Journal of Machine Learning Research 14 (2013) 1563-1603 Submitted 11/10; Revised 9/12; Published 6/13

Sub-Local Constraint-Based Learning of Bayesian Networks Using A Joint Dependence Criterion

Rami Mahdi RAMIMAHDI@YAHOO.COM
Department of Genetic Medicine
Weill Cornell Medical College
New York, NY, 10065, USA

Jason Mezey JGM45@CORNELL.EDU
Department of Biological Statistics and Computational Biology
Cornell University
Ithaca, NY, 14853, USA

Editor: Peter Spirtes

Abstract

Constraint-based learning of Bayesian networks (BN) from limited data can lead to multiple testing problems when recovering dense areas of the skeleton and to conflicting results in the orientation of edges. In this paper, we present a new constraint-based algorithm, light mutual min (LMM) for improved accuracy of BN learning from small sample data. LMM improves the assessment of candidate edges by using a ranking criterion that considers conditional independence on neighboring variables at both sides of an edge simultaneously. The algorithm also employs an adaptive relaxation of constraints that, selectively, allows some nodes not to condition on some neighbors. This relaxation aims at reducing the incorrect rejection of true edges connecting high degree nodes due to multiple testing. LMM additionally incorporates a new criterion for ranking v-structures that is used to recover the completed partially directed acyclic graph (CPDAG) and to resolve conflicting v-structures, a common problem in small sample constraint-based learning. Using simulated data, each of these components of LMM is shown to significantly improve network inference compared to commonly applied methods when learning from limited data, including more accurate recovery of skeletons and CPDAGs compared to the PC, MaxMin, and MaxMin hill climbing algorithms. A proof of asymptotic correctness is also provided for LMM for recovering the correct skeleton and CPDAG.

Keywords: Bayesian networks, skeleton, constraint-based learning, mutual min

1. Introduction

Learning a Bayesian network (BN) from observational data is a reverse engineering process that can provide insight into the direct relations between observed variables and can be used to establish causation (Pearl, 1988; Spirtes et al., 2001). BNs have been used in learning applications in a number of fields including systems biology (Friedman et al., 2000; Friedman, 2004), medicine (Cowell et al., 1999), and artificial intelligence (Russell and Norvig, 2009) where emerging applications and the growing availability of complex data sets containing thousands of variables are requiring faster, more scalable, and more accurate methods.

Unconstrained learning of a BN is a search for a network that fits the observational data with the highest posterior probability. However, due to the large number of all possible networks, which

is super exponential in the number of variables being modeled, an exhaustive search is not possible for more than a few tens of variables (Chickering et al., 2004) while heuristic search methods tend to converge to suboptimal solutions. This scalability issue is a known limitation of score-based methods, where a scoring criterion such as the Bayesian information criterion (BIC) (Cooper and Herskovits, 1992) or the minimum description length criterion (MDL) (Lam and Bacchus, 1994) is used to rank candidate networks.

An alternative learning approach to score-based search is the use of conditional independence testing, also referred to as constraint-based learning. Methods in this class such as the IC algorithm (inductive causation) (Pearl, 1988), PC algorithm (Spirtes et al., 2001), and TPDA algorithm (three-phase dependency analysis) (Cheng et al., 2002), first recover the skeleton of the network and edges are oriented afterward. The learning is performed in a way to ensure that the resulting network is consistent with the conditional independences/dependencies entailed by the observations. Under the faithfulness assumption (see Section 3 for definition), constraint-based methods were shown to recover the correct network as the size of the observed samples approaches infinity (Zhang and Spirtes, 2003; Kalisch and Bühlmann, 2007). Moreover, their computational complexity has a polynomial order when the maximum number of connections per node is bounded. Constraint-based methods have also been used in combination with score-based search in what typically is referred to as hybrid learning. Hybrid methods such as the SC algorithm (sparse candidate) (Friedman et al., 1999), MMHC algorithm (MaxMin hill climbing) (Tsamardinos et al., 2006), and the COS algorithm (constrained optimal search) (Perrier et al., 2008) first recover a super structure of the skeleton using a constraint-based approach. Afterward, a constrained score-based search is used to find an optimal network where the search is restricted only to edges existing in the super structure. This strategy can reduce the size of the search space considerably and can lead to higher score solutions.

Although constraint-based learning can, under appropriate assumptions, recover the correct graph in the asymptotic limit, its performance in real applications depends heavily on the accuracy of independence testing, which in turn is sensitive to noise and sample size. As will be discussed in this paper, when learning from small sample data, the use of conditional independence testing can lead to multiple testing problems that deteriorate the accuracy of skeleton recovery and to conflicting results in the orientation of edges. In addition, errors in the first phase of recovering the skeleton can also deteriorate the precision of orienting the edges, resulting in a propagated error.

2. Contribution

We present a new constraint-based algorithm, light mutual min (LMM), for learning Bayesian networks that has properties well suited for learning from limited sample data. For skeleton recovery, LMM improves the assessment of candidate edges by using a new mutual dependence criterion that considers conditional independence on subsets of neighboring variables at both sides of an edge simultaneously. In addition, we implement an adaptive relaxation of independence constraints in dense areas of the graph by, selectively, allowing only one node of a connected pair to be aware of the edge. This relaxation is only performed whenever asymmetric evidence of conditional independence is found between a pair of connected variables, where the aim is to reduce the accidental rejection of true edges connecting high degree nodes due to the multiple testing problem in the case of limited training samples. Another consequence of this relaxation is that sets of the recognized neighbors that are used for conditional independence testing remain small, leading to a considerable reduction in the number of required independence tests.

Moreover, we present a new approach to recover v-structures in a given skeleton based on the level of induced dependence caused by common neighbors, and LMM is extended to recover the completed partially directed acyclic graph (CPDAG) of the equivalence class. The name light mutual min is motivated by the fact that the algorithm is fast and uses reduced sets of neighbors (light) in addition to ranking edges using a measure that combines the minimal dependence from both sides of an edge (mutual min). Also, we refer to LMM as a sub-local approach due to using reduced sets of neighbors for independence testing as compared to local algorithms such as the PC and MaxMin algorithms which use subsets of all neighbors for conditional independence testing.

For completeness, a proof of asymptotic correctness is provided to show that the proposed approaches will recover the correct skeleton and CPDAG when the number of observations grows sufficiently large. Also, to empirically assess performance, we compare LMM to the PC and MaxMin algorithms, two of the most popular and computationally efficient BN learning methods (Spirtes et al., 2001; Tsamardinos et al., 2006; Kalisch et al., 2010). Based on empirical evaluation using simulated data, LMM is shown to significantly and consistently outperform both the PC and MaxMin algorithms in recovering the skeleton of the graph in terms of both accuracy and speed when learning from limited sample data. In addition, the extended LMM is found to recover more accurate CPDAGs than the PC algorithm, while being competitive with the MMHC algorithm in the small network case and more accurate in the large network case.

The rest of this paper is organized as follows: Section 3 presents necessary definitions. Section 4 presents related work. Section 5 presents a discussion of the limitations of existing methods. Section 6 presents our proposed methods. Section 7 presents experimental results and comparison to other methods. A proof of asymptotic correctness for the presented methods is provided in the Appendix for the recovery of both the skeleton and the CPDAG.

Availability: Our implementation of all presented methods in addition to a supplementary material of further discussion and illustrations are made available with this publication and they also can be accessed at `http://www.mloss.org/software/view/460/` or alternatively at `http://mezeysoftware.bscb.cornell.edu/index.php/LMM`.

3. Definitions and Preliminaries

In this section, we present necessary definitions and notations following Pearl (1988) and Neapolitan (2004) with slight variations.

Definition 1 (Directed Graph) *A directed graph $G = (V,E)$ consists of a set of nodes representing variables $V = \{1,...,p\}$ and a set of edges $E \subseteq V \times V$ where an edge $(i,j) \in E$ implies that $E(i,j)$ is an edge pointing from V_i (parent) to V_j (child). Two variables i and j are adjacent in G if and only if $(i,j) \in E$ or $(j,i) \in E$.*

In this paper, the notation CP_i^* is used to refer to the true set of child and parent variables of the variable i while the notation CP_i is used to refer to the set of child and parent variables of i that are being inferred by the algorithm. The set of child and parent variables are also sometimes referred to as the neighbor set of i.

Definition 2 (Directed Acyclic Graph (DAG)) *A directed graph $G = (V,E)$ is said to be acyclic if and only if for every node V_i in the graph, there does not exist a path of connected and directed edges such that starting from node V_i and following the direction of edges can lead back to the same node V_i.*

Definition 3 (Skeleton) *The skeleton of a directed graph $G = (V,E)$ is a graph that contains all the nodes and edges of G such that all the edges have no directions. Every undirected edge in the skeleton represents a parent-child relation without informing who is the parent and who is the child.*

Definition 4 (Conditional Independence) *Two variables x and y are conditionally independent given a set of variables $Z_{\setminus x,y}$ w.r.t a probability distribution P, denoted as $x \perp\!\!\!\perp y \mid Z$, if and only if $P(x,y \mid Z = z) = P(x \mid Z = z) \times P(y \mid Z = z)$, $\forall z$ where $P(Z = z) > 0$.*

The notation $Z_{\setminus x,y}$ means x and y are excluded from the conditioning set Z which is always the case for all methods presented in this paper even if it is not explicitly stated. Examples of methods to determine conditional independence are the use of statistical tests of partial correlation and the G-squared measure (Neapolitan, 2004).

Definition 5 (Bayesian Network) *A directed acyclic graph $G = (V,E)$ is said to be a Bayesian network w.r.t a probability distribution P if it satisfies the Markov condition (local Markov property): Every variable $x \in V$ is independent of any subset of its non-descendant variables conditioned on the set of its parents.*

Definition 6 (V-Structure) *An ordered triplet of nodes (x,w,y) forms a v-structure in a DAG if and only if x and y meet head to head at w ($x \rightarrow w \leftarrow y$) while x and y are not directly connected in the graph.*

Definition 7 (Faithfulness) *A graph G and a probability distribution P are said to satisfy the faithfulness condition (or to be faithful to one another) if and only if, based on the Markov condition, G entails all and only the conditional independence relations in P.*

Many researchers have suggested some constraints on BN inference that would facilitate finding sound solutions. The most used of these constraints is the faithfulness property, and it has been argued that, in most cases, the true BN will have such a property (Spirtes et al., 1993). As a consequence of the faithfulness assumption (Spirtes et al., 2001), an edge between a node x and a node y exists in G if and only if there does not exist a set $Z_{\setminus x,y}$ such that x and y are independent when conditioned on $Z_{\setminus x,y}$: $(\nexists Z \subseteq V_{\setminus x,y}\ s.t.\ x \perp\!\!\!\perp y \mid Z, w.r.t\ P)$.

Definition 8 (Blocked Path) *In a directed graph $G = (V,E)$, a path Pa of connected edges between two distinct nodes $x,y \in V$ is said to be blocked by a set of nodes $Z \subseteq V_{\setminus x,y}$ if one of the following holds:*

1. *There is a node $w \in Z$ on the path Pa where the edges incident to w on Pa meet head-to-tail at w ($.. \rightarrow w \rightarrow ..$ or $.. \leftarrow w \leftarrow ..$).*

2. *There is a node $w \in Z$ on the path Pa where the edges incident to w on Pa meet tail-to-tail at w ($.. \leftarrow w \rightarrow ..$).*

3. *There is a node w on the path Pa, such that w and all of w's descendants are not in Z, and the edges incident to w on Pa meet head-to-head at w ($.. \rightarrow w \leftarrow ..$).*

Definition 9 (D-Separation) *In a DAG $G = (V,E)$, two distinct nodes $x,y \in V$ are said to be d-separated by $Z \subseteq V_{\setminus x,y}$, denoted by $Dsep_G(x,y \mid Z)$, if every path between x and y is blocked by Z. Moreover, two disjoint sets of nodes $X,Y \subset V$ are said to be d-separated by $Z \subseteq V - (X \cup Y)$ if and only if every $x \in X$ and every $y \in Y$ are d-separated by Z.*

Theorem 10 (Pearl, 1988, D-Separation $\Leftrightarrow$ Conditional Independence) *Given a faithful BN of a DAG G and a probability distribution P, every d-separation in G entails a conditional independence relation in P and every conditional independence relation in P is represented by a d-separation in G:*

$$Dsep_G(x,y \mid Z) \Leftrightarrow x \perp\!\!\!\perp y \mid Z \ (w.r.t\ P).$$

Definition 11 (Markov Equivalence) *Two DAGs $G_1 = (V,E_1)$ and $G_2 = (V,E_2)$ are called Markov equivalent if for every three mutually disjoint subsets $X,Y,Z \subset V$, X and Y are d-separated by Z in G_1 if and only if X and Y are also d-separated by Z in G_2:*

$$Dsep_{G_1}(X,Y \mid Z) \Leftrightarrow Dsep_{G_2}(X,Y \mid Z).$$

Based on Theorem 10, two Markov equivalent DAGs entail the same set of conditional independence relations. Also, when given the same observational data, it is possible that there exist multiple equivalent DAGs that are equally likely to have generated the same observations. With the absence of any external information, this equivalence bounds our inference ability to learning the set of equivalent DAGs as opposed to learning a single causal DAG. In spite of this limitation, the structural characteristics shared by equivalent DAGs (Theorem 12), can still be very informative about the underlying causal relations.

Theorem 12 (Verma and Pearl, 1990, Equivalence Class of DAGs) *Two DAGs $G^{(1)}$ and $G^{(2)}$ are equivalent if and only if they have the same skeleton and contain the same set of v-structures.*

Typically, the class of equivalent DAGs is represented by the completed partially directed acyclic graph (CPDAG). A partially directed acyclic graph (PDAG) is a graph where some edges are directed and some are undirected. A PDAG is said to be complete if (1) every directed edge exists also in every DAG in the equivalence class of the DAG and (2) for every undirected edge $i - j$, there exists a DAG with $i \rightarrow j$ and a DAG with $i \leftarrow j$ in the same equivalence class.

4. Local Constraint-Based Algorithms

Several constraint-based methods were developed to recover the skeleton of BNs (Pearl, 1988; Spirtes et al., 2001; Cheng et al., 2002; Tsamardinos et al., 2006) and all these methods share common properties in that, a local search is typically performed to identify possible marginal or conditional independence between pairs of variables using a statistical test such as the G-squared test or the partial correlation test (Neapolitan, 2004). In this paper, we restrict the discussion and the comparison to two representative methods: the PC and MaxMin algorithms, which are two of the most popular methods. A detailed comparison among several methods, including the PC and MaxMin algorithms, can be found in the work of Tsamardinos et al. (2006).

The PC algorithm (Spirtes et al., 2001) starts with a fully connected graph where unnecessary edges get iteratively deleted one at a time. For every node V_i, the conditional independences are tested along the existing edges by conditioning on all subsets of the current neighbors and edges are deleted whenever dependences are found to be insignificant. The extended version of the PC algorithm uses the recovered skeleton to recover the CPDAG of the equivalence class by identifying v-structures and using an additional set of CPDAG orientation rules (Meek, 1995).

In contrast to the PC algorithm, the MaxMin algorithm (Tsamardinos et al., 2006) starts with an empty graph. Afterward, for every node in the graph, the algorithm performs a forward selection of neighbors followed by a backward elimination regardless of which is a parent and which is a child. In either phase, the algorithm tests for independence by conditioning on all subsets of recognized neighbors. In addition, a post-processing step is performed in which edges are deleted if the dependence between two nodes does not appear to be significant from both sides simultaneously. The extended version of the algorithm, MaxMin hill climbing (MMHC), is a hybrid algorithm that uses a score-based search constrained by the recovered skeleton to recover the CPDAG of the equivalence class.

5. Difficulties when Learning from Small Sample Data

Using independence testing statistics to learn a BN from small sample data gives rise to a number of issues that can deteriorate the accuracy of both the recovery of the skeleton and the orientation of the edges. In this paper, we will focus on the PC and the MaxMin algorithms as case studies. We note that these issues are a consequence of sampling and do not contradict that these algorithms were proven to recover the correct network in the asymptotic limit.

