Blankets Joint Posterior score for learning Markov network structures

Federico Schlüter[a], Yanela Strappa[a], Diego Milone[b], Facundo Bromberg[a]

[a] *DHARMa Lab, Dept of Information Systems. Facultad Regional Mendoza, Universidad Tecnológica Nacional, Mendoza, Argentina. Tel.: +54-261-5240066*
[b] *Research Institute for Signals, Systems and Computational Intelligence, sinc(i), FICH-UNL/CONICET Santa Fe, Argentina.*

Abstract

Markov networks are extensively used to model complex sequential, spatial, and relational interactions in a wide range of fields. By learning the Markov network independence structure of a domain, more accurate joint probability distributions can be obtained for inference tasks or, more directly, for interpreting the most significant relations among the variables. Recently, several researchers have investigated techniques for automatically learning the structure from data by obtaining the probabilistic maximum-a-posteriori structure given the available data. However, all the approximations proposed decompose the posterior of the whole structure into local sub-problems, by assuming that the posteriors of the Markov blankets of all the variables are mutually independent. In this work, we propose a scoring function for relaxing such assumption. The *Blankets Joint Posterior* score computes the joint posterior of structures as a joint distribution of the collection of its Markov blankets. Essentially, the whole posterior is obtained by computing the posterior of the blanket of each variable as a conditional distribution that takes into account information from other blankets in the network. We show in our experimental results that the proposed approximation can improve the sample complexity of state-of-the-art competitors when learning complex networks, where the independence assumption between blanket variables is clearly incorrect.

Key words: Markov network, structure learning, scoring function, blankets posterior, irregular structures

*Corresponding author
Email addresses: federico.schluter@frm.utn.edu.ar (Federico Schlüter)

1. Introduction

A Markov network (MN) is a popular probabilistic graphical model that efficiently encodes the joint probability distribution for a set of random variables of a specific domain [1, 2, 3]. MNs usually represent probability distributions by using two interdependent components: an independence structure, and a set of numerical parameters over the structure. The first is a qualitative component that represents structural information about a problem domain in the form of conditional independence relationships between variables. The numerical parameters are a quantitative component that represents the strength of the dependences in the structure. There is a large list of applications of MNs in a wide range of fields, such as computer vision and image analysis [4, 5, 6], computational biology [7], biomedicine [8, 9], and evolutionary computation [10, 11], among many others. For some of these applications, the model can be constructed manually by human experts, but in many other problems this can become unfeasible, mainly due to the dimensionality of the problem.

Learning the model from data consists of two interdependent problems: learning the structure; and given the structure, learning its parameters. This work focuses on the task of learning the structure, which is useful for a variety of tasks. The structures learned may be used to construct accurate models for inference tasks (such as the estimation of marginal and conditional probabilities) [12, 13, 14], and may also be interesting per se, since they can be used as interpretable models that show the most significant interactions of a domain [15, 16, 17, 18, 19]. The first scenario is known in practice as the density estimation goal of learning, and the second one is known as the knowledge discovery goal of learning [Chapter 16 [3]].

An interesting approach to MN structure learning is to use constraint-based (also known as independence-based) algorithms [20, 21, 22, 23]. Such algorithms proceed by performing statistical independence tests on data, and discard all structures inconsistent with the tests. This is an efficient approach, and it is

correct under the assumption that the distribution can be represented by a graph, and that the tests are reliable. However, the algorithms that follow this approach are quite sensitive to errors in the tests, which may be unreliable for large conditioning sets [20, 3]. A second approach to MN structure learning is to use score-based algorithms [24, 25, 15, 26]. Such algorithms formulate the problem as an optimization, combining a strategy for searching through the space of possible structures with a scoring function measuring the fitness of each structure to the data. The structure learned is the one that achieves the highest score in the search.

It is important to mention that both constraint-based and score-based approaches have been originally motivated by distinct learning goals. According to the existing literature [3], constraint-based methods are generally designed for the knowledge-discovery goal of learning [22, 21], and their quality is often measured in terms of the correctness of the structure learned (structural errors). In contrast, most score-based approaches have been designed for the density estimation goal of learning [12, 13, 14], and they are in general evaluated in terms of inference accuracy. For this reason, score-based algorithms often work by considering the whole MN at once during the search, interleaving the parameter learning step. This makes them more accurate for inference tasks. However, since learning the parameters is known to be NP-hard for MNs [27], it has a negative effect on their scalability.

Recently, there has been a surge of interest towards efficient methods based on a strategy that follows a score-based approach, but with the knowledge discovery goal in mind. Basically, an undirected graph structure is learned by obtaining the probabilistic maximum-a-posteriori structure given the available data [28, 19, 29]. This hybrid strategy achieves scalability, as well as reliable performance. Such contributions consist in the design of efficient scoring functions for MN structures, expressing the problem formally as follows: given a complete training data set D, find an undirected graph $G^\star$ such that

$$G^\star = \arg\max_{G \in \mathcal{G}} \Pr(G|D), \tag{1}$$

where $\Pr(G|D)$ is the posterior probability of a structure given D, and $\mathcal{G}$ is the familiy of all the possible undirected graphs for the domain size. This class of algorithms has been shown to outperform constraint-based algorithms in the quality of the learned structures, with competitive computational complexities. The method proposed in this paper follows this approach.

Since there are no feasible exact methods for computing the posterior of MN structures, different approximations have been proposed. An important assumption commonly made by the current state-of-the-art methods is to suppose that the posterior of the structure is decomposable [30, 31, 3, 28, 19, 29]. It means that the whole posterior can be computed as a product of the posteriors of the Markov blankets that compose the structure, which are smaller posteriors that can be computed independently. In fact, this is a good approximation that improves the efficiency of search. The research line of this work aims at designing a better approximation of the posterior, by relaxing such independence assumption. For this, the contribution of this work is the *Blankets Joint Posterior* (BJP), a scoring function that estimates $\Pr(G|D)$ as the joint posterior probability of the Markov blankets of G. This is achieved by formulating $\Pr(G|D)$ in a novel way that relaxes the independence assumption between the blankets. Essentially, the whole posterior is obtained by computing the posterior of the blanket of each variable as a conditional distribution that takes into account information from other blankets in the network. In our experiments we show that the proposed approximation can improve the sample complexity of state-of-the-art scores when learning networks with complex topologies, that commonly appear in real-world problems.

After providing some preliminaries, notations and definitions in Section 2, we introduce the BJP scoring function in Section 3. Section 4 presents the experimental results for several study cases. Finally, Section 5 summarizes this work, and poses several possible directions of future work.

2. Background

We begin by introducing the notation used for MNs. Then we provide some additional background about these models and the problem of learning their independence structure, and also discuss the state-of-the-art of MN structure learning.

2.1. Markov networks

Let V be a finite set of indexes, lowercase subscripts for denoting particular indexes, e.g., $i, j \in V$, and uppercase subscripts for subsets of indexes, e.g., $W \subseteq V$. Let X_V be the set of random variables of a domain, denoting single variables as single indexes in V, e.g., $X_i, X_j \in X_V$ where $i, j \in V$. For a MN representing a probability distribution $P(X_V)$, its two components are denoted as follows: G, and θ. G is the structure, an undirected graph $G = (V, E)$ where the nodes $V = \{0, ..., n-1\}$ are the indices of each random variable X_i of the domain, and $E \subseteq \{V \times V\}$ is the edge set of the graph. A node i is a neighbor of j when the pair $(i, j) \in E$. The edges encode direct probabilistic influence between the variables. Similarly, the absence of an edge manifests that the dependence could be mediated by some other subset of variables, corresponding to conditional independences between these variables.

A variable X_i is conditionally independent of another non-adjacent variable X_j given a set of variables X_Z if $\Pr(X_i \mid X_j, X_Z) = \Pr(X_i \mid X_Z)$. This is denoted by $\langle X_i \perp X_j | X_Z \rangle$ (or $\langle X_i \not\perp X_j | X_Z \rangle$ for the dependence assertion). As proven by [32], the independences encoded by G allow the decomposition of the joint distribution into simpler lower-dimensional functions called factors, or potential functions. The distribution can be factorized as the product of the potential functions $\phi_c(V_c)$ over each clique V_c (i.e., each completely connected sub-graph) of G, that is

$$P(V) = \frac{1}{Z} \prod_{c \in \text{cliques}(G)} \phi_c(V_c), \tag{2}$$

where Z is a constant that normalizes the product of potentials. Such potential functions are parameterized by the set of numerical parameters θ.

For each variable X_i of a MN, its Markov blanket is composed by the set of all its neighbor nodes in the graph. We denote the blanket of a variable X_i as B^{X_i}. An important concept that is satisfied by MNs is the Local Markov property, formally described as:

> **Local Markov property**. A variable is conditionally independent of all its non-neighbor variables given its MB. That is

$$\langle X_i \perp \{X_V \setminus (B^{X_i} \cup X_i)\} \mid B^{X_i} \rangle. \tag{3}$$

By using this property, the conditional independences of $P(X_V)$ can be read from the structure G. This is done by considering the concept of separability. Each pair of non-adjacent variables (X_i, X_j) is said to be separated by a set of variables $X_Z \subseteq X_V \setminus \{X_i, X_j\}$ when every path between X_i and X_j in G contains some node in X_Z [1].

In machine learning, statistical independence tests are a well-known tool to decide whether a conditional independence is supported by the data. Examples of independence tests used in practice are Mutual Information [33], Pearson's χ^2 and G^2 [34], the Bayesian statistical test of independence [35], and the Partial Correlation test for continuous Gaussian data [20]. Such tests require the construction of a contingency table of counts for each complete configuration of the variables involved; as a result, they would have an exponential cost in the number of variables [36]. For this reason, the use of the local Markov property has a positive effect for learning independence structures, allowing the use of smaller tests. Accordingly, the BJP score introduced in this work takes advantage of this property by computing a set of conditional probabilities that are more reliable and less expensive.

2.2. Scoring metrics for MN structure learning

The MN structure is learned from a training dataset $D = \{D_1, ..., D_d\}$, assumed to be a representative sample of the underlying distribution $P(X_V)$. Commonly, D has a tabular format, with a column for each variable of the domain X_V, and one row per data point. This work assumes that each variable is

discrete, with a finite number of possible values, and that no data point in D has missing values. As mentioned in the introduction, this work focuses on methods for computing $\Pr(G|D)$. For this reason, in this subsection we review two recently proposed scoring functions that approximate it: the Marginal Pseudo-Likelihood (MPL) score [19], and the Independence-based score (IB-score) [28].

2.2.1. Marginal pseudo-likelihood score

Marginal Pseudo Likelihood (MPL) is a scoring function for MN structure learning recently proposed [19], based on the computation of the pseudo-likelihood score for Markov networks. In [19] was shown that MPL is a small sample analytical version of the pseudo-Bayesian information criterion (PIC) score, a previous work introduced by [29] as a modification of the BIC score for Markov networks. Both MPL and PIC scores approximate the posterior of structures by considering $P(G \mid D) \propto P(D \mid G) \times P(G)$. Since the data likelihood of the graph $P(D \mid G)$ is in general extremely hard to evaluate, they utilize the well-known approximation called pseudo-likelihood [37]. The contribution of MPL has been designed in order to be a tractable alternative, that can be evaluated in closed form for chordal and non-chordal Markov networks.

The MPL score approximates the posterior of an independence structure by using standard Bayesian calculations, with the closed-form expression

$$P(D \mid G) = \prod_{j=1}^{n} \prod_{l=1}^{q_j} \frac{\Gamma(\alpha_{jl})}{\Gamma(\alpha_{jl} + c_{jl})} \prod_{i=1}^{r_j} \frac{\Gamma(\alpha_{ijl} + c_{ijl})}{\Gamma(\alpha_{ijl})}, \tag{4}$$

where n is the number of variables in the domain, q_j is the number of configurations of B^{X_j}, r_j is the number of configurations of variable X_j, c_{ijl} is the frequency in D of the ijl configuration (corresponding to the i-th configuration of X_j and l-th configuration of its blanket B^{X_j}), and α_{ijl} are the hyperparameters, computed according to $\alpha_{ijl} = \frac{N}{r_j \cdot q_j}$, with N being the equivalent sample size, used to adjust the prior.

The above formula can be factorized into variable-wise marginal conditional likelihoods, that is, a sum of variable-wise scores. This decomposition is exploited to speed-up the search procedure for finding the MPL-optimal structure.

For this an efficient algorithm is proposed by its authors in order to ensure applicability in high-dimensional settings. The optimization technique proposed exploits the structural decomposition of the score by breaking down the problem into two phases. In a first phase, the problem is decomposed in n independent Markov blanket discovery problems, locally optimizing the MPL for each node. For this, it uses an approximate deterministic hill-climbing procedure similar to the well-known IAMB algorithm [38]. In a second optimization phase, the learned Markov blankets are combined into a coherent structure which is MPL-optimal. This phase uses a greedy hill-climbing algorithm, searching for the structure with maximum MPL score, but only restricting the search space to the conflicting edges (i.e., edges learned for only one of its two variables). A detailed description of this algorithm can be seen at [19, Section 4.2 on p. 10].

2.2.2. The Independence-based score

The independence-based score (IB-score) [28] is also based on the computation of the posterior, but using the statistics of a set of conditional independence tests. This score computes the posterior $\Pr(G \mid D)$ by combining the outcomes of a set of conditional independence assertions that completely determine G. Such set is called the *closure* of the structure, denoted $\mathcal{C}(G)$. Thus, when using IB-score, the problem of structure learning is posed as the maximization of the posterior of the closure for each structure:

$$G^\star = \arg\max_{G \in \mathcal{G}} \Pr(\mathcal{C}(G) \mid D). \tag{5}$$

Applying the chain rule over the posterior of the closure,

$$\Pr(\mathcal{C}(G) \mid D) = \prod_{c_i \in \mathcal{C}(G)} \Pr(c_i | c_1, \ldots, c_{i-1}, D), \tag{6}$$

the IB-score approximates such probability by assuming that all the independence assertions c_i in the closure $\mathcal{C}(G)$ are mutually independent. The resulting scoring funtion is computed as

$$\text{IB-score}(G) = \sum_{c_i \in \mathcal{C}(G)} \log \Pr(c_i \mid D), \tag{7}$$

where each term $\log \Pr(c_i \mid D)$ is computed by using the Bayesian statistical test of conditional independence [35, 39]. Appendix C presents a summary of the formulas used by this statistical test, which is also used in our BJP scoring function, proposed in the next section.

