Bayesian Networks: a Non-Frequentist Approach for Parametrization, and a more Accurate Structural Complexity Measure

Sylvain Gelly, Olivier Teytaud

▶ **To cite this version:**

HAL Id: inria-00112838

https://hal.inria.fr/inria-00112838

Submitted on 9 Nov 2006

Sylvain Gelly and Olivier Teytaud

Bayesian Networks: a Non-Frequentist Approach for Parametrization, and a more Accurate Structural Complexity Measure

Bayesian Networks: a Non-Frequentist Approach for Parametrization, and a more Accurate Structural Complexity Measure

Bayesian Networks Learning

Sylvain Gelly — Olivier Teytaud

TAO-inria, LRI, UMR 8623 (CNRS - Universite Paris-Sud), bat 490
Universite Paris-Sud 91405 Orsay Cedex

ABSTRACT. The problem of calibrating relations from examples is a classical problem in learning theory. This problem has in particular been studied in the theory of empirical processes (providing asymptotic results), and through statistical learning theory. The application of learning theory to bayesian networks is still uncomplete and we propose a contribution, especially through the use of covering numbers. We deduce multiple corollaries, among which a non-frequentist approach for parameters learning and a score taking into account a measure of structural entropy that has never been taken into account before. We then investigate the algorithmic aspects of our theoretical solution, based on BFGS and adaptive refining of gradient calculus. Empirical results show the relevance of both the statistical results and the algorithmic solution.

RÉSUMÉ. L'apprentissage à partir d'exemples est un problème classiquement étudié, au niveau théorique, via la théorie du processus empirique (fournissant des résultats asymptotiques) ou la théorie de l'apprentissage. L'application de ces théories aux réseaux bayésiens est incomplète et nous proposons une contribution, essentiellement via les nombres de couverture. Nous en déduisons de nombreux corollaires et notamment une approche non-fréquentiste pour l'apprentissage de paramètres et un score prenant en compte une mesure d'entropie structurelle qui affine les classiques mesures basées sur le nombre de paramètres seulement. Nous proposons alors des méthodes algorithmiques pour traiter de l'apprentissage qui découle de nos propositions, basées sur BFGS et l'affinage adaptatif du calcul du gradient.

KEYWORDS: learning in bayesian networks, structural score, learning theory.

MOTS-CLÉS : réseaux bayésiens, score sur les structures, théorie de l'apprentissage.

1. Introduction

Bayesian networks are a well known and powerful tool for representing and reasoning on uncertainty. One can refer to (Pearl, 2000),(Naim *et al.,* 2004) for a general introduction to bayesian networks. Learning the structure and the parameters of bayesian networks can be done through either expert information or data. Here, we only address the problem of learning from data, i.e. learning a law of probability given a set of examples distributed according to this law. Although a lot of algorithms exist for learning in bayesian networks from data, several problems remain.

First, when designing a bayesian model, one can have different goals e.g. i) evaluating qualitatively some probabilities; ii) evaluating expectations (of gain or loss). In the first case, evaluating a risk is roughly the question: does a given event happen with probability 10^{-30} or 10^{-5} ? Then, the use of logarithms, leading to maximum likelihood, is justified. In the second case, if we look for the expectation of f (vector of possible values indexed by possible states), the approximation of the real probability vector P by a probability vector Q leads to an error bounded (thanks to Cauchy-Schwartz inequality) by $||P - Q||_2 \times ||f||_2$. Therefore, optimizing a criterion monotonous as a function of the L^2 norm ($||P - Q||_2$) is the natural approach.

Second, when the postulated structure is not the right one, maximum likelihood (frequency approach for probability estimation) leads to very unstable results, far from minimizing $||P - Q||_2$.

We then propose a non-standard and tractable loss function for bayesian networks and evidences of the relevance of this loss function for this $||P - Q||_2$ criterion.

We also provide complexity measures of bayesian networks models. We then show some pros of this complexity measure:

– optimizing an ad hoc compromise between this complexity measure and the empirical L^2 error leads to an optimal-in-size structure as the number of independently identically distributed (i.i.d) examples grows to infinity;

– the score shows the influence of an entropy term, correlated with the sample-complexity of the network, *among networks for which usual scores are equal*;

– it only depends on the class of distributions that is modeled by the network, i.e. we can work on Markov-equivalent structures.

The paper is organized as follows: in section 2 we present an overview of our results. In section 3 we briefly survey some classical ways to learn bayesian networks from data and discuss the contribution of this paper in regard of existing results. In section 4 we introduce formally the problem and the notations. Section 5 first recalls some classical results of learning theory and presents our result about evaluation of VC-dimensions and covering numbers. We then generalize our results to more general bayesian networks, with hidden variables, in section 5.4. Section 6 shows corollaries applied to structure learning, parameters learning, universal consistency, and others. Section 7 presents algorithmic details. Section 8 presents empirical results.

2. Overview of results

The usual learning methods for parameters (section 3.1) lead asymptotically to the best parameters if the structure of the bayesian network is exact. The advantage of the more usual frequentist approach is that it is very fast and only involves local variables; choosing $P(B|A)$ only depends of the frequency of A, B-combinations. However, we show that the classical frequentist method is unstable and not-optimal for some natural criterions if the structure does not match the decomposition of the joint law. We show the consistency of our computationally harder method, based on a global fitting of the parameters (section 6.2).

We obtain risk bounds; i.e., given a number of examples, and after learning, we can claim that the probability to get an error larger than ϵ is bounded by some δ. Equivalently, we can classically deduce the *sample complexity*, i.e. the number of examples required to get an error lower than ϵ with probability at least $1 - \delta$.

We address also the case with hidden variables (section 5.4). We apply these bounds either in the case of a finite number of variables (section 6.3) and infinite number of variables (section 6.4).

Section 6.5 and theorem 8 provides an algorithm that guarantees universal consistency and asymptotic convergence towards the optimal loss (for the L^2-norm) asymptotically in the number of i.i.d examples. Moreover, convergence to a minimal structure, in the sense of a user-defined complexity-measure (including classical optimality measures), is proved.

The comparison between our complexity measure and usual ones gives insights on what is important for measuring the complexity of a structure. The first lemmas help calculating the *covering-numbers* of the set of bayesian networks for a given structure. These covering numbers are directly related to the complexity of the structure. Theorem 7 states a bound that contains R and $H(r)$ where R is the number of parameters of the structure and where $H(r) = -\sum_{k=1}^{a}(r(k)/R)\ln(r(k)/R)$ with $r(k)$ the number of parameters for the node k. Hence, $H(r)$ is the entropy of the number of parameters calculated over the nodes. We verify empirically (see Figure 4) that for a fixed R, $H(r)$ is correlated with the sample complexity. Hence, our score has an advantage on the AIC, BIC or MDL measure because these measures do not take into account this $H(r)$. See Figure 1 for an illustration of the role of the entropy term.

In section 7, we then present algorithms for optimizing our loss function. We use BFGS (the most standard quasi-Newton method, see (Broyden, 1970, Fletcher, 1970, Goldfarb, 1970, Shanno, 1970)), and the complexity of the loss function implies some non-straightforward algorithms for evaluating both the value of the loss function and the gradient of the loss function.

Empirical results (section 8) then show both the statistical relevance of our approach and that it is computationally tractable.
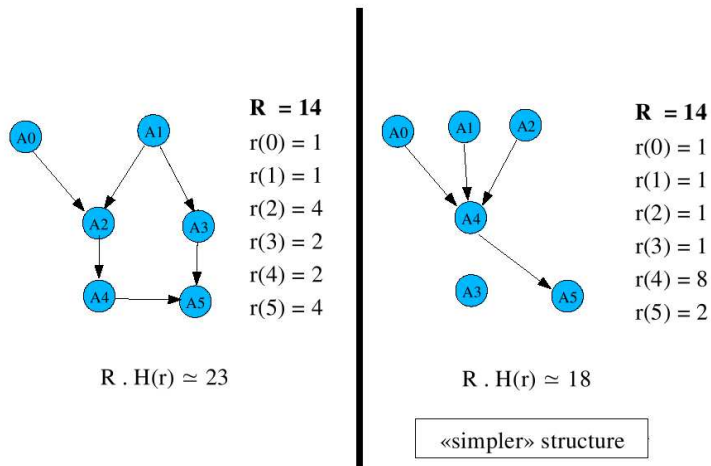
R = 14
r(0) = 1
r(1) = 1
r(2) = 4
r(3) = 2
r(4) = 2
r(5) = 4

R . H(r) ≃ 23

R = 14
r(0) = 1
r(1) = 1
r(2) = 1
r(3) = 1
r(4) = 8
r(5) = 2

R . H(r) ≃ 18

«simpler» structure

Figure 1. *Role of the entropy term in the score of a bayesian network structure. The two structures have the same number of parameters ($R = 14$), but have different distributions of the parameters over the structure. Hence, they have different entropy terms, and the right hand structure is considered "simpler" by our score*

3. Bayesian network learning

The problem of learning a bayesian network can be divided in two parts ((Naim *et al.,* 2004)):

– Learning the structure of the network, which is related to a graph, and not to the values of the probabilities.

– Given a structure, learning the parameters of the bayesian network, i.e. the conditional probabilities among variables.

Learning the structure, is a much more challenging problem than estimating the parameters.

3.1. *Learning parameters*

The classical approach for learning parameters is the likelihood maximization. This leads, with the classical decomposition of the joint probability in a product, to estimate separately each term of the product with the data. This method asymptotically converges toward the true probability, if the proposed structure is exact.

The main other method, the bayesian method, rather tries to calculate the most probable parameters given the data, and this is equivalent, with the Bayes theorem, to weight the parameters with an *a priori* law. The most used *a priori* is the Dirichlet distribution (see for example (Robert, 1994)).

3.2. *Structure learning*

Structure learning can be divided in two different methods:

– Find dependencies (and independencies and conditional dependencies) between the random variables, and deduce the structure of the graph.

– Map every structure of bayesian network to a score and search into the space of all structures for a "good" bayesian network, i.e., a structure with a good score.

The space of all structures is super-exponential, so heuristics must be defined when using the second method (limiting to the tree structures, sorting the nodes, greedy search). The search could also be done in the space of Markov equivalent structures (in which structures that encode the same probability law are identified), which has better properties ((Chickering, 2002a)). Our work, among other results, provide a score to the structures of bayesian networks, and so is closer to the second category. This score includes the influence of the structural entropy of the network, and is constant on markov-equivalent structures.

Some works already consider the influence of latent variables, but our work is orthogonal to this direction as we here have a complexity that uses the entropy $H(r)$ of the network of dependencies, that gives different scores to networks with the same number of parameters, whenever these networks have the same number (possibly 0) of latent variables. However, our score has not been improved in the direction of latent variables: the bounds that we prove also hold in the case of latent variables, but our score is exactly the same as for the same network with all variables observable. It is likely that the score could be improved; see (Geiger *et al.,* 1998) for more informations on statistical properties of networks with latent variables.

3.2.1. *Learning dependencies*

The principle of this method is the research of the independencies (conditionally or not) between the variables. We can cite the algorithms IC, IC*, (Pearl, 2000), PC, (Spirtes *et al.,* 1993), and more recently BN-PC of Cheng et al. (Cheng *et al.,* 1997a), (Cheng *et al.,* 1997b),(Cheng *et al.,* 2002).

The classical statistical tests used to test the independencies between variables is the χ^2 test. For hidden variables, the method is more complex, and we must distinguish several types of dependencies. We will not go further on this point here.

3.2.2. *Algorithms based on a score*

The notion of score of a structure is generally based on Occam's razor principle. The score measures the "complexity" of the structure. Therefore, the algorithm chooses a compromise between the empirical error made by the structure (quantified through the marginal likelihood or an approximation thereof (Chickering *et al.,* 1997)) and the complexity of this structure.

Here follows some well known scores for bayesian networks.

– AIC criteria (Akaike, 1970) or BIC (Schwartz, 1978) use essentially $Dim(bn)$ to penalize the complexity of the bayesian network, where $Dim(bn)$ denotes the "dimension" of the bayesian network, which counts the number of parameters.

– The Minimum Description Length (MDL) principle (Rissanen, 1978) uses the number of arcs and the number of bits used to code the parameters.

– The bayesian approach puts an *a priori* probability on the structure. For example, the bayesian Dirichlet score (Cooper *et al.,* 1992) assumes a Dirichlet *a priori* on the parameters. Some variants exist, like BDe (Heckerman *et al.,* 1994) , or BDgamma (Borglet *et al.,* 2002) which uses an hyperparameter, or methods using *a priori* probabilities on each child/parent relation (given for example by an expert).

4. Problem definition and notations

For the sake of clarity, we restrict our attention to binary random variable, without loss of generality. We present below bayesian networks and their notations, and some preliminary results.

