Learning Bayesian Networks with Discrete Variables from Data*

Peter Spirtes and **Christopher Meek**
Department of Philosophy
Carnegie Mellon University
Pittsburgh, PA 15213

Abstract

This paper describes a new greedy Bayesian search algorithm (GBPS) and a new "combined" algorithm PC+GBPS for learning Bayesian networks. Simulation tests of these algorithms with previously published algorithms are presented.

Introduction

A Bayesian network consists of two distinct parts: a directed acyclic graph (DAG or belief-network structure) and a set of parameters for the DAG. The DAG in a Bayesian network can be used to represent both causal hypotheses and sets of probability distributions. Under the causal interpretation, a DAG represents the causal relations in a given population with a set of vertices V when there is an edge from a to b if and only if a is a direct cause of b relative to V. (We adopt the convention that sets of variables are capitalized and italicized, and individual variables are italicized.) Under the statistical interpretation (in the discrete case considered in this article) a DAG G can be taken to represent the set of all distributions that factor according to G in the following way:

$$P(V) = \prod_{x \in V} P(x|\Pi_x)$$

where Π_x is the set of parents of x in the DAG.

In this paper we will consider the following search problem. First, we will assume that we are searching for the correct causal structure represented by a DAG, rather than merely searching for a DAG that is a good predictor. Thus when we score how well a search algorithm performs, we will do so on on the basis of how close the answer provided by the algorithm is to the structure that actually generated the data. We will also assume that the user has no background knowedge about the correct causal structure other than that there are no latent variables.

Many researchers have recently investigated how the DAG part of a Bayesian network can be learned from data and optional background knowledge. Two distinct approaches to learning DAGs have been developed. The conditional independence approach (see e.g. Spirtes, Glymour and Scheines 1991 and 1993, Verma and Pearl 1991) uses tests of conditional independence to construct sets of DAGs that entail the same conditional independence relations. The Bayesian search approach (see Buntine 1991,Cooper and Herskovits 1992, Heckerman et al. 1994a and 1994b, Chickering et al. 1995) uses a Bayesian scoring metric combined with a search algorithm to look for the DAG with the highest posterior probability. Each approach has advantages and disadvantages.

This paper describes a greedy Bayesian pattern search algorithm (henceforth called GBPS) devised by Meek that is generally more accurate, but slower, than either the PC algorithm (which is a conditional independence search) or the greedy Bayesian DAG search algorithm (henceforth called GBDS) described in Chickering et al. However, a combined algorithm (henceforth called the combined algorithm) that uses the PC algorithm to generate an initial starting point for the GBPS is considerably faster than the GBPS, and approximately as accurate. (The PC algorithm can also be combined with the GBDS, but preliminary tests revealed that its performance was not as good as combining it with the GBPS). We emphasize that although the GBDS can be used in this kind of search, it was not primarily designed for this problem, and performs well when given sufficient background knowledge about the causal structure. In the following sections we briefly describ Bayesian networks, the PC algorithm, and two greedy Bayesian algorithms, and an algorithm which combines the PC algorithm and one of the Bayesian searches. We conclude by describing of some simulation tests of each of these algorithms. In these simulation tests, except at small samples sizes, on every dimension of reliability the combined algorithm and the GBPS tend to do better than the other two. (Singh and Valtorta (1993) also combined the PC algorithm with a Bayesian search algorithm with some success. Here we combine the PC algorithm with a different Bayesian search algorithm (one not requiring

*Research for this paper was supported by the Office of Navel Research grant ONR #N00014-93-1-0568.

a vertex ordering), and do more extensive simulation tests.)

Bayesian Networks

Bayesian network models are used extensively both in Artificial Intelligence (in expert systems) and in Statistics. Factor analysis models, path models with independent errors, recursive linear structural equation models with independent errors, and various kinds of latent variable models are all instances of Bayesian networks.

The factorization of a distribution according to a DAG G over a set of variables V entails a set of conditional independence relations among the variables in V. Assumptions linking the statistical and causal interpretations of DAG are discussed in Spirtes, Glymour and Scheines (1993).

