Simulation Studies on DAG Structure Learning

Zhixuan Shao[*] Xiancheng Lin[†] Xuezhen Li[‡]

Department of Statistics
University of California, Davis

December 11, 2020

Abstract

Learning the DAG structure of a Bayesian Networks (BN) is of great interest with applications in many fields. In this paper, we introduce three types of methods: constraint-based, score-based and hybrid methods. We implement several typical algorithms of these types using **bnlearn** package in R. We compare the behaviours of different algorithms in both discrete DAG and Gaussian Bayesian Network (GBN), with synthetic data generated from relatively small networks. We show that if the sample size is large enough, all the algorithms can recover the true model. By contrast, when the sample size is not large, their performance become quite different, where score-based methods outperform the others.

1 Introduction

Bayesian Networks (BN) have been widely used in machine learning applications [1]. The structure of a BN takes the form of a directed acyclic graph (DAG) and plays a vital part in causal inference [2] with many applications in medicine, genetics [3], economics, and epidemics. Its structure learning problem is however NP-hard [4] and stimulates a proliferation of literature. Due to the NP-hardness, traditional DAG learning methods usually deal with *discrete variables* or *jointly Gaussian variables* [5]. This simulation study will also focus on these two cases.

[*]ID: 917804098
[†]ID: 917804917
[‡]ID: 915577631

1.1 Assumptions

Learning such BNs, especially from observational data, presents significant challenges. In particular, some additional assumptions are needed:

- There must be a one-to-one correspondence between the nodes in the DAG and the random variables in $\boldsymbol{X}$; this means in particular that there must not be multiple nodes which are deterministic functions of a single variable. Observations are treated as independent realisations of the set of nodes.
- All the relationships between the variables in $\boldsymbol{X}$ must be conditional independencies, because they are by definition the only kind of relationships that can be expressed by a BN.
- Every combination of the possible values of the variables in $\boldsymbol{X}$ must represent a valid, observable event. This assumption implies a *strictly positive global distribution*, which is needed to have uniquely determined Markov blankets and, therefore, a uniquely identifiable model.
- A DAG G and a joint distribution P are *faithful* to each other if all and only the conditional independencies true in P are entailed by G [2], i.e., for any subsets of nodes $\boldsymbol{A}$, $\boldsymbol{B}$, $\boldsymbol{C}$ in $\boldsymbol{V}$

  $$X_{\boldsymbol{A}} \perp\!\!\!\perp_G X_{\boldsymbol{B}} | X_{\boldsymbol{C}} \Longleftrightarrow X_{\boldsymbol{A}} \perp\!\!\!\perp_P X_{\boldsymbol{B}} | X_{\boldsymbol{C}}$$

  The faithfulness condition is needed to enable recovering the DAG G from the observed data $\widehat{P}$.

Naturally, since the DAG may have equivalent independence structures, and thus not identifiable, the best we can do is to find a graphical representation for the equivalence class of G, namely the *CPDAG* of G (an equivalence class that share the same skeleton and v-structures).

Several algorithms have been presented in literature for DAG structure learning, thanks to the application of results arising from probability, information and optimisation theory. Despite the variety of theoretical backgrounds and terminology they can all be traced to only three approaches: *constraint-based*, *score-based* and *hybrid*.

1.2 Constraint-Based Methods

Constraint-based algorithms are based on the seminal work of Pearl on maps and its application to causal graphical models. His *Inductive Causation* (IC) algorithm [2] provides a framework for learning the DAG structure of BNs using conditional independence tests.

The IC algorithm consists of three steps. The first step identifies which pairs of variables are connected by an arc, regardless of its direction. The second step deals with the identification of the v-structures among all the pairs of non-adjacent nodes i and j with a common neighbour k. The third and last step of the IC algorithm identifies compelled arcs and orients them recursively to obtain the CPDAG describing the equivalence class identified by the previous steps.