5.1 Unused Conditional Independence Testing Information

In local constraint-based learning algorithms, an edge E_{xy} is usually excluded from the skeleton if either node, x or y, finds at least one subset of their neighboring variables to induce complete conditional independence between x and y. This approach is analogous to searching for two subsets of variables $Z_x \subseteq CP_x$ and $Z_y \subseteq CP_y$ that induce conditional independence between x and y with the highest statistical confidence based on the observational data. If the maximum of the confidence about conditional independence on either Z_x or Z_y is found to be greater than a threshold α, E_{xy} gets excluded from the skeleton and included otherwise. Although this approach is sufficient to recover the correct skeleton in the infinite sample case, its use in learning from limited sample data ignores information about how probable we are to be correct in rejecting the conditional independence hypothesis with the lower confidence. For example, when an edge E_{xy} is being evaluated using two tests of conditional independence with p-values of 0.04 and 0.03, we are more likely to be incorrect to include E_{xy} in the skeleton than if the p-values of the two tests were 0.04 and 0.01. Though we are equally likely to be correct in rejecting the first null hypothesis of conditional independence in both cases, we are more likely to be correct in rejecting the second conditional independence hypothesis in the second case. Therefore, to improve the accuracy of constraint-based learning, candidate edges should be ranked based on a joint confidence criterion that combines the outcome of conditional independence tests at both sides of the edge simultaneously as complementary sources of information.

5.2 Increased Type II Error in Dense Regions of the Graph: $P(reject\,E_{xy}|true\,E_{xy})$

The type II error in this context refers to the main null hypothesis that the edge does not exist ($\bar{H}_0 : E_{xy} \notin Skeleton$). In constraint based learning, an edge E_{xy} is typically rejected if at least one conditional independence hypothesis gets accepted when conditioning on all subsets of neighbors (CP_x and CP_y). However, as the number of recovered connections to a certain node x increases, the probability of incorrectly inferring conditional independence between x and other variables tends to increase. There are two reasons for this behavior:

1. *Multiple Testing*: In order to recover a correct edge E_{xy}, the number of conditional independence tests that must be correctly rejected grows fast with the number of current neighbors (CP_x or CP_y). However, due to limited training samples, the probability of incorrectly accepting an individual independence hypothesis is greater than zero. Therefore, the chance of rejecting a correct edge increases as the number of independence tests increases (Tsamardinos and Brown, 2008).

2. *Vanishing Dependence Coefficients*: Conditioning on a larger set of parent and/or child variables of a variable x can, in many cases, lead to smaller dependence coefficients with other parent and child variables (see supplementary material for a poof of multiple cases). For example, conditioning on a larger set of child variables of x can result in smaller partial correlations with its parent or other child variables. This in turn increases the chance of incorrectly deciding that the true partial correlation is zero and hence incorrectly rejecting the edge when learning from small sample data.

The increased type II error is a consequence of the fact that connectivity density varies across the network where some nodes are connected to more neighbors than others. Therefore, a fixed threshold for accepting or rejecting the individual null hypothesis of conditional independence does not take this into account.

5.3 Conflicting Results in the Recovery of the CPDAG

The constraint-based method for identifying the direction of BN edges relies on the identification of v-structures based on the concept of separation sets (Meek, 1995; Spirtes et al., 2001). Using this method, it is possible to recover two or more conflicting v-structures (i.e., $x \rightarrow w \leftarrow y$ and $w \rightarrow y \leftarrow u$). The PC algorithm which is the most widely used constraint-based CPDAG recovery algorithm does not offer any resolution for such conflicts. Also, it was not until recently that an algorithm, called Edge-Opt (Fast, 2010), was proposed to resolve conflicting CPDAG results. Edge-Opt resolves conflicts by performing a heuristic search for a DAG that maximizes the number of satisfied d-separation constraints as implied by the observed data in addition to performing a tie-breaking based on a score criterion. However, Edge-Opt considers all d-separation constraints equally significant which, for limited sample data, does not take into account that marginal and conditional independences are inferred from the data with different confidence levels.

In this paper, we propose a fundamentally different approach than Edge-Opt, where we take advantage of the confidences at which the constraints are inferred to rank v-structures and to resolve conflicts. The proposed approach, in addition to being easy to implement, does not require a heuristic search or any scoring criterion of the global network.

6. Methods: Light Mutual Min Algorithm (LMM)

In the proposed algorithm, we attempt to address the above issues related to skeleton recovery using two main techniques. First, candidate edges are ranked using a new measure that combines independence tests when conditioning on all subsets of neighboring variables of the first and second node simultaneously. The proposed ranking criterion is an estimate of the joint conditional posterior probability that the dependence between the corresponding two variables cannot be explained away by subsets of neighboring variables of either the first or the second variable. Second, to ease the multiple testing problem, a new method is presented to relax independence constraints in dense areas of the graph. Moreover, to address the conflicting results issue in orienting edges, we propose a new criterion to rank all candidate v-structures. This ranking offers a method to simultaneously identify v-structures and resolve conflicts. All methods presented are illustrated for the multivariate Gaussian case. However, the same approach can also be extended to other cases with necessary modifications.

6.1 Joint Criterion of Conditional Dependence for All Conditioning Sets

In the multivariate Gaussian case, two variables i and j are considered independent when conditioning on the set $Z_{\setminus i,j}$ if and only if they have a zero partial correlation ($\rho_{ij|Z} = 0 \Leftrightarrow i \perp\!\!\!\perp j|Z$). Since the true partial correlation is unknown, a statistical test is typically used to test whether the sample partial correlation $\hat{\rho}_{ij|Z}$ is significant or not (Neapolitan, 2004). In constraint-based learning, an edge E_{ij} is rejected from the graph if and only if at least one independence test is accepted when conditioning on all subsets of the neighbor sets CP_i or CP_j separately. In contrast, in our method, we suggest using a joint dependence criterion that combines independence tests from both sides of an edge simultaneously. To do so, we first follow the approach of Schäfer and Strimmer (2005) to compute an estimate of the posterior probability that the true partial correlation is not zero when given the sample partial correlation $\hat{\rho}_{ij|Z}$.

From Hotelling (1953), when the true partial correlation is zero ($H_0 : \rho = 0$), the sample partial correlation has the following null distribution:

$$f_0(\rho) = (1-\rho^2)^{(\kappa-3)/2} \frac{\Gamma(\kappa/2)}{\pi^{1/2}\Gamma[(\kappa-1)/2]}. \tag{1}$$

In (1), Γ is a gamma function with κ degrees of freedom, which in this case should be set to $(N - |Z| - 1)$ where N is the number of samples and $|Z|$ is the size of the conditioning set. In contrast, for the alternative hypothesis ($H_A : \rho \neq 0$), ρ can have any value in the range [-1,1], and unless we possess prior information about its distribution, for simplicity, we assume it follows a uniform distribution ($f_A(\rho) = 0.5,\ \forall \rho \in [-1,1]$ and 0 otherwise).

Using the sample partial correlation and the distributions of ρ under both the null and the alternative hypothesis, the posterior probability of the true partial correlation being non-zero can be computed as:

$$P(\rho_{ij|Z} \neq 0 \mid \hat{\rho}_{ij|Z}) = \frac{\pi_A \times f_A(\hat{\rho}_{ij|Z})}{\pi_A \times f_A(\hat{\rho}_{ij|Z}) + \pi_0 \times f_0(\hat{\rho}_{ij|Z})}. \tag{2}$$

In (2), π_0 and π_A are the prior probabilities of the null and the alternative hypothesis respectively where $\pi_0 + \pi_A = 1$. although these priors are generally not known *a priori*, they can be approximated by the user or empirically estimated using likelihood maximization (see Section 6.1.1 for details).

Assuming the faithfulness property, when given the correct neighbor set of a variable i (CP_i^*), the dependence of i on another variable j, with a true neighbor set (CP_j^*) that is neither a child nor a parent of i, can sufficiently be explained away by at least one subset of either CP_i^* or CP_j^*. Therefore, when learning from limited observation data D and CP_i is our current best estimate of the neighbor set of i, an estimate of a one-sided conditional posterior probability that the edge E_{ij} is part of the correct graph can be computed as follows:

$$P(E_{ij}|D)_{[CP_i]} = \min_{Z \subseteq CP_i,\, j \notin Z} P(\rho_{ij|Z} \neq 0 \mid \hat{\rho}_{ij|Z}). \tag{3}$$

Similarly, given the estimated neighbor set of j (CP_j), one can also compute the one-sided conditional posterior $P(E_{ij}|D)_{[CP_j]}$. Here, we refer to $P(E_{ij}|D)_{[CP_i]}$ as one-sided because it ignores the information about independence tests when conditioning on subsets of the neighbor set of the other variable. We also call it conditional because we restrict the conditioning tests to subsets of CP_i as if it contains all child and parent variables of i.

In order to take advantage of the mutual dependence between a parent and a child variable and the asymptotic property that this dependence cannot be explained away by any set of other variables, in the proposed approach, we rank edges using an estimate of the joint conditional posterior probability that none of the subsets of either of the two neighbor sets can make the two variables conditionally independent as follows:

$$P(E_{ij}|D)_{[CP_i,CP_j]} = P(E_{ij}|D)_{[CP_i]} \times P(E_{ij}|D)_{[CP_j]}. \tag{4}$$

Equation (4) can be interpreted as an estimate of the joint conditional posterior probability that the minimal partial correlations from conditioning on subsets of neighboring variables of i and j separately are both significant. In Section A of the Appendix, ranking edges using Equation (4) is shown to be sufficient for recovering the correct skeleton in the asymptotic limit. For the rest of this paper, we use the acronyms one-sided CPPD and joint CPPD to refer to the quantities measured by Equations (3) and (4) respectively, where CPPD is short for conditional posterior probability of dependence (for all conditioning sets) between the two corresponding variables.

What distinguishes ranking edges using Equation (4) is that it combines the two one-sided CPPDs to make one decision about each edge. In contrast, other constraint-based methods such as the PC and MaxMin algorithms reject the edge if a single subset of either CP_i or CP_j renders i and j conditionally independent. This is equivalent to ranking edges using the minimum of the two one-sided conditional posteriors in the right hand side of Equation (4) and ignoring how larger than the minimum the other one-sided conditional posterior was in comparison. While either the minimum or the product can work perfectly when given a very large number of samples, combining two sources of information about the edge as in Equation (4) is anticipated to improve the estimation of the posterior that the edge is part of the correct graph in limited sample problems. In this work, we empirically compare the two ranking criterion and show that the proposed criterion provides a consistent and significant improvement in the accuracy of skeleton recovery.

However, we should note that, for simplification, the factorization in the right hand side in (4) does ignore a possible correlation between the minimal partial correlations from both sides of the edge and it also ignores the possibility that CP_i and CP_j might be overlapping. In Section B of the Appendix, we elaborate on the effect of overlapping neighbor sets on ranking edges and use examples to illustrate why the overlap of neighbor sets is not likely to have a significant negative effect on the accuracy of ranking edges.

Though all methods presented in this paper are restricted to the multivariate normal case, the proposed ranking criterion (Equation 4) can be extended to incorporate other types of independence testing statistics. For example, the method proposed by Margaritis and Thrun (2001) for computing a posterior of conditional independence for the non-linear case, based on multi-resolution discretization, can be used in Equation (2) to compute the posterior of conditional dependence. Other conditional independence testing methods (i.e., G2 statistic, Neapolitan, 2004) can also be used by incorporating the correct null and alternative distributions. These extensions, however, require further research and empirical analysis that goes beyond the scope of the current work.

6.1.1 ESTIMATING PRIORS

Although the priors π_0 and π_A in Equation (2) are not known *a priori*, one can anticipate that BN learning is typically applied to sparse networks ($\pi_0 >> \pi_A$). In addition, likelihood maximization can be used to empirically estimate these priors. For example, in the case of zero order partial correlations (conditioning on the empty set: $\emptyset$), π_A can be selected to maximize the log of the likelihood of the sample partial correlations generated from a mixture of the null and the alternative distributions as follows:

$$\hat{\pi}_A = arg \max_{0<\pi_A<1} log \prod_{i,j} [\pi_A \times f_A(\hat{\rho}_{ij}) + (1-\pi_A) \times f_0(\hat{\rho}_{ij})]. \tag{5}$$

Although π_A and π_0 are expected to vary between the zero order and higher order partial correlations, the $\hat{\pi}_A$ estimated in (5) can be considered as an upper bound for the alternative prior for the purpose of computing Equation (2). This is because as we take the least significant of all partial correlations when conditioning on more than one set, including the empty set, dependence becomes more likely to be explained away. Therefore, in all experiments reported in this paper, as a heuristic and for simplicity, using the $\hat{\pi}_A$ estimated in (5), we set π_A and π_0 in Equation (2) to $\hat{\pi}_A/2$ and $(1-\hat{\pi}_A/2)$ respectively for all partial correlations. To maximize (5), we used a line search where π_A was restricted to [0.001, 0.2].

6.2 Adaptive Reduction of Independence Testing

While in theory, a locally adaptive threshold can resolve the increased type II error problem described in Section 5.2, estimating an accurately variable threshold is a nontrivial problem. As an alternative, we employ a heuristic procedure to relax independence testing across the network, such that dense regions in the graph get the most relaxation.

When inferring a skeleton from small sample data, pairs of nodes are expected to show asymmetric one-sided CPPD (Equation 3). This is because every node in the graph has its own neighbor set, and parameter estimation from small sample data is not perfect. In addition, due to the multiple testing problem, nodes with many recognized neighbors will tend to show smaller CPPD with other variables. The proposed approach takes advantage of this asymmetry to identify which node of every pair to be connected might be suffering from multiple testing. Afterward, candidate edges connecting to the identified node become candidates for relaxation of constraints. This relaxation is performed whenever a new edge E_{ij} is added to the skeleton by selectively updating the neighbor sets of i and j using the following rules:

1. Both neighbor sets of i and j are updated ($j \in CP_i$, $i \in CP_j$) if the one-sided CPPD between i and j is symmetric or almost symmetric ($|P(E_{ij}|D)_{[CP_i]} - P(E_{ij}|D)_{[CP_j]}| < \omega$, for small and constant $\omega \in (0,1]$).

2. Only the neighbor set of i is updated ($j \in CP_i$, $i \notin CP_j$) if i has the higher one-sided CPPD ($P(E_{ij}|D)_{[CP_i]} > P(E_{ij}|D)_{[CP_j]} + \omega$).

3. Only the neighbor set of j is updated ($j \notin CP_i$, $i \in CP_j$) if j has the higher one-sided CPPD ($P(E_{ij}|D)_{[CP_j]} > P(E_{ij}|D)_{[CP_i]} + \omega$).

In this paper, when a node i is added to the neighbor set of a node j, j is said to be aware of the connection E_{ij}. The process of occasionally making only one of a connected pair of nodes aware of a recovered edge will result in a reduction in the number of tests used to evaluate edges connecting to the other node.

In the given rules, ω is a threshold to decide whether dependence is significantly asymmetric between two nodes. In the limited sample case, when $\omega = 1$, the selective reduction of independence testing will not be performed, whereas when ω is set to a small positive value ($\omega \in (0,1)$), the reduction of independence testing will be applied whenever the difference of the one-sided CPPD (Equation 3) is greater than ω. Though any value of ω in the range $(0,1]$ can be used, in this paper, we limit the comparison to two extreme cases ($\omega = 1$ or 10^{-6}). Also, in the Appendix, we provide a proof that the proposed algorithm recovers the correct skeleton in the asymptotic limit for all $\omega > 0$.

Note that our approach relies mainly on selective reduction of independence testing to mitigate the effect of multiple testing. The proposed approach can also be integrated with other heuristics to control for multiple testing. For example, the false discovery rate control methods proposed by Tsamardinos and Brown (2008) or Li and Wang (2009) can potentially be integrated in Equation (3) for even further inference accuracy. This integration however is beyond the scope of the current work.

6.2.1 MOTIVATION FOR THE SELECTIVE REDUCTION OF INDEPENDENCE TESTING

In the limited sample case, the proposed method of selectively updating neighbor sets serves as an adaptive reduction/relaxation of the independence testing where highly connected nodes in the graph get the most relaxation due to their rapidly growing independence from other variables as a result of the increased type II error. As a consequence of this relaxation, the set of neighbors used in conditional independence testing remains small and the multiple testing issue is less likely to contribute to the rejection of true edges. Furthermore, the smaller neighbor sets lead to a dramatic decrease in the number of conditional independence tests and thus much faster learning.

Although making a node not aware of some of its neighbors might enable it to accept a new edge without taking into consideration all its neighbors, leading to a possible false positive identification, this is not expected to be a common behavior due to the following reasons:

1. Due to ranking edges by the product of the one-sided CPPDs from both sides of an edge, for an edge to be incorrectly selected by the algorithm, the dependence has to appear incorrectly significant from both sides simultaneously.

2. Assigning information about edges to the nodes that show higher one-sided CPPD serves to assign neighbor sets of minimal size in a way that provides a maximal mutual information

with the nodes to which they are assigned. For example, if in a new edge E_{ij}, i shows higher one-sided CPPD than j ($P(E_{ij}|D)_{[CP_i]} > P(E_{ij}|D)_{[CP_j]} + \omega$), this means the set CP_j contains variables that partially contain the information that node i provides about j. Therefore, by exclusively assigning the information about the edge to node i, less information is lost than if the information about the edge was exclusively assigned to node j. Also, since node j has low one-sided CPPD on i, it is less likely that node i is needed to block the dependence of j on non-true neighbors.

