The IBMAP approach for Markov network structure learning

Federico Schlüter · Facundo Bromberg · Alejandro Edera

Received: date / Accepted: date

Abstract In this work we consider the problem of learning the structure of Markov networks from data. We present an approach for tackling this problem called IBMAP, together with an efficient instantiation of the approach: the IBMAP-HC algorithm, designed for avoiding important limitations of existing independence-based algorithms. These algorithms proceed by performing statistical independence tests on data, trusting completely the outcome of each test. In practice tests may be incorrect, resulting in potential cascading errors and the consequent reduction in the quality of the structures learned. IBMAP contemplates this uncertainty in the outcome of the tests through a probabilistic maximum-a-posteriori approach. The approach is instantiated in the IBMAP-HC algorithm, a structure selection strategy that performs a polynomial heuristic local search in the space of possible structures. We present an extensive empirical evaluation on synthetic and real data, showing that our algorithm outperforms significantly the current independence-based algorithms, in terms of data efficiency and quality of learned structures, with equivalent computational complexities. We also show the performance of IBMAP-HC in a real-world application of knowledge discovery: EDAs, which are evolutionary algorithms that use structure learning on each generation for modeling the distribution of populations. The experiments show that when IBMAP-HC is used to learn the structure, EDAs improve the convergence to the optimum.

Keywords Markov network · structure learning · independence tests · knowledge discovery · EDAs

F. Schlüter, F. Bromberg, A. Edera
Lab. DHARMa of Artificial Intelligence,
Departamento de Sistemas de información,
Facultad Regional Mendoza, Universidad Tecnológica Nacional, Argentina.
Tel.: +54-261-5244566
E-mail: {federico.schluter,fbromberg,aedera}@frm.utn.edu.ar

1 Introduction

We present in this work the IBMAP (Independence-Based Maximum a Posteriori) approach for robust learning of Markov network structures from data, together with IBMAP-HC, an efficient hill-climbing instantiation of the approach. Markov networks and Bayesian networks belong to the family of *probabilistic graphical models* [19], a computational framework for compactly representing joint probability distributions. There is a large list of applications of graphical models in a wide range of fields, such as in the areas of computer vision and image analysis [27,23], computational biology [15], biomedicine [38, 41], evolutionary computation [20,3,36], among many others. Probabilistic graphical models are composed by an undirected (Markov networks) or directed (Bayesian networks) graph G, and a set of numerical parameters Θ. Each node in the graph G represents a random variable of the domain, and the edges encode conditional independences among them. For this reason, the graph G is also called the *independence structure* of the distribution. The importance of these independences is that they factorize the joint distribution over the domain variables into factors over subsets of variables, resulting in important reductions in the space complexity for representing the distribution [17]. The structure can be obtained from the knowledge of a human expert, but commonly it is hard to obtain, and not always enough to design an accurate structure. An interesting problem that has attracted considerable attention is learning automatically the independence structure from categorical data drawn from an unknown probability distribution [19,42]. However, this problem is known to be in general an NP-hard problem, since the number of structures grows super-exponentially [10]. For Markov network structure learning, there are two broad approaches mainly considered in the literature: *score-based* [14,27,22,16], and *independence-based* (also known as constraint-based) algorithms [39,9,25,4]. On the one hand, the score-based algorithms combine a measure of the goodness of fit of each structure to the data with a metric for the complexity of the structure; for instance, to maximize the log-likelihood of the maximum likelihood parameters given the structure. Recently, several efficient instantiations of this approach have been developed, such as [32,13,40]. On the other hand, the independence-based algorithms proceed by performing statistical independence tests on data, and based on the outcome of the tests discards all structures inconsistent with the test. This approach is efficient, and correct under some assumptions, but in practice presents quality problems: one of the assumptions is the correctness of independence tests, which may not be true in practice when the amount of data is insufficient. It is important to mention that both score-based and independence-based approaches have been motivated by distinct learning goals. According to the existent literature [19], score-based approaches are better suited for the density estimation goal, that is, tasks where inferences or predictions are required [28]. In contrast, independence-based methods are better suited for other learning goals, such as feature selection for classification, or knowledge discovery [39,4,5]. IBMAP follows the independence-based approach for learning the structure of a Markov

network. Our approach has been designed to be more robust when the assumption of correctness of statistical tests is not valid. Instead of trusting the outcome of statistical tests on data, IBMAP considers explicitly the posterior probability of independences given the data. As explained in detail later on, these posteriors of tests are combined into the posterior of the whole structure (given the data), deciding on the output structure following the well-known maximum-a-posteriori approach. This clearly circumvents the cascading error of traditional independence-based algorithms, as the true structure is no longer discarded on an incorrect test, it only results in a lower posterior probability. With further tests, the posterior probability of the true structure may increase again. In order to evaluate the improvements in the quality of the structures produced by our approach, we performed detailed and systematic experiments on both synthetic datasets and real-world datasets. In all those cases we compared the structural errors of the structures learned by IBMAP-HC against those learned by representative state-of-the-art competitors: GSMN [8,9], and HHC-MN, a simple adaptation for Markov networks of an independence-based structure learning algorithm for Bayesian networks, called HHC [5]. We note that structural errors as quality measure is the most appropriate for knowledge discovery algorithms such as those using the independence-based approach. Additionally, we tested the performance of IBMAP-HC in a real-world application: *Estimation of Distribution algorithms* (EDAs) [30]. These evolutionary algorithms are able to solve problems that are known to be hard for traditional Genetic Algorithms [20]. EDAs are variations of the well-known evolutionary algorithms, that replace the crossover and mutation stages for generating a new population of solutions with a sampling of a probability distribution learned from the selected population. Our experiment in EDAs is motivated by the fact that the quality of structure learning is expected to influence the results of the optimization. This occurs because the structure learning step is made for each generation of the optimization, and the populations are generated by sampling from the distribution learned. The more accurate the structure learned, the more effective is the sampling for generating good solutions. In our experiment we tested IBMAP-HC in the Markovianity Optimization Algorithm (MOA) [36], a state-of-the-art EDA, based on Markov network structure learning. We show that MOA improves its convergence to the optimum when IBMAP-HC is used to learn the structure. The rest of this work is organized as follows. Section 2 presents an overview of the independence-based learning approach and motivates our contribution. Section 3 presents the IBMAP approach, and Section 4 details our IBMAP-HC algorithm. Section 5 shows our experiments on synthetic and real datasets, and Section 6 shows our experiments on EDAs. Finally, Section 7 summarizes this work, and poses several possible directions of future work. The paper also includes two appendices at the end.

2 Background

This section provides some background on Markov networks, defines the problem of structure learning, and motivates our independence-based approach. Hereon, we use capital letters to denote single random variables, and the sets of variables in bold. A Markov network representing an underlying distribution $P(\mathbf{V})$ over a domain of $n = |\mathbf{V}|$ random variables $\mathbf{V}$ consists in an undirected graph G, and a set of potential functions, defined by a set of numerical parameters Θ. The graph G is a map of the conditional independences in $P(\mathbf{V})$, and such independences can be read from the graph through *vertex separation*, considering that each pair of variables (X, Y) are said to be vertex separated by a set of variables $\mathbf{Z} \subseteq \mathbf{V} \setminus \{X, Y\}$ when every path between X and Y in G contains some node in $\mathbf{Z}$ [31]. The distribution $P(\mathbf{V})$ can be factorized into a product of *potential* functions $\phi_c(V_c)$ over the completely connected sub-graphs (a.k.a., *cliques*) V_c of its structure G [17], that is,

$$P(\mathbf{V}) = \frac{1}{Z} \prod_{c \in cliques(G)} \phi_c(V_c),$$

where Z is the *partition function*, a constant that normalizes the product of potentials. Such potential functions are parameterized by the set of numerical parameters Θ. The problem of structure learning takes as input a dataset D, which is assumed to be a representative sample of the underlying distribution $P(\mathbf{V})$. Commonly, D is structured in a tabular format, with one column per random variable in the domain $\mathbf{V}$, and one row per data point. The optimal solution of the problem is a perfect-map of $P(\mathbf{V})$ [31], that is, a structure that encodes all the dependences and all the independences present in $P(\mathbf{V})$. The closer to a perfect-map, the better is the structure learned, and the better is the resulting Markov network for representing $P(\mathbf{V})$. Independence-based algorithms learn a perfect-map by performing a succession of statistical independence tests, discarding at each iteration all structures inconsistent with the outcome of the test, and deciding on the tests to perform next based on the outcomes learned so far.

A statistical independence test is a statistic computed from D for testing if two random variables X and Y are conditionally independent, given some conditioning set of variables $\mathbf{Z}$; where X, Y and $\mathbf{Z}$ are disjoint subsets of the domain $\mathbf{V}$. This *independence assertion* is denoted by $\langle X \perp\!\!\!\perp Y | \mathbf{Z} \rangle$ (or $\langle X \not\perp\!\!\!\perp Y | \mathbf{Z} \rangle$ for the *dependence assertion*). The computational cost of a test is proportional to the number of rows in D, and the number of variables involved in the test. Examples of independence tests used in practice are Mutual Information [11], Pearson's χ^2 and G^2 [2], the Bayesian test [24], and for continuous Gaussian data the *partial correlation* test [39], among others. There are several advantages of independence-based algorithms. First, they can learn the structure without interleaving the expensive task of parameter estimation, reaching sometimes polynomial complexities in the number of statistical tests performed. If the complete model is required, the parameters

can be estimated only once for the learned structure. Another important advantage of such algorithms is that they are guaranteed to learn the correct structure of the underlying distribution, as long as the following assumptions hold: *i) graph-isomorphism*, i.e., the independences in the distribution can be encoded in an undirected graph; *ii)* the underlying distribution is *strictly positive*, i.e., $P(\mathbf{V}) > 0$, for every assignment of $\mathbf{V}$; and *iii)* the outcomes of tests are correct, i.e., the independences learned are true in $P(\mathbf{V})$. Unfortunately, the third assumption is rarely true in practice, as the number of contingency tables for which a statistic has to be computed grows exponentially with the number of variables in the conditioning set of the test. Therefore, the effective dataset from which the statistic is computed decreases exponentially in size, thus degrading exponentially the quality of the statistics. When tests outcome incorrect independences, independence-based algorithms produce what is commonly called *cascade errors* [39] that not only discard the true underlying structure, but further confuse the algorithm in the test to perform next. Our approach tackles this main issue of independence-based algorithms by contemplating the uncertainty in the outcome of the tests through a probabilistic maximum-a-posteriori approach.