The $\mathcal{C}(G)$ set proposed by the authors of IB-score is the *Markov blanket closure* [28, Definition 2], formally proven to correctly and completely determining a MN structure. This set is obtained by determining the blanket of each variable $X_i \in X_V$ with the following set of conditional independence and dependence assertions:

$$\left\{\langle X_i \perp X_j | B^{X_i}\rangle \ : \ X_j \notin B^{X_i}\right\} \ \bigcup \ \left\{\langle X_i \not\perp X_j | B^{X_i} \backslash \{X_j\}\rangle \ : \ X_j \in B^{X_i}\right\}. \quad (8)$$

That is, for each neighbor of X_i ($X_j \in B^i$) a conditional dependence assertion between both variables conditioning on $B^i \setminus \{X_j\}$ is added to $\mathcal{C}(G)$; and for each non-neighbor of X_i ($X_j \notin B^i$), a conditional independence assertion between both variables conditioned on B^i is added to $\mathcal{C}(G)$;.

Together with the IB-score, an efficient algorithm called IBMAP-HC was presented to learn the structure by using a heuristic local search over the space of possible structures. IBMAP-HC has been proven to significantly outperform its independence-based competitors in terms of quality. A detailed description of this algorithm can be seen at [28, on p. 6]. The optimization made by IBMAP-HC is a heuristic hill-climbing procedure. The search is initialized by computing the score for an empty structure (with no edges), and n nodes. The hill-climbing search starts with a loop that iterates by selecting the next candidate structure at each iteration. A naïve implementation of hill-climbing would select the neighbor structure with maximum score, computing the score for the $\binom{n}{2}$ neighbors that differ in one edge. Such expensive computation is avoided by selecting the next candidate with a heuristic that estimates the optimal neighbor by fliping the most promising edge, that is, the edge with lowest local contribution to the score. For this, the heuristic simply decomposes the posterior of the structure in $\binom{n}{2}$ pairwise scores, since the number of neighbors differing by one edge is the same than the number of different pairs of variables.

Then, the heuristic simply flips the edge corresponding to the pair with the lowest pairwise score. Once the next candidate is selected, its score is computed to be compared to the best scoring structure found so far. The algorithm stops when the neighbor proposed does not improve the current score.

3. Blankets Joint Posterior scoring function

We introduce now our main contribution, the Blankets Joint Posterior (BJP) scoring function. Consider some graph G representing the independence structure of a positive MN. It is a well-known fact that, by exploiting the graphical properties of such models, the independence structure can be decomposed as the unique collection of the blankets of the variables [3, Theorem 4.6 on p. 121]. Thus, the computation of the posterior probability of G given a dataset D is equivalent to the joint posterior of the collection of blankets of G, that is,

$$\Pr(G \mid D) = \Pr(B^{X_0}, B^{X_1}, \ldots, B^{X_{n-1}} \mid D). \tag{9}$$

In contrast with previous works, where the blanket posteriors are simply assumed to be independent [19, 28, 29], we applied the chain rule to (9), obtaining

$$\Pr(B^{X_0}, \ldots, B^{X_{n-1}} \mid D) = \prod_{i=0}^{n-1} \Pr\left(B^{X_i} \middle| \left\{ B^{X_j} \right\}_{j=0}^{i-1}, D \right). \tag{10}$$

In this way, the posterior probability of each blanket can be described in terms of conditional probabilities, using the training dataset D as evidence, together with the blanket of the other variables. Thus, the joint posterior of all the blankets can be computed taking advantage of how the blankets are mutually related, instead of assuming them to be independent.

The computation of $\Pr(B^{X_0}, \ldots, B^{X_{n-1}} \mid D)$ has to be done progressively, first calculating the posterior of the blanket of a variable directly from data, and then, the knowledge obtained so far can be used as evidence to compute the posterior of the blankets of other variables. However, this decomposition is not unique, since each possible ordering for the variables is associated to a

particular decomposition. The basic idea underlying the computation of BJP is to sort the blankets by their size in ascending order, where by size we mean the number of configurations of the Markov blanket. This ordering is optimal, because it avoids the computation of expensive and unreliable probabilities, thus improving data efficiency. This is due to the fact that as the size of the blanket increases, greater amounts of data are required for accurately estimating its posterior probability. By using the proposed ordering, the posterior for variables with fewer blankets are computed first, and this information is used as evidence when computing the posterior for variables with bigger blankets. As a result, the information obtained from the more reliable blanket posteriors is used for computing less reliable blankets posteriors. It is important to note that, in theory, the correctness of BJP does not depend on which ordering is used. However, this is important for practical implementation because it can affect the data efficiency of the score. In Section 3.1 we show a complete example of the BJP computation which illustrates the importance of the ordering used. Additionally, Appendix B extends the example with an empirical test for the performance of BJP when different arbitrary orderings are used.

We now proceed to find a closed-form expressin for computing the BJP score. Given an undirected graph G, denote ψ the ordering vector which contains the variables sorted by their size in ascending order. Therefore, we reformulate (10) as

$$BJP(G) = \prod_{i=0}^{n-1} \Pr\left(B^{\psi_i} \middle| \left\{B^{\psi_j}\right\}_{j=0}^{i-1}, D\right). \tag{11}$$

We now proceed to express the posterior of a blanket in terms of probabilities of conditional independence and dependence assertions. The computation of $\Pr(B^{\psi_i}|\{B^{\psi_j}\}_{j=0}^{i-1}, D)$ can be derived from the posterior of the independences and dependences represented by each blanket:

$$\begin{aligned}\Pr\left(B^{\psi_i} \middle| \left\{B^{\psi_j}\right\}_{j=0}^{i-1}, D\right) = &\prod_{\psi_k \notin B^{\psi_i}} \Pr\left(\langle \psi_i \perp \psi_k | B^{\psi_i} \rangle \middle| \left\{B^{\psi_j}\right\}_{j=0}^{i-1}, D\right) \times \\ &\prod_{\psi_k \in B^{\psi_i}} \Pr\left(\langle \psi_i \not\perp \psi_k | B^{\psi_i} \setminus \{\psi_k\} \rangle \middle| \left\{B^{\psi_j}\right\}_{j=0}^{i-1}, D\right).\end{aligned} \tag{12}$$

In this way, the whole score is the product of the posterior probability of each blanket, computed in terms of posterior probabilities conditioned in other blankets. The particular way of determining the posterior of each blanket of (12) is inspired by the Markov blanket closure (see Section 2.2.2).

The two factors in (12) can be interpreted as follows:

- The first product computes the probability of independence between ψ_i and its non-adjacent variables, conditioned on its blanket, given the previously computed blankets and the dataset D. It is computed as

$$\Pr\left(\langle \psi_i \perp \psi_k | B^{\psi_i} \rangle \middle| \left\{B^{\psi_j}\right\}_{j=0}^{i-1}, D\right) = \begin{cases} \Pr(\langle \psi_i \perp \psi_k | B^{\psi_i} \rangle \mid D) & \\ & \text{if } i < k, \\ 1 & \text{if } i > k. \end{cases} \tag{13}$$

  Here, $i < k$ indexes over the variables for which the blanket posterior probability is not already computed. For the remaining variables the posterior of independence will be simply inferred as 1.

- The second product in (12) computes the posterior probability of dependence between ψ_i and its adjacent variables, conditioned on its remaining neighbors, given the blankets computed previously and the dataset D. It is computed as

$$\Pr\left(\langle \psi_i \not\perp \psi_k | B^{\psi_i} \setminus \{\psi_k\} \rangle \middle| \left\{B^{\psi_j}\right\}_{j=0}^{i-1}, D\right) = \begin{cases} \Pr(\langle \psi_i \not\perp \psi_k | B^{\psi_i} \setminus \{\psi_k\} \rangle \mid D) & \\ & \text{if } i < k, \\ 1 & \text{if } i > k. \end{cases} \tag{14}$$

  Here, again $i < k$ indexes over the variables for which the blanket posterior is not already computed. For the remaining variables the posterior of dependence will be inferred as 1.

The only approximation in BJP is made in (12), by assuming that all the independence and dependence assertions that determine the blanket of a variable

ψ_i are mutually independent. This is a common assumption, made implicitly by all the constraint-based MN structure learning algorithms [23], and also by the IB-score, MPL, and the PIC scoring functions. For the computation of the posterior probabilities of independence $\Pr(\langle \psi_i \perp \psi_k | B^{\psi_i} \rangle \mid D)$ and dependence $\Pr(\langle \psi_i \not\perp \psi_k | B^{\psi_i} \setminus \{\psi_k\} \rangle \mid D)$ used in (13) and (14), respectively, BJP uses the Bayesian test of [39, 35, 40], in the same way as the IB-score explained in the previous section. Precisely, this statistical test computes the posterior of independence and dependence assertions, and has been proven to be statistically consistent in the limit of infinite data. A summary of the formulas used by the Bayesian test is shown in Appendix C.

An important property of a scoring function is the correctness. By *correctness* we mean that, under the assumption that the generating distribution is faithful to a Markov network structure, the probabilities computed in (12), (13) and (14) are necessary and sufficient to calculate the posterior probability of a MN structure. The following theorem establishes that the BJP scoring function is indeed correct:

Theorem 1. *Let G be an undirected independence structure of a positive graph-isomorph distribution $P(X_V)$. The BJP scoring function of G is "correct" in the sense that the posterior probability that computes is equivalent to the posterior probability of a MN structure.*

Proof of Theorem 1. The formal proof of this theorem is presented in Appendix A.

Now we briefly discuss the computational complexity of the BJP scoring function. For a fixed MN structure, the computational cost of BJP is directly determined by the number of statistical tests that must be perfomed on the data. As stated in (11), BJP computes the posterior probability of the blanket for the n variables of the domain. For each, it is required to perform $n - 1$ statistical tests on data, by using (12). Then, one half of the tests are inferred when computing the posterior of independences and dependences of (13) and

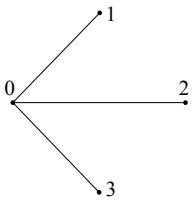


Figure 1: Example of an undirected graph with 4 nodes and hub topology

(14). Thus, only $\frac{n(n-1)}{2}$ tests are required for computing the BJP score of a structure.

3.1. Example of BJP score computation

For the sake of clarity, this section shows the complete computation of the BJP score for an illustrative example. Consider an example probability distribution $\Pr(X_V)$ with four binary variables $X_V = \{X_0, X_1, X_2, X_3\}$, represented by a MN whose independence structure G is given by the graph of Figure 1. Given a dataset D, the BJP score can be computed by following steps:

a) Build a vector ψ, with the nodes sorted by their size in ascending order. Since all the variables have the same domain size, the following vector is optimal: $\psi = (X_1, X_2, X_3, X_0)$, according to their degree as shown in the graph.

b) By following (11), the computation of $BJP(G)$ is given by:

$$\begin{aligned} BJP(G) = \quad & \Pr\left(B^{X_1} \middle| D\right) \\ & \times \Pr\left(B^{X_2} \middle| B^{X_1}, D\right) \\ & \times \Pr\left(B^{X_3} \middle| B^{X_1}, B^{X_2}, D\right) \\ & \times \Pr\left(B^{X_0} \middle| B^{X_1}, B^{X_2}, B^{X_3}, D\right). \end{aligned}$$

c) Compute each term of the above expression by following (12), resulting

in:

$$\begin{aligned}
\Pr\left(B^{X_1}\middle|D\right) = &\ \Pr\left(\langle X_1 \perp X_2|X_0\rangle\middle|D\right)\\
&\times \Pr\left(\langle X_1 \perp X_3|X_0\rangle\middle|D\right)\\
&\times \Pr\left(\langle X_1 \not\perp X_0|\emptyset\rangle\middle|D\right).\\
\Pr\left(B^{X_2}\middle|B^{X_1},D\right) = &\ \Pr\left(\langle X_2 \perp X_1|X_0\rangle\middle|B^{X_1},D\right)\\
&\times \Pr\left(\langle X_2 \perp X_3|X_0\rangle\middle|B^{X_1},D\right)\\
&\times \Pr\left(\langle X_2 \not\perp X_0|\emptyset\rangle\middle|B^{X_1},D\right).\\
\Pr\left(B^{X_3}\middle|B^{X_1},B^{X_2},D\right) = &\ \Pr\left(\langle X_3 \perp X_1|X_0\rangle\middle|B^{X_1},B^{X_2},D\right)\\
&\times \Pr\left(\langle X_3 \perp X_2|X_0\rangle\middle|B^{X_1},B^{X_2},D\right)\\
&\times \Pr\left(\langle X_3 \not\perp X_0|\emptyset\rangle\middle|B^{X_1},B^{X_2},D\right).\\
\Pr\left(B^{X_0}\middle|B^{X_1},B^{X_2},B^{X_3},D\right) = &\ \Pr\left(\langle X_0 \not\perp X_1|X_2,X_3\rangle\middle|B^{X_1},B^{X_2},B^{X_3},D\right)\\
&\times \Pr\left(\langle X_0 \not\perp X_2|X_1,X_3\rangle\middle|B^{X_1},B^{X_2},B^{X_3},D\right)\\
&\times \Pr\left(\langle X_0 \not\perp X_3|X_1,X_2\rangle\middle|B^{X_1},B^{X_2},B^{X_3},D\right).
\end{aligned}$$

d) By replacing Equations (13) and (14) in the factors of the above expression, one half of the tests can be inferred, and only the following probabilities must be computed from data by using the Bayesian statistical test:

$$\begin{aligned}
&\Pr\left(B^{X_1}\middle|D\right) = \Pr\left(\langle X_1 \perp X_2|X_0\rangle\middle|D\right) \times \Pr\left(\langle X_1 \perp X_3|X_0\rangle\middle|D\right) \times \Pr\left(\langle X_1 \not\perp X_0|\emptyset\rangle\middle|D\right).\\
&\Pr\left(B^{X_2}\middle|B^{X_1},D\right) = 1 \times \Pr\left(\langle X_2 \perp X_3|X_0\rangle\middle|D\right) \times \Pr\left(\langle X_2 \not\perp X_0|\emptyset\rangle\middle|D\right).\\
&\Pr\left(B^{X_3}\middle|B^{X_1},B^{X_2},D\right) = 1 \times 1 \times \Pr\left(\langle X_3 \not\perp X_0|\emptyset\rangle\middle|D\right).\\
&\Pr\left(B^{X_0}\middle|B^{X_1},B^{X_2},B^{X_3},D\right) = 1 \times 1 \times 1.
\end{aligned}$$

The inferred tests are the 1s at each equation. This example allows us to illustrate the intuition behind BJP, since the sample complexity of the blanket posterior for variables X_1, X_2, and X_3 is lower than that of X_0. Moreover, in this example it is clear that the posterior distribution of B^{X_0} is not independent of the posterior distributions of B^{X_1}, B^{X_2} and B^{X_3}. Clearly, the posterior of B^{X_0} is harder to evaluate than the posterior of the remaining variables, and then, computing $\Pr(B^{X_0}|B^{X_1},B^{X_2},B^{X_3},D)$ could be more informative that only computing $\Pr(B^{X_0}|D)$ independently of the rest of blankets. Appendix B

shows two experiments with the graph of Figure 1, for checking empirically how the ordering affects the performance of BJP.

