A Bayesian Approach for Learning Bayesian Network Structures

Abstract

We introduce a Bayesian approach method based on the Gibbs sampler for learning the Bayesian Network structure. For this, the existence and the direction of the edges are specified by a set of parameters. We use the non-informative discrete uniform prior to these parameters. In the Gibbs sampling, we then sample from the full conditional distribution of these parameters, then a set of DAGs is obtained. For achieving a single graph that represents the best graph fitted on data, Monte Carlo Bayesian estimation of the probability of being the edge between nodes is calculated. The results on the benchmark Bayesian networks show that our method has higher accuracy in structure learning than the state-of-the-art algorithms.

Keywords: Bayesian Approach, Gibbs Sampler, Bayesian Network Structure, Monte Carlo Bayesian Estimation

1 INTRODUCTION

A probabilistic graphical model is a statistical model embodying a set of conditional independence relationships associated with a graph as an underlying structure such as Bayesian networks (BNs) [Pearl, 1988]. A BN is a probabilistic graphical model for representing knowledge about an uncertain domain where each node corresponds to a random variable, and each edge represents the conditional probability for the corresponding random variables [Heckerman, 1998]. In other words, a BN is defined by a tuple consisting of a Directed Acyclic Graph (DAG) and a set of parameters, representing the strength and the shape of the relationships between variables. We consider the learning of Bayesian networks structures from data, which is an NP-hard problem [Chickering et al., 2004]. Different approaches have been proposed to face this problem, and they can be classified into three categories: constraint-based, score-based, and hybrid methods. Constraint-based methods strategically test conditional independence relationships between pairs of variables, first determining the existence of edges before inferring orientations (Spirtes and Glymour [1991], Meek [1995]). In the score-based approach, heuristics are designed to optimize some scoring criterion that evaluates the goodness-of-fit of a proposed structure to the available data (Heckerman et al. [1995]). Finally, hybrid methods combine the two strategies, optimizing a score over a reduced space of structures restricted through a constraint-based approach ([Singh and Valtorta, 1993, Tsamardinos et al., 2006, Gasse et al., 2014]. An extensive review of existing algorithms and software tools is given in [Scutari et al., 2019, Scanagatta et al., 2019].

In a Bayesian framework, structure learning is based on the posterior distribution $P(B|\mathbf{D})$ where B is the BN structure and $\mathbf{D}$ is the data on p variables (Koller and Friedman [2009]). In this work, we consider a Bayesian approach to the discovery of BN structures [Friedman and Koller, 2003, Heckerman et al., 2006]. A Bayesian approach to structure learning quantifies the strength with which the data and any available prior knowledge jointly support each possible graph structure in the form of posterior probabilities. However, the number of possible DAGs grows super-exponentially with the number of variables being represented, making a full comparison of Bayesian posterior probabilities associated with alternative structures intractable. The Markov Chain Monte Carlo (MCMC) method as applied to graphical structures provides a numerical solution to this problem ([Su and Borsuk, 2016]). This approach is prevalent, and variations have been used by Madigan et al. [1995] and Giudici and Castelo [2003]. Madigan et al. [1995] proposed the original version of the MCMC in which each move in the Markov chain consists of basically a single edge change to the current graph (B). This algorithm is a classical Metropolis-Hastings sampler. The acceptance

probability is $r(B', B)$:

$$r(B', B) = \min\{1, \frac{\#\text{nbd}(B)P(B'|\mathbf{D})}{\#\text{nbd}(B')P(B|\mathbf{D})}\}, \qquad (1.1)$$

where B' is the proposal BN, $P(B|\mathbf{D})$ is the posterior probability of a graph given a database of cases $\mathbf{D}$, and $\#\text{nbd}(B)$ is the number of neighbors that consist of the current graph B and a set of graphs with either one edge more or one edge fewer than the current graph.

While the original version of MCMC generally performs well in little spaces with a few variables, it is relatively slow in convergence with a larger number of variables, and the chain is getting trapped in local high probability [Madigan et al., 1995]. For overcoming this problem, some methods have been introduced. For example, Friedman and Koller [2003] proposed a variety of the MCMC algorithm based on the node ordering. Niinimaki et al. [2012] proposed an algorithm based on the partial node ordering to make smoother sampling space. Su and Borsuk [2016] improved the structure of MCMC for BNs through the Markov blanket resampling and Goudie and Mukherjee [2016] used a specific Gibbs sampling based on the entire sets of parents for multiple nodes from the appropriate conditional distribution.

This paper focuses on the Bayesian approach, which takes prior knowledge and combines it with data to discover the BN structure. The Bayesian approach to learning BNs amounts to searching for network-structures with high relative posterior probabilities. In this way, we introduce a novel Gibbs sampler. For this, the existence and the direction of the edges in B are specified by a set of parameters. We also use the non-informative discrete uniform prior on these parameters. We sample from the full conditional distribution of these parameters, so a set of DAGs is obtained. We finally use the Monte Carlo Bayesian estimation of the probability of being the edge between nodes for achieving a best graph fitted on data. The proposed algorithm finds the most true and few additional and missing edges compared to the state-of-the-arts algorithms.
Compared to score-and-search algorithms (i.e. [Bartlett and Cussens, 2013]; [Scanagatta et al., 2017]), our Gibbs sampler has the following advantages:
At each step after the burin-in, the Gibbs sampler acts like the algorithm for selecting the forward or backward variable in the regression. In this way, based on the full conditional distributions, the status of the edge between two nodes (existing or not) as a sample of Bernoulli distribution is checked. Thus, samples of the posterior distribution of the graph are obtained. According to these samples, under each loss function (i.e., mean squared error, mean absolute error and $\cdots$), an estimation of the probability of being edge between the two nodes can be calculated. It is clear that by having these probabilities, not only do we obtain an approximation of the graph, but we will also have an approximation of the uncertainty corresponding to the status of the edge (existing or not). Consequently, the greater the probability will result in a greater chance of having an edge and less uncertainty corresponding to that edge. Furthermore, using the samples of the posterior distribution, we can also calculate the credible confidence Highest Posterior Density (HPD), Bayes Factor, and Hypothesis Testing related to each edge.