The first two steps in the original IC algorithm cannot be applied to any real-world problem due to the exponential number of possible conditional independence relationships. This has led to the development of improved algorithms such as:

- *PC*: Spirtes and Glymour (1991) [6] proposed a general systematic way of searching for the separating sets U_{ij} in step 1. Starting with sets U_{ij} of cardinality 0, then cardinality 1, and so on, edges are recursively removed from a complete graph as soon as separation is found. This refinement, called the PC algorithm (after its authors, Peter and Clark), enjoys polynomial time in graphs of finite degree because, at every stage, the search for a separating set U_{ij} can be limited to nodes that are adjacent to i and j.
- *Grow-Shrink* (GS): based on the Grow-Shrink Markov blanket algorithm [7], a simple forward selection Markov blanket detection approach;
- *Incremental Association* (IAMB): based on the Incremental Association Markov blanket algorithm [8], a two-phase selection scheme;

All these algorithms, with the exception of PC, first learn the Markov blanket of each node. This preliminary step greatly simplifies the identification of neighbours. This in turn results in a significant reduction in the number of conditional independence tests, and therefore of the overall computational complexity of the learning algorithm.

As far as the quality of the learned CPDAGs is concerned, on average Inter-IAMB produces fewer false positives than GS, IAMB or Fast-IAMB while having a comparable number of false negatives. The PC algorithm as extended by Bühlmann et al [9]. is also competitive.

Conditional independence tests used to learn discrete BNs include the *mutual information* test and the classic *Pearson's* χ^2 test for contingency tables. In the case of GBMs, conditional independence tests include the exact t test for Pearson's correlation and *Fisher's* Z test.

1.3 Score-Based Methods

Score-based learning algorithms represent the application of heuristic optimisation techniques to the problem of learning the structure of a DAG. Each candidate DAG is assigned a network score reflecting its goodness of fit, which the algorithm then attempts to maximise.

The most popular methods of this type are *greedy search* algorithms such as *hill-climbing* with *random restarts* or *tabu search* [10]. These algorithms explore the search space starting from a network structure (usually without any arc) and adding, deleting or reversing one arc at a time until the score can no longer be improved.

As far as network scores are concerned, the only two options in common use are posterior probabilities arising from flat priors such as *Bayesian Dirichlet equivalent uniform* (BDe) for discrete DAG and *Bayesian Gaussian equivalent uniform* (BGe) [11] for GBNs, and the BIC score. The latter is the default choice in `bnlearn`.

1.4 Hybrid Methods

Hybrid learning algorithms combine constraint-based and score-based algorithms to offset the respective weaknesses and produce reliable network structures in a wide variety of situations. The two best-known members of this family are the *Sparse Candidate algorithm* (SC) [12] and the *Max-Min Hill-Climbing* algorithm (MMHC) [13].

Both these algorithms are based on two steps, called *restrict* and *maximise*. In the first one, the candidate set for the parents of each node i is reduced from the whole node set V to a smaller set $C_i \subset V$ of nodes whose behaviour has been shown to be related in some way to that of X_i. This in turn results in a smaller and more regular search space. The second step seeks the network that maximises a given score function, subject to the constraints imposed by the C_i sets.

For example, the Max-Min Hill Climbing algorithm estimates the skeleton using a constraint-based method named *Max-Min Parents and Children* (MMPC), and then orients the edges by using a greedy search algorithm. Sparsity-inducing regularization approaches have also been used to develop efficient hybrid methods.

2 Simulation Design

2.1 Discrete DAG

Figure 1 shows the DAG model together with its corresponding CPDAG. We denote this DAG as G_1. Note that one edge is undirected in the CPDAG of G_1.

There are 6 variables in the model. Each variable follows a multinomial distribution conditioning on its parents. Each variable has three levels, named "a", "b", and "c", except variable **F** only has 2 levels "a" and "b". The conditional probability, namely, the contingency tables are given below.