3.2. BJP versus existent methods

As mentioned before, MPL and IB-score are two recently proposed methods for computing the probabilistic maximum-a-posteriori structure given data. It is important to note that the three scores have been designed from different points of view. Due to the difficulty of evaluating likelihood-based scores for non-chordal graphs, MPL proposed a metric that does not impose the restriction of chordality. Instead, IB-score has been designed for tackling the problems of constraint-based algorithms: these algorithms procedd by performing statistical independence tests on data, trusting the outcome of each test completely. In practice some tests may be incorrect, resulting in potential cascading errors. IB-score tackles this problem through a probabilistic maximum-a-posteriori approach that combines the outcomes of statistical independence tests. The BJP score proposed in this work is strongly influenced by the IB-score viewpoint. However, the research of this work aims at designing a better approximation of the posterior, by relaxing the independence assumption between blanket posteriors (made by both IB-score and MPL).

Another important difference is the decomposability properties of each score. On the one hand, the MPL score has an analytic expression that factorizes into variable-wise marginal conditional likelihoods, as can be seen in (4). This allows MPL to be optimized as proposed by its authors, by decomposing the problem in n independent Markov blanket discovery problems, locally optimizing the MPL for each node. By optimizing the score in this way, they are exploiting the independence assumption between blankets for speeding-up the search procedure. Instead, BJP tackles the negative effects that such assumption carries on in the quality of structures learned. On the other hand, the IB-score has an analytic expression that depends on the choice of the closure set used, as can be seen in (7) and (8). The efficient optimization proposed for IB-score is called the IBMAP-HC algorithm, and it does not decompose the score. This algorithm

optimizes the score of the whole structure, without assuming the blankets to be independent.

The independence assumption affects the data efficiency of the scoring functions. The main drawback of the MPL is that, as a result of the independence assumption, it over-specifies the node-wise conditional distributions. This has a negative effect on the data efficiency, especially for networks with hub nodes[1]. Regarding IB-score, its main drawback is related to the use of the Markov blanket closure, which allows to correctly compute $\Pr(G|D)$. Again, by assuming the Markov blankets to be mutually independent, the IB-score computes redundant probabilities. BJP improves the data efficiency problems caused by the redundancies in the IB-score by sorting the blankets of the graph by their size in ascending order and then computing the conditional distributions that involve other blankets as evidence. Precisely, in our approach only one probability is computed for each pair, and the redundant ones are inferred. For this reason, it is expected that for data scarcity conditions the BJP scoring function improves over both MPL and IB-score.

3.3. Optimization

The goal of this work is to propose a score for approximating the posterior of structures by relaxing the independence assumption between blankets. For this reason, we want to evaluate and compare the scoring functions independently of the search process used. For learning the structure with a score, the naïve optimization consists in maximizing over all the possible undirected graphs for some specific problem domain, as shown in (1). Since the discrete optimization space of the possible graphs $\mathcal{G}$ grows rapidly with the number of variables n, the search is clearly intractable even for small domain sizes. Thus, in Section 4 we show a comparison of the performance of BJP against MPL and IB-score when

[1]This is because the conditional distributions are specified in terms of complete Markov blankets even if only a subset of a Markov blanket is sufficient for shielding a node from a particular part of the network.

using brute force maximization (i.e., exhaustive search). It allows to study the convergence of the scoring functions to the exact solution for low-dimensional problems, without any particular search mechanism. Additionally, we show several experiments with higher dimensional domains. In these experiments we used the IBMAP-HC algorithm explained in Section 2.2.2 for maximizing the BJP score, as an efficient approximate solution.

4. Experimental evaluation

This section presents several experiments in order to determine the merits of BJP in practical terms. Two sets of experiments are presented, one from low-dimensional problems, and another for high-dimensional problems. For the low-dimensional setting, we used brute force (i.e., exhaustive search) to study the convergence of the scoring functions to the exact solution, what we later in Section 4.1 call the consistency experiments. We compare BJP against the two recently proposed scoring functions that approximate the posterior of MN structures: MPL and IB-score. The goal is to prove experimentally that the sample complexity for successfully learning the exact structure of BJP can be better than for the competitors, independently of the optimization mechanism used. Exhaustive search is limited to low-dimensional settings as the search space grows exponentially with the square of the number of variables, so for the high-dimensional setting, we used the IBMAP-HC algorithm for comparing the performance of BJP against several state-of-the-art competitors. These experiments were performed in order to prove that BJP can identify structures with fewer structural errors than the competitor state-of-the-art algorithms for realistic scenarios. The software to carry out the experiments has been developed in Java, and it is publicly available[2].

For the experiments we selected a set of networks where the topologies exhibit irregularities, which is a common property in many real-world networks

[2]http://dharma.frm.utn.edu.ar/papers/bjp

[41]. According to [42], the irregularity of an undirected graph can be computed by summing the imbalance of its edges:

$$irr(G) = \sum_{(i,j)\in E(G)} |d_G(i) - d_G(j)|, \tag{15}$$

where $d_G(i)$ is the degree of the node i in that graph. Clearly $irr(G) = 0$ if and only if G is regular. For non-regular graphs $irr(G)$ is a measure of the lack of regularity. Since in our experiments we test only domains with discrete binary variables, we used the irregularity of the underlying structure as an external control variable that determines the importance of the independence assumption between blankets for decomposable scores. Thus, as larger the irregularities the larger is the difference between the sizes of the infered blankets and their matching ones, resulting in larger expected improvements against competitors.

4.1. Consistency experiments

A MN scoring function is consistent when the structure which maximizes the score over all the possible structures is the correct one, in the limit of infinite data. However, in practice the data is often too scarce to satisfy this condition, and the sample size needed to reach the correct structure varies across different scoring functions. This is referred to as the *sample complexity* of the score. The experiments here presented were carried out in order to measure the sample complexity of the three different scoring functions: MPL, IB-score and BJP. This is achieved by measuring their ability to return, by brute force, the exact independence structure of the MN which generated the data.

To make this comparative study, we selected the six different target structures shown in Figure 2. These graphs represent different cases of irregularity, according to (15). The first target structure is regular (irr = 0), the second has a little irregularity, the third and fourth structures are irregular structures with a hub topology, and the fifth and sixth target structures have maximum irregularity for $n = 6$. As mentioned before, the irregularity is used here as a parameter for determining the importance of the independence assumption

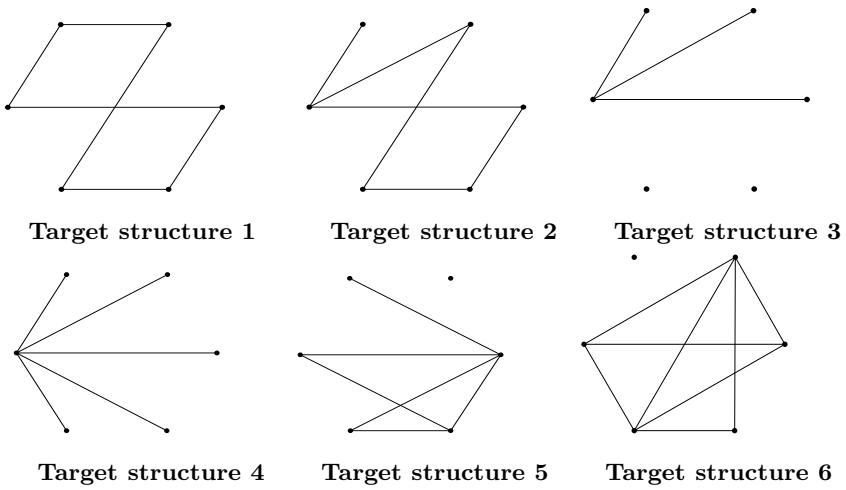


Figure 2: Independence structures for the first set of experiments: model 1 is regular ($irr = 0$); model 2 has $irr = 10$; model 3 has $irr = 18$; model 4 has $irr = 20$; models 5 and 6 have the maximum irregularity for six variables ($irr = 26$).

between blankets. Thus, in terms of sample complexity, we expect larger improvements of BJP over the competitors when the irregularity of the underlying structure increases.

For constructing a probability distribution from these independence structures according to (2), random numeric values were assigned to the parameters of their maximal clique factors, sampled independently from a uniform distribution over $(0, 1)$. Ten distributions were generated for each target structure, considering only binary discrete variables. Then, for each one, ten different random seeds were used to obtain 100 datasets for each graph, by using the Gibbs sampling tool of the open-source Libra toolkit [43]. The Gibbs sampler was run with 100 burn-in and $100,000$ sampling iterations.

Since we have $n = 6$ variables, the search space consists of $2^{\binom{6}{2}} = 32768$ different undirected graphs. The experiment consisted in evaluating the number of true structures returned by each score over the 100 datasets. This is called here the success rate of the scoring function. The success rate is computed for increasing dataset sizes $\mathcal{N}_D = \{250, 500, 1000, 2000, 4000, 8000\}$. Of course,

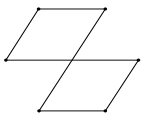
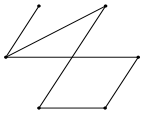
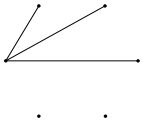
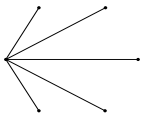
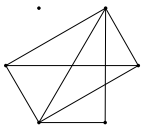
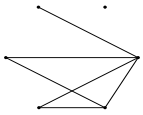
Target structure	Irr	$\mathcal{N}_D$	Success rate MPL	IB-score	BJP
1 	0	250	0.00	0.00	0.00
		500	0.00	0.00	0.01
		1000	0.01	0.05	0.03
		2000	0.04	0.15	0.12
		4000	0.15	0.25	0.21
		8000	0.28	0.35	0.34
2 	10	250	0.00	0.00	0.000
		500	0.00	0.00	0.01
		1000	0.00	0.04	0.02
		2000	0.02	0.15	0.16
		4000	0.10	0.27	0.25
		8000	0.18	0.39	0.39
3 	18	250	0.00	0.06	0.04
		500	0.03	0.09	0.12
		1000	0.10	0.17	0.19
		2000	0.17	0.22	0.27
		4000	0.22	0.45	0.49
		8000	0.34	0.58	0.61
4 	20	250	0.00	0.00	0.00
		500	0.00	0.03	0.02
		1000	0.00	0.06	0.10
		2000	0.00	0.14	0.18
		4000	0.00	0.29	0.36
		8000	0.00	0.44	0.50
5 	26	250	0.00	0.01	0.01
		500	0.00	0.02	0.01
		1000	0.00	0.10	0.11
		2000	0.00	0.23	0.26
		4000	0.03	0.56	0.54
		8000	0.21	0.75	0.76
6 	26	250	0.00	0.00	0.00
		500	0.00	0.00	0.00
		1000	0.00	0.04	0.13
		2000	0.00	0.28	0.37
		4000	0.02	0.66	0.61
		8000	0.27	0.80	0.82

Table 1: Success rate of BJP, IB-score and MPL over 100 datasets for the target structures on Figure 2. For each row, the ranking of the methods is represented by the shade of the cells, such that the lightest cell marks the highest success rate and the darkest cell marks the lowest success rate.

since greater sizes of the dataset lead to better estimations, $\mathcal{N}_D$ affects the quality of the structure learned. Therefore, a score is considered better than another score when its success rate converges to 1 for lower values of $\mathcal{N}_D$.

Table 1 shows the results of the experiment. The first column shows the target structures, the second shows their irregularity, the third shows each sample size $\mathcal{N}_D$ used, and the fourth shows the success rate. For all the cases, it can be seen how the success rate of the three scoring functions grows with the sample size $\mathcal{N}_D$. At each row, the ranking of the methods is represented by the shade of the cells, such that the lightest cell marks the highest success rate and the darkest cell marks the lowest success rate. The results in the fourth column show that BJP has a better success rate in almost all cases. For all the cases, MPL has a slower convergence than IB-score and BJP. For structures 1 and 2, IB-score shows better convergence than BJP, but they would eventually converge similarly for greater $\mathcal{N}_D$ sizes. This is an expected result, because these structures are regular, and the approximation of BJP and IB-score are very similar for computing $\Pr(G|D)$. In contrast, for structures 3, 4, 5 and 6, BJP has in general the best success rate. This is also an expected result, according to the irregularity of the underlying structures. Accordingly, the best improvement of BJP over IB-score is for model 6 (which has maximal irregularity) and $\mathcal{N}_D = \{1000, 2000\}$, with an improvement of success rate of up to 9%. When compared with MPL, BJP obtains the best improvement in success rate of up to 59%, also for model 6 and $\mathcal{N}_D = \{4000\}$.

In general, these results are consistent with the hypothesis of this work, since BJP has been designed to improve the computation of $\Pr(G|D)$, and the irregularity highlights the cases where an improvement of the sample complexity is expected, due to the independence assumption between blankets made by the state-of-the-art scores. The following section shows the performance of the three scoring functions for more complex domains.

