A Score Based Ranking of the Edges for the PC Algorithm

A. Cano, M. Gómez-Olmedo, and S. Moral
Department of Computer Science and Artificial Intelligence
University of Granada
18071 - Granada, Spain

Abstract

The result of applying the PC learning algorithm can depend of the order in which independence tests are carried out. Even if these tests are ordered by increasing size of conditional sets, the PC algorithm does not take into account which edges are weaker in order to be considered to be removed before the stronger edges. This paper proposes a new learning algorithm which scores the edges according to a Bayesian metric and adds them to the final graph according to this score. Then, conditional independence tests are carried out to remove edgess as in the PC algorithm. Also, this algorithm is hybridized with a variation of the PC algorithm consisting in determining minimum size cut sets between two nodes to study the deletion of an edge. Some experiments are carried out to evaluate the performance of the new proposals against the PC algorithm.

1 Introduction

There are two main approaches to learning Bayesian networks from data. One is based on scoring and searching (Cooper and Herskovits, 1992; Heckerman, 1995). Its main idea is to define a global measure (score) which evaluates a given Bayesian network model as a function of the data. The problem is solved by searching in the space of possible Bayesian network models trying to find the network with optimal score. The other approach (constraint learning) is based on carrying out several independence tests on the database and building a Bayesian network in agreement with tests results. The main example of this approach is the PC algorithm (Spirtes et al., 1993).

In the past years, searching and scoring procedures have received more attention, due to some clear advantages (Heckerman et al., 1999). One is that constraint based learning makes categorical decisions from the very beginning. These decisions are based on statistical tests that may be erroneous and these errors will affect all the future algorithm bahaviour. On the other hand, the PC algorithm has some advantages. One of them is that it has an intuitive basis and under some ideal conditions it has guarantee of recovering a graph equivalent to the one being a true model for the data. It can be considered as a smart selection and ordering of the questions that have to be done in order to recover a causal structure.

In this paper, our basic idea is to combine the PC strategies with additional procedures to improve its performance. In this line, we can cite the work of van Dijk et al. (2003) who propose a combination of order 0 and 1 tests of the PC algorithm with a scoring and searching procedure; Dash and Druzdzel (1999) carry out several PC algorithms with different orders of the variables, which are scored afterwards with a Bayesian metric. In Abellán et al. (2006) we studied several variations of the basic PC strategy. Of them, the best performance was obtained by considering the minimum cut sets to carry out independence tests and changing the Chi-square based tests to Bayesian tests.

In this paper, we propose an algorithm, which considers two graphs: a graph of candidate edges and a graph of added edges (final graph). Each link in the candidates graph is scored with a Bayesian metric. While there are edges with a positive score, the edge with great-

est score is added to the final graph. Each time that an edge is added to the final graph, some conditional independence tests are carried out to deleted links from the candidates graph as in the PC algorithm. These tests are used to update the scores of the candidates graph. We will also test an hybridized version of this algorithm with the algorithm presented in Abellán et al. (2006), consisting in applying the new algorithm but doing only tests of order 0 and 1, and then the final graph is used as the initial structure for the algorithm presented in Abellán et al. (2006) considering minimum cut sets for the independence tests. Tsamardinos et al. (2006) follow a similar idea, but with some differences: the candidate links are locally computed for each single node (instead of following a global procedure) and the procedure finishes with a greedy optimization of a Bayesian score (without the orientation phase of the PC algorithm).

We will describe the new algorithms and we will make some experiments showing their performance when learning Asia and Alarm networks (Beinlich et al., 1989). The quality of the learned networks will be measured by the number of missing-added links and the Kullback-Leibler distance of the learned network to the original one.

The paper is organized as follows: Section 2 is devoted to describe the fundamentals of the PC algorithm and the variations introduced in Abellán et al. (2006). Section 3 introduces the new algorithm and its hybridization; in Section 4 the results of the experiments are reported and discussed; Section 5 is devoted to the conclusions.