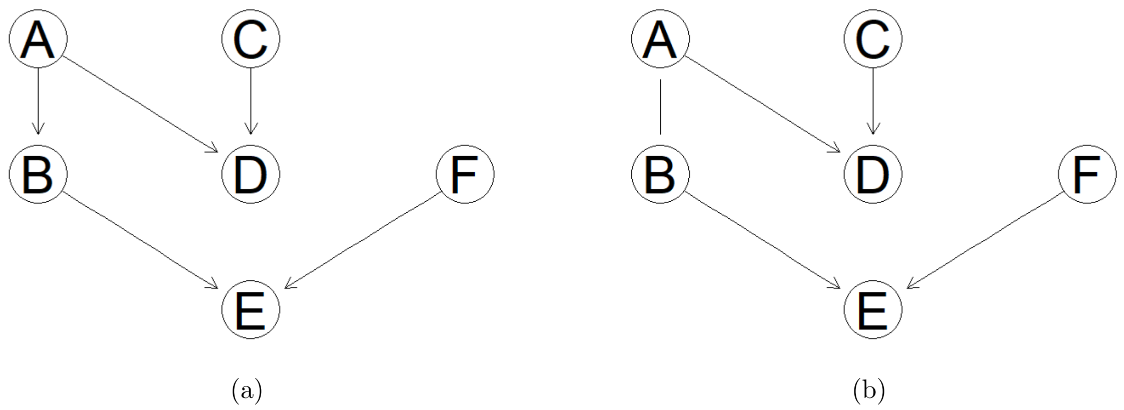


Figure 1: The left panel shows the DAG G_1. The right panel shows the CPDAG of G_1.

A	a	b	c
	1/3	1/3	1/3

C	a	b	c
	0.75	0.2	0.05

F	a	b
	0.5	0.5

B	a	b	c
A=a	0.8	0.1	0.1
A=b	0.4	0.2	0.4
A=c	0.1	0.1	0.8

D		a	b	c
C = a	**A**=a	0.8	0.1	0.1
	A=b	0.2	0.1	0.7
	A=c	0.4	0.2	0.4
C = b	**A**=a	0.1	0.8	0.1
	A=b	0.90	0.05	0.05
	A=c	0.3	0.4	0.3
C = c	**A**=a	0.1	0.1	0.8
	A=b	0.25	0.50	0.25
	A=c	0.15	0.45	0.40

E		a	b	c
$\mathbf{F} = a$	**B**=a	0.8	0.1	0.1
	B=b	0.4	0.5	0.1
	B=c	0.2	0.2	0.6
$\mathbf{F} = b$	**B**=a	0.3	0.4	0.3
	B=b	0.1	0.1	0.8
	B=c	0.25	0.50	0.25

We denote the above distribution as P_1. We then generate n independent samples from P_1. Specifically, we take n to be 500 and 5000 in our experiments.

2.2 Gaussian Bayesian Network (GBN)

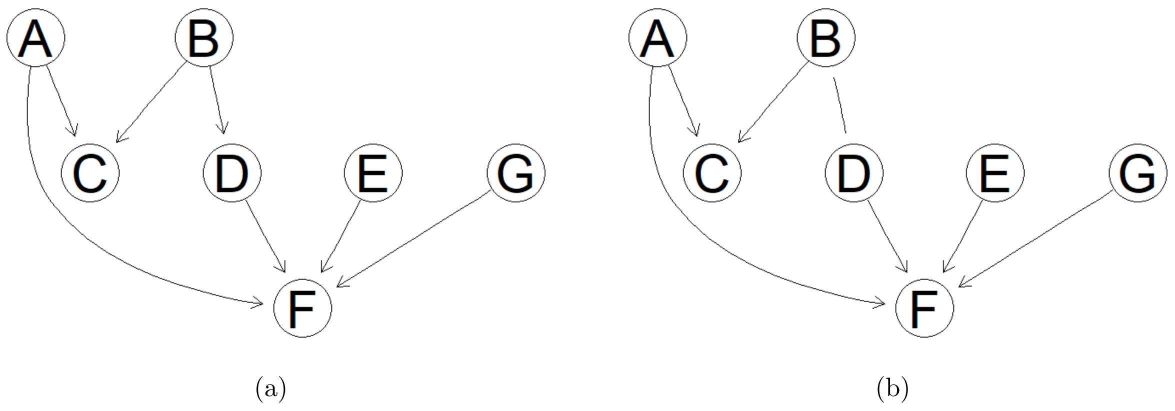


Figure 2: The left panel shows the DAG G_2. The right panel shows the CPDAG of G_2.