The organization of the paper is as follows. Section 2 introduces our proposed method. Experimental evaluation and Discussion are presented in sections 3 and 5, respectively.

2 PROPOSED METHOD

Suppose we have a domain of p variables $\mathbf{U}$ and a complete database of cases $\mathbf{D}$. We wish to determine the Bayesian network structure B over $\mathbf{U}$. Our approach for estimating the marginal posterior $\pi(B|\mathbf{D})$ is Gibbs sampling. Gibbs sampler involves ordering the parameters and sampling from the conditional distribution for each parameter given the current value of all the other parameters and repeatedly cycling through this updating process [Gelfand and Smith, 1990]. For learning BN using the Gibbs sampler, we redefine the unknown structure B by a set of new parameters $\mathbf{e} = \{e_{uv}, 1 \leq u < v \leq p\}$ as follows:

$$e_{uv} = \begin{cases} 0 & \text{There is no edge between } u \text{ and } v, \\ 1 & \text{There is a directed edge from } u \text{ to } v, \\ -1 & \text{There is a directed edge from } v \text{ to } u \end{cases}$$

In other words, the existence and the direction of the edges in B are specified by a set of parameters $\{e_{uv}\}$. This means that

$$\pi(B|\mathbf{D}) = \pi(\{e_{uv}\}|\mathbf{D}). \qquad (2.1)$$

The samples from $\pi(\{e_{uv}\}|\mathbf{D})$ represent the estimated BN. Thus, we need to calculate the following full conditional distribution in the context of Gibbs sampling:

$$\pi(e_{uv}|\{e_{-uv}\}, \mathbf{D}) \propto P(\mathbf{D}|\{e_{uv}\})\pi(e_{uv}|\{e_{-uv}\}), \quad (2.2)$$

where $\{e_{-uv}\}$ is a set of edges except e_{uv}. To determine the full conditional distribution in equation (2.2), we also need to determine the prior distribution $\{e_{uv}\}$ which is a critical issue in Bayesian analysis. Since in number of cases, faithful prior information about parameters is available (expert knowledge), the proposed method concern cases where no reliable information is available. This has led us to use a non-informative discrete uniform prior. More precisely, we consider the non-informative discrete uniform prior on all acyclic network. More precisely, if B is an arbitrary graph and the $\mathbf{B}_p$ is the set of all DAGs corresponding to p nodes, then

$$\pi(B) = \frac{1}{\#n(\mathbf{B}_p)}, \qquad B \in \mathbf{B}_p$$

where $\#n(\mathbf{B}_p)$ is the number of graphs in $\mathbf{B}_p$. This means that if $\{e_{uv}\}$ is the set of edges in B, then for $\pi(e_{uv} =$

$c|\{e_{-uv}\}), c = -1, 0, 1$, if all values of c will result in a acyclic network, then:

$$\pi(e_{uv} = c|\{e_{-uv}\}) = \frac{\pi(e_{uv}=c,\{e_{-uv}\})}{\sum_{i=-1}^{1}\pi(e_{uv}=i,\{e_{-uv}\})} = \frac{\frac{1}{\#n(\mathbf{B}_p)}}{\frac{1}{\#n(\mathbf{B}_p)}+\frac{1}{\#n(\mathbf{B}_p)}+\frac{1}{\#n(\mathbf{B}_p)}} = \frac{1}{3},$$

and if for one value of c, the graph becomes cyclic, then the prior probability $\pi(e_{uv} = c|\{e_{-uv}\})$ is uniformly distributed only on the other values of c that make the graph acyclic. More specifically, $\pi(e_{uv} = c|\{e_{-uv}\})$ is zero if c makes the graph acyclic and is

$$\pi(e_{uv} = c|\{e_{-uv}\}) = \begin{cases} 0 & c \text{ makes the graph cyclic} \\ \frac{\frac{1}{\#n(\mathbf{B}_p)}}{\frac{1}{\#n(\mathbf{B}_p)}+\frac{1}{\#n(\mathbf{B}_p)}+0} = \frac{1}{2} & c \text{ makes the graph acyclic} \end{cases}$$

With these settings, we can take samples from the posterior $\pi(\{e_{uv}\}|\mathbf{D})$ in the framework of the Gibbs sampling. These samples indicate the BNs at iterations of Gibbs sampling. More precisely, if $\{e_{uv}^{(t)};\ t = 1, \cdots, T\}$, is a sample from the posterior distribution of e_{uv}, then Monte Carlo Bayesian estimation of the probability of occurrence of $c \in -1, 0, 1$ between nodes u and v is as follows:

$$\hat{p}_{uv}^{c} = \frac{n_{uv}^{c}}{T} \tag{2.3}$$

where n_{uv}^{c} is the number of times that the values of c is observed in the posterior samples e_{uv}. In this way, we reject $H_0 : e_{uv} = c \in \{-1, 0, 1\}$, if $\hat{p}_{uv}^{c}$ is less than a threshold (say τ). If the Bayesian hypothesis tester wishes to guard against falsely rejecting H_0, he may decide to reject H_0 only if $\hat{p}_{uv}^{c}$ is less than some smaller number. Therefore, τ can be interpreted as a tuning parameter that the level of sparsity of the DAG is adjusted by it. When we believe the DAG is a high-sparse matrix, by considering a value greater than half for τ, higher accuracy is expected in the inferences.