2 The PC Algorithm

Assume that we have a set of variables $\mathbf{X} = (X_1, \ldots, X_n)$ with a global probability distribution about them P. By an uppercase bold letter $\mathbf{A}$ we will represent a subset of variables of $\mathbf{X}$. By $(\mathbf{A} \perp \mathbf{B}|\mathbf{C})$ we will denote that sets $\mathbf{A}$ and $\mathbf{B}$ are conditionally independent given $\mathbf{C}$.

The PC algorithm assumes *faithfulness*. This means that there is a directed acyclic graph, G, such that the independence relationships among the variables in $\mathbf{X}$ are exactly those represented by G by means of the d-separation criterion (Pearl, 1988). The PC algorithm is based on the existence of a procedure which is able to say when $(\mathbf{A} \perp \mathbf{B}|\mathbf{C})$ is verified in graph G. It first tries to find the skeleton (underlying undirected graph) and on a posterior step makes the orientation of the edges. Our variations are mainly applied to the first part (determining the skeleton). So we shall describe it with some detail:

1. Start with a complete undirected graph G'
2. $i = 0$
3. **Repeat**
   4. **For each** $X \in \mathbf{X}$
      5. **For each** $Y \in ADJ_X$
         6. Test whether $\exists \mathbf{S} \subseteq ADJ_X - \{Y\}$ with $|\mathbf{S}| = i$ and $(X \perp Y|\mathbf{S})$
         7. **If** this set exists
            8. Make $S_{XY} = \mathbf{S}$
            9. Remove $X - Y$ edge from G'
   10. $i = i + 1$
11. **Until** $|ADJ_X| \leq i, \quad \forall X$

In this algorithm, ADJ_X is the set of nodes adjacent to X in graph G'. The basis is that if the set of independencies is faithful to a graph, then there no link between X and Y, if and only if there is a subset $\mathbf{S}$ of the adjacent nodes of X such that $(X \perp Y|\mathbf{S})$. For each pair of variables, S_{XY} will contain such a set if it is found. This set will be used in the posterior orientation stage.

The orientation step will proceed by looking for sets of three variables $\{X, Y, Z\}$ such that edges $X - Z, Y - Z$ are in the graph by not the edge $X - Y$. Then, if $Z \notin S_{XY}$, it orients the edges from X to Z and from Y to Z creating a v-structure: $X \rightarrow Z \leftarrow Y$. Once, these orientations are done, then it tries to orient the rest of the edges following two basic principles: not to create cycles and not to create new v-structures. It is possible that the orientation of some of the edges has to be arbitrarily selected.

If the set of independencies is faithful to a graph and we have a perfect way of determining whether $(X \perp Y|\mathbf{S})$, then the algorithm has

guarantee of producing a graph equivalent (represents the same set of independencies) to the original one.

However, in practice none of these conditions are verified. Independencies are decided in the light of statistical tests based on a set of data $\mathcal{D}$. The usual way of doing these tests is by means of a chi-square test based on the cross entropy statistic measured in the sample (Spirtes et al., 1993). Statistical tests have errors and then, even if faithfulness hypothesis is verified, it is possible that we do not recover the original graph. The number of errors of statistical tests increases when the sample is small or the cardinality of the conditioning set $\mathbf{S}$ is large (Spirtes et al., 1993, p. 116). In both cases, due to the nature of frequentist statistical tests, there is a tendency to always decide independence (Cohen, 1988). This is one reason of doing statistical tests in increasing order of the cardinality of the sets to which we are conditioning.

In the PC algorithm it is possible that we delete the link between X and Y by testing the independence $(X \perp Y|\mathbf{S})$, when $\mathbf{S}$ is a set containing nodes that do not appear in a path (without cycles) from X to Y. The inclusion of these nodes is not theoretically wrong, but statistical tests make more errors when the size of the conditioning set increases, then it can be a source of problems in practice. For this reason, Steck and Tresp (1999) proposed to reduce $ADJ_X - \{Y\}$ in Step 6, by removing all the nodes that are not in a path from X to Y. In (Abellán et al., 2006), we considered any subset $CUT_{X,Y}$ disconnecting X and Y in the graph in which the link $X - Y$ has been deleted, playing the role of $ADJ_X - \{Y\}$. Consider that in the skeleton, we want to see whether link $X - Y$ can be deleted, then we first remove it, and if the situation is the one in Figure 1, we consider $CUT_{X,Y} = \{Z\}$. This version of the algorithm will be called BPC algorithm. The computation of this set needs some extra time, but it can be done in polynomial time with a modification of Ford-Fulkerson algorithm (Acid and de Campos, 1996).