In addition, as the number of training samples increases, fewer pairs of variables will show asymmetric dependence leading to less relaxation of independence testing. As a result, in the case of a large number of training samples, every node will become aware of all of its neighbors, and hence the algorithm still retains its correctness in recovering the true skeleton in the asymptotic limit (see Appendix for proof). Note that, this approach is heuristic and the algorithm behavior will vary depending on the number of observational samples and the local density of edges in the graph. Section 6.3.1 gives an example that illustrates the behavior of the proposed relaxation of independence testing during skeleton recovery. Moreover, the second part of the supplementary material gives further illustrations with further discussion of the motivation for the proposed approach.

6.3 LMM Algorithm for Skeleton Recovery

The simplified version of LMM (Algorithm 1) has two major phases: forward selection and backward elimination. In the forward selection, the algorithm incrementally adds new edges with the highest joint CPPD (Equation 4) until a certain number of edges are added or a threshold is reached. In the backward elimination phase, the algorithm incrementally eliminates edges with the least joint CPPD. As edges are added to the skeleton, sets of neighbors are updated using the rules stated in Section 6.2.

As the forward selection phase of LMM proceeds, the joint CPPD (Equation 4) along a previously recovered edge E_{ij} might become less significant as more edges are added due to the recovery of new neighbors, such as common parent variables that explain the observed dependence between i and j. Therefore, to improve the efficiency of the algorithm (Aliferis et al., 2010), the recovered edge E_{ij} with the least joint CPPD becomes a candidate for early deletion. Also, as the network grows in the first phase, the node with the higher one-sided CPPD among a pair of connected nodes might change. For example, when an edge E_{ij} is first added, i has a higher one-sided CPPD on j and hence only the neighbor set of i is updated to include j. However, as more edges are added that connect to i, j might become the node with the higher one-sided CPPD and hence, j should be made aware of the edge ($CP_j \leftarrow CP_j \bigcup i$). To address these issues, every three iterations of forward selection, we perform the following three operations:

1. OPT1: Delete the edge E_{ij} with the least joint CPPD if it is less than the joint CPPD of any yet unrecovered edge.

2. OPT2: Set ($CP_j \leftarrow CP_j \cup i$) if E_{ij} is the recovered edge with the highest joint CPPD where j is more or almost equally dependent on i ($P(E_{ij}|D)_{[CP_j]} > P(E_{ij}|D)_{[CP_i]} - \omega$) but it is not aware of the edge ($i \notin CP_j$).

3. OPT3: Set ($CP_j \leftarrow CP_j - i$) if E_{ij} is the recovered edge with the least joint CPPD where j is less dependent on i ($P(E_{ij}|D)_{[CP_j]} < P(E_{ij}|D)_{[CP_i]} - \omega$) but it is aware of the edge ($i \in CP_j$).

Algorithm 1 : LMM_Simplified(D, ω, γ, [$MinSize$], [$MaxSize$])

Input: Data D, Relaxation Parameter $\omega \in (0,1]$, Joint CPPD Threshold $\gamma \in (0,1]$
Input: [Optional]: Maximum # of Edges to Add in Forward Selection: $MaxSize$
Input: [Optional]: Minimum # of Edges to Leave in Backward Elimination: $MinSize$
Output: Skeleton: $\hat{G}_S$

$\hat{G}_S \leftarrow \emptyset$ //$\emptyset$ is the empty set
$CP_i \leftarrow \emptyset, \forall i$
%Phase I:Forward Selection
repeat
  $E_{ij} \leftarrow arg \max_{E_{xy} \notin \hat{G}_S} P(E_{xy}|D)_{[CP_x, CP_y]}$
  $addEdge(i, j)$
until $|\hat{G}_S| = MaxSize$ or $\max_{E_{xy} \notin \hat{G}_S} P(E_{xy}|D)_{[CP_x, CP_y]} < \gamma$
%Phase II:Backward Elimination
repeat
  $E_{ij} \leftarrow arg \min_{E_{xy} \in \hat{G}_S} P(E_{xy}|D)_{[CP_x, CP_y]}$
  $deleteEdge(i, j)$
until $|\hat{G}_S| = MinSize$ or $\min_{E_{xy} \in \hat{G}_S} P(E_{xy}|D)_{[CP_x, CP_y]} > \gamma$
return $\hat{G}_S$

Procedure addEdge(i, j)
$\hat{G}_S \leftarrow \hat{G}_S \cup E_{ij}$
if $P(E_{ij}|D)_{[CP_i]} > P(E_{ij}|D)_{[CP_j]} - \omega$ **then**
  $CP_i \leftarrow CP_i \cup j$
end if
if $P(E_{ij}|D)_{[CP_j]} > P(E_{ij}|D)_{[CP_i]} - \omega$ **then**
  $CP_j \leftarrow CP_j \cup i$
end if
End Procedure

Procedure deleteEdge(i, j)
$\hat{G}_S \leftarrow \hat{G}_S - E_{ij}$
$CP_i \leftarrow CP_i - j$
$CP_j \leftarrow CP_j - i$
End Procedure

Note that the OPT1 operation tries to ensure that at every stage of the forward selection, the recovered edges are the edges that thus far showed the most significant mutual dependence that has not yet been explained away by any of the recovered relations. On the other hand, OPT2 and OPT3 serve to ensure that the selective reduction of independence testing complies with the rules stated in Section 6.2. The selection of the edge with the highest joint CPPD in OPT2 and the edge with the least joint CPPD in OPT3 is consistent with creating neighbor sets of maximum mutual information

with the nodes to which they are assigned. Algorithm 2 details the changes to the forward selection of the simplified LMM (Algorithm 1: Lines 5-6).

Algorithm 2 : LMM(D, ω, γ)

% Update Algorithm 1 by adding the following lines between Lines 5 and 6.
5: ....
if iterationNumber is multiple of 3 **then**
% OPT1
$E_{ij} \leftarrow arg \min_{E_{xy} \in \hat{G}_S} P(E_{xy} \mid D)_{[CP_x, CP_y]}$
if $P(E_{ij} \mid D)_{[CP_i, CP_j]} < \max_{E_{xy} \notin \hat{G}_S} P(E_{xy} \mid D)_{[CP_x, CP_y]}$ **then**
deleteEdge(i, j)
end if
% OPT2
$E_Q \leftarrow \{E_{ij} : E_{ij} \in \hat{G}_S, i \notin CP_j, j \in CP_i, P(E_{ij} \mid D)_{[CP_j]} > P(E_{ij} \mid D)_{[CP_i]} - \omega\}$
if $E_Q \neq \emptyset$ **then**
$E_{ij} \leftarrow arg \max_{E_{xy} \in E_Q} P(E_{xy} \mid D)_{[CP_y]}$
$CP_j \leftarrow CP_j \cup i$
end if
% OPT3
$E_Q \leftarrow \{E_{ij} : E_{ij} \in \hat{G}_S, i \in CP_j, j \in CP_i, P(E_{ij} \mid D)_{[CP_j]} < P(E_{ij} \mid D)_{[CP_i]} - \omega\}$
if $E_Q \neq \emptyset$ **then**
$E_{ij} \leftarrow arg \min_{E_{xy} \in E_Q} P(E_{xy} \mid D)_{[CP_y]}$
$CP_j \leftarrow CP_j - i$
end if
end if
6: ....

Note, LMM algorithm is presented here in high level language. Details on how we implemented and optimized LMM are presented in the supplementary material and also can be seen in the source code which we are making available with this publication. In the case of applying OPT1-3, to avoid closed loops of adding and deleting the same edges or updating the same sets of neighbors, in our implementation, edges added in the last ten iterations are not considered for either of OPT1, OPT2, or OPT3 operations.

6.3.1 SIMULATED EXAMPLE

To illustrate the behavior of LMM, we simulated the network in Figure 1, and generated 1,000 samples as described in Section 7.1 below. We used a relatively large number of observations to ease the replication of our results and to test whether or how well LMM can recover the correct network given the proposed method of reducing independence testing when given a large set of samples.

Figure 2 (a) shows a network of 13 edges recovered by LMM forward selection. A solid connection point (circle) indicates that the node is aware of the connection. For example, in Figure 2 (a), node F is connected to all of H, K, and J, but it is only aware of its neighbor J: $CP_F = \{J\}$. Such

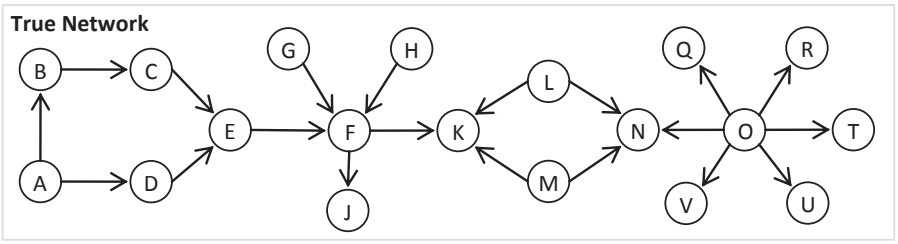

Figure 1: A simulated network of 19 nodes and 20 edges

a result is expected, since once a node starts conditioning on child variables it can develop smaller partial correlations with its parent variables (see Supplement). On the other hand, node J has only one parent and therefore, it will always be aware of it as a neighbor and hence its dependence on grandparent variables (nodes E, G, and H) or sibling variables (node K) will always be successfully blocked.

Figure 2 (b) shows a network of 25 edges recovered by continuing the forward selection for the same network and the same run of LMM. False edges are represented by red dotted lines. In this case, although the resulting connectivity information is not complete (distributed), it is sufficient to block candidate false edges. For example, all child variables of node O are aware of O as a neighbor and therefore, their mutual dependence is blocked (e.g., $P(E_{QR}|D)_{[CP_Q]} \cong 0$). On the other hand, node O is only aware of three of its child variables and that does not have any effect of increasing the chances of recovering false edges among its child variables. Also, note that all the false edges recovered in Figure 2 (b) were only selected due to forcing LMM to recover more than 20 edges and the dependence along these edges is already blocked (e.g., $P(E_{JG}|D)_{[CP_J]} \cong 0$) and they will be eliminated in the backward elimination phase as shown in Figure 2 (c).

The sub-network of node O and its child variables in Figure 2 is a good example of why LMM can outperform other algorithms such as the PC or MaxMin in limited sample problems. For example, in PC algorithm, to correctly recover the edge E_{ON}, all independence tests of conditioning on all subsets of the other 5 neighbors must be rejected while in LMM, node O was aware of three neighbors only and therefore the chances of rejecting the edge due to multiple testing was reduced without increasing the risk of false positive edges among the child variables of O. However, this is still a heuristic approach, and it is possible that in more complex examples the dependence along a false edge is only partially blocked (see the supplementary material for more examples and illustrations).

6.3.2 RANKING OF SKELETON EDGES

The LMM algorithm for skeleton recovery as given in Algorithm 1 can be used to recover skeletons in any of the following three approaches:

1. By using a threshold value ($\gamma \in (0,1]$) for the joint CPPD to decide which edges to be accepted or rejected.

2. By specifying the number of edges to be added in the forward selection and the number of edges to remain after backward elimination ($MaxSize > MinSize > 0$, and $\gamma = 0$).

3. By allowing the algorithm to add many edges in the forward selection (large $MaxSize$, and $\gamma = 0$) and letting the backward elimination eliminate all of them one at a time ($MinSize = 0$).

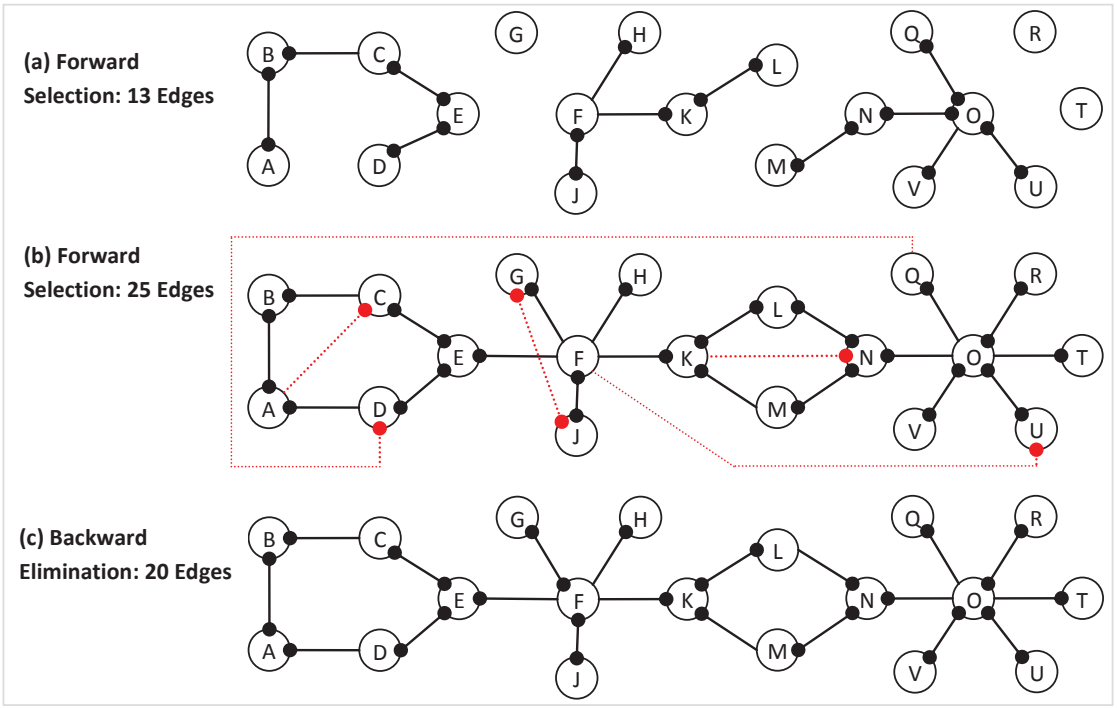

Figure 2: LMM output during forward selection and backward elimination of the same algorithm run at iterations where a) 13 edges have been added in forward selection, b) 25 edges have been added in forward selection, and c) 20 edges have remained in the skeleton after backward elimination has started. Red dotted lines represent false edges. A solid connection point (filled circle) at the end of an edge indicates the node is aware of its neighbor at the other side of the edge.

The order at which edges are deleted in the backward elimination is then used to rank edges where the last deleted edges become the most significant.

Using a threshold for the joint CPPD (Approach 1) makes running LMM similar to running the PC and MaxMin algorithms where a threshold for independence testing is also needed. However, providing the optimal value for any of these thresholds is not trivial for users in limited sample problems. Running LMM by specifying the number of edges to be added and deleted (Approach 2) can be an ideal alternative in cases where the users have a rough estimate of the correct number of edges. For example, if the user thinks the correct network contains about a 1,000 edges, the user can set *MinSize* to 1,000 and *MaxSize* to a slightly larger number (e.g., 1,200). Nonetheless, using LMM to provide a rank of edges (Approach 3) instead of recovering a single skeleton can be more useful, in that, it relieves users from providing a threshold or guessing the number of edges ahead of time. In addition, the rank of edges has a simple interpretation, where the top edges have the highest confidence of being correct while the later edges have less confidence of being correct. The justification for this ranking procedure is that at every iteration of the backward elimination,

the edge with the least joint CPPD is deleted, and thus, the remaining edges are considered to have higher evidence of direct dependence (higher joint CPPD) and are thus more likely to be correct.

To our knowledge, this flexibility property of this last approach has no match in the algorithmic procedure of either the PC or the MaxMin algorithms where one needs to re-run the algorithm many times using different thresholds for independence testing to be able to rank edges. In addition, our approach can be used as a practical alternative to re-sampling and model averaging methods that are typically used to rank edges (Neapolitan, 2004). However, we should note that, ranking skeleton edges is not new to BN learning. For example, Tsamardinos and Brown (2008) ranked edges based on their p-values to control for false discovery rate in skeleton recovery. A similar approach for ranking edges was also proposed by Armen (2011).

In all experiments reported, the skeletons produced by LMM were generated using the proposed ranking procedure (Approach 3). The only exception of this configuration was the experiments of computational complexity comparison where we used a threshold parameter (Approach 1: $\gamma \in (0,1]$) for the joint CPPD in order to make the computational complexity comparison as fair as possible. Also, note that, one can also use a hybrid approach by using a non-zero γ in Approaches 2 and 3 above. This later case, however, is not used in any of the presented empirical results.