3 The independence-based MAP approach

We now describe the main contribution of this work: the IBMAP approach for Markov network structure learning. Our approach avoids the cascade errors of traditional independence-based algorithms that completely trust the outcome of the statistical tests. For this, the central idea of IBMAP is to pose the structure learning task as a maximum-a-posteriori problem, by computing the posterior probability of each possible structure given data. Formally:

$$G^\star = \arg\max_G \Pr(G \mid D). \tag{1}$$

In our approach, the posterior $\Pr(G \mid D)$ is computed by combining the outcome of a set of conditional independence assertions that determine the structure G. We call this set the *closure* of the structure. The remainder of this section describes how to use the closure for computing the posteriors $\Pr(G \mid D)$. Next, in Section 4, the IBMAP-HC algorithm is presented as an efficient instantiation of the MAP optimization. Let us first define formally the concept of a closure:

Definition 1 (Closure) Let G be an undirected independence structure of a positive graph-isomorph distribution $P(\mathbf{V})$. The *closure* of G is a set of conditional independence assertions, $\mathcal{C}(G) = \{c_i\}$, that are sufficient for determining G completely.

Given the above definition, it is possible to replace G by $\mathcal{C}(G)$ in Eq. (1), obtaining:

$$G^\star = \arg\max_G \Pr(\mathcal{C}(G) \mid D). \tag{2}$$

The posterior of the closure given data can be seen as a joint probability distribution over its individual independence assertions, given data. By applying the chain rule over the assertions in $\mathcal{C}(G)$, we obtain:

$$\Pr(\mathcal{C}(G) \mid D) = \prod_{c_i \in \mathcal{C}(G)} \Pr(c_i | c_1, \ldots, c_{i-1}, D). \tag{3}$$

To the best of the author's knowledge, no method exists for computing exactly the probabilities $\Pr(c_i|c_1, \ldots, c_{i-1}, D)$ of independence assertions conditioned on other independence assertions and data. A common approximation is to assume that all the independence assertions in the closure *are mutually independent*. This assumption is made implicitly by all the independence-based Markov network structure learning algorithms [34], because the statistical tests are used as a black box, only using data for deciding independence for each assertion c_i. The result of applying this approximation to Eq. (3) is the following expression:

$$\Pr(\mathcal{C}(G) \mid D) \approx \prod_{c_i \in \mathcal{C}(G)} \Pr(c_i \mid D),$$

which expressed in terms of logarithms to avoid underflow, results in the following expression that we call the *IB-score*:

$$\sigma(G) = \sum_{c_i \in \mathcal{C}(G)} \log \Pr(c_i \mid D). \tag{4}$$

For computing the posteriors of each term $\log \Pr(c_i \mid D)$ we use the Bayesian test of conditional independence [24, 25]. Finally, since the log function is monotonic, the maximization of the IBMAP approach can be expressed as:

$$G^\star \approx \arg\max_G \ \sigma(G). \tag{5}$$

Although computable, this expression is still intractable, as there are $2^{\binom{n}{2}}$ possible undirected structures in the search space.

4 The IBMAP-HC algorithm

This section presents our structure learning algorithm *IBMAP-HC*, our instantiation of the IBMAP approach. IBMAP-HC performs a heuristic hill-climbing search in the space of possible structures, thus its name. We first give a high-level overview of the algorithm, and then we describe some specific aspects, such as the closure used for computing the IB-score, the heuristic used for speeding-up the search, and the complexity of the overall algorithm. IBMAP-HC searches the structure with maximum IB-score, considering as neighboring structures all those structures that result from flipping only one edge (i.e., single-edge additions or deletions). Algorithm 1 presents its pseudocode. The algorithm has as input parameter a dataset D, used for computing the statistical independence tests. The search starts at line 1 by creating a

structure G with n nodes (the number of variables in the domain) and no edges. Then, the IB-score of G is computed in line 2 and saved in the variable *current-score*. The hill-climbing search starts in the loop of line 3. The loop iterates by calling the *select-next-structure* function at line 4 to select the neighbor of G with maximum score, which is saved in variable G'. Since the number of possible neighbor structures is $\binom{n}{2}$, this function is a heuristic for selecting the best neighbor, avoiding the expensive cost of computing the IB-score for all of them. This is explained in detail in Section 4.2. Then, in line 5 the score of the best neighbor is computed, and saved in the variable *neighbor-score*. The algorithm stops when the neighbor proposed does not improve the current score, a condition checked at line 6. If the termination criterion is not reached, the variables G and *current-score* are assigned with the values of the variables G' and *neighbor-score* in lines 9 and 10, and the process is repeated until a local optimum is found. For computing the IB-score σ of the candidate

Algorithm 1 IBMAP-HC (dataset D)

```
1: G ← empty structure with n nodes          // n is the domain size
2: current-score ← σ(G)
3: repeat
4:    G' ← select-next-structure(G, σ(G))     // see Algorithm 2 and Section 4.2
5:    neighbor-score ← σ(G')                  // see incremental computation in Section 4.1
6:    if neighbor-score ≤ current-score then
7:       return G                             // local maximum reached
8:    else
9:       G ← G'
10:      current-score ← neighbor-score       // an ascent in the hill-climbing search
```

structures (lines 2 and 5) we define a closure called the *Markov blanket closure*, presented in the next subsection. This closure has been designed to determine a structure with a number of independence tests which is quadratic in the number of variables in the domain.

4.1 Markov blanket closure

The *Markov blanket closure* is a closure set that follows Definition 1. This closure has been designed using the *Markov blanket* of a domain variable X, denoted here $\mathbf{B}_X$. In terms of graphs, the Markov blanket of X is defined as the set of all the nodes connected by an edge to the node of X in the structure [31,19], i.e., its adjacency set. In terms of independences, this allows to consider that X is conditionally independent of all its non-adjacent variables in the graph, given its Markov blanket. By this property, we define the Markov blanket closure as a set of closures that can be computed independently, one for each variable. Formally:

Definition 2 (Markov blanket closure) The *Markov blanket closure* of a structure G is a set of assertions determined by the union of a set $\mathcal{C}_X(G)$ of

independence and dependence assertions for each variable X in the domain $\mathbf{V}$, i.e.,

$$\mathcal{C}(G) = \bigcup_{X \in \mathbf{V}} \mathcal{C}_X(G), \tag{6}$$

where each $\mathcal{C}_X(G)$ is the union of two mutually exclusive sets of assertions:

$$\begin{aligned} \mathcal{C}_X(G) = &\Big\{ \langle X \not\perp\!\!\!\perp Y | \mathbf{B}_X \setminus \{Y\} \rangle \ : \ Y \in \mathbf{B}_X \Big\} \ \cup \\ &\Big\{ \langle X \perp\!\!\!\perp Y | \mathbf{B}_X \rangle \ : \ Y \notin \mathbf{B}_X \Big\}, \end{aligned} \tag{7}$$

that is, for each neighbor of X ($Y \in \mathbf{B}_X$) add a conditional dependence assertion between both variables conditioning on $\mathbf{B}_X \setminus \{Y\}$; and for each non-neighbor of X ($Y \notin \mathbf{B}_X$), add a conditional independence assertion between both variables conditioned on $\mathbf{B}_X$.

The following theorem states that the Markov blanket closure is indeed a closure, that is, it completely determines the structure G used to construct it.

Theorem 1 *Let G be an undirected independence structure of a positive graph-isomorph distribution $P(\mathbf{V})$. The* Markov blanket closure *of G is a set of conditional independence assertions that are sufficient for completely determining the structure G.*

Proof The formal proof of this theorem is presented in Appendix A.

This closure contains $n \times (n-1)$ assertions, a number which is quadratic in the size of the domain, that is, $n-1$ assertions for each of the n variables. This allows to decompose the computation of the IB-score of Eq. (4) in n independent *variable IB-scores*:

$$\sigma(G) = \sum_{X \in \mathbf{V}} \sigma_X(G), \tag{8}$$

where $\sigma_X(G) = \sum_{c_i \in \mathcal{C}_X(G)} \log \Pr(c_i \mid D)$. This decomposition permits to compute incrementally the score of any neighbor structure G', based on a previous computation of the score of a structure G. Given that G and G' differs by an edge (X, Y), the only blankets affected are $\mathbf{B}_X$ and $\mathbf{B}_Y$, requiring to recompute only σ_X and σ_Y, and reusing the $(n-2)$ remaining variable IB-scores. Consequently, the cost of computing $\sigma(G')$ from $\sigma(G)$ in line 5 of Algorithm 1 is reduced from $n \times (n-1)$ to $2 \times (n-1)$ tests, i.e., from $O(n^2)$ to $O(n)$ tests. Finally, for convenience of the explanation of the *select-next-structure* function in the next section, let us further decompose Eq. (8) considering that each variable IB-score $\sigma_X(G)$ is composed by $(n-1)$ terms $\sigma_{X,Y}(G)$, called *pairwise IB-scores*, as follows:

$$\sigma(G) = \sum_{X \in \mathbf{V}} \sum_{Y \in \mathbf{V} \setminus \{X\}} \sigma_{X,Y}(G). \tag{9}$$

According to Eq. (7), each pairwise IB-score $\sigma_{X,Y}$ is obtained by computing the following posterior from data:

$$\sigma_{X,Y}(G) = \left\{ \begin{array}{ll} \log \Pr(\langle X \not\perp\!\!\!\perp Y | \mathbf{B}_X - \{Y\} \rangle \mid D) & \text{if } (X,Y) \text{ is an edge in } G, \\ \log \Pr(\langle X \perp\!\!\!\perp Y | \mathbf{B}_X \rangle \mid D) & \text{otherwise.} \end{array} \right\}. \tag{10}$$

The next section shows the heuristic used by the *select-next-structure* function for reducing the computation time of finding the neighbor of a structure that maximizes the IB-score.

4.2 Heuristic for selecting the best neighbor structure

The naïve procedure for selecting the neighbor structure with maximum score would iterate over all the $\binom{n}{2}$ neighbors that differ in one edge, computing the IB-score of each one. For each neighbor, $n \times (n-1)$ statistical tests need to be performed for computing its IB-score using the Markov blanket closure, resulting in a total cost of $O(n^4)$ tests for each ascent in the hill-climbing search. By incrementally computing the IB-score of each neighbor, the cost of each ascent still results in a cost of $2 \times (n-1)$ statistical tests for each structure, with a total cost of $O(n^3)$ tests for each ascent. In order to reduce this expensive computation time, IBMAP-HC uses a heuristic that estimates the optimal neighbor without a single test computation, i.e., a cost of $O(1)$ test computations. The *select-next-structure* function is shown in Algorithm 2.

