International Journal of Pattern Recognition and Artificial Intelligence
© World Scientific Publishing Company

Learning and Evaluating Bayesian Network Equivalence Classes from Incomplete Data

Hanen Borchani, Nahla Ben Amor and Fédia Khalfallah

LARODEC, Institut Supérieur de Gestion de Tunis, 41 Avenue de la liberté, 2000 Le Bardo, Tunisie.

hanene.borchani@gmail.com, nahla.benamor@gmx.fr, fedia.khalfallah@fsegt.rnu.tn

In this paper, we propose a new method, named Greedy Equivalence Search-Expectation Maximization (GES-EM), for learning Bayesian networks from incomplete data. Our method extends the recently proposed Greedy Equivalence Search (GES) algorithm[10] to deal with incomplete data. For the quality evaluation of learned networks, we make use of the expected Bayesian Information Criterion (BIC) scoring function. In addition, we propose a new structural evaluation criterion. This so-called SEC criterion is more suitable than existing structural evaluation criteria, since it is based on the comparison of learned networks to the generating ones through Completed Partially Directed Acyclic Graphs ($CPDAGs$). Experimental results show that GES-EM algorithm yields more accurate structures than the standard Alternating Model Selection-Expectation Maximization (AMS-EM) algorithm[15].

Keywords: Bayesian networks; Bayesian network equivalence classes; structure learning; incomplete data; structural evaluation.

1. Introduction

Bayesian networks, also known as belief networks, are becoming a popular tool for representing uncertainty in artificial intelligence[19,26]. They have been implemented in several real applications in different areas such as medical diagnosis, pattern recognition, credit assessment and fraud detection.

Learning Bayesian networks is an unsupervised learning problem, which aim to discover dependency structure, or set of structures, between the variables of interest. This task can be achieved via eliciting knowledge from experts. Probably, this may works well for smaller networks, but when the number of variables becomes large, elicitation becomes a tedious and expensive process. Furthermore, there may be situations where the expert is either unwilling or unavailable.

Hence, several researches and very important results appeared in the field of learning Bayesian networks from data, and especially learning the graph structure from samples[5,10,11,15,17,21,24,30].

Most learning methods, to date, are based on the assumption that the data are complete, that is, the values of all variables are observed for each record in the

data[5,10,11,17,20]. Unfortunately, this assumption is unrealistic since most real-world data rarely conform to the ideal of being complete and quite often involve missing information.

The source of the missingness may come from noisy measurements, equipment malfunction, not entered data due to misunderstanding or from domains in which some variables are never observed. To deal with this problem, some approaches proceed either by deleting records with missing values or using ad-hoc methods to fill in the missing data. Such procedures may however lead to biased or poor networks and therefore erroneous analysis.

Current few state-of-the-art approaches, dealing with learning Bayesian networks from incomplete data, avoid such ad-hoc methods and, using any sound inference algorithm, take into account available observed data in order to estimate the missing values[14,15,24,30,32]. Based on these estimations, they next search for the best network.

These approaches make use of search algorithms operating either in the *Bayesian network search space* (*B-space*) or the *tree search space* that presents the drawback of being a subspace less rich than the *B-space*. Recently, an alternative *equivalence class search space* (*E-space*) was proposed by Chickering[10] which proved, in complete data case, that this space presents better advantages than *B-space*.

This paper proposes a new approach for learning Bayesian networks from incomplete data that takes advantage of the *E-space* via extending the Greedy Equivalence Search (GES) algorithm, proposed in[10], to deal with incomplete data. In order to estimate missing values, we adopt the most commonly used Expectation Maximization (EM) algorithm[12]. Thus, this novel extension of GES, which we call GES-EM algorithm, deals with model selection as well as parameter estimation from incomplete data.

For the quality evaluation of learned networks, we make use of the expected Bayesian Information Criterion (BIC) scoring function that measures how well the networks fit the data. In addition, we propose a new structural evaluation criterion (SEC) based on the comparison of learned networks to the generating ones through Completed Partially Directed Acyclic Graphs ($CPDAGs$). This criterion is simple to compute, it avoids the incoherencies of existing structural evaluation criteria, and allows therefore a more accurate comparison between structures.

This paper is organized as follows: Section 2 recalls basics of Bayesian networks and briefly reviews existing learning methods both from complete and incomplete data. Section 3 presents our new approach for learning Bayesian network equivalence classes from incomplete data. Section 4 proposes a new structural evaluation criterion based on $CPDAGs$. Finally, Section 5 describes experimental study comparing the proposed GES-EM algorithm and the standard AMS-EM algorithm proposed by Friedman[15].

2. Background

This section first provides some useful notations and basic concepts relevant to Bayesian networks. Then, it briefly reviews existing learning methods, and it ends with a recall of existing evaluation criteria.

2.1. *Bayesian networks*

The following syntactical conventions are used: $\mathbf{U} = \{X_1, ..., X_n\}$ denotes the universe defined as a finite set of n discrete random variables. A variable is denoted by an upper case letter (e.g. X, Y, X_i) and a state or value of that variable by the same lower-case letter (e.g. x, y, x_i). A set of variables is denoted by a bold-face capitalized letter (e.g. $\mathbf{X}$, $\mathbf{Y}$, $\mathbf{Pa}(X_i)$) and the corresponding bold-face lower-case letter (e.g. $\mathbf{x}$, $\mathbf{y}$, $\mathbf{pa}(X_i)$) denotes an assignment or state for each variable in a given set. Calligraphic letters (e.g. $\mathcal{B}$, $\mathcal{G}$, $\mathcal{E}$) denotes statistical models.

A Bayesian network $\mathcal{B}$ is a pair $(\mathcal{G}, \mathbf{\Theta})$. The first component is the structure $\mathcal{G}$ which is a directed acyclic graph (DAG), where the set of vertices or nodes represents the variables, and the set of directed edges or arcs corresponds to dependence relationships between these variables.

The second component, namely the set of parameters $\mathbf{\Theta} = \{\mathbf{\Theta_1}, \mathbf{\Theta_2}, ..., \mathbf{\Theta_n}\}$, specifies all the conditional probability distributions. $\mathbf{\Theta_i} = P(X_i \mid \mathbf{Pa}(X_i))$ denotes the conditional probability distribution of each node X_i given its parents, denoted by $\mathbf{Pa}(X_i)$.

Let r_i be the number of distinct states of X_i and q_i be the number of different possible configurations of $\mathbf{Pa}(X_i)$, then $\mathbf{\Theta_i} = \{\theta_{ijk}, 1 \leq j \leq q_i, 1 \leq k \leq r_i\}$ such that $\theta_{ijk} = P(X_i = k \mid \mathbf{Pa}(X_i) = j)$ denotes the conditional probability that the ith variable is in state k, given its parents are in the jth configuration.

A Bayesian network $\mathcal{B}$ encodes a unique joint probability distribution over $\mathbf{U}$ expressed by:

$$P(X_1, ..., X_n) = \prod_{i=1}^{n} P(X_i \mid \mathbf{Pa}(X_i)) \,. \tag{1}$$

Two Bayesian network structures $\mathcal{G}$ and $\mathcal{G}'$ are said to be equivalent, denoted by $\mathcal{G} \approx \mathcal{G}'$, if they can be used to represent the same set of probability distributions. More formally, Chickering provides the following definition[7]:

Definition 1. *Two DAGs $\mathcal{G}$ and $\mathcal{G}'$ are equivalent if for every Bayesian network $\mathcal{B} = (\mathcal{G}, \mathbf{\Theta})$, there exists a Bayesian network $\mathcal{B}' = (\mathcal{G}', \mathbf{\Theta}')$ such that $\mathcal{B}$ and $\mathcal{B}'$ define the same probability distribution, and vice versa.*

Moreover, Verma and Pearl[33] derive the following theorem which determines graphically the equivalence of two $DAGs$. This theorem uses the notion of *skeleton* and *v-structure* explained in what follows:

Theorem 1. *Two DAGs are equivalent if and only if they have the same skeletons and the same v-structures.*

Definition 2. *The skeleton of a DAG $\mathcal{G}$ is the undirected graph resulting from ignoring the directionality of every edge.*

Definition 3. *A v-structure in a DAG $\mathcal{G}$ is an ordered triple of nodes (X, Y, Z), such that $\mathcal{G}$ contains the directed edges $X \rightarrow Y$ and $Z \rightarrow Y$, where X and Z are not adjacent.*

