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

Sub-Local Constraints Based Learning of Bayesian Networks Using A Joint Dependence Criterion (Supplementary Material)

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

1. Special Properties of Partial Correlation

In this section, we consider some of the characteristics of partial correlations between directly dependent variables when conditioning on a large set of true neighbors of either variable. This material is complementary to the discussion of Section 5 (main paper). The considered characteristics are summarized in Conjecture 1. Here, we only provide a proof for three special cases (problems 1-3 below). A proof for the generalized case is nontrivial due to the large number of possible complicated interactions that could be linking all neighbors.

Conjecture 1. *In the continuous multivariate Gaussian case, the magnitude of the conditional partial correlation between a variable x and another true neighbor variable y (parent or child) tends to decrease as the conditioning set containing other true neighbors increases in size ($|\rho_{xy|CP_x^+}| < |\rho_{xy|CP_x}|$, $CP_x \subset CP_x^+$) in all of the following cases:*
Case 1: y is a child of x and the set $\{CP_x^+ - CP_x\}$ is a set of parents of x.
Case 2: y is a child of x and the set $\{CP_x^+ - CP_x\}$ is a set of other children of x.
Case 3: y is a parent of x and the set $\{CP_x^+ - CP_x\}$ is a set of children of x.

Note that, in other cases, the partial correlation can increase in magnitude as a result of induced dependence. An example of this is when y is a parent of x and the set $\{CP_x^+ - CP_x\}$ is also a set of other parents of x. These cases do not defy the issues highlighted in Section 5 because constraints based methods always use the least significant partial correlation when conditioning on all subsets of neighbors. Therefore, cases where the partial correlation increases do not make a difference.

Problems 1 to 3 below are simplified cases of those in Conjecture 1 and the provided proofs show that this behavior is guaranteed in the infinite sample limit, which is also indicative of their expected behavior in the limited sample case.

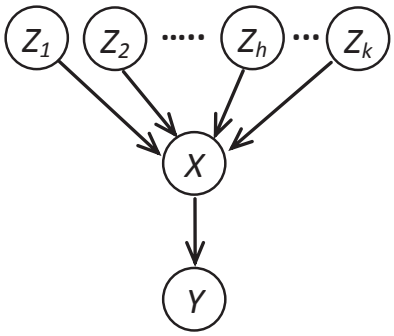

Figure 1: Case 1

Problem 1: Prove that the more we know about the parent variables of a variable X, the less dependent X will appear on its child variables.

Given the DAG in Figure 1 with the following parametrization:

$$Z_i \sim N(0,1)$$

$$X = \sum_{j=1}^{k} a_j Z_j + \varepsilon_0, \quad \varepsilon_0 \sim N(0,\sigma_0^2)$$

$$Y = bX + \varepsilon_1, \quad \varepsilon_1 \sim N(0,\sigma_1^2)$$

prove the following is true in the asymptotic sample limit:

$$|\rho_{XY|Z_{[1,t+1]}}| < |\rho_{XY|Z_{[1,t]}}|,$$

where

$$Z_{[1,t]} = \{Z_1, Z_2, ..Z_t\},\ Z_{[1,t+1]} = Z_{[1,t]} \cup Z_{t+1},\ \textit{and}\ t < k.$$

Proof
The following is true by substitution:

$$Y = b\sum_{j=1}^{k} a_j Z_j + \varepsilon_2, \quad \varepsilon_2 = (b\varepsilon_0 + \varepsilon_1) \sim N(0, b^2\sigma_0^2 + \sigma_1^2).$$

When regressing X and Y on $\{Z_1, Z_2 \ldots, Z_t\}$, asymptotically, we obtain the following regressions and residuals:

$$X = \sum_{j=1}^{t} a_j Z_j + r_{XZ_{[1,t]}}, \quad r_{XZ_{[1,t]}} = \sum_{j=t+1}^{k} a_j Z_j + \varepsilon_0 \sim N(0, \sigma_0^2 + \sum_{j=t+1}^{k} a_j^2),$$

$$Y = b\sum_{j=1}^{t} a_j Z_j + r_{YZ_{[1,t]}}, \quad r_{YZ_{[1,t]}} = b\left[\varepsilon_0 + \sum_{j=t+1}^{k} a_j Z_j\right] + \varepsilon_2 \sim N(0, b^2\left[\sigma_0^2 + \sum_{j=t+1}^{k} a_j^2\right] + \sigma_1^2).$$