Remark 1. The articles and approaches reviewed so far are all related to the situation that the ordering of variables is not available. But, for several applications in genetics, finance, and climate sciences, a location or time-based ordering of variables is naturally available. For example, in genetic data sets, the variables can be genes or SNPs located contiguously on a chromosome, and their spatial location provides a natural ordering. More examples can be found in Huang et al. [2006], Shojaie and Michailidis [2010], Yu and Bien [2017], Khare et al. [2016] and Cao et al. [2019]. Learning DAG for such applications is restricted to determine the parent of variables given the ordering of the variables. Hence, when the ordering of the variables is presented, e_{uv} gets only two values:

$$e_{uv} = \begin{cases} 0 & \text{There is no edge between } u \text{ and } v, \\ 1 & \text{There is a directed edge from } u \text{ to } v \end{cases}$$

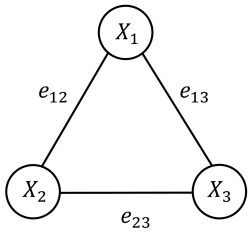

Figure 1: An example

In this way, all values of c will result in a acyclic network, and for the full conditional prior of e_{uv} we have:

$$\pi(e_{uv} = c|\{e_{-uv}\}) = \frac{\frac{1}{\#n(\mathbf{B}_p)}}{\frac{1}{\#n(\mathbf{B}_p)}+\frac{1}{\#n(\mathbf{B}_p)}} = \frac{1}{2}, \quad c \in \{0, 1\}$$

2.1 ILLUSTRATION AN EXAMPLE

We explain our idea by an example. Assume that we have three discrete nodes $\mathbf{U} = \{x_1, x_2, x_3\}$ (Figure 1). For Gibbs sampler, we must calculate the full conditional probabilities:

$$\pi(e_{12}|\{e_{-12}\}, \mathbf{D}) = \pi(e_{12}|\mathbf{D}, e_{13}, e_{23}),$$
$$\pi(e_{13}|\{e_{-13}\}, \mathbf{D}) = \pi(e_{13}|\mathbf{D}, e_{12}, e_{23}),$$
$$\pi(e_{23}|\{e_{-23}\}, \mathbf{D}) = \pi(e_{23}|\mathbf{D}, e_{12}, e_{13}).$$

Note that in Gibbs sampling, we start with a random BN. Thus, at iteration t of the Gibbs sampling, we need to determine the value of $c \in \{-1, 0, 1\}$ by the following way:

$$\pi(e_{12}^{t} = c|\mathbf{D}, e_{13}^{t-1}, e_{23}^{t-1})$$
$$\propto P(\mathbf{D}|e_{12}^{t} = c, e_{13}^{t-1}, e_{23}^{t-1}).\pi(e_{12}^{t} = c|e_{13}^{t-1}, e_{23}^{t-1}),$$
$$\pi(e_{13}^{t} = c|\mathbf{D}, e_{12}^{t}, e_{23}^{t-1})$$
$$\propto P(\mathbf{D}|e_{13}^{t} = c, e_{12}^{t}, e_{23}^{t-1}).\pi(e_{13}^{t} = c|e_{12}^{t}, e_{23}^{t-1}),$$
$$\pi(e_{23}^{t} = c|\mathbf{D}, e_{12}^{t}, e_{13}^{t})$$
$$\propto P(\mathbf{D}|e_{23}^{t} = c, e_{12}^{t}, e_{13}^{t}).\pi(e_{23}^{t} = c|e_{12}^{t}, e_{13}^{t})$$

Now, we want to estimate the value of c at iteration t of the Gibbs sampling. If the graph is acyclic for all values of c, then:

$$\pi(e_{12}^{t} = c|\mathbf{D}, e_{13}^{t-1}, e_{23}^{t-1}) = \frac{P(\mathbf{D}|e_{12}^{t} = c, e_{13}^{t-1}, e_{23}^{t-1})}{\sum_{i=-1}^{1} P(\mathbf{D}|e_{12}^{t} = i, e_{13}^{t-1}, e_{23}^{t-1})}$$

Also, it can easily concluded that if the graph for $c = -1$ becomes cyclic, then $\pi(e_{12}t = -1|\mathbf{D}, e_{13}^{t-1}, e_{23}^{t-1})$ becomes zero and for the other values of $c \in \{0, 1\}$, we have:

$$\pi(e_{12}^{t} = c|\mathbf{D}, e_{13}^{t-1}, e_{23}^{t-1}) = \frac{P(\mathbf{D}|e_{12}^{t} = c, e_{13}^{t-1}, e_{23}^{t-1})}{\sum_{i=0}^{1} P(\mathbf{D}|e_{12}^{t} = i, e_{13}^{t-1}, e_{23}^{t-1})}$$

Sampling from these full conditional distributions is repeated until we get burn-in networks.

As can be seen, to sample from the full conditional distributions assumed in equation (2.2), it is necessary to specify the likelihood function $P(\mathbf{D}|\{e_{uv}\}) = P(\mathbf{D}|B)$. In the following, this likelihood function is determined under discrete and continuous variables.