The PC algorithm performs a chi-square statistical test to decide about independence.

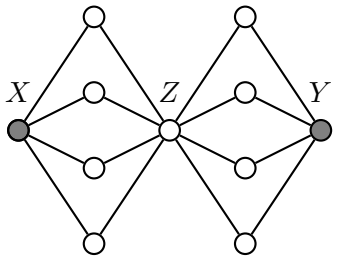

Figure 1: An small cut set

However, as shown by Moral (2004), sometimes statistical tests make too many errors. They try to keep the Type I error (deciding dependence when there is independence) constant to the significance level. However, if the sample is large enough this error can be much lower by using a different decision procedure, without an important increase in Type II error (deciding independence when there is dependence). Cooper (1997) had proposed a different independence test based on a Bayesian score, but only when conditioning to 0 or 1 variable. In (Abellán et al., 2006) we proposed to do all the statistical tests by using a Bayesian Dirichlet score (Heckerman, 1995) with a global sample size s equal to 1.0. The test $(X \perp Y|\mathbf{S})$ is carried out by comparing the scores of X with $\mathbf{S}$ as parents and of X with $\mathbf{S} \cup \{Y\}$ as parents. If the former is larger than the later, the variables are considered independent, and in the other case, they are considered dependent. The score of X with a set of parents $Pa(X) = \mathbf{Z}$, denoted as $BDe(X, \mathbf{Z})$ is the logarithm of:

$$\prod_{\mathbf{z}} \left(\frac{\Gamma(s')}{\Gamma(N_{\mathbf{z}} + s')} \prod_{x} \frac{\Gamma(N_{\mathbf{z},x} + s'')}{\Gamma(s'')} \right)$$

where $N_{\mathbf{z}}$ is the number of occurrences of $[\mathbf{Z} = \mathbf{z}]$ in the sample, $N_{\mathbf{z},x}$ is the number of occurrences of $[\mathbf{Z} = \mathbf{z}, X = x]$ in the sample, s' is s divided by the number of possible values of $\mathbf{Z}$, and s'' is equal to s' divided by the number of values of X.

We considered other variations as *refinement* and *triangle resolution*, but they did not prove to be very effective to recover the causal structure.

3 The Score Based PC Algorithm

This algorithm could be carried out with classical statistical tests or with Bayesian scores. In preliminary experiments, the statistical tests did not show a good performance in measuring the strength of an edge, so finally only Bayesian scores have been finally considered.

Instead of starting with a full graph G', now we start with two graphs: a full graph of candidate links G_c and an empty graph of added links G_a. We also have an array T in which we store a numerical value for each link $X-Y$, representing its strength.

The following steps are carried out:

- First, we compute the strength $T[X-Y]$ of each edge, by measuring the Bayesian score of Y conditioned to X minus the score of Y (conditioned to the empty set).
- If the score of an edge is negative, then it is removed from the graph of candidates G_c.
- While there are links in G_c, we select the edge with greatest score $X-Y$. This edge is removed from G_c and added to G_a.
- Once $X-Y$ is added to G_a, we make all the new necessary independence tests for all the candidate edges $Z-V$ in G_c. This is done in the following way:
  - We consider one of the extreme nodes, Z, of the edge $Z-V$ and compute the set of its adjacent nodes in graph G_a such that there is a path from each one of these nodes to the other extreme, V, before adding link $X-Y$ (denoted as $PADJ_Z$) and the set of adjacent nodes to Z such that there is a path from these nodes to V after adding link $X-Y$, but they are not in $PADJ_Z$ (denoted as $NewPADJ_Z$).
  - For each $\mathbf{S} \subseteq PADJ_Z \cup NewPADJ_Z$ such that $\mathbf{S} \cap NewPADJ_Z \neq \emptyset$ we compute the degree of dependence of Z and V given $\mathbf{S}$, by means of the expression: $Dep(Z,V|\mathbf{S}) = BDe(Z,\mathbf{S} \cup \{V\}) - BDe(Z,\mathbf{S})$.
    If this value is less or equal to zero, we can consider that Z is independent of V given $\mathbf{S}$ and the link $Z-V$ is removed from G_c.
    If $Dep(Z,V|\mathbf{S})$ is positive, then $T[Z-V]$ is set to the minimum of its present value and $Dep(Z,V|\mathbf{S})$.
  - If the link is not removed, we repeat above computations for the other extreme V of $Z-V$.