Example 1. *Let us consider the four DAGs $\mathcal{G}^1$, $\mathcal{G}^2$, $\mathcal{G}^3$ and $\mathcal{G}^4$ given in Figure 1. The four DAGs have the same skeletons and the same v-structure in Z. Moreover $\mathcal{G}^4$ has an other v-structure in Y which is not the case for the remaining DAGs, which have not any other v-structures. This means that within the four DAGs only $\mathcal{G}^1$, $\mathcal{G}^2$ and $\mathcal{G}^3$ are equivalent.*

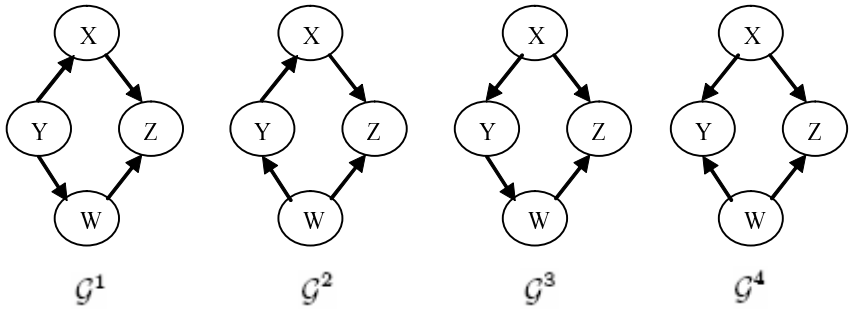

Fig. 1. Example of *DAGs*

Each set of equivalent *DAGs* defines an equivalence class of Bayesian networks, denoted by $\mathcal{E}$, which can be represented via a unique *Completed Partially Directed Graph* (*CPDAG*), denoted by $\mathcal{P}^c$[10]. This kind of representation is also called *essential graph* by Andersson et al.[3] or *completed pattern* by Verma and Pearl[33].

A Partially Directed Acyclic Graph (*PDAG*) is a graph that contains both directed and undirected edges. A *compelled* edge is an edge that exists with the same orientation for every *DAG* member of an equivalence class. If an edge is not compelled, then it is *reversible*. Thus, a *CPDAG* $\mathcal{P}^c$ corresponding to an equivalence class $\mathcal{E}$ is a *PDAG* consisting of a directed edge for every compelled edge in the equivalence class, and an undirected edge for every reversible edge in the equivalence class[10].

Every *DAG* $\mathcal{G}$ in an equivalence class $\mathcal{E}$ is a *consistent extension* of the *CPDAG* representation, $\mathcal{P}^c$, for that class, i.e. $\mathcal{G}$ has exactly the same skeleton and the same set of v-structures as $\mathcal{P}^c$, and every directed edge in $\mathcal{P}^c$ has the same orientation in $\mathcal{G}$[10].

2.2. *Learning Bayesian networks from data*

The problem of learning Bayesian networks from data can be defined as follows: Given a data $\mathbf{D} = \{\mathbf{d}_1, \mathbf{d}_2, ..., \mathbf{d}_N\}$ of N records, find a Bayesian network $\mathcal{B}$ that best fits $\mathbf{D}$[15]. Since this problem has been shown to be NP-hard[8], most commonly used approaches resort to heuristic search varying on the formulation of the search space, the scoring function used to assess the goodness-of-fit of particular states in the defined space, and the search algorithm used to identify one or several high-scoring states.

The formulation of the search space requires the definition of three components[10], namely, the set of states, the representation scheme for the states and the set of operators. The set of states represents the logical set of solutions to the search problem (e.g. the set of all Bayesian networks defined over the same set of variables), the representation scheme defines an efficient way to represent the states (e.g. DAG), and the set of operators is used by the search algorithm to transform the representation of one state to another in order to traverse the space in a systematic way[10]. Learning algorithms depend on the nature of the data. Indeed, we can distinguish two cases:

2.2.1. *Learning from complete data*

The most prevalent approach in the literature on learning Bayesian network structures from complete data is to perform, in conjunction with some decomposable scoring function, a Greedy Search (GS) algorithm to the space of Bayesian networks, known as *B-space*[11].

One of the commonly used scoring functions is the Bayesian Information Criterion (BIC) score[29] expressed by:

$$BIC(\mathcal{G} : \mathbf{D}) = logP(\mathbf{D} \mid \mathcal{G}) - \frac{logN}{2} Dim(\mathcal{G}) \,. \tag{2}$$

where $Dim(\mathcal{G})$ is the number of parameters in the network. The first term, $logP(\mathbf{D} \mid \mathcal{G})$, known as the log likelihood function, measures how well the structure predicts the data. The second term, $\frac{logN}{2} Dim(\mathcal{G})$, is a penalty term aiming to avoid overfitting and prefer simpler networks.

The BIC score is *decomposable*, i.e. it can be written as a sum of local scores relative to each node X_i and its parents expressed by:

$$BIC_i(X_i, \mathbf{Pa}(X_i) : \mathbf{D}) = \sum_{k=1}^{r_i} \sum_{j=1}^{q_i} N_{ijk} log\theta_{ijk} - \frac{logN}{2} Dim(X_i, \mathbf{Pa}(X_i)) \,. \tag{3}$$

with N_{ijk} are the *sufficient statistics* of $\mathbf{x}_i$ in $\mathbf{D}$, that is, the number of instances the event $(X_i = k \mid \mathbf{Pa}(X_i) = j)$ was observed in $\mathbf{D}$, and $Dim(X_i, \mathbf{Pa}(X_i))$ is the number of parameters required to represent $P(X_i \mid \mathbf{Pa}(X_i))$.

This decomposability property and the use of sufficient statistics have significant impact on the computational efficiency of the GS algorithm, as well as alternative

search algorithms for learning Bayesian networks from complete data. Indeed, the merit of each local change is easily deduced via computing only the local scores for nodes whose parent set has changed and leaving the others unaltered.

Thus, GS algorithm performs, at each iteration, the local change yielding the maximal global score, until it reaches a local maximum. This problem of local maxima is addressed, typically, by performing GS with random restarts[17].

Moreover, there are additional problems when the GS algorithm traverses through the *B-space*, especially due to the fact that this space allows equivalent DAG representations for the same network[2]. Indeed, the application of many operators to a given current network will result in equivalent networks which may be also equivalent to the current one. Consequently, the GS algorithm can waste time re-scoring many equivalent networks. In addition, the choice of the next network can be arbitrary, since most commonly used scoring functions are shown to be score-equivalent[7], that is, they assign the same score to equivalent networks, and this can therefore affects considerably the quality and accuracy of final network returned by the GS algorithm.

To overcome these problems, several works have proposed to perform the search over an alternative search space, namely the *equivalence class search space* (*E-space*)[2,9,10,23]. Notably, Chickering[10] has made a significant contribution by providing an interesting *E-space* implementation where the states are equivalence classes of Bayesian networks, represented via $CPDAGs$. The applied operators in this space are the *Insert* and *Delete* ones. Chickering[10] has provided validity conditions for each operator and proved that they can be scored efficiently using any decomposable score-equivalent scoring function.

A two phase greedy search algorithm, called *Greedy Equivalence Search* (GES)[10,23], is used to search over the defined *E-space*. GES starts with an equivalence class corresponding to no dependencies. In the first phase, it considers all valid *Insert* operators that can be applied to the current state, evaluates each resulting neighbor, and moves to the highest-scoring one if doing so improves the score. Once a local maximum is reached, a second phase proceeds from the previous local maximum using *Delete* operators. When the second phase reaches a local maximum, that equivalence class is returned as solution[10]. Chickering has proved that GES algorithm leads to better solutions than the traditional GS algorithm, and identifies optimal structures with finite sample sizes[10].

2.2.2. *Learning from incomplete data*

When the data are incomplete, that is, some values of some variables are missing, or some variables are never observed, known as *hidden* or *latent* variables, the learning problem becomes much harder. One difficulty is that the scoring function cannot be computed efficiently since we do not have direct access to sufficient statistics. Moreover, a local change in one part of the network affects the evaluation of a change in another part of the network. Thus, at each iteration, all neighbors of the

current network should be entirely re-evaluated[15].

Few approaches have been proposed to cope with this problem. The most prominent one is the *Structural Expectation Maximization* (SEM) algorithm[15], which ensures learning Bayesian networks from incomplete data via combining the standard *Expectation Maximization* (EM) algorithm[12] for parameter optimization with the GS algorithm for model selection[11].