Using the residuals, the partial correlation can be computed as follows:

$$\rho_{XY|Z_{[1,t]}} = \frac{N\sum_{i=1}^{N} r_{XZ_{[1,t]}}(i) r_{YZ_{[1,t]}}(i) - \sum_{i=1}^{N} r_{XZ_{[1,t]}}(i) \sum_{i=1}^{N} r_{YZ_{[1,t]}}(i)}{\sqrt{N\sum_{i=1}^{N}\left[r_{XZ_{[1,t]}}(i)\right]^2 - \left(\sum_{i=1}^{N} r_{XZ_{[1,t]}}(i)\right)^2}\sqrt{N\sum_{i=1}^{N}\left[r_{YZ_{[1,t]}}(i)\right]^2 - \left(\sum_{i=1}^{N} r_{YZ_{[1,t]}}(i)\right)^2}}.$$

Using properties of normal distributions, the above equation can be simplified to:

$$\rho_{XY|Z_{[1,t]}} = \frac{\sum_{i=1}^{N} r_{XZ_{[1,t]}}(i) r_{YZ_{[1,t]}}(i)}{\sqrt{\sigma_0^2 + \sum_{j=t+1}^{k} a_j^2}\sqrt{b^2\left[\sigma_0^2 + \sum_{j=t+1}^{k} a_j^2\right] + \sigma_1^2}}.$$

Since it is the case that:

$$r_{XZ_{[1,t]}}(i) r_{YZ_{[1,t]}}(i) = \left(\sum_{j=t+1}^{k} a_j Z_j + \varepsilon_0\right) \times \left(b\left[\varepsilon_0 + \sum_{j=t+1}^{k} a_j Z_j\right] + \varepsilon_2\right)$$

$$= b\left[\sum_{j=t+1}^{k} a_j Z_j + \varepsilon_0\right]^2 + \left(\sum_{j=t+1}^{k} a_j Z_j + \varepsilon_0\right) \times \varepsilon_2,$$

therefore:

$$\sum_{i=1}^{N} r_{XZ_{[1,t]}}(i) r_{YZ_{[1,t]}}(i) = b\left[\sigma_0^2 + \sum_{j=t+1}^{k} a_j^2\right].$$

By substitution, the partial correlation is:

$$\rho_{XY|Z_{[1,t]}} = \frac{sign(b)}{\sqrt{1 + \dfrac{\sigma_1^2}{b^2\left[\sigma_0^2 + \sum_{j=t+1}^{k} a_j^2\right]}}};$$

and since $a_{t+1}^2 > 0$:

$$\left[\sum_{j=t+2}^{k} a_j^2\right] < \left[\sum_{j=t+1}^{k} a_j^2\right]$$

then using inequality rules the following is true:

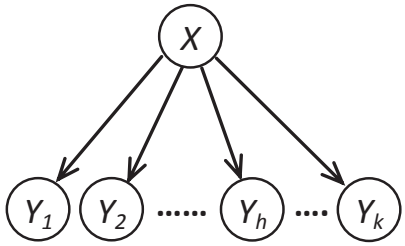

Figure 2: Case 2

$$\frac{|sign(b)|}{\sqrt{1+\frac{\sigma_1^2}{b\left[\sigma_0^2+\sum_{j=t+2}^{k}a_j^2\right]}}} < \frac{|sign(b)|}{\sqrt{1+\frac{\sigma_1^2}{b\left[\sigma_0^2+\sum_{j=t+1}^{k}a_j^2\right]}}},$$

which leads to:

$$|\rho_{XY|Z_{[1,t+1]}}| < |\rho_{XY|Z_{[1,t]}}|.$$

Problem 2: Prove that the more child variables we know of a node X, the less dependent X will appear on the other but yet undiscovered child variables.
Given the DAG in Figure 2 parametrized as follows:

$$X \sim N(0,1),$$

$$Y_j = b_j X + \varepsilon_j, \quad \varepsilon_j \sim N(0,\sigma_j^2),$$

prove the following is asymptotically true:

$$|\rho_{XY_h|Y_{[1,t+1]}}| < |\rho_{XY|Y_{[1,t]}}|,$$

where

$$Y_{[1,t]} = \{Y_1, Y_2, ..Y_t\},\ Y_{[1,t+1]} = Y_{[1,t]} \cup Z_{t+1},\ and\ t+1 < h \leq k.$$