To compute $NewPADJ_Z$, we consider the nodes U in $ADJ_Z - PADJ_Z$, then this node is included in $NewPADJ_Z$ if there is a path from one of the extremes of $X-Y$ to V and a path from U to the other extreme.

The PC algorithm makes the statistical tests in increasing order of the cardinality of the conditional sets, but for the same cardinality the order is arbitrary. It can be the case that we have two links $X-Y$ and $U-V$ and that for a given cardinality, the link that is removed depends of whether X or U is considered first in step 4 of PC algorithm. Here, we try to make less arbitrary this selection, by measuring the strength of the links and adding the strongest links first, leaving the others for candidates to be removed.

Another source of inconsistencies is the following: the PC algorithm starts with a full network and then it removes the links in subsequent steps. When in step 6 of the algorithm, we are testing the independence of X and Y given a subset $\mathbf{S} \subseteq ADJ_X - \{Y\}$, then it is possible that the result is independence and the link $X-Y$ is removed, but later some link connecting X with a node of $\mathbf{S}$ is also removed. So, link $X-Y$ is removed by making a test conditional to a set containing nodes that are not adjacent to X in the final learned graph. This would not be a problem if all the tests were always correct, but the errors are more probable when the size of the conditional sets increases. So, this is a real problem in practice.

The new algorithm has some similarities with K2 greedy algorithm (Cooper and Herskovits, 1992) as adds the edges in increasing order of scoring, but there are some differences: first

it keeps the idea from PC algorithm of carrying out independence tests to remove candidate edges; and it leaves the orientation of the edges for a posterior step (K2 adds oriented links and our algorithm non-oriented edges).

The algorithm does not have the guarantee of recovering a directed acyclic graph under the faithfulness hypothesis and assuming that the statistical tests make always the right decision, as when a link is added to the final graph, it is never considered for deletion again. For having guarantee, it would be necessary that given a graph, the maximum of the scores T computed by the algorithm for a subset of the edges of the candidates graph is always obtained for an edge that is present in the original graph. However, the guarantee can be recovered if after our algorithm, the PC algorithm is applied to graph G_a (instead of starting with a full graph). This is precisely the basis of our hybridized version of the algorithm (called MPC algorithm):

1. To apply our algorithm, but doing only tests of order 1: instead of testing independence for each $\mathbf{S} \subseteq PADJ_Z \cup NewPADJ_Z$ such that $\mathbf{S} \cap NewPADJ_Z \neq \emptyset$, the independence is tested for every set $\mathbf{S} = \{U\}$, where $U \in NewPADJ_Z$.

2. To apply BPC algorithm, starting with graph $G' = G_a$, instead of the full graph, and considering minimum size cut sets.

That is, the idea of links strength is only applied to order 0 and 1 statistical tests. The rest works as in BPC algorithm. The basis is that in the experiments, we can observe that a great numbers of links are deleted precisely with these types of tests. Also, our algorithm had lost the property of doing the statistical tests in increasing cardinality of the conditional sets, and this new hybridized version recovers this property as the first part only carry out tests of order 0 and after order 1 tests. Then we apply BPC algorithm to carry our tests in increasing cardinality (starting with 1, as order 0 tests are not necessary).

4 Experiments

We have done some experiments with the Asia and Alarm networks for testing the PC variations. In all of them, we have generated samples of different sizes by simulation using logic sampling (Henrion, 1988). Then, we have tried to recover the original network from the samples by using the different variations of the PC algorithm including the orientation step. We have considered the following measures of error in this process: number of missing links, number of added links, and the Kullback-Leibler distance of the learned probability distribution to the original one[1]. Kullback-Leibler distance is a more appropriate measure of error when the objective is to approximate the joint probability distribution for all the variables and the measures of number of differences in links is more appropriate when our objective is to recover the causal structure of the problem. We do not consider the number of wrong orientations as our variations are mainly focused in the skeleton discovery phase of the PC algorithm.