4.2. Structural errors analysis

In this section, experiments in the higher-dimensional setting are presented. The goal here is to show that, for more practical scenarios, structure learning using the BJP score can improve the quality of the structures over state-of-the-art competitors. In order to ensure tractability, the experiments in this section

require approximate search mechanisms for the optimization of each score. As described in Section 3.2, the inherent decomposability properties of each function make them suitable for different optimization methods. For this reason, a comparison using a single search algorithm for all scores would arbitrarily bias the results and would not reflect realistic applications. Therefore, we have decided to evaluate each score using the optimization method proposed by its authors as the most favorable alternative, in hopes of achieving a reasonably fair comparison. We compared BJP against the following state-of-the-art structure learning methods, which are also applicable in high dimensions:

- *GSMN*: The Grow-Shrink Markov network structure learning algorithm [21]. This is a standard state-of-the-art constraint-based algorithm. GSMN proceeds by learning the blanket of each variable with the well-known GS algorithm [40], and then constructs the solution structure by adding an edge between each variable and the variables found in its Markov blanket. This algorithm is very efficient, as it is not a search-based algorithm, but very prone to errors when data is not sufficient for performing accurate statistical independence tests.

- *IB-score*: The Independence-based score [28], optimized by using IBMAP-HC (the heuristic hill-climbing optimization explained in Section 2.2.2).

- *MPL*: The Marginal pseudo-likelihood score [19], optimized by using the efficient method in two phases proposed by its authors (described in Section 2.2.1).

Note that, in the scope of this set of experiments, we use the name of each score to refer to the combination of optimization method and scoring function as described above.

The selected structures for the experiments capture the properties of several real-world problems, where the target structure has few nodes with large degrees, and the remaining nodes have very small degree. Examples of problems with this characteristic include gene networks, protein interaction networks and

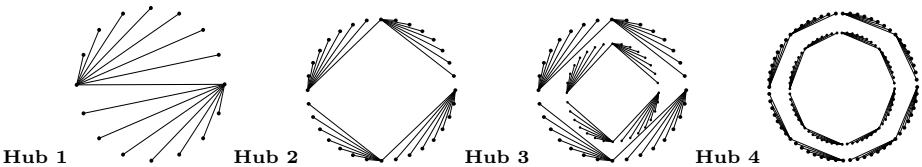


Figure 3: Structures with a hub topology and 16, 32, 64 and 128 nodes

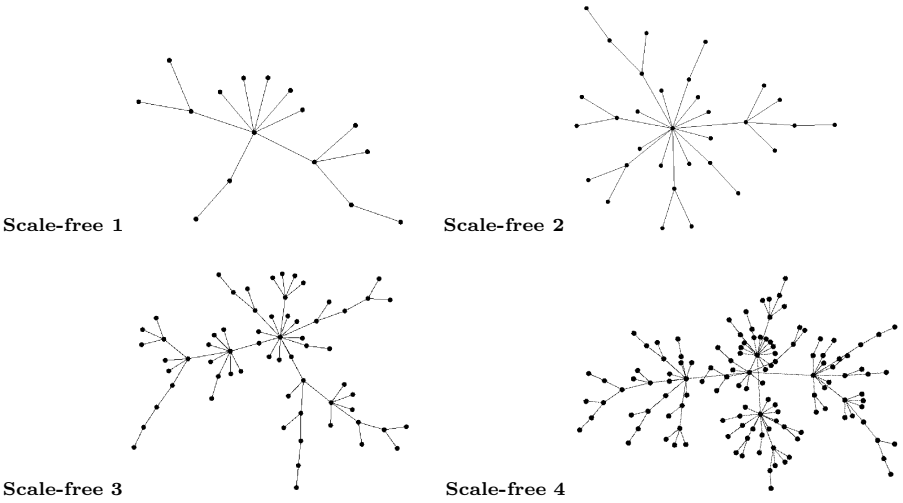
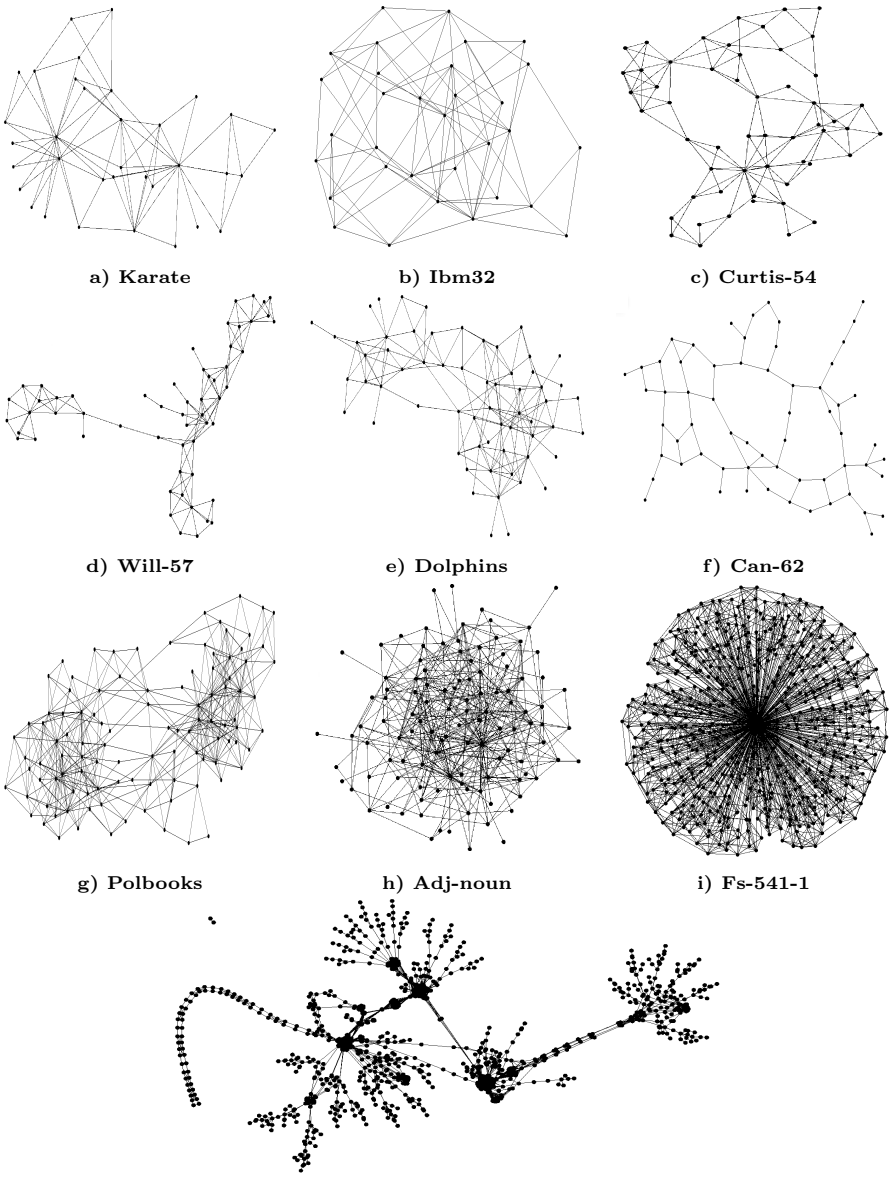


Figure 4: Scale-free structures with 16, 32, 64 and 128 nodes

social networks [41]. Thus, for this comparative study, we used three types of structures: networks with hub topologies, scale-free networks generated by the Barabasi-Albert model [44], and real-world networks, taken from the sparse matrix collection [45]. These structures have an increasing complexity both in n and in irr. The hub networks are shown in Figure 3, the scale-free networks are shown in Figure 4, and the real-world networks are shown in Figure 5.

For each target structure we generated 10 random distributions and 10 random samples for each distribution, with the Gibbs sampler tool of the Libra toolkit. Thus, a total of 100 datasets were obtained for each graph, with the same procedure explained in the previous section. As a quality measure, we report the type-I errors (false positives), type-II errors (false negatives), and



a) Karate **b) Ibm32** **c) Curtis-54**

d) Will-57 **e) Dolphins** **f) Can-62**

g) Polbooks **h) Adj-noun** **i) Fs-541-1**

i) Eris-1176

Figure 5: Real-world networks

Hamming distance (sum of false positives and false negatives) between the hundred learned structures and the underlying one. We measure the statistical significance of these quality measures by comparing the average and standard deviation over the 100 repetitions, where by statistically significant we simply mean that there is no overlap between the intervals of the means plus/minus their standard deviations. As in the previous section, the algorithms were executed for increasing dataset sizes $\mathcal{N}_D = \{250, 500, 1000, 2000, 4000, 8000\}$, to assess how their accuracy evolves with data availability.

Table 2 shows the comparison of BJP against its competitors for the hub structures of Figure 3. The table shows the name of the structures, their sizes n, and their irregularities, in the first, second and third columns, respectively. The dataset sizes $\mathcal{N}_D$ are in the fourth column. The next columns show the average and standard deviation of type-I errors, type-II errors, and the Hamming distance over the 100 repetitions for all the structure learning algorithms: GSMN, MPL, IB-score and BJP. At each row of the table, the ranking of the Hamming distance is represented by the shade of the cells, such that the lightest cell marks the lowest Hamming distance (best results) and the darkest cell marks the highest Hamming distance. Additionally, the plots of Figure 6 shows a summary of the improvements of BJP over each competitior, with a bar for each dataset and each dataset size. Table 3 shows the runtime (in seconds) corresponding to the whole learning process. All the experiments were performed on an Intel(R) Core(TM) i7-4770 CPU, with 3.40GHz, and 32 GB of main memory.

When analyzing these results, it can be seen that for all the algorithms the more complex the underlying structure (determined by n and irr), the larger is the number of structural errors (Hamming distance column) for any score and any value of $\mathcal{N}_D$. The results show that BJP obtains the best performance for all the cases, reducing the number of average Hamming distance errors of the structures learned by its competitors. It can be seen that, for all the target structures, GSMN has the slowest convergence in $\mathcal{N}_D$. Since GSMN follows a traditional constraint-based approach, it is expected for it to obtain low qualities when data are insufficient. When compared with both MPL and IB-score,

Target structure	n	irr	$\mathcal{N}_D$	Structural errors											
				GSMN			MPL			IB-score			BJP		
				Type-I	**Type-II**	**HD**	**Type-I**	**Type-II**	**HD**	**Type-I**	**Type-II**	**HD**	**Type-I**	**Type-II**	**HD**
Hub 1	16	392	250	11.84 (1.06)	7.75 (0.35)	19.59 (0.98)	0.90 (0.12)	12.23 (0.13)	13.13 (0.19)	0.24 (0.19)	7.33 (0.96)	7.91 (1.04)	0.64 (0.25)	6.79 (0.91)	7.76 (1.04)
			500	7.89 (0.81)	6.17 (0.39)	14.06 (0.81)	0.52 (0.09)	11.22 (0.14)	11.74 (0.16)	0.12 (0.15)	5.83 (0.80)	6.28 (0.85)	0.60 (0.22)	5.01 (0.72)	5.95 (0.82)
			1000	5.02 (0.50)	4.57 (0.34)	9.59 (0.57)	0.24 (0.06)	10.19 (0.13)	10.43 (0.14)	0.05 (0.14)	4.59 (0.68)	4.87 (0.71)	0.41 (0.22)	3.72 (0.58)	4.47 (0.67)
			2000	3.45 (0.37)	3.60 (0.31)	7.05 (0.48)	0.19 (0.05)	9.20 (0.14)	9.39 (0.15)	0.07 (0.16)	3.29 (0.54)	3.69 (0.59)	0.43 (0.19)	2.51 (0.45)	3.27 (0.54)
			4000	2.65 (0.30)	2.51 (0.28)	5.16 (0.36)	0.14 (0.05)	8.06 (0.14)	8.20 (0.14)	0.11 (0.13)	2.15 (0.42)	2.37 (0.46)	0.33 (0.20)	1.63 (0.36)	2.29 (0.43)
			8000	1.84 (0.25)	1.94 (0.24)	3.78 (0.33)	0.12 (0.05)	7.13 (0.14)	7.25 (0.14)	0.09 (0.13)	1.53 (0.35)	1.77 (0.39)	0.09 (0.16)	1.16 (0.30)	1.59 (0.36)
Hub 2	32	1916	250	74.31 (2.50)	15.31 (0.47)	89.62 (2.46)	2.84 (0.20)	24.42 (0.17)	27.26 (0.29)	0.84 (0.13)	24.95 (0.23)	25.79 (0.25)	2.39 (0.23)	22.69 (0.30)	25.08 (0.33)
			500	48.97 (2.29)	12.16 (0.47)	61.12 (2.32)	1.84 (0.17)	22.52 (0.17)	24.36 (0.25)	0.81 (0.14)	21.19 (0.26)	22.00 (0.29)	1.93 (0.24)	18.05 (0.37)	19.98 (0.40)
			1000	30.33 (1.51)	9.31 (0.49)	39.64 (1.49)	1.00 (0.13)	20.56 (0.17)	21.56 (0.23)	0.49 (0.10)	17.01 (0.31)	17.50 (0.34)	1.90 (0.24)	13.59 (0.33)	15.49 (0.44)
			2000	19.85 (1.18)	6.70 (0.42)	26.55 (1.24)	0.47 (0.08)	18.47 (0.17)	18.94 (0.20)	0.44 (0.10)	12.42 (0.34)	12.86 (0.34)	1.97 (0.21)	9.61 (0.30)	11.58 (0.31)
			4000	14.76 (0.94)	5.07 (0.35)	19.83 (1.00)	0.34 (0.08)	16.34 (0.18)	16.68 (0.20)	0.23 (0.07)	9.11 (0.32)	9.34 (0.32)	1.64 (0.19)	6.72 (0.24)	8.36 (0.30)
			8000	10.30 (0.68)	4.02 (0.34)	14.32 (0.74)	0.29 (0.07)	14.27 (0.17)	14.56 (0.18)	0.30 (0.07)	6.75 (0.28)	7.05 (0.27)	1.91 (0.20)	5.05 (0.23)	6.96 (0.28)
Hub 3	64	6624	250	267.81 (2.68)	33.91 (0.74)	301.72 (2.66)	8.97 (0.31)	51.56 (0.28)	60.53 (0.49)	0.40 (0.10)	56.16 (0.31)	56.56 (0.31)	2.37 (0.25)	51.68 (0.41)	54.05 (0.47)
			500	226.12 (4.36)	28.21 (0.89)	254.33 (4.32)	5.47 (0.27)	47.41 (0.29)	52.88 (0.44)	0.18 (0.05)	50.37 (0.38)	50.55 (0.38)	2.03 (0.21)	42.83 (0.56)	44.86 (0.53)
			1000	153.75 (4.11)	23.31 (0.80)	177.06 (4.16)	3.34 (0.21)	42.88 (0.32)	46.22 (0.44)	0.20 (0.07)	42.14 (0.50)	42.34 (0.52)	1.85 (0.24)	34.53 (0.67)	36.38 (0.66)
			2000	92.65 (3.00)	17.81 (0.75)	110.46 (3.11)	2.09 (0.19)	38.26 (0.36)	40.35 (0.43)	0.17 (0.07)	33.30 (0.63)	33.47 (0.63)	1.82 (0.27)	27.42 (0.81)	29.24 (0.78)
			4000	57.68 (1.74)	13.64 (0.51)	71.32 (1.69)	1.13 (0.14)	33.86 (0.39)	34.99 (0.43)	0.08 (0.04)	26.20 (0.67)	26.28 (0.67)	1.46 (0.23)	21.04 (0.84)	22.50 (0.78)
			8000	42.06 (1.60)	10.68 (0.49)	52.74 (1.55)	0.68 (0.12)	29.90 (0.36)	30.58 (0.39)	0.10 (0.05)	20.71 (0.77)	20.81 (0.76)	1.60 (0.27)	17.89 (0.91)	19.49 (0.82)
Hub 4	128	24496	250	605.32 (2.79)	74.57 (1.04)	679.89 (3.17)	29.66 (0.52)	104.62 (0.41)	134.28 (0.81)	0.06 (0.06)	106.61 (5.58)	106.67 (5.59)	1.57 (0.27)	92.24 (6.19)	93.98 (6.31)
			500	664.65 (3.81)	60.53 (1.17)	725.18 (3.98)	18.87 (0.39)	95.09 (0.41)	113.96 (0.64)	0.10 (0.05)	97.71 (5.14)	97.72 (5.14)	0.87 (0.26)	80.22 (5.43)	81.26 (5.50)
			1000	627.37 (5.99)	48.70 (1.24)	676.07 (5.96)	11.82 (0.36)	86.42 (0.44)	98.24 (0.67)	0.10 (0.05)	84.34 (4.47)	84.36 (4.47)	0.37 (0.16)	64.94 (4.47)	65.48 (4.50)
			2000	473.70 (7.23)	37.92 (1.07)	511.62 (7.24)	6.96 (0.33)	77.29 (0.44)	84.25 (0.60)	0.11 (0.05)	69.90 (3.75)	69.90 (3.75)	0.40 (0.16)	50.63 (3.63)	51.19 (3.65)
			4000	292.78 (5.61)	28.13 (1.02)	320.91 (5.76)	3.97 (0.23)	68.73 (0.48)	72.70 (0.53)	0.11 (0.05)	57.81 (3.28)	57.81 (3.28)	0.67 (0.27)	42.63 (3.37)	43.47 (3.37)
			8000	167.22 (4.07)	21.22 (0.79)	188.44 (4.05)	2.24 (0.23)	60.37 (0.52)	62.61 (0.61)	0.11 (0.05)	46.96 (3.07)	46.96 (3.07)	0.29 (0.17)	38.47 (3.53)	38.93 (3.51)