4.1. *Notations*

Let A_1, $\ldots A_a$ be a binary random variables. We note A_b, where b is a subset of $[[1,a]]$, the random variable product of A_i where $i \in b$: $A_b = (A_{b_1}, A_{b_2}, \ldots, A_{b_{|b|}})$. Bayesian networks modelize a law of probability by $P((A_1, \ldots, A_a) = (a_1, \ldots, a_a)) = \Pi_{i \in [[1,a]]} P(A_i = a_i | A_{K_i} = a_{K_i})$, i.e. in short $P(A_{[[1,a]]}) = \Pi_i P(A_i | A_{K_i})$. We will distinguish in the sequel:

– a **bayesian network** is a family $K_1, \ldots, K_a$ of subsets of $[[1,a]]$ such that (i) $i \notin K_i$ (ii) there's no sequence $i_1, i_2, \ldots, i_k$ such that $i_{j+1} \in K_{i_j}$ and $i_k = i_1$ (i.e., the K_i note a Directed Acyclic Graph; the vertices are $[[1,a]]$, and the edge $i \to j$ is in the graph if and only if $i \in K_j$).

– an **instanced bayesian network** ibn, associated with a bayesian network bn, which is a law on $(A_1, \ldots, A_a)$ such that $ibn(A_1, \ldots, A_a) = \prod_j P(A_j | A_{K_j})$.

With bn a bayesian network, and ibn an instance of bn, we will say by abuse that $ibn \in bn$. We will map ibn with a vector of size 2^a corresponding to all the probabilities of all events $(A_1 = v_1, \ldots, A_a = v_a)$ for $v \in [[0,1]]^a$.

Without loss of generality, when we work on one particular bayesian network, we can renumber the vertices in topological order, so that $\forall i, i < K_i$ i.e. i is smaller than every element in K_i ($\forall i, \forall j \in K_i, i < j$). This will be assumed for all section 5.

We call **parameter** of an instanced bayesian network one of the real numbers $P(A_j|A_{K_j})$. We call **number of parameters** of a bayesian network bn, and we note $p(bn) = \sum_j 2^{\#K_j}$, where $\#b$ is the cardinal of b.

We consider $\hat{P}$ an empirical law (i.e. a sample of Dirac masses located at examples). Let P be a target law of probability. The sample leading to $\hat{P}$ is assumed independent and identically distributed (i.i.d.). We note E and $\hat{E}$ the expected value operators associated to P and $\hat{P}$ respectively. We note χ the random variable over $\{0,1\}^{2^a}$ such that $\sum_i \chi_i = 1$ and $P(\chi_i = 1) = P(A_1, .., A_a)$.

For Q a vector of size 2^a, of sum 1, identified to a probability distribution on the random vector $(A_1, \ldots, A_a)$ (more precisely $Q(i)$ is the probability of $(A_1 = a_1, ..., A_a = a_a)$, with $(a_1, ...a_a)$ the ith tuple of size a, among the 2^a possible tuples), we define

$$L(Q) = E_\chi \left(\sum_{i \in [1, 2^a]} |Q(i) - \chi(i)|^2 \right)$$

and $\hat{L}(Q) = \hat{E}(\sum_{i \in [1, 2^a]} |Q(i) - \chi(i)|^2)$. If bn is a bayesian network, we note $L(bn) = \inf_{ibn \in bn} L(ibn)$.

4.2. *Preliminary lemmas and propositions*

To spot the interest of $L(.)$ and $\hat{L}(.)$, we can note the

Lemma: With $N(Q) = \sum_{i \in [1, 2^a]} (P(i) - Q(i))^2$. and $\hat{N}(Q) = \sum_{i \in [1, 2^a]} (\hat{P}(i) - Q(i))^2$, we claim:

$$L(Q) = N(Q) + 1 - \sum_{i \in [1, 2^a]} P_i^2$$

$$\hat{L}(Q) = \hat{N}(Q) + 1 - \sum_{i \in [1, 2^a]} \hat{P}_i^2$$

Moreover, we claim the

Proposition A: *With probability $1 - \delta$, with $x^* \in argminL(.) = argminN(.)$, for all $\hat{x} \in argmin\hat{L} = argmin\hat{N}$, with $\sup_\delta X$ the $1 - \delta$ quantile of X:*

$$L(\hat{x}) \leq L(x^*) + 2\sup_\delta |L - \hat{L}|$$

Proof: The three following inequalities hold:

$$L(\hat{x}) \leq \hat{L}(\hat{x}) + \sup_{\delta} |L - \hat{L}|$$

$$\hat{L}(\hat{x}) \leq \hat{L}(x^*)$$

$$\hat{L}(x^*) \leq L(x^*) + \sup_{\delta} |L - \hat{L}|$$

Summing this three inequalities, we get the expected result. ∎

And finally:

Proposition B: *With probability* $1 - \delta$, *with* $x^* \in argminL(.) = argminN(.)$, *For all* $\hat{x} \in argmin\hat{L} = argmin\hat{N}$, *with* $\sup_{\delta} X$ *the* $1 - \delta$ *quantile of X:*

$$N(\hat{x}) \leq N(x^*) + 2\sup_{\delta} |L - \hat{L}|$$

Proof: Consequence of the lemma above and proposition A. ∎

All these elements confirm the interest of $\hat{L}$, which has both the interest of being an empirical average and the advantage of being closely related to natural cost functions.

5. Learning theory results

The VC dimension ((Vapnik *et al.,* 1971)) is the most classical tool of learning theory. It quantifies the inaccuracy of a learning depending on the size of the search space. The first version of VC-dimension was about classification, but there exists also the VC-dimension of spaces of real-valued functions (see e.g. (Vapnik, 1995)). This type of calculus has already been done in (Wocjan *et al.,* 2002, Nakamura *et al.,* 2005). We show similar results in section 5.1. The use of covering numbers, already known on the time of (Kolmogorov *et al.,* 1961), allows more precise bounds, as shown in section 5.2.

We will note $F(H, \delta)$ the smallest real Δ such that $P(\sup_{h \in H} |\hat{L}(h) - L(h)| \geq \Delta/\sqrt{n}) \leq \delta$, with n the number of examples. $F(H, \delta)$ depends upon n, but in many cases the dependency upon n can be removed (i.e. the supremum on n is not a bad approximation) and so we often refer to $F(H, \delta)$.

5.1. *Bounds based on VC dimension*

For a bayesian network bn, with probability at least $1 - \delta$:

$$\sup_{ibn \in bn} |\hat{L}(ibn) - L(ibn)| \leq F(\{ibn\}, \delta)/\sqrt{n}$$

The application $(a_1, \ldots, a_a) \mapsto \log P(A_1 = a_1, \ldots, A_a = a_a)$ is linear in the log of the parameters of the bayesian network. Hence, the VC-dimension of b_n is upper-bounded by the number of parameters. Combining with increasing functions preserves the VC dimension, and so the VC dimension of bn, seen as applications mapping $A_{[[1,a]]}$ to a probability is upper bounded by the number of parameters. We then deduce the

Theorem C: *The VC dimension of the set bn of instanced bayesian networks is upper bounded by the number of parameters V of bn. So thanks to classical results of learning theory*

$$P(\exists ibn \in bn | \hat{L}(ibn) - L(ibn)| \geq \epsilon) < 8(32e/\epsilon) \log(128e/\epsilon))^V \exp(-n\epsilon^2/32)$$

if $n \geq V$, and the covering number of ibn for the metric $d(ibn_1, ibn_2) = E(|ibn_1(A_{[[1,a]]}) - ibn_2(A_{[[1,a]]})|)$ is upper bounded by $e(R + 1)(4e/\epsilon)^R$.

Proof: These results are classical in learning theory. See e.g. (Antony *et al.,* 1999, Th18.4 and 17.4) for the upper bound on the probability and (Antony *et al.,* 1999, Th18.4, p251) for the covering number. We note that our results, even if they use a crude norm $N1(.)$, defined in the sequel, are better.

5.2. *Bound based on the covering number*

The covering numbers are a classical tool of learning theory. Inequalities of large deviations coming from this tool are very loose and conservative, but yet they are usually much tighter than those coming from VC-dimension.

5.2.1. *Introduction*

If one can cover $\mathcal{F}$ with $N1(\mathcal{F}, \epsilon)$ ϵ-balls (i.e. balls of radius ϵ) for the distance $d(x, y) = \sum |x_i - y_i|$, if $\hat{L}$ and L are between 0 and 2, then:

1) the risk, for a given function, to have a deviation $|\hat{L} - L|$ more than 2ϵ, is bounded by $2 \exp(-2n\epsilon^2)$;

2) The risk to have at least one of the centers of the balls having a deviation more than 2ϵ is upper bounded by $2N1(\mathcal{F}, \epsilon) \exp(-2n\epsilon^2)$;

3) If $d(f, g) \leq \epsilon \Rightarrow |L(f) - L(g)| \leq \epsilon$ and $d(f, g) \leq \epsilon \Rightarrow |\hat{L}(f) - \hat{L}(g)| \leq \epsilon$, (which is the case here, see lemma 2), then the risk to have at least a function in $\mathcal{F}$ having a deviation more than 4ϵ is upper bounded by $2N1(\mathcal{F}, \epsilon) \exp(-2n\epsilon^2)$. Indeed, if for all g of ϵ-skeleton C, we have $|\hat{L}(g) - L(g)| \leq 2\epsilon$, so we can map every f to one $g \in C$ such that $d(f, g) < \epsilon$ and so

$$|\hat{L}(f) - L(f)| \leq |\hat{L}(f) - \hat{L}(g)| + |\hat{L}(g) - L(g)| + |L(g) - L(f)| \leq \epsilon + 2\epsilon + \epsilon \leq 4\epsilon$$

The risk to have, among $\mathcal{F}$, a deviation more than ϵ is then upper bounded by $\delta = 2N1(\mathcal{F}, \epsilon/4) \exp(-2n(\epsilon/4)^2)$.

Then we can claim:

Proposition (maximal deviation for a given covering number):

$$\sqrt{n}F(\mathcal{F},\delta) \leq \inf\{\epsilon|\log(2N1(\mathcal{F},\epsilon/4)) - n\epsilon^2/8 \leq \log\delta\}$$

A lot of variations of this type of result exist in the literature. See for example (Vidyasagar, 1997) and (Antony *et al.,* 1999).

The covering number $N_\infty(\mathcal{F},\epsilon)$ of $\mathcal{F} = [0,1]^{2^a}$ is upper bounded by $\lceil 1/2\epsilon \rceil^{2^a}$ for the distance $d(x,y) = \sup_i |x_i - y_i|$.

The covering number $N1(\mathcal{F},\epsilon)$ of $\mathcal{F} = \{ibn \in bn\}$ is upper bounded as explained in the following subsection for the distance $d(x,y) = \sum |x_i - y_i|$. These both covering numbers deal with multi-valued functions; this is different from usual covering numbers.

5.2.2. *Covering number of $\mathcal{F}$*

We assume, without loss of generality that the nodes of the bayesian network are topologically sorted ($i < K_i$ for i node of the BN). So, there exists a partition $(E_k)_{k\in[[1,K]]}$ of the set of nodes such that:

– If $k \leq k'$ then $\forall(i,j) \in E_k \times E_{k'}, i \leq j$
– There is no edge between two nodes of a same E_k.

Such a partition $(E_k)_{k\in K}$ always exists; e.g. E_k can be reduced to the kth node in a topological order (or to the kth layer), and in this case $K = a$. Many other partitions may exist, depending on the structure of the bayesian network. We call depth the number k corresponding to the partition E_k and l_k the number of the last element (node) of E_k. By convention, $E_0 = \emptyset$ and $l_0 = 0$. In the sequel, we term E_k the kth level of the bayesian network.

Lemma 1:

$$N1(F_k, 2^{nbn(k)}\epsilon' + \epsilon) \leq N(F_{k-1},\epsilon)N_\infty(T_k,\epsilon')$$

where

– F_k for $k \leq K$ indicates the set of the functions calculated by the bayesian network until the level k (i.e. using only the nodes of $\bigcup_{i=1}^{k} E_i$).

– N_∞ indicates the covering number for the sup norm.

– T_k indicates the set of the vectors of the conditional probabilities involved in the transition from the level $k-1$ to the level k, i.e. all the parameters in the $P(A_i|A_{K_i})$ for i in level k.

– $nbn(k)$ indicates the number of nodes of the bayesian network in level k, i.e. $nbn(k) = \#E_k$;

$-\ l_k = \sum_{i=1}^{k} nbn(i);$

Lemma 2: $|L(Q) - L(Q')| \leq \sum_i |Q_i - Q'_i|.$

One can derive the **lemma 3:** $N_\infty([0,1]^h, \epsilon) \leq \lceil \frac{1}{2\epsilon} \rceil^h.$

Lemma 4: $N_\infty(T_k, \epsilon) \leq \lceil \frac{nbn(k)}{2\epsilon} \rceil^{r(k)}.$

where T_k indicates the set of the vectors of the conditional probabilities involved in the transition from the level $k-1$ to the level k (as in lemma 1) and where $r(k)$ indicates the number of parameters of the network involved in the transition between level $k-1$ and k.