Algorithm 2 select-next-structure (G, $\sigma(G)$)

1: $(X^*, Y^*) \leftarrow \underset{(X,Y) \in (\mathbf{V} \times \mathbf{V}), X \neq Y}{\arg\min} \sigma_{X,Y}(G) + \sigma_{Y,X}(G)$
2: $G' \leftarrow G$ with (X^*, Y^*) flipped
3: **return** G'

It has as input parameter the current structure G and its corresponding score $\sigma(G)$, which at this point is already computed. The function first selects in line 1 the "optimal" pair (X^*, Y^*) as the least accurate edge (or absence of edge) in the current structure G. It can be done by representing $\sigma(G)$ as a data structure which contains the $n \times (n-1)$ pairwise scores $\sigma_{X,Y}(G)$, using the decomposable form of Eq. (9). Then, the best neighbor G' is constructed in line 2 as a copy of G with the pair (X^*, Y^*) flipped, and this is returned. To understand the minimization shown in line 1 of Algorithm 2, note that the number of neighbors differing by one edge is the same than the number of different pairs of variables (X, Y), i.e., $n \times (n-1)/2$ pairs. From this point of view, Eq. (9) can be seen as a sum of two pairwise IB-scores per each pair of variables, resulting in the following expression of the IB-score:

$$\sigma(G) = \sum_{(X,Y) \in \mathbf{V} \times \mathbf{V}, X \neq Y} \sigma_{X,Y}(G) + \sigma_{Y,X}(G). \tag{11}$$

With this form of $\sigma(G)$, it is clear that the minimization finds the pair (X^*, Y^*) whose contribution to $\sigma(G)$ is the smallest. The assumption made by the heuristic is that the structure resulting from flipping (X^*, Y^*) would be similar than maximizing the IB-score among the neighboring structures. As explained in Section 4.1, for incrementally computing $\sigma(G')$ from $\sigma(G)$ only $\sigma_X(G')$ and $\sigma_Y(G')$ need to be recomputed. The approximation made in the minimization consists in assuming that $\sigma_X(G') \approx \sigma_{X,Y}(G')$, and $\sigma_Y(G') \approx \sigma_{Y,X}(G')$, ignoring the remaining terms $\sigma_{X,W}(G')$ and $\sigma_{Y,W}, W \subseteq \mathbf{V} \setminus \{X, Y\}$. This is based in the fact that, from G to G', it is expected a strong change in the terms $\sigma_{X,Y}$ and $\sigma_{Y,X}$, since the posterior of dependence is used in one structure, and the posterior of independence is used in the other. In contrast, the terms ignored are assumed to have a mild change, because only the Markov blanket of X and Y has a change, and therefore these assertions only vary in the conditioning set. The approximation is possible because the pairwise IB-scores corresponding to the flipped edge $\sigma_{X,Y}(G')$ and $\sigma_{X,Y}(G)$ are complementary in both structures G and G', since the posterior of independence and the posterior of dependence sums 1. It allows to estimate $\sigma_{X,Y}(G')$ from the same pairwise IB-score $\sigma_{X,Y}(G)$, without a single test computation. This estimation is made implicitly by the minimization. This heuristic assumes that the ignored terms should have a minimal impact in the search for the optimal neighbor. This is of course an approximation, and only empirical results may shed light on its effectiveness. In the worst case, the approximation would result in the selection of a sub-optimal neighbor. This, however, is not different from many optimization algorithms that follow sub-optimal paths (e.g., the well-known Metropolis-Hastings search algorithm that may follow a sub-optimal neighbor according to its proposal distribution). Given the complexity of the problem, the impact of this approximation can only be assessed empirically. Later experiments show that despite this approximation, our approach is useful for avoiding the cascade effect of traditional independence-based algorithms, outperforming always the state-of-the-art algorithms when data are scarce. Additionally, Appendix B presents empirical measurements of the landscape of the IB-score for several synthetic datasets, showing that in most cases, our structure selection strategy finds nearly optimal scores.

4.3 Complexity of IBMAP-HC

This section summarizes the resulting computational cost of the whole algorithm using the hill-climbing search, the Markov blanket closure, and the *select-next-structure* function. To begin, the most expensive operation of the algorithm is the computation of the IB-score of the initial structure at line 1 of Algorithm 1, which is computed non-incrementally, using the $n \times (n-1)$ tests of the Markov blanket closure; this is a cost of $O(n^2)$ tests. Next, in the main loop of Algorithm 1, calling the *select-next-structure* function has a cost of $O(1)$, and the incremental computation of $\sigma(G')$ at line 5 requires to compute $2 \times (n-1)$ tests; this is a cost of $O(n)$. Finally, denoting by M the number

of ascents until termination, the overall computational cost of the algorithm is $O(n^2 + Mn)$. Since M can be obtained only empirically, the experimental section shows measurements of M on different scenarios, proving empirically that M is not a source of an extra degree in the complexity because it grows sub-linearly with n, resulting in an overall computational complexity of $O(n^2)$ statistical tests.

5 Experimental results

This section describes several experiments on synthetic and real datasets for testing empirically the robustness of our approach IBMAP, and the efficiency of our algorithm IBMAP-HC. We report a detailed and systematic experimental comparison between IBMAP-HC and state-of-the-art independence-based structure learning algorithms. We show a comparison of the quality of structures learned by our solution, against the quality of structures learned by GSMN [9], a state-of-the-art independence-based algorithm in terms of quality. We introduce also a competitor called HHC-MN as an adaptation for learning the structure of Markov networks of the HHC algorithm [5], a state-of-the-art independence-based algorithm for learning Bayesian networks. For comparing all the algorithms on the same ground, we ran all of them using the Bayesian test [24] as statistical independence test. The GSMN algorithm learns a structure by finding the Markov blanket of each variable of the domain with the GS algorithm [26], and then the solution structure is constructed by adding an edge between each variable and the variables found in its Markov blanket. The GS algorithm learns the Markov blanket of a variable X in two phases: the *grow* and *shrink* phases. During the grow phase, the algorithm increases the tentative Markov blanket with every variable Y that is found dependent on X, conditioning on the currently tentative Markov blanket. At the end of this phase, the tentative Markov blanket contains all members of the true Markov blanket, but potentially includes some false positives that are non-members. These false positives are removed during the shrink phase, where variables found independent of X conditioned on the current Markov blanket are removed from this set. At the end of this phase, the tentative Markov blanket matches the true Markov blanket, under the assumption of correctness of tests. The computational complexity of this algorithm is $O(n^2)$ in the number of independence tests for discovering the structure. The HHC algorithm learns the structure by learning the set of parents and children (PC) of each variable through the interleaved HITON-PC with symmetry correction algorithm [6, 4]. The pseudo-code of this algorithm can be seen at [4] (Figure 6, page 192). For learning the PC of a variable X, this algorithm starts with an empty candidate PC set, ranking the variables by priority for inclusion in the candidate set by unconditional dependence with X, and discarding the variables found unconditionally independent with X. Then, the algorithm utilizes an inclusion heuristic function that accepts each variable into the candidate PC set. If any variable inside the candidate set becomes independent with X given some

subset of the candidate set, then the algorithm removes that variable from the candidate set and never considers it again. The inclusion function and the elimination strategy are iterated interleaved until there are no more variables to examine for inclusion. The complexity of the HITON-PC is $O(n2^{\tau})$, where τ is the largest size of the PC set found, and the complexity of HHC is $O(n^2 2^{\tau})$, because HITON-PC is executed for each variable of the domain. For Markov networks, the equivalent of the PC of a variable are its neighbors, which corresponds to its Markov blanket. It is therefore expected that HITON-PC learns the Markov blanket of a Markov network, and thus it can be used as part of HHC to learn the undirected structure. This fact is not proven analytically here, but confirmed empirically for all the cases considered in this section. To get a Markov network learning algorithm we simply omit the final step of HHC that orients the edges to obtain the Markov blanket from the PC set, denoting the resulting algorithm by HHC-MN. The three following subsections describe our experiments over synthetic (Sections 5.1 and 5.2) and real datasets (Section 5.3).

5.1 Synthetic data experiments: random underlying structures

A first set of experiments was conducted on synthetic datasets, generated by using a Gibbs sampler on randomly generated Markov networks (structure plus parameters). This allows a systematic and controlled study, and provides datasets with known underlying structures to control the complexity of the problem, and to better assess the quality of the structures learned by each algorithm. For measuring the structural errors of the structures learned, we report the *Hamming distance* between the learned structure and the underlying one, i.e., the sum of false positive and false negative edges of the learned structure. Another quality measure that we use in this work for assessing the structures learned, is the well known F-measure, a harmonic mean of precision and recall quality measures, commonly used in the information retrieval community. Precision indicates how good was the algorithm in learning correct independences (that is, the relation between the true independences that were found, over all independences found by the algorithm). Instead, recall indicates how good was the algorithm in learning independences, but over all the correct independences present in the real structure (that is, the relation between the correct independences that were found, over the total of independences in the underlying structure). Then, the F-measure is computed as follows:

$$\text{F-measure} = \frac{2 \times precision \times recall}{precision + recall}.$$

Additionally, at the end of this section, we show the runtime of our experiments, in order to discuss the computational complexities of the competitor algorithms. The synthetic random Markov networks were generated for domains of $n \in \{100, 200, 500\}$ binary variables. For each domain size, 10 random networks were generated for increasing connectivities $\tau \in \{1, 2, 4, 8\}$, by

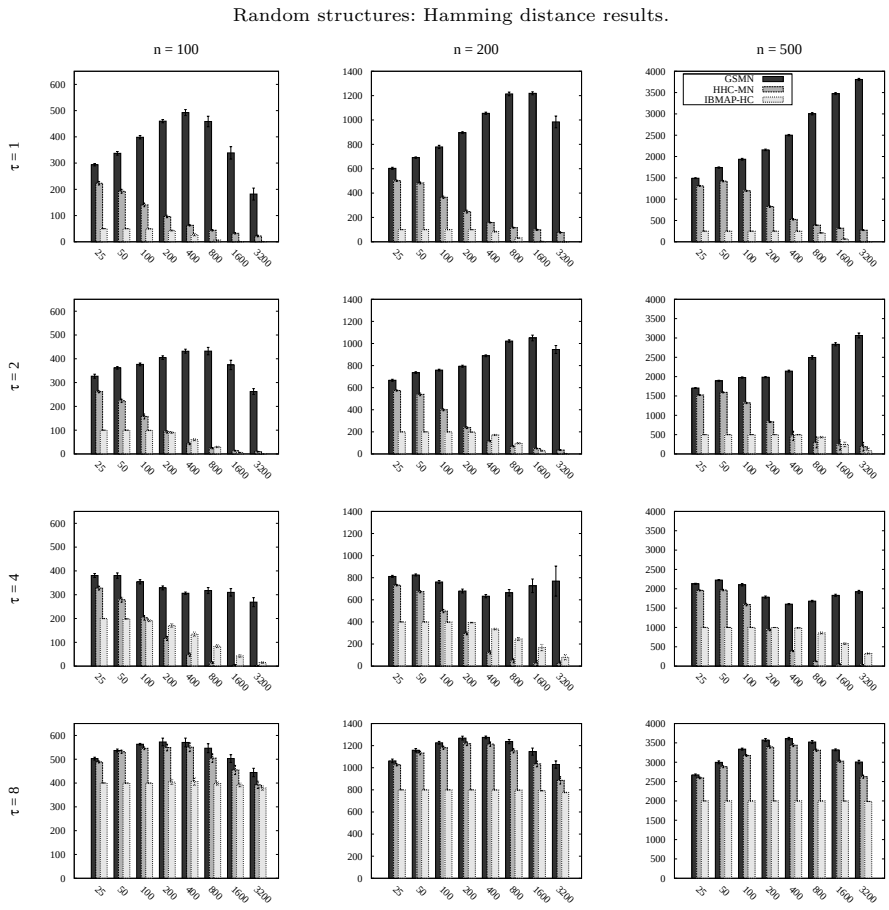

Fig. 1 Mean and standard deviation over 10 repetitions of the Hamming distance of the models learned by algorithms GSMN (black bars), HHC-MN (gray bars), and IBMAP-HC (light gray bars) for increasing sizes of random synthetic datasets, domain sizes $n = 100$ (first column), $n = 200$ (second column), and $n = 500$ (third column), and $\tau \in \{1, 2, 4, 8\}$ in the rows.