Experiments have been carried out in Elvira environment (Elvira Consortium, 2002). The sample sizes we have used are: 100, 500, 1000, 5000, 10000, and for each sample size, we have repeated the experiment 100 times. The algorithms we have tested are the following:

- The classical PC algorithm with tests done by comparing Bayesian scores.
- The BPC algorithm as introduced in (Abellán et al., 2006) with minimal separating sets and score based tests.
- The new algorithm (HPC) consisting in adding edges in increasing strength value.
- The hybridized version (MPC).

Table 1 contains the average number of missing links, Table 2 the average number of added links, Table 3 the average Kulback-Leibler distance, and finally Table 4 contains the average running times of the different algorithms for the Asia network, whereas Tables 6, 7, and 8 contain the same data for the Alarm network.

[1]The parameters are estimated with a Bayesian Dirichlet approach with a global sample size of 2.

	100	500	1000	5000	10000
BPC	3,07	1,75	1,43	0,56	0,45
HPC	2,88	1,78	1,29	0,5	0,36
PC	4,46	3,35	3,23	2,77	2,62
MPC	3,01	1,74	1,34	0,51	0,41

Table 1: Average number of missing links (Asia)

	100	500	1000	5000	10000
BPC	1,61	0,79	0,73	0,19	0,25
HPC	1,4	0,52	0,31	0,15	0,13
PC	0,36	0,17	0,18	0,03	0,06
MPC	1,51	0,87	0,64	0,16	0,18

Table 2: Average number of added links (Asia)

	100	500	1000	5000	10000
BPC	0,2034	0,0842	0,0500	0,0209	0,0248
HPC	0,2147	0,0678	0,0344	0,0175	0,0192
PC	0,2934	0,1883	0,1957	0,2042	0,2020
MPC	0,2248	0,1644	0,1444	0,117	0,076

Table 3: Aver. K-L distance (Asia)

	100	500	1000	5000	10000
BPC	0,0540	0,2548	0,5327	2,9321	6,1866
HPC	0,0414	0,2408	0,5098	2,8342	6,1092
PC	0,0567	0,3508	0,7404	4,3941	9,3542
MPC	0,052	0,2784	0,6056	3,3988	7,1766

Table 4: Average time (Asia)

	100	500	1000	5000	10000
BPC	22,1	11,3333	7,8666	4,5	3,833
HPC	18,3	10,3	7,733	5,4666	5
PC	27,833	18,533	14,566	8,466	7,066
MPC	19,6	9,5	6,1	4,033	3,666

Table 5: Aver. number of missing links (Alarm)

	100	500	1000	5000	10000
BPC	13,9666	7,0333	4,9	3,7666	3,6333
HPC	9,1333	5,6666	4,8666	5,066	5,1333
PC	3,7	0,66	0,3660	0	0,033
MPC	11,2	5,5	2,8666	2,3	3

Table 6: Aver. number of added links (Alarm)

	100	500	1000	5000	10000
BPC	4,4597	2,3188	1,7611	0,8902	0,6839
HPC	3,9895	1,8093	1,3292	0,6411	0,6009
PC	5,0719	3,2705	2,5012	1,1374	0,7460
MPC	4,3594	2,5936	1,8357	0,7917	0,6289

Table 7: Aver. K-L distance (Alarm)

	100	500	1000	5000	10000
BPC	2,8917	8,4428	16,6012	92,8290	197,5764
HPC	0,9107	4,7952	10,5295	62,3101	132,7068
PC	1,1023	6,8248	16,2425	114,1656	252,8078
MPC	1,881	6,6598	13,1989	76,6548	168,1496

Table 8: Average time (Alarm)

In these results we highlight the following facts:

- In the simple Asia network new algorithm (HPC) has a better performance than BPC in terms of added links, deleted links and Kulback-Leibler distance, being more efficient in time, for all the sample sizes. With respect to PC algorithm and the Asia network, HPC has more errors in terms of added links, but the total number of errors (added + missing links) is lower in the new HPC algorithm.