Two DAGs that entail exactly the same set of conditional independence relations among the variables in V are said to be Markov equivalent. A class of Markov equivalent DAGs can be represented by a *pattern* (see Verma and Pearl 1990, and Spirtes, Glymour and Scheines 1991), which is a graphical structure consisting of a mixture of directed and undirected edges. More precisely, a DAG G is in the set of DAGs represented by pattern Π if and only if (i) G has the same adjacency relations as Π; (ii) if the edge between a and b is oriented $a \rightarrow b$ in Π, then it is oriented $a \rightarrow b$ in G; (iii) if there are edges $x \rightarrow y \leftarrow z$ in G, and no edge between x and z in G, (in which case we say the edges form an *unshielded collider* at z) then there are edges $x \rightarrow y \leftarrow z$ in Π.

The input to each algorithm described in this paper is a sample with no missing values from a population described by some DAG G. (Although each of the algorithms considered in this paper can take various kinds of optional background knowledge as input, no tests of the use of background knowledge were done for this paper.) The output of each of the algorithms is a pattern, rather than a DAG. The reason for this is that conditional independence relations alone do not distinguish between DAGs represented by the same pattern, and the scoring metric employed in the Bayesian search algorithms assigns the same score to each pair of DAGs represented by the same pattern. Hence the best one could do in principle using these procedures is to find a correct pattern. The pattern representing the set of DAGs containing G is called the *true pattern*.

The PC Algorithm

The PC algorithm takes as input raw data for a set of random variables assumed to be discrete or multivariate normal. We assume there are no latent variables. The output of the algorithm is a pattern. The algorithm is described in detail in Spirtes, Glymour, and Scheines (1993). The implementation in this test made some minor modifications to the PC algorithm to improve its performance.

If the population from which the sample input was drawn perfectly fits a DAG G all of whose variables have been measured, and the population distribution P contains no conditional independencies except those entailed by the factorization of P according to G, then in the large sample limit the PC algorithm produces the true pattern.

The degree of a vertex is the number of vertices adjacent to it. In the large sample limit, in the worst case the number of conditional independence tests required by the algorithm is bounded above by n^{k+2} where k is the maximum degree of any vertex in the true DAG. While we have no formal expected complexity analysis of the problem, the worst case is clearly rare, and the average number of conditional independence tests required for graphs of maximal degree k is much smaller. In practice it is possible to recover sparse graphs with as many as a hundred variables. Of course the computational requirements increase exponentially with k.

Bayesian Search Algorithms

The following brief account summarizes the assumptions and scoring metric introduced for discrete DAGs by Heckerman et al. (1994a), uses their notation, and directly quotes their Assumptions 4 through 6 and Theorem 1. The BDe metric they introduce is a variant and generalization of a metric introduced in Cooper and Herskovits (1992). The BDe metric assigns equal scores to any pair of DAGs that are Markov equivalent, as long as they have the same prior probability. Let U be a set of variables. A *case* is a single instance of all the variables in U. The first three assumptions basically state that the database was generated from a single Bayesian network with discrete variables, and there are no missing values.

A *database* is a sequence of cases $C_1, \ldots, C_m$. $\phi_{\vec{k}}$ is the multinomial parameter for the event $U = \vec{k}$. For two disjoint sets of variables $X, Y \subseteq U$, $\theta_{X=\vec{k}_X|Y=\vec{k}_Y}$ is the conditional probability of $X = \vec{k}_X$ given $Y = \vec{k}_Y$. The state assignments are omitted when the meaning is clear from the context, and the conditional bar is omitted when Y is empty. The event B_S^e corresponding to a Bayesian network B_S holds if and only if for $\theta_{x_i|x_1,\ldots,x_{i-1}}$ it is the case that

$$\forall Q \subset \Pi_i : \theta_{x_i|x_1,\ldots,x_{i-1}} \neq \theta_{x_i|Q}$$

where Π_i is the set of parents of x_i.