considering as edges the first $n\tau/2$ variable pairs of a random permutation of the set of all variable pairs. It is worth mentioning that with increasing values of τ, it is increasingly difficult to learn the structure. Given these Markov networks, we report the quality of structures learned by GSMN, HHC-MN, and IBMAP-HC using portions of each dataset with increasing number of datapoints $D \in \{25, 50, 100, 200, 400, 800, 1600, 3200\}$, for each (n, τ) combination. The independence structure determines the factorization of the distribution into potential functions over subset of variables, one per clique in the structure. To determine a complete model we must determine the numerical parameters that quantify these potential functions. For the datasets generated to correctly and strongly represent the direct dependencies encoded by the

edges, we considered in these experiments pairwise cliques for the factorization of the models, that is, two-variable factors $\phi(X, Y)$ for each edge in the random structure generated, and set the numerical parameters so that the correlation between them is strong. For that, we forced the parameters to result in a log-odds ratio of each pairwise factor $\varepsilon_{X,Y} = \log\left(\frac{\phi(X=0,Y=0)\phi(X=1,Y=1)}{\phi(X=0,Y=1)\phi(X=1,Y=0)}\right)$ to be equal to 1.0 for all edges (see [2]). This results in an equation over the values of the potential function with 4 unknowns. We then randomly chose 3 parameters in the range [0, 1], and solved for the remaining one. Figures 1

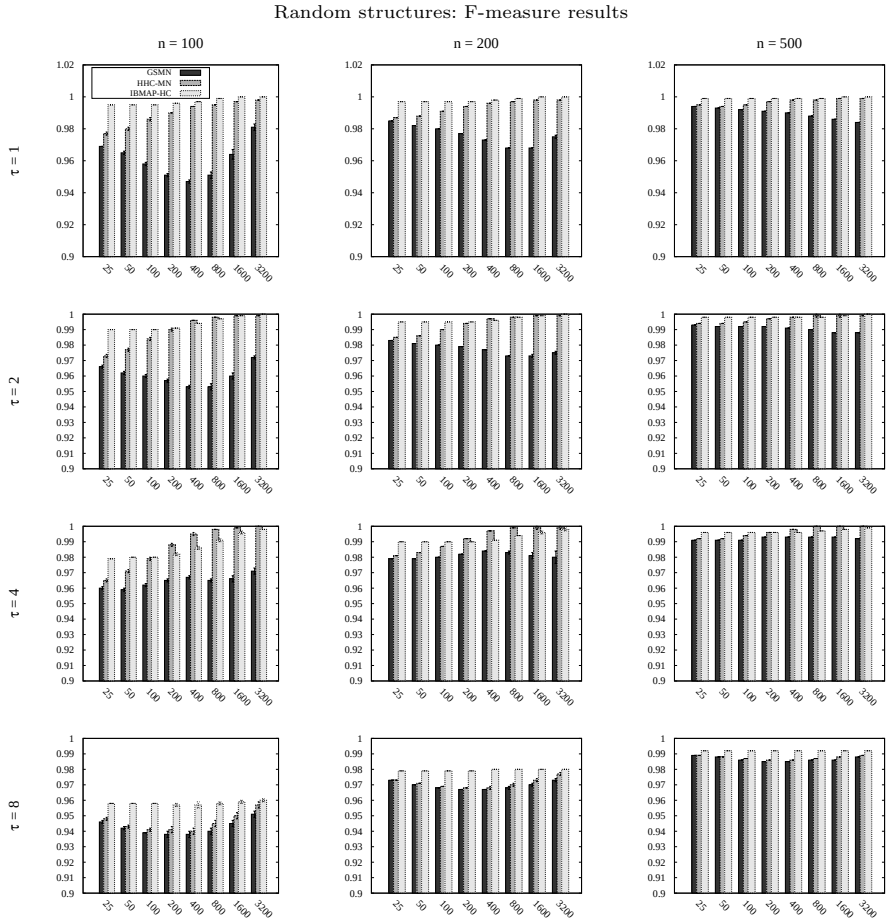

Fig. 2 Mean and standard deviation over 10 repetitions of the F-measure of the models learned by algorithms GSMN (black bars), HHC-MN (gray bars), and IBMAP-HC (light gray bars) for increasing dataset sizes of random synthetic datasets, domain sizes $n = 100$ (first column), $n = 200$ (second column), and $n = 500$ (third column), and $\tau \in \{1, 2, 4, 8\}$ in the rows.

and 2 show the mean values and standard deviations over the ten repetitions

of the Hamming distances and F-measure for the structures learned by the algorithms considered, respectively. The plots are ordered by columns for different n values, and by rows for different τ values. As expected, the results show that for all the algorithms, the more complex the underlying structure (determined by n and τ), the larger is the number of structural errors for any value of D used. It can be seen that for any algorithm and for any fixed value of D, the amount of errors grows with n (different columns), and also it grows with τ (different rows). Since GSMN and HHC-MN follow the traditional independence-based approach, it is expected for them to obtain very good qualities when data are sufficient, i.e., those cases with larger values of D and lower values of τ. The figures show clearly that both, IBMAP-HC and HHC-MN always learn structures with qualities significantly better (lower Hamming distance, and higher F-measure) than that of GSMN. For all the cases of n and τ, GSMN has the slowest convergence in D to reduce the structural errors. For the selected domain sizes, GSMN tends to add many false positives in the grow phase, which requires the shrink phase to perform unreliable tests involving many variables. It produces numerous cascade errors. In the case of HHC-MN, it can be seen that the structural errors are reduced significantly with respect to GSMN. These improvements are obtained by the use of its elimination strategy, as well as the interleaving of the inclusion heuristic function with the elimination strategy. When compared to IBMAP-HC, the latter always outperforms HHC-MN in terms of structural errors, except in the following specific cases:

- $\tau = 2, n \in \{100, 200, 500\}, D \in \{400, 800\}$
- $\tau = 4, n \in \{100, 200, 500\}, D \geq 200.$

In the above cases the data seem to be sufficient for HHC-MN to outperform our algorithm IBMAP-HC. This is because for $\tau < 8$ the underlying structures have not a dense topology, and the elimination strategy results to be very efficient. In contrast, for the case of $\tau = 8$, the data are not sufficient for HHC-MN to work as well, due to the exponential size of tests required in the elimination strategy. In this extreme case, the conditioning sets are at average of 8 variables, and in those cases the tests require larger amounts of data to be reliable. In general, the figures confirm that IBMAP-HC always outperforms significantly the competitors when data are scarce ($D \leq 100$). This confirm our hypothesis that the probabilistic approach of IBMAP avoids the cascade effect of traditional independence-based algorithms. Also, when the data are sufficient ($D > 100$) the qualities obtained are very competitive. Figure 3 shows the corresponding running times of the same experiment, expressed in milliseconds. To give the times more meaning, take into account that all our experiments were performed on an AMD Athlon(tm), with 3.0 GHz and 4 GB of main memory. Our results show clearly that GSMN is the more expensive algorithm in all the cases of $\tau \in \{1, 2, 4\}$. This is because it tend to add many false positives in the grow phase, and then the shrink phase require to perform tests that contains many variables, which is a source of extra computational cost. There are some extreme cases where IBMAP-HC is

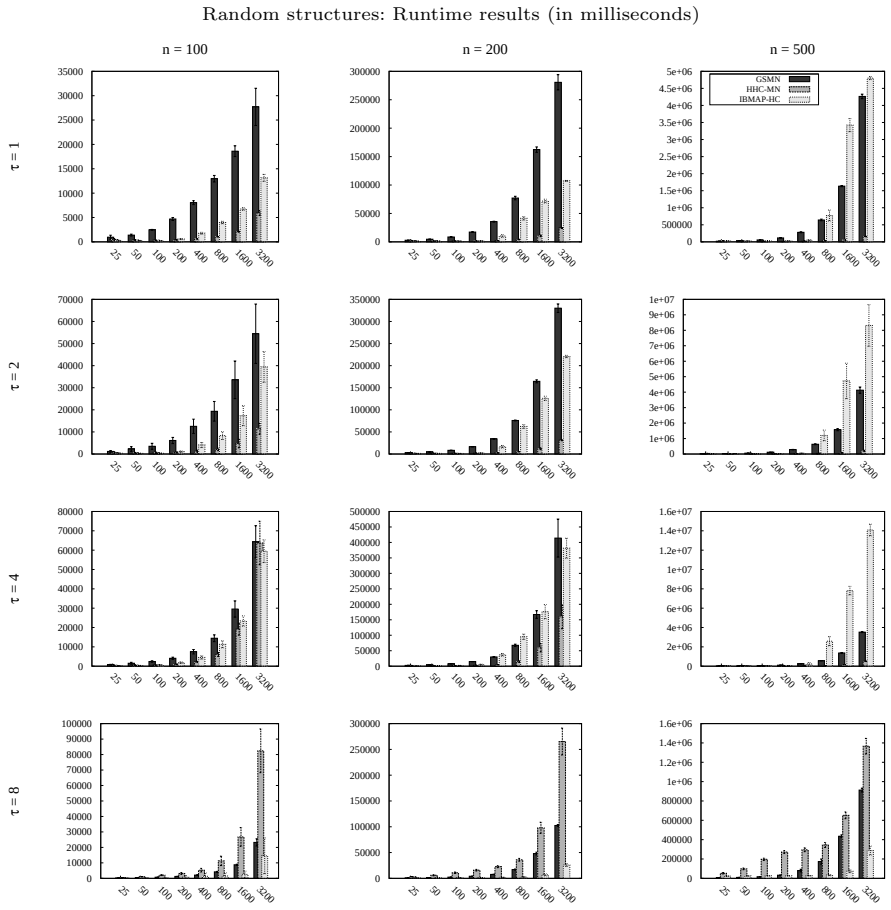

Fig. 3 Mean and standard deviation over 10 repetitions of the runtime required by algorithms GSMN (black bars), HHC-MN (gray bars), and IBMAP-HC (light gray bars) for increasing dataset sizes of random synthetic datasets, domain sizes $n = 100$ (first column), $n = 200$ (second column), and $n = 500$ (third column), and $\tau \in \{1, 2, 4, 8\}$ in the rows.