We denote the DAG in Figure 2 as G_2.

There are 7 variables in the model. Each variable follows a normal distribution conditioning on its parents.

- $\mathbf{A} = 1 + \epsilon_A, \quad \epsilon_A \sim \mathcal{N}(0, 1^2)$
- $\mathbf{B} = 2 + \epsilon_B, \quad \epsilon_B \sim \mathcal{N}(0, 3^2)$
- $\mathbf{E} = 3.5 + \epsilon_E, \quad \epsilon_E \sim \mathcal{N}(0, 2^2)$
- $\mathbf{G} = 5 + \epsilon_G, \quad \epsilon_G \sim \mathcal{N}(0, 2^2)$
- $\mathbf{C} = 2 + 2\mathbf{A} + 2\mathbf{B} + \epsilon_C, \quad \epsilon_C \sim \mathcal{N}(0, 0.5^2)$
- $\mathbf{D} = 6 + 1.5 \cdot \mathbf{B} + \epsilon_D, \quad \epsilon_D \sim \mathcal{N}(0, 0.33^2)$

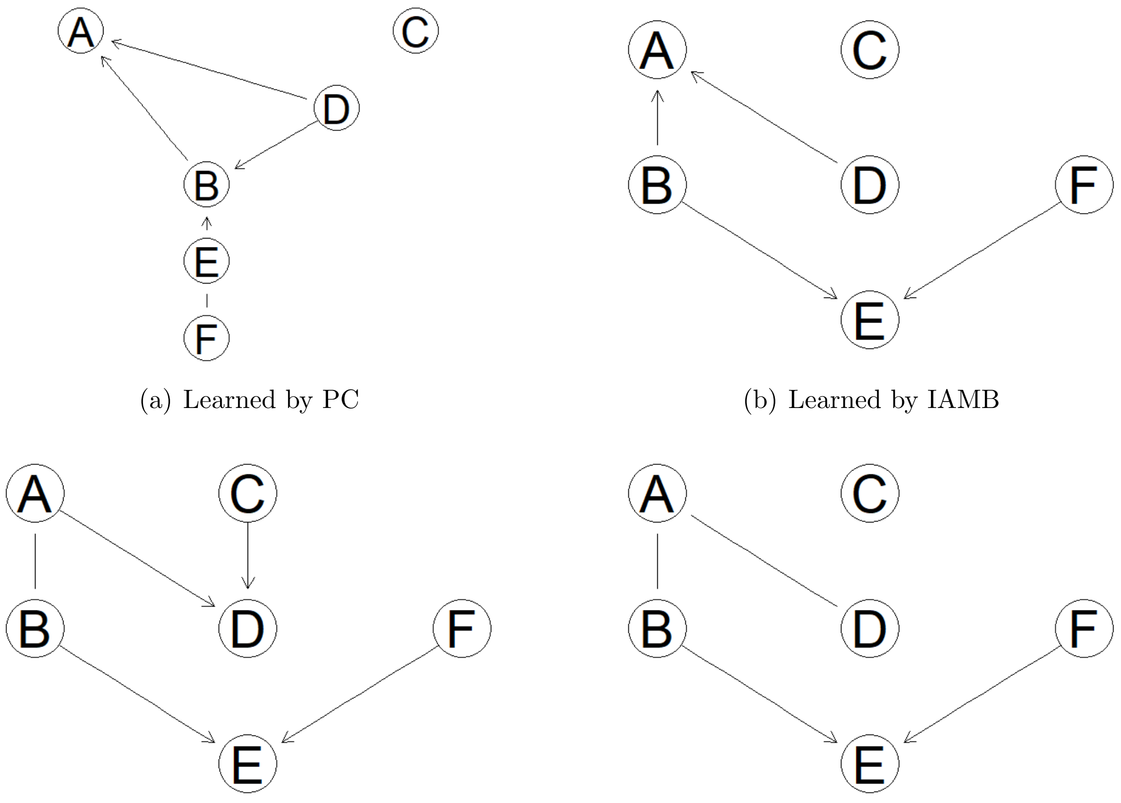
- $\mathbf{F} = 2\mathbf{A} + \mathbf{D} + \mathbf{E} + 1.5 \cdot \mathbf{G} + \epsilon_F, \quad \epsilon_F \sim \mathcal{N}(0, 1^2)$

We denote the above distribution as P_2. Again we generate n independent samples from P_2, and take n to be 500 and 5000.