Proof
When regressing X on $\{Y_1, Y_2, .., Y_t\}$ we obtain:

$$X = \sum_{j=1}^{t} c_j Y_j + r_{XY_{[1,t]}}, \quad r_{XY_{[1,t]}} \sim N(0, \sigma^2_{XY_{[1,t]}}).$$

By substitution, regressing Y_h on $\{Y_1, Y_2, ..., Y_t\}$ yields:

$$Y_h = b_h \sum_{j=1}^{t} c_j Y_j + r_{Y_h Y_{[1,t]}},\ r_{Y_h Y_{[1,t]}} = b_h r_{XY_{[1,t]}} + \varepsilon_h \sim N(0, b_h^2 \sigma^2_{XY_{[1,t]}} + \sigma_h^2).$$

Using the residuals to compute the partial correlation:

$$\rho_{XY_h|Y_{[1,t]}} = \frac{N\sum_{i=1}^{N} r_{XY_{[1,t]}}(i) r_{Y_hY_{[1,t]}}(i) - \sum_{i=1}^{N} r_{XY_{[1,t]}}(i) \sum_{i=1}^{N} r_{Y_hY_{[1,t]}}(i)}{\sqrt{N\sum_{i=1}^{N}\left[r_{XY_{[1,t]}}(i)\right]^2 - \left(\sum_{i=1}^{N} r_{XY_{[1,t]}}(i)\right)^2}\sqrt{N\sum_{i=1}^{N}\left[r_{Y_hY_{[1,t]}}(i)\right]^2 - \left(\sum_{i=1}^{N} r_{Y_hY_{[1,t]}}(i)\right)^2}},$$

$$\rho_{XY_h|Y_{[1,t]}} = \frac{b_h\sum_{i=1}^{N}\left(r_{XY_{[1,t]}}(i)\right)^2 + \sum_{i=1}^{N}\varepsilon_h r_{XY_{[1,t]}}(i)}{\sqrt{\sum_{i=1}^{N}\left[r_{XY_{[1,t]}}(i)\right]^2}\sqrt{\sum_{i=1}^{N}\left[r_{Y_hY_{[1,t]}}(i)\right]^2}} = \frac{b_h\sigma^2_{XY_{[1,t]}}}{\sqrt{\sigma^2_{XY_{[1,t]}}}\sqrt{b_h^2\sigma^2_{XY_{[1,t]}} + \sigma_h^2}},$$

$$\rho_{XY_h|Y_{[1,t]}} = \frac{sign(b_h)}{\sqrt{1 + \frac{\sigma_h^2}{b_h^2\sigma^2_{XY_{[1,t]}}}}} = \frac{sign(b_h)}{\sqrt{1 + \frac{\sigma_h^2}{b_h^2 E\left[r^2_{XY_{[1,t]}}\right]}}},$$

where $E(r)$ is the expected value of r.

Since the more predictors we use to regress X on, the smaller the variance of the residuals, given that the new predictors are not independent from the target X and are not a mere redundancy of existing predictors, it follows that:

$$E\left[\left(r_{XY_{[1,t+1]}}\right)^2\right] < E\left[\left(r_{XY_{[1,t]}}\right)^2\right]$$

and therefore,

$$\left|\rho_{XY_h|Y_{[1,t+1]}}\right| < \left|\rho_{XY|Y_{[1,t]}}\right|.$$

Problem 3: Prove that the more child variables we know of a node Y, the less dependent Y will appear on its true parent X.
Given the DAG in Figure 3 parametrized as follows:

$$X \sim N(0,1),$$

$$Y = aX + \varepsilon_0, \quad \varepsilon_0 \sim N(0,\sigma_0^2),$$

$$Z_j = b_jY + \varepsilon_j, \quad \varepsilon_j \sim N(0,\sigma_j^2).$$

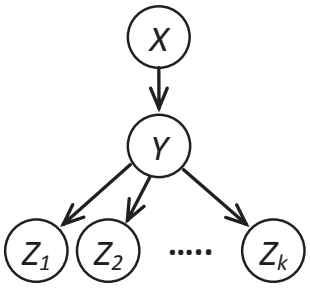

Figure 3: Case 3

prove the following is asymptotically true:

$$|\rho_{XY|Z_{[1,t+1]}}| < |\rho_{XY|Z_{[1,t]}}|,$$

where

$$Z_{[1,t]} = \{Z_1, Z_2, ..Z_t\},\ Z_{[1,t+1]} = Z_{[1,t]} \cup Z_{t+1},\ and\ t < k.$$