The EM algorithm iteratively optimizes the current set of parameters from incomplete data using an *Expectation* step (E-step) followed by a *Maximization* one (M-step). The E-step computes, using an inference engine, the expected sufficient statistics according to the observed values in $\mathbf{D}$ and the current set of parameters. Next, the M-step takes the expected sufficient statistics as though they were the actual ones and determine the new optimal estimate of the parameters. The algorithm iterates until convergence[12].

The SEM algorithm extends the idea of EM algorithm to learn both the structure and parameters from incomplete data. It begins with a randomly initialized network. At each iteration, it first determines the neighbors of the current state and then computes their corresponding expected scores.

Let $\mathbf{O}$ denotes the set of observable variables and $\mathbf{M} = \mathbf{U} - \mathbf{O}$ denotes the set of variables partially or completely unobserved. The expected score of any neighbor $\mathcal{G}^i$, denoted by the auxiliary function Q, is deduced by taking expectation over all possible values of $\mathbf{m}$ given the observed variables $\mathbf{O}$ and the current state $(\mathcal{G}, \mathbf{\Theta})$[15]:

$$Q(\mathcal{G}^i, \mathbf{\Theta^i} : \mathcal{G}, \mathbf{\Theta}) = E_{P(\mathbf{m}|\mathbf{O}:\mathcal{G},\mathbf{\Theta})}[logP(\mathbf{O}, \mathbf{m} \mid \mathcal{G}^i, \mathbf{\Theta^i})] - \frac{logN}{2} Dim(\mathcal{G}^i)\,. \qquad (4)$$

where $E_{P(\mathbf{m}|\mathbf{O}:\mathcal{G},\mathbf{\Theta})}[logP(\mathbf{O}, \mathbf{m} \mid \mathcal{G}^i, \mathbf{\Theta^i})]$ is the *expected log likelihood*. Friedman[15] showed that the expected BIC score has an analogue decomposition as the actual one expressed by (3). The only difference is that it uses expected sufficient statistics instead of the actual ones.

Next, SEM algorithm selects the neighbor which has the highest expected score. This process iterates and converges to a local maximum when there is no further improvement in the expected score.

To avoid early convergence to undesirable local maxima, Friedman[15] introduced a variant of SEM algorithm, called the *Alternating Model Selection-Expectation Maximization* (AMS-EM) algorithm, which alternates between iterations that optimize the parameters for the current network, and iterations that search for different network neighbors.

Other approaches, based on the same idea of interleaving structure search with EM-like iteration, have been also proposed. Singh[30], as SEM algorithm, focuses on learning a single *DAG* model from incomplete data. Whereas, Meila and Jordan[24] consider the problem of learning mixtures models, where each mixture component is a spanning tree. Thiesson et al.[32] method selects also among mixtures, but, where each mixture component is a *DAG* model. François and Leray[14] propose two variants of AMS-EM algorithm in the space of trees.

2.3. *Existing structural evaluation criteria*

As stated above, the goal of GES-EM algorithm is to identify a single network with the highest expected BIC score. Consequently, the expected score of the best found state detects how well this state fits the data and represents a good indicator of the algorithm's performance.

Furthermore, in many circumstances, learning accuracy is measured through comparing the learned Bayesian network to a reference. Nevertheless, when manipulating real data, it is difficult to compare the obtained results to a certain reference because usually, we do not have any reference. We should only refer to the decision and experience of experts, which is not enough to decide whether such a learning algorithm is good and reliable or not.

Therefore, in order to test learning algorithms, one may apply them to several synthetic databases generated from random structures. Then, the learned structures (LS in Figure 2) are compared to the initial ones (IS in Figure 2) using any structural evaluation criterion.

For instance, if we want to evaluate the performance of the GES-EM algorithm and compare it with any other algorithm for learning Bayesian networks from incomplete data, we should apply both of them on the same generated database and compare the learned structures as shown by Figure 2.

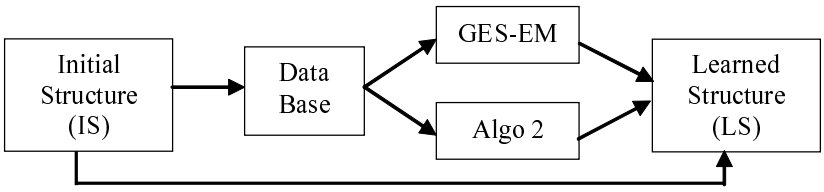

Fig. 2. Comparison of IS and LS

Aliferis and Cooper[1] proposed qualitative criteria in order to compare two structures represented via two $DAGs$. They suggest to compare the initial structure (IS) to the learned one (LS) link by link. Indeed, for each (IS, LS) pair, they compute two metrics:

- M1: percentage of arcs in IS that are present in LS (percentage of correctly-found arcs). This criterion shows the degree of accuracy of LS.
- M2: percentage of arcs in LS that are absent in IS, that is, percentage of superfluous arcs.

We may have two additional criteria[20]:

- A^-: percentage of arcs in IS that are absent in LS (percentage of missing arcs). This criterion shows the degree of incompleteness.

- A˜: percentage of arcs existing in both LS and IS but oriented in opposite directions (percentage of badly-oriented arcs). This criterion shows the degree of ambiguity between both structures.

A new criterion overcoming limitation of these criteria will be proposed in Section 4.

3. Greedy equivalence search-expectation maximization algorithm

As mentioned in Section 2.2, the state-of-the-art approaches dealing with Bayesian network learning from incomplete data make use of search algorithms operating either in the Bayesian network search space (*B-space*)[15,30,32] or the tree search space[14,24].

Although, searching over the tree search space fades away exponential computation demands and leads to fast solutions, it can unfortunately return bad quality networks, since it is a subspace less rich than the B-space.

In the other hand, the *B-space* is super exponential. Indeed, as shown by Robinson in[28], the number of different Bayesian network structures having n nodes is $n^{2^{O(n)}}$.

Moreover, the *B-space* allows redundant or multiple representations for the same state. Arguably, this causes possibly disruptions during applying search algorithms (e.g. greedy search), due to arbitrary choices of the next state, and affects, therefore, the accuracy of the resulting network.

Thus, in order to learn more accurate networks from incomplete data, we propose a new approach that takes advantage of the *E-space* proposed by[10] as well as the GES algorithm[10,23]. The main idea of the proposed algorithm, named Greedy Equivalence Search Expectation Maximization (GES-EM), is to alternate between EM iterations, which optimize the parameters for the current equivalence class, and structure search iterations, which aim to identify the possible neighbors of the current equivalence class using the GES algorithm.

The GES-EM algorithm makes use of the *E-space* definition provided by Chickering in[10]. That is, it moves among equivalence classes of Bayesian networks, represented via $CPDAGs$, and uses the same *Insert* and *Delete* operators.

Nonetheless, as model selection criterion, we will use the expected BIC scoring function expressed by (4). As stated earlier, this expected scoring function is decomposable with respect to the *B-space*. In our context, since we have changed the search space, we should, as well, verify that this function assigns the same expected scores to structures in the same equivalence class.

Let $\mathcal{G}^i$ and $\mathcal{G}^j$ be two equivalent $DAGs$ belonging to the same equivalence class. According to the expected BIC formula given by (4), it is clear that the expected log likelihood of the data given the structure $\mathcal{G}^i$ is equal to the one computed given the structure $\mathcal{G}^j$, since they were computed with respect to the same estimated

probability $P(\mathbf{m} \mid \mathbf{O} : \mathcal{G}, \mathbf{\Theta})$. More formally:

$$E_{P(\mathbf{m}|\mathbf{O}:\mathcal{G},\mathbf{\Theta})}[logP(\mathbf{m}, \mathbf{O} \mid \mathcal{G}^i, \mathbf{\Theta^i}))] = E_{P(\mathbf{m}|\mathbf{O}:\mathcal{G},\mathbf{\Theta})}[logP(\mathbf{m}, \mathbf{O} \mid \mathcal{G}^j, \mathbf{\Theta^j}))] \,. \quad (5)$$

In addition, Chickering has proved that Bayesian networks with equivalent structures require the same number of parameters[7], i.e. $Dim(\mathcal{G}^i) = Dim(\mathcal{G}^j)$. This means that equivalent $DAGs$ belonging to the same equivalence class have the same expected scores. More formally:

Proposition 1. *Let $\mathcal{E}$ be an equivalence class, then $\forall\ \mathcal{G}^i, \mathcal{G}^j \in \mathcal{E}$ s.t. $i \neq j$,*

$$Q(\mathcal{G}^i, \mathbf{\Theta^i} : \mathcal{G}, \mathbf{\Theta}) = Q(\mathcal{G}^j, \mathbf{\Theta^j} : \mathcal{G}, \mathbf{\Theta}) \,. \quad (6)$$

From Proposition 1, it follows that all $DAGs$ contained within the same equivalence class $\mathcal{E}$ have the same expected BIC scores. Therefore, the equivalence class $\mathcal{E}$ can be evaluated using any DAG member.