2.2 DISCRETE DATA

Suppose we have a domain of discrete variables, we wish to determine the likelihood function $P(\mathbf{D}|B)$. For this, we need to use the "Complete Data", "Multinomial Sample", "Drichlet", "Parameter Independence" and "Parameter Modularity" assumptions taken from Heckerman et al. [1995].
"Complete Data": This assumption says that all databases are complete.
"Multinomial Sample": Given domain $\mathbf{U}$ and $\mathbf{D}$, let D_l denote the first $l-1$ cases in the database. In addition, let x_{il} and Π_{il} denote the variable x_i, and the parent set Pa_i in the l'th case, respectively. Then, for a Bayesian network structure B in $\mathbf{U}$, there exist positive parameters $\mathbf{\Theta}$ such that, for $i = 1, \cdots, p$, and for all $k, k_1, \cdots, k_{i-1}$,

$$P(x_{il} = k | x_{1l} = k_1, \cdots, x_{(i-1)l} = k_{i-1}, D_l, \mathbf{\Theta}) = \theta_{ijk}, \tag{2.4}$$

where j is the state of Pa_{il} consistent with $\{x_{1l} = k_1, \cdots, x_{(i-1)l} = k_{i-1}\}$. Note that θ_{ijk} denote the Multinomial Parameters. If we let N_{ijk} be the number of cases in database D in which $x_i = k$ and $Pa_i = j$, then

$$P(\mathbf{D}|\mathbf{\Theta}, B) = \prod_i \prod_j \prod_k \theta_{ijk}^{N_{ijk}}. \tag{2.5}$$

"Drichlet Assumption": Let define $\mathbf{\Theta}_{ij} = \cup_{k=1}^{r_i} \theta_{ijk}$, $\mathbf{\Theta}_i = \cup_{j=1}^{q_i} \mathbf{\Theta}_{ij}$ and $\mathbf{\Theta} = \cup_{i=1}^{p} \mathbf{\Theta}_i$, in which r_i is the number of states of variable x_i and q_i is the the number of states of Pa_i. Then the proper prior distribution of $\mathbf{\Theta}_{ij}$ is Dirichlet. This assumption says that there exists exponents N'_{ijk}, which depend on given network B, that satisfy

$$\pi(\mathbf{\Theta}_{ij}|B) = c. \prod_k \theta_{ijk}^{N'_{ijk}-1}, \tag{2.6}$$

where c is a normalization constant. When every parameter set of B has a Dirichlet distribution, we simply say that $\pi(\mathbf{\Theta}|B)$ is also Dirichlet. Combining the Dirichlet assumption and equation (2.5), the following posterior probability is obtained:

$$\pi(\mathbf{\Theta}|\mathbf{D}, B) = c. \prod_i \prod_j \prod_k \theta_{ijk}^{N'_{ijk}+N_{ijk}-1}. \tag{2.7}$$

Based on the fact that the Dirichlet distributions are conjugate for the database, the posterior distribution of each parameter $\mathbf{\Theta}$ also remains in Dirichlet family.

"Parameter Independence": Given a network structure B, we have:

a) $\pi(\mathbf{\Theta}|B) = \prod_{i=1}^{n} \pi(\mathbf{\Theta}_i|B)$,
b) $\pi(\mathbf{\Theta}_i|B) = \prod_{i}^{q_i} \pi(\mathbf{\Theta}_{ij}|B)$.

The assumption (a) says that the parameters associated with each variable in a network structure are independent. This assumption is called Global Parameter Independence. Assumption (b) says that the parameters associated with each state of the parents of a variable are independent. This assumption is called Local Parameter Independence.

"Parameter Modularity": Given two network structures B_1 and B_2, if x_i has the same parents in B_1 and B_2, then:

$$\pi(\mathbf{\Theta}_{ij}|B_1) = \pi(\mathbf{\Theta}_{ij}|B_2).$$

This assumption says that the densities for parameters $\mathbf{\Theta}_{ij}$ depend only on the structure of the network.

Based on the consequences of these assumptions, the following formula is obtained for the likelihood function $P(\mathbf{D}|\{e_{uk}\})$. For calculating this formula, we use the BDe (Bayesian Dirichlet likelihood equivalent) metric (Heckerman et al. [1995]).

$$P(\mathbf{D}|\{e_{uv}\}) = \prod_{i=1}^{p} \prod_{j=1}^{q_i} \frac{\Gamma(N'_{ijk})}{\Gamma(N'_{ijk}+N_{ij})} \prod_{k=1}^{r_i} \frac{\Gamma(N'_{ijk}+N_{ijk})}{\Gamma(N'_{ijk})} \tag{2.8}$$

In this paper, we use a simple uninformative assignment $N^{ijk} = 1$. This special case of BDe is called K2 metric.

Remark 2. According to the product rule of probability, we have:

$$\pi(\mathbf{\Theta}, B|\mathbf{D}) = \pi(\mathbf{\Theta}|B, \mathbf{D})\pi(B|\mathbf{D}). \tag{2.9}$$

Hence, to sample from the joint posterior distribution $\mathbf{\Theta}$ and B, we first sample from the posterior $\pi(B|\mathbf{D})$ (as discussed in the previous subsection) and replace it in full conditional posterior $\pi(\mathbf{\Theta}|B, \mathbf{D})$. We then have samples of the posterior distribution $\pi(\mathbf{\Theta}, B|\mathbf{D})$. Using these posteriors samples, the Monte Carlo Bayesian estimation of the Multinomial parameters can also be computed.