Proof
When regressing Y on $\{Z_1, Z_2, .., Z_t\}$ we obtain:

$$Y = \sum_{j=1}^{t} c_j Z_j + r_{YZ_{[1,t]}}, \quad r_{YZ_{[1,t]}} \sim N(0, \sigma^2_{YZ_{[1,t]}}).$$

By regressing X on $\{Z_1, Z_2, ..., Z_t\}$ and substitution:

$$X = \frac{1}{a}\sum_{j=1}^{t} c_j Z_j + r_{XZ_{[1,t]}}, \quad r_{XZ_{[1,t]}} = \frac{r_{YZ_{[1,t]}}}{a} - \frac{\varepsilon_0}{a} \sim N\left(0, \frac{\sigma^2_{YZ_{[1,t]}}}{a^2} + \frac{\sigma^2_0}{a^2}\right).$$

Using the residuals to compute the partial correlation:

$$\rho_{XY|Z_{[1,t]}} = \frac{N\sum_{i=1}^{N} r_{YZ_{[1,t]}}(i) r_{XZ_{[1,t]}}(i) - \sum_{i=1}^{N} r_{YZ_{[1,t]}} \sum_{i=1}^{N} r_{XZ_{[1,t]}}(i)}{\sqrt{N\sum_{i=1}^{N}[r_{YZ_{[1,t]}}(i)]^2 - \left(\sum_{i=1}^{N} r_{YZ_{[1,t]}}(i)\right)^2}\sqrt{N\sum_{i=1}^{N}[r_{XZ_{[1,t]}}(i)]^2 - \left(\sum_{i=1}^{N} r_{XZ_{[1,t]}}(i)\right)^2}},$$

$$\rho_{XY|Z_{[1,t]}} = \frac{\sum_{i=1}^{N}\left(\frac{r_{YZ_{[1,t]}}(i)}{a} - \frac{\varepsilon_0}{a}\right)\left(r_{YZ_{[1,t]}}(i)\right)}{\sqrt{\sum_{i=1}^{N}[r_{YZ_{[1,t]}}(i)]^2}\sqrt{\sum_{i=1}^{N}[r_{XZ_{[1,t]}}(i)]^2}},$$

$$\rho_{XY|Z_{[1,t]}} = \frac{\sum_{i=1}^{N} \frac{1}{a}\left(r_{YZ_{[1,t]}}(i)\right)^2 - \sum_{i=1}^{N}\left(\frac{r_{YZ_{[1,t]}}(i)\varepsilon_0}{a}\right)}{\sqrt{\sigma_3^2}\sqrt{\frac{\sigma^2_{YZ_{[1,t]}}}{a^2} + \frac{\sigma_0^2}{a^2}}},$$

$$\rho_{XY|Z_{[1,t]}} = \frac{\frac{\sigma^2_{YZ_{[1,t]}}}{a}}{\sqrt{\sigma^2_{YZ_{[1,t]}}}\sqrt{\frac{\sigma^2_{YZ_{[1,t]}}}{a^2} + \frac{\sigma_0^2}{a^2}}} = \frac{sign(a)}{\sqrt{1 + \frac{\sigma_0^2}{\sigma^2_{YZ_{[1,t]}}}}},$$

$$\rho_{XY|Z_{[1,t]}} = \frac{sign(a)}{\sqrt{1 + \frac{\sigma_0^2}{E\left(r^2_{YZ_{[1,t]}}\right)}}}.$$

Since the more predictors we use for Y, the smaller the variance of the residuals will be, given that the new predictors are not independent from the target Y and are not a mere redundancy of existing predictors, it follows that:

$$E\left(r^2_{YZ_{[1,t+1]}}\right) < E\left(r^2_{YZ_{[1,t]}}\right),$$

and therefore,

$$|\rho_{XY|Z_{[1,t+1]}}| < |\rho_{XY|Z_{[1,t]}}|.$$

2. Motivation of LMM Relaxation of Independence Testing (Part 2)

This section provides complementary discussion to Section 6.2.1 of the main paper that aims at explaining the motivation for the proposed relaxation of independence testing. In the Appendix of the paper, LMM was shown to recover the correct skeleton when given very large sample size. However, modeling the exact behavior of LMM for skeleton recovery is not trivial when learning from limited sample data, especially with the relaxation of independence testing. As an alternative, In this section, we illustrate the behavior of LMM when recovering simplified and small networks. The presented cases are not a proof of optimality but rather a supportive argument that the relaxation procedure employed by LMM can work well in skeleton recovery. Also, for the completeness of the discussion, we present some cases in which LMM might fail to reject all false edges due to relaxation of testing in limited sample problems.