Table 2: Structures with hub topology: average and standard deviation of type-I errors, type-II errors and Hamming distance over 100 repetitions. For each row, the ranking of the Hamming distance is represented by the shade of the cells, such that the lightest cell marks the lowest Hamming distance and the darkest cell marks the highest Hamming distance.

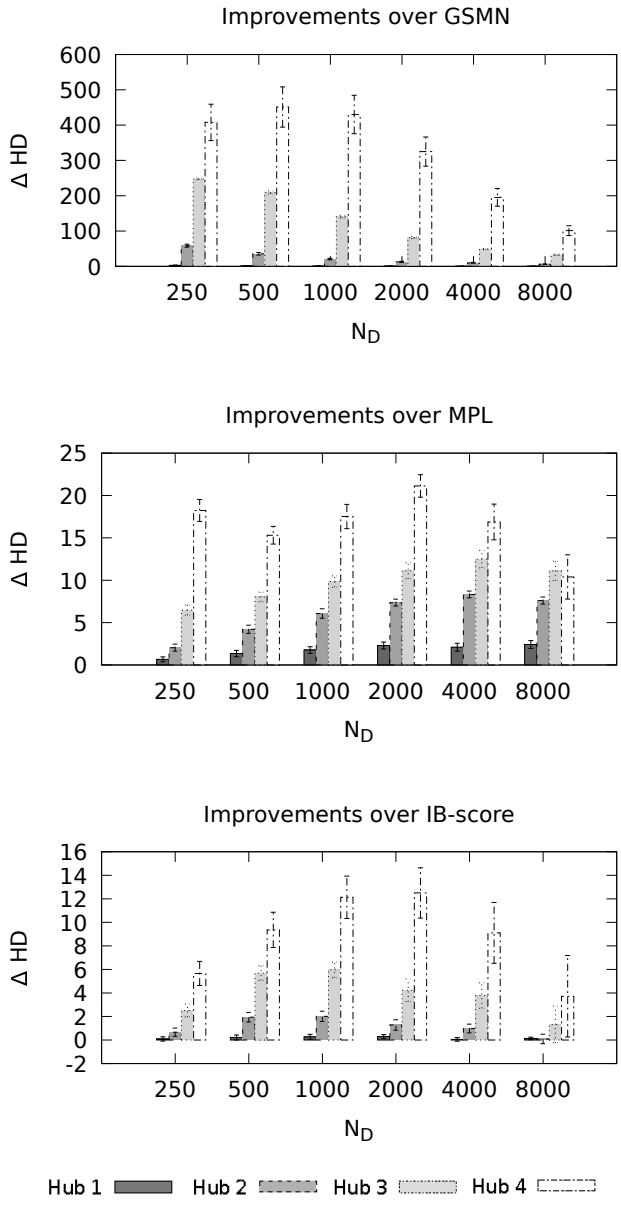


Figure 6: Hamming distance improvements of BJP over competitors for structures with hub topology. Δ HD denotes the improvement in Hamming distance over each competitor.

the improvements of the BJP score are larger as the complexity (n and irr) grows. These improvements are statistically significant for all the cases against

GSMN and MPL. Against IB-score, the improvements of BJP can be seen for all the cases, except three. In general, these results confirm that BJP is more accurate can improve the quality of the learning process against competitors with better improvements when the structures are highly irregular. Regarding the type-I and type-II errors (false positives and false negatives), it can be seen that GSMN tends to add many false positives, whereas the other score-and-search methods tend to add many false negatives. This is because GSMN adds false positives in the grow phase of the GS algorithm, and then the shrink phase must perform tests that contain many variables, which tend to be more unreliable, thus limiting the ability of the algorithm to delete incorrect edges. Instead, the other three score-and-search approaches lead to produce more false negatives, because the hill-climbing approaches used start the search from an empty structure.

In terms of the respective runtimes shown (in seconds) in Table 3, it can be seen that for all the algorithms, the more complex the underlying structure (determined by n and irr), the larger is the runtime for any value of D used. As expected, the most efficient approach is GSMN, since it follows a simple traditional constraint-based approach, that performs a polynomial number of statistical tests to learn the structure. Regarding the runtime of the search mechanism used with BJP, it can be seen that it compares favorably to the other similar search-and-score solutions (MPL and IB-score). Although we simply used an existent optimization method for BJP, it shows a good performance in both quality and runtime, when compared against competitors.

Table 4 shows the comparison of BJP against its competitors for the scale-free networks of Figure 4. The information of the table is organized in the same way as in Table 2. The summary of the improvements of BJP over each competitior can be seen in the plots of Figure 7 with a bar for each dataset and each dataset size. In contrast with the hub structures, in the scale-free networks the size of the blankets in the underlying network is more variable. This can explain the differences in the trends of the Hamming distance, when compared with the results obtained for the hub networks. It can be seen that for all

Target structure	n	irr	$\mathcal{N}_D$	Runtime GSMN	MPL	IB-score	BJP
			250	0.20 (0.00)	0.16 (0.00)	0.20 (0.00)	0.06 (0.01)
			500	0.82 (0.00)	0.14 (0.02)	0.29 (0.05)	0.11 (0.01)
			1000	0.12 (0.00)	0.19 (0.02)	0.74(0.02)	0.25 (0.04)
Hub 1	16	392	2000	0.25 (0.01)	0.41 (0.06)	2.39 (0.08)	0.60 (0.07)
			4000	0.47 (00.01)	1.09 (0.01)	6.75 (0.22)	1.34 (0.02)
			8000	0.86 (00.02)	2.90 (0.05)	17.53 (0.59)	2.59 (0.02)
			250	0.49 (0.01)	0.42 (4.94)	0.81 (0.01)	0.39 (0.00)
			500	0.34 (0.01)	0.59 (0.00)	1.50 (0.01)	0.92 (0.01)
			1000	0.51 (0.01)	1.35 (0.02)	3.87 (0.04)	2.15 (0.02)
Hub 2	32	1916	2000	0.99 (0.04)	3.00 (0.05)	11.39 (0.14)	5.28 (0.05)
			4000	1.94 (0.06)	7.67 (0.10)	29.32 (0.36)	11.63 (0.09)
			8000	3.18 (0.13)	22.45 (0.28)	76.58 (1.03)	23.75 (0.18)
			250	2.10 (0.17)	3.09 (0.03)	1.79 (0.02)	1.37 (0.00)
			500	2.15 (0.18)	4.90 (63.37)	4.96 (0.07)	3.86 (0.05)
			1000	3.05 (0.21)	10.33 (0.11)	17.24 (0.22)	10.39 (0.12)
Hub 3	64	6624	2000	4.49 (0.36)	24.73 (0.28)	57.95 (0.81)	25.991 (0.38)
			4000	8.76 (0.76)	61.75 (0.66)	180.92 (3.02)	63.64 (0.83)
			8000	19.47 (1.62)	207.48 (2.08)	627.50 (11.27)	156.24 (3.15)
			250	7.24 (0.44)	58.92 (0.32)	5.86 (0.13)	8.31 (0.13)
			500	10.95 (0.58)	78.53 (0.49)	26.56 (0.42)	25.14 (0.37)
			1000	20.00 (1.02)	129.33 (0.77)	101.26 (1.05)	74.80 (0.77)
Hub 4	128	24496	2000	34.28 (1.94)	259.68 (1.74)	331.32 (3.19)	198.77 (2.14)
			4000	61.08 (3.58)	777.84 (6.36)	1252.88 (19.89)	473.05 (6.97)
			8000	120.64 (6.35)	3102.53 (28.43)	4913.07 (89.81)	1185.91 (23.83)

Table 3: Structures with hub topology: average and standard deviation of the learning runtime (in seconds) over 100 repetitions. For each row, the ranking of the runtimes is represented by the shade of the cells, such that the lightest cell marks the lowest runtime and the darkest cell marks the highest runtime.