6.4 Complexity Analysis of Skeleton Recovery

The computational complexity of constraint-based methods is a measure of the number of independence tests needed to recover the skeleton. However, approximating the exact number of tests needed to execute LMM or other algorithms is a non-trivial problem. Alternatively and similar to other authors (Kalisch and Bühlmann, 2007; Tsamardinos et al., 2006), we use the worst case as an upper bound on the expected complexity.

In the worst case, the three algorithms, PC, MaxMin, and LMM, have a computational complexity of $O(p^2 \times 2^q)$ where p is the number of nodes and q is the maximum size of all neighbor sets in the recovered network ($q = \max_i |CP_i|$). This is because the worst case assumes all nodes have the same maximum connectivity, and therefore we need to perform $O(p^2 \times 2^q)$ independence tests to evaluate candidate edges. However, when using LMM with limited sample data, the proposed adaptive reduction method of independence testing discussed in Section 6.2 (Algorithm 1: Lines 12-18) forces the algorithm to avoid using some neighbors in conditional independence tests. Therefore, although the node with the maximum connectivity is connected to q edges in the true DAG, during the skeleton recovery, it will only be aware of a fraction of them, leading to a reduced number of independence tests (i.e., $O(p^2 \times 2^{q/\delta}) \, s.t. \, \delta \geq 1$). Unfortunately, estimating this fraction before running the algorithm is not possible. Therefore, in this paper, we mainly rely on the empirical evaluation to compare the time complexity of LMM to the other two algorithms.

One of the popular techniques to speed constraint-based learning is by limiting the size of the conditioning sets. This method can reduce computational complexity and it can be used with the PC, MaxMin or LMM algorithms. In the supplementary material, we elaborate on how we implemented LMM and on the optimization techniques we used to avoid redundant computations while making the search for the next edge to add or delete computationally efficient.

6.5 Extending the Skeleton to the Equivalence Class

Learning a directed graph from observational data is typically restricted to learning the completed partially directed acyclic graph (CPDAG). This restriction is motivated by the equivalence nature

of some Bayesian networks, in that, multiple networks with partially different orientations of edges can produce the same observational data. As a consequence, learning the complete directed graph from observational data alone can be impossible in many cases (Meek, 1995; Spirtes et al., 2001).

Once a skeleton is recovered, different approaches can be used to extend it to a CPDAG. Examples of these methods are the score-based constrained hill climbing search algorithm (Tsamardinos et al., 2006) and the constraint-based approaches such as the PC algorithm (Meek, 1995; Spirtes et al., 2001). We stress that our contribution in recovering the skeleton can be used with any other algorithm to recover the CPDAG. However, in this work, we focus on constraint-based learning due to its easy implementation and its computational scalability. In addition, we improve upon the classical constraint-based inference by presenting a new method for resolving conflicting v-structures.

Recovering CPDAGs using constraint-based inference relies on the recovery of the v-structures (see definitions) which are proven to be common among all DAGs of the same equivalence class. From properties of v-structure relations (see Lemma 13 below), a common child variable w in a v-structure relation ($x \to w \leftarrow y$) induces a conditional dependence between the parent variables x and y. Moreover, since x and y are not adjacent in the skeleton, w cannot belong to any set $Z_{\setminus x,y}$ that makes x and y conditionally independent. This property is the basic idea in constraint-based methods (Spirtes et al., 2001). However, this approach can produce inconsistent results. For example, in a case of limited training samples, it is possible to recover multiple conflicting v-structures.

To improve the accuracy of constraint-based inference of CPDAG, we present a new approach for ranking v-structures that can be used to resolve conflicts. The proposed method is derived from the following Lemma (see Neapolitan, 2004, Chapter 2):

Lemma 13 *Suppose we have a DAG $G = (V,E)$ and an uncoupled meeting $x - w - y$. Then the following are equivalent:*

1. *$x - w - y$ is a head-to-head meeting: $x \to w \leftarrow y$.*
2. *There exists a set not containing w that d-separates x and y.*
3. *All sets containing w do not d-separate x and y.*

Based on Lemma 13 and using the partial correlation method, when $x \to w \leftarrow y$ is a correct v-structure and we have data with a large sample size n, then both of the following statements are correct (see Appendix for proof):

1. $\lim_{n\to\infty} \left[\min_{Z \subseteq V_{\setminus x,y,w}} P(\rho_{xy|Z} \neq 0 \mid \hat{\rho}_{xy|Z}) \right] = 0.$
2. $\lim_{n\to\infty} \left[\min_{Z \subseteq V_{\setminus x,y}, w \in Z} P(\rho_{xy|Z} \neq 0 \mid \hat{\rho}_{xy|Z}) \right] = 1.$

Although, Lemma 13 was shown to be accurate in true DAGs (Neapolitan, 2004, Chapter 2), it might not always hold when learning DAGs in real applications due to the unguaranteed accuracy of independence testing methods based on small sample data. As a result, it is possible to recover wrong or conflicting results (e.g., conflicting v-structures: $x \to w \leftarrow y$ and $w \to x \leftarrow u$).

In order to identify v-structures in a given skeleton and to resolve conflicts, in the proposed approach, we rank all candidate v-structures based on the confidence that the candidate head of

the v-structure is a common child of the other two variables. To compute this confidence criterion for every candidate v-structure ($x \rightarrow w \leftarrow y$), we compute the joint CPPD (Equation 4) between x and y when w is added to all conditioning sets and the joint CPPD when w is removed from all the conditioning sets. The difference between the two joint CPPDs is then used as a measure of the level of the induced dependence between x and y caused by w. If the joint CPPD is found to increase when w is added to all conditioning sets, w is concluded to be a common child of both x and y. Also, in a case where two conflicting v-structures get accepted, we take advantage of the amount of increase in the joint CPPD to perform tie-breaking and ignore the v-structure with the smaller induced dependence. The justification for this approach follows directly from Lemma 13, in that, w induces dependence between x and y if and only if it is their common child, and in small sample problems, it becomes intuitive to use the increase in the joint CPPD when w is added to all conditioning sets as a measure of the confidence that w is a common child.

Algorithm 3 presents a summary of the proposed constraint-based orientation of a given skeleton. Note that, testing for induced dependence from common neighbors does not have the multiple testing problem. This is because the number of conditioning sets that contain the common neighbor is equal to the number of conditioning sets that do not contain the common neighbor. Therefore, in the orientation algorithm, we use the complete set of neighbors found in the skeleton. To distinguish these types of sets, we use bar notation on top of the set name (i.e., $\overline{CP_x}$).

The set of additional orientation rules in Algorithm 3 are also used in other constraint based learning methods of the CPDAG such as the PC and TPDA algorithms. Meek (1995) provides a detailed discussion of these rules and their correctness for finding the CPDAG.

As mentioned earlier (Section 5.3), to our knowledge, there is only one other algorithm called Edge-Opt (Fast, 2010) that attempts to resolve conflicts in the constraint-based orientation of BN edges. Our approach is fundamentally different from Edge-Opt, in that, Edge-Opt considers all constraints to be equally significant, while our approach takes advantage of the differences in confidence at which the constraints are inferred and uses these differences to rank candidate v-structures. Moreover, in addition to being easy to implement, our approach does not require any heuristic search or computing a score criterion of the global network to perform tie-breaking. However, we should note that our approach does not solve conflicting edges resulting from the additional orientation rules (R1-4). In our implementation, whenever a conflict is found along a certain edge, it is left undirected.

In the experimental validation, we were not able to empirically compare our approach to Edge-Opt because its publicly available implementation does not support the multivariate Gaussian case. Nonetheless, we compare our implementation to a variant of itself where a random tie-break is used to resolve conflicting v-structures and show that the proposed tie-breaking method provides a significant and consistent improvement in the accuracy of CPDAG orientation. We also compare the proposed approach to the PC and MaxMin hill climbing algorithms. Moreover, in the Appendix, we provide a complete proof that the proposed approach recovers the correct CPDAG in the asymptotic limit.

7. Experiments and Comparison

For comparisons, we restrict our experiments to multivariate Gaussian simulated data and the partial correlation method is used for independence testing (or dependence measurement) in all algorithms. All PC algorithm results reported are performed using the publicly available PCAlg tool (Kalisch

Algorithm 3 : LMM_EO(G_S, D)

Input: Skeleton: G_S, Data: D

$Q \leftarrow \emptyset$ //$\emptyset$ is the empty set
for all $\langle x, w, y \rangle$ where $E_{xy} \notin G_S$ and $E_{wy}, E_{wx} \in G_S$ **do**
  $CD_{w+}(x,w,y) \leftarrow \min_{Z \subseteq \overline{CP_x}, w \in Z} P(\rho_{xy|Z} \neq 0 \mid \hat{\rho}_{xy|Z}) \times \min_{Z \subseteq \overline{CP_y}, w \in Z} P(\rho_{xy|Z} \neq 0 \mid \hat{\rho}_{xy|Z})$
  $CD_{w-}(x,w,y) \leftarrow \min_{Z \subseteq \overline{CP_x}, w \notin Z} P(\rho_{xy|Z} \neq 0 \mid \hat{\rho}_{xy|Z}) \times \min_{Z \subseteq \overline{CP_y}, w \notin Z} P(\rho_{xy|Z} \neq 0 \mid \hat{\rho}_{xy|Z})$
  $CD(x,w,y) = CD_{w+}(x,w,y) - CD_{w-}(x,w,y)$
  if $CD(x,w,y) > 0$ **then**
    $Q \leftarrow Q \cup < x, w, y >$
  end if
end for
while Q is not empty **do**
  $< x, w, y > \leftarrow arg \max_{<j,k,l> \in Q} CD(j,k,l)$
  $Q \leftarrow Q - < x, w, y >$
  if the edges $x - w$ and $y - w$ are not oriented **then**
    Orient $x - w$ into $x \rightarrow w$
    Orient $y - w$ into $y \rightarrow w$
  end if
end while

% Apply the Additional Orientation Rules as Follows:

repeat
  Orient $i - j$ into $i \rightarrow j$ whenever any of the following is correct:
  R1: There exists an arrow $k \rightarrow i$ *s.t.* k and j are not adjacent.
  R2: There exists a directed path from i to j (i.e., $i \rightarrow \{\} \rightarrow j$).
  R3: There exist two chains $i - k \rightarrow j$ and $i - l \rightarrow j$.
  R4: There exist two chains $i - k \rightarrow l$ and $k \rightarrow l \rightarrow j$ *s.t.* k and l are adjacent.
until no more orientations are found

et al., 2010). The original MaxMin algorithm tool (Aliferis et al., 2003) only supports the discrete data case. We therefore re-implemented the algorithm to use the partial correlation test to recover the skeleton. For the CPDAG recovery, we used the publicly available toolkit BNlearn (Scutari, 2010) which implements a variant of MaxMin hill climbing that supports the BIC score criterion for continuous data, which we used for all MMHC reported experiments. Also, in all experiments, the hill climbing search was performed with a Tabu search with a list of 100 possible solutions which is the same setting used by the authors of the algorithm (Tsamardinos et al., 2006).

7.1 Simulating Data

For every experiment, the true model is randomly generated (Kalisch and Bühlmann, 2007) as follows:

1. Fix an ordering of variables.

2. Fill the adjacency matrix A with zeros.

3. Randomly fill entries in the lower triangle matrix A with ones by independent realizations of Bernoulli random variables with a success probability s where $0 < s < 1$ (s represents the level of sparseness of the network)

4. Replace each entry with a 1 in the adjacency matrix by independent realizations of a uniform random variable in the range $[-1,-0.1] \cup [0.1,1]$.

These steps will result in an adjacency matrix A whose entries are either zero or in the range [-1,-0.1] or [0.1,1]. Afterward, in the corresponding DAG, if A_{ij} is not zero, then node j is a parent of node i with a coefficient A_{ij}. Using this randomization setting, in the case of p variables, the expected number of neighbors CP_i^* for a node i can be estimated as: $E(|CP_i^*|) = s \times (p-1)$, while the expected number of all edges can be estimated as: $E(|G|) = s \times (p-1) \times p/2$. Therefore, the density of the graph is linearly proportional to s. For the rest of this paper, when the simulated networks are said to contain X edges, this means s was selected so that the expected number of edges was X. However, the exact number of edges in each network will vary slightly due to randomization.

In our empirical analysis, we have generated a single set of observational samples from every simulated network. Once the adjacency matrix of the simulated network is fixed, every observational sample is recursively generated as follows:

$$X^{(1)} = \varepsilon^{(1)} \sim N(0, \sigma_1^2).$$

$$\forall i = 2,3...p : X^{(i)} = \varepsilon^{(i)} + \sum_{j=1}^{i-1} A_{ij} \times X^{(j)} \ , s.t.\ \varepsilon^{(i)} \sim N(0, \sigma_i^2).$$

where $\varepsilon^{(1)}, \varepsilon^{(2)}, ...\varepsilon^{(p)}$ are independent random normal variables representing the marginal noise. In all reported experiments, the variances of these noise variables are randomly sampled from an inverse gamma distribution: $\sigma_i^2 \sim InvGamma(\alpha = 2, \beta = 1)$, $\forall i$.

7.2 Results and Discussion

To evaluate the proposed algorithm we performed multiple experiments with various settings in an effort to cover a wide variety of possible cases. In every case, we simulated multiple networks and carried out a semi-exhaustive evaluation such that, all methods are compared at different levels of recovery where they are used to recover many solutions of varying sizes and the comparison is illustrated at each level. To finish all the results presented, we used 5 workstations with 8 cores each with a total computation time of about 4 weeks. Moreover, for computational reasons and due to the low statistical power when learning from limited sample data, the individual conditioning sets were restricted not to contain more than four variables for all methods. The only exception was the performance evaluation experiment based on large sample data where we let the conditioning sets contain up to six variables.

For comparison, the receiver operating characteristics curve (ROC) is used to evaluate the accuracy of skeleton recovery while the true positive rate (TPR) plots (TPR against number of retrieved edges) are used to evaluate the accuracy of CPDAG inference. Also, in Section C of the Appendix, we provide additional illustrations of CPDAG inference comparisons using the structural Hamming

distance (SHD) metric. Every method was used to recover multiple skeletons or CPDAGs of varying sizes for the same network and the evaluation plots (ROC, TPR, and SHD curves) of every method were then generated for every single network. For compact presentation, we only present the average of these evaluation curves for every set of networks in addition to bar plots of specified cases to demonstrate the variance of inference accuracy among all networks of the same set. In the case of PC and MaxMin, for every network, each algorithm was run many times with different thresholds ($\alpha = 10^{-30}$, 10^{-10}, 10^{-6}, 10^{-4}, 10^{-3}, 0.003, 0.01, 0.02, 0.05, 0.1, 0.2, 0.3, 0.5). We also, in some cases, used additional alpha values greater than 0.5 to recover larger numbers of edges. In contrast, for all experiments except for computational complexity, LMM was run only once for every network using Approach 3 described in Section 6.3.2. In the forward selection, LMM was set to add as many edges as 2.5 times the number of nodes ($MaxSize = 2.5 \times numberofnodes$) and the order at which edges were removed in the backward elimination was used to rank all selected edges ($MinSize = 0$). The choice of 2.5 was mainly to ensure that the number of edges added in the forward selection is large enough to work well for most cases.

7.2.1 SKELETON RECOVERY

In the first set of experiments, we evaluate LMM for recovering skeletons using 900 networks divided into 9 groups simulated as follows:

1. Fixed number of nodes $p = 200$.
2. Three levels of connectivity density: $\sim$100, $\sim$200, and $\sim$400 edges.
3. Three levels of number of observations: 30, 100, and 300 samples.
4. For every configuration (same number of edges and same number of observations), a set of 100 different networks were generated with separate observational data for each.

First, to assess the effect of the new joint dependence criterion and the proposed adaptive reduction of independence testing, we compared LMM to two variants of itself LMM-1 and LMM-2. In all experiments presented, LMM was configured to adaptively reduce the independence testing as described in Section 6.2 by setting ω to very small value ($\omega = 10^{-6}$). In contrast, LMM-1 and LMM-2 were configured not to use the adaptive reduction of independence testing by setting ω to 1. Also, LMM-1 was set to use the proposed joint CPPD criterion (Equation 4) while LMM-2 was set to use the minimum of the one-sided CPPDs from both sides ($\hat{P}(E_{ij} \mid D)_{[CP_i,CP_j]} = min(P(E_{ij} \mid D)_{[CP_i]}, P(E_{ij} \mid D)_{[CP_j]})$) as a mutual dependence criterion which made it the most similar to the PC and MaxMin algorithms in assessing edges.