LMM relaxation of independence testing is not expected to have negative influence on recovering true positives as a consequence of the faithfulness assumption, in that, if two variables are connected in the true network, then there does not exist any subset of other variables that can render them conditionally independent. Therefore, all cases presented in this section, are focused on whether LMM can successfully reject false edges or not.

In all presented examples, we consider the extreme case of testing relaxation (very small ω in Algorithm 1). We refer to a candidate solution as stable (or likely) outcome of applying LMM if the criterion of distributing connectivity information is fulfilled as follows:

$$\forall E_{ij} \in \hat{G}_S,\ j \notin CP_i \ if\ and\ only\ if\ P(E_{ij}|D)_{[CP_i]} > P(E_{ij}|D)_{[CP_j]}\,.$$

Since LMM is set to distribute connectivity information based on this criterion, then all output solutions that do not satisfy this property are considered not stable solutions or unlikely output of LMM where we expect further iterations of the algorithm to fulfill the distribution property by changing the corresponding neighbor sets.

In all presented cases, a candidate false edge becomes part of the solution if and only if neither of the corresponding two nodes is aware of all the neighboring variables that are sufficient to prove the conditional independence (e.g., set of all common parents). For visual illustration, false edges are represented by dotted lines and the awareness of a node i of its connection to node j ($j \in CP_i$) is represented by a solid circle at the end i of the edge E_{ij}.

For simplicity, in all cases, we assume the observational data is large but finite. As a result, whenever, we condition on the right d-separation set, we expect to correctly reject the false edge. However, since the data is finite, we also expect to find some asymmetric one-sided CPPDs between pairs of nodes as a result of Conjecture 1 (or the proofs of Problems 1-3) and the fact that nodes tend to have different set of neighbors.

2.1 Example 1

Given the two networks (a) and (b) in Figure 4, the outputs (ab-1) and (ab-2) are possible outcomes of applying LMM where the edge E_{BC} is correctly rejected due to conditioning on the common and only parent (A) from at least one side of the edge: B or C. On the other hand, the output (ab-3) is not a stable outcome since node A and node C have equal one-sided CPPD on each other due to having the same neighbor set $\{B\}$. Therefore, the algorithm is expected to make C aware of the neighbor A. This step will enable the algorithm to find that C is conditionally independent from B ($P(E_{BC}|D)_{[CP_C]} \cong 0$) and hence the edge E_{BC} can be successfully rejected.

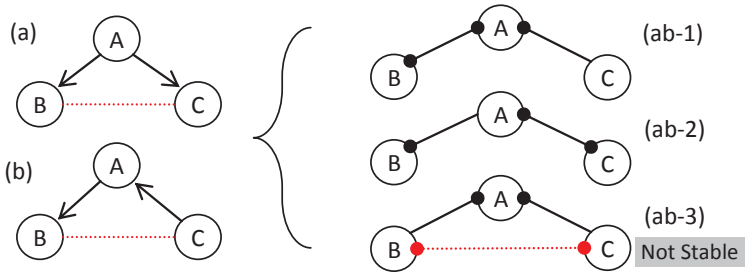

Figure 4: Given the true networks in (a) and (b), the solutions (ab-1) and (ab-2) are possible outcome of LMM but (ab-3) is not due to the expected equality of the one-sided CPPDs between A and C or B and A. Dotted red lines are candidate false edges. Solid circle at the end of the edge indicates the node is aware of the neighbor at the other end of the edge.

