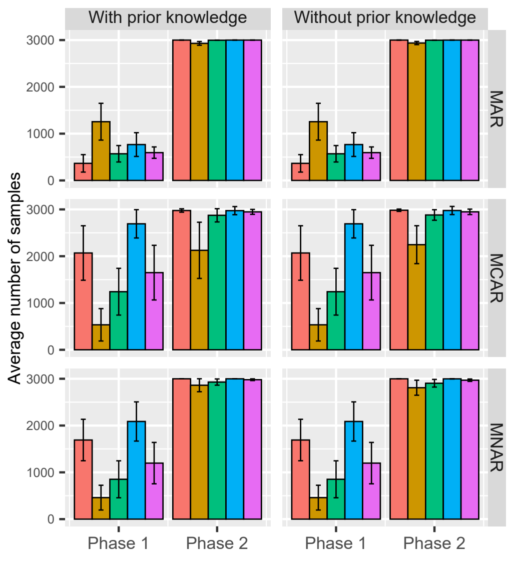


Figure 7: Average number of samples used for the two learning phases of CBSL algorithm for each missing data mechanism and learning type. Each dataset is represented by a color (red: ALARM, yellow: ASIA, green: CHILD, blue: HAILFINDER, violet: INSURANCE).

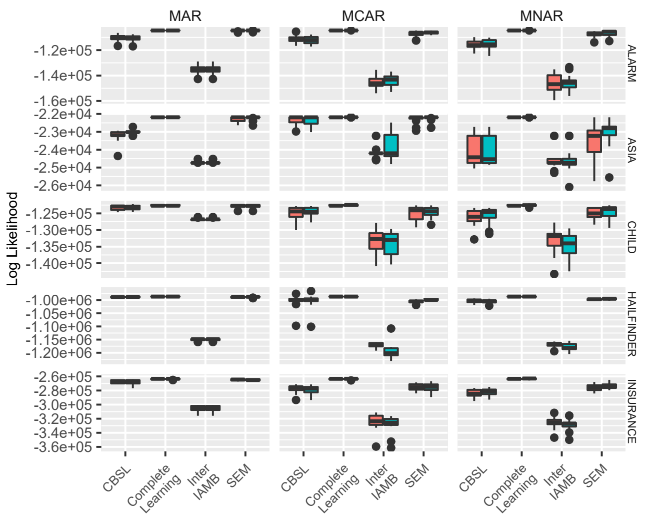


Figure 8: Log Likelihood of the DAGs for each algorithm (red: without prior knowledge, blue: with prior knowledge)

Adel, T.; and de Campos, C. 2017. Learning Bayesian Networks with Incomplete Data by Augmentation. *Proceedings of the AAAI Conference on Artificial Intelligence*, 31(1).

Akaike, H. 1974. A New Look at the Statistical Model Identification. *IEEE Transactions on Automatic Control*, 19(6): 716–723.

Azur, M. J.; Stuart, E. A.; Frangakis, C.; and Leaf, P. J. 2011. Multiple Imputation by Chained Equations: What Is It and How Does It Work? *International Journal of Methods in Psychiatric Research*, 20(1): 40–49.

Beinlich, I. A.; Suermondt, H. J.; Chavez, R. M.; and Cooper, G. F. 1989. The ALARM Monitoring System: A Case Study with Two Probabilistic Inference Techniques for Belief Networks. In *Proceedings of the 2nd European Conference on Artificial Intelligence in Medicine*, 247–256. Springer-Verlag.

Binder, J.; Koller, D.; Russell, S.; and Kanazawa, K. 1997. Adaptive Probabilistic Networks with Hidden Variables. *Machine Learning*, 29(2): 213–244.

Bodewes, T.; and Scutari, M. 2021. Learning Bayesian Networks from Incomplete Data with the Node-Average Likelihood.

Castelo, R.; and Siebes, A. 2000. Priors on Network Structures. Biasing the Search for Bayesian Networks. *International Journal of Approximate Reasoning*, 24(1): 39–57.

Chickering, D. M. 1996. Learning Bayesian Networks Is NP-Complete. In Fisher, D.; and Lenz, H.-J., eds., *Learning from Data: Artificial Intelligence and Statistics V*, Lecture Notes in Statistics, 121–130. New York, NY: Springer. ISBN 978-1-4612-2404-4.

Colombo, D.; and Maathuis, M. H. 2014. Order-Independent Constraint-Based Causal Structure Learning. *Journal of Machine Learning Research*, 15(116): 3921–3962.

Fernández, A.; Nielsen, J. D.; and Salmerón, A. 2010. Learning Bayesian Networks for Regression from Incomplete Databases. *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems*, 18(01): 69–86.