Roughly speaking, the GES-EM algorithm starts with an initial equivalence class $\mathcal{E}^0$, containing a unique empty Bayesian network represented via the structure $\mathcal{G}^0$, and the randomly initialized parameter set $\mathbf{\Theta}^0$, then, it executes a single edge insertion phase followed by a single edge deletion one.

Single edge insertion phase. The first phase, outlined by Algorithm 1, starts by running EM algorithm to convergence, using the current state ($\mathcal{G}^0$,$\mathbf{\Theta}^0$), in order to find improved parameter values $\mathbf{\Theta}'^0$ for the initial structure $\mathcal{G}^0$. Next, it applies all valid $Insert$ operations to $\mathcal{E}^0$ in order to obtain the set of its neighbors. This set is denoted by $\mathbf{P}^+(\mathcal{E}^0)$, since each resulting neighbor, represented by a $PDAG$ $\mathcal{P}$, is not necessarily completed.

Therefore, we should convert each neighbor $\mathcal{P} \in \mathbf{P}^+(\mathcal{E}^0)$ to its potential corresponding consistent extension $\mathcal{G}$ by applying the conversion procedure *PDAG-to-DAG* proposed by Dor and Tarsi in[13].

This conversion, first, allows us to check the validity of the applied $Insert$ operations. Indeed, if a resulting $PDAG$ from an $Insert$ operation admits a consistent extension, we say that the operator is valid. Otherwise, the operation is not valid and we do not allow its application to the search space.

In addition, using Proposition 1, the expected BIC score can only be computed for consistent extension relative to each equivalence class in order to evaluate it. This computation, first, needs an inference phase in the current structure in order to estimate the current probability distribution on the missing data.

If the higher obtained expected score is lower than the one found in the previous iteration, then a local maximum is reached and the second phase starts from the previous best state. Otherwise, the best equivalence class is generated, via converting the best selected consistent extension $\mathcal{G}$ to its corresponding $CPDAG$ representation. This is performed via the conversion algorithm *DAG-to-CPDAG* as described by Chickering in[9].

Algorithm 1. Single edge insertion phase

```
begin
    i ← 0;  Convergence ← false;  Stop ← false;
    while (not Convergence) and (not Stop) do
        Run EM, using G^i and θ^i, producing θ'^i;
        Generate P^+(E^i) via applying all valid Insert operators to E^i;
        G^+(E^i) ← PDAG-to-DAG(P^+(E^i));
        if G^+(E^i) ≠ ∅ then
            Compute Q(G, θ : G^i, θ'^i) for each G ∈ G^+(E^i) using (4);
            (G^{i+1}, θ^{i+1}) ← argmax Q(G, θ : G^i, θ'^i);
            if Q(G^{i+1}, θ^{i+1} : G^i, θ'^i) > Q(G^i, θ'^i : G^i, θ'^i) then
                E^{i+1} ← DAG-to-CPDAG(G^{i+1});
                i ← i + 1;
            else
                Convergence ← true;
        else
            Stop ← true;
    end
```

The resulting equivalence class becomes the current state and the search pursue until no consistent extension is generated or a local maximum is reached.

Single edge deletion phase. The second phase of GES-EM algorithm is quite analogous to the first one, except that it replaces *Insert* operations by *Delete* ones, in order to determine the neighbors of the current equivalence class.

To show that our algorithm is useful, we need to prove that it, indeed, improves the real score in each iteration. This result can be obtained using the following theorem given by Friedman[15]:

Theorem 2. *If* $Q(\mathcal{G}^i, \mathbf{\Theta^i} : \mathcal{G}, \mathbf{\Theta}) > Q(\mathcal{G}, \mathbf{\Theta} : \mathcal{G}, \mathbf{\Theta})$*, then:*

$$Score(\mathcal{G}^i, \mathbf{\Theta^i}) > Score(\mathcal{G}, \mathbf{\Theta}).$$

Indeed, the GES-EM algorithm selects at each iteration, during the first and the second phase, a state $(\mathcal{G}^{i+1}, \mathbf{\Theta^{i+1}})$ having a higher expected score than the previous one. This means that, at each iteration, it improves the real score until it converges.

4. A new structural evaluation criterion based on *CPDAGs*: SEC

Most of existing structural evaluation criteria (see Section 2.3) are based on comparing the *DAG* representations of the initial and learned networks. These criteria

can give a general evaluation. Nevertheless, they present many deficiencies and do not always provide accurate evaluation.

Example 2. *Let us consider the four structures given by Figure 3.* $\mathcal{G}^1$ *is the initial structure from which a synthetic data* $\mathbf{D}$ *is randomly generated.* $\mathcal{G}^2$, $\mathcal{G}^3$ *and* $\mathcal{G}^4$ *are the learned structures issued from three different learning algorithms applied to* $\mathbf{D}$. *So, in order to compare these algorithms, the structural evaluation criteria M1, M2,* A^- *and* $A^\sim$ *are computed as shown in Table 1.*

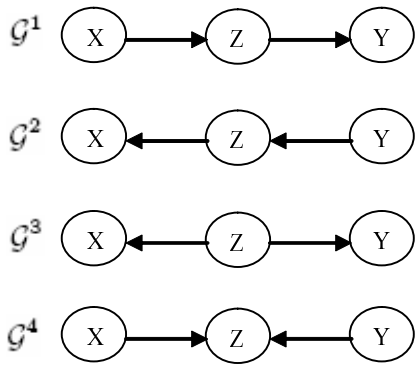

Fig. 3. Example of *DAGs*

Table 1. Structural evaluation using M1, M2, A^- and $A^\sim$ criteria

	M1	M2	A^-	$A^\sim$
$(\mathcal{G}^1,\ \mathcal{G}^2)$	0%	0%	0%	100%
$(\mathcal{G}^1,\ \mathcal{G}^3)$	50%	0%	0%	50%
$(\mathcal{G}^1,\ \mathcal{G}^4)$	50%	0%	0%	50%

According to Table 1, we notice that the M1 and $A^\sim$ *percentages computed for the pair* $(\mathcal{G}^1,\ \mathcal{G}^2)$ *are worse than those computed for* $(\mathcal{G}^1,\ \mathcal{G}^4)$. *This means that the learned structure* $\mathcal{G}^4$ *is better than* $\mathcal{G}^2$. *Nevertheless, this is not the case since* $\mathcal{G}^1$ *is strictly equivalent to* $\mathcal{G}^2$ *but not to* $\mathcal{G}^4$. *All considered criteria are unable to detect these subtle differences and provide consequently wrong comparison results.*

Moreover, the M1, M2, A^- *and* $A^\sim$ *percentages are exactly the same for both pairs* $(\mathcal{G}^1,\ \mathcal{G}^3)$ *and* $(\mathcal{G}^1,\ \mathcal{G}^4)$. *Once again these criteria do not provide reliable results since we may prefer* $\mathcal{G}^4$ *to* $\mathcal{G}^3$ *structure, which is a bad choice because in reality we have* $\mathcal{G}^3$ *is strictly equivalent to* $\mathcal{G}^1$.

To avoid mentioned incoherences, we propose a new structural evaluation criterion (SEC) that allows a more accurate comparison between Bayesian network

structures, and thereby between learning algorithms. Contrary to existing criteria, our idea is to compare $CPDAG$ representations instead of the DAG ones. In other terms, comparing two structures $\mathcal{G}$ and $\mathcal{G}'$ becomes comparing their corresponding $CPDAGs$ $\mathcal{P}^c$ and $\mathcal{P}^{c'}$.

To this end, we can distinguish for each couple of nodes (X, Y) four possible cases summarized by Table 2.