2.2 Example 2

Given the true network structure (c) in Figure 5, after the first connection between A and one of its child variables is recovered (i.e., E_{AD}), A will develop smaller or equally one sided CPPD to the rest of its child variables (e.g., $P(E_{AG}|D)_{[CP_A]} \leq P(E_{AG}|D)_{[CP_G]}$) due to multiple testing or due to developing partial correlations smaller in magnitude (see Conjecture 1: case 2, and problem 2 for proof). Therefore, a recovery of a new connection to another child will result in the new child to be aware of the parent A and therefore conditionally independent from the rest of the child variables. The output (c-1) is one possible stable outcome of using LMM to recover this network and all other possible distributions of connectivity information will result in a similar pattern that is expected to enable LMM to reject all false edges. For any false edges (i.e., E_{BG}) to be incorrectly recovered, both nodes must not be aware of the common parent A (i.e., output c-2). However, this solution is not possible due to the same reason given for the previous example.

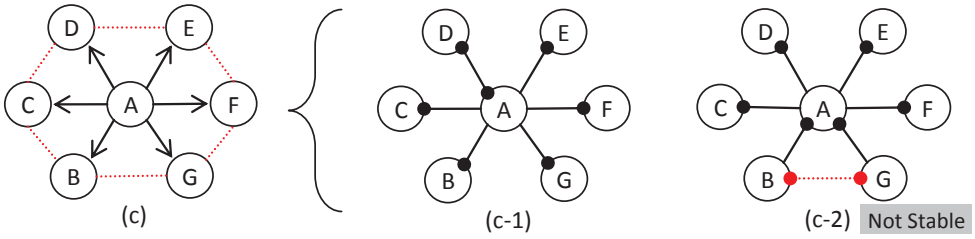

Figure 5: Given the true networks in (c), case (c-1) is possible but (c-2) is not due to the equal conditional dependence between A and C or B and A. Dotted red lines are candidate false edges. Solid circle at the end of the edge indicates the node is aware of the neighbor at the other end of the edge.

2.3 Example 3

Given the network structure (d) in Figure 6, (d-1) is one possible outcome in which in spite of distributing connectivity information, the false edge E_{BC} is correctly rejected ($P(E_{BC}|D)_{[CP_B,CP_C]} \cong 0$). Note that for both nodes B and C, once they are aware of one parent, that does not make them less dependent on the other parent. On the other hand, a hypothesized connectivity knowledge distribution as in output (d-2) that would allow a recovery of the false edge E_{BC} is not likely outcome since the nodes C and D are equally dependent on each other ($P(E_{CD}|D)_{[CP_C]} = P(E_{CD}|D)_{[CP_D]}$) since they share same neighbors $\{B\}$ and hence node C must be made aware of the neighbor D. This is also true for the pairs (A,B), (A,C), and (D,B). Similarly, the output (d-3) which would also allow the edge E_{BC} to be incorrectly recovered is also not possible outcome. This is because nodes C and D are equally dependent due to having the same neighbors $\{B\}$ and therefore node C must be made aware of the neighbor D. Similarly, C must be made aware of the neighbor A for the same reason. Note that, once C is aware of D, that does not reduce its dependence on A. In all possible outputs in this case, the output (d-1) is the only output that does not need any further update of the connectivity information to satisfy the distribution criterion.

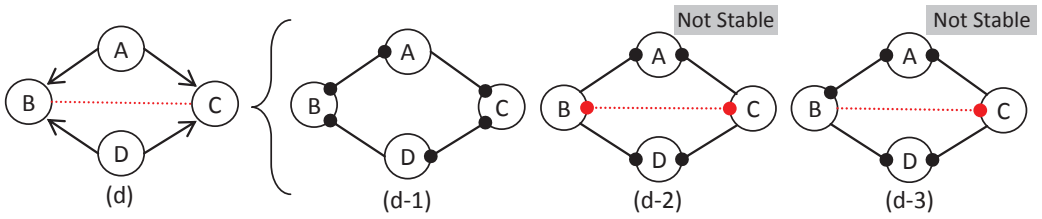

Figure 6: Given the true networks (d), case (d-1) is possible outputs of applying LMM, but cases (d-2) and (d-3) are not. Dotted red lines are candidate false edges. Solid circle at the end of the edge indicates the node is aware of the neighbor at the other end of the edge.