2.3 CONTINUOUS DATA

For continuous data, besides Parameter Independence and Parameter Modularity, we use other assumptions taken from Geiger and Heckerman [1994] for calculating $P(\mathbf{D}|B)$. It is assumed that complete continuous data $\mathbf{D}$ is a random sample from a multivariate normal distribution with unknown means $\mathbf{m}$ and unknown precision matrix W, and the prior distribution and W is a normal-Wishart distribution. Then the posterior joint distribution of $\mathbf{m}$ and W is as follows: The conditional distribution of $\mathbf{m}$ given W is a multivariate normal distribution and marginal of W is Wishart. Considering these assumptions, the BGe which stands for Bayesian metric for Gaussian networks having score equivalence is obtained and $P(\mathbf{D}|B)$ is calculated for continuous data (Geiger and Heckerman [1994]).

2.4 ORDERING-BASED SEARCH ALGORITHMS

In this paper, we use the topological ordering of variables as input of the proposed Gibbs sampler. Topological ordering

is a linear ordering of vertices such that for every directed edge u and v, vertex u comes before v in the ordering. Topological ordering for a graph is not possible if the graph is not a DAG. Fortunately, for a given ordering on the nodes of a Bayesian network, the problem of finding the best-scoring networks consistent with respect to ordering is not NP-hard. Thus, we compare the results of the proposed method with the order-based algorithms.

One of the most popular ordering-based search algorithm is K2 (Cooper and Herskovits [1992]). This greedy algorithm uses a prior ordering of nodes as input for reducing the complexity of the search space. A novel algorithm that obtains a node ordering from data has been proposed by (Chen et al. [2008]). By using this ordering as input to the K2 algorithm the structure of the BN is learned. This algorithm aims to identify information for the correct ordering of nodes and runs in three phases. First, it finds an undirected network (UDN) using mutual information (MI) and interdependence tests. Second, the UDN is refined using d-separation and the conditional independence test to eliminate possible false edges and add true edges that may have been missed in the original UDN. Finally, orientations are assigned for each edge using inter-dependency tests and Bayesian scoring metrics.

Another ordering-based method is the one introduced by Liu et al. [2007]. This method is based on the ordering-based Max-Relevance and Min-Redundancy Greedy (OMRMRG) which is an Ordering based Max Relevance and Min Redundancy Greedy algorithm. OMRMRG presents an ordering-based greedy search method with a greedy pruning procedure, applies Max-Relevance and Min-Redundancy feature selection method, and proposes Local Bayesian Increment function according to BIC score function. The OMRMRG algorithm is divided into two parts. The first part is to learn Bayesian network given an ordering on the variables. The second part is to learn Bayesian network without the constraint of an ordering on the variables.

The state-of-the-art algorithms for ordering-based method are Ko and Kim [2014], Tabar et al. [2018], and Behjati and Beigy [2020]. Ko and Kim [2014] introduced a measure to evaluate conditional frequency for node ordering. They focus on the discrete Dirichlet probability density function of a child depending on its parents. Based on the fact that the candidate parents are identified by estimated Markov Blanket, Tabar et al. [2018] first estimated the Markov Blanket of a variable by using the L1-regularized Markov Blanket. They then determined the candidate parents of a variable through its Markov blanket by introducing a new scoring function based on the dependency criterion. Finally the candidate parents were used as input for the K2 algorithm for learning Bayesian network structure. Behjati and Beigy [2020] introduced a hybrid algorithm which is based on a partial ordering learned from data. They reduced the super-exponential search space of structures to the smaller ordering space of nodes.

3 PROPOSED ALGORITHM

In general, the argument of the proposed algorithm is as follows:

- simulate: The total number of samples to be simulated,
- burn-in: The burn-in time. burn-in time is the number of samples in MCMC sampling to be discarded as burn-in phase,

then the Gibbs algorithm can be summarized as follows:

1. Let $(x_1, \cdots, x_p)$ be a topological ordering of the variables.
2. Determine the starting value of the parameters $\{e_{uv}\}$ and $\mathbf{\Theta}$.
3. for r=1 to simulate:
   - for u=1 to (p-1):
     - for v=(u+1) to p:
       a) Compute $P(\mathbf{D}|\{e_{uv}\})$ (using K2 or BGe).
       b) Sample a state c for e_{uv} using $\pi(e_{uv} = c|\{e_{-uv}\}, \mathbf{D})$.
   - if r>burn-in, then:
     - Storage simulated value for $\{e_{uv}\}$.
4. In step 3, $T = simulate - burnin$ network structures generated. If we denote them as $B^{(t)}, \cdots, B^{(T)}$, then for t=1 to T:
   - Generate $\mathbf{\Theta}^{(t)}$ using sampling from $\pi(\mathbf{\Theta}|\mathbf{D}, B^{(t)})$.
5. Calculate the Monte Carlo Bayesian estimation of the parameters as follows:
   - $\hat{\mathbf{\Theta}} = \frac{1}{T}\sum_{t=1}^{T} \mathbf{\Theta}^{(t)}$.
   - $\hat{p}_{uv}^{c} = \frac{1}{T}\sum_{t=1}^{T} e_{uv}^{(t)}, c \in \{0, 1\}$.
     - if $\hat{p}_{uv}^{c} > \tau$, then:$\rightarrow \hat{e}_{uv} = c$,