Precisely, for a fixed k, T_k is the set of $P(E_k | \bigcup_{i=1}^{k-1} E_i)$, the $E_i, i = 1, ..., k$ taking the 2^{l_k} possible values. $r(k)$ indicates the number of the $P(A_i | A_{K_i})$ with $A_i \in E_k$, i.e. the number of parameters for this level.

Lemma 5: *Let K be the number of levels E_k; then*

$$LN1(K) \leq \sum_{i=1}^{K} r(i) \ln(\lceil \frac{nbn(i)2^{nbn(i)-1}}{\Delta_i} \rceil)$$

where $\epsilon_i > 0, i = 1...K$, $\epsilon_i < \epsilon_K, i = 1...K-1$, $\epsilon_0 = 0$, $\Delta(i) = \epsilon_i - \epsilon_{i-1}$ and $LN1(i) = log(N1(F_i, \epsilon_i))$ and with the notation $LN1(0) = 0$.

Theorem 6:

$$LN1(\epsilon) \leq \sum_{k=1}^{K} r(k) \ln(nbn(k)2^{nbn(k)-1} + \epsilon) - \sum_{k=1}^{K} r(k) \ln(\epsilon r(k)/R)$$

with $R = \sum_{i=1}^{K} r(i)$, $\epsilon = \epsilon_K$ et $LN1(\epsilon) = LN1(F_K, \epsilon)$, in particular for K the number of the last level.

Theorem 7: *The partition $\{E_k\}$ minimizing the bound of theorem 6 is the one in which all the E_k contain only one node. We have then:*

$$LN1(\epsilon) \leq \sum_{k=1}^{a} r(k) \ln(1 + \epsilon) - \sum_{k=1}^{a} r(k) \ln(\epsilon r(k)/R)$$

$$\leq R \ln((1+\epsilon)/\epsilon) + RH(r)$$

where $H(r) = -\sum_{k=1}^{a}(r(k)/R) \ln(r(k)/R)$.

REMARK. — For a fixed R (total number of parameters), our inequality has a (non-surprising) term in $log((1/\epsilon)^R)$ and a term which is the entropy of the vector $(r(1), \ldots, r(a))$, which shows that the less the parameters are equally distributed, the more the covering number is limited. Note also that asymptotically (ϵ close to 0), the complexity mainly depends on R; this is in the spirit of BIC/AIC criterions (Schwartz, 1978, Akaike, 1970).

Combining the proposition above and this bound yields a bound on the precision that provides a structural complexity penalization; see section 5.2.3 for the detailed resolution. For R large, the main term is $4\exp(H)$.

Proof of lemma 1:

Consider $k \geq 1$ fixed. Let $Pa(E_k)$ be the set of nodes that are parent of at least one node in E_k. Let X be the set of vectors of size $2^{\sum_{i=1}^{k-1} \#E_i} = 2^{l_{k-1}}$ with sum 1 of all instanced bayesian networks of a given structure until the level $k-1$. This set X is a set of vector summing to 1, and is exactly the set of functions computed by the bayesian network until level k. More precisely $X = \{(P(A_1 = a_1, ..., A_{l_{k-1}} = a_{l_{k-1}}))_{a \in \{0,1\}^{l_{k-1}}}\}$. Let Y be the set of vectors of size $2^{\sum_{i=1}^{k} \#E_i} = 2^{l_k}$ representing the probabilities of $ibn \in bn$ until the level k (similar to X, except that we include level k).

Let's cluster the mappings from nodes in $[[1, l_{k-1}]]$ to $\{0, 1\}$ (i.e. applications that give an assignement to each node until level $k-1$) by classes C_i such that for all $x \in C_i$ the assignements in $Pa(E_k)$ are identical. Let's note $\tilde{X}_i$ the restriction of X to nodes in C_i. Let N be the number of such classes. Let $t_i^j, i \in [1, N], j \in [1, 2^{nbn(k)}]$ be the probability of the jth value of the new variables (of level k), given a value of the class $\tilde{X}_i$ (each value of the variables in $\tilde{X}_i$ is appropriate because, by definition of $Pa(.)$, the new variables depend only on $Pa(E_k)$ among $E_1, ..., E_k$).

Let $y, y' \in Y$. We can then claim $y = (y_1, y_2, ..., y_N)$ with $y_i = (t_i^1 \tilde{x}_i, t_i^2 \tilde{x}_i, ..., t_i^{2^{nbn(k)}} \tilde{x}_i)$ and $y' = (y'_1, y'_2, ..., y'_N)$ with $y'_i = (t'^1_i \tilde{x}'_i, t'^2_i \tilde{x}'_i, ..., t'^{2^{nbn(k)}}_i \tilde{x}'_i)$, where $\tilde{x}_i \in \tilde{X}_i$ and $\tilde{x}'_i \in \tilde{X}_i$.

Let $\epsilon' = sup_{i,j}|t_i^j - t'^j_i|$ and $\epsilon = sup_i \|\tilde{x}_i - \tilde{x}'_i\|_1$. Then:

$$\|y - y'\| = \sum_{i=1}^{N} \sum_{j=1}^{2^{nbn(k)}} \|((t_i^j - t'^j_i)\tilde{x}_i + t'^j_i(\tilde{x}_i - \tilde{x}'_i)\|_1$$

$$\|y - y'\| \leq \sum_{i=1}^{N} \sum_{j=1}^{2^{nbn(k)}} \epsilon'\|\tilde{x}_i\|_1 + t'^j_i\|\tilde{x}_i - \tilde{x}'_i\|_1$$

$$= \sum_{i=1}^{N} 2^{nbn(k)}\epsilon'\|\tilde{x}_i\|_1 + \|\tilde{x}_i - \tilde{x}'_i\|_1 \leq 2^{nbn(k)}\epsilon' + \epsilon$$

Therefore,

$$N1(F_k, 2^{nbn(k)}\epsilon' + \epsilon) \leq N(F_{k-1}, \epsilon)N_\infty(T_k, \epsilon')$$

∎

Proof of lemma 2:

$$|L(Q) - L(Q')| = |E \sum_i (Q_i - \chi_i)^2 - \sum_i (Q'_i - \chi_i)^2| \leq E|\sum_i (Q_i - \chi_i)^2 - \sum_i (Q'_i - \chi_i)^2|$$

$$\leq E|\sum_i |(Q_i - \chi_i) - (Q'_i - \chi_i)|| \leq E|\sum_i |Q_i - Q'_i|| \leq \sum_i |Q_i - Q'_i|$$

∎

Proof of lemma 4: Consider some fixed k.

Then $P(E_k|\bigcup_{i=1}^{k-1} E_i) = \prod_{A_i \in E_k} P(A_i|A_{K_i})$.
The $P(A_i|A_{K_i})$ are probabilities and therefore lie between 0 and 1.
Let R_k be the set of the indexes of the parameters for level k.
We consider a fixed ϵ-skeleton S of $[0,1]^{R_k}$.
Consider $(p_i)_{i \in R_k}$ a set of parameters for the $P(A_i|A_{K_i})$ at level k.
By definition, t_j, the j-th coefficient of the level k, is equal to $t_j = \prod_{i \in h_j} p_i$ where h_j is the list of the indexes of the parameters involved in the calculus of the coefficient a_j; we note that $|h_j| = nbn(k)$.
So t is the vector of the coefficients, p is the vector of the parameters. Each element in t is the product of elements in p. We will note this $t = \Pi(p)$.

We now prove that

$$\{\prod s; s \in S\} \text{ is a } (nbn(k) \times \epsilon) - \text{skeleton of } \prod([0,1]^{R_k}) \qquad [1]$$

In order to prove equation 1, we note by induction on $nbn(k)$ that

$$\|\prod(p) - \prod(p')\|_\infty \leq nbn(k)\|p - p'\|_\infty \qquad [2]$$

where $(p, p') \in ([0,1]^{R_k})^2$.

Finally, using lemma 3:

$$N_\infty(R_k, \epsilon) \leq \lceil\frac{1}{2\epsilon}\rceil^{r(k)} \qquad [3]$$

Hence, using equations 3, 2 and 1: $N_\infty(T_k, \epsilon) \leq \lceil\frac{nbn(k)}{2\epsilon}\rceil^{r(k)}$ ∎

Proof of lemma 5:

From lemma 4,

$$N_\infty(T_k, \epsilon) \leq \lceil\frac{nbn(k)}{2\epsilon}\rceil^{r(k)}$$

Let K be the number of levels.

From lemma 1, $\forall \epsilon, \epsilon' > 0, \forall 1 \geq k \leq K$:

$$N1(F_k, 2^{nbn(k)}\epsilon' + \epsilon) \leq N(F_{k-1}, \epsilon)N_\infty(T_k, \epsilon')$$

Therefore, with redefining ϵ as $2^{nbn(k)}\epsilon' + \epsilon$:

$$N1(F_k, \epsilon) \leq N(F_{k-1}, \epsilon - \epsilon')N_\infty(T_k, \frac{\epsilon'}{2^{nbn(k)}})$$

Hence, replacing ϵ' by $\epsilon - \epsilon'$:

$$N1(F_k, \epsilon) \le N(F_{k-1}, \epsilon') N_\infty(T_k, \frac{\epsilon - \epsilon'}{2^{nbn(k)}})$$

with $\epsilon = \epsilon_K$, this equation becomes, for all $\epsilon \ge 0$,

$$LN1(\epsilon) \le \sum_{k=1}^{K} r(k) \ln(N_\infty(T_k, \epsilon_k - \epsilon_{k-1}))$$

$$LN1(\epsilon) \le \sum_{k=1}^{K} r(k) \ln(\frac{nbn(k)2^{nbn(k)-1}}{\epsilon_k - \epsilon_{k-1}})$$

$$LN1(\epsilon) \le \sum_{k=1}^{K} r(k) \ln(\frac{nbn(k)2^{nbn(k)-1}}{\Delta_k})$$

■

Proof of theorem 6:

Bounding the integer part, we can transform the lemma 5 as follows:

$$LN1(\epsilon) \le \sum_{k=1}^{K} r(k) \ln(nbn(k)2^{nbn(k)-1} + \Delta_k) - \sum_{k=1}^{K} r_k \ln(\Delta_k)$$

and bounding Δ_k by ϵ,

$$LN1(\epsilon) \le \sum_{k=1}^{K} r(k) \ln(nbn(k)2^{nbn(k)-1} + \epsilon) - \sum_{k=1}^{K} r_k \ln(\Delta_k)$$

In particular, with $\Delta_k = \frac{r(k)}{\sum_k r(k)}$ (which comes from the Kuhn-Tucker condition for the maximization $\sum r_k \ln(\Delta_k)$ under the constrain $\sum Delta_k = \epsilon$), we get

$$LN1(\epsilon) \le \sum_{k=1}^{K} r(k) \ln(nbn(k)2^{nbn(k)-1} + \epsilon) - \sum_{k=1}^{K} r_k \ln(\epsilon r(k)/R)$$

■

Proof of theorem 7:

The theorem 6 holds for any splitting of the network in levels, provided that a node of a level k does not depend upon another node of the same level. ($E_k \bigcap Pa(E_j) = \emptyset$).

We can now optimize the result by changing the limits of the levels.

Let $k(i)$ be the level in which the node i belongs. Let $s(i)$ be the number of parameters associated to the node i, i.e. $2^{\#K_i}$. We have then $\forall i \in [1, a], r_{k(i)} = \sum_{j=l_{k(i)}}^{l_{k(i)+1}} s(j)$.

Then using Theorem 6:

$$LN1(\epsilon) \leq \sum_{k=1}^{K} r(k) \ln(\frac{R(nbn(k)2^{nbn(k)-1} + \epsilon)}{\epsilon r(k)})$$

hence:

$$LN1(\epsilon) \leq \sum_{k=1}^{K} (\sum_{j=l_k}^{l_{k+1}} s(j)) \ln(\frac{R(nbn(k)2^{nbn(k)-1} + \epsilon)}{\epsilon \sum_{j=l_k}^{l_{k+1}} s(j)})$$

so:

$$LN1(\epsilon) \leq \sum_{i=1}^{a} s(i) \ln(\frac{R((l_{k(i)+1} - l_{k(i)})2^{l_{k(i)+1}-l_{k(i)}-1} + \epsilon)}{\epsilon \sum_{j=l_{k(i)}}^{l_{k(i)+1}} s(j)})$$

$$LN1(\epsilon) \leq \sum_{i=1}^{a} s(i) \ln(\frac{R(c(k(i)) + \epsilon)}{\epsilon \sum_{j=l_{k(i)}}^{l_{k(i)+1}} s(j)})$$

with $c(k) = (l_{k+1} - l_k)2^{l_{k+1}-l_k-1}$

Let's assume that there exists a E_k of cardinal > 1. To simplify the notations, we can assume, without loss of generality that $k = 1$ and $s(l_1) = \min_{i \in E_1} s(i)$. Let $l = l_1$. Let $C(n) = n2^n$.