2.4 Example 4

Given the network structure (e) in Figure 7, the edges E_{BC} and E_{AD} are possible false positives. When applying LMM, the output (e-1) is one possible outcome in which in spite of distributing connectivity information the false edges E_{BC} and E_{AD} are correctly rejected. On the other hand, the output cases (e-3) and (e-4) are not possible due to similar reasoning of example 3. However, the solution output (e-2) can be stable and is a possible output. This case is stable because every pair of connected nodes who share the same neighbors are already made aware of each other. It is quiet possible in such a case the edge E_{BC} is incorrectly recovered. However, even then, the dependence along the edge E_{BC} is partially blocked, in that node C is aware of the node D which is part of the separation set $\{A, D\}$ and node B is aware of the node A which is also part of the separation set, therefore, although the mutual dependence B and C is still significant, it is expected to be low in magnitude (small joint CPPD).

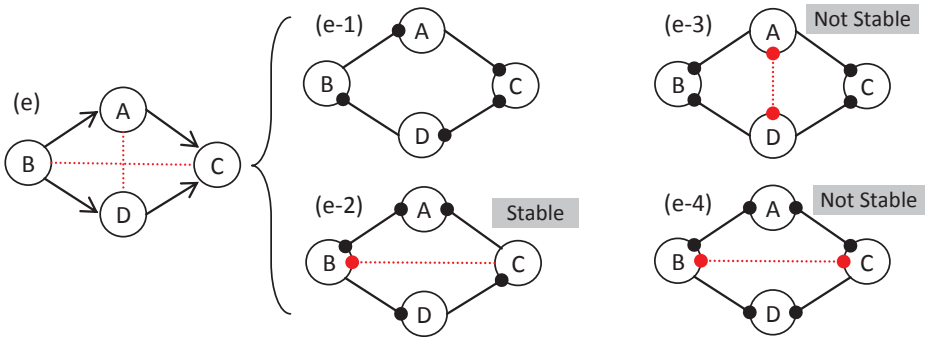

Figure 7: Given the true networks (e), case (e-1) and (e-2) are possible outputs of applying LMM, but cases (e-3) and (e-4) are not. Dotted red lines are candidate false edges. Solid circle at the end of the edge indicates the node is aware of the neighbor at the other end of the edge.

3. Optimized Implementation of LMM for Skeleton Recovery

The pseudo-code in Algorithm 1 and 2 (main paper) describes, in high level language, the proposed search strategy to construct a skeleton or to rank edges. In this section, we provide details on how we implemented LMM in a way that avoids most redundant computations while making the search for the next best edge to add or delete fast.

3.1 Data Structures

For representing information about the process of the inference of a network structure of d nodes, our implementation uses the following representations:

1. A $d \times d$ symmetric adjacency matrix ($adjM$). The entry $adjM[x][y]$ is set to 1 if x and y are connected, 0 if x and y are disconnected, and -1 if x and y are flagged to never be connected.

2. An array of objects of type set ($cpSets$) that stores the current neighbor sets. The entry $cpSets[x]$ represents the current set CP_x in algorithm 1 and 2.

3. A $d \times d$ matrix ($cppdM$) to store the current one sided CPPD of each node on other nodes for the given current $cpSets$. The entry $cppdM[x][y]$ stores the value of $P(E_{xy}|D)_{[CP_x]}$.

4. A list of all currently connected edges ($connectedEList$).

5. A list of all candidate edges that are not connected yet ($unConnectedEList$).

6. Every candidate edge is represented as an object that contains the indexes of the two nodes and their joint CPPD.

In our Java implementation, we created connectedEList and unConnectedEList from ArrayList class which is a dynamically allocated array. Reading and writing in ArrayList is of a constant complexity. Though inserting into an ArrayList may involve shifting some entries, inserting n entries to an ArrayList is of an amortized complexity of $O(n)$ and thus, inserting a single entry can be considered, on average, to have a constant complexity.

3.2 Initialization

When the algorithm starts, all entries in $adjM$ and $cppdM$ are set to zero while $connectedEList$, $unConnectedEList$, and all sets in $cpSets$ are empty. The initialization, then, continues as follows:

1. Initialize all entries in $cppdM$ to the one sided CPPDs (Equation 3, main paper). Since, initially, all nodes have identical neighbor sets (empty set), the $cppdM$ is perfectly symmetric at this point, and one only needs to compute the upper triangle of $cppdM$ and fill the lower triangle accordingly.