4 CONVERGENCE CONDITIONS FOR PROPOSED ALGORITHM

Convergence of a Gibbs sampler can be followed from the Hammersley-Clifford theorem (Besag [1974]). The theorem gives a positivity condition that is sufficient to prove that the univariate conditional distributions used by the Gibbs sampler uniquely define the joint distribution. The required condition is that the support of the joint distribution is given by the Cartesian product of the supports of the marginal distributions. In the DAG setting, the acyclicity requirement means that this positivity condition is not satisfied. Consider a DAG consisting of two correlated random variables X_1 and X_2. The correlation means that both

the graph with a single edge $(1,2)$ and the graph with a single edge $(2,1)$ have positive posterior probability. Thus $\pi(e_{uv}=1|D)>0$ and $\pi(e_{uv}=-1|D)>0$ in the posterior marginal distributions. However, in the joint posterior distribution $\pi(e_{uv}=1, e_{uv}=-1|D)=0$ because the corresponding graph (the complete graph) is cyclic. The complete graph is thus not in the support of the joint distribution but is in the Cartesian product of the supports of the marginal distributions. An alternative sufficient condition for uniqueness of the joint distribution and convergence of the Gibbs sampler when positivity is not satisfied is given by Besag [1974]. In the present context, the condition requires that for every pair $B', B'' \in \mathbf{B}_p$ of DAGs, there exists a finite sequence $B_1, \cdots, B_N$, with $B_1 = B', B_N = B''$, and such that B_t and B_{t-1} differ in only a single edge, and that $\pi(B_t|D)>0$ for all $t=1,\cdots,N$. When the graph prior $\pi(B)>0$ for all $B \in \mathbf{B}_p$, this condition is clearly satisfied and so the convergence of the Gibbs sampler is guaranteed.

5 EXPERIMENTAL EVALUATION

In this section, we perform simulation experiments to illustrate the potential advantages of our algorithm. The existence of the original DAGs also allows us to define important terms, which indicate the performance of our approach. In this line, we compare the edge scores by computing the number of edges that are correct, missing, reversed and additional compared to the original DAG by the following definitions:

- True Positive (TP): The edges that are detected with the same edge direction of the original DAG.
- False Positive (FP): The edges that are presented in learned DAG and not presented in the original DAG (additional and reversed edges)
- True Negative (TN): The edges that are not presented in both learned and original DAG.
- False Negative (FN): The edges that are not detected in learned DAG and are presented in original DAG (missing edges).

The model selection performance of these four methods is then compared using several different measures of structure such as accuracy rate (ACC). Hence, ACC is defined as follows:

$$ACC = \frac{TP+TN}{TP+TN+FN+FP}, \tag{5.1}$$

One would like the ACC value to be as close to 1 as possible.

5.1 SIMULATION I: LEARNING SOME GUASSIAN DAGS

In this section, we simulate a Gaussian DAGs for different nodes (p) and sample size (n). We then compare the results with the method introduced by [Cao et al., 2020] in which they focus on Gaussian DAG models and consider a flexible and general class of these 'DAG-Wishart' priors with multiple shape parameters. The same as proposed method in this paper, [Cao et al., 2020] without loss of generality assumed a node ordering.
The obtained results are provided in Table 1. As shown, our proposed method has higher accuracy than DAG-Wishart algorithm. In other words, the accuracy of our proposed method for all different p and n is higher than DAG-Wishart. The best accuracy for DAG-Wishart is 0.96 and for our proposed method is 1.

Table 1: Accuracy for proposed method and DAG-Wishart method

p	n	Proposed Method	DAG-Wishart
20	100	0.98	0.93
20	200	1	0.94
50	1000	1	0.96

5.2 SIMULATION II: LEARNING SOME STANDARD GAUSSIAN DAGS

In this section, we present the other experimental results carried out with our algorithm on the four standard Gaussian DAGs from bnlearn: $Gaussian.test$, $ECOLI70$, $MAGI-IRRI$ and $ARTH150$. We generate 5000 cases from each DAG in order to perform multiple tests and estimate more precise metrics.

1. The $Gaussian.test$ data set contains 7 Gaussian variables and 7 arcs.
2. The $ECOLI70$ has 46 nodes and 70 arcs (Schäfer and Strimmer [2005]).
3. The $MAGIC-IRRI$ has 64 nodes and 102 arcs (Scutari et al. [2014]).
4. The ARTH150 has 107 nodes and 150 arcs (Opgen-Rhein and Strimmer [2007]).

The results of this competition are summarized in Table 2 where the values of TP, TN, FP, FN, and ACC are reported. The interesting point is that our proposed algorithm for learning DAGs among continuous variables has highest accuracy. Note that we consider the value greater than half for τ. Thanks to the devices with 8 NVIDIA V100 GPU, the running time of for Gaussian.test, ECOLI, MAGICS and ARTH150 is 30 seconds, and 100, 200 and 350 minutes respectively and it shows that the running time is reasonable.

5.3 SIMULATION III: LEARNING SOME STANDARD DISCRETE DAGS

In this section, we present the experimental results carried out with our algorithm on the five standard discrete

Table 2: Edge Scores

Data	Edge Type	$\tau \geq 0.5$
Gaussian.test	TP	7
	FP	0
	TN	14
	FN	0
	ACC	1
ECOLI70	TP	69
	FP	2
	TN	963
	FN	1
	ACC	0.99
MAGIC-IRRI	TP	101
	FP	1
	TN	1913
	FN	1
	ACC	0.99
ARTH150	TP	149
	FP	2
	TN	5519
	FN	1
	ACC	0.99

Table 3: Edge Scores

Data	Edge Type	$\tau \geq 0.5$	Behjat	Tabar	RSMAX2
Learning.test	TP	5	4	4	5
	FP	0	1	1	0
	TN	10	10	10	10
	FN	0	0	0	0
	ACC	1	0.93	0.93	1
Asia	TP	8	6	5	5
	FP	0	1	4	3
	TN	20	20	19	20
	FN	0	1	0	0
	ACC	1	0.928	0.857	0.890
Insurance	TP	49	43	45	21
	FP	1	10	9	31
	TN	299	294	293	292
	FN	2	4	4	7
	ACC	0.991	0.960	0.962	0.891
Alarm	TP	43	41	41	13
	FP	1	13	15	33
	TN	619	609	609	605
	FN	3	3	1	15
	ACC	0.993	0.975	0.975	0.927
Hailfinder	TP	61	50	50	34
	FP	17	35	40	32
	TN	1457	1432	1425	1474
	FN	5	23	25	0
	ACC	0.985	0.962	0.957	0.970

DAGs: *Learning.test*, *Asia*, *Insurance*, *Alarm* and *Hailfinder*.