Table 2. Four possible cases of evaluation

$X \quad Y$	no arc between X and Y
$X \rightarrow Y$	one arc from X to Y
$X \leftarrow Y$	one arc from Y to X
$X - Y$	a non oriented arc between X and Y

According to these cases, we propose an error classification scale (a, b, c) such that $0 < a < b < c$. This scale will be used to penalize different types of errors as shown by Table 3. In fact,

Table 3. Different types of possible errors

Error type	**Penality**
No errors	0
An arc is replaced by a non-oriented arc or a non oriented arc is added	a
A new arc is created: risk of adding a dependence or an independence	b
The orientation of a compelled arc is changed so a new independence and a new dependence are added	c

- If the link between two variables is exactly the same in both structures, there is no error.
- If in the first structure there is a non-oriented arc, then whatever its state in the second one, it will be regarded as an error of type a (not a very serious error: this reversible edge has either been added (absent in the second structure), or was existing and the orientation is not important).
- If an arc does not exist in one structure and does in the other one, then whatever its direction, the estimation of the error that has been committed is b (more serious than the preceding one). Indeed, this situation shows that for each arc that does not exist, there corresponds a compelled arc which must not only change orientation but also absolutely exist in the structure.
- If one of the arcs is oriented toward a certain direction and the correspond-

ing arc in the other structure is oriented toward the opposite direction, then this is the most critical situation since a compelled edge has been changed by another compelled one. The error that has been committed is of value c since this kind of differences should be strongly penalized. In fact, this case can generate both additions and removals of conditional independencies at the same time.

In order to compare two $CPDAGs$ $\mathcal{P}^c$ and $\mathcal{P}^{c'}$, we should analyze the eventual link existing between each couple of nodes. Table 4 summarizes all possible cases which can occur for any couple (X, Y).

Table 4. Possible cases of comparison relative to the couple (X, Y)

	$\mathcal{P}^{\mathbf{c}}$			
$\mathcal{P}^{\mathbf{c}'}$	$X \quad Y$	$X \rightarrow Y$	$X \leftarrow Y$	$X - Y$
$X \quad Y$	0	b	b	a
$X \rightarrow Y$	b	0	c	a
$X \leftarrow Y$	b	c	0	a
$X - Y$	a	a	a	0

Then, it is possible to define an error matrix denoted by $EM(\mathcal{P}^c, \mathcal{P}^{c'})$ relative to all the nodes. This matrix is symmetric with respect to its diagonal, since the penalization relative to (X, Y) is equal to the one relative to (Y, X).

Once the error matrix is defined, we can compute a unique value relative to the global error, denoted by $GE(\mathcal{P}^c, \mathcal{P}^{c'})$, as follows:

$$GE(\mathcal{P}^c, \mathcal{P}^{c'}) = \sum_{i=1}^{n-1} \sum_{j-i+1}^{n} EM(\mathcal{P}^c, \mathcal{P}^{c'})(i, j)\,. \tag{7}$$

Note that only the half of the matrix is considered due to its symmetry. $GE(\mathcal{P}^c, \mathcal{P}^{c'})$ belongs to the interval [0, MaxE] where MaxE represents the maximum error value expressed by:

$$MaxE = c \cdot m + b((\frac{n \cdot (n-1)}{2}) - m)\,. \tag{8}$$

where n is the number of nodes, and m is the number of arcs in the initial structure.

Thus, in order to obtain values between 0 and 1, we carry out a normalization of the global error by dividing it by the maximum error value as follows:

$$SEC(\mathcal{P}^c, \mathcal{P}^{c'}) = \frac{GE(\mathcal{P}^c, \mathcal{P}^{c'})}{MaxE}\,. \tag{9}$$

We can then compare all pairs of structures and generate an average indicating the rate of representativeness of learned structures compared with initial ones. The closer one gets to 0, the more equivalent the structures will be.

Example 3. *Let us consider the CPDAGs* $\mathcal{P}^{c1} = \mathcal{P}^{c2} = \mathcal{P}^{c3}$ *and* $\mathcal{P}^{c4}$ *given in Figure 4 and relative to the DAGs* $\mathcal{G}^1$, $\mathcal{G}^2$, $\mathcal{G}^3$ *and* $\mathcal{G}^4$ *given in Figure 3.*

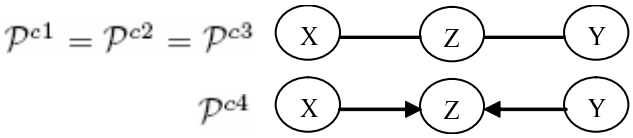

Fig. 4. Example of $CPDAGs$

Using the error classification scale ($a = 1$, $b = 2$, $c = 3$)*, we can establish the error matrices* $EM(\mathcal{P}^{c1}, \mathcal{P}^{c2})$, $EM(\mathcal{P}^{c1}, \mathcal{P}^{c3})$ *and* $EM(\mathcal{P}^{c1}, \mathcal{P}^{c4})$ *as shown in Table 5.*

Table 5. The error matrices $EM(\mathcal{P}^{c1}, \mathcal{P}^{c2})$, $EM(\mathcal{P}^{c1}, \mathcal{P}^{c3})$ and $EM(\mathcal{P}^{c1}, \mathcal{P}^{c4})$

$EM(\mathcal{P}^{c1}, \mathcal{P}^{c2})$	X	Y
Z	0	0
X		0

$EM(\mathcal{P}^{c1}, \mathcal{P}^{c3})$	X	Y
Z	0	0
X		0

$EM(\mathcal{P}^{c1}, \mathcal{P}^{c4})$	X	Y
Z	1	1
X		0

According to these matrices, we have $GE(\mathcal{P}^{c1}, \mathcal{P}^{c2}) = 0$, $GE(\mathcal{P}^{c1}, \mathcal{P}^{c3}) = 0$, $GE(\mathcal{P}^{c1}, \mathcal{P}^{c4}) = 2$, *and* $MaxE = 8$. *Then, SEC values are easily deduced as shown in Table 6.*

Table 6. Structural evaluation using the SEC criterion

	SEC
$(\mathcal{G}^1,\ \mathcal{G}^2)$	0
$(\mathcal{G}^1,\ \mathcal{G}^3)$	0
$(\mathcal{G}^1,\ \mathcal{G}^4)$	0.25

According to the SEC criterion, we notice that the structure $\mathcal{G}^1$ *is strictly equivalent to structures* $\mathcal{G}^2$ *and* $\mathcal{G}^3$*, which is not the case for* $\mathcal{G}^4$*. This result confirms what has been said previously.*

Example 4. *Let us assume that the following structures* $\mathcal{G}^7$ *and* $\mathcal{G}^8$ *are two*

Bayesian network structures deduced from two different learning algorithms applied on a same database generated from the initial structure $\mathcal{G}^6$:

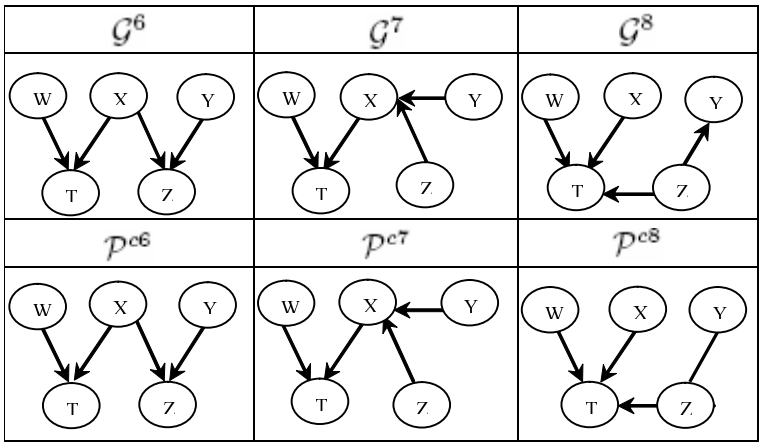

Fig. 5. The structures $\mathcal{G}^6$, $\mathcal{G}^7$, $\mathcal{G}^8$ and their corresponding $CPDGs$ $\mathcal{P}^{c6}$, $\mathcal{P}^{c7}$, $\mathcal{P}^{c8}$

Given the error matrices, shown in Table 7, and determined using the error scale $(a = 1, b = 2, c = 3)$*, we have* $GE(\mathcal{P}^{c6}, \mathcal{P}^{c7}) = 7$*, and* $GE(\mathcal{P}^{c6}, \mathcal{P}^{c8}) = 5$*. Resulting SEC values are then computed in addition to the M1, M2,* A^- *and* $A^\sim$ *criteria (see Table 8).*

Table 7. The error matrices $EM(\mathcal{P}^{c6}, \mathcal{P}^{c7})$ and $EM(\mathcal{P}^{c6}, \mathcal{P}^{c8})$

$EM(\mathcal{P}^{c6}, \mathcal{P}^{c7})$	Y	Z	T	W
X	2	3	0	0
Y		2	0	0
Z			0	0
T				0

$EM(\mathcal{P}^{c6}, \mathcal{P}^{c8})$	Y	Z	T	W
X	0	2	0	0
Y		1	0	0
Z			2	0
T				0

Note that, the M1, M2, A^- *and* $A^\sim$ *criteria give the same results for both learned structures* $\mathcal{G}^7$ *and* $\mathcal{G}^8$*, but SEC criterion detects that the first is better. In fact, this reflects the choice we made of the error classification scale, i.e.* $a < b < c$*: the change of an oriented arc by a non-oriented one (error a) is less important than*

Table 8. Structural evaluation using M1, M2, A⁻, A˜ and SEC criteria

	M1	M2	A⁻	A˜	SEC
$(\mathcal{G}^6,\ \mathcal{G}^7)$	50%	25%	25%	25%	0.292
$(\mathcal{G}^6,\ \mathcal{G}^8)$	50%	25%	25%	25%	0.208

the creation of a supplementary arc (error b) which is less important than an error c of an orientation for a compelled edge.