We are going to prove that the bound is better for the partition such that we remove l_1 from the level k, therefore adding a level only composed of node l_1. We note that this new partition respect the constraints if these were respected for the first partition. By iteratively repeating this improvement, we reach the partition with one node per level.

The terms of the bound which are modified for the first and second partition are respectively:

$$\sum_{i=1}^{l-1} s(i) \ln(\frac{C(l)}{s(l) + \sum_{j=1}^{l-1} s(j)}) + s(l) \ln(\frac{C(l)}{s(l) + \sum_{j=1}^{l-1} s(j)})$$

and

$$\sum_{i=1}^{l-1} s(i) \ln(\frac{C(l-1)}{\sum_{j=1}^{l-1} s(j)}) + s(l) \ln(\frac{1}{s(l)})$$

The difference between the two bounds can be written:

$$d = \sum_{i=1}^{l-1} s(i) \ln(\frac{C(l) \sum_{j=1}^{l-1} s(j)}{C(l-1)(s(l) + \sum_{j=1}^{l-1} s(j))}) + s(l) \ln(\frac{C(l)s(l)}{s(l) + \sum_{j=1}^{l-1} s(j)})$$

So:

$$d = A \ln\left(\frac{C(l)A}{C(l-1)(A+B)}\right) + B \ln\left(\frac{C(l)B}{A+B}\right)$$

with $A = \sum_{j=1}^{l-1} s(j)$ et $B = s(l)$. Therefore:

$$\frac{d}{A+B} = \lambda \ln(\lambda C(l)/C(l-1)) + (1-\lambda)ln(C(l)(1-\lambda))$$

with $\lambda = \frac{A}{A+B}$.

The minimum of this expression is for $\lambda = \frac{C(l-1)}{1+C(l-1)}$ and so $0 \leq \ln\left(\frac{C(l)}{1+C(l-1)}\right) \leq \frac{d}{A+B}$ ∎

5.2.3. *Summary of the results*

We have calculated an upper bound on the covering number of the family of instanced bayesian networks $ibn \in bn$ for a given structure bn. This structure determines the number of parameters $r(k)$ for $k \in [1, K]$ (and $R = \sum_{k=1}^{a} r(k)$).

Then, theorem 7 states that for all $\epsilon > 0$:

$$LN1(\epsilon) \leq \sum_{k=1}^{a} r(k) \ln(1+\epsilon) - \sum_{k=1}^{a} r(k) \ln(\epsilon r(k)/R) \qquad [4]$$

The lemma 2 states that the conditions $d(f,g) \leq \epsilon \Rightarrow |L(f) - L(g)| \leq \epsilon$ and $d(f,g) \leq \epsilon \Rightarrow |\hat{L}(f) - \hat{L}(g)| \leq \epsilon$ are true. So we can here apply the results stated in the subsection 5.2.1, and then the risk to have, among $\mathcal{F}$, a deviation more than ϵ is then upper bounded by $\delta = 2N1(\mathcal{F}, \epsilon/4)\exp(-2n(\epsilon/4)^2)$. Therefore, $F(\mathcal{F}, \delta) \leq \sqrt{n} \inf\{\epsilon | \log(2N1(\mathcal{F}, \epsilon/4)) - n\epsilon^2/8 \leq \log \delta\}$. We can then rewrite this as

$$P(\sup_{ibn \in bn} |\hat{L}(ibn) - L(bn)| > \epsilon) \leq 2N1(bn, \epsilon/4)\exp(-n\epsilon^2/8) \qquad [5]$$

with $\tilde{bn} = \{i\tilde{bn}; ibn \in bn\}$, and $F(\tilde{bn}, \delta) \leq F(bn, \delta)$.

And the equation above (5), using equation (4), can be solved in ϵ (depending upon R, $H = H(r)$, n and δ, as follows:

$$C = 2(H - \frac{1}{R}\log(\delta/2)) \qquad B = \frac{4n}{R}\exp(C)$$

$$A = -R \times W_{Lambert}(B) - 2\log(\delta/2) + 2RH \qquad \epsilon = \frac{4}{\exp(-A/(2R))}$$

$$\epsilon = 4\left(\frac{\delta}{2}\right)^{-\frac{1}{R}}\exp\left(-\frac{1}{2}W_{Lambert}\left(\frac{4n}{R}\frac{e^{2H}}{\frac{\delta}{2}^{\frac{2}{R}}}\right) - \frac{1}{R} + H\right)$$

Where $W_{Lambert}$ is the (main) function such as $W_{Lambert}(x) \times e^{W_{Lambert}(x)} = x$.

Therefore, a natural bound on the generalization error is $\hat{L} + \epsilon(R, H(r), n, \delta)$ where n is the number of examples, δ is a risk threshold, and R and $H(r)$ only depend on the structure.

This provides a score $\epsilon(R, H(r), n, \delta)$ for structural learning. For R large, the main term of the complexity score is $4\exp(H)$. Note that the score also involves the risk threshold δ. A somewhat subtle element must be pointed out. Usually, structural risk minimization consists in the minimization of $\hat{L}(s) + R(m(s))$ where R is some regularization term, m is some complexity measure, s is the structure, $\hat{L}(s)$ is the minimal empirical error with structure s. If you know the covering numbers of $\{s; m(s) \leq M\}$ as a function of M, then you can design $R(.)$ such that minimizing $\hat{L}(s) + R(m(s))$ ensures universal consistency. In the case of our bound, we compute the covering numbers for a fixed structure, and not for all structures with complexity upper bounded by some quantity. However, this can be handled by (i) a priori choosing an increasing sequence of structures, as we will do in corollary C3, or (ii) spreading the risk δ among structures with a given number of parameters: consider $\epsilon(R, H(r), n, \delta/N)$ where N is the number of structures with less than R parameters, and you can ensure a bound $\epsilon(R, H(r), n, \delta/N)$ on $L - \hat{L}$ uniformly on all structures with less that R parameters. Dividing by N is the crude approach; using the micro-choice bounds (i.e. unequally spreading the risk δ) can be helpful (Langford *et al.,* 2003).

We will now show that the score is constant among Markov-equivalent structures.

5.3. *Score-equivalence: the score is the same for Markov-equivalent structures*

As state before, an important notion in comparing different structures is the Markov equivalence property (Chickering, 2002a). Two bayesian networks are Markov-equivalent if they encode the same probability laws. So it is important that the proposed score has the property of *score equivalance* which means that the score give the same results for two equivalent structures. We note $Dim(\mathcal{G})$ the number of parameters in the bayesian network corresponding to $\mathcal{G}$.

We prove here that two equivalent structures have:

– the same R.
– the same $H(r)$.
– the same $\hat{L}$.

Hence, this prove that they have the same score. The proofs use lemma 1, theorem 2 and theorem 3 from (Chickering, 1995).

5.3.1. *Reminder*

We remind here these results for convenience. We note $\mathcal{G} = (U, E_{\mathcal{G}})$ an acyclic directed graph (DAG) (i.e. a bayesian network). We note $Pa(x)_{\mathcal{G}}$ the set of the parents of the node x in the graph $\mathcal{G}$. We note $\mathcal{G} \approx \mathcal{G}'$ if the bayesian networks based on $\mathcal{G}$ and $\mathcal{G}'$ are equivalent.

Definition (definition 2 in (Chickering, 1995)) *An edge $e = x \rightarrow y \in E_{\mathcal{G}}$ is covered in $\mathcal{G}$ if $Pa(y)_{\mathcal{G}} = Pa(x)_{\mathcal{G}} \cup \{x\}$.*

Lemma A (lemma 1 in (Chickering, 1995)) *Let $\mathcal{G}$ be any DAG containing the edge $x \rightarrow y$, and let $\mathcal{G}'$ be the directed graph identical to $\mathcal{G}$ except that the edge between x and y in $\mathcal{G}'$ is $y \rightarrow x$. Then $\mathcal{G}'$ is a DAG that is equivalent to $\mathcal{G}$ if and only if $x \rightarrow y$ is a covered edge in $\mathcal{G}$.*

Lemma B (theorem 2 in (Chickering, 1995)) *Let $\mathcal{G}$ and $\mathcal{G}'$ be any pair of DAGs such that $\mathcal{G} \approx \mathcal{G}'$. There exists a sequence of distinct edge reversals in $\mathcal{G}$ with the following properties:*

– The edge reversed from $r_{i-1}(\mathcal{G})$ to $r_i(\mathcal{G})$ is a covered edge in $r_{i-1}(\mathcal{G})$.

– Each $r_i(\mathcal{G})$ is a DAG and $r_i(\mathcal{G}) \approx \mathcal{G}'$.

– After all reversals, $r_N(\mathcal{G}) = \mathcal{G}'$.

where $r_i(\mathcal{G})$ is $\mathcal{G}$ after the i first edge-reversals and N is the number of edge-reversals.

Lemma C (theorem 3 in (Chickering, 1995)) *If $\mathcal{G} \approx \mathcal{G}'$ then $Dim(\mathcal{G}) = Dim(\mathcal{G}')$.*

5.3.2. *Results*

The first point (same R for two equivalent structures) is exacly the lemma C.

From lemma B, we only need to show that $H(r)$ stays the same for two equivalent structures ($\mathcal{G}$ and $\mathcal{G}'$) with only one edge $x \rightarrow y$ which becomes $y \rightarrow x$, and we only have to consider the case of an edge $x \rightarrow y$ that is covered in $\mathcal{G}$. From the definition, we have $Pa(y)_{\mathcal{G}} = Pa(x)_{\mathcal{G}} \cup \{x\}$. Then, with $r_{\mathcal{G}}(x)$ the number of parameters of node x in $\mathcal{G}$:

– $r_{\mathcal{G}}(y) = 2r_{\mathcal{G}}(x)$.

– We also have $y \rightarrow x$ covered in $\mathcal{G}'$ and then $Pa(x)_{\mathcal{G}'} = Pa(y)_{\mathcal{G}'} \cup \{y\}$, so $r_{\mathcal{G}'}(x) = 2r_{\mathcal{G}'}(y)$.

– As $x \rightarrow y$ is the only edge reverted between $\mathcal{G}$ and $\mathcal{G}'$:

   - the only changing terms in $H(r)$ is those corresponding to x and y.

   - and also $r_{\mathcal{G}}(y) = 2r_{\mathcal{G}'}(y)$

– so $r_{\mathcal{G}'}(y) = r_{\mathcal{G}}(x)$ and $r_{\mathcal{G}'}(x) = r_{\mathcal{G}}(y)$

So the entropy term $H(r)$ is the same in $\mathcal{G}$ and in $\mathcal{G}'$.

Two equivalent structures of bayesian networks give the same space of possible probability distributions. Hence, the $\hat{L}$ are the same.

This concludes the 3 points stated above, and so all measures only depending on R, $H(r)$ and $\hat{L}$ are *score equivalent*.

5.4. *Results with hidden variables*

We here consider the case in which variables are hidden, so only a part of all the variables are involved in the calculus of $\hat{L}$ or L. It is important to note that it is not equivalent to reduce the bayesian network to a smaller bayesian network. For example, a network with one hidden variable B and observed variables A_i for $i \in [1, d]$, with dependencies $P(A_i|B)$, has only $2d + 1$ parameters and encodes laws that are very difficult to model (i.e. would need much more parameters) with a bayesian network which has only the A_i as variables.

By mapping a bayesian network to a vector (of sum 1) of the probabilities it calculates, a bayesian network in which some variables are hidden can be mapped to a reduced vector. If all the variables are binary (this is the case in this paper), the number of probabilities to code is divided by 2 for each variable which becomes hidden. An instance $\tilde{ibn}$ of a bayesian network $\tilde{bn}$ which has v variables, and among them l hidden variables, can be identified to an element of $[0, 1]^{2^{v-l}}$ summing to 1, whereas the corresponding instantiated bayesian network ibn which has no hidden variables provides 2^v probabilities (hence a vector in $[0, 1]^{2^v}$, summing to 1). $\tilde{ibn}$ then equals $summation(ibn)$, where $summation(.)$ is the ad hoc summation operator.