1. The Learning.test DAG has 6 variables and 5 edges. This synthetic data set used as a test case in the bnlearn package.
2. The *Asia* DAG has 8 variables and 8 edges Lauritzen and Spiegelhalter [1988]. This is about lung diseases (tuberculosis, lung cancer or bronchitis) and visits to Asia
3. The Insurance DAG has 27 variables and 52 edges (Binder et al. [1997]). Insurance is a network for evaluating car insurance risks.
4. The *Alarm* DAG has 37 nodes and 46 arcs (Beinlich et al. [1989]). It was introduced as a network for monitoring patients in intensive care.
5. The *Hailfinder* DAG has 56 nodes and 66 arcs (Abramson et al. [1996]). Hailfinder is a network designed to forecast severe summer hail in northeastern Colorado.

For Discrete DAGs, we compare the results with the state-of-the-art algorithms Tabar et al. (2018), Behjati and Beigy (2020), and the RSMAX2 hybrid algorithm. The results of this competition are summarized in Table 3. According to results, the accuracy of the proposed method for all DAGs is higher than 0.98 and outperforms other methods.

The running time for discrete DAGs Learning.test, Asia, Insurance, Alarm and Hialfinder is 20 seconds, 50 seconds, 60 minutes, 200 minutes and 400 minutes respectively.

6 DISCUSSION

As it is known, the number of BN structures is super-exponential in the number of random variables in the domain. Consequently, the summation of all possible structures can be computed in a closed form only for small domains, or those in which we have supplemental constraints that restrict the space. In this paper we focus on both continuous and discrete data and take advantage of a novel Gibbs sampler for learning BN. The results for both continuous and discrete data suggest that our method can estimate the structures of BNs with reasonable accuracy.
It should be noted that one of the advantages of our proposed method is that it works even without considering the node ordering. For this purpose, we need to consider the three states $\{-1, 0, 1\}$ for each edge (e_{uv}). This means that at every step of the Gibbs sampler, we need to sample $\{-1, 0, 1\}$ from a multinomial distribution. We also need to check the acyclicity by

- Proposition 1.4.2 in "Digraphs Theory, Algorithms, and Applications" by Bang-Jensen and Gutin, page 13 or
- by using the Bayes Net Toolbox for Matlab written by Kevin Murphy, 2014 (`https://github.com/bayesnet/bnt`).

In this paper, for simplicity and demonstrating the accuracy of the proposed algorithm, we assume that order belongs to the input. Working without node ordering is under progress. As a simple example, we perform the proposed algorithm on small DAGs Learning.test and Gaussian.test DAGS. The proposed method without considering the node ordering has also the highest accuracy (ACC=1) with no additional, missing and reversed edges. In this case, the running time is definitely more than proposed method by considering node ordering.

References

Bruce Abramson, John Brown, Ward Edwards, Allan Murphy, and Robert L Winkler. Hailfinder: A bayesian system for forecasting severe weather. *International Journal of Forecasting*, 12(1):57–71, 1996.

Mark Bartlett and James Cussens. Advances in bayesian network learning using integer programming. *arXiv preprint arXiv:1309.6825*, 2013.

Shahab Behjati and Hamid Beigy. Improved k2 algorithm for bayesian network structure learning. *Engineering Applications of Artificial Intelligence*, 91:103617, 2020.

Ingo A Beinlich, Henri J Suermondt, R Martin Chavez, and Gregory F Cooper. *The ALARM monitoring system: A case study with two probabilistic inference techniques for belief networks*. Springer, 1989.

Julian Besag. Spatial interaction and the statistical analysis of lattice systems. *Journal of the Royal Statistical Society: Series B (Methodological)*, 36(2):192–225, 1974.

John Binder, Daphne Koller, Stuart Russell, and Keiji Kanazawa. Adaptive probabilistic networks with hidden variables. *Machine Learning*, 29(2):213–244, 1997.

Xuan Cao, Kshitij Khare, and Malay Ghosh. Posterior graph selection and estimation consistency for high-dimensional bayesian dag models. *The Annals of Statistics*, 47(1):319–348, 2019.

Xuan Cao, Kshitij Khare, and Malay Ghosh. Consistent bayesian sparsity selection for high-dimensional gaussian dag models with multiplicative and beta-mixture priors. *Journal of Multivariate Analysis*, 179:104628, 2020.

Xue-Wen Chen, Gopalakrishna Anantha, and Xiaotong Lin. Improving bayesian network structure learning with mutual information-based node ordering in the k2 algorithm. *IEEE Transactions on Knowledge and Data Engineering*, 20(5):628–640, 2008.

Max Chickering, David Heckerman, and Chris Meek. Large-sample learning of Bayesian networks is np-hard. *Journal of Machine Learning Research*, 5:1287–1330, 2004.

Gregory F Cooper and Edward Herskovits. A bayesian method for the induction of probabilistic networks from data. *Machine learning*, 9(4):309–347, 1992.

Nir Friedman and Daphne Koller. Being Bayesian about network structure. a Bayesian approach to structure discovery in Bayesian networks. *Machine learning*, 50(1-2): 95–125, 2003.