Target structure	n	irr	$\mathcal{N}_D$	Structural errors											
				GSMN			MPL			IB-score			BJP		
				Type-I	Type-II	HD	Type-I	Type-II	HD	Type-I	Type-II	HD	Type-I	Type-II	HD
Scale-free 1	16	364	250	10.90 (2.92)	8.80 (0.95)	19.70 (2.60)	0.70 (0.12)	11.65 (0.18)	12.35 (0.24)	0.30 (0.31)	11.00 (1.03)	11.30 (1.05)	1.00 (0.37)	10.20 (0.90)	11.20 (0.80)
			500	9.10 (2.36)	5.90 (1.18)	15.00 (2.69)	0.39 (0.08)	10.24 (0.16)	10.63 (0.18)	0.60 (0.49)	9.40 (0.81)	10.00 (0.93)	1.30 (0.68)	8.70 (0.83)	10.00 (1.03)
			1000	6.70 (1.27)	4.70 (0.91)	11.40 (1.31)	0.24 (0.06)	8.90 (0.18)	9.14 (0.20)	0.30 (0.22)	6.80 (1.02)	7.10 (1.06)	1.20 (0.29)	6.10 (0.82)	7.30 (0.74)
			2000	3.30 (1.21)	3.70 (0.61)	7.00 (1.58)	0.08 (0.04)	7.45 (0.16)	7.53 (0.16)	0.50 (0.24)	4.60 (0.65)	5.10 (0.69)	0.60 (0.23)	4.60 (0.81)	5.20 (0.67)
			4000	2.90 (1.14)	2.30 (0.48)	5.20 (0.95)	0.02 (0.02)	6.19 (0.15)	6.21 (0.15)	0.40 (0.32)	3.30 (0.57)	3.70 (0.48)	0.50 (0.44)	3.00 (0.61)	3.50 (0.58)
			8000	1.00 (0.57)	2.00 (0.57)	3.00 (0.86)	0.02 (0.02)	4.90 (0.15)	4.92 (0.15)	0.00 (0.00)	2.30 (0.65)	2.30 (0.65)	0.30 (0.22)	2.00 (0.57)	2.30 (0.71)
Scale-free 2	32	1612	250	70.60 (5.99)	16.50 (0.84)	87.10 (6.26)	2.62 (0.17)	24.89 (0.21)	27.51 (0.31)	1.10 (0.45)	25.40 (1.12)	26.50 (1.12)	3.38 (1.08)	22.50 (1.25)	25.88 (1.23)
			500	43.20 (3.26)	14.80 (1.30)	58.00 (2.47)	1.33 (0.15)	22.75 (0.24)	24.08 (0.31)	0.30 (0.31)	22.10 (1.22)	22.40 (1.38)	1.62 (0.62)	18.75 (1.24)	20.38 (1.67)
			1000	27.70 (4.14)	11.90 (0.69)	39.60 (3.75)	0.74 (0.12)	20.08 (0.24)	20.82 (0.29)	0.50 (0.24)	17.80 (1.19)	18.30 (1.29)	1.00 (0.56)	16.12 (1.23)	17.12 (1.32)
			2000	18.90 (1.16)	8.80 (0.60)	27.70 (1.21)	0.48 (0.10)	17.79 (0.22)	18.27 (0.25)	0.90 (0.45)	12.70 (1.03)	13.60 (0.86)	1.75 (0.91)	10.38 (0.79)	12.12 (0.98)
			4000	12.40 (1.68)	7.20 (0.88)	19.60 (1.82)	0.37 (0.08)	15.76 (0.19)	16.13 (0.21)	1.20 (0.36)	9.20 (1.05)	10.40 (1.14)	2.12 (0.94)	8.38 (1.28)	10.50 (1.25)
			8000	7.70 (1.14)	4.90 (0.92)	12.60 (1.44)	0.24 (0.07)	14.17 (0.22)	14.41 (0.23)	0.33 (0.25)	6.22 (0.92)	6.56 (1.05)	1.50 (0.62)	5.50 (0.74)	7.00 (0.79)
Scale-free 3	64	6428	250	262.30 (5.83)	35.10 (1.85)	297.40 (5.04)	8.33 (0.33)	50.78 (0.39)	59.11 (0.63)	0.25 (0.39)	57.50 (2.61)	57.75 (2.60)	0.67 (1.01)	54.67 (1.01)	55.33 (1.01)
			500	209.90 (8.14)	29.50 (1.65)	239.40 (8.19)	4.29 (0.24)	45.85 (0.42)	50.14 (0.56)	0.25 (0.39)	51.75 (3.03)	52.00 (3.21)	0.67 (0.51)	43.33 (4.14)	44.00 (3.82)
			1000	156.90 (6.52)	23.80 (1.71)	180.70 (6.70)	2.42 (0.18)	40.63 (0.42)	43.05 (0.51)	0.00 (0.00)	43.25 (6.39)	43.25 (6.39)	0.33 (0.51)	35.67 (5.35)	36.00 (4.88)
			2000	104.40 (5.09)	18.90 (1.77)	123.30 (5.60)	1.36 (0.16)	35.35 (0.45)	36.71 (0.51)	0.25 (0.39)	33.25 (4.64)	33.50 (4.56)	3.33 (2.82)	24.33 (3.08)	27.67 (3.65)
			4000	57.40 (7.50)	15.40 (1.18)	72.80 (7.07)	0.72 (0.11)	30.65 (0.38)	31.37 (0.38)	0.00 (0.00)	26.25 (2.26)	26.25 (2.26)	1.33 (1.34)	20.00 (2.32)	21.33 (1.01)
			8000	34.70 (2.64)	11.60 (1.38)	46.30 (2.76)	0.55 (0.10)	26.97 (0.38)	27.52 (0.39)	0.00 (0.00)	19.00 (1.70)	19.00 (1.70)	0.67 (1.01)	15.33 (2.53)	16.00 (3.16)
Scale-free 4	128	26188	250	599.60 (8.98)	74.80 (3.61)	674.40 (10.44)	26.91 (0.76)	104.51 (0.74)	131.42 (1.37)	0.00 (0.00)	123.20 (0.70)	123.20 (0.70)	3.80 (1.25)	112.60 (1.49)	116.40 (1.76)
			500	667.00 (8.13)	59.40 (2.51)	726.40 (9.40)	15.95 (0.60)	93.49 (0.80)	109.44 (1.23)	0.00 (0.00)	110.70 (1.91)	110.70 (1.91)	0.20 (0.19)	100.80 (2.53)	101.00 (2.48)
			1000	635.80 (12.85)	49.90 (2.03)	685.70 (12.09)	9.14 (0.48)	82.33 (0.82)	91.47 (1.11)	0.10 (0.14)	92.90 (2.56)	93.00 (2.59)	0.70 (0.48)	82.40 (3.01)	83.10 (3.07)
			2000	492.00 (15.20)	41.20 (2.22)	533.20 (15.53)	4.89 (0.37)	72.58 (0.79)	77.47 (0.99)	0.10 (0.14)	79.10 (3.31)	79.20 (3.30)	1.00 (0.37)	63.50 (3.76)	64.50 (3.83)
			4000	308.70 (11.72)	31.20 (2.42)	339.90 (10.25)	2.40 (0.27)	63.04 (0.82)	65.44 (0.91)	0.00 (0.00)	62.00 (3.22)	62.00 (3.22)	1.10 (0.69)	45.40 (3.20)	46.50 (3.12)
			8000	164.90 (8.91)	24.30 (2.03)	189.20 (9.32)	1.51 (0.21)	55.58 (0.82)	57.09 (0.85)	0.00 (0.00)	47.90 (2.50)	47.90 (2.50)	1.10 (0.84)	33.20 (2.41)	k34.30 (2.53)

Table 4: Scale-free networks models: average and standard deviation of type-I errors, type-II errors and Hamming distance over 100 repetitions. For each row, the ranking of the Hamming distance is represented by the shade of the cells, such that the lightest cell marks the lowest Hamming distance and the darkest cell marks the highest Hamming distance.

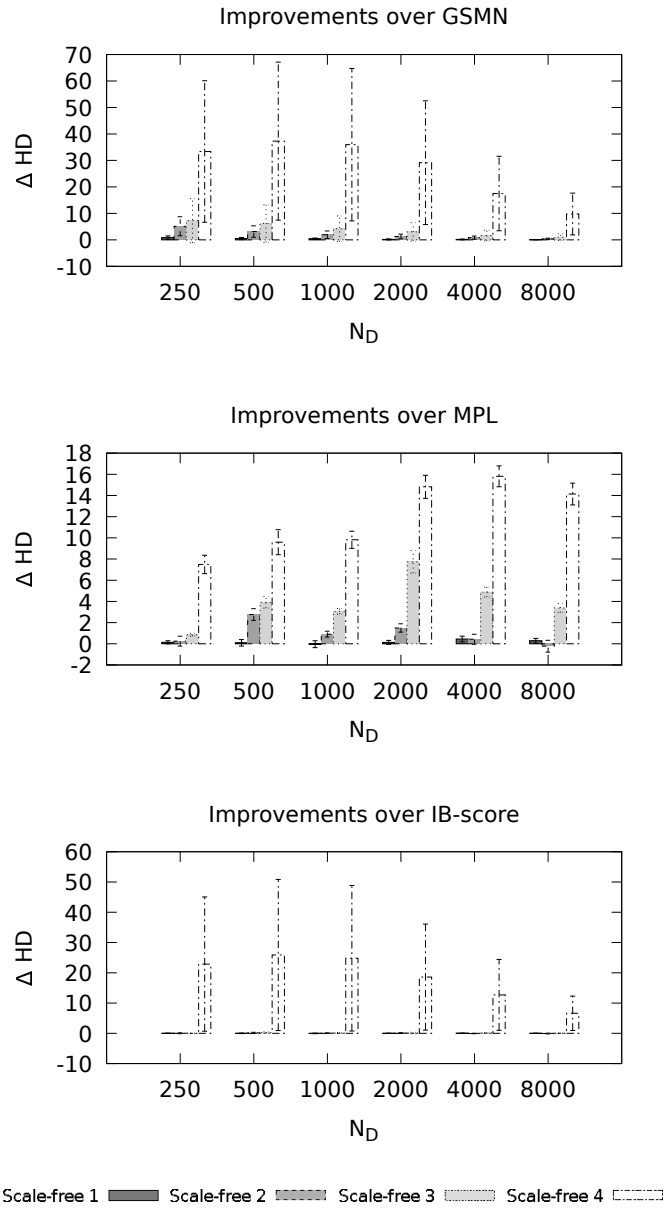


Figure 7: Hamming distance improvements of BJP over competitors for scale-free network models. Δ HD denotes the improvement in Hamming distance over each competitor.

the cases BJP reduces the number of average Hamming distance of GSMN and MPL. The improvements over MPL are statistically significant for all the cases,

except three. When compared with IB-score, BJP shows better average number of errors for all the cases, except three. As illustrated in Figure 7, the best improvements of BJP over the three competitors can be seen for the Scale-free 4 model, with improvements of more than 30 edges corrected against GSMN. Against MPL and IB-score, again the best improvements of BJP can be seen for the Scale-free 4 model, with improvements of more than 15 edges corrected. In general, these results confirm that the approximation of BJP is more accurate as n and irr grow. Regarding the two types of errors (false positives and false negatives), and the runtimes shown (in seconds) in Table 5, it can be seen that they are similar to the case of the hub networks.

Target structure	n	irr	$\mathcal{N}_D$	**Runtime**			
				GSMN	**MPL**	**IB-score**	**BJP**
Scale-free 1	16	364	250	0.33 (0.04)	0.12 (0.01)	0.33 (0.08)	0.12 (0.02)
			500	0.12 (0.03)	0.12 (0.01)	0.40 (0.07)	0.16 (0.02)
			1000	0.20 (0.06)	0.25 (0.02)	0.77 (0.10)	0.36 (0.03)
			2000	0.30 (0.06)	0.70 (0.04)	1.89 (0.27)	0.72 (0.07)
			4000	0.55 (0.12)	1.90 (0.10)	3.88 (0.71)	1.42 (0.09)
			8000	0.78 (0.08)	6.46 (0.49)	7.91 (1.20)	2.91 (0.18)
Scale-free 2	32	1612	250	0.63 (0.08)	0.50 (0.02)	0.93 (0.15)	0.56 (0.15)
			500	0.40 (0.07)	0.79 (0.04)	1.35 (0.16)	0.96 (0.15)
			1000	0.51 (0.06)	2.12 (0.11)	4.35 (0.74)	1.73 (0.21)
			2000	0.76 (0.04)	5.18 (0.25)	19.52 (5.68)	5.20 (1.18)
			4000	1.58 (0.07)	12.58 (0.56)	77.37 (23.74)	10.52 (1.83)
			8000	2.93 (0.23)	41.34 (2.66)	354.14 (128.41)	25.33 (6.36)
Scale-free 3	64	6428	250	1.81 (0.09)	4.74 (0.17)	3.83 (1.54)	2.11 (0.49)
			500	1.79 (0.19)	8.20 (0.31)	9.86 (2.01)	6.06 (1.29)
			1000	2.65 (0.19)	19.55 (0.71)	29.05 (9.65)	13.87 (2.70)
			2000	4.10 (0.29)	46.99 (1.61)	95.45 (22.52)	46.07 (4.06)
			4000	6.60 (0.34)	122.24 (4.47)	275.86 (31.93)	99.06 (8.25)
			8000	14.27 (1.35)	433.10 (15.46)	1124.33 (384.80)	221.92 (17.90)
Scale-free 4	128	26188	250	7.48 (0.50)	72.69 (2.45)	6.71 (0.83)	12.50 (2.81)
			500	11.05 (0.43)	106.37 (3.55)	36.51 (4.01)	30.74 (7.84)
			1000	20.09 (0.98)	196.18 (6.70)	140.10 (11.26)	95.15 (20.05)
			2000	34.30 (1.23)	429.13 (17.04)	404.00 (39.31)	271.97 (60.10)
			4000	58.12 (2.76)	1202.84 (65.42)	1469.52 (182.56)	634.66 (61.60)
			8000	104.42 (6.74)	5103.34 (373.99)	7650.26 (1314.47)	1736.44 (282.31)

Table 5: Scale-free networks models: average and standard deviation of the learning runtime (in seconds) over 100 repetitions. For each row, the ranking of the runtimes is represented by the shade of the cells, such that the lightest cell marks the lowest runtime and the darkest cell marks the highest runtime.

Finally, Table 6 show the results for the real-world networks of Figure 5. Again, the information of this table is organized in the same way as in the previous tables, and the improvements of BJP are summarized in the plots of