Let r_i be the number of possible different instantiations of x_i, and q_i be the number of possible different instantiations of Π_i. Order the possible instantiations of x_i into a list of values from 1 to r_i, and similarly list the possible instantiations of Π_i into a list of values from 1 to q_i. Let N_{ijk} the number of occurrences in the database of the k^{th} instantiation of x_i and the j^{th} instantiation of Π_i. Let

$$\Theta_{ij} \equiv \bigcup_{k=1}^{r_i} \{\theta_{ijk}\} \text{ and } \Theta_{B_S} \equiv \bigcup_{i=1}^{n} \bigcup_{j=1}^{q_i} \{\theta_{ijk}\}$$

Let ρ represent a density and ξ represent background knowledge that is conditioned on.

Assumption 4 (Parameter Independence): For all belief-network structures B_S,

$$\rho(\Theta_{B_S}|B^e_S,\xi) = \prod_i \prod_j \rho(\Theta_{ij}|B^e_S,\xi)$$

Assumption 5 (Parameter Modularity): If x_i has the same parents in any two belief-network structures B_{S1} and B_{S2}, then for $j = 1,\ldots,q_i$,

$$\rho(\Theta_{ij}|B^e_{S1},\xi) = \rho(\Theta_{ij}|B^e_{S2},\xi)$$

A *complete* DAG is one with no missing edges.

Assumption 6: For every complete belief-network structures B_{S_C} and for all $\Theta_{ij} \subseteq \Theta_{B_{S_C}}$, $\rho(\Theta_{ij}|B^e_{S_C},\xi)$ has a Dirichlet distribution. Namely there exists exponents N'_{ijk} such that

$$\rho(\Theta_{ij}, B^e_{S_C},\xi) = c \cdot \prod_k \theta_{ijk}^{N'_{ijk}}$$

where c is a normalization constant.

Heckerman et al. (1994a) show that if DAGs that entail the same set of conditional independence and dependence relations are equally likely given any database, then $N'_{ijk}+1 = K \cdot p(x_i = k, \Pi_i = j|B^e_{S_C},\xi)$ where K is a constant. The constant K is often called an equivalent sample size, and controls how fast the data changes the posterior probability of a Bayesian network.

Theorem 1 *Given Assumptions 1 through 6,*

$$p(D, B^e_{S_C}|\xi) =$$

$$p(B^e_{S_C}|\xi) \cdot \prod_{i=1}^{n}\prod_{j=1}^{q_i} \frac{\Gamma(K \cdot p(\Pi_i = j|B^e_{S_C},\xi))}{\Gamma(N_{ij} + K \cdot p(\Pi_i = j|B^e_{S_C},\xi))} \cdot$$

$$\prod_{k=1}^{r_i} \frac{\Gamma(N_{ijk} + K \cdot p(x_i = k, \Pi_i = j|B^e_{S_C},\xi))}{\Gamma(K \cdot p(x_i = k, \Pi_i = j|B^e_{S_C},\xi))}$$

This scoring metric can be combined with a variety of search algorithms. The basic problem is that even for a small number of variables the number of different possible Bayesian networks is too large to make an exhaustive search feasible. A number of heuristic algorithms are described in Cooper and Herskovits (1992) and Heckerman et al. (1994a). The simplest algorithm (which we will call the greedy Bayesian DAG search or GBDS) which is studied below starts with some initial DAG, and does a greedy search which at each stage makes the single modification graph (an edge addition, edge removal, or edge reversal) to the current graph which most improves the scoring metric. Chickering et al. (1995) shows that this strategy is successful when the initial Bayesian network is close to the true Bayesian network. They suggest that it will not perform well when starting from a Bayesian network with no edges, and that is confirmed by this simulation study.