more expensive than GSMN, such as $n = 500$, $\tau \in \{1, 2, 4\}$, and $D \geq 800$. In those cases, the hill-climbing search of IBMAP-HC seem to be the more expensive alternative. HHC-MN is the algorithm that requires lowest computation time for the cases of $\tau \in \{1, 2, 4\}$, and $D \geq 200$. This is because the inclusion heuristic interleaved with the elimination strategy is really effective when the underlying structure has a low value of τ, and D is sufficiently large to obtain more reliable tests. In these situations, the algorithm converge to the termination criterion quickly. Instead, in the case of $\tau = 8$ (last row), HHC-MN is the most expensive algorithm. This is due to the exponential cost of the elimination strategy, that performs a test for all the subsets of the current conditioning set, which in this case is 8, on average. To conclude this section, we show an additional experiment to confirm empirically

that IBMAP-HC achieves polynomial time complexities with the number of random variables in the domain, as stated in Section 4.3. This is shown by Figure 4, that presents measurements of M (number of ascents in the hill-climbing search) for increasing problem sizes n. Such results were obtained for datasets generated in the same way as the previous experiments. The figure shows the average values of M over ten repetitions, for problems with increasing values of $n \in \{4, 12, 16, 20, 24, 30, 50, 75, 100, 200, 500\}$ in the X-axis, $D = 1000$, and a line for each $\tau \in \{1, 2, 4, 8\}$, indicating that M (Y-axis) grows sub-linearly. We omit results for different D values because they are similar.

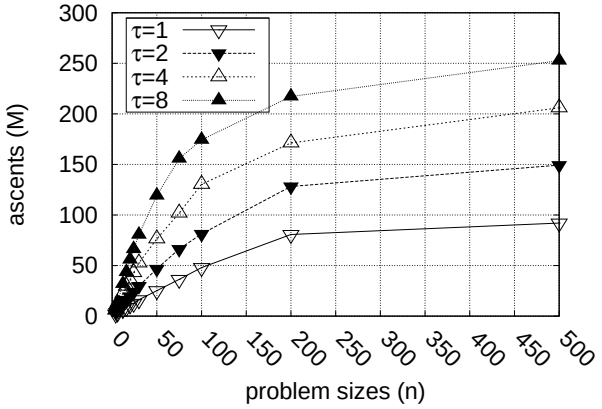

Fig. 4 Measurements in the number of ascents M (Y-axis) in the hill-climbing search of IBMAP-HC for increasing values of n (X-axis), and $\tau \in \{1, 2, 4, 8\}$, $D = 1000$.

5.2 Synthetic data experiments: Ising models

A second set of experiments over synthetic datasets were conducted over underlying structures with a different topology: the Ising spin glasses models, that are mathematical models of ferro-magnetism in statistical mechanics, also used in the last decades in many other domains, such as computer vision applications [23]. Using such models as underlying structure, ten datasets were generated for random Ising models with $n \in \{100, 200, 500, 750\}$ binary variables. Figure 5 shows the results for ten different random repetitions. The graphs in this figure are ordered by rows for different n values, and showing the mean value and standard deviation of the Hamming distance, the F-measure and the runtime in the first, second and third columns, respectively. These figures show clearly that both, IBMAP-HC and HHC-MN always learn structures with lower Hamming distance, and higher F-measure than GSMN (first and second column). In all the cases, the GSMN algorithm has the slowest convergence in D to reduce the structural errors among the three algorithms. With respect to HHC-MN, it can be seen that it has always lower structural quality

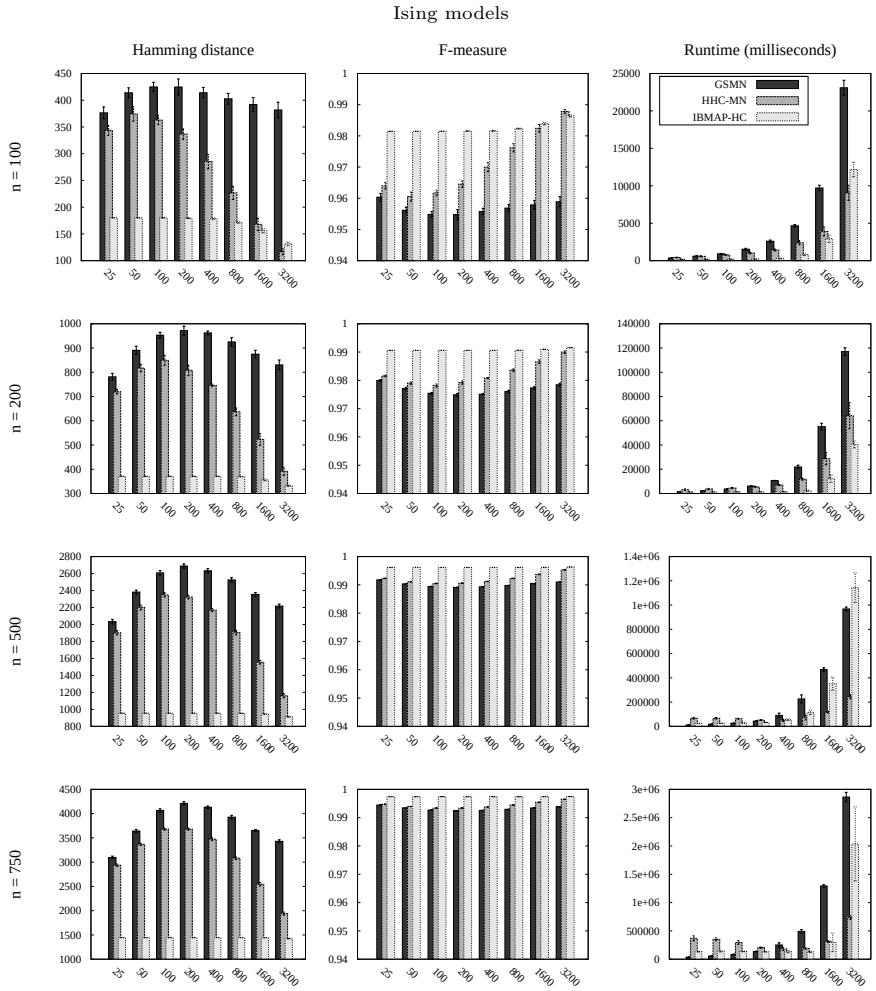

Fig. 5 Mean and standard deviation over 10 repetitions of the Hamming distance (first column), F-measure (second column) and runtime (third column) of algorithms GSMN (black bars), HHC-MN (gray bars), and IBMAP-HC (light gray bars) for increasing dataset sizes of Ising synthetic datasets, and domain sizes $n \in \{100, 200, 500, 750\}$ in the rows.

than IBMAP-HC, except in the specific case of $n = 100$, $D = 3200$, where the data seem to be sufficient for HHC-MN to improve the quality of IBMAP-HC. In general, the figures confirm that IBMAP-HC outperforms significantly the competitors in terms of quality. This also confirm our hypothesis that the probabilistic approach of IBMAP avoids the cascade effect of the traditional independence-based algorithms. With regard to the computational complexity results (third column), Figure 3 shows the corresponding running times, expressed in milliseconds. The computer used for running these experiments was the same described in the previous section. These results show clearly that

GSMN is the more expensive algorithm for all the cases, except in the specific cases:

- $n \in \{200, 500, 750\}, D \leq 100$, where HHC-MN is the more expensive;
- $n \in \{500\}, D = 3200$, where IBMAP-HC is the more expensive.

For the rest of the cases, IBMAP-HC has the better runtime, except in the following cases, where HHC-MN has the better runtime:

- $n \in \{100, 750\}, D = 3200$;
- $n \in \{500\}, D \geq 800$.

The analysis of these runtimes is similar to the analysis of the previous section, with GSMN with an expensive cost, due to the large amount of expensive tests (many false positives in the conditioning set), HHC-MN with a very good performance when data are sufficient, and IBMAP-HC with the best performance when data are not sufficient ($D < 200$).

5.3 Benchmark datasets experiments

In this section we show our experiments on real-world benchmark datasets, obtained from the UCI Repositories of machine learning [1] and KDD datasets [18]. Since the underlying network is unknown in these datasets, it is not possible to compute neither the Hamming distance nor the F-measure. Instead, we utilize the *accuracy*, a quality measure that counts the number of conditional independences present in data, which are correctly encoded by the structure learned. This measure was used for the same purpose in other related works [9, 25, 7]. In contrast with other measures that evaluate the density of the complete probability distribution (e.g. the Conditional Marginal Log-Likelihood), the accuracy is better suited for the goal of learning of this work (knowledge discovery) because it evaluates specifically structural errors. The accuracy is defined as a normalized measure for counting the number of matches in a comparison of the independence queries that hold in a *test set*, and also hold in the structure learned from a *training set*. The conditional independences are read from the learned structure by vertex separation (see Section 2). If $\mathcal{T}$ denotes the set of all possible conditional independence queries over the set of domain variables $\mathbf{V}$, it is checked for how many queries $t \in \mathcal{T}$, t is independent (or dependent) in both the test set, and the learned structure from the training set. Then, the number of matches is normalized by $|\mathcal{T}|$. Unfortunately, the size of $\mathcal{T}$ is exponential, so the approximated accuracy is computed over a randomly sampled subset $\widehat{\mathcal{T}}$, uniformly distributed for each possible conditioning set size. In our experiments we used $|\widehat{\mathcal{T}}| = 100 \times \binom{n}{2}$, i.e., a hundred of conditional independence queries per conditioning set size. We conducted our experiment using 19 real-world datasets, listed in Table 1, column one. The datasets are sorted by domain size (n) in the second column. For each dataset D, we shuffled the data and then divided it into a training set for learning the structure (75%), and a test set for computing the accuracy (25%). The

table also shows information about the number of attributes (second column), and the number of datapoints available in the train and test sets (third and fourth column). For each dataset we used the train set as input to the GSMN, HHC-MN, and IBMAP-HC algorithms, and the accuracy obtained for the structure learned for each algorithm is shown in the fifth, sixth and seventh columns, respectively. For each dataset, the best performance among the three algorithms is indicated in bold. These results show that in 10 of 19 datasets IBMAP-HC resulted in better accuracy, 6 cases resulted in ties (2 with GSMN, 1 with HHC-MN, and 3 with both), and for the remaining cases, the best results are obtained by HHC-MN(2 cases) and GSMN (1 case). The cases where IBMAP-HC always outperforms it competitors are those with $n \geq 16$. In those cases, data seem to be scarce (see the third column). That is consistent with our results in synthetic datasets, where IBMAP-HC outperforms always its competitors when data are scarce.