Target structure	n	irr	$\mathcal{N}_D$	Structural errors: GSMN Type-I	GSMN Type-II	GSMN HD	MPL Type-I	MPL Type-II	MPL HD	IB-score Type-I	IB-score Type-II	IB-score HD	BJP Type-I	BJP Type-II	BJP HD
Karate	34	2044	250	64.20 (7.36)	22.10 (5.76)	86.30 (7.64)	3.90 (0.97)	54.70 (3.99)	58.60 (4.31)	2.09 (1.25)	49.82 (6.37)	51.91 (5.67)	2.60 (1.60)	49.30 (6.73)	51.90 (5.57)
			500	62.20 (5.29)	13.10 (5.17)	75.30 (3.91)	2.10 (0.97)	47.70 (3.01)	49.80 (3.50)	1.38 (1.01)	42.62 (8.21)	44.00 (8.00)	3.40 (1.11)	38.80 (5.02)	42.20 (5.17)
			1000	54.90 (3.56)	5.10 (3.60)	60.00 (2.92)	1.80 (1.04)	42.20 (2.70)	44.00 (3.37)	2.00 (1.01)	25.25 (8.45)	27.25 (8.54)	1.40 (0.68)	24.60 (6.91)	26.00 (7.06)
			2000	47.70 (3.10)	1.10 (1.17)	48.80 (3.80)	1.60 (0.89)	38.90 (1.92)	40.50 (1.92)	0.88 (0.71)	16.25 (7.56)	17.12 (7.95)	1.30 (0.82)	10.00 (4.99)	11.30 (4.89)
			4000	39.40 (6.47)	0.30 (0.48)	39.70 (6.60)	2.60 (1.30)	35.40 (1.64)	38.00 (1.05)	1.12 (1.06)	6.75 (3.45)	7.88 (3.44)	2.20 (1.19)	3.60 (2.86)	5.80 (2.99)
			8000	32.80 (6.96)	0.00 (0.00)	32.80 (6.96)	4.20 (0.93)	32.40 (1.49)	36.60 (1.01)	0.10 (0.22)	2.50 (0.76)	2.60 (0.76)	2.30 (1.29)	0.90 (0.62)	3.20 (1.28)
Ibm32	32	2228	250	55.60 (3.63)	25.70 (1.33)	81.30 (3.31)	4.80 (1.75)	60.50 (2.14)	65.30 (3.64)	7.50 (2.33)	57.00 (4.44)	64.50 (3.16)	6.50 (2.98)	54.30 (5.03)	60.80 (2.20)
			500	54.90 (4.38)	15.40 (2.49)	70.30 (4.27)	2.30 (1.11)	55.30 (1.05)	57.60 (1.92)	5.20 (1.78)	45.50 (3.11)	50.70 (2.97)	4.40 (1.97)	33.40 (3.17)	37.80 (3.01)
			1000	51.70 (3.19)	5.60 (1.25)	57.30 (3.70)	0.50 (0.50)	45.40 (1.30)	45.90 (1.43)	3.00 (1.45)	28.70 (2.90)	31.70 (2.40)	3.50 (1.11)	25.20 (2.97)	28.70 (2.62)
			2000	47.40 (4.13)	2.00 (1.20)	49.40 (4.17)	0.20 (0.30)	38.30 (1.79)	38.50 (1.92)	2.50 (0.95)	18.50 (3.74)	21.00 (3.86)	0.90 (0.62)	18.30 (2.62)	19.20 (2.88)
			4000	42.60 (4.69)	0.40 (0.49)	43.00 (4.63)	0.00 (0.00)	30.30 (1.66)	30.30 (1.66)	3.30 (1.25)	7.50 (1.86)	10.80 (2.02)	1.00 (0.74)	8.90 (1.77)	9.90 (2.17)
			8000	41.50 (4.61)	1.50 (1.16)	43.00 (4.94)	0.10 (0.14)	22.70 (1.12)	22.80 (1.13)	2.00 (1.20)	2.70 (1.56)	4.70 (1.82)	0.50 (0.50)	3.10 (1.02)	3.60 (1.06)
Curtis-54	54	3140	250	117.70 (5.56)	18.50 (2.71)	136.20 (5.60)	1.70 (0.94)	74.80 (1.59)	76.50 (2.33)	0.80 (0.45)	76.20 (4.18)	77.00 (4.22)	0.80 (0.56)	70.40 (3.99)	71.20 (3.67)
			500	118.30 (6.59)	7.80 (1.81)	126.10 (5.95)	0.40 (0.36)	64.00 (2.08)	64.40 (2.03)	0.40 (0.36)	58.70 (3.82)	59.10 (3.62)	0.20 (0.30)	56.40 (3.17)	56.60 (3.10)
			1000	121.30 (6.52)	2.00 (1.37)	123.30 (5.98)	0.00 (0.00)	52.40 (1.34)	52.40 (1.34)	0.00 (0.00)	40.10 (2.52)	40.10 (2.52)	0.10 (0.22)	39.30 (3.31)	39.40 (3.33)
			2000	112.10 (5.55)	0.80 (0.87)	112.90 (5.98)	0.00 (0.00)	40.10 (1.26)	40.10 (1.26)	0.10 (0.22)	22.60 (3.57)	22.70 (3.61)	0.30 (0.34)	18.60 (5.99)	18.90 (5.82)
			4000	93.90 (3.87)	0.20 (0.30)	94.10 (3.81)	0.10 (0.22)	30.20 (1.75)	30.30 (1.73)	0.00 (0.00)	7.50 (2.40)	7.50 (2.40)	0.60 (0.68)	3.80 (2.30)	4.40 (2.28)
			8000	78.50 (5.21)	0.10 (0.22)	78.60 (5.25)	0.10 (0.22)	23.90 (0.91)	24.00 (0.88)	0.00 (0.00)	2.12 (1.32)	2.12 (1.32)	1.30 (1.00)	0.90 (0.84)	2.20 (0.99)
Will-57	57	4156	250	127.00 (5.33)	22.70 (3.89)	149.70 (6.10)	1.50 (0.83)	78.00 (2.15)	79.50 (2.83)	0.60 (0.49)	81.30 (6.62)	81.90 (6.47)	0.70 (0.67)	78.70 (6.83)	79.40 (6.58)
			500	128.00 (4.69)	9.50 (2.19)	137.50 (5.47)	0.20 (0.30)	66.60 (2.13)	66.80 (2.07)	0.20 (0.30)	63.40 (3.54)	63.60 (3.52)	0.10 (0.22)	60.60 (5.90)	60.70 (5.84)
			1000	119.70 (7.40)	2.50 (1.25)	122.20 (8.30)	0.00 (0.00)	55.60 (1.67)	55.60 (1.67)	0.10 (0.22)	44.40 (3.46)	44.50 (3.43)	0.10 (0.22)	42.40 (5.98)	42.50 (5.86)
			2000	109.50 (5.10)	0.70 (0.67)	110.20 (5.04)	0.00 (0.00)	47.60 (0.83)	47.60 (0.83)	0.00 (0.00)	25.30 (3.72)	25.30 (3.72)	0.20 (0.45)	23.60 (4.76)	23.80 (4.56)
			4000	90.30 (6.42)	0.10 (0.22)	90.40 (6.49)	0.10 (0.22)	38.20 (1.19)	38.30 (1.37)	0.00 (0.00)	10.70 (3.15)	10.70 (3.15)	0.40 (0.59)	9.00 (6.72)	9.40 (6.55)
			8000	73.70 (4.69)	0.00 (0.00)	73.70 (4.69)	0.00 (0.00)	28.90 (0.84)	28.90 (0.84)	0.10 (0.22)	2.60 (0.89)	2.70 (0.82)	1.10 (1.07)	2.80 (2.80)	3.90 (2.71)
Can-62	62	4104	250	220.50 (3.85)	7.20 (1.28)	227.70 (4.68)	0.10 (0.22)	49.50 (3.07)	49.60 (2.98)	0.00 (0.00)	56.30 (2.21)	56.30 (2.21)	1.00 (0.74)	38.40 (1.49)	39.40 (1.97)
			500	226.30 (9.66)	1.40 (0.89)	227.70 (9.85)	0.10 (0.22)	36.20 (5.47)	36.30 (5.38)	0.00 (0.00)	40.50 (4.06)	40.50 (4.06)	0.00 (0.00)	14.00 (1.05)	14.00 (1.05)
			1000	213.40 (4.70)	0.10 (0.22)	213.50 (4.80)	0.20 (0.30)	7.80 (2.32)	8.00 (2.42)	0.00 (0.00)	9.50 (2.14)	9.50 (2.14)	0.00 (0.00)	7.60 (0.76)	7.60 (0.76)
			2000	157.80 (8.49)	0.00 (0.00)	157.80 (8.49)	1.30 (0.67)	3.40 (0.89)	4.70 (0.58)	1.10 (0.70)	2.00 (0.66)	3.10 (0.78)	0.00 (0.00)	2.60 (0.36)	2.60 (0.36)
			4000	95.80 (7.58)	0.00 (0.00)	95.80 (7.58)	2.40 (1.38)	0.00 (0.00)	2.40 (1.38)	0.40 (0.49)	1.60 (0.49)	2.00 (0.33)	0.00 (0.00)	1.00 (0.00)	1.00 (0.00)
			8000	51.00 (6.55)	0.00 (0.00)	51.00 (6.55)	1.45 (0.73)	0.00 (0.00)	1.45 (0.73)	0.20 (0.30)	0.00 (0.00)	0.20 (0.30)	0.00 (0.00)	0.10 (0.22)	0.10 (0.22)
Dolphins	62	6480	250	134.10 (8.32)	60.10 (6.59)	194.20 (7.91)	13.90 (2.81)	112.80 (4.31)	126.70 (6.20)	6.50 (3.88)	120.40 (11.55)	126.90 (8.04)	6.80 (4.48)	118.40 (10.12)	125.20 (6.42)
			500	127.20 (5.07)	35.20 (7.33)	162.40 (5.86)	6.00 (2.05)	100.60 (5.66)	106.60 (6.70)	2.90 (1.31)	103.20 (10.33)	106.10 (9.39)	5.30 (2.62)	96.80 (9.91)	102.10 (7.91)
			1000	114.00 (5.12)	17.00 (3.50)	131.00 (6.42)	2.10 (0.97)	86.40 (2.96)	88.50 (2.94)	2.60 (1.25)	69.00 (6.92)	71.60 (7.19)	4.40 (1.49)	61.50 (6.02)	65.90 (5.96)
			2000	101.10 (6.78)	6.70 (1.73)	107.80 (6.64)	0.50 (0.69)	73.70 (2.90)	74.20 (3.13)	1.60 (0.89)	49.00 (5.07)	50.60 (5.29)	3.10 (1.77)	44.20 (5.80)	47.30 (5.46)
			4000	88.10 (3.83)	4.30 (1.73)	92.40 (4.92)	0.60 (0.68)	62.40 (3.05)	63.00 (3.08)	0.80 (0.45)	31.70 (4.52)	32.50 (4.54)	1.60 (0.49)	26.10 (5.04)	27.70 (4.86)
			8000	85.40 (5.80)	7.30 (3.86)	92.70 (8.96)	0.20 (0.30)	50.60 (2.40)	50.80 (2.48)	0.70 (0.48)	19.90 (2.85)	20.60 (3.01)	2.70 (0.67)	10.20 (2.99)	12.90 (3.20)
Polbooks	105	30374	250	284.90 (6.41)	331.60 (6.48)	616.50 (11.70)	151.12 (19.55)	363.62 (7.21)	514.75 (14.43)	0.17 (0.20)	434.89 (1.26)	435.06 (1.19)	1.53 (0.59)	427.00 (1.74)	428.53 (1.96)
			500	287.00 (11.98)	304.10 (9.59)	591.10 (20.52)	114.29 (20.92)	364.71 (6.32)	479.00 (15.53)	0.06 (0.12)	428.22 (3.13)	428.28 (3.07)	0.73 (0.77)	417.53 (4.06)	418.27 (3.96)
			1000	282.10 (13.68)	276.80 (10.56)	558.90 (23.37)	76.71 (20.20)	362.29 (7.21)	439.00 (23.10)	0.11 (0.16)	414.00 (4.12)	414.11 (4.11)	0.07 (0.14)	407.87 (4.14)	407.93 (4.05)
			2000	269.00 (12.80)	250.50 (7.46)	519.50 (19.67)	50.43 (11.67)	359.00 (8.86)	409.43 (19.53)	0.00 (0.00)	399.72 (4.38)	399.72 (4.38)	0.00 (0.00)	393.53 (4.68)	393.53 (4.68)
			4000	254.70 (19.17)	233.40 (12.42)	488.10 (31.30)	32.00 (6.79)	346.86 (12.26)	378.86 (17.55)	2.33 (3.46)	379.39 (8.43)	381.72 (6.04)	0.87 (1.87)	374.73 (7.59)	375.60 (6.41)
			8000	237.50 (19.82)	216.40 (8.55)	453.90 (27.47)	19.14 (4.24)	334.00 (11.23)	353.14 (12.10)	1.80 (3.89)	362.27 (10.07)	364.07 (7.17)	0.93 (2.01)	356.57 (6.15)	357.50 (5.30)
Adj-Noun	112	39728	250	326.20 (12.07)	323.20 (6.98)	649.40 (18.72)	141.50 (12.66)	364.10 (5.67)	505.60 (9.13)	0.00 (0.00)	424.30 (0.58)	424.30 (0.58)	4.90 (4.34)	417.20 (2.04)	422.10 (2.96)
			500	320.10 (10.29)	301.00 (7.28)	621.10 (16.54)	89.80 (8.80)	371.50 (6.84)	461.30 (9.60)	0.20 (0.30)	421.00 (1.88)	421.20 (1.84)	1.70 (1.00)	411.20 (2.57)	412.90 (2.75)
			1000	300.30 (10.94)	266.50 (7.10)	566.80 (17.09)	62.60 (3.47)	368.30 (3.82)	430.90 (6.28)	0.10 (0.22)	413.70 (2.49)	413.80 (2.35)	0.40 (0.49)	401.20 (3.37)	401.60 (3.26)
			2000	287.50 (12.39)	244.30 (7.15)	531.80 (19.11)	43.50 (4.31)	356.20 (4.97)	399.70 (8.45)	0.00 (0.00)	400.80 (3.61)	400.80 (3.61)	0.10 (0.22)	387.40 (5.33)	387.50 (5.23)
			4000	270.40 (9.11)	222.00 (5.12)	492.40 (13.80)	28.90 (2.69)	343.10 (2.91)	372.00 (4.61)	0.00 (0.00)	384.40 (6.23)	384.40 (6.23)	0.00 (0.00)	373.80 (5.60)	373.80 (5.60)
			8000	255.30 (7.01)	199.50 (4.79)	454.80 (11.18)	18.40 (2.90)	328.90 (4.76)	347.30 (6.99)	0.00 (0.00)	366.30 (5.90)	366.30 (5.90)	0.00 (0.00)	356.30 (4.92)	356.30 (4.92)
fs-541-1	541	1028224	250	2268.00 (61.47)	2290.20 (28.69)	4558.20 (87.40)	204.50 (2.36)	2546.40 (2.90)	2750.90 (2.58)	0.00 (0.00)	2465.70 (0.48)	2465.70 (0.48)	6.70 (0.34)	2390.30 (1.05)	2397.00 (0.81)
			500	2186.40 (62.99)	2192.80 (37.17)	4379.20 (98.13)	154.10 (2.34)	2530.90 (3.32)	2685.00 (2.05)	0.00 (0.00)	2464.60 (1.06)	2464.60 (1.06)	3.70 (0.34)	2332.00 (1.00)	2335.10 (0.91)
			1000	2000.60 (70.20)	2002.20 (58.48)	4002.80 (127.69)	103.50 (1.89)	2421.40 (3.34)	2524.90 (3.20)	0.00 (0.00)	2462.50 (2.31)	2462.50 (2.31)	2.70 (0.34)	2329.00 (2.44)	2331.70 (2.42)
			2000	1805.00 (75.60)	1732.60 (78.13)	3537.60 (151.85)	79.30 (2.28)	2411.30 (4.31)	2490.60 (3.29)	0.00 (0.00)	2457.40 (3.07)	2457.40 (3.07)	1.00 (0.00)	2327.70 (2.60)	2328.70 (2.60)
			4000	1582.20 (77.67)	1428.80 (79.77)	3011.00 (156.92)	59.80 (2.32)	2359.20 (2.30)	2419.00 (2.53)	0.00 (0.00)	2445.10 (6.09)	2445.10 (6.09)	0.00 (0.00)	2325.90 (2.85)	2325.90 (2.85)
			8000	1390.40 (75.13)	1141.20 (53.79)	2531.60 (128.22)	44.20 (2.53)	2247.70 (4.23)	2291.90 (3.49)	0.00 (0.00)	2300.30 (2.47)	2396.20 (2.35)	0.00 (0.00)	2323.90 (4.80)	2322.60 (4.41)
eris1176	1176	28285256	250	4567.50 (18.97)	7566.70 (14.31)	12134.20 (28.28)	301.60 (0.95)	8493.50 (2.21)	8795.10 (2.31)	0.00 (0.00)	8686.50 (0.90)	8686.50 (0.90)	3.50 (0.60)	8560.90 (1.80)	8564.40 (1.73)
			500	4920.10 (26.68)	7341.40 (10.50)	12261.50 (30.77)	252.10 (1.07)	8466.90 (4.54)	8721.50 (3.63)	0.00 (0.00)	8680.00 (3.15)	8680.00 (3.15)	2.30 (0.48)	8552.10 (4.59)	8554.40 (4.61)
			1000	5185.10 (47.48)	7234.70 (8.27)	12419.80 (47.83)	202.10 (1.07)	8424.50 (6.37)	8626.60 (6.21)	0.00 (0.00)	8607.92 (15.40)	8607.92 (15.40)	0.90 (0.70)	8547.90 (7.94)	8548.80 (7.83)
			2000	5128.30 (69.11)	7154.10 (7.11)	12282.40 (70.90)	152.20 (1.04)	8432.90 (8.65)	8585.10 (8.93)	0.00 (0.00)	8437.00 (196.03)	8437.00 (196.03)	2.60 (1.38)	8428.80 (5.79)	8431.40 (5.93)
			4000	4980.20 (51.27)	7090.80 (6.22)	12071.00 (55.17)	102.20 (0.99)	8049.20 (4.87)	8151.40 (5.53)	0.00 (0.00)	8099.50 (87.87)	8099.50 (87.87)	2.60 (0.49)	k8086.30 (29.60)	8088.90 (29.42)
			8000	4975.75 (41.87)	7028.75 (9.55)	12004.50 (49.23)	39.00 (9.99)	7818.44 (12.62)	7857.44 (14.62)	0.00 (0.00)	7875.20 (142.93)	7875.20 (142.93)	0.00 (0.00)	7862.60 (20.81)	7862.60 (20.81)