Greedy Pattern Search algorithm

Most of the heuristic searches described in the literature use the space of DAG's as the search space. The idea for the GBPS is to transform the search space from a space of DAG's to a space of patterns. Given the assumption that our metric has the property of Marko score equivalence this simplification of the search space is reasonable.

```
procedure GBPS(pat; data);
begin
  graph:=pat-to-graph(pat);
  current-score:=score(graph,data);
  max-score:=current-score;
  while max-score <= current-score do
  begin
    new-pat:=add-best-edge-to-pat(pat);
    graph:=pat-to-graph(new-pat);
    current-score:=score(graph,data);
    if current-score > max-score then
    begin
      max-score:=current-score;
      pat:=new-pat;
    end;
  end;
  current-score:=max-score;
  while max-score <= current-score do
  begin
    new-pat:=remove-worst-edge-in-pat(pat);
    graph:=pat-to-graph(new-pat);
    current-score:=score(graph,data);
    if current-score > max-score then
    begin
      max-score:=current-score;
      pat:=new-pat;
    end;
  end;
  return(pat);
end;
```

The algorithm is best described as a greedy pattern search. Given an initial pattern (an empty pattern when running alone or a pattern obtained from another algorithm such as the PC algorithm) the algorithm simply adds edges to the pattern until no furthur addition improve the score and then attempts to remove edges until no furthur removals improve the score. The details of the addition and removal process are described in more detail below.

Although patterns are the elements of the search space, the scoring functions are defined for DAGs. Thus the algorithm needs the function pattern-to-graph which takes a pattern and returns a DAG from the class of graphs represented by the given pattern. Orientation rules for finding a DAG from a pattern can be found in Meek (1995). The resulting DAG and the data are used to obtain the score for the pattern.

The function add-best-edge-to-pattern takes a pattern and returns the pattern with the best score obtainable from the given graph by a single edge addition. When the algorithm considers the addition of an undirected edge between two variables a and b it also consider adding (if possible) unshielded colliders at a or unshielded colliders at b. The algorithm also checks to insure that the resulting set of edges is actually a pattern. For instance, if the current pattern is $a - c - d - b$ then the patterns $b \rightarrow a \leftarrow c - d - b$ and $a - c - d \rightarrow b \leftarrow a$ will be checked when considering the addition of an edge between a and b but the set of edges in the mixed graph $a - c - d - b - a$ is not the pattern for any directed acyclic graph.

The function remove-worst-edge-in-pattern takes a pattern and returns the pattern with the best score obtainable by the removal of one edge from the given pattern. As in case of edge removals there is a combinatorial aspect to the removal edges. If the current pattern is $a - b - c - a$ and we are considering the removal of the edge between a and b then the two patterns which the algorithm will check are $b \rightarrow c \leftarrow a$ and $b - c - a$.

PC + GBPS

The PC algorithm can be used to provide an initial pattern for the GBPS algorithm. The PC algorithm can get close to the correct pattern in a relatively short period of time. This shortens the amount of time that the GBPS algorithm spends on its search.

Simulation Tests

The average degree of the vertices in the graphs considered are 2, 3, or 4; the number of variables is 10; and the sample sizes are 100, 250, 500, 1000, 2500. For each combination of these parameters, 10 DAGs were generated, and a single parameterization of each DAG obtained, and a single sample taken from each parameterized DAG. All pseudo-random numbers were generated by the UNIX "random" utility. Each sample is generated in three stages:

1. The DAG is pseudo-randomly generated.
2. The conditional probabilities (DAG parameters) are pseudo-randomly generated.
3. A sample for the model is pseudo-randomly generated.

It should be emphasized that each of these algorithms has parameters that can be set by the user, and the algorithms were not tested to find optimal parameter settings, although some preliminary tests were done to determine generally what parameter settings were reasonable. PC always used a significance level of 0.0001. (This unusual significance level was selected because it tends to produce sparse DAGs, which are good starting points for the GBPS. Hence this significance level improved the performance of the combined algorithm, but degraded the performance of the PC algorithm. In general, adjusting parameters for the sample size would improve the performance of all of the search algorithms.) The GBPS and GBDS always used a pseudo-sample size of 10 and assigned each Bayesian network an equal prior probability. The greedy Bayesian search algorithm that was tested here was chosen for ease of implementation, and other search algorithms might do somewhat better. See Chickering et al. (1995) and Heckerman et al. (1994a) for simulation tests of several different Bayesian search algorithms.