		Train	Test	**accuracy**		
Dataset	n	D	D	**GSMN**	**HHC-MN**	**IBMAP-HC**
baloons	5	14	5	**0.950**	0.897	**0.950**
balance-scale	5	468	156	**0.516**	**0.516**	**0.516**
iris	5	112	37	0.695	**0.742**	0.736
lenses	5	17	6	**0.881**	0.875	**0.881**
hayes-roth	6	98	33	**0.516**	**0.516**	**0.516**
car	7	1295	432	0.629	0.641	**0.703**
monks-1	7	416	139	**0.905**	**0.905**	**0.905**
nursery	9	9719	3240	0.392	0.415	**0.649**
ecoli	9	251	84	0.523	0.591	**0.694**
machine	10	156	52	0.590	0.567	**0.679**
cmc	10	1104	368	**0.759**	0.711	0.726
tic-tac-toe	10	718	239	0.671	**0.684**	0.498
echocardiogram	13	45	15	0.696	**0.745**	**0.745**
crx	16	489	163	0.578	0.593	**0.609**
hepatitis	20	59	20	0.496	0.633	**0.796**
imports-85	25	144	28	0.368	0.377	**0.596**
flag	29	145	48	0.446	0.451	**0.803**
dermatology	35	268	53	0.234	0.265	**0.754**
bands	38	207	69	0.399	0.408	**0.546**

Table 1 Accuracy for several benchmark data sets. The structure is learned using a subsample called train set, and the accuracy is computed using the test set. For each evaluation measure, the best performance is indicated in bold.

6 IBMAP-HC for Estimation of Distribution Algorithms

In contrast to benchmark datasets that comes from arbitrary applications, we present now results of evaluating IBMAP-HC in a real-world application of knowledge-discovery: the *Estimation of Distribution algorithms* (EDAs) [30, 20]. These are variations of the well-known evolutionary algorithms, that perform the same *selection* and *variation* stages, but replace the *crossover* and

mutation stages with the *estimation* and *sampling* in the task of generating a new population. The former stage *estimate* a probability distribution from the current population, generating the next population by *sampling* from it (thus their name). In the *estimation* stage, EDAs estimate the probability distribution from the dataset corresponding to the current population. This is because they associate each gene to a random variable, each individual to a joint assignment of these variables, and the selected population to a sample of the distribution. The rationale for replacing crossover methods with estimation is that by estimating the distribution from the selected individuals, that is, those best fitted, the sampling stage would produce novel, yet well-fitted individuals. Recently, several Markov network based EDAs has been proposed to model the distribution of populations [33,3,35,36]. As a test-bed we considered the *Markovianity Optimization Algorithm* (MOA) [36]. This is a state-of-the-art MN-based EDA that learns the Markov network structure from the population using an efficient structure learning algorithm based on mutual information (MI), a simple independence-based structure learning algorithm, described in detail in the same work, and designed specifically for MOA. The sampling in MOA is conducted through a variation of a Gibbs sampler that requires only the structure of the model, avoiding the need to learn the model parameters. The implementation of MI in MOA takes advantage of experts information indicating the maximum number of neighbor variables that a variable can have, denoted here k. We tested MI for different values of k (results not shown here), observing great sensitivity of MI to its value. Our algorithm IBMAP-HC does not use such a parameter. In the experiments below we set the value of k for MI to be the closest to the true value, resulting in the best possible performance of MI, i.e., the strongest competitor for IBMAP-HC. We conducted experiments to compare IBMAP-HC as an

	MOA		**MOA'**	
n	D^*	f^*	D^*	f^*
15	50	267.50 (35.45)	50	202.50 (14.19)
30	200	1170.00 (94.87)	100	475.00 (42.49)
60	800	5200.00 (98.46)	200	1050.00 (52.70)
90	800	5560.00 (126.49)	400	2220.00 (63.25)
120	1600	11200.00 (871.53)	800	4400.00 (312.33)

Table 2 Results of MOA and MOA' (that uses IBMAP-HC) for the OneMax problem, for increasing problem sizes (rows) in terms of critical population size D^*, and mean and standard deviation over 10 repetitions of the number of fitness evaluations f^* required to obtain the global optimum. Lower values of D^* and f^* are better.

alternative structure learning within MOA, denoted MOA$'$, and denoting by MOA the original version that uses MI. The thesis is that a better structure learning algorithm improves the convergence of MOA, that is, the optimum is reached computing fewer evaluations of the fitness of individuals. Both versions were tested on two benchmark functions widely used in the EDA's literature:

Royal Road and *OneMax*, both bit-string optimization tasks, detailed in [29]. The reason these benchmark functions are widely used is that they are hard to optimize, because the fitness landscape is flat for large areas and then discontinuous. In the context of evolutionary algorithms these functions model each bit-string as a chromosome and each bit as a gene. In the Royal Road problem, the variables are arranged in groups of size γ. Its goal is to maximize the number of 1s in the string, but adding γ to the fitness count only when a group has all 1s, otherwise adding 0. For example, in the case of $\gamma = 4$, an individual 111110011111 is separated in the groups [1111] [1001] [1111], and only the first and third group contribute 4 to the fitness count, which in the example equals 8. The underlying independence structure that should be learned therefore contains cliques of size γ, one per group. In our experiments we used $\gamma = 1$ and $\gamma = 4$. The former is known in the literature as *OneMax*. In the example, the fitness is 10 for OneMax. Clearly, the optimal individual for both problems is 111111111111. In the experiments, MOA is iterated for

	MOA		**MOA'**	
n	D^*	f^*	D^*	f^*
16	100	545.00 (59.86)	50	337.50 (176.09)
32	400	3800.00 (210.82)	400	2140.00 (134.99)
64	800	9120.00 (252.98)	800	4440.00 (126.49)
92	1600	18400.00 (533.33)	800	5080.00 (500.67)
120	1600	31120.00 (822.31)	1600	9840.00 (386.44)

Table 3 Results of MOA and MOA' (that uses IBMAP-HC) for the Royal Road problem, for increasing problem sizes (rows) in terms of critical population size D^*, and mean and standard deviation over 10 repetitions of the number of fitness evaluations f^* required to obtain the global optimum. Lower values of D^* and f^* are better.

1000 generations or until the optimum is reached, whatever happened first. For several runs differing in the initial (random) population, we measured the *success rate* as the fraction of times the optimum is found. A commonly used performance measure in EDAs is the *critical population size* D^*; the minimum population size for which the success rate is 100%. Smaller D^* values have a double benefit on runtime: (i) fewer fitness evaluations for reaching the optima, and (ii) faster distribution estimation. We report D^* and the number of fitness evaluations required for that population size, denoted f^*. More robust algorithms are expected to require smaller D^* and f^* values. To measure D^* in Royal Road and OneMax, each version of MOA was run 10 times for each of the population sizes $D = \{50, 100, 200, 400, 800, 1600, 3200\}$. Then, for the measured D^*, we report the average and standard deviation of f^* on each of those runs. In all the experiments, the population is truncated with a selection size of 50% and an elitism of 50%; used for preventing diversity loss. In MOA, the parameter k was set to 3 and 1 in Royal Road and OneMax, respectively. Results are presented in Table 2 for the OneMax problem, and Table 3 for the Royal Road problem. For both algorithms MOA and MOA$'$, each table

reports the values of D^* as well as both the average and standard deviation of f^*, for increasing problem sizes $n \in \{15, 30, 60, 90, 120\}$ for the OneMax problem, and $n \in \{16, 32, 64, 92, 120\}$ for the Royal Road problem (the domain size should be a multiple of $\gamma = 4$). Lower values of D^* and f^* are better. In both tables, the results show that MOA$'$ always present equal or lower values of D^* than that of MOA, and also MOA$'$ always outperforms MOA in f^*. For Royal Road, the larger improvement is for $n = 92$ where MOA$'$ requires 75% fewer fitness evaluations f^* and D^* is halved. For OneMax, the larger improvement is for $n = 60$ where MOA$'$ requires 80% fewer fitness evaluations f^* and D^* is reduced to a quarter. An interpretation of these results is that IBMAP-HC estimates better the distribution at each iteration. To confirm this hypothesis we compared the structures learned by the two algorithms over our synthetic datasets. For a dataset with $n = 75$, $D = 100$, $\tau = 2$, the Hamming distances of MI and IBMAP-HC were 132, and 75, respectively. For $\tau = 4$ they were 233 and 143, respectively; and for $\tau = 8$, 395 and 388, respectively. These results show clearly that the quality of IBMAP-HC indeed outperforms that of MI. Finally, we highlight that the efficiency of IBMAP-HC allowed it to be run in large problems up to 120 genes in size, estimating the structure over many generations.

7 Conclusions and future work

This paper proposes IBMAP, a novel independence-based maximum-a-posteriori approach for learning the structure of Markov networks; and IBMAP-HC, an efficient instantiation of IBMAP. Our approach avoids the cascade errors of traditional independence-based algorithms that completely trust the outcome of statistical tests. For this, the central idea of IBMAP is to pose the structure learning task as a maximum-a-posteriori problem, by computing the posterior probability of each possible structure given data. Experiments comparing IBMAP-HC against state-of-the-art independence-based algorithms indicate that our method improves in most cases over the independence-based competitors with equivalent computational complexities. IBMAP-HC was also tested in a practical, challenging setting: Estimation of Distribution algorithms, resulting in faster convergence to the optimum than a state-of-the-art Markov network EDA algorithm, for the selected benchmark functions. Our experimental results and the conclusions of Appendix B confirm the effectiveness of our structure selection strategy. Therefore, we believe that it is worth guiding our future work in improving the IB-score as a measure of $\Pr(G \mid D)$, i.e., relaxing the independence assumption made in Equation (4), as well as exploring alternative closure sets. Also, it is clearly worthwhile considering testing our approach in more practical real-world testbeds, potentially comparing its performance against state-of-the-art score-based algorithms, such as [16, 32, 13, 40].

8 Acknowledgements

This work was funded by the grant PICT-241 of the National Agency of Scientific and Technological Promotion, FONCyT, Argentina; the grant PID-1205 of the National Technological University, Argentina; and the scholarship program for teachers of the National Technological University and the Ministry of Science, Technology and Productive Innovation; Argentina. Special thanks to Roberto Santana and Siddhartha Shakya for their help and support while implementing our experiments on EDAs.