5. Experimental results

In order to evaluate the proposed GES-EM algorithm, we proceed to compare it to the standard AMS-EM algorithm[15].

For the experiments, we considered two well-known Bayesian networks: Asia[22] with 8 nodes and 8 arcs, and Insurance[4] with 27 nodes and 52 arcs. We randomly sampled data sets of different sizes using the probabilistic logic sampling method[18]. Then, we randomly removed some values from each one to get incomplete data sets with various missing value rates (%mv).

For each data set, we run the GES-EM and AMS-EM algorithms starting from an empty network and setting the maximum number of iterations to 30, for experiments carried out over Asia, and to 80, for those carried out over Insurance.

For each experiment, the following evaluation criteria have been considered :

- *Expected BIC score (eBIC)*: which measures the accuracy of the learned equivalence class. This criterion represents, obviously, the best algorithm's performance indicator, since the common goal of any search algorithm is to identify the structure with the highest scoring function.
- *SEC criterion*: which provides further evaluation of the quality of learned networks, through comparing them to their initial corresponding networks based on $CPDAGs$. Note that, for the penalization scale, we consider a=1, b=2 and c=3. Moreover, in order to compute the SEC criterion, initial Asia and Insurance networks, as well as, learned $DAGs$ resulting from the AMS-EM algorithm have been converted to their corresponding $CPDAG$ representations using the *DAG-to-CPDAG* algorithm[9].
- *M1, M2, A^- and $A^{\sim}$ criteria*: which compute respectively the percentage of correctly found arcs, the percentage of superfluous arcs, the percentage of missing arcs, and the percentage of badly oriented arcs. Note that, in order to compute these criteria, learned $CPDAGs$ resulting from the GES-EM algorithm have been converted to $DAGs$ using the *CPDAG-to-DAG* algorithm[13].
- *Iteration number (It)*: carried out by the algorithm before converging to a local maximum, i.e. the number of operators used to transform the initial empty network into a best learned one.

Table 9 and Table 10 summarize the different results. Note that, we omitted the A˜ criterion since, in all experiments, its corresponding value is 0.

Table 9. Evaluation of the GES-EM and AMS-EM algorithms, over Asia network, using eBIC score, SEC, M1, M2, A^- criteria and iteration number

Asia network							
Data size		1000					
%mv	Algo.	eBIC	SEC	M1	M2	A^-	It
10	GES-EM	-2437.932	0.265	37.5	50	62.5	9
	AMS-EM	-2435.774	0.296	50	75	50	13
20	GES-EM	-2477.661	0.250	37.5	37.5	62.5	11
	AMS-EM	-2489.370	0.234	37.5	37.5	62.5	30
30	GES-EM	-2531.917	0.250	37.5	37.5	62.5	13
	AMS-EM	-2530.397	0.250	37.5	62.5	62.5	11
Data size		5000					
%mv	Algo.	eBIC	SEC	M1	M2	A^-	It
10	GES-EM	-11475.447	0.265	37.5	50	62.5	11
	AMS-EM	-11503.659	0.265	37.5	37.5	62.5	11
20	GES-EM	-12007.689	0.265	37.5	50	62.5	11
	AMS-EM	-12015.982	0.312	37.5	75	62.5	15
30	GES-EM	-12300.971	0.250	37.5	37.5	62.5	13
	AMS-EM	-12301.909	0.281	50	37.5	62.5	30
Data size		10000					
%mv	Algo.	eBIC	SEC	M1	M2	A^-	It
10	GES-EM	-23572.637	0.265	37.5	50	62.5	11
	AMS-EM	-23594.559	0.328	25	62.5	75	12
20	GES-EM	-23875.514	0.265	37.5	50	62.5	11
	AMS-EM	-23936.069	0.359	25	75	75	13
30	GES-EM	-23783.227	0.250	25	37.5	75	20
	AMS-EM	-25497.921	0.265	25	37.5	75	30
Data size		20000					
%mv	Algo.	eBIC	SEC	M1	M2	A^-	It
10	GES-EM	-45176.625	0.265	37.5	50	62.5	11
	AMS-EM	-45185.646	0.312	37.5	75	62.5	13
20	GES-EM	-45928.973	0.234	37.5	50	62.5	18
	AMS-EM	-45900.385	0.359	25	75	75	17
30	GES-EM	-47624.757	0.265	37.5	50	62.5	20
	AMS-EM	-47672.792	0.328	37.5	62.5	62.5	30

A deep analysis of the obtained results leads to several conclusions of interest. Note first that, GES-EM algorithm outperforms in most cases the AMS-EM algorithm. Indeed, it returns more often higher expected BIC scores and thereby higher quality networks.

Moreover, as one can expect, the quality of learned networks deteriorates as the rate of missing values increases. Obviously, in these situations, both algorithms performed less better since there are less information in the data set. However, we may appreciate that the GES-EM algorithm consistently outperforms the AMS-EM algorithm.

In order to make easier further comparisons, we have plotted respectively in Figure 6 and Figure 7, the obtained expected BIC scores as a function of the missing value percent, for each considered data size. In each of the plots, the horizontal axis shows the missing value percent and the vertical one shows the expected BIC score (higher is better).

Table 10. Evaluation of the GES-EM and AMS-EM algorithms, over Insurance network, using eBIC score, SEC, M1, M2, A^- criteria and iteration number

Insurance network							
Data size		250					
%mv	Algo.	eBIC	SEC	M1	M2	A^-	It
10	GES-EM	-4770.383	0.134	15.38	28.84	84.61	25
	AMS-EM	-4865.054	0.124	19.23	19.23	80.76	80
20	GES-EM	-4876.673	0.123	17.30	21.15	82.69	41
	AMS-EM	-4983.330	0.119	11.53	9.61	88.46	80
30	GES-EM	-5296.691	0.116	9.61	13.46	90.38	57
	AMS-EM	-5322.122	0.128	7.69	17.30	92.30	80
Data size		500					
%mv	Algo.	eBIC	SEC	M1	M2	A^-	It
10	GES-EM	-8699.023	0.134	21.15	34.61	78.84	33
	AMS-EM	-9510.934	0.127	15.38	23.07	84.61	80
20	GES-EM	-9583.650	0.131	13.46	25	86.53	72
	AMS-EM	-9797.159	0.118	13.46	15.38	86.53	80
30	GES-EM	-10041.220	0.114	11.53	11.53	88.46	95
	AMS-EM	-10489.546	0.122	9.61	15.38	90.38	80
Data size		1000					
%mv	Algo.	eBIC	SEC	M1	M2	A^-	It
10	GES-EM	-16943.555	0.143	19.23	40.38	80.76	53
	AMS-EM	-17734.696	0.143	19.23	34.61	80.76	80
20	GES-EM	-18516.039	0.119	15.38	19.23	84.61	66
	AMS-EM	-19297.025	0.132	11.53	26.92	88.46	80
30	GES-EM	-19234.232	0.126	9.61	23.07	90.38	114
	AMS-EM	-21075.808	0.126	5.76	15.38	94.23	80
Data size		5000					
%mv	Algo.	eBIC	SEC	M1	M2	A^-	It
10	GES-EM	-82177.034	0.151	15.38	38.46	84.61	49
	AMS-EM	-82344.536	0.153	19.23	44.23	80.76	80
20	GES-EM	-88565.878	0.132	19.23	30.76	80.76	73
	AMS-EM	-88878.722	0.153	17.30	42.30	82.69	80
30	GES-EM	-91532.091	0.112	15.38	13.46	84.61	89
	AMS-EM	-91993.448	0.127	5.76	19.23	94.23	80

We observe that, either for experiments carried out on Asia or Insurance network, the GES-EM algorithm performs as well or often better than the AMS-EM algorithm. The most striking GES-EM performance lies in experiments carried out on the Insurance network, for the three first data set sizes, namely 250, 500 and 1000. This is clearly deduced from Figure 7, where the wide gap, between the expected BIC scores returned respectively by GES-EM and AMS-EM algorithm, illustrates the substantial gain and clear dominance of the GES-EM algorithm over the AMS-EM algorithm for all missing value percents.