3 Results

We use the `bnlearn` package to implement all three types of algorithms.

For constraint-based algorithms, we use `pc.stable` to implement the PC algorithm. Besides, we use the `iamb` to implement the IAMB algorithm. The default independence test for discrete DAGs is the *mutual information* test, while the default test for GBNs is the *Pearson's correlation*. The significant level is $\alpha = 0.05$ as default. For score-based algorithms, we use `hc` to implement the Hill-Climbing Greedy Search. For hybrid type, we use `mmhc` to implement the MMHC algorithm. Then We choose BIC as the score function.



(a) Learned by PC

(b) Learned by IAMB

(c) Learned by HC (True CPDAG)

(d) Learned by MMHC

Figure 3: Results of learning G_1 from $\widehat{P}_1$ when $n = 500$.

Although for score-based and hybrid methods, the algorithms in `bnlearn` return a fully directed graph, the direction of some edges are actually not identifiable. In particular, the score does not change if the direction of any of these edge is reversed. Therefore, we will evaluate the results based on the CPDAGs, and compare them with the CPDAG of the true model.

3.1 Discrete DAG

Table 1: Comparing the learned structure with the true CPDAG of G_1

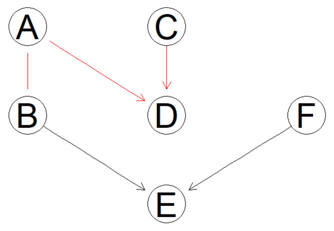
	TP	FP	FN	HD	SHD
PC	0	5	5	2	6
IAMB	2	3	2	1	3
HC	5	0	0	0	0
MMHC	3	2	1	1	2

TP stands for "True Positive".
FP stands for "False Positive".
FN stands for "False Negative".
SHD stands for Structural Hamming distance, which is defined as the number of edge insertions, deletions or flips in order to transform one graph to another graph.
HD stands for Hamming distance between their *skeletons*.

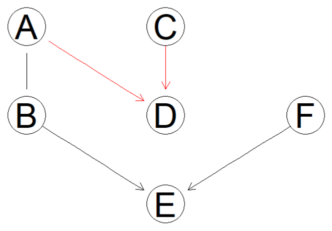
First we set the sample size $n = 500$. The learned structures are given in Figure 3, and their comparisons with the true CPDAG is given in Table 1 and Figure 4. Only the Hill-climbing algorithm gives the exactly correct CPDAG. Both IAMB and MMHC fails to identify an edge between **C** and **D**, and also fail to decide the edge direction between **A** and **D**. IAMB makes one more mistake than MMHC, though, which gives the wrong direction $\mathbf{B} \to \mathbf{A}$. However, the PC algorithm performs worst, which surprisingly fails to identify any single edge.

Note that all of them except HC fails to identify the edge between **C** and **D**. This may be caused by the weak "arc strength" of this edge, i.e., the relatively low mutual information given other variables. Therefore, when the sample size n is not large enough, algorithms may fail to identify this connection.

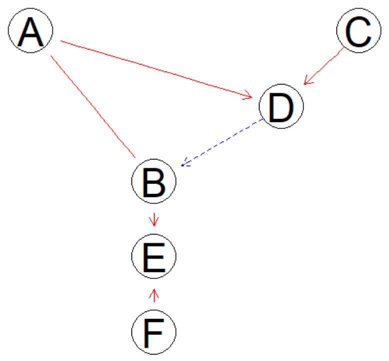
When $n = 5000$, all four methods recovered the CPDAG successfully. As expected, these algorithms are all able to identify the true CPDAG with sufficiently large sample size.