A Correctness of the Markov blanket closure

This appendix presents a formal proof that the Markov blanket closure described in Definition 2 of Section 4.1 is in fact a closure, i.e., its independence assertions completely determine the structure used to generate it. Let us start by reproducing some necessary theoretical results extracted from [19,21,31]: the *pairwise Markov property*, the *Intersection property* of conditional independence, and the *Strong Union property* of conditional independence, all satisfied by any Markov network G of a positive graph-isomorph distribution P:

Definition 3 (Pairwise Markov property) Let G be a Markov network of some graph-isomorph distribution P, then

$$(X, Y) \notin E(G) \Leftrightarrow \langle X \perp\!\!\!\perp Y | V \setminus \{X, Y\} \rangle \text{ in } P. \tag{12}$$

Definition 4 (Intersection) The conditional independences among random variables of a positive distribution P satisfy the *Intersection* property (expressed in counter-positive form):

$$\langle X \not\perp\!\!\!\perp Y | \mathbf{Z} \rangle \wedge \langle X \perp\!\!\!\perp W | \mathbf{Z}, Y \rangle \Rightarrow \langle X \not\perp\!\!\!\perp Y | \mathbf{Z}, W \rangle \tag{13}$$

for all $(X \neq Y \neq W) \notin \mathbf{Z}$.

Definition 5 (Strong Union) The conditional independences among random variables of a graph-isomorph distribution P satisfy the following *Strong Union* property of conditional independence:

$$\langle X \perp\!\!\!\perp Y | \mathbf{Z} \rangle \Rightarrow \langle X \perp\!\!\!\perp Y | \mathbf{Z}, W \rangle \tag{14}$$

for all $(X \neq Y) \notin \mathbf{Z}$.

We present now two auxiliary lemmas that relate independences with edges in the graph:

Lemma 1

$$\langle X \perp\!\!\!\perp Y | \mathbf{B}_X \setminus \{Y\} \rangle \Rightarrow \ (X, Y) \notin E(G). \tag{15}$$

Proof. The proof proceeds by first applying the Strong union property to the l.h.s. to obtain $\langle X \perp\!\!\!\perp Y | \mathbf{V} \setminus \{X, Y\} \rangle$, and then applying the pairwise property to conclude the r.h.s. $(X, Y) \notin E(G)$. □

For the remaining of the proof we need to argue that something similar to the counter-positive of Lemma 1 holds:

Lemma 2

$$\langle X \not\perp\!\!\!\perp Y | \mathbf{B}_X \setminus \{Y\} \rangle \ \wedge \ \forall W \notin \mathbf{B}_X \langle X \perp\!\!\!\perp W | \mathbf{Z}, Y \rangle \Rightarrow \ (X, Y) \in E(G). \tag{16}$$

Proof. The proof proceeds by extending the conditioning set $\mathbf{B}_X \setminus \{Y\}$ of the l.h.s. to the whole domain $V \setminus \{X, Y\}$, to then apply the counter-positive of Eq. (12) and reach the r.h.s. $(X, Y) \in E(G)$. For that, we apply the intersection property of Eq. (13) iteratively, by taking at each iteration the pair containing one of the independences in the l.h.s., and, in the first iteration the dependence in the l.h.s., and the following iterations the dependence resulting from applying intersection. In all cases, we take $\mathbf{Z} = \mathbf{B}_X \setminus \{Y\}$. Let see this process in detail. In the first iteration we take from the l.h.s. the dependence and the independence for the first W, obtaining, by intersection, the dependence $\langle X \not\perp\!\!\!\perp Y | \mathbf{Z}, W \rangle$. We can now take the resulting dependence, with the independence for the following W, denoted for convenience W'. It seems that intersection can no longer be applied because the respective conditioning sets $\mathbf{Z} \cup \{W\}$ and $\mathbf{Z} \cup \{Y\}$ does not match. However, by graph-isomorphism of P, we have that the *Strong Union* property of conditional independence is satisfied in P, and therefore any independence given some conditioning set follows from the same independence given a subset of this conditioning set, in particular then, we have that $\langle X \perp\!\!\!\perp W' | \mathbf{Z}, W, Y \rangle$, and intersection can therefore be applied, resulting in $\langle X \not\perp\!\!\!\perp Y | \mathbf{Z}, W, W' \rangle$. Following this iteratively, we reach $\langle X \not\perp\!\!\!\perp Y | V \setminus \{X, Y\} \rangle$, where the resulting conditioning set $\mathbf{V} \setminus \{X, Y\}$ is the result of $\mathbf{Z} = \mathbf{B}_X \setminus \{Y\} \cup \mathbf{B}_X$, recalling $X \notin \mathbf{B}_X$. □

We can now prove our main theorem:

Theorem 1 *Let G be an undirected independence structure of a positive graph-isomorph distribution $P(\mathbf{V})$. The* Markov blanket closure *of G is a set of conditional independence assertions that are sufficient for completely determining the structure G.*

Proof. We prove the above theorem by proving that all the edges and no edges in G are determined by the assertions contained in $\mathcal{C}(G)$. We do it separately for absence and existence of edge between any two variables X and Y:

i) **For edge absence:** Let $(X, Y) \notin E(G)$. Then, by definition, the closure contains the two independence assertions: $\langle X \perp\!\!\!\perp Y | \mathbf{B}_X \setminus \{Y\} \rangle$ and $\langle Y \perp\!\!\!\perp X | \mathbf{B}_Y \setminus \{X\} \rangle$, which, by Eq. (15) of Lemma 1 both imply $(X, Y) \notin E(G)$.

ii) **For edge existence:**
Similarly, let $(X, Y) \in E(G)$. Then, by definition, the closure contains the dependence assertion: $\langle X \not\perp\!\!\!\perp Y | \mathbf{B}_X \setminus \{Y\} \rangle$. Also, for all W s.t. $(X, W) \notin E(G)$ (i.e., $W \notin \mathbf{B}_X$), the closure contains $\langle X \perp\!\!\!\perp W | \mathbf{B}_X \rangle$. Then, by Eq. (16) of Lemma 2 we have that $(X, Y) \in E(G)$. □

B IBMAP landscape analysis

In this appendix we report the results of an experiment that analyzes empirically the landscape of the IB-score function on synthetic datasets. The experiment consists in an analysis of the surface of the IB-score over the complete search space of possible structures. The aim is to assess how good is the hill-climbing search for maximizing the IB-score. Due to the exponential number of possible networks for each domain, in a first instance we explore how the complete landscape of IB-score looks like for datasets with a small domain size $n = 6$. For this experiment, we used synthetic datasets similar to those used in Section 5.1. The plots in Figure 6 show in the Y-axes the values of the IB-score for all the possible structures, and sort the structures in the X-axes, by its Hamming distance to the true underlying structure in the dataset (this is, from zero, to $\binom{n}{2}$). Note that the scores of the structures appear in log probabilities, because they was computed as shown in Equation (4). With this layout, the structures in the left (near to zero) are those with less structural errors, and are also those expected to have a higher value of the IB-score. Therefore, the structures in the right are expected to have lower values of the IB-score. Also, indicated with a diamond, the structures found by the algorithm IBMAP-HC are shown for each case. The plots are ordered in the columns for increasing values of the dataset $D \in \{10, 100, 1000\}$, and in the rows, the different values of $\tau \in \{1, 2, 4, 8\}$, increasing the complexity of the problem. From the analysis of such plots, it is observed how the landscape shapes to a decreasing curve as increasing the value D (see the tendency from left to right columns, and not the change in scale in the Y-axis). This is achieved because the precision of the statistical tests improves

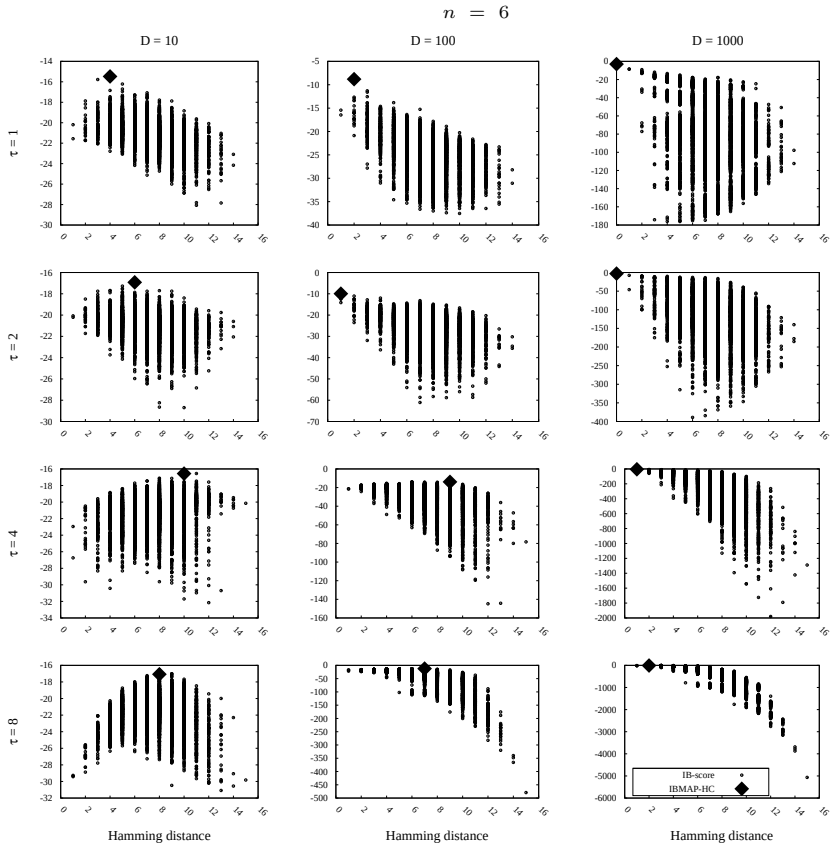

Fig. 6 Complete landscape of the IB-score for synthetic datasets with $n = 6$, for increasing dataset sizes $D = 10$ (first column), $D = 100$ (second column), and $n = 1000$ (third column), and $\tau \in \{1, 2, 4, 8\}$ in the rows. The X-axis sort the structures in the Hamming distance with the correct structure. The Y-axis shows the IB-score for all the structures in the landscape. The structure found by IBMAP-HC is indicated by a diamond.

with increasing D. In second place, the diamond that indicates the position in the landscape of the structure learned by the IBMAP-HC algorithm, achieves always the structure with highest score value. It can be also observed how the error of the structure learned by IBMAP-HC is closer to zero while increasing D. A second instance of this experiment was made for a domain size $n = 20$. In this instance, the landscape contains a total size of $2^{\binom{20}{2}}$. As it is impossible to show the IB-score for the complete landscape, we show only a subset obtained by generating randomly 5 structures deferring in m edges to the true structure, with m from 0 to $\binom{20}{2}$ in the X-axis. Such results are shown in Figure 7. From the analysis of such plots, the same conclusions are observed. To conclude this appendix, it is worth noting that our results confirm the effectiveness of our structure selection strategy in maximizing the IB-score over the complete landscape. For that reason, we conclude that it is worth guiding our future work only in the improvement of the IB-score as a measure of $\Pr(G \mid D)$.