Figure 3, shows an averaged plot of the false positive rate (FPR) and the true positive rate (TPR) of the three algorithms in recovering the skeletons of four sets of the simulated networks where the number of edges is $\sim$100 and $\sim$400 while the number of observations is 30 and 100 samples. In addition, Figure 4 shows the bar plots of the area under the ROC (auROC) for the three methods for skeleton recovery of 9 networks sets when the FPR is restricted to [0-0.06]. Note that, because these are partial ROC curves, the auROC is normalized by the maximum possible auROC which is 0.06.

Based on both Figures 3 and 4, the proposed joint CPPD criterion (LMM and LMM-1) is shown to consistently improve the accuracy of skeleton recovery as compared to just taking the minimum of

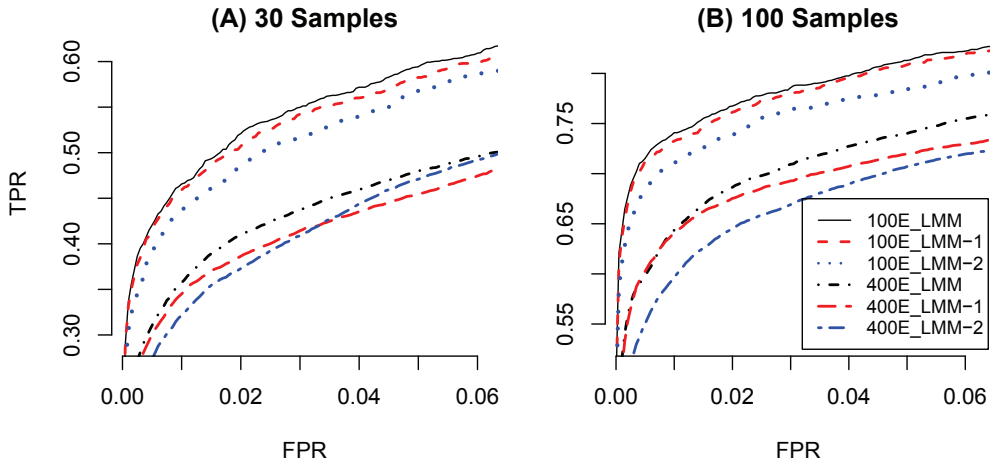

Figure 3: Average ROC of skeleton recovery of LMM, LMM-1, and LMM-2 when the number of true edges is ∼100, and ∼400 and the number of observations is: A) 30 and B) 100 samples. X-Axis is the false positive rate (FPR) and Y-Axis is the average true positive rate (TPR).

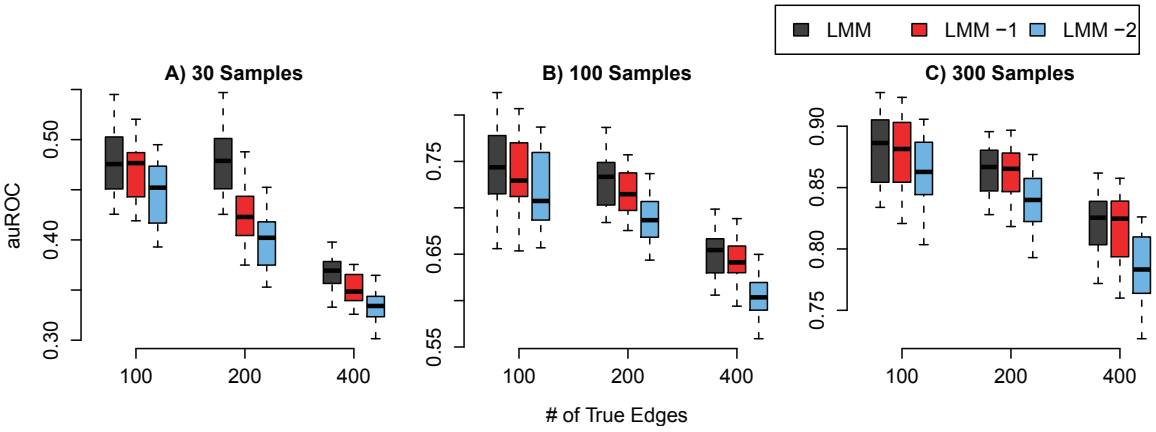

Figure 4: Bar plots of the auROC of skeleton recovery of LMM, LMM-1, and LMM-2 when the number of true edges is ∼100, ∼200, and ∼400 and the number of observations is: A) 30 B) 100 and C) 300 Samples. X-Axis is the correct number of edges in the true networks and Y-Axis is the area under the partial ROC (auROC) where FPR is constrained to $[0, 0.06]$. The plotted auROC is also normalized by the maximum possible area under the partial ROC.

the one-sided CPPDs from both sides of an edge (LMM-2). Also, the proposed adaptive reduction of independence testing (used in LMM only) is shown to improve the accuracy further, especially when the number of observations is small and the true graph is not sparse. This is consistent with multiple testing being a problem when learning from small sample data. Multiple testing is also not expected

to be a significant issue when recovering sparse networks where the variance of connection density is small. We also compared the performance of LMM to the PC and MaxMin algorithms. Figure 5 shows the average partial ROC of the three algorithms in recovering the skeletons of four network sets. Note that FPRs greater than 0.035 are ignored because solutions with more than 0.035 FPR are unlikely to have any practical use and it also is impractical to run the PC and MaxMin algorithms to recover denser skeletons since the number of required independence tests grows exponentially with the size of neighborhood.

In addition, we computed the area under the ROC curve where FPR is bounded to [0-0.035] for every case. Figure 6 shows the bar plots of the auROC for the three methods for the 9 network sets where the auROC is normalized by the maximum possible auROC, which is 0.035.

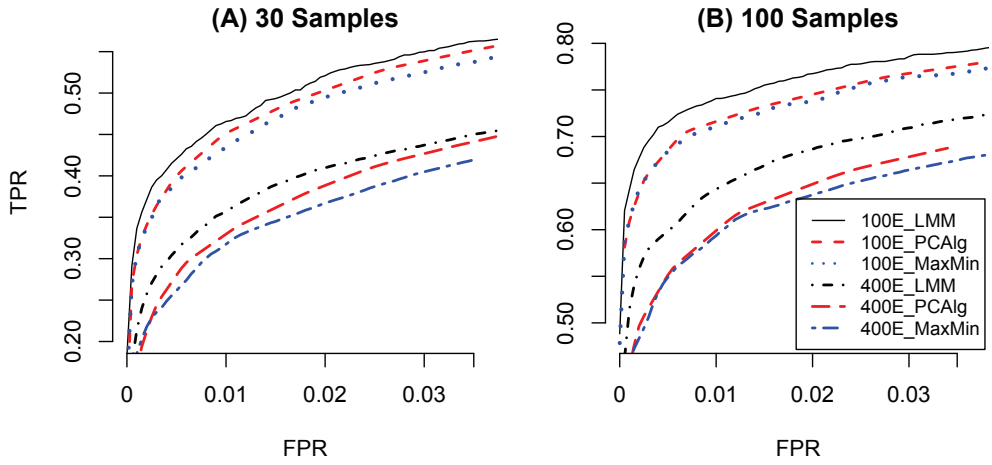

Figure 5: Average ROC of skeleton recovery of LMM, PC, and MaxMin when the number of true edges is ∼100, and ∼400 and the number of observations is: A) 30 and B) 100 samples. X-Axis is the false positive rate (FPR) and Y-Axis is the average true positive rate (TPR).

Based on both Figures 5 and 6, LMM consistently outperforms both algorithms in all cases. Also, similar to first evaluation, the improvement is less significant when the correct graph is generally sparse (i.e., 100 edges), since multiple testing is not an issue in very sparse graphs. The bar plots show the improvement to be consistent and not a result of outliers.

7.2.2 COMPUTATIONAL COMPLEXITY OF SKELETON RECOVERY

To experimentally compare the time complexity of the three algorithms (LMM, PC, and MaxMin) for skeleton recovery, we perform two types of comparisons. The first aims at evaluating the computational complexity when we have a large sample size, where the three algorithms can recover skeletons with high accuracy. The second comparison aims at evaluating the computational complexity when the sample size is limited.

For the first evaluation, we simulated 9 sets of networks where every set contains 40 different networks (see Table 1). Afterward, a set of 10,000 observational samples were generated from every network.

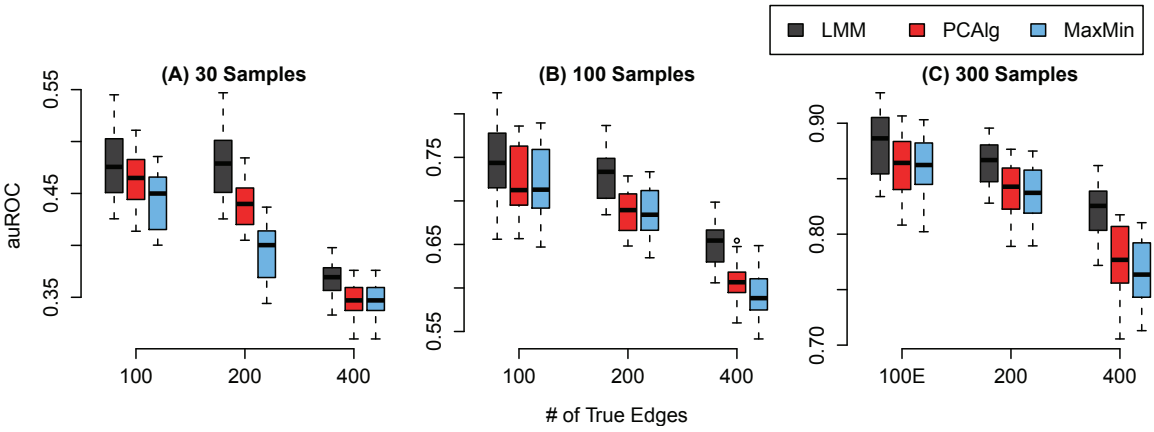

Figure 6: Bar plots of the auROC of skeleton recovery of LMM, PC, and MaxMin when the number of true edges is ∼100, ∼200, and ∼400 and the number of observations is: A) 30 B) 100 and C) 300 samples. X-Axis is the correct number of edges in the true networks and Y-Axis is the area under the partial ROC (auROC) where FPR is constrained to $[0, 0.035]$. The plotted auROC is also normalized by the maximum possible area under the partial ROC.

In order to compare complexity when the three methods recover highly accurate skeletons, we performed a search for the optimal parameters that resulted in the least distance between the recovered skeletons and the true skeletons as measured by structural hamming distance.

In the case of PC and MaxMin, we ran each algorithm to recover every network using different values for the threshold of the independence tests ($\alpha = 10^{-4}, 3\times10^{-4}, 10^{-3}, 3\times10^{-3}, 0.01, 0.03, 0.1$). Similarly, we ran LMM multiple times using Approach 1 described in Section 6.3.2 with different thresholds for the CPPD (Algorithm 1:$\gamma = 0.01, 0.03, 0.1, 0.3$). Afterward, for every method and every network set, the threshold that resulted in the most accurate skeletons was chosen. Note that, here, we define the distance between two skeleton as the number of operations of adding and deleting edges that are needed to transform one skeleton into the other. This distance is a special case of the structural Hamming distance (SHD) between two PDAGs defined by Tsamardinos et al. (2006).

Table 1 shows the average SHD and time in seconds when each method is used with the optimal parameters to recover every set of networks. The table, also, shows the standard deviation of both SHD and time in small font.

Based on Table 1, in terms of accurate recovery, the three algorithms are shown to perform equally well when given a large number of samples. On the other hand, in terms of computational time, both MaxMin and LMM outperform the PC algorithm significantly in all cases. However, LMM seems to be slightly slower than MaxMin especially when the true graph is sparse. It is important to note that the proposed relaxation in LMM only works when the observational sample size is small (see Section 6.2). We should also note that, both LMM and MaxMin used in the experiments presented in this section are implemented in Java while the PC algorithm is implemented in R which is known to be slower than Java. This partially explains why the PC algorithm was the slowest performing algorithm in this experiment.

		Average SHD			Average Time		
Nodes	Edges	PC	MaxMin	LMM	PC	MaxMin	LMM
100	100	**2**$_{\pm 1.2}$	2.4$_{\pm 1.3}$	2.1$_{\pm 1.3}$	3.7$_{\pm 0.9}$	**0.3**$_{\pm 0.1}$	0.8$_{\pm 0.3}$
	200	25$_{\pm 9.4}$	**24**$_{\pm 1}$	26$_{\pm 9}$	32$_{\pm 8}$	4.2$_{\pm 2.7}$	**3.2**$_{\pm 1.1}$
	300	101$_{\pm 22}$	**92**$_{\pm 21}$	95$_{\pm 22}$	107$_{\pm 20}$	20$_{\pm 5}$	**18** $_{\pm 7}$
300	300	6.6$_{\pm 1.8}$	5$_{\pm 1.4}$	**3.4**$_{\pm 0.7}$	24$_{\pm 3.6}$	**1**$_{\pm 0.3}$	3.1$_{\pm 1}$
	600	39$_{\pm 3.5}$	38$_{\pm 11}$	**31**$_{\pm 12}$	121$_{\pm 19}$	15$_{\pm 4.1}$	**14**$_{\pm 4.3}$
	900	188$_{\pm 31}$	**184**$_{\pm 24}$	186$_{\pm 30}$	917$_{\pm 91}$	118$_{\pm 13}$	**113**$_{\pm 27}$
1,000	1,000	19$_{\pm 4}$	30$_{\pm 5.3}$	**15**$_{\pm 3.5}$	254$_{\pm 7}$	**7**$_{\pm 0.5}$	18$_{\pm 0.6}$
	2,000	64$_{\pm 11}$	64$_{\pm 13}$	**58**$_{\pm 12}$	612$_{\pm 59}$	**53**$_{\pm 10}$	76$_{\pm 14}$
	3,000	295$_{\pm 31}$	278$_{\pm 34}$	**269**$_{\pm 30}$	3,235$_{\pm 370}$	**995**$_{\pm 146}$	1,210$_{\pm 201}$

Table 1: Average execution time in seconds and average SHD of each algorithm when recovering skeletons of BNs of given sizes. Best results for each network set are in bold face. Each result is presented as mean $\pm$ standard deviation.

Nonetheless, the superior computational advantage of MaxMin is only valid when the sample size is large. As will be shown in the next experiment, the complexity of MaxMin increases fast when the given sample size is small due to the correction step of asymmetric connectivity being deferred to the last phase of the algorithm.

In order to compare the computational complexity of the three algorithms when the sample size is small, we only consider one set of 40 networks of 300 nodes and $\sim$600 edges each. We generated 100 observational samples from every network and every algorithm is used to recover many solutions of different sizes for every network. We also compare to the unrelaxed version of LMM (LMM-1) where we disable the adaptive reduction of independence testing (Algorithm 1: $\omega = 1$). The goal here is to test how computational complexity can grow as we attempt to retrieve various number of edges. Learning BN from small sample data is approximate in nature. While in many cases users are interested in a solution with the least SHD, in other cases, users may also be interested in various levels of recall (true positive rate), which can only be achieved by retrieving various numbers of edges. Here, we compare the four methods (LMM, LMM-1, PC, and MaxMin) when used to retrieve skeletons of different sizes up to 1.8 times the size of the true skeleton. This is done by running every method on every network many times using different thresholds for the independences tests. Figure 7 shows both the average TPR and the average computation time versus the number of retrieved edges.