- The hybridized algorithm (MPC) has an intermediate behaviour between BPC and HPC in terms of error and time in the Asia network. The Kullback-Leibler distance is worse than in both algorithms.

- In the Alarm network, the new algorithm is better in terms of total errors (added + missing links) than BPC for small or medium sample sizes (less or equal than 1000). However, the behaviour deteriorates for larger sample sizes. The hybridized algorithm is the best in terms of total errors for medium or larger sample sizes. However, the Kulback-Leibler distance is always lower for the new HPC algorithm. This can be interpreted in the following way: even if for large sample sizes, we can make more errors than in the BPC or MPC algorithms, these errors are less important (for example the missing links represent weaker dependencies).

- The new algorithm is always more efficient in time than all the other algorithms. This may be due to the fact that adding first the more important dependencies and making tests taking them into account, makes these

tests more successful in removing candidate links. The extra time necessary to compute $NewPADJ_Z$ is not as high as the time we save making less independence tests.

We were a bit surprised by the fact that the HPC algorithm does not show the best performance in terms of total number of errors in Alarm network for large sample sizes. Then, we analyzed the type of errors that the algorithm was doing in concrete cases. We have found that the errors are almost the same (same missing or added links) for different databases. In the case of missing links, these are usually one link pointing to one variable with several parents. Furthermore, only for a few combinations of values of the other variables, this new variable is really relevant (for the other combinations, the missing parent has little influence). When we make the test, it should produce dependence and it produces independence in a systematic way (with different databases). Our explanation is the following: assume that we are testing independence of X and Y given $\mathbf{S}$, by comparing the scores $BDe(X, \mathbf{S} \cup \{Y\})$ and $BDe(X, \mathbf{S})$. Both scores, can be expressed as a sum in the different values $\mathbf{s}$ of $\mathbf{S}$. If Y is relevant to X only for some values of $\mathbf{s}$, it is possible that for these values, we have $BDe(X, \mathbf{s} \cup \{Y\}) > BDe(X, \mathbf{s})$, but for the other values we may have the opposite inequality $BDe(X, \mathbf{s} \cup \{Y\}) < BDe(X, \mathbf{s})$. The final score is computed by adding in the different values $\mathbf{s}$. Then, the final result will depend of which of the differences is larger. Then, if we have asymmetrical independencies (or very weak dependencies) for a majority of values of $\mathbf{s}$, then the test will produce an error. This is due to the nature of the test. So, very little can be done by playing with the strategy of ordering the different independence tests.

The case of added links is different. As we have said, even under the faithfulness hypothesis and with no errors in the tests, our HPC algorithm does not have guarantee of recovering the original network, as it is possible that some of the tests necessary for the deletion of a link are never considered. When, this algorithm is combined with the BPC algorithm in the MPC algorithm, then the number of added links decreases, as more additional tests are carried out.

5 Conclusions

In this paper we have proposed two new strategies to organize the statistical tests in the PC algorithm: the HPC and the MPC algorithms. In general, the results of the HPC algorithm are better than in the classical PC algorithm and BPC algorithm proposed in Abellán et al. (2006). Furthermore, the HPC algorithm if the fastest in time. The MPC algorithm is not as good, but it produces the least number of errors in Alarm network for large sample sizes.

We recognize that the experiments in this paper are clearly insufficient and that more extensive experiments are necessary to determine in which conditions is appropriate to apply the new algorithms, but even that we want to highlight two points:

The first one is that though, at the present, general search algorithms are more common in practice, we believe that it is very promising to work in this type of PC based strategies. We believe that there are good opportunities of improving performance in terms of errors and time. In general, algorithms in graphs are fast compared with the time devoted to carry out statistical tests, so it is worthy to make some work in graph computations to save some statistical tests. We believe, that the orientation step could also use some of the ideas of this paper (sometimes there are conflicts among the different rules used for orientation, and it could be useful to decide with the help of an score).