As $summation(.)$ is 1-lipschitz for the distance $d(x, y) = \sum |x_i - y_i|$ (i.e. $d(\tilde{x}, \tilde{y}) \leq d(x, y)$), we deduce:

Proposition maximal deviation in a bayesian network with hidden variables:

The risk to have a deviation at least ϵ for a $\tilde{ibn} \in \tilde{bn}$ is upper bounded as follows:

$$P(\sup_{\tilde{ibn} \in \tilde{bn}} |\hat{L}(\tilde{ibn}) - L(\tilde{bn})| > \epsilon) \leq 2N1(bn, \epsilon/4) \exp(-n\epsilon^2/8)$$

with $\tilde{bn} = \{\tilde{ibn}/ibn \in bn\}$.

REMARK. — We can notice that we don't improve the bound in spite of the fact that the number of variables is lower. We can of course bound $F(\tilde{bn}, \delta)$ by $F([0, 1]^{v-l}, \delta)$ if the number of hidden variables is so large that this rough bound becomes the best.

6. Paradigms of learning

Many applications of the calculus above can be defined, in the same spirit of use of covering numbers, to give:

– non-parametric non-asymptotic confidence intervals;

– universally consistent algorithms.

We state in the sections below some of the numerous corollaries one can deduce from the calculus of covering numbers above. These corollaries also hold with hidden variables.

6.1. *Choice between several structures of bayesian network*

Let's assume that someone has to choose between several structures $bn_1, \ldots, bn_h$. Consider the algorithm that chooses bn_{i_0} such as $\inf_{ibn \in bn_{i_0}} \hat{L}(ibn) + F(bn_{i_0}, \delta)/\sqrt{n}$ is minimal and chooses $\hat{ibn} \in bn_{i_0}$ such as $\hat{ibn} = argmin_{ibn \in bn_{i_0}} \hat{L}(ibn)$. So, the algorithm chooses the *structure* minimizing the empirical error penalized by a term depending upon the complexity of the structure. Then, it chooses the *bayesian network* of this structure minimizing the empirical error.

Corollary C1: Then, $L(\hat{ibn}) \leq L(ibn') + \epsilon$ for all $ibn' \in \cup bn_i$, with $\epsilon = 3 \sup F(bn_i, \delta)/\sqrt{n}$, with a risk upper bounded by $h\delta$.

Proof:

Define $ibn* = argmin_{ibn \in \cup_i bn_i} L(ibn)$. Define $\hat{ibn}_i = argmin_{ibn \in bn_i} \hat{L}(ibn)$.

Then, $P(L(\hat{ibn}_i) - \hat{L}(\hat{ibn}_i) > \epsilon/3) \leq \delta$ with $\epsilon = 3 \sup F(bn_i, \delta)/\sqrt{n}$.

So, simultaneously for all i, $L(\hat{ibn}_i) - \hat{L}(\hat{ibn}_i) \leq \epsilon/3$, with probability $1 - h\delta$.

And therefore, $L(\hat{ibn}) \leq \hat{L}(\hat{ibn}) + \epsilon/3$ and by definition of $\hat{ibn}$ (note that $\hat{L}(\hat{ibn}) + F(bn_{i_0})/\sqrt{n} \leq \hat{L}(ibn*) + F(bn*, \delta)/\sqrt{n}$ where $ibn* \in bn*$), $L(\hat{ibn}) \leq \hat{L}(ibn^*) + 2\epsilon/3$ and therefore $L(\hat{ibn}) \leq L(ibn^*) + \epsilon$ ∎

(the constant 3 in ϵ is not optimal)

This provides a natural criteria to choose between several structures, in the spirit of the method of *"structural risk minimization"*, which is classical in learning theory.

6.2. *Comparison between local and global fitting: consistency of the minimization of $\hat{L}$*

Corollary C2: Consider bn a bayesian network. Then for any distribution P,

$$L(argmin_{ibn \in bn} \hat{L}) \to \inf_{bn} L$$

whereas for some distributions P,

$$L(ibn \ \forall i, \ ibn(A_{\{i\} \cup K_i})/ibn(A_{K_i}) = \hat{P}(A_{\{i\} \cup K_i})/\hat{P}(A_{K_i})) \not\rightarrow inf_{bn} L$$

(i.e., calibrating each coefficient of bn on $\hat{P}$ leads asymptotically to a non-optimal ibn), with $ibn(B)$ for B a set of variable, is the probability given by the bayesian network ibn for the variables B.

Proof: The convergence $L(argmin_{ibn \in bn} \hat{L}) \rightarrow \inf_{bn} L$ is an immediate consequence of the finiteness of the covering number for any ϵ; the VC-dimension being finite, the convergence is indeed almost sure. One can note that the same result holds with well-chosen nested families bn_n, increasing with n, as explained in section 6.4.

The counter-example for the second result is as follows (Figure 2). Let P be the law defined by: $P(A = true \wedge B = true) = a$, $P(A = false \wedge B = false) = 1 - a$ (and P=0 for the 2 others events). Assume that $bn = \{P(A), P(B)\}$ (so the structure bn assume the independence).

Then calibrating bn on $\hat{P}$ leads to $ibn(A) = \hat{P}(A) \rightarrow a$, $ibn(B) = \hat{P}(B) \rightarrow a$. $N(ibn)$ (equals to L plus a constant) is, for $x = ibn(A)$ and $y = ibn(B)$ (i.e. x is the probability given by the bayesian network ibn for the event $A = true$, and y the probability of $B = true$):

$$(xy - a)^2 + x^2(1 - y)^2 + y^2(1 - x)^2 + ((1 - x) \times (1 - y) - (1 - a))^2$$

the derivative of this expression w.r.t x (as well as w.r.t. y), in $x = a, y = a$ is positive for $0 < a < \frac{1}{2}$ and negative for $\frac{1}{2} < a < 1$. So, the solution $x = a, y = a$ is not the minimum of this equation except if $a = \frac{1}{2}$. ∎

Another example of bad behavior of the frequentist approach is the fact that $P(B|A)$ can not be estimated if A never occurs. With our method, such cases can be handled directly, without specialization of the algorithm.

6.3. *Universal consistency and bound with a finite number of variables*

We assume that a heuristic system is given in order to rank dependencies between variables, for the building of the structure. This method, whenever required, provides a dependency $A_j \rightarrow A_i$ that increases a dependency $P(A_i|A_{K_i})$ to a dependency $P(A_i|A_{K_i \cup \{j\}})$. This method is designed to increase step by step the complexity of the structure.

Consider the following algorithm, for $\epsilon(n)$ a sequence converging to 0 as $n \rightarrow \infty$:

– Consider n the number of examples and δ the risk threshold chosen by the user;

– Heuristically sort the list of dependencies (possibly using a separate database);

– As long as the next dependency added to bn does not lead to $F(bn, \delta)/\sqrt{n} > \epsilon(n)$, add the dependency the most suitable according to the heuristic;

– Choose $i\hat{b}n \in bn$ minimizing $\hat{L}$;

– Claim $L(ibn) \le \hat{L}(ibn) + F(bn, \delta)/\sqrt{n}$.

Corollary C3:

– with confidence at least $1 - \delta$, the bound provided on $L(ibn)$ holds;

– asymptotically in the number of examples, $L(i\hat{b}n)$ converges to $inf_{ibn}L(ibn)$ (inf among any ibn, independently of the structure, and not only $inf_{ibn \in bn}L(ibn)$), at least if the heuristic, within a finite number of increases of the structure, leads to bn such that $inf_{ibn \in bn}L(ibn) = inf_{ibn}L(ibn)$ (this is a small and natural hypothesis as the heuristic can simply lead to the complete graph between observable variables if the number of dependencies is sufficiently large).

The proof is a consequence of the convergence of $F(bn, \delta)/\sqrt{n}$ to 0 (as it is upper bounded by $\epsilon(n)$) as $n \to \infty$.

6.4. *Universal consistency and confidence intervals with infinitely many variables*

We consider here an infinite number of states, but a finite number of examples. Variable j of example i is noted $a_{i,j}$. The sequence of vectors [1] $(a_{i,1}, \ldots, a_{i,743}, \ldots)$ for $i \in \mathbb{N}$ is assumed independently identically distributed. The algorithm is as follows:

1) the user provides n, ϵ and δ; an oracle provides the $a_{i,j}$ when they are required by the program.

2) evaluate bn maximal for the inclusion[2] (chosen by any heuristic among multiple possible solutions, provided that bn increase as n increases), such that $F(bn, \delta)$ is upper-bounded by ϵ ; the variables modelled by b_n are the observable ones among the union of the A_j and A_{K_j} such that bn is defined by the $P(A_j|A_{K_j})$;

3) choose $ibn \in bn$ minimizing $\hat{L}$;

4) provide to the user a bound $L(ibn) \le \hat{L}(ibn) + F(bn, \delta)/\sqrt{n}$;

Corollary C4:

Let's note $mod(bn)$ the set of events which are deterministic functions of observable variables modelled by bn.

– for any E event depending upon a finite number of A_j, $ibn(E)$ is evaluated if n is large enough and its value converges to $P(E)$ as $n \to \infty$, if at least the heuristic method guarantees that for a given increasing sequence of integers k_i, the number of dependencies is bounded by k_i as long as the ith observable variable is not added to the network (this is a natural requirement).

---

1. There are infinitely many vectors but these vectors are countable.
2. We say that a bayesian network bn_1 is included in a bayesian network bn_2 if any dependency in bn_1 is a dependency in bn_2 within a renumbering of latent variables.

– the bound provided on $L(ibn)$ holds with probability at least $1 - \delta$.

– thanks to the Borell-Cantelli lemma (see e.g. (Vidyasagar, 1997, p26)), one can write that if $\sum_n \delta_n$ is finite (for example $\delta_n = 1/n^2$) and if $F(bn_n, \delta_n)/\sqrt{n} \to 0$ as $n \to \infty$, with bn_n the structure chosen for a number n of examples, then there is almost sure convergence of $\sup |P(E) - ibn(E)|$ for $E \in mod(b_n)$ to 0; we must ensure $\delta_n \leq \delta$ to assert, moreover, that the bound $\hat{L}(ibn) + F(bn, \delta)/\sqrt{n}$ holds.

6.5. *Universal consistency and convergence to the right network of dependencies*

We propose in this section an algorithm in order to build bayesian networks having two important properties:

– it is universally consistent;

– the "size" of the structure converges to the optimal one.

The second point is not trivial, as it is very difficult to guarantee convergence to a non-redundant structure. The meaning of "size" in this theorem can be multiple. The penalization measure for a structure can be user-defined as soon as the hypotheses holds. We show below that for some penalizations we can find classical definitions of optimality as *parameters-optimality* or *inclusion-optimality* (Chickering *et al.*, 2002).

Precisely, we claim the

Theorem 8: universal consistency and convergence to the right structure *Define*

$$ibn \in argmin_{U(ibn) \leq n} \hat{L}(ibn) + R(ibn, n)$$

where U is an application which associates a real number to any instantiated bayesian network, such that $\forall (ibn_1, ibn_2) \in bn \; U(ibn_1) = U(ibn_2)$ (i.e., two bayesian networks having the same structure have the same image through U) , and where $R(ibn, n) = R'(ibn)R(n)$ associates a real number to an instantiated bayesian network ibn and to a sample size n.

We note in the sequel (by abuse of notation) $U^{-1}(n) = \{ibn; U(ibn) \leq n\}$.

Then:

1) **universal consistency:** *if H0, H1 and H2 hold, then $L(ibn)$ almost surely goes to L^*;*

2) **convergence of the size of the structure:** *if H0, H1, H2 and H3 hold, then $R'(ibn) \to R'(ibn^*)$ where $ibn^* \in \arg\min_{L^*=L(ibn)} R'(ibn)$.*

H0: for n sufficiently large, $ibn^* \in U^{-1}(n)$;

H1: $sup_{ibn \in U^{-1}(n)} R'(ibn)R(n) \to 0$ as $n \to \infty$;

H2: $F(U^{-1}(n), 1/n^2)/\sqrt{n} \to 0$ as $n \to \infty$;

H3: $F(U^{-1}(n), 1/n^2)/(R(n)\sqrt{n}) \to 0$ as $n \to \infty$;

Proof:

Define $bn = U^{-1}(n)$ and $\epsilon(bn, n) = sup_{ibn \in U^{-1}(n)} |\hat{L}(ibn) - L(ibn)|$.

Let's prove the universal consistency under hypothesis H0, H1, H2.

$$L(ibn) \leq \hat{L}(ibn) + \epsilon(bn, n)$$

$$\leq \inf_{ibn' \in bn} \hat{L}(ibn') + R(ibn', n) - R(ibn, n) + \epsilon(bn, n)$$

$$\leq \inf_{ibn' \in bn} L(ibn') + \epsilon(bn, n) + R(ibn', n) - R(ibn, n) + \epsilon(bn, n)$$

$$\leq \inf_{ibn' \in bn} L(ibn') + R(ibn', n) + 2\epsilon(bn, n)$$

Thanks to H1, we only have to prove that $\epsilon(bn, n) \rightarrow 0$ almost surely.