The results were scored as follows. For each algorithm and each sample we take the patterns output by the algorithm and call the pattern with the highest score the algorithm's *output pattern* for that sample. An *edge existence error of commission* occurs when any pair of variables are adjacent in the output pattern but not in the true pattern. If an edge e between a and b occurs in both the true and output patterns, there is an *edge direction error of commission* when e has an arrowhead at a in the output pattern but not in the true pattern, (and similarly for b.) *Errors of omission* are defined analogously in each case. The results of the simulation are shown in graphs and tables at the end of the article. The *percentage of edge existence errors of commission* is the average over the trial distributions of the ratio of the number of actual edge errors of commission to the number of possible edge errors of commission (i.e. the number of edges in the true pattern) multiplied by 100. The other errors are calculated in an analogous way. For each error type and each algorithm, the percentage of errors is plotted against sample size and average degree.

The numerical results of the simulations are tabulated below; a sense of the major differences in reliabilites of the algorithms can be gleaned from the diagrams of error surfaces. In general, the GBPS and combined search algorithms perform about as well or better than the other two procedures, with two exceptions: at small sample sizes the PC algorithm erroneously adds fewer edges than the other algorithms; and, at small sample sizes the GBDS search erroneously adds fewer arrowheads than other algorithms. At larger sample sizes this advantage of PC over GBPS and the combined search almost vanishes, and at larger sample sizes the GBDS search usually becomes worse at arrowhead addition errors than GBPS or combined search. GBDS performs particularly badly with edge addition errors, which actually increase as sample sizes increase above 500. PC shows similarly bad behavior with increasing sample size for arrowhead addition. Particularly at smaller sample sizes the poor performance of the PC algorithm on edge omission is in part because of the non-optimized small significance level that was used.

Unless confronted with a problem in which only edge addition errors matter, GBPS and the combined search procedure PC + GBPS seem clearly preferable to the others in reliability, and their reliability, at least in this simulation, is remarkably good.

Future work should investigate the reliabilities of the procedures under more optimal parameter adjustments, and the reliabilities of the more promising procedures with normally distributed variables, and with varying degrees of prior knowledge about the target structure. Future work should also consider the predictive accuracy of models selected by these procedures.

We also compared the speed of the combined algorithm and the GBPS algorithm for DAGs with different numbers of variables (10, 15, 20). In each case the combined algorithm was roughly twice as fast as the GBPS algorithm. This is also confirmed by running both of the algorithms on the Alarm data, a DAG with 36 variables. On a sample size of 10000 both the GBPS and the combined algorithm reconstructed the correct pattern for the Alarm network with only one error (the omission of an edge.) This took about 5 hours on a Sparcstation 20 for the GBPS algorithm, and about 2.5 hours for the comgined algorithm. Singh and Valtorta (1993) report similar edge error rates for their combined algorithm for the Alarm network.

Acknowledgements

We would like to thank Clark Glymour, Thomas Richardson and anonymous reviewers for comments on a previous draft of this paper.

Appendix: Keys to table and figure

The following codes are used in Table 1. D is the average degree of the vertices; A is the algorithm (where 1 represents the PC algorithm, 2 represents the combined PC + Bayesian search, and 3 represents GBPS and 4 represents GBDS); Size is the sample size; %emiss is the percentage of edge existence errors of omission; %eadd is the percentage of edge existence errors of comission; %amiss is the percentage of edge direction errors of omission; %aadd is the percentage of edge direction errors of comission.