2. Using $cppdM$, we compute the joint CPPD (Equation 4, main paper) for all candidate edges and start filling the list $unConnectedEList$. Since the joint CPPD will either stay the same or decreases as the corresponding nodes add new neighbors in later iterations, the current joint CPPD can be used to exclude edges with the least joint CPPD from the search completely using either of the following strategies:

- If the algorithm is passed a threshold parameter γ for the joint CPPD, we exclude all edges with a joint CPPD less than γ.
- Fill *unConnectedEList* with a fixed but a significant fraction of candidate edges that have the highest initial joint CPPD and exclude the rest.

For the results presented in Section 7.2.2 (main paper, Table 1 and Figure 7), we have used the first strategy because LMM was run using a threshold parameter γ in a similar setting to the PC and MaxMin algorithms. In contrast, in all other experiments, to avoid guessing γ, we have used the second strategy where we let *unConnectedEList* add up to $d \times min(\frac{d-1}{2}, 100)$ edges. This later approach was in all cases equivalent to filtering using a very small γ (conservative filtering).

3. Whenever an edge is excluded from the search, it is removed from both *unConnectedEList* and *connectedEList* and the corresponding entry in $adjM$ is set to -1, and the corresponding one sided CPPDs in $cppdM$ are never updated in later iterations.

3.3 Iterative Updates and Construction of the Skeleton

As illustrated in Algorithm 1 (main paper), LMM builds the graph iteratively using forward selection and backward elimination of edges. To ensure a fast search for the next edge to add or delete while maintaining the joint CPPD of all candidate edges consistent with the current neighbor sets, we use the following approach:

1. At the first iteration, *unConnectedEList* is sorted according to the initial joint CPPD of each edge.

2. Whenever two nodes are connected, the corresponding edge is added to *connectedEList* and immediately removed from *unConnectedEList*.

3. Whenever, an edge is disconnected, it is removed from *connectedEList* and added to *unConnectedEList*.

4. During the forward selection (Algorithm 1, lines 1-6), whenever an edge is added to *unConnectedEList* or *connectedEList*, the two lists are maintained sorted by inserting the edge in the appropriate location. Since the lists are sorted, we use binary search to identify the appropriate location for insertion. As a result of being sorted all the time, finding the edge with the highest joint CPPD in *unConnectedEList* or the edge with the lowest joint CPPD in *connectedEList* is always very fast.

5. During backward elimination (Algorithm 1, lines 7-10), the list *unConnectedEList* is never updated because we are no longer considering adding any new edges.

6. Whenever a neighbor set of a node x is updated to include or exclude a neighbor z, the one sided CPPD of x on every other node y and the joint CPPD of the corresponding edges are

updated as follows:

$\forall y = 1,2 \ldots d$ s.t. $y \notin \{x,z\}$, do the following:
If $adjM[x][y] = -1$, do nothing.
Else update $cppdM[x][y]$ using Equation (3) and the new neighbor set $cpSets[x]$.
    If $cppdM[x][y]$ did not change, do nothing.
    Else, look up E_{xy} in either *connectedEList* or *unConnectedEList*.
        Remove E_{xy} from the edge list.
        Update the value of the joint CPPD associated with the object E_{xy}.
        Reinsert E_{xy} to the same edge list while maintaining the list sorted.

7. In the case of inserting a node z to the neighbor set of x, to update $cppdM[x][y]$, we compute $P(E_{xy}|D)_{[CP_x]}$ using only the conditioning sets that contain z and update $cppdM[x][y]$ only if this new $P(E_{xy}|D)_{[CP_x]}$ is less than the current one. This method will update cppdM correctly while avoiding half of the computation.

8. To look up the edges E_{xy} in *connectedEList* or *unConnectedEList* to update their joint CPPDs, we perform one scan of each list to find all edges that contains x. An alternative to this method is to use $adjM$ and the current joint CPPD (base on $cppdM$ before the update) to perform a binary search to locate each edge separately.

As illustrated by our implementation of LMM, the only overhead of computation we have added to traditional conditional independence testing is the one time sort of unConnectedEList, and the searches used to look up and insert edges in unConnectedEList and connectedEList. When using fast sorting method, such as Quick-Sort, and using early filtering (step 2) to exclude unpromising edges, the one-time sort of *unConnectedEList* induces relatively small complexity. Also, since binary search is of complexity $O(log_2(arraysize))$, inserting edges to *connectedEList* and *unConnectedEList* and maintaining them sorted is also of low complexity. Similarly, looking up edges in connectedEList and unConnectedEList to update their joint CPPDs using a one time scan for all edges or a binary search for each edge is of relatively low complexity.