This better performance is obviously due to the fact that our algorithm, contrary to AMS-EM which operates in *B-space*, performs search over equivalence classes of Bayesian networks, and makes use of more efficient operators than those used by the AMS-EM.

Moreover, the obtained SEC values relative to the networks learned using the GES-EM algorithm are usually lower than those relative to the networks learned using the AMS-EM algorithm (see Figure 8 and Figure 9). This means that our algorithm generates networks of better accuracy, more closer to initial ones.

Note also that the SEC criterion allows further analysis which may be unde-

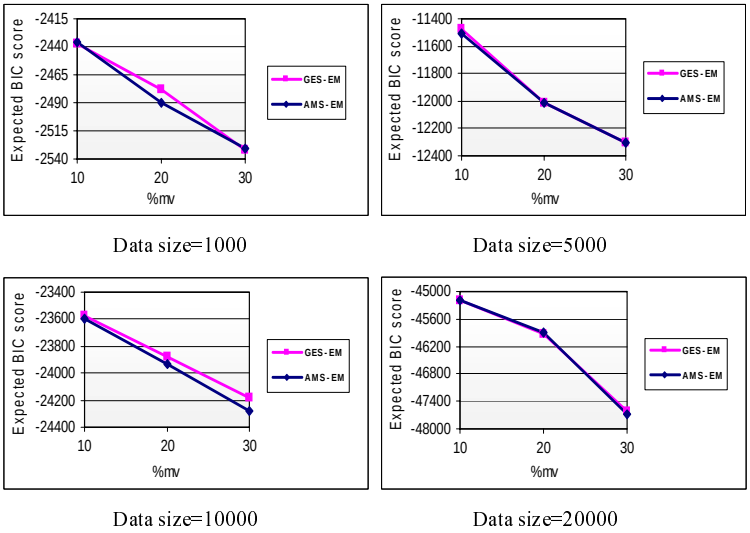

Fig. 6. Curves representing expected BIC scores as a function of missing value percents, for different data sizes (Asia)

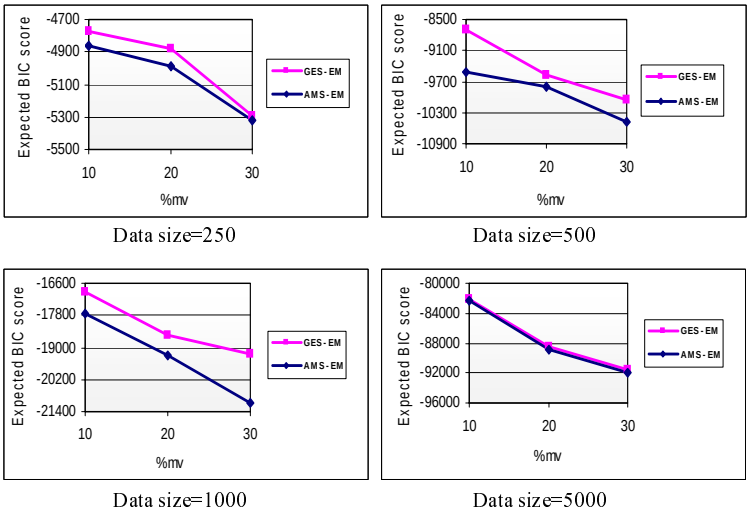

Fig. 7. Curves representing expected BIC scores as a function of missing value percents, for different data sizes (Insurance)

tectable by the expected BIC scoring function. For instance, with Asia network, the GES-EM algorithm leads to the same SEC values (i.e. 0.265), and thereby to the same learned networks, for all data sizes with 10 percent of missing values, which

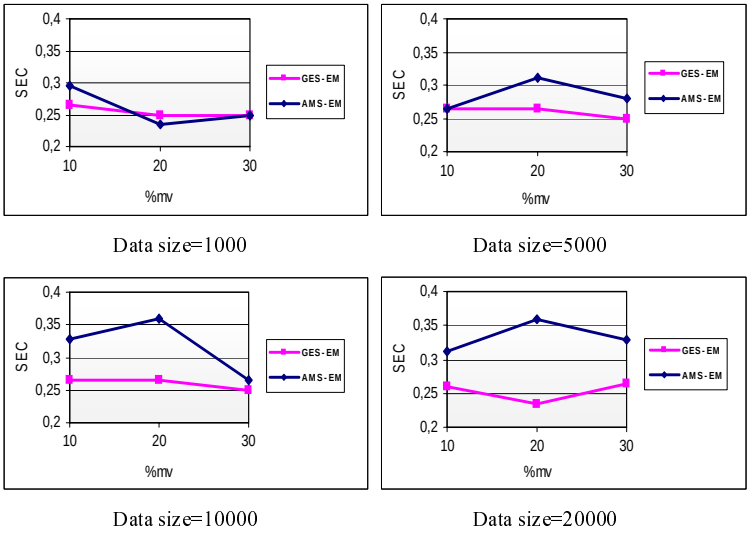

Fig. 8. Curves representing SEC values as a function of the missing value percents, for different data sizes (Asia)

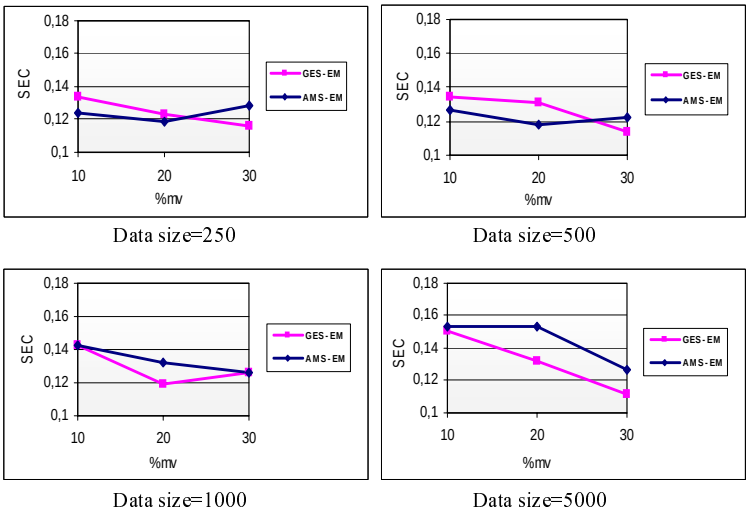

Fig. 9. Curves representing SEC values as a function of missing value percents, for different data sizes (Insurance)

means that with this small network, it is useless to consider a large data size. Such remark is not detected by the expected BIC score.

Furthermore, the SEC criterion provides a more significant results than tradi-

tional structural criteria, namely M1, M2 and A^-. For example, for experiments on Asia network (with 10000 instances and % mv = 30), these criteria give exactly the same results for both structures generated by the GES-EM and AMS-EM algorithms, while the SEC criterion allows to deduce that the first is better.

Through additional investigation of Figure 8 and Figure 9, we can observe that for higher percent of missing values, the SEC criterion is better. This is especially due to the choice we made concerning the error classification scale (i.e. $a < b < c$): not detecting dependencies (i.e. error a or b) is less important than introducing falsely new dependencies (i.e. error c). In fact, as the percent of missing values increases, the data will contain less information, which in turn decreases the percentage of discovered dependencies. In such situations, we will obtain lower error numbers, and thereby lower SEC values.

In addition, according to Table 9 and Table 10, it is interesting to notice that GES-EM algorithm learns more accurate networks with less iteration number and usually escapes bad local maxima. The iteration number may also indicate that our algorithm is faster than AMS-EM algorithm, which is confirmed by Table 11.