A	D	Size	%emiss	%eadd	%amiss	%aadd
1	2	100	58.000	1.429	82.353	50.746
2	2	100	28.000	5.714	37.838	17.757
3	2	100	28.000	5.714	38.889	10.185
4	2	100	34.000	6.000	67.647	9.184
1	3	100	60.667	1.333	80.000	51.807
2	3	100	42.000	4.667	44.444	23.333
3	3	100	43.333	4.333	44.231	17.797
4	3	100	40.667	4.667	54.386	15.702
1	4	100	75.500	1.600	95.000	56.410
2	4	100	56.000	5.600	40.426	31.783
3	4	100	58.000	5.200	43.478	28.689
4	4	100	57.000	4.000	52.174	16.667
1	2	250	37.000	0.000	61.538	5.000
2	2	250	17.000	2.000	23.077	6.299
3	2	250	17.000	2.000	23.077	6.299
4	2	250	23.000	2.286	42.857	8.403
1	3	250	58.000	0.000	72.414	22.680
2	3	250	36.667	3.667	38.000	26.429
3	3	250	37.333	3.667	28.571	28.777
4	3	250	35.333	4.000	45.098	19.580
1	4	250	69.000	0.000	58.065	12.903
2	4	250	40.500	1.200	35.211	23.353
3	4	250	42.500	1.200	32.353	23.457
4	4	250	44.000	2.000	62.687	8.280
1	2	1000	14.000	0.000	16.279	6.202
2	2	1000	3.000	0.571	11.321	1.418
3	2	1000	2.000	0.286	5.660	2.797
4	2	1000	6.000	2.000	47.059	5.109
1	3	1000	28.000	0.333	42.647	20.270
2	3	1000	8.667	1.333	11.957	7.692
3	3	1000	10.000	1.667	13.187	8.939
4	3	1000	13.333	5.000	53.488	9.195
1	4	1000	33.500	0.000	59.259	41.081
2	4	1000	20.500	0.800	16.667	12.037
3	4	1000	20.000	0.400	12.745	11.468
4	4	1000	25.000	4.000	37.234	9.223
1	2	2500	7.000	0.286	24.138	13.281
2	2	2500	0.000	0.000	0.000	1.471
3	2	2500	0.000	0.286	1.562	2.941
4	2	2500	1.000	5.714	39.683	5.926
1	3	2500	6.667	0.667	21.569	21.348
2	3	2500	2.667	1.667	11.111	8.696
3	3	2500	0.000	0.000	0.000	5.319
4	3	2500	6.000	7.000	35.922	12.291
1	4	2500	15.500	0.000	54.472	45.581
2	4	2500	7.500	4.800	9.420	9.914
3	4	2500	7.000	1.200	9.353	10.730
4	4	2500	9.500	6.400	36.567	6.140

Table 1: Empirical results

References

Buntine, W. 1991. Theory refinement on Bayesian networks. In *Proc. of the Seventh conf. on Uncertainty in Art. Int.*, 50–62. Morgan Kaufmann.

Chickering, D.; Geiger, D.; and Heckerman, D. 1995. Learning Bayesian networks: Search methods and experimental results. In *Preliminary papers of the fifth int. workshop on Art. Int. and Statistics*, 112–128.

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

Heckerman, D.; Geiger, D.; and Chickering, D. 1994a. Learning Bayesian networks: The combination of knowledge and statistical data. In *Proceedings of the Tenth conference on Uncertainty in Artificial Intelligence*, 293–302. Morgan Kaufmann.

Heckerman, D.; Geiger, D.; and Chickering, D. 1994b. Learning Bayesian networks: The combination of knowledge and statistical data. Technical Report MSR-TR-94-09, Microsoft Research.

Meek, C. 1995. Causal inference and causal explanation with background knowledge. In *Proceedings of Uncertainty in Artificial Intelligence.* To appear.

Singh, M., and Valtorta, M. 1993. An algorithm for the construction of Bayesian network structures data. In *Proc. of the Ninth conf. on Uncertainty in Art. Int.*, 259–265. Morgan Kaufmann.

Spirtes, P.; Glymour, C.; and Scheines, R. 1991. An algorithm for fast recovery of sparce causal graphs. *Social science computer review* 9:62–72.

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

Verma, T., and Pearl, J. 1991. Equivalence and synthesis of causal models. In *Proceedings of the Sixth Conference in Art. Int.*, 220–227. Mountain View, CA: Association for Uncertainty in AI.