By definition of $F(., .)$, $P(\epsilon(bn, n) \geq F(bn, 1/n^2)/\sqrt{n}) \leq 1/n^2$.

In particular, for any ϵ, H2 implies that for n sufficiently large, $F(bn, 1/n^2)/\sqrt{n} < \epsilon$, and so $P(\epsilon(bn, n) > \epsilon) \leq 1/n^2$. Thanks to the Borell-Cantelli lemma, the sum of the $P(\epsilon(bn, n) > \epsilon)$ being finite for any $\epsilon > 0$, $\epsilon(bn, n)$ almost surely converges to 0.

We have achieved the proof of consistency. We now start the proof of the convergence of the size of the structure.

Thanks to H0, if n is sufficiently large, $ibn^* \in bn$. We restrict our attention to such n.

$$\hat{L}(ibn) + R(ibn, n) \leq \hat{L}(ibn^*) + R(ibn^*, n)$$

$$R'(ibn)R(n) \leq R'(ibn^*)R(n) + \hat{L}(ibn^*) - \hat{L}(ibn)$$

$$R'(ibn)R(n) \leq R'(ibn^*)R(n) + L^* + 2\epsilon(bn, n) - L(ibn)$$

$$R'(ibn) \leq R'(ibn^*) + 2\epsilon(bn, n)/R(n)$$

It is then sufficient, using H3, to show that $\epsilon(bn, n)/R(n) \rightarrow 0$ almost surely. Let's show this by the Borell-Cantelli lemma as well. By definition of $F(., .)$, $P(\epsilon(bn, n) \geq F(bn, 1/n^2)/\sqrt{n}) \leq 1/n^2$.

In particular, for any ϵ, H3 implies that for n sufficiently large, $F(bn, 1/n^2)/(R(n)\sqrt{n}) < \epsilon$, and so $P(\epsilon(bn, n)/R(n) > \epsilon) \leq 1/n^2$. Thanks to the Borell-Cantelli lemma, the sum of the $P(\epsilon(bn, n)/R(n) > \epsilon)$ being finite for any $\epsilon > 0$, $\epsilon(bn, n)/R(n)$ almost surely converges to 0. ∎

We prove optimality for some criterion R'. Let's now see some **Links to classical optimality criteria:**

For a given structure bn:

– if $\exists ibn \in bn$, $ibn = P$ then bn is called an I-map of P. It is equivalent to say that all the (conditional or marginal) independencies given by the structure bn are included in the law P.

– if all (conditional or marginal) independencies of P are in bn then the structure is called a D-map of P.

– if the both conditions above are fulfilled, then the structure bn is called a P-map of P.

One of the objectives of learning the structure of a bayesian network is to have a good approximation of the joint law. However, asymptotically we only need that bn is an I-map of P. By definition of ibn^* and from the lemma of section 4.2, then ibn^* is an I-map of P.

The second, more difficult to obtain objective is that the structure contains as many independencies of the law as possible. We note $bn \leq bn'$ if the set of laws representable by the structure bn is included in the set of laws representable by bn'. It is equivalent to say that every (conditional or marginal) independencies of bn' are found in bn. We note $bn < bn'$ if the inclusion is strict.

A structure bn which has the minimal number of parameters among the I-maps of P is called parameter-optimal. A structure bn which is minimal for $\leq$ among the I-maps of P is called inclusion-optimal.

When the law P can be perfectly represented by a bayesian network, that is to say that there exists a P-map of P, it is well known that the parameter-optimal and inclusion-optimal models are the same and are in the same Markov-equivalence class of bayesian networks. In this case, we can set $R'(bn)$ equal to the number of parameters of the structure bn. Then H1 holds, and ibn converges towards ibn^* which is a representant of the optimal class.

When no P-map exists for P, then there can be multiple inclusion-optimal models, and multiple parameter-optimal models. But from theorem 2 of (Chickering *et al.,* 2002) (coming from a theorem of (Chickering, 2002b)), all the parameter-optimal models are also inclusion-optimal. Again, using $R'(bn)$ as the number of parameters of the structure bn, we have that ibn converges towards a parameter-optimal model.

Finally, we can also state that the proposed score (see 5.2.3) is *asymptotically consistent* (in the sense of (Chickering *et al.,* 2002)), as if the number of examples goes to infinity, the dominant term becomes the number of parameters.

7. Algorithmic

We have shown in sections above that the optimization of $\hat{L}$ leads to better generalization properties than the usual local method, at least for some reasonable loss functions. Unfortunately, in its basic form, $\hat{L}$ is difficult to evaluate and to optimize. We propose:

– other more practical formulations of $\hat{L}$, and algorithms for computing it (section 7.1),

– methods for adapting these algorithms to the computation of the gradient (section 7.2).

– optimization methods (7.3), including adaptive precision (based on estimates of the precision of the computation of the gradient) and BFGS.

However, in spite of efforts below, it is clear that optimizing our measure is much more difficult than optimizing the standard KL-divergence. The L^2-norm has some pros (it avoids some stability problems of the KL-divergence, it is more tailored to the evaluation of expectations) and some drawbacks (it is computationally much harder, and it neglects very small probabilities).

7.1. *Objective functions*

We here present in the following sections:

– a reformulation of the loss function $\hat{L}$;

– an exact method for the computation of $\hat{L}$;

– a Monte-Carlo method for the computation of $\hat{L}$;

– a method inspired by the quota method for the computation of $\hat{L}$;

7.1.1. *Introduction*

Lemma: $\hat{L}(Q) = 1 + S + \frac{1}{n} \sum_{e=1}^{n} -2Q(i_e)$ with n the number of examples, i_e the rank of the example e in lexicographic order (if e is the example where all the variables are false, then $i_e = 1, \ldots$), and $S = \sum_{i=1}^{2^a} Q(i)^2$.

Proof: $\hat{L}(Q) = \frac{1}{n} \sum_{e=1}^{n} \sum_{i=1}^{2^a} (Q(i) - \chi(e))^2$ with $\chi(e)$ the vector χ representing the example e.

$$\hat{L}(Q) = \frac{1}{n} \sum_{e=1}^{n} \left((Q(i_e) - 1)^2 + \sum_{i=1, i \neq i_e}^{2^a} Q(i)^2 \right)$$

$$\hat{L}(Q) = \frac{1}{n} \sum_{e=1}^{n} \left(-2Q(i_e)^2 + 1 + \sum_{i=1}^{2^a} Q(i)^2 \right)$$

$$\hat{L}(Q) = 1 + S + \frac{1}{n} \sum_{e=1}^{n} -2Q(i_e)^2$$

∎

The term $\sum_{e=1}^{n} -2Q(i_e)^2$ is easily tractable, as it can be computed in $O(an)$. Hence, computing $\hat{L}(Q)$ is difficult due to S. We therefore propose other formulations of S that allow computational feasibility.

REMARK. — Many works have been devoted to the computation of sums of probabilities (marginalization for inference). S is a sum of squared probabilities so it is likely that techniques like those involved in (Lauritzen *et al.,* 1988, Cozman, 2000, Kschischang *et al.,* 2001, Guo *et al.,* 2002) could be applied in this context also. We can then expect huge improvements in our computation times/precisions.

7.1.2. *Properties of the objective function*

As we want to optimize $\hat{L}$, we want to examine the properties of the objective function. The computation of the gradient will be examined in section 7.2.

S (see above) is convex, and one could expect that $\hat{L}$ is also convex. Indeed, the convexity of $\hat{L}$ depends on the distribution of the examples, by the term $-\frac{2}{n}\sum_{e=1}^{n}Q(i_e)^2$. Figure 3 shows a counter example, i.e. an objective function $\hat{L}$ which is not convex. However, we have observed experimentally that $\hat{L}$ is often roughly convex.
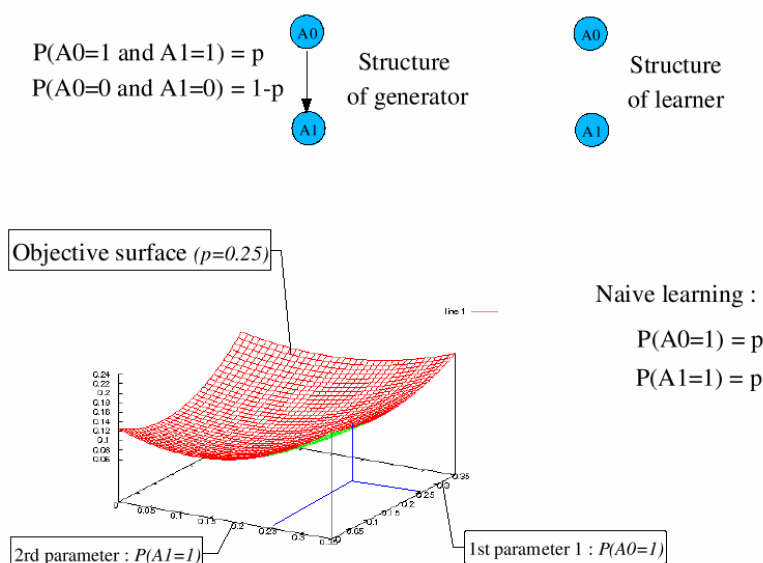

Figure 2. *An illustration of why optimizing globally the parameters is consistent. We plot here the objective surface (L) in function of the two parameters for a 2-nodes "naive" bayesian network. The structure of the learner is presented on the up-right of the figure. The structure of the generator of the law is presented on the up-left of the figure. The parameters coming from frequentist learning method is represented by the intersection of the three lines parallel to the axis on the three dimensional graphic. We see that this point is not the optimum of the objective function*
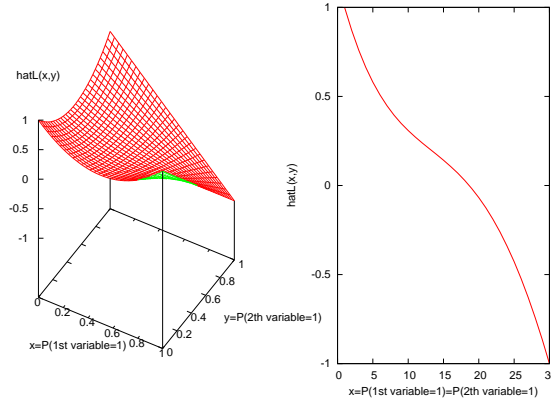
Figure 3. *Left hand graph: objective function $\hat{L}$ in function of the two parameters for a 2 nodes "naive" bayesian network.* ***Right hand graph****: diagonal cut of the left hand graph. This shows that the objective function $\hat{L}$ can be no convex*

7.1.3. *Exact method for the evaluation of S*

Thanks to the decomposition of the product law we propose an algorithm to compute S in less (probably much less) than 2^a operations. The number of operations required to compute S depends on the structure of the bayesian network. A simpler structure leads to less calculus.

Roughly speaking, the main ideas are as follows:

– we start with a set of nodes F_1, called the front, reduced at the empty set at the beginning, and that moves across the bayesian network (roughly, in "reverse order" for the topological order, as far as this notion makes sense for a set of nodes), and C_1, initialized to the empty set.

– during all the process, S is known at all the nodes in the front.

– for $t = 1, \ldots, a$, the front evolves as follows:

    - a_t is the tth node in reverse topological order (multiple possible choices are heuristically decided as explained below);

    - we add this point to the front;

    - we add in the front some points according to rules defined below;

    - S is computed for any new element in the front so that S is always known for all elements in the front.

F^1 and C^1 are the empty list.

For any $t \in [[1, a]]$,

– F^t is a list of f_t subsets of $[[1, a]]$; $F^t = (F_1^t, ..., F_{f_t}^t)$, is initialized for $t = 1$ at the empty list $F^1 = ()$;

– C^t is a list of f_t subsets of $[[1, a]]$; $C^t = (C_1^t, ..., C_{f_t}^t)$, is initialized for $t = 1$ at the empty list $C^1 = ()$;

– dom_i^t is a subset of $[[1, a]]$;

– L^t is a list of f_t applications from dom_i^t to $\{0, 1\}$.

– $a_t \in [[1, a]]$ is the node chosen (the choice is performed as defined below) at step t.