(a) Learned by IAMB



(b) Learned by MMHC



(c) Learned by PC

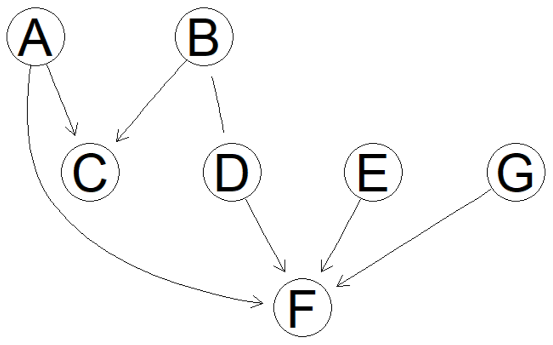
Figure 4: Comparing the learned structures with the true CPDAG of G_1.

3.2 GBN

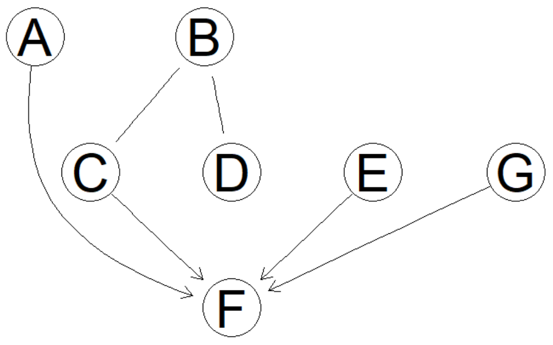
When $n = 500$, the learned structures are given in Figure 5, and their comparisons with the true CPDAG is given in Table 2 and Figure 6. Again, The HC algorithm successfully gives the true CPDAG, and so does IAMB. MMHC and PC, however, make similar mistakes. They both fail to identify edge between **A** and **C**, **D** and **F**, and the edge direction between **B** and **C**. PC, however, makes one more mistake by wrongly drawing an edge $\mathbf{C} \rightarrow \mathbf{F}$.

A possible limitation of the PC-Algorithm is that it requires partial correlations between adjacent nodes to be bounded away from 0; this requirement, which needs to hold for all conditioning sets U_{ij}, is known as *restricted strong faithfulness* [14].

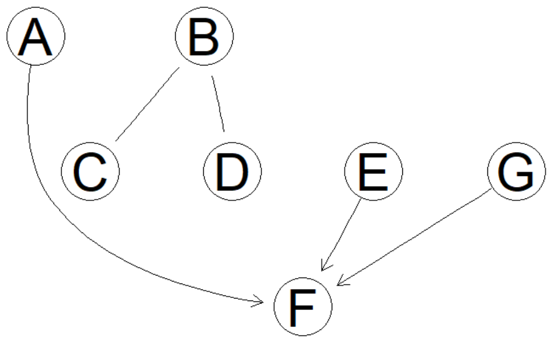
When $n = 5000$, again, all four methods recovered the CPDAG successfully, which verifies that these algorithms are also able to identify the true CPDAG in the Gaussian case.



(a) Learned by IAMB/HC (True CPDAG)



(b) Learned by PC

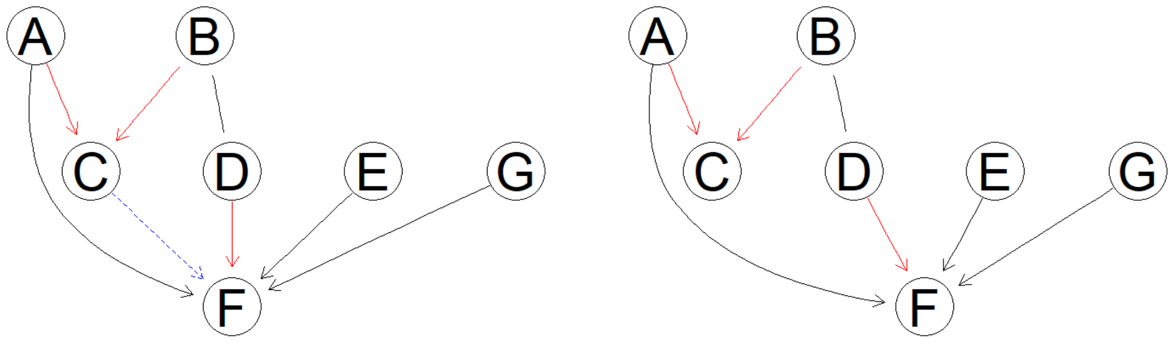


(c) Learned by MMHC

Figure 5: Results of learning G_1 from $\widehat{P}_1$ when $n = 500$.