Friedman, N. 1997. Learning Belief Networks in the Presence of Missing Values and Hidden Variables. In *Proceedings of the Fourteenth International Conference on Machine Learning*, ICML '97, 125–133. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc. ISBN 978-1-55860-486-5.

Friedman, N. 1998. The Bayesian Structural EM Algorithm. In *Proceedings of the Fourteenth Conference on Uncertainty in Artificial Intelligence*, UAI'98, 129–138. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc. ISBN 978-1-55860-555-8.

Friedman, N.; Goldszmidt, M.; and Wyner, A. 1999. Data Analysis with Bayesian Networks: A Bootstrap Approach. In *Proceedings of the Fifteenth Conference on Uncertainty in Artificial Intelligence*, UAI'99, 196–205. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc. ISBN 978-1-55860-614-2.

Heckerman, D.; Geiger, D.; and Chickering, D. M. 1995. Learning Bayesian Networks: The Combination of Knowledge and Statistical Data. *Machine Learning*, 20(3): 197–243.

Jakobsen, J. C.; Gluud, C.; Wetterslev, J.; and Winkel, P. 2017. When and How Should Multiple Imputation Be Used for Handling Missing Data in Randomised Clinical Trials – a Practical Guide with Flowcharts. *BMC Medical Research Methodology*, 17(1): 162.

Lauritzen, S. L.; and Spiegelhalter, D. J. 1988. Local Computation with Probabilities on Graphical Structures and their Application to Expert Systems (with discussion). *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 50(2): 157–224.

Little, R. J. A. 1988. A Test of Missing Completely at Random for Multivariate Data with Missing Values. *Journal of the American Statistical Association*, 83(404): 1198–1202.

Margaritis, D. 2003. *Learning Bayesian Network Model Structure from Data*. Ph.D. thesis, Carnegie-Mellon Univ Pittsburgh Pa School of Computer Science.

Margolin, A. A.; Nemenman, I.; Basso, K.; Wiggins, C.; Stolovitzky, G.; Favera, R. D.; and Califano, A. 2006. ARACNE: An Algorithm for the Reconstruction of Gene Regulatory Networks in a Mammalian Cellular Context. *BMC Bioinformatics*, 7(1): S7.

McLachlan, S.; Dube, K.; Hitman, G. A.; Fenton, N. E.; and Kyrimi, E. 2020. Bayesian Networks in Healthcare: Distribution by Medical Condition. *Artificial Intelligence in Medicine*, 107: 101912.

Pearl, J. 1988. *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference*. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc. ISBN 978-1-55860-479-7.

R Core Team. 2021. R: A Language and Environment for Statistical Computing. R Foundation for Statistical Computing.

Scanagatta, M.; Corani, G.; Zaffalon, M.; Yoo, J.; and Kang, U. 2018. Efficient Learning of Bounded-Treewidth Bayesian Networks from Complete and Incomplete Data Sets. *International Journal of Approximate Reasoning*, 95(C): 152–166.

Schwarz, G. 1978. Estimating the Dimension of a Model. *The Annals of Statistics*, 6(2): 461–464.

Scutari, M. 2010. Learning Bayesian Networks with the Bnlearn R Package. *Journal of Statistical Software*, 35(3): 1–22.

Scutari, M. 2015. Bayesian Network Constraint-Based Structure Learning Algorithms: Parallel and Optimised Implementations in the Bnlearn R Package. arXiv:1406.7648.

Scutari, M. 2016. An Empirical-Bayes Score for Discrete Bayesian Networks. In *Proceedings of the Eighth International Conference on Probabilistic Graphical Models*, 438–448. PMLR.

Scutari, M.; and Nagarajan, R. 2013. Identifying Significant Edges in Graphical Models of Molecular Networks. *Artificial Intelligence in Medicine*, 57(3): 207–217.

Silander, T.; Leppä-aho, J.; Jääsaari, E.; and Roos, T. 2018. Quotient Normalized Maximum Likelihood Criterion for Learning Bayesian Network Structures. In *Proceedings of the Twenty-First International Conference on Artificial Intelligence and Statistics*, 948–957. PMLR.

Smith, V. A.; Yu, J.; Smulders, T. V.; Hartemink, A. J.; and Jarvis, E. D. 2006. Computational Inference of Neural Information Flow Networks. *PLOS Computational Biology*, 2(11): e161.

Spiegelhalter, D. J.; Dawid, A. P.; Lauritzen, S. L.; and Cowell, R. G. 1993. Bayesian Analysis in Expert Systems. *Statistical Science*, 8(3): 219 – 247.

Yaramakala, S.; and Margaritis, D. 2005. Speculative Markov Blanket Discovery for Optimal Feature Selection. In *Fifth IEEE International Conference on Data Mining (ICDM'05)*, 4 pp.–.