They are defined by induction ($t \in [[1, a]]$) by:

– a_t is chosen among the last nodes in topological order among the nodes that are different of the a_s for $s < t$ (i.e. $a_t \in [[1, a]] \setminus \{a_s; s < t\}$ and a_t has no successor in $[[1, a]] \setminus \{a_s; s < t\}$); if many a_t are possible, it is chosen such that $|C'^t|$ (defined below) is minimal;

– $I_t = \{i \in [1, f_t] / a_t \in C_i^t\}$ is a list of integers;

– $C'_t = \bigcup_{i \in I_t} C_i^t \bigcup K_{a_t} \setminus \{a_t\}$ is a subset of $[[1, a]]$;

– $C^{t+1} = (C_i^t)_{i \notin I_t, 1 \leq i \leq f_t}.(C'_t)$ where $a.b$ is the concatenation of lists a and b;

– $S^{t+1} = (S_i^t)_{i \notin I_t, 1 \leq i \leq f_t} \cdot \left(c' \in 2^{C'_t} \mapsto \sum_{a_t} P(a_t|c')^2 \prod_{i \in I_t} S_i^t(c'_{|dom_i^t}) \right)$

(where $a_{|b}$ is the restriction of a to the domain b)

– $F^{t+1} = (F_i^t)_{i \notin I_t, 1 \leq i \leq f_t} \cdot \left(\bigcup_{i \in I_t} F_i^t \cup \{a_t\} \right)$ where . denotes the concatenation of families

– $f_{t+1} = |F^{t+1}|$ (length of the list)

– $dom^{t+1} = (dom_i^t)_{i \notin I_t, 1 \leq i \leq f_t} \cdot 2^{(C'_t)}$

S is equal to the product of the S^t.

One can verify by induction that for any $t \in [[1, a]]$, for any $1 \leq i \leq f_t$, S_i^t, the following holds: $S_i^t : c \mapsto \sum_{v \in 2^{F_i^t}} P(v|c)^2$. This implies the consistency of the exact method.

7.1.4. *Approximate methods for the computation of S*

$\hat{L}$ and its gradient are hard to compute. In this section we define algorithms approximating S in an efficient manner. The most simple is Monte-Carlo, and we define improvements of Monte-Carlo based on regular samplings.

$S = \sum_{i=1}^{2^a} Q(i)^2$ can be written as $S = EQ(i)$, E being the expectation under the law Q. We are going to approximate this expectation thanks to a finite sample drawn according to law Q. Now, $\widehat{\hat{L}}$ is the estimate of $\hat{L}$ where S is replaced by an empirical mean on a finite sample.

S is the most computationally expensive term of $\hat{L}$, the other one (see section 4.1) being quickly computable in an exact manner. We present here the Monte-Carlo

method, and improved other methods, for the computation of S. The same methods can also be used for the computation of ∇S (see section 7.2) , both of them being necessary either for the gradient descent or for the BFGS method.

We also present the estimation of variance in the case of Monte-Carlo; the same estimate will be used for other approximate methods.

7.1.4.1. Monte-Carlo method for the computation of S

The most straightforward solution is the Monte-Carlo method: just simulate the law Q associated to the network and average the results. S is therefore approximated by $\sum_{j=1}^{n} Q(e_j)$ where the e_j are i.i.d among $0, 1^a$ with distribution of probability Q.

Now, let's consider the estimation of variance. We consider the case of the approximation of $\nabla \hat{L}$, the case of $\hat{L}$ being similar.

For the sake of clarity, we note $g = \nabla \hat{L}$ the exact gradient and $\hat{g} = \widehat{\nabla \hat{L}}$ the approximate gradient. Then,

$$||\hat{g} - g||^2 = \sum_{i=1}^{d} (\hat{g}_i - g_i)^2 \simeq \sum_i (\frac{\sigma_i N_i}{\sqrt{n}})^2 = \frac{1}{n} \sum_i \sigma_i^2 N_i^2$$

We now assume independence of the N_i. This is an approximation. Then,

$$E||\hat{g} - g||^2 = \frac{1}{n} \sum_i \sigma_i^2 E N_i^2 = \frac{1}{n} \sum_i \sigma_i^2 \text{ as } E N_i^2 = 1$$

$$Var||\hat{g} - g||^2 = \frac{1}{n^2} \sum_i \sigma_i^4 Var(N_i^2) = \frac{2}{n^2} \sum \sigma_i^4 \text{ as } Var(N_i) = 2$$

where the N_i are independent standard normal variables (expectation 0 variance 1), σ_i is the standard deviation of the gradient restricted to coordinate i, n is the number of draws for the Monte-Carlo method.

We can then use as bound on $||\hat{g} - g||^2$ the formula $\frac{1}{n} \left(\sum \sigma_i^2 + \sqrt{2 \sum \sigma_i^4} \right)$.

7.1.4.2. Quotas Method for the computation of S

A more stable solution is defined as follows. We consider the 2^a possible values of the whole set of variables, in lexicographic order, with their probabilities $q_1, q_2, \ldots,$ q_{2^a}. Then, we consider $x_i = \frac{2i-1}{2 \times 2^a}$ for $i = 1, \ldots, 2^a$. Then, we consider the average of the q_{j_i} where j_i is minimal such that $\sum_{h=1}^{j_i} q_h \geq x_i$.

If the lexicographic order is with respect to an ordering of variables in topological order, this is easy to implement until large number of examples.

7.2. *Computation of the gradient*

The gradient of S is the main difficulty in the computation of the gradient of $\hat{L}$. We show here how ∇S can be evaluated in a similar manner as S.

Consider the following high-level definition of S:

$$S = \sum_j S_j$$

where j is an index on all possible assignments of the a variables, and

$$S_j = \Pi_{i \in I_j} p_i^2 \Pi_{i \in I'_j} (1 - p_i)^2$$

where $\forall j; I_j \cap I'_j = \emptyset$ and $\forall i; |\{j; i \in I_j \cup I'_j\}| = 1$. Then

$$\frac{\partial S_j}{\partial p_i} = 0 \text{ if } i \notin I_j \cup I'_j$$

$$= 2S/p_i \text{ if } i \in I_j$$

$$= -2S/(1 - p_i) \text{ if } i \in I'_j$$

So, the Monte-Carlo method can be adapted in the following manner:

– draw examples as in the computation of $\hat{L}$;

– for each example, adapt the at most a parameters that are concerned (one per variable).

So, for a given number of examples, the algorithmic complexity is at most multiplied by a. The quota method can be adapted in the same way.

The exact method can be adapted in the following manner:

– for each parameter p_i of the bayesian network:

    - fix the value of the parent-variables, in order to make p_i relevant;

    - evaluate S for the bayesian network with these fixed values;

    - apply formulas above providing $\partial S/\partial p_i$.

BFGS is able of approximating the hessian thanks to successive gradients in a very efficient manner; so the gradient will be enough for optimization-algorithms below (no hessian required).

7.3. *Optimization*

We now turn our attention to the optimization methods suitable for $\hat{L}$.

Gradient descent is a very simple solution for non-linear optimization. It is used for comparison with BFGS. BFGS is a classical algorithm for non-linear optimization, with the following characteristics:

– superlinear in many cases;

– needs only the gradient and approximates the hessian thanks to the successive values of the gradient.

We used Opt++ and LBFGSB, freely available on the web, as BFGS optimization softwares. The results presented below come from LBFGSB, a limited BFGS algorithm for bound-constrained optimization.

8. Experiments

We defined in section 5 some objective functions with good statistical properties. We defined in 6 some algorithms for the optimization of these objective functions.

We present in the following subsections:

– the questions (both about statistical significance and algorithmic complexity) that we want to answer by empirical studies;

– the empirical results.

We aim at answering the following questions:

1) is the entropy of the network relevant or just a second-order theoretical point?

2) are our algorithms for the computation of S and its derivative efficient in the following cases:

  - exact method;

  - Monte-Carlo method with/without random seed or with quota;

in particular, depending upon the dimension/sample size.

3) is the superiority of the optimization of $\hat{L}$ on the local method as shown in section 6.2 validated by practical experiments ?

8.1. *Is the entropy of the network relevant or just a second-order theoretical point?*

We have shown in theorem 7 and in section 5.2 that the deviations $L - \hat{L}$ were bounded above by a term depending on the *entropy* of the network, and not only on the number of parameters. We now experiment this element as follows:

– generate randomly a bayesian network,

– randomly draw a data set D with this bayesian network,

– for many values of k:

    - generate randomly many learners $l_1, \ldots, l_m$ with k parameters, with entropy $H_1, \ldots, H_m$;

    - learn (i.e. optimize $\hat{L}$) with each of these learners. Define $\hat{L}_i$ the empirical error of l_i.

- evaluate L for each of these learning; define L_i the generalization error of learner l_i.

- plot $L_i - \hat{L}_i$ as a function of H_i

We show the result with 10 nodes in Figure 4. Hence, this shows experimentally that, with a number of parameters fixed, the entropy term reflects well the complexity of the structure, whereas it doesn't appear in usual scores. This confirms the theoretical results.
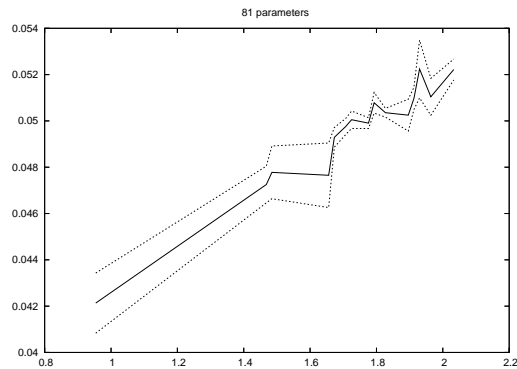

Figure 4. *X-ordinate: entropy. Y-ordinate: average of $L_i - \hat{L}_i$ ($\pm$ standard deviation). The positive correlation is clear*

8.2. *Are our algorithms for the computation of S and its derivative efficient ?*

We below perform experiments in order to validate the approximation of S and ∇S (sub-section 1) and then test them inside an optimization loop (sub-section 2).

8.2.1. *Preliminary experiments on the approximations of ∇S*

We compare below i) the exact method ii) the Monte-Carlo method iii) the quota method.

The experimental setup is as follows: 10 bayesian networks are generated; their gradients are computed with each method; we compute the relative error; we averaged the results. We experimented random bayesian networks: each node i has 2 parents randomly drawn among $[[i - 6, i - 3]]$ and 2 parents $i - 2$ and $i - 1$.

The results are the followings for the computation of the derivative. The sample size is the sample size of the approximate methods. These experiments have been ran on a pentium 4, 3.0 GHz.

Algorithm	Time	Relative error
nb nodes=20, sample size = 10000		
Exact	0.68± 0.07	0.
Monte-Carlo	0.04 ± 0.003	0.07 ± 0.02
Quotas	0.04 ± 0.005	0.01 ± 0.003
nb nodes=30, sample size = 10000		
Exact	1.85± 0.15	0.
Monte-Carlo	0.06± 0.004	0.11± 0.02
Quotas	0.06± 0.000	0.05± 0.03
nb nodes=50, sample size = 30000		
Exact	6.26± 0.33	0.
Monte-Carlo	0.29± 0.01	0.19± 0.04
Quotas	0.30± 0.01	0.17± 0.04

The results are:

– the exact method is validated (error 0 at each run);

– results are better for the quotas method than for the naive Monte-Carlo method, at least for a moderate number of parameters. For larger dimension (50 nodes, roughly 700 parameters), the quotas method is roughly equivalent to the Monte-Carlo method; as usually in the general case of quasi-Monte-Carlo methods, the case of large dimension is difficult (see e.g. (Srinivasan, 2000)).

8.2.2. *Optimization through the approximate computation of S and ∇S*

We have shown above that approximate methods are precise and fast. We now show that the whole optimization algorithm based on the approximate methods are reliable. The goal is to find optimal values of parameters of the BN for $\hat{L}$. We therefore plot the evolution of the (exact) objective function $\hat{L}$ when we use BFGS with the approximations of S and the approximations of ∇S.

The experimental setup is as follows:

– define $\hat{\hat{L}}$ the approximation of $\hat{L}$ by the quota method;

– optimize $\hat{\hat{L}}$ thanks to BFGS; increase the number of examples when

$$Var(\widehat{\nabla \hat{L}}) \geq \alpha \times ||\widehat{\nabla \hat{L}}||^2$$

for typically $\alpha = 0.1$ where

    - $\hat{L}$ is the objective function (see section 4.1);

    - $\widehat{\nabla \hat{L}}$ is the estimate of the gradient of $\hat{L}$ through the quota method;

    - $Var(\widehat{\nabla \hat{L}})$ is the variance of the estimate of $\nabla \hat{L}$ by the quota-method,

The structure of the bayesian networks used for learning is as shown in section 8.2.1. This generation is in favor of the exact method as the network has bounded

width. The results are presented in Figure 5. Thanks to the particular structure of the network, the exact method remains very efficient even for 30 nodes; the approximate method is however faster than the exact one. Note that the approximate method can deal with the general case, whereas the exact one is not tractable in the general case.
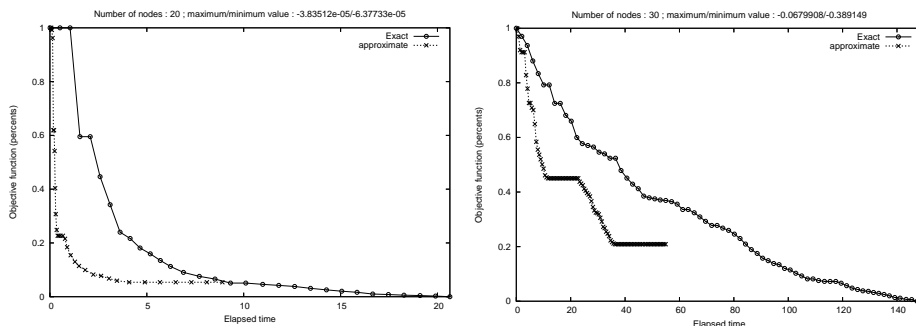

Figure 5. *Evolution of $\hat{L}$ as time increases, when i) BFGS uses the exact $\nabla \hat{L}$ and $\hat{L}$; ii) BFGS uses the approximate $\widehat{\nabla \hat{L}}$ and $\widehat{\hat{L}}$. First: 20 nodes. Second: 30 nodes. The structure and parameters are randomly drawn as explained in section 8.2.1. We see that the computations are much faster with the approximate methods*

These experiments have shown that the optimization is practical, and so we demonstrate the "proof of concept" of the method. In order to treat much larger bayesian networks, we can expect huge improvements as pointed out in 7.1.1 from adaptation of state of the art inference algorithms.