Based on Figure 7, LMM retrieves more accurate skeletons in much less time than all other methods including LMM-1. By taking advantage of adaptive relaxation of constraints, LMM can ease the multiple testing problem and at the same time reduce the computational complexity. On the other hand, the MaxMin algorithm computational time grew fast as we tried to retrieve denser solutions. To understand this behavior, we debugged the code of MaxMin as it recovers the skeleton from small sample data and found that this dramatic increase in computation time is due to the correction of asymmetric connectivity step being deferred to the last phase of the algorithm (Tsamardinos et al., 2006). MaxMin works by searching for child and parent variables for every node separately and at the end, for an edge to be selected, both nodes have to appear as neighbors to each other. However, in the case of small sample data, the asymmetric connectivity seems to

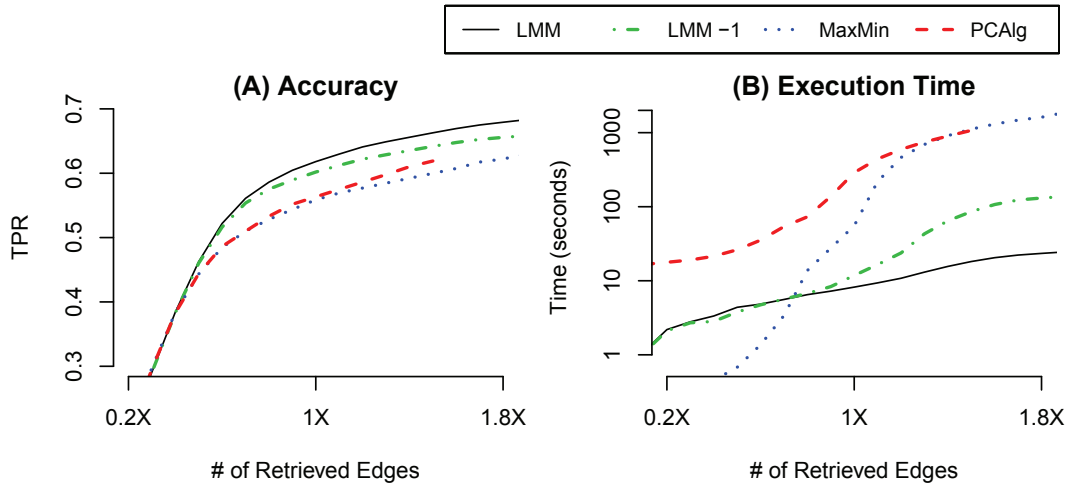

Figure 7: Average TPR (A) and computation time (B) when using LMM, LMM-1, PC, and MaxMin to retrieve varying numbers of edges for 40 networks (each composed of 300 nodes and ∼600 edges) based on limited data (100 samples). X-Axis is the number of retrieved edges normalized by the true number of edges (X = Number of edges in the correct network). Time-Axis is log-scaled.

be common and a lot of edges are rejected leading to a small number of recovered edges, even when every node identified a relatively large number of candidate neighbors. Therefore, to retrieve a larger number of edges, one will need to use a larger α leading to larger neighbor sets in the first phase and hence many more independence tests. A similar finding about the effect of symmetry correction on the computational efficiency of MaxMin has also been reported and discussed by Aliferis et al. (2010). However, unlike the suggestion made by Aliferis et al. (2010) which eliminates the symmetry correction step, all variants of LMM presented in this work avoid the deterioration of computational requirements without sacrificing the asymptotic correctness of the inference by performing early correction of symmetric connectivity. For example, when considering LMM-1, once an edge E_{xy} is added or deleted, both neighbor sets CP_x and CP_y are updated accordingly and immediately. This early joint update of neighbor sets informs nodes not to condition on false neighbors early on as the algorithm proceeds. This explains why LMM-1 is faster than MaxMin for dense solutions, although it does not relax independence testing.

We again note that in addition to this low computational cost, the algorithm offers an ease of use property where users can make the algorithm retrieve a fixed number of edges or a ranked list of edges (see Section 6.3.2).

7.2.3 CPDAG RECOVERY

To evaluate the extended LMM for CPDAG recovery, we compared it to the PC and MMHC algorithms by using each method to recover multiple CPDAGs of different sizes for every single network. Afterward, a plot of the true positive rate (TPR) versus the number of retrieved edges is generated for each network. When the curves are plotted in conjunction with a curve representing a hypothetical optimal recovery, the plot becomes a compact representation where one can read the

TPR, false positives (FP), and false negatives (FN) at every point on the curve. In addition, comparing two methods using the curve of TPR versus the number of retrieved edges has the same semantic as a ROC curve where higher always means better, in that, at a fixed number of retrieved edges (X-Axis), higher TPR also means lowers FP and FN. This makes it a more informative representation of results than the structural Hamming distance plot (SHD) used by other authors (Tsamardinos et al., 2006). In the Appendix, we provide an illustrative example comparing the two metrics and we also provide the SHD plots for all experiments.

First, we used all algorithms to recover CPDAGs for two sets of 100 networks each. The first set is simulated so that every network had ∼100 edges while the second set is simulated so that every network had ∼300 edges. Figure 8 shows the average TPR of the CPDAGs recovered by LMM_EO, LMM_EO-1, MMHC, and PC algorithm for each set separately when given the same number of training samples (100 samples). LMM_EO-1, refers to using LMM_EO with a random tie breaking for conflicting v-structures. The figure shows the average TPR of each algorithm at different levels of recovery (number of retrieved edges) up to 30-60% more edges than the true number of edges. In the case of using MMHC for recovering network of ∼300 edges (Figure 8:B), we were not able to get BNLearn to recover CPDAGs with more than ∼10% edges more than the true CPDAG due to the deletion of edges performed by hill climbing search to maximize the score.

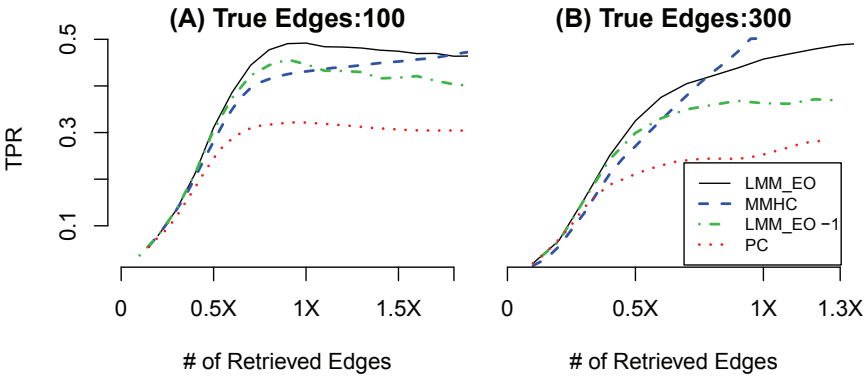

Figure 8: Average TPR when recovering CPDAGs of networks of 100 nodes and A) ∼100 or B) ∼300 true edges when the number of samples is 100. X-Axis is the number of retrieved edges (X = Number of edges in the correct network). Y-Axis is the average true positive rate (TPR).

As illustrated by the comparison of LMM_EO and LMM_EO-1 in Figure 8, the proposed approach for resolving conflicting v-structures consistently improves the accuracy of orienting the edges as compared to random selection among conflicting v-structure. Also LMM is shown to significantly outperform the PC algorithm while being competitive to MMHC in this case. To illustrate the variance of recovery among all recovered networks, Figure 9 shows the bar plots for the TPR when recovering networks containing as many edges as 0.7, 1.0, and 1.3 times the number of true edges in each network. Figure 9 is complementary to Figure 8 where the improvement is shown to be consistent and not a result of outliers.

In the second set of experiments, we used six sets of networks of sizes 100, 500, and 2500 nodes and two different levels of observations: 100 and 300 samples. All networks are simulated so

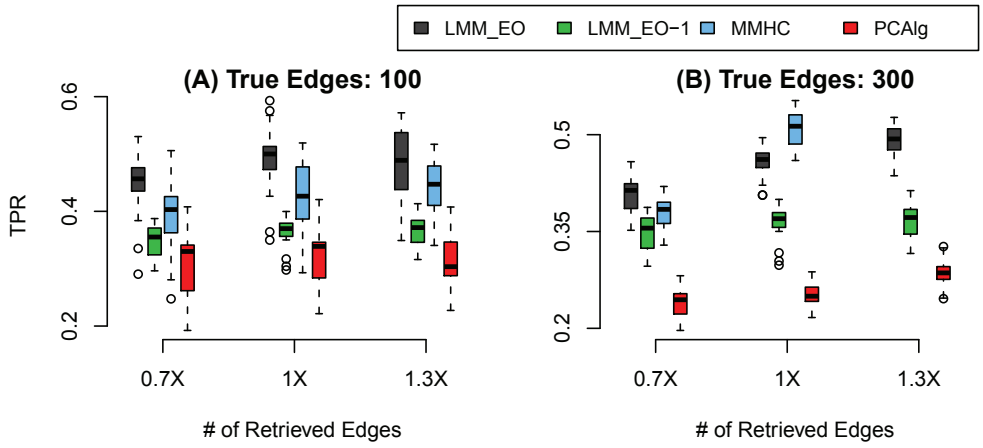

Figure 9: TPR of CPDAGs recovery of networks of 100 nodes and A) ∼100 or B) ∼300 edges. X-Axis is the number of retrieved edges (X = Number of edges in the correct network). Y-Axis is the true positive rate (TPR).

that the number of edges is two times the number of nodes in every network. Figure 10 shows the average TPR for every method for the recovery of the CPDAGs of every set while Figure 11 shows the corresponding bar plots for the TPR when the number of recovered edges is as many as 0.7 and 1.0 times the number of true edges.

Based on Figures 10 and 11, in terms of accurate recovery, LMM scales well for larger networks, where at the same level of density and number of observations, LMM has almost the same level of TPR. In addition, LMM has far more accurate recovery than the PC algorithm for different network sizes. When compared to MMHC, in the 100 nodes case, LMM seems to have comparable accuracy. However, in the 500 and 2,500 variables cases, LMM is more accurate. The deterioration of MMHC could be a result of the local minimum problem, in that, global optimization techniques, such as the Tabu search with hill climbing, become less effective as the search space grows very large and the solution becomes prone to converging to a bad local minimum.

Finally, to empirically test the correctness of LMM, we used LMM to recover the CPDAGs of medium size networks (500 nodes) with two different levels of density (500 and 1000 edges) as the number of observations becomes large. For every setting, 100 different networks were generated. Figure 12 shows the average TPR when recovering CPDAGs of different sizes at three different levels of observations (100, 1,000, and 10,000 samples).

Figure 12 shows that the recovered CPDAGs by LMM converge steadily towards the true CPDAGs as the number of observations grows large. For example, when the number of observations increased ten-fold, the difference between the recovered CPDAGs of the accurate size (correct number of edges) and the true CPDAGs decreased about four-fold each time.

8. Conclusion

In this work, we have presented a new constraint-based algorithm, light mutual min (LMM), for structural learning of Bayesian networks. When the observational data are limited in size, LMM improves the assessment of candidate skeleton edges by ranking them based on an estimate of the

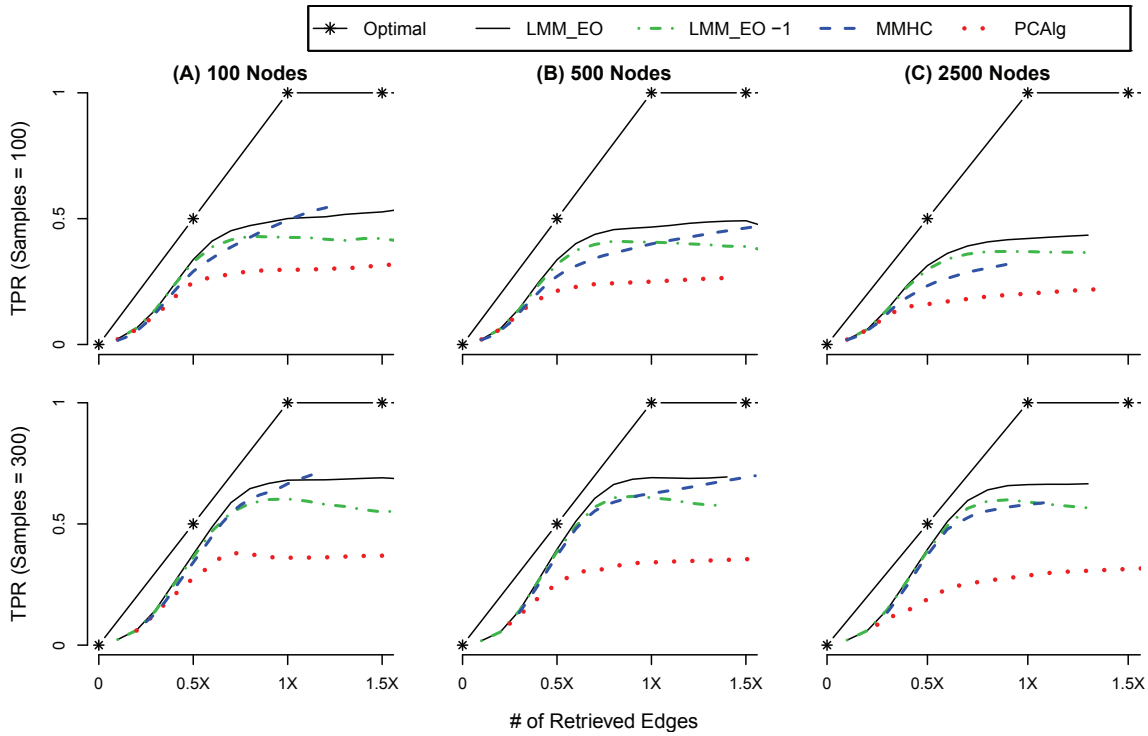

Figure 10: Average TPR when recovering CPDAGs of networks of A) 100, B) 500, and C) 2500 nodes with ∼200, ∼1,000, and ∼5,000 true edges respectively, when the number of samples is 100 (1st row) and 300 (2nd row). X-Axis is the number of retrieved edges (X = Number of edges in the correct network). Y-Axis is the average true positive rate (TPR)

joint conditional posterior that none of the neighboring variables at either sides of an edge can render the considered two variables conditionally independent. This approach is motivated by the asymptotic property, under the faithfulness assumption, that a parent and a child variable cannot be found independent when conditioning on any neighboring variables on either side of the edge. Therefore, considering conditional independence tests at both sides of the edge simultaneously provides complementary sources of evidence that can improve the assessment of candidate edges. In addition, to ease the multiple testing problem in recovering dense areas of the skeleton, LMM employs an adaptive relaxation of independence testing by, selectively, allowing some nodes not to condition on some of their neighbors. This relaxation is only performed whenever asymmetric evidence of conditional dependence is found between a pair of connected variables, where the aim is to reduce the accidental rejection of true edges connecting high degree nodes due to multiple testing. As the number of observational samples increases, the asymmetric evidence of dependence is expected to become less common and therefore the relaxation will be applied less often. This property makes the proposed technique work adaptively depending on the number of observational samples while at the same time maintains the asymptotic correctness of the algorithm. Moreover, this relaxation re-

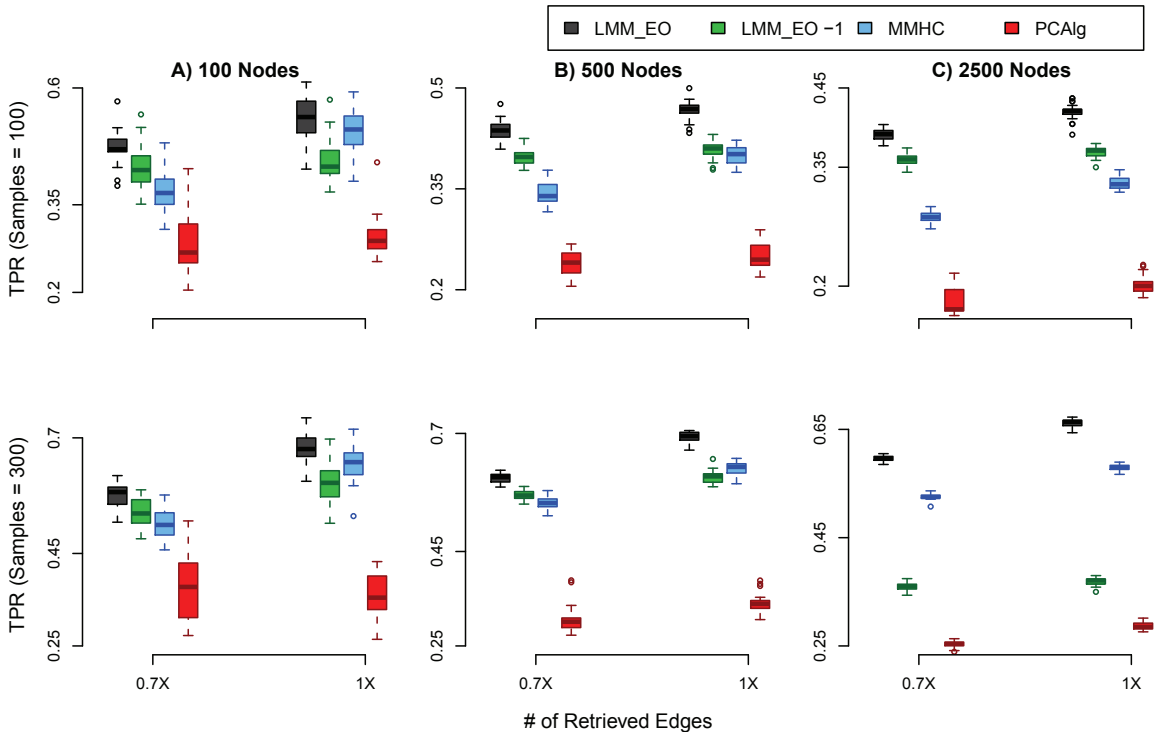

Figure 11: TPR of CPDAGs recovery of networks of size: A) 100, B) 500, and C) 2500 nodes with ∼200, ∼1,000, and ∼5,000 true edges when number of samples is 100 (1st row) and 300 (2nd row). X-Axis is the number of retrieved edges (X = Number of edges in the correct network). Y-Axis is the true positive rate (TPR).

sults in a significant reduction in the number of independence tests making LMM a computationally competitive learning tool for high-dimensional graphs and graphs with non-sparse connectivity.