Maxime Gasse, Alex Aussem, and Haytham Elghazel. A hybrid algorithm for Bayesian network structure learning with application to multi-label learning. *Expert Systems with Applications*, 41(15):6755–6772, 2014.

Dan Geiger and David Heckerman. Learning gaussian networks. In *Proceedings of the Tenth international conference on Uncertainty in artificial intelligence*, pages 235–243. Morgan Kaufmann Publishers Inc., 1994.

Alan E Gelfand and Adrian FM Smith. Sampling-based approaches to calculating marginal densities. *Journal of the American statistical association*, 85(410):398–409, 1990.

Paolo Giudici and Robert Castelo. Improving markov chain monte carlo model search for data mining. *Machine learning*, 50(1-2):127–158, 2003.

Robert JB Goudie and Sach Mukherjee. A gibbs sampler for learning DAGs. *Journal of Machine Learning Research*, 17(2):1–39, 2016.

David Heckerman. A tutorial on learning with Bayesian networks. In *Learning in graphical models*, pages 301–354. Springer, 1998.

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

David Heckerman, Christopher Meek, and Gregory Cooper. A Bayesian approach to causal discovery. In *Innovations in Machine Learning*, pages 1–28. Springer, 2006.

Jianhua Z Huang, Naiping Liu, Mohsen Pourahmadi, and Linxu Liu. Covariance matrix selection and estimation via penalised normal likelihood. *Biometrika*, 93(1):85–98, 2006.

Kshitij Khare, Sang Oh, Syed Rahman, and Bala Rajaratnam. A convex framework for high-dimensional sparse cholesky based covariance estimation. *arXiv preprint arXiv:1610.02436*, 2016.

Song Ko and Dae-Won Kim. An efficient node ordering method using the conditional frequency for the k2 algorithm. *Pattern Recognition Letters*, 40:80–87, 2014.

Daphne Koller and Nir Friedman. *Probabilistic graphical models: principles and techniques*. MIT press, 2009.

Steffen L Lauritzen and David J Spiegelhalter. Local computations with probabilities on graphical structures and their application to expert systems. *Journal of the Royal Statistical Society. Series B (Methodological)*, pages 157–224, 1988.

Feng Liu, Fengzhan Tian, and Qiliang Zhu. A novel ordering-based greedy bayesian network learning algorithm on limited data. In *Seventh IEEE International Conference on Data Mining Workshops (ICDMW 2007)*, pages 495–500. IEEE, 2007.

David Madigan, Jeremy York, and Denis Allard. Bayesian graphical models for discrete data. *International Statistical Review/Revue Internationale de Statistique*, pages 215–232, 1995.

C Meek. Causal inference and causal explanation with background knowledge in uncertainty in artificial intelligence 11, 1995.

Teppo Niinimaki, Pekka Parviainen, and Mikko Koivisto. Partial order mcmc for structure discovery in Bayesian networks. *arXiv preprint arXiv:1202.3753*, 2012.

Rainer Opgen-Rhein and Korbinian Strimmer. From correlation to causation networks: a simple approximate learning algorithm and its application to high-dimensional plant gene expression data. *BMC systems biology*, 1(1):1–10, 2007.

Judea Pearl. *Probabilistic reasoning in intelligent systems: networks of plausible inference*. Elsevier, 1988.

Mauro Scanagatta, Giorgio Corani, and Marco Zaffalon. Improved local search in bayesian networks structure learning. In *Advanced Methodologies for Bayesian Networks*, pages 45–56. PMLR, 2017.

Mauro Scanagatta, Antonio Salmerón, and Fabio Stella. A survey on Bayesian network structure learning from data. *Progress in Artificial Intelligence*, 8(4):425–439, 2019.

Juliane Schäfer and Korbinian Strimmer. A shrinkage approach to large-scale covariance matrix estimation and implications for functional genomics. *Statistical applications in genetics and molecular biology*, 4(1), 2005.

Marco Scutari, Phil Howell, David J Balding, and Ian Mackay. Multiple quantitative trait analysis using bayesian networks. *Genetics*, 198(1):129–137, 2014.

Marco Scutari, Catharina Elisabeth Graafland, and José Manuel Gutiérrez. Who learns better Bayesian network structures: Accuracy and speed of structure learning algorithms. *International Journal of Approximate Reasoning*, 115:235–253, 2019.

Ali Shojaie and George Michailidis. Penalized likelihood methods for estimation of sparse high-dimensional directed acyclic graphs. *Biometrika*, 97(3):519–538, 2010.

Moninder Singh and Marco Valtorta. An algorithm for the construction of Bayesian network structures from data. In *Uncertainty in Artificial Intelligence*, pages 259–265. Elsevier, 1993.

Peter Spirtes and Clark Glymour. An algorithm for fast recovery of sparse causal graphs. *Social science computer review*, 9(1):62–72, 1991.

Chengwei Su and Mark E Borsuk. Improving structure mcmc for Bayesian networks through markov blanket resampling. *Journal of Machine Learning Research*, 17 (118):1–20, 2016.

Vahid Rezaei Tabar, Farzad Eskandari, Selva Salimi, and Hamid Zareifard. Finding a set of candidate parents using dependency criterion for the k2 algorithm. *Pattern Recognition Letters*, 111:23–29, 2018.

Ioannis Tsamardinos, Laura E Brown, and Constantin F Aliferis. The max-min hill-climbing Bayesian network structure learning algorithm. *Machine learning*, 65(1): 31–78, 2006.

Guo Yu and Jacob Bien. Learning local dependence in ordered data. *The Journal of Machine Learning Research*, 18(1):1354–1413, 2017.