The second point is that some more work is necessary to determine how to carry out the statistical tests. Classical statistical tests have known problems (Moral, 2004; Abellán et al., 2006). But, as we have shown here, Bayesian tests based on scores can produce systematic errors, for example in the case of asymmetrical independencies (or situations in which some of the dependencies are really weak). Perhaps, if for a value $\mathbf{s}$, we have $BDe(X, \mathbf{s} \cup \{Y\}) > BDe(X, \mathbf{s})$,

we should consider X and Y dependent given $\mathbf{S}$, with some correction given that we make multiple comparisons.

Acknowledgments

This work has been jointly supported by the Spanish Ministry of Education and Science under project TIN2007-67418-C03-03, by European Regional Development Fund (FEDER), and by the Spanish research programme Consolider Ingenio 2010: MIPRCV (CSD2007-00018).

References

J. Abellán, M. Gómez-Olmedo, and S. Moral. 2006. Some variations on the PC algorithm. In *Proceedings of the Third European Workshop on Probabilistic Graphical Models (PGM' 06)*, pages 1–8.

S. Acid and L.M. de Campos. 1996. Finding minimum d-separating sets in belief networks. In *Proceedings of the Twelfth Annual Conference on Uncertainty in Artificial Intelligence (UAI–96)*, pages 3–10, Portland, Oregon.

I.A. Beinlich, H.J. Suermondt, R.M. Chavez, and G.F. Cooper. 1989. The Alarm monitoring system: A case study with two probabilistic inference techniques for belief networks. In *Proceedings of the Second European Conference on Artificial Intelligence in Medicine*, pages 247–256. Springer-Verlag.

J. Cohen. 1988. *Statistical power analysis for the behavioral sciences (2nd edition)*. Erlbaum, Hillsdale, NJ.

Elvira Consortium. 2002. Elvira: An environment for probabilistic graphical models. In J.A. Gámez and A. Salmerón, editors, *Proceedings of the 1st European Workshop on Probabilistic Graphical Models*, pages 222–230.

G.F. Cooper and E.A. Herskovits. 1992. A Bayesian method for the induction of probabilistic networks from data. *Machine Learning*, 9:309–347.

G.F. Cooper. 1997. A simple constraint-based algorithm for efficiently mining observational databases for causal relationships. *Data Mining and Knowledge Discovery*, 1:203–224.

D. Dash and M.J. Druzdzel. 1999. A hybrid anytime algorithm for the construction of causal models from sparse data. In *Proceedings of the Fifteenth Annual Conference on Uncertainty in Artificial Intelligence (UAI-99)*, pages 142–149. Morgan Kaufmann.

D. Heckerman, C. Meek, and G. Cooper. 1999. A Bayesian approach to causal discovery. In C. Glymour and G.F. Cooper, editors, *Computation, Causation, and Discovery*, pages 141–165. AAAI Press.

D. Heckerman. 1995. A tutorial on learning with Bayesian networks. Technical Report MSR-TR-95-06, Microsoft Research.

M. Henrion. 1988. Propagating uncertainty by logic sampling in Bayes networks. In J. Lemmer and L.N. Kanal, editors, *Uncertainty in Artificial Intelligence, 2*, pages 149–164. Horth-Holland, Amsterdam.

S. Moral. 2004. An empirical comparison of score measures for independence. In *Proceedings of the Tenth International Conference IPMU 2004, Vol. 2*, pages 1307–1314.

J. Pearl. 1988. *Probabilistic Reasoning with Intelligent Systems*. Morgan & Kaufman, San Mateo.

P. Spirtes, C. Glymour, and R. Scheines. 1993. *Causation, Prediction and Search*. Springer Verlag, Berlin.

H. Steck and V. Tresp. 1999. Bayesian belief networks for data mining. In *Proceedings of the 2nd Workshop on Data Mining und Data Warehousing als Grundlage moderner entscheidungsunterstuetzender Systeme*, pages 145–154.

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

S. van Dijk, L.C. can der Gaag, and D. Thierens. 2003. A skeleton-based approach to learning Bayesian networks from data. In *Proceedings of the Seventh Conference on Principles and Practice of Knowledge Discovery in Databases*, pages 132–143. Springer Verlag.