Table 2: Comparing the learned structure with the true CPDAG of G_2

	TP	FP	FN	HD	SHD
PC	4	3	2	3	4
IAMB	7	0	0	0	0
HC	7	0	0	0	0
MMHC	4	3	1	2	3



(a) Learned by PC

(b) Learned by MMHC

Figure 6: Comparing the learned structures with the true CPDAG of G_2.

4 Conclusions

In this study, we assessed the three classes of BN structure learning methods: Constraint-based, score-based and hybrid methods with a discrete DAG and a Gaussian Bayesian Network. Both of them are relatively small networks with 6 and 7 variables, respectively. For each BN, our synthetic data has sample size 500 or 5000. We chose PC and IAMB algorithms, hill-climbing algorithm and Max-Min Hill Climbing algorithm for the three classes, respectively. We compared the CPDAGs learned by these algorithms with the true CPDAG.

With a limited sample size $n = 500$ and relatively low dimension $p < 10$, Hill-Climbing algorithm outperformed all other methods in both discrete and Gaussian cases. The classical PC algorithm performs worst under limited sample size. The large false positive rate may arise from the fact that PC algorithm condition on a large number of sets, leading to a high overall type-I error rate (rejecting edges too often). IAMB algorithm exhibited better performance than PC. Unexpectedly, MMHC does not perform as well as HC. The reason may be that in such an easy setting like ours, hybird algorithms do not exhibit advantages over score-based methods. Their advantages may become obvious under high dimensions or sparsity.

We also verify that all these algorithms can identify the true CPDAG with large enough sample size ($n = 5000$).

References

[1] Clark N Glymour and Gregory Floyd Cooper. *Computation, causation, and discovery.* Aaai Press, 1999.

[2] Judea Pearl. *Causality.* Cambridge university press, 2009.

[3] Sascha Ott, Seiya Imoto, and Satoru Miyano. Finding optimal models for small gene networks. In *Biocomputing 2004*, pages 557–567. World Scientific, 2003.

[4] David Maxwell Chickering, David Heckerman, and Christopher Meek. Large-sample learning of bayesian networks is np-hard. *Journal of Machine Learning Research*, 5(Oct):1287–1330, 2004.

[5] Karthik Mohan, Mike Chung, Seungyeop Han, Daniela Witten, Su-In Lee, and Maryam Fazel. Structured learning of gaussian graphical models. In *Advances in neural information processing systems*, pages 620–628, 2012.

[6] Peter Spirtes, Clark N Glymour, Richard Scheines, and David Heckerman. *Causation, prediction, and search.* MIT press, 2000.

[7] Dimitris Margaritis. Learning bayesian network model structure from data. Technical report, Carnegie-Mellon Univ Pittsburgh Pa School of Computer Science, 2003.

[8] Ioannis Tsamardinos, Constantin F Aliferis, Alexander R Statnikov, and Er Statnikov. Algorithms for large scale markov blanket discovery. In *FLAIRS conference*, volume 2, pages 376–380, 2003.

[9] Markus Kalisch and Peter Bühlmann. Estimating high-dimensional directed acyclic graphs with the pc-algorithm. *Journal of Machine Learning Research*, 8(Mar):613–636, 2007.

[10] Remco Ronaldus Bouckaert. *Bayesian belief networks: from construction to inference.* PhD thesis, 1995.

[11] Dan Geiger and David Heckerman. Learning gaussian networks. In *Uncertainty Proceedings 1994*, pages 235–243. Elsevier, 1994.

[12] Nir Friedman, Lise Getoor, Daphne Koller, and Avi Pfeffer. Learning probabilistic relational models. In *IJCAI*, volume 99, pages 1300–1309, 1999.

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

[14] Caroline Uhler, Garvesh Raskutti, Peter Bühlmann, and Bin Yu. Geometry of the faithfulness assumption in causal inference. *The Annals of Statistics*, pages 436–463, 2013.