In addition, we proposed a new approach to recover v-structures in a given skeleton based on the confidence about the dependence induced by the addition of the common neighbor to conditioning sets. The advantage of this method is the ability to rank candidate v-structures, providing a tool for resolving conflicts. When LMM is extended to recover the CPDAG of the equivalence class, the proposed conflict resolution method improved the accuracy of the recovered CPDAGs.

Appendix A. Proof of Correctness

In this section, we provide proofs of correctness of the proposed methods for the recovery of both the correct skeleton and CPDAG when given a very large number of observational samples (asymptotic limit). The provided proofs are similar in style and content to the proofs that were given for the MaxMin (Tsamardinos et al., 2006) and the PC algorithms (Kalisch and Bühlmann, 2007) with necessary modifications. First, we have to make the following necessary assumptions:

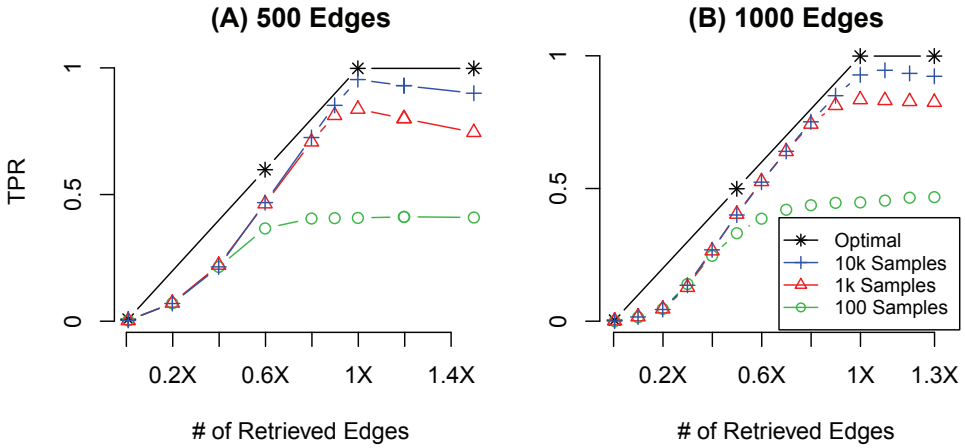

Figure 12: Average TPR of the recovered CPDAGs as the number of observations grows large when recovering networks of 500 nodes and A) ∼500 edges, and B) ∼1000 edges. X-Axis is the number of retrieved edges (X = Number of edges in the correct network). Y-Axis is the true positive rate (TPR).

A1: The observational samples are realizations of i.i.d random vectors $X_1, X_2, ...X_n$ with $X_i \in \Re^p$ and they are generated from a DAG $G = (V, E)$ with p nodes and a corresponding probability distribution P_n.

A2: The distribution P_n is multivariate Gaussian and faithful to the DAG G.

A3: The partial correlation $\rho_{ij|Z}$ between two variables i and j when conditioned on a set Z is an indicator of conditional independence under P_n where

$$\rho_{ij|Z} = 0 \; iff \; i \perp\!\!\!\perp j|Z.$$

A3 implies that the partial correlation method is an appropriate way to measure conditional independence from the given data and it is a necessary assumption for any constraint-based approach that uses this method for independence testing to be asymptotically correct. Based on A3 and given that the sample partial correlation $\hat{\rho}_{ij|Z}$ is an asymptotically correct estimator of $\rho_{ij|Z}$ (Hotelling, 1953), the following becomes correct:

$$\lim_{n\to\infty} |\hat{\rho}_{ij|Z}| = 0 \; iff \; i \perp\!\!\!\perp j|Z.$$

As a result and by substitution in Equation (2), for all positive and non-infinitesimal π_0 and π_A where $(\pi_0 + \pi_A = 1)$, the following also becomes correct:

$$\lim_{n\to\infty} P(\rho_{ij|Z} \neq 0|\hat{\rho}_{ij|Z}) = \begin{cases} 0 & iff \; i \perp\!\!\!\perp j|Z; \\ 1 & otherwise. \end{cases}$$

A.1 Correctness of LMM for Skeleton Recovery

The proof of LMM correctness for skeleton recovery relies on the symmetric dependence between parent and child variables in the asymptotic limit. The proof is given by Lemma 17 which also depends on Lemmas 14, 15 and 16 .

Lemma 14 *If E_{ij} is a true edge in the skeleton of a DAG $G = (V,E)$, then the one-sided CPPD $P(E_{ij} \mid D)_{[Z]} = 1$, $\forall Z \subseteq V_{\setminus i,j}$.*

Proof From assumption A1, $\nexists B \subseteq V_{\setminus i,j}$ s.t. $i \perp\!\!\!\perp j \mid B$. Based on this fact and based on assumption A3, then $P(\rho_{ij|B} \neq 0 | \hat{\rho}_{ij|B}) = 1$, $\forall B \subseteq Z$ and $\forall Z \subseteq V_{\setminus i,j}$. Therefore, by substitution in Equation (3), if E_{ij} is part of the correct skeleton then $P(E_{ij} \mid D)_{[Z]} = 1, \forall Z \subseteq V_{\setminus i,j}$. ■

Lemma 15 *If G_S is the set of all edges in the skeleton of G, and $\hat{G}_S^f$ is the set of all edges recovered in the forward selection phase of LMM(D,ω,γ), then $G_S \subseteq \hat{G}_S^f, \forall \gamma < 1$.*

Proof From Lemma 14, $\forall E_{ij} \in G_S$, $P(E_{ij}|D)_{[Z]} = 1, \forall Z \subseteq V_{\setminus i,j}$. Therefore, by substitution in Equation (4), the joint CPPD $P(E_{ij}|D)_{[CP_i,CP_j]} = 1, \forall CP_i, CP_j \subseteq V$. As a result, the first phase of LMM will not exit before all edges in G_S are recovered and therefore, $G_S \subseteq \hat{G}_S^f$. ■

Lemma 16 *If CP_i^* is the set of all parent and child variables of node i in the correct skeleton G_S, and CP_i is the set of neighbors of node i recovered in the forward selection phase of LMM(D,ω,γ), then $\forall \omega > 0$, $CP_i^* \subseteq CP_i$ (assuming $\gamma < 1$ and ω is not infinitesimal).*

Proof From Lemma 15, all $E_{ij} \in G_S$ are correctly recovered in the forward selection. Also, from Lemma 14, $P(E_{ij} \mid D)_{[CP_i]} = 1$, and $P(E_{ij} \mid D)_{[CP_j]} = 1, \forall i,j \; s.t. \; E_{ij} \in G_S$, and hence:

$$\lim_{n\to\infty} |P(E_{ij} \mid D)_{[CP_i]} - P(E_{ij} \mid D)_{[CP_j]}| < \omega, \; \forall \omega > 0.$$

As a result, lines 14 and 17 of LMM (Algorithm 1) will always get executed whenever a new edges is added when given a very large number of training samples. Therefore, after the forward selection, every node will be aware of all its child and parent variables, and thus $CP_i^* \subseteq CP_i$, $\forall i$.

Note that Lemma 16 emphasizes the property that the proposed relaxation of independence testing is adaptive, such that as the sample size of the data increases, the asymmetric one-sided CPPD among pairs of nodes connected in the true DAG occur less often and absent altogether when the sample size grows very large. ■

Lemma 17 *If G_S is the set of all edges in the skeleton of the graph G, and $\hat{G}_S$ is the set of all edges recovered by LMM(D,ω,γ), then $G_S = \hat{G}_S$.*

Proof From Lemma 15 and 16, after forward selection, all true edges were correctly recovered ($G_S \subseteq \hat{G}_S^f$), and the neighbor set of every node contains all its parent variables ($pa(i) \subseteq CP_i$, $\forall i$).

However, it is also expected that the forward selection might have recovered some false edges $E_{false} = \hat{G}_S^f - G_S$.

For every edge $E_{ij} \in \hat{G}_S^f$, since the graph is acyclic, then at least one of either i or j is a non-descendant of the other. From the local Markov property, a variable is independent from all its non-descendant variables if its parent variables are known. Therefore, $\forall\, E_{ij} \in E_{false}$, since $j \notin pa(i)$ and $i \notin pa(j)$, then either $P(E_{ij}|D)_{[CP_i]} = 0$ or $P(E_{ij}|D)_{[CP_j]} = 0$ (or both are zero) and by substitution in Equation (4):

$$P(E_{ij}|D)_{[CP_i,CP_j]} = P(E_{ij}|D)_{[CP_i]} \times P(E_{ij}|D)_{[CP_j]} = 0,\ \forall E_{ij} \in E_{false}.$$

On the other hand, if $E_{ij} \in \hat{G}_S^f$ is a correct edge, if i is the descendant of j then j is the parent of i (not a descendant of i). However, i cannot become independent of j since j will always be excluded from conditioning sets. Also, under the faithfulness property, i and j cannot be made independent when conditioning on any set of other variables and as a result:

$$P(E_{ij}|D)_{[CP_i,CP_j]} = P(E_{ij}|D)_{[CP_i]} \times P(E_{ij}|D)_{[CP_j]} = 1,\ \forall E_{ij} \in G_S.$$

Based on these conclusions, once the backward elimination starts, all false edges will have a joint CPPD of zero while all true edges will have a joint CPPD of one and therefore all false edges will be eliminated and thus: $G_S = \hat{G}_S$. ■

A.2 Correctness of LMM_Orientation for CPDAG Recovery

Lemma 18 *If CPDAG(G) is the complete partially acyclic graph of the equivalence class of the correct graph G, and $\widehat{G}$ is the partially acyclic graph recovered by LMM_EO, then CPDAG$(G) = \widehat{G}$.*

Proof From Lemma 17, in the asymptotic limit, LMM will recover the correct skeleton of G:

$$skeleton(\widehat{G}) = skeleton(G)$$

Also, it follows from Lemma 13 (Section 6.5): for every non-adjacent pair x and y with a common neighbor w, that w induces dependence whenever added to any set Z that makes x and y independent if and only if w is a common child of both x and y. Therefore,

$$\min_{Z \subseteq CP_x, w \in Z} P(\rho_{xy|Z} \neq 0 \mid \hat{\rho}_{xy|Z}) > \min_{Z \subseteq CP_x, w \notin Z} P(\rho_{xy|Z} \neq 0 \mid \hat{\rho}_{xy|Z})$$

and

$$\min_{Z \subseteq CP_y, w \in Z} P(\rho_{xy|Z} \neq 0 \mid \hat{\rho}_{xy|Z}) > \min_{Z \subseteq CP_y, w \notin Z} P(\rho_{xy|Z} \neq 0 \mid \hat{\rho}_{xy|Z})$$

if and only if $\{x, y\} \subseteq pa(w)$.

By direct substitution in line 5 of Algorithm 3, it follows that $CD(x, w, y) > 0$ if and only if $x \rightarrow w \leftarrow y$ is a correct v-structure in the true graph G. Therefore, lines 1-17 of Algorithm 3, will recover all and only the correct v-structures of the true graph:

$$vstructures(\widehat{G}) = vstructures(G).$$

The rest of the proof follows from the correctness of the orientation rules (R:1-4) that was provided by Meek (1995), in that, when the correct skeleton and the correct and complete set of v-structures are given then, the four rules (Algorithm 3: lines 18-24) will recover the correct CPDAG of G. Therefore, $\widehat{G} = CPDAG(G)$. ■

Appendix B. Effect of Overlapping Neighbor Sets on Ranking Edges

As stated in Section 6.1, the factorization in the right hand side of Equation (4) ignores the possibility that the two neighbor sets CP_i and CP_j might be overlapping. For example, if two nodes x and y found the common neighbor w, who also happen to be the only neighbor of both x and y, to induce conditional independence with the highest statistical significance at both sides of the edge, the joint conditional posterior in (4) becomes $P(\rho_{xy|w} \neq 0|\hat{\rho}_{xy|w})^2$. Since $P(\rho_{xy|w} \neq 0|\hat{\rho}_{xy|w})$ is always less than one, the joint posterior in this case becomes a lower bound of the posterior that x and y are not conditionally independent. However, as shown in the following examples, the overlap of neighbor sets is not expected to have a negative effect on the overall ranking of edges. The presented examples are not a proof of robustness but rather a supportive argument that the proposed ranking criterion is not expected to be sensitive to the issue of overlapping neighbors sets. To illustrate why this is the case, we analytically compute the expected rank of multiple edges under different circumstances while controlling for all other variables by making the following assumptions about the learning problem:

B1: G is the correct skeleton and the inference is based on finite data D.

B2: At the current iteration of the algorithm, CP_x, CP_y, CP_i, and CP_j are the inferred neighbor sets of x, y ,i , and j respectively. For simplicity and to control for other variables we will also assume:

(a) All neighbor sets have the same size.

(b) All neighbor sets contain the correct neighbors of each corresponding node.

(c) All conditioning sets are of the same size, or alternatively the effect of the size of the condition set on the sample partial correlation is negligible.

Again, all of these assumptions aim to control for variables and allow for an analytical solution for Equation (4) for the purpose of discussion.

B3: In all examples, we are only considering the average case, in that, we are sampling similar problems an extremely large number of times and we are only computing the average scenario. In all these problems, the size of the observational data and the structure of the skeleton are fixed while all other parameters of the simulation are randomized. As a result and based on Theorem 10, assumption B1, and the properties of partial correlations, it follows that there exist two constants ρ_0 and ρ_0^+ such that:

- If $D_{sep_G}(x,y|Z)$, then $E(|\rho_{xy|Z}|) = \rho_0$.
- If $\neg D_{sep_G}(x,y|Z)$, then $E(|\rho_{xy|Z}|) = \rho_0^+$.
- $1 \geq \rho_0^+ > \rho_0 \geq 0$.

These properties of the partial correlations are not new assumptions but rather a consequence of the assumption that partial correlation can identify conditional independence more accurately than random when learning from finite sample data. Based on this conclusion and by substitution in Equations (2) and (3), it follows that there also exist two constants $\mathcal{P}_0$ and $\mathcal{P}_0^+$ such that the following also holds:

- If $\exists Z \subseteq CP_x$ s.t. $D_{sep_G}(x,y|Z)$, then $E(P(E_{xy}|D)_{[CP_x]}) = \mathcal{P}_0$.
- If $\nexists Z \subseteq CP_x$ s.t. $D_{sep_G}(x,y|Z)$, then $E(P(E_{xy}|D)_{[CP_x]}) = \mathcal{P}_0^+$.
- $1 \geq \mathcal{P}_0^+ > \mathcal{P}_0 \geq 0$.

Under Assumptions B1-3, we now analytically compute the expected rank of two edges (E_{xy}, E_{ij}) in different circumstances where we assume two nodes to have full overlap of neighbors and the other two nodes to have no common neighbors.

Case 1: $E_{xy}, E_{ij} \in edges(G)$ and $CP_x = CP_y$ while $CP_j \cap CP_i = \emptyset$.
Analysis: Since E_{xy} and E_{ij} are in the true skeleton, there is no set Z that d-separates x and y or i and j. Using assumption B3 and substitution in Equation (4), the expected joint conditional posterior of both edges are equal:

$$E(P(E_{xy}|D)_{[CP_x,CP_y]}) = E(P(E_{ij}|D)_{[CP_i,CP_j]}) = (\mathcal{P}_0^+)^2.$$

As seen by this example, having common neighbors did not have negative or positive effect on ranking a correct edge against another correct edge in the average case.