Table 11. Execution time ratio (GES-EM time execution/AMS-EM time execution)

	Asia				Insurance			
%mv	1000	5000	10000	20000	250	500	1000	5000
10	0.614	0.938	0.774	0.978	0.447	0.561	0.754	0.897
20	0.292	0.658	0.726	1.050	0.722	1.060	0.736	0.954
30	1.076	0.553	0.681	0.748	0.894	1.034	1.153	1.287

Table 11 also shows that, for most cases, the execution time ratios are less than one, which means that our algorithm is more efficient than the AMS-EM algorithm. This is especially due to the fact that our algorithm do not waste time generating and scoring equivalent networks, as does the AMS-EM algorithm, since it performs search over equivalence classes of Bayesian networks and scores only the consistent extension relative to each equivalence class neighbor.

6. Conclusion

This paper proposes a novel method, named GES-EM, for learning Bayesian network equivalence classes from incomplete data. The basic idea of our method is to use the *E-space* instead of the traditional used search spaces, i.e *B-space* or *tree-space*. We proved theoretically and experimentally that our algorithm converges, albeit as all EM-based algorithms, to a local maximum.

In addition, in order to assess the quality of learned networks, we propose a new structural evaluation function, named SEC, based on the comparison of the learned networks to the generating ones through $CPDAGs$. We considered, as well, for the experimental study, the expected BIC scoring function, the iteration number, and the execution time.

Experimental results show that GES-EM algorithm consistently outperforms AMS-EM algorithm, often by a non-trivial margin. Indeed, it yields, in less execution time, to more accurate and higher scoring networks than the AMS-EM algorithm.

Note that, the recent paper of Riggelsen[27] partly deals with the same topic considered in this work and presents, by another approach, corroborating results. More precisely, instead of EM, it predicts missing values on a per variable basis, by focusing only on those observed variables that directly influence the variables that are missing, and traverses the equivalence class search space, by way of simulation, using the work of[6] instead of the GES algorithm.

The GES-EM algorithm ensures the learning of equivalence classes incorporating some hidden variables, under the condition that the number and the location of these variables are known. A further work will be to extend it to discover hidden variables.

Another interesting issue is incremental learning of Bayesian networks from incomplete data. Few works deal with this problem and existing ones have been mostly restricted to updating the parameters assuming a fixed structure[16,25,31]. Thus, we aim to extend the GES-EM algorithm to deal with incremental learning, of both structure and parameters, from incomplete data.

References

1. C. F. Aliferis and G. F. Cooper. An Evaluation of an Algorithm for Inductive Learning of Bayesian Belief Networks using Simulated Data Sets. *In Proceedings of the Tenth Conference on Uncertainty in Artificial Intelligence*, pages 8-14, 1994.
2. S. A. Andersson, D. Madigan, M. D. Perlman, and C. M. Tiggs. A Graphical Characterization of Lattice Conditional Independence Models. *Annals of Mathematics and Artificial Intelligence*, 21:27-50, 1997.
3. S. A. Andersson, D. Madigan, and M. D. Perlman. A Characterization of Markov Equivalence Classes for Acyclic Digraphs. *The Annals of Statistics*, 25:505-541, 1997.
4. J. Binder, D. Koller, S. Russell and K. Kanazawa. Adaptive Probabilistic Networks with Hidden Variables. *Machine Learning*, 29:213-244, 1997.
5. W. L. Buntine. A Guide to the Literature on Learning Probabilistic Networks from Data. *IEEE Transactions on Knowledge and Data Engineering*, 8:195-210, 1996.
6. R. Castelo and T. Kocka. On Inclusion-Driven Learning of Bayesian Networks. *Journal of Machine Learning Research*, 4:527-574, 2003.
7. D. M. Chickering. A Transformational Characterization of Equivalent Bayesian Networks. *In the Eleventh Conference on on Uncertainty in Artificial Intelligence*, pages 87-98, 1995.
8. D. M.Chickering. Learning Bayesian Networks is NP-Complete. In D. Fisher and H. Lenz, editors, *Learning from Data: Artificial Intelligence Statistics V*, Springer-Verlag, pages 121-130, 1996.
9. D. M. Chickering. Learning Equivalence Classes of Bayesian Network Structures. *In Proceedings of the Twelfth Conference on Uncertainty in Artificial Intelligence*, pages 150-157, 1996.
10. D. M. Chickering. Optimal Structure Identification With Greedy Search. *Journal of Machine Learning Research*, 3:507-554, 2002.

11. G. F. Cooper and E. Herskovits. A Bayesian Method for the Induction of Probabilistic Networks from Data. *Machine Learning*, 9:309-347, 1992.
12. A. P. Dempster, N. M. Laird, and D. B. Rubin. Maximum Likelihood from Incomplete data via the EM algorithm. *Journal of the Royal Statistical Society*, B39:1-38, 1977.
13. D. Dor and M. Tarsi. A simple Algorithm to Construct a Consistent Extension of a Partially Oriented Graph. Technical Reprort R-185, Cognitive Systems Laboratory, UCLA Computer Science Department, 1992.
14. O. François and P. Leray. Bayesian Network Structural Learning and Incomplete Data. *International and Interdisciplinary Conference on Adaptive Knowledge Representation and Reasoning*, Espo, Finland, 2005.
15. N. Friedman. Learning Belief Networks in the Presence of Missing Values and Hidden Variables. *In Fourteenth Conference of Machine Learning*, San Francisco, CA, Morgan Kaufmann, pages 125-133, 1997.
16. N. Friedman and M. Goldszmidt. Sequential Update of Bayesian Network Structure. *In Proceedings of the Thirteenth Conference on Uncertainty in Artificial Intelligence*, pages 165-174, 1997.
17. D. Heckerman. A Tutorial on Learning Bayesian Networks. Technical Report MSR-TR-95-06, Microsoft Research, 1995.
18. M. Henrion. Propagating Uncertainty in Bayesian Networks by Probabilistic Logic Sampling. *Uncertainty in Artificial Intelligence*, 2:149-163, 1988.
19. F. V. Jensen. *An Introduction to Bayesian Networks*. UCL Press, University College, London, 1996.
20. F. Khalfallah and K. Mellouli. Détection de la Structure d'un Réseau Bayésien à Partir d'une Base de Données. *Revue de l'Intelligence Artificielle*, 2002.
21. F. Khalfallah, S. Chaabène, and K. Mellouli. Structure Learning of Bayesian Network from Distributed Homogeneous Databases. *ACIDCA ICMI'2005. Computational Intelligence*. Tozeur, Tunisia, 2005.
22. S. L. Lauritzen and D. J. Spiegelhalter. Local Computations with Probabilities on Graphical Structures and their Application to Expert Systems. *Journal of the Royal Statistical Society*, 50(2):157-224, 1988.
23. C. Meek. *Graphical Models: Selecting Causal and Statistical Models*. PhD thesis, Carnegie Mellon University, 1997.
24. M. Meila and M. I. Jordan. Estimating Dependency Structure as a Hidden Variable. *In NIPS* 10, 1998.
25. R. M. Neal, G. E. Hinton. A View of The EM Algorithm that Justifies Incremental Sparse and Other Variants. M. I. Jordan, Ed., Learning in Graphical Models, Kluwer Academic Publishers, Boston, 1998.
26. J. Pearl. *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference*. Morgan Kaufmann Publishers, San Francisco (California), 1988.
27. C. Riggelsen. Learning Bayesian Networks from Incomplete Data: an Efficient Method for Generating Approximate Predictive Distributions. *SIAM Conference on Data Mining*, 2006.
28. R. W. Robinson. Counting Unlabeled Acyclic Diagraphs. In C. H. C. Little, editors, *Combinatorial Mathematics V, Lecture Notes in Mathematics*, 622:28-43, 1977.
29. G. Schwarz. Estimating the Dimensions of a Model. *Annals of Statistics*, 6:461-464, 1978.
30. M. Singh. Learning Bayesian Networks from Incomplete Data. In AAAI97, pages 27-31, 1997.
31. D. J. Spiegelhalter and S. L. Lauritzen. Sequential Updating of Conditional Probabilities on Directed Graphical Structures. *Networks*, 20:579-605, 1990.

32. B. Thiesson, C. Meek, D. M. Chickering, and D. Heckerman. Learning Mixtures of DAG Models. *In Proceedings of the fourteenth Conference on Uncertainty in Artificial Intelligence*, pages 504-513, 1998.
33. T. Verma and J. Pearl. Equivalence and Synthesis of Ccausal Models. *In Proceedings of the Sixth Conference on Uncertainty in Artificial Intelligence*, San Francisco, Morgan Kaufmann, 1990.