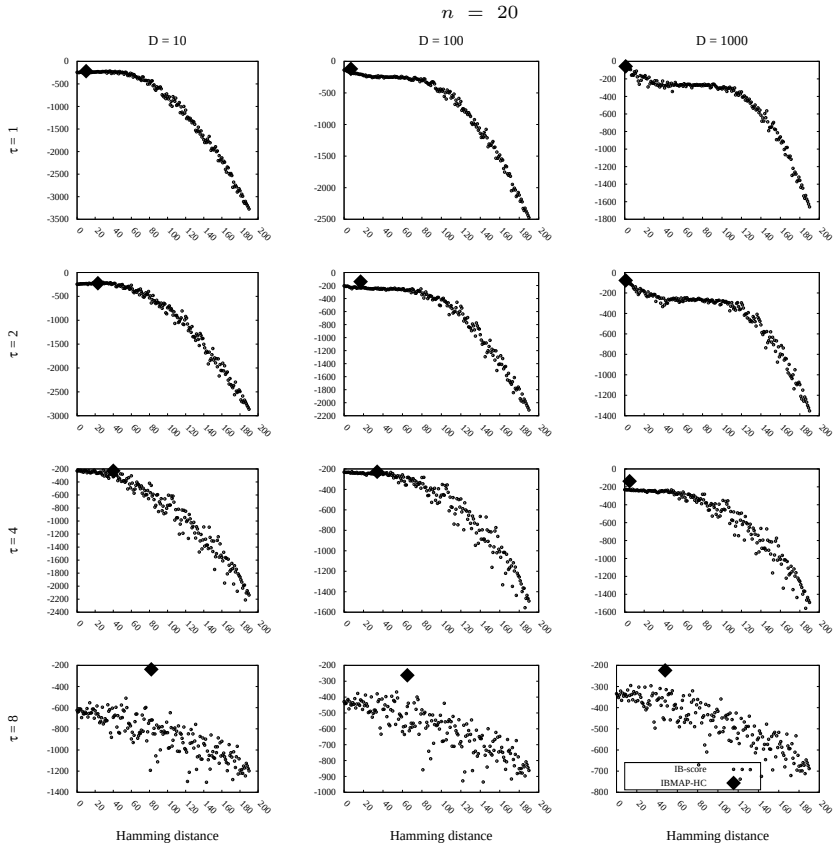

Fig. 7 A fraction of the landscape of the IB-score for synthetic datasets with $n = 20$, for increasing dataset sizes $D = 10$ (first column), $D = 100$ (second column), and $n = 1000$ (third column), and $\tau \in \{1, 2, 4, 8\}$ in the rows. The X-axis sort the structures in the Hamming distance with the correct structure. The Y-axis shows the IB-score for all the structures in the landscape. The structure found by IBMAP-HC is indicated by a diamond.

References

1. A. Asuncion, D.N.: UCI machine learning repository (2007)
2. Agresti, A.: Categorical Data Analysis, 2nd edn. Wiley (2002)
3. Alden, M.: MARLEDA: Effective Distribution Estimation Through Markov Random Fields. Ph.D. thesis, Dept of CS, University of Texas Austin (2007)
4. Aliferis, C., Statnikov, A., Tsamardinos, I., Mani, S., Koutsoukos, X.: Local Causal and Markov Blanket Induction for Causal Discovery and Feature Selection for Classification Part I: Algorithms and Empirical Evaluation. JMLR **11**, 171–234 (2010)
5. Aliferis, C., Statnikov, A., Tsamardinos, I., Mani, S., Koutsoukos, X.: Local Causal and Markov Blanket Induction for Causal Discovery and Feature Selection for Classification Part II: Analysis and Extensions. JMLR **11**, 235–284 (2010)
6. Aliferis, C., Tsamardinos, I., Statnikov, A.: HITON, a novel Markov blanket algorithm for optimal variable selection. AMIA Fall (2003)
7. Bromberg, F., Margaritis, D.: Improving the Reliability of Causal Discovery from Small Data Sets using Argumentation. JMLR **10**, 301–340 (2009)

8. Bromberg, F., Margaritis, D., Honavar, V.: Efficient markov network structure discovery using independence tests. In: In Proc SIAM Data Mining, p. 06 (2006)
9. Bromberg, F., Margaritis, D., V., H.: Efficient Markov Network Structure Discovery Using Independence Tests. JAIR **35**, 449–485 (2009)
10. Chickering, D.M.: Learning Bayesian networks is NP-Complete. In: D. Fisher, H. Lenz (eds.) Learning from Data: Artificial Intelligence and Statistics V, pp. 121–130. Springer-Verlag (1996)
11. Cover, T.M., Thomas, J.A.: Elements of information theory. Wiley-Interscience, New York, NY, USA (1991)
12. Cressie, N.: Statistics for spatial data. Terra Nova 4(5):613–617, DOI 10.1111/j.1365-3121.1992.tb00605.x
13. Davis, J., Domingos, P.: Bottom-Up Learning of Markov Network Structure. In: ICML, pp. 271–278 (2010)
14. Della Pietra, S., Della Pietra, V.J., Lafferty, J.D.: Inducing Features of Random Fields. IEEE Trans. PAMI. **19**(4), 380–393 (1997)
15. Friedman, N., Linial, M., Nachman, I., Pe'er, D.: Using Bayesian Networks to Analyze Expression Data. Journal of computational biology, pp. 601–620 (2000)
16. Ganapathi, V., Vickrey, D., Duchi, J., Koller, D.: Constrained Approximate Maximum Entropy Learning of Markov Random Fields. In: Uncertainty in Artificial Intelligence, pp. 196–203 (2008)
17. Hammersley, J. M., Clifford, P.: Markov fields on finite graphs and lattices (1968).
18. Hettich, S., Bay, S.D.: The UCI KDD archive (1999)
19. Koller, D., Friedman, N.: Probabilistic Graphical Models: Principles and Techniques. MIT Press (2009)
20. Larrañaga, P., Lozano, J.A.: Estimation of Distribution Algorithms. A New Tool for Evolutionary Computation. Kluwer Pubs (2002)
21. Lauritzen, S.L.: Graphical Models. Oxford University Press (1996)
22. Lee, S.I., Ganapathi, V., Koller, D.: Efficient structure learning of Markov networks using L1-regularization. In: NIPS (2006)
23. Li, S.: Markov random field modeling in image analysis. Springer, 2009.
24. Margaritis, D.: Distribution-Free Learning of Bayesian Network Structure in Continuous Domains. In: Proceedings of AAAI (2005)
25. Margaritis, D., Bromberg, F.: Efficient Markov Network Discovery Using Particle Filter. Comp. Intel. **25**(4), 367–394 (2009)
26. Margaritis, D., Thrun, S.: Bayesian network induction via local neighborhoods. In: Proceedings of NIPS06 (2000)
27. McCallum, A.: Efficiently inducing features of conditional random fields. In: Proceedings of Uncertainty in Artificial Intelligence (UAI) (2003)
28. Minka, T.: Divergence measures and message passing. Tech. rep., Microsoft Research (2005)
29. Mitchell, M.: An Introduction to Genetic Algorithms. MIT Press, Cambridge, MA, USA (1998)
30. Mühlenbein, H., Paaß, G.: From recombination of genes to the estimation of distributions I. binary parameters. In: H.M. Voigt, W. Ebeling, I. Rechenberg, H.P. Schwefel (eds.) Parallel Problem Solving from Nature PPSN IV, *Lecture Notes in Computer Science*, vol. 1141, pp. 178–187. Springer Berlin / Heidelberg (1996). 10.1007/3-540-61723-X_982
31. Pearl, J.: Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference. Morgan Kaufmann Publishers, Inc. (1988)
32. Ravikumar, P., Wainwright, M.J., Lafferty, J.D.: High-dimensional Ising model selection using L1-regularized logistic regression. Annals of Statistics **38**, 1287–1319 (2010). DOI 10.1214/09-AOS691
33. Santana, R.: Estimation of distribution algorithms with kikuchi approximations. Evol. Comput. **13**(1), 67–97 (2005). DOI 10.1162/1063656053583496. URL http://dx.doi.org/10.1162/1063656053583496
34. Schlüter, F.: A survey on independence-based markov networks learning. Artificial Intelligence Review pp. 1–25 (2012). URL http://dx.doi.org/10.1007/s10462-012-9346-y. 10.1007/s10462-012-9346-y

35. Shakya, S., McCall, J.: Optimization by estimation of distribution with deum framework based on markov random fields. International Journal of Automation and Computing **4**(3), 262–272 (2007). URL http://www.springerlink.com/index/10.1007/s11633-007-0262-6
36. Shakya, S., Santana, R., Lozano, J.A.: A markovianity based optimisation algorithm. Genetic Programming and Evolvable Machines **13**(2), 159–195 (2012)
37. Shekhar, S., Zhang, P., Huang, Y., Vatsavai, R. R.: Trends in spatial data mining. Data mining: Next generation challenges and future directions, 357-380 (2003).
38. Schmidt, M., Murphy, K., Fung, G., Rosales, R.: Structure learning in random fields for heart motion abnormality detection. In: Computer Vision and Pattern Recognition, 2008. CVPR 2008. IEEE Conference on, pp 1 –8, DOI 10.1109/CVPR.2008.4587367
39. Spirtes, P., Glymour, C., Scheines, R.: Causation, Prediction, and Search. Adaptive Computation and Machine Learning Series. MIT Press (2000)
40. Van Haaren, J., Davis, J.: Markov network structure learning: A randomized feature generation approach. In: Proceedings of the Twenty-Sixth AAAI Conference on Artificial Intelligence (2012) URL https://lirias.kuleuven.be/handle/123456789/345604
41. Van Haaren, J., Davis, J., Lappenschaar, M., Hommersom, A.: Exploring disease interactions using Markov networks. In: Proceedings of the AAAI-2013 (HIAI-2013). Bellevue, Washington, United States, 15 July, (2013)
42. Wainwright, M. J., Jordan, M. I: Graphical models, exponential families, and variational inference. Foundations and Trends in Machine Learning, 1(1-2), 1-305. (2008)
43. Welsh, D.J.A.: Complexity: knots, colourings and counting. Cambridge University Press, New York, NY, USA (1993)