Case 2: $E_{xy}, E_{ij} \notin edges(G)$ and $CP_x = CP_y$ while $CP_j \cap CP_i = \emptyset$.
Analysis: Given that E_{xy} is not part of the correct skeleton, it follows from the faithfulness assumption that there is at least one subset Z in either CP_x or CP_y that d-separates x and y. Since CP_x and CP_y are identical, the d-separation set exists in both CP_x and CP_y (Tsamardinos and Brown, 2008). Therefore, using assumption B3 and substituting in Equation (4), the expected joint conditional posterior of E_{xy} becomes:

$$E(P(E_{xy}|D)_{[CP_x,CP_y]}) = (\mathcal{P}_0)^2.$$

On the other hand, when assessing the Edge E_{ij}, there are two possibilities (Tsamardinos and Brown, 2008):

1. $\exists Z_1 \subseteq CP_i$ s.t. $D_{sep_G}(i,j|Z_1)$ and $\exists Z_2 \subseteq CP_j$ s.t. $D_{sep_G}(i,j|Z_2)$. Using Assumption B3 and by substitution in Equation (4), the expected joint conditional posterior of E_{ij} becomes:

$$E(P(E_{ij}|D)_{[CP_i,CP_j]}) = (\mathcal{P}_0)^2.$$

2. There is at least one subset in either CP_i or CP_j (not both) that d-separates i and j. Using Assumption B3 and by substitution in Equation (4), the expected joint conditional posterior of E_{ij} becomes:

$$E(P(E_{ij}|D)_{[CP_i,CP_j]}) = (\mathcal{P}_0) \times (\mathcal{P}_0^+).$$

As illustrated by Case 1 and Case 2, on average, the algorithm is expected to rank correct edges (using Equation 4) higher than false edges regardless whether there is or there is not overlap among neighbor sets:

$$(\mathcal{P}_0^+)^2 > (\mathcal{P}_0) \times (\mathcal{P}_0^+) > (\mathcal{P}_0)^2.$$

On the other hand, it is evident from Case 2 that Equation (4) might introduce bias in ranking false edges among each other in the presence of overlapping neighbor sets. This bias however is not enough to rank false edges higher than correct edges and this is the most crucial part of the learning algorithm.

Appendix C. Illustration of the CPDAG Evaluation Metric

Figure 13 shows an illustrative comparison between the structural Hamming distance (SHD) curve and the TPR curve. In both plots, the X-axis is the number of retrieved edges normalized by the correct number of edges. Also, all the metrics: true positives (TP), false negatives (FN), false positives (FP), and SHD are normalized by the number of correct edges. SHD counts the number of operations (add edge, delete edge, or change orientation) needed to transform a given CPDAG to the correct CPDAG. TP counts the number of recovered edges that are correct in presence and orientation. FP counts the number of incorrectly recovered or miss-oriented edges. FN counts the number of true edges that are either not recovered or miss-oriented.

In the TPR plot, the FN becomes (1-TP) while FP becomes the number of retrieved edges minus TP, and both of these metrics can be easily observed in the plot as shown in Figure 13:B. In contrast, it is not trivial to extract such information from the SHD curve.

Note that a mis-oriented edges is counted only once in the SHD metric, while it is counted twice in the TPR plot, once as a FP (false direction) and another as a FN (missing correct direction).

All presented plots in this paper aim at comparing methods. However, in the case of model selection, the solution with the least SHD should be the target. Another choice is to seek a CPDAG with the highest TP and the lowest FP (e.g., highest TP$-$FP). Also, we note that many authors usually use the X-axis to represent the threshold used in independence testing. However, in all plots in this paper, we used the number of retrieved edges as alternative because LMM does not use a threshold similar to MaxMin or PC algorithm.

Figures 14, 15, and 16 plot the average SHD versus the number of retrieved edges for the same experimental results plotted by the TPR plots in Figures 8, 10, and 12.

References

C. Aliferis, I. Tsamardinos, A. Statnikov, and L. Brown. *Causal Explorer:Causal Probabilistic Network Learning Toolkit for Biomedical Discovery*, 2003. URL `http://www.dsl-lab.org/causal_explorer`.

C.F. Aliferis, A. Statnikov, I. Tsamardinos, S. Mani, and X.D. Koutsoukos. Local causal and Markov blanket induction for causal discovery and feature selection for classification part ii: Analysis and extensions. *The Journal of Machine Learning Research*, 11:235–284, 2010.

A. Armen. Estimation and control of the false discovery rate in Bayesian network skeleton identification, with application to biological data. Master's thesis, Computer Science Department, University of Crete, Heraklion, Crete, Greece, 2011.

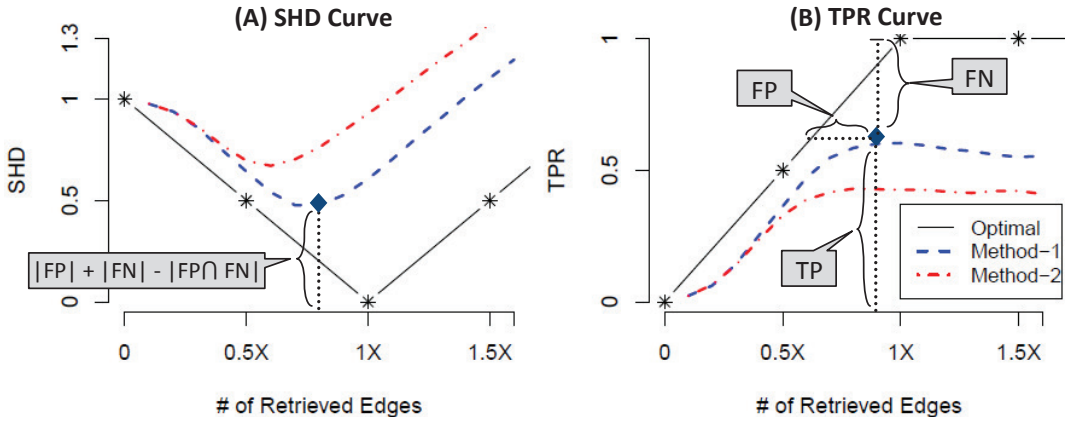

Figure 13: Illustration of A) SHD and B) TPR plots. In both plots, the X-Axis is the number of retrieved edges normalized by the number of correct edges. The metrics: FP, FN, TP, and SHD are also normalized by the correct number of edges. In the SHD plot, lower is better while in the TPR Plot, higher is better.

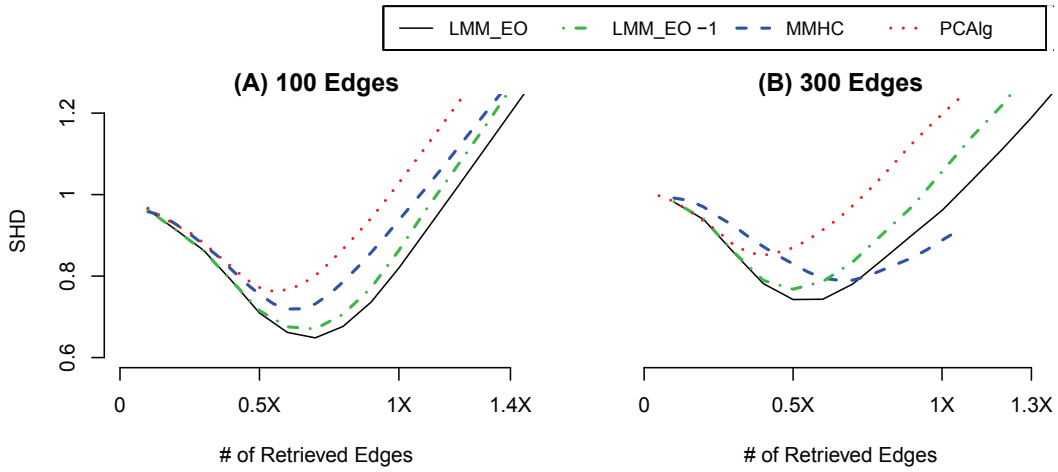

Figure 14: Average SHD when recovering CPDAGs of networks of 100 nodes and A) ∼100 or B) ∼300 true edges when the number of samples is 100. X-Axis is the number of retrieved edges (X = Number of edges in the correct network). Y-Axis is the average normalized structural Hamming distance (SHD).

J. Cheng, R. Greiner, J. Kelly, D. Bell, and W. Liu. Learning Bayesian networks from data: An information-theory based approach. *Artificial Intelligence*, 137(1-2):43 – 90, 2002. ISSN 0004-3702.

D. Chickering, D. Heckerman, and C. Meek. Large-sample learning of Bayesian networks is NP-hard. *Journal of Machine Learning Research*, 5:1287–1330, 2004. ISSN 1532-4435.

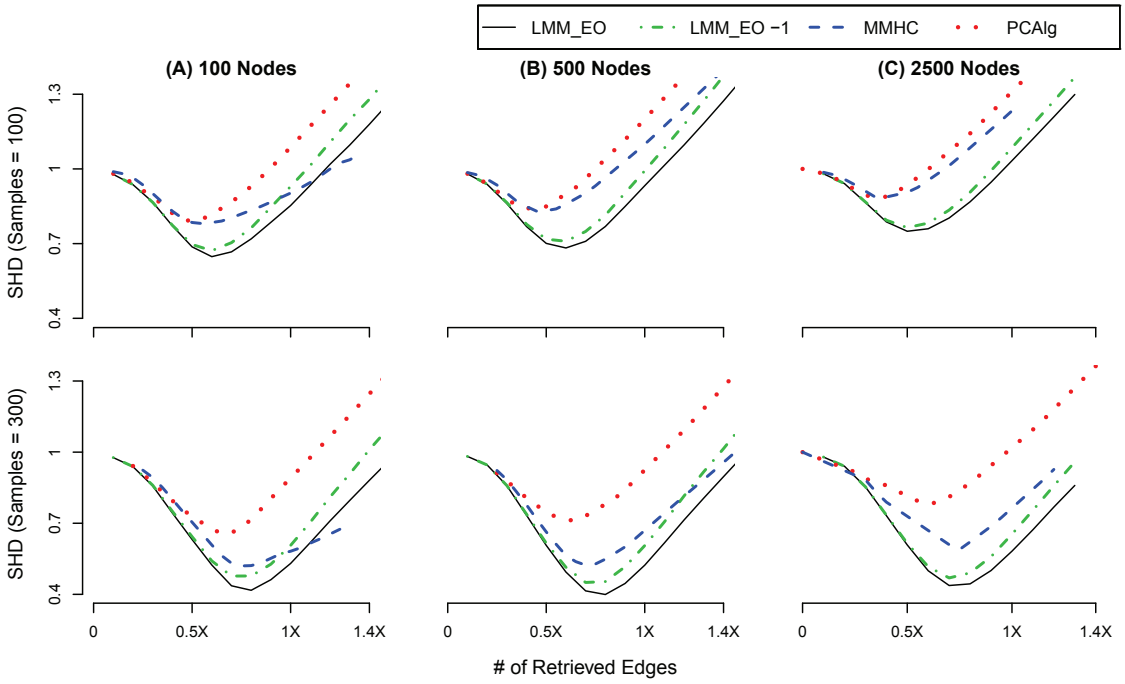

Figure 15: Average SHD when recovering CPDAGs of networks of A) 100, B) 500, and C) 2500 nodes with ∼200, ∼1,000, and ∼5,000 true edges respectively, when the number of samples is 100 (1st row) and 300 (2nd row). X-Axis is the number of retrieved edges (X = Number of edges in the correct network). Y-Axis is the average normalized structural Hamming distance (SHD).

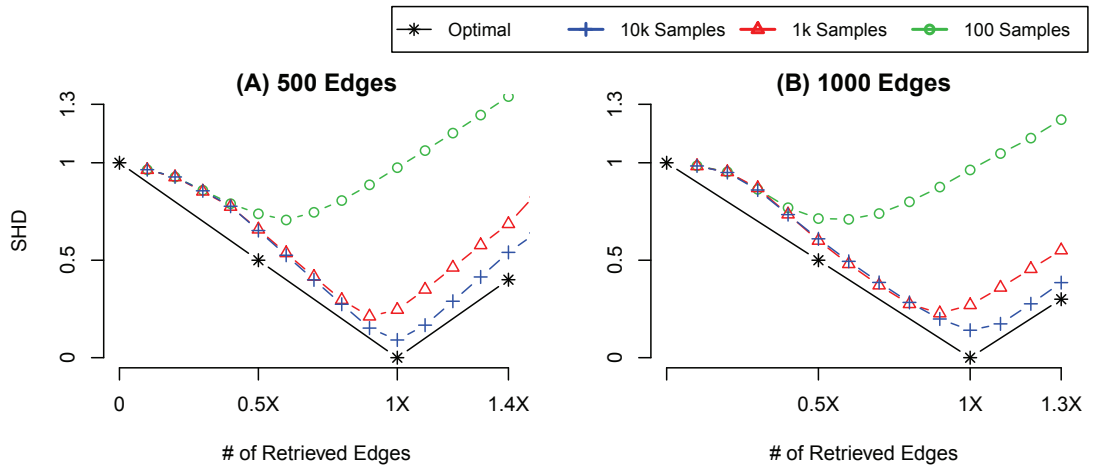

Figure 16: Average SHD of CPDAG recovery by LMM_EO as the number of observations grows large when recovering networks of 500 variables and A) ∼500 edges, and B) ∼1000 edges. X-Axis is the number of retrieved edges (X = Number of edges in the correct network). Y-Axis is the average normalized structural Hamming distance (SHD).

G. Cooper and E. Herskovits. Using Bayesian networks to analyze expression data. *Machine Learning*, 9:309–347, 10 1992.

R. Cowell, S. Lauritzen, P. David, and D. Spiegelhalter. *Probabilistic Networks and Expert Systems*. Springer-Verlag New York, Inc., Secaucus, NJ, USA, 1999. ISBN 0387987673.

A.S. Fast. *Learning the Structure of Bayesian Networks with Constraint Satisfaction*. PhD thesis, University of Massachusetts Amherst, Amherst, MA, USA, 2010.

N. Friedman. Inferring cellular networks using probabilistic graphical models. *Science Magazine*, 303(5659):799–805, 2004.

N. Friedman, I. Nachman, and D. Peer. Learning Bayesian network structure from massive datasets: The sparse candidate algorithm. In *5th Conference on Uncertainty in Artificial Intelligence*, pages 206–215, 1999.

N. Friedman, M. Linial, I. Nachman, and D. Pe'er. Using Bayesian networks to analyze expression data. *Journal of Computational Biology*, pages 601–620, 2000. doi: 10.

H. Hotelling. New light on the correlation coefficient and its transforms. *Journal of the Royal Statistical Society. Series B (Methodological)*, 15(2):193–232, 1953.

M. Kalisch and P. Bühlmann. Estimating high-dimensional directed acyclic graphs with the PC-algorithm. *Journal of Machine Learning Research*, 8:613–636, 2007. ISSN 1532-4435.

M. Kalisch, M. Maechler, and D. Colombo. *PCAlg: Estimation of CPDAG/PAG and causal inference using the IDA algorithm*, 2010. URL `http://CRAN.R-project.org/package=pcalg`. R package version 1.0-2.

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

J. Li and Z.J. Wang. Controlling the false discovery rate of the association/causality structure learned with the PC algorithm. *The Journal of Machine Learning Research*, 10:475–514, 2009.

D. Margaritis and S. Thrun. A Bayesian multiresolution independence test for continuous variables. In *Proceedings of the 17th Conference on Uncertainty in Artificial Intelligence*, pages 346–353. Morgan Kaufmann Publishers Inc., 2001.

C. Meek. Causal inference and causal explanation with background knowledge. In *Proceedings of the 11th Annual Conference on Uncertainty in Artificial Intelligence*, pages 403–441, San Francisco, CA, USA, 1995. Morgan Kaufmann.

R. Neapolitan. *Learning Bayesian Networks*. Pearson Printice Hall, London, UK, 2004.

J. Pearl. *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 1988. ISBN 1558604790.

E. Perrier, S. Imoto, and S. Miyano. Finding optimal Bayesian network given a super-structure. *Journal of Machine Learning Research*, 9:2251–2286, 2008.

S.J. Russell and P. Norvig. *Artificial Intelligence: A Modern Approach*. Prentice hall, Englewood Cliffs, NJ, USA, 2009.

J. Schäfer and K. Strimmer. An empirical Bayes approach to inferring large-scale gene association networks. *Bioinformatics*, 21(6):754, 2005.

M. Scutari. Learning Bayesian networks with the BNlearn R package. *Journal of Statistical Software*, 35(3):1–22, 2010.

P. Spirtes, C. Glymour, and R. Scheines. *Causation, Prediction, and Search*. Springer, New York, NY, USA, 1993.

P. Spirtes, C. Glymour, and R. Scheines. *Causation, Prediction, and Search, 2nd Edition*, volume 1. The MIT Press, Cambridge, MI, USA, 2001.

I. Tsamardinos and L.E. Brown. Bounding the false discovery rate in local Bayesian network learning. In *Proceedings of the 23rd National Conference on Artificial Intelligence*, pages 1100–1105, 2008.

I. Tsamardinos, L. Brown, and C. Aliferis. The max-min hill-climbing Bayesian network structure learning algorithm. *Machine Learning*, 65:31–78, 10 2006. ISSN 1532-4435.

T.S. Verma and J. Pearl. Equivalence and synthesis of causal models. In *Proceedings of the 6th Conference on Uncertainty in Artificial Intelligence*, pages 220–227, 1990.

J. Zhang and P. Spirtes. Strong faithfulness and uniform consistency in causal inference. In *Proceedings of the 19th Conference on Uncertainty in Artificial Intelligence*, pages 632–639. Citeseer, 2003.