8.3. *Is the superiority of the optimization of $\hat{L}$ on the local method as shown in section 6.2 validated by practical experiments ?*

The experimental setup is as follows:

– randomly draw one generator G;

– randomly draw structure S for learning;

– generate n examples from G;

– learn a bayesian network with structure S on these examples;

– compute L with respect to the distribution associated to G.

In our experiments, all networks have size 10 nodes, and for the sake of statistical significance, 10 different generators were drawn and tested, and the random draws of the structures were paired (i.e., the 50 different S drawn for each generator are the same for all the generators). We tested n between 100 and 900. The results are presented in Figure 6.

We plot the difference between the generalization error and the generalization error of the generator (best possible error with this distribution). Zero error can then be achieved with a perfect structural learning algorithm, and perfect parameters. Here
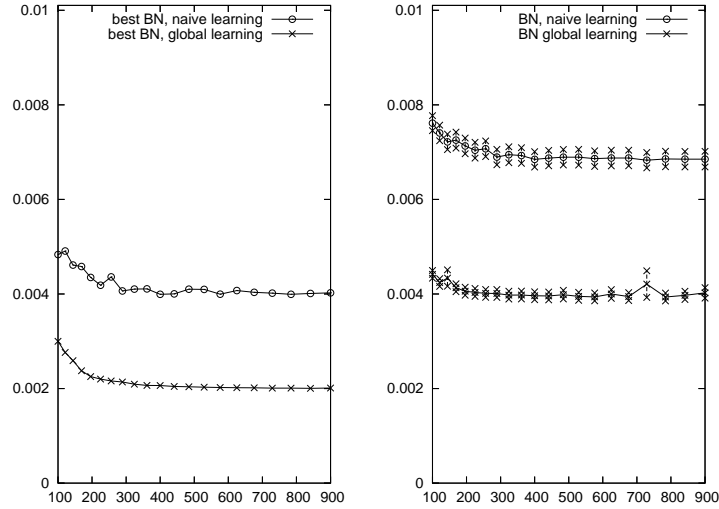
Figure 6. *X-coordinate: Number of examples. Y-coordinate: Error in generalization minus optimal generalization error ($L - L(g)$ where g is the generator). Results are averaged among 10 randomly drawn generators and 50 randomly drawn structures for learning*

we can experimentally see that the global optimization method divide by a factor of two the error, without any structural learning algorithms, which are generally very costly and suboptimal.

9. Conclusion

In this paper, we have proposed insights in bayesian network learning using statistical learning theory. We:

1) have proposed a criterion of quality L of an instanced bayesian network, and evidences of its relevance for some applications (section 1);

2) have proved bounds on covering numbers of bayesian networks (theorem 7, section 5);

3) have proposed scores for choosing between structures, thanks to a score taking into account the structural entropy of the network. Our score is related to BIC/AIC as its main term depends on the number of parameters, but it also uses the entropy. Multiple corollaries have been shown in section 6 and the relevance of the structural entropy has been shown in section 8.1, in spite of the traditional looseness of statistical-learning-theory bounds;

4) have proposed a paradigm for parameter-learning which is better than the traditional frequentist method for the criterion L (this is shown theoretically in section 6.2 and practically in section 8.3);

5) have proposed an algorithm with guaranteed universal consistency and almost sure convergence towards a structure with optimal size (theorem 8).

We then proposed new algorithms (section 7) in order to treat our loss function which is more complicated and expensive than the frequentist parametrization.

For parameter learning, the drawback of the loss function is the computational overhead for learning the parameters. However, thanks to an ad-hoc algorithm, it is tractable (see section 8) and in order to learn in larger bayesian networks, we can expect improvements using adaptations of inference algorithms(see remark in section 7.1.1).

For structure learning, the relevance of the entropy term in the estimation of the complexity of a structure is theoretically and experimentally shown (theorem 7 and section 8.1). More complete structural learning experiments, using a score taking into account the entropy of the structure, have yet to be conducted.

As for many works based on learning theory, a limit of this paper is that bounds are usually much too conservative. However, they are usually good indicators for penalty-terms design (e.g., weight decay works, in spite of the fact that theoretical upper bounds are very loose).

Using the full algorithm (for structure-learning and parameter-learning) is computationaly hard (yet tractable for small networks). However, there are practical and not-too-expensive applications of this paper:

– inclusion of a regularization based on R and $H(r)$ instead of R only; this can be done in usual algorithms just by modifying the penalty (e.g. our work suggests $R(1 + H(r)/\ln(1/\epsilon))$ instead of R, where ϵ is an estimate of the precision);

– parameter-learning, in particular:

    - our approach works even for datasets with unseen conjunctions of the parents of some nodes;

    - our method includes latent variables without any special treatment; it avoids the use of the EM-algorithm or of any heuristic;

    - our objective-function is more natural when the goal is the estimation of a macroscopic quantity and not individual probability; typically, when estimating a probability that is not a function of a very small number of variables (e.g., overall fault probability when there are many differents possible faults, each of which being modelized by one binary random variable). Our objective function is not relevant when the goal is to make a difference between 10^{-3} and 10^{-7} for the probability of one single assignement of nodes (whereas the KL-divergence is efficient for this); it is relevant if the goal is to to know if some event has probability 0.3 or 0.5 or if the goal is the evaluation of the sum of many individual probabilites, none of them being predominant.

10. References

Akaike H., « Statistical predictor identification », *Ann. Inst. Statist. Math.*, vol. 22, p. 203-217, 1970.

Antony M., Bartlett P., *Neural network learning : Theoretical Foundations*, Cambridge University Press, 1999.

Borglet C., Kruse K., *Graphical Models - Methods for data analysis and Mining*, John Wiley and Sons, Chichester, UK, 2002.

Broyden C. G., « The convergence of a class of double-rank minimization algorithms, II: The new algorithm », *J. Inst. Maths. Applics.*, vol. 6, p. 222-231, 1970.

Byrd R. H., Lu P., Nocedal J., Zhu C., « A limited memory algorithm for bound constrained optimization », *SIAM J. Sci. Comput.*, vol. 16, n° 5, p. 1190-1208, 1995.

Cheng J., Bell D., Liu W., « An algorithm for bayesian network construction from data », *6th International Workshop on Artificial Intelligence and Statistics*, p. 83-90, 1997a.

Cheng J., Bell D., Liu W., « Learning belief networks from data : An information theory based approach », *sixth ACM International Conference on Information and Knowledge Management CIKM*, p. 325-331, 1997b.

Cheng J., Greiner R., Kelly J., Bell D., Liu W., « Learning Bayesian networks from data : An information theory based approach », *Artificial Intelligence*, vol. 137, p. 43-90, 2002.

Chickering D., « A Transformational Characterization of Equivalent Bayesian Network Structures », *Proceedings of the 11th Annual Conference on Uncertainty in Artificial Intelligence (UAI-95)*, Morgan Kaufmann, San Francisco, CA, p. 87-98, 1995.

Chickering D. M., « Learning equivalence classes of bayesian-network structures », *J. Mach. Learn. Res.*, vol. 2, p. 445-498, 2002a.

Chickering D. M., « Optimal structure identification with greedy search », *Journal of Machine Learning Research*, vol. 3, p. 507-554, 2002b.

Chickering D. M., Heckerman D., « Efficient Approximations for the Marginal Likelihood of Bayesian Networks with Hidden Variables », *Mach. Learn.*, vol. 29, p. 181-212, 1997.

Chickering D. M., Meek C., « Finding Optimal Bayesian Networks », *UAI*, p. 94-102, 2002.

Cooper G., Hersovits E., « A Bayesian method for the induction of probabilistic networks from data », *Machine Learning*, vol. 9, p. 309-347, 1992.

Cozman F., « Generalizing Variable Elimination in Bayesian Networks », *Proceedings of the Iberamia/Sbia 2000 Workshops (Workshop on Probabilistic Reasoning in Artificial Intelligence*, Editora Tec Art, Sao Paulo, p. 27-32, 2000.

Fletcher R., « A New Approach to Variable-Metric Algorithms », *Computer Journal*, vol. 13, p. 317-322, 1970.

Geiger D., Meek C., « Graphical Models and Exponential Families », *Proceedings of Fourteenth Conference on Uncertainty in Artificial Intelligence,* Madison, WI, Morgan Kaufmann, p. 156-165, August, 1998.

Goldfarb D., « A Family of Variable-Metric Algorithms Derived by Variational Means », *Mathematics of Computation*, vol. 24, p. 23-26, 1970.

Guo H., Horvitz E., Hsu W., Santos E., « A Survey of Algorithms for Real-Time Bayesian Network Inference », *Joint Workshop on Real- Time Decision Support and Diagnosis*, 2002.

Heckerman D., Geiger D., Chickering M., « Learning Bayesian networks : The combination of knowledge and statistical data », *Proceedings of the 10th Conference on Uncertainty in Artificial Intelligence*, 1994.

Kolmogorov A.-N., Tikhomirov V.-M., « ϵ-entropy and ϵ-capacity of sets in functional spaces », *Amer. Math. Soc. Translations*, vol. 17, p. 277-364, 1961.

Kschischang F. R., Frey B., Loeliger H.-A., « Factor graphs and the sum-product algorithm », *IEEE Trans. Inform. Theory*, vol. 47, n° 2, p. 498-519, 2001.

Langford J., Blum A., « Microchoice Bounds and Self Bounding Learning Algorithms », *Mach. Learn.*, vol. 51, n° 2, p. 165-179, 2003.

Lauritzen S., Spiegelhalter D., « Local computations with probabilities on graphical structures and their applications to expert systems », *J. Royal Statistical Society B*, vol. 50, p. 157-224, 1988.

Meza J. C., « OPT++: An Object-Oriented Class Library for Nonlinear Optimization », 1994.

Meza J., Hough P., Williams P., « Opt++ », 1994.

Naim P., Wuillemin P.-H., Leray P., Pourret O., Becker A., *Réseaux bayésiens*, Eyrolles, 2004.

Nakamura A., Schmitt M., Schmitt N., Simon H., « Inner Product Spaces for Bayesian Networks », *Journal of Machine Learning Research*, vol. 6, p. 1383-1403, 2005.

Pearl J., « Causality: models, reasonings and inference », *Cambridge University Press*, 2000.

Rissanen J., « Modeling by Shortest Data Description », *Automatica*, vol. 14, p. 465-471, 1978.

Robert C., « The Bayesian Choice : a decision theoric motivation », *Springer, New York*, 1994.

Schwartz G., « Estimating the dimension of a model », *The annals of Statistics*, vol. 6, p. 461-464, 1978.

Shanno D. F., « Conditioning of Quasi-Newton Methods for Function Minimization », *Mathematics of Computation*, vol. 24, p. 647-656, 1970.

Spirtes P., Glymour C., Scheines R., « Causation, prediction, and search », 1993.

Srinivasan A., « Low-discrepancy Sets for High-Dimensional Rectangles: A Survey », *Bulletin of the European Association for Theoretical Computer Science*, vol. 70, p. 67-76, 2000.

Vapnik V., *The Nature of Statistical Learning Theory*, Springer, N.Y., 1995.

Vapnik V., Chervonenkis A., « On the uniform convergence of relative frequencies of events to their probabilities », *Theory of probability and its applications*, vol. 16, p. 264-280, 1971.

Vidyasagar M., « A Theory of Learning and Generalization », *Springer*, 1997.

Wocjan P., Janzing D., Beth T., « Required sample size for learning sparse bayesian networks with many variables », *LANL e-print cs.LG/0204052*, 2002.

Zhu C., Byrd R., P.Lu, Nocedal J., « L-BFGS-B: a limited memory FORTRAN code for solving bound constrained optimization problems », 1994.