Empirical Error Surfaces

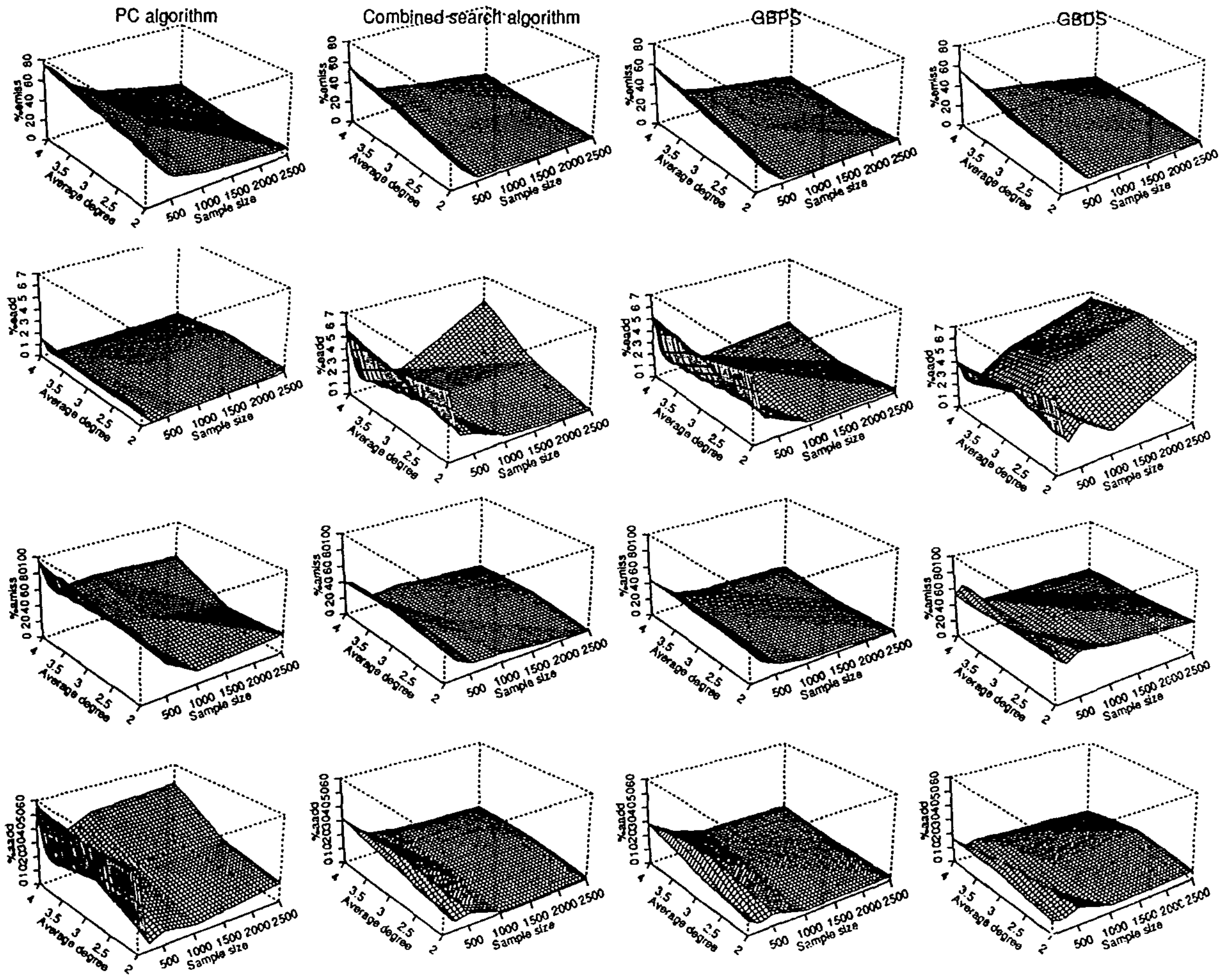

Figure 1: Empirical results