Table 6: Real-world networks: average and standard deviation of type-I errors, type-II errors, and Hamming distance over 100 repetitions. For each row, the ranking of the Hamming distance is represented by the shade of the cells, such that the lightest cell marks the lowest Hamming distance and the darkest cell marks the highest Hamming distance.

Figure 8. In both, the table and the plots, the real network structures are ordered by their complexity (in n and irr). Again, the trends in these results are consistent to those in the previous experiments. For all the problems, BJP improves the average Hamming distance of the structures learned for all the cases when $\mathcal{N}_D < 4000$. The best improvements of BJP can be seen for the more irregular networks: Polbooks, Adj-noun, fs-541-1 and eris-1176. As can be seen in the plots of Figure 8, there are improvements of more than $4,0000$ edges corrected for the eris-1176 dataset over GSMN, improvements of more than 300 edges corrected over MPL, and improvements of more than 120 edges corrected against IB-score. This is coherent, since those are the most complex networks, and the best improvements are obtained when data is scarcer. Regarding the two types of errors (false positives and false negatives), it can be seen that they are similar to the case of the hub and scale-free networks.

When analyzing the runtimes of real-world networks, shown in Table 7, it can be seen that they are consistent to the cases of the hub and scale-free networks. An interesting difference can be seen for GSMN, which is the fastest algorithm for all the cases except for eris-1176. This is because for higher domain sizes and dense networks, GSMN tends to add many false positives in the grow phase, which requires a shrink phase performing unreliable tests with many variables. It produces numerous cascade errors, and it is the source of its expensive computational cost. Regarding the runtime of the BJP optimization, it can be seen that for almost all the cases the runtime over MPL and IB-score is improved.

In general, the results discussed confirm that BJP always outperforms the competitors when data are scarce. Also, the improvements are greater both in quality and runtime, for the more complex models. This confirms the hypothesis that the BJP can improve the quality of the learning process against competitors with better improvements when the structures are highly irregular.

Target structure	n	irr	$\mathcal{N}_D$	Runtime GSMN	MPL	IB-score	BJP
			250	0.49 (0.06)	5.30 (2.88)	5.01 (1.82)	1.79 (0.37)
			500	0.30 (0.06)	12.02 (7.71)	14.60 (10.37)	2.95 (0.45)
			1000	0.47 (0.07)	22.78 (6.94)	213.57 (122.01)	11.08 (4.79)
Karate	34	2044	2000	0.93 (0.13)	40.74 (5.80)	1220.47 (1068.23)	51.54 (16.91)
			4000	1.92 (0.32)	118.66 (20.14)	9557.99 (4920.78)	195.53 (75.77)
			8000	3.99 (0.47)	320.12 (41.05)	30963.00 (4700.50)	665.71 (138.21)
			250	2.09 (0.37)	3.94 (0.67)	1.97 (0.33)	1.37 (0.23)
			500	1.23 (0.24)	6.18 (0.83)	2.41 (0.50)	2.13 (0.23)
			1000	1.40 (0.54)	26.23 (3.57)	12.74 (1.61)	2.83 (0.33)
Ibm-32	32	2228	2000	3.17 (0.59)	72.41 (12.50)	31.25 (3.19)	6.46 (0.48)
			4000	7.49 (1.30)	184.44 (23.46)	48.06 (3.34)	15.46 (0.94)
			8000	14.32 (2.10)	512.50 (36.66)	119.43 (8.98)	35.31 (1.06)
			250	0.99 (0.09)	12.09 (0.80)	11.65 (2.44)	5.42 (0.54)
			500	1.13 (0.08)	28.83 (2.78)	28.50 (5.01)	11.34 (0.66)
			1000	2.13 (0.16)	83.15 (5.05)	83.29 (8.31)	29.48 (1.72)
Curtis-54	54	3140	2000	3.32 (0.50)	244.78 (14.42)	278.24 (58.14)	86.36 (10.00)
			4000	7.98 (0.49)	689.47 (40.53)	1466.95 (928.87)	240.60 (11.72)
			8000	17.11 (1.84)	2015.04 (85.23)	4665.29 (1282.94)	742.02 (36.45)
			250	1.02 (0.03)	13.14 (0.81)	9.67 (2.15)	5.70 (0.77)
			500	1.08 (0.08)	31.50 (3.00)	25.20 (3.75)	12.33 (1.43)
			1000	2.11 (0.14)	85.83 (5.21)	75.41 (9.51)	31.92 (3.61)
Will-57	57	4156	2000	3.55 (0.20)	232.11 (13.66)	245.99 (38.86)	87.39 (6.86)
			4000	7.90 (0.76)	672.76 (30.07)	886.78 (191.47)	274.25 (38.37)
			8000	16.62 (1.62)	2383.00 (111.73)	3077.88 (649.21)	787.46 (82.54)
			250	4.53 (1.07)	16.44 (0.79)	5.27 (0.66)	1.00 (0.06)
			500	5.00 (1.34)	28.08 (2.22)	8.61 (1.45)	1.72 (0.12)
			1000	8.74 (1.42)	62.63 (2.88)	32.11 (2.72)	2.35 (0.09)
Can-62	62	4104	2000	11.51 (1.71)	148.71 (3.11)	44.96 (0.89)	6.63 (0.47)
			4000	9.75 (2.29)	299.15 (4.75)	97.64 (3.86)	20.50 (0.32)
			8000	12.79 (2.33)	824.69 (52.26)	351.87 (13.05)	41.90 (0.99)
			250	1.19 (0.10)	24.03 (4.09)	12.46 (4.69)	7.12 (2.11)
			500	1.18 (0.08)	48.47 (8.56)	30.54 (9.04)	16.73 (3.64)
			1000	2.24 (0.20)	126.58 (19.41)	120.44 (20.92)	55.37 (5.81)
Dolphins	62	6480	2000	3.75 (0.26)	349.26 (37.06)	337.56 (40.51)	144.14 (18.98)
			4000	8.33 (1.27)	981.07 (101.01)	1092.66 (158.92)	386.28 (38.91)
			8000	22.83 (3.95)	3591.12 (237.77)	4171.51 (268.50)	1331.72 (69.26)
			250	1.95 (0.05)	135.11 (23.21)	4.33 (0.68)	4.12 (0.58)
			500	2.09 (0.16)	348.94 (29.62)	13.49 (2.50)	10.90 (2.17)
			1000	3.01 (0.36)	749.11 (35.61)	56.38 (8.58)	29.31 (3.82)
Polbooks	105	30374	2000	5.61 (0.70)	1455.56 (501.95)	170.71 (18.60)	85.34 (9.03)
			4000	12.09 (1.11)	2344.57 (1782.20)	648.50 (190.16)	246.44 (27.90)
			8000	26.11 (2.57)	4462.01 (3410.56)	3726.04 (1105.15)	971.55 (168.34)
			250	2.05 (0.05)	147.00 (24.21)	1.86 (0.45)	4.50 (2.13)
			500	2.17 (0.19)	317.36 (36.53)	6.56 (2.22)	9.38 (2.32)
			1000	2.88 (0.28)	521.98 (63.52)	27.30 (5.91)	27.00 (5.24)
Adj-Noun	112	39728	2000	5.63 (0.53)	814.73 (207.03)	108.96 (17.36)	80.54 (13.08)
			4000	12.62 (0.82)	1465.95 (275.37)	449.34 (118.64)	224.87 (31.94)
			8000	29.47 (2.01)	3443.87 (816.50)	2172.69 (749.05)	807.18 (243.59)
			250	42.01 (10.19)	557.00 (45.35)	457.24 (10.62)	335.18 (19.06)
			500	63.49 (14.98)	1285.26 (23.10)	658.34 (75.57)	553.15 (189.99)
			1000	131.19 (39.57)	2491.99 (139.66)	1515.28 (179.20)	888.04 (251.58)
Fs-541-1	541	1028224	2000	359.13 (132.18)	5614.95 (244.36)	2579.46 (237.00)	1061.59 (253.20)
			4000	733.69 (234.20)	8471.54 (405.78)	4503.30 (214.50)	2077.23 (619.18)
			8000	1163.95 (174.92)	15401.00 (1818.94)	7591.12 (309.70)	3122.31 (724.63)
			250	2009.40 (1963.49)	951.38 (149.21)	807.21 (18.75)	598.82 (34.04)
			500	2015.10 (1114.42)	2411.80 (43.34)	1162.23 (133.41)	988.23 (339.42)
			1000	5751.87 (5564.75)	6066.89 (1604.85)	3117.78 (589.74)	1586.53 (449.46)
eris1176	1176	28285256	2000	7426.46 (3495.60)	12242.50 (2106.78)	5532.42 (1199.91)	1896.58 (452.35)
			4000	10897.91 (4307.99)	20404.40 (5570.57)	9789.21 (2062.05)	3711.06 (1106.20)
			8000	16643.80 (5480.49)	19794.60 (9550.46)	13340.90 (610.16)	5578.12 (1294.57)

Table 7: Real networks: average and standard deviation of the learning runtime (in seconds) over 100 repetitions. For each row, the ranking of the runtimes is represented by the shade of the cells, such that the lightest cell marks the lowest runtime and the darkest cell marks the highest runtime.

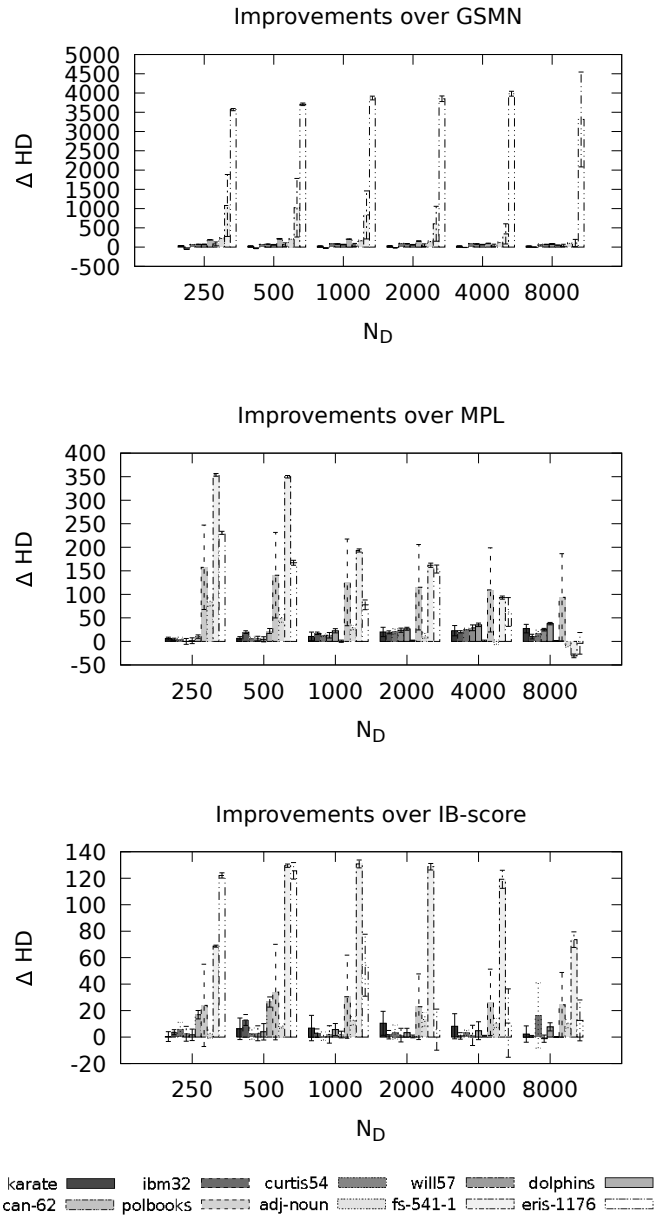


Figure 8: Hamming distance improvements of BJP over competitors for real-world networks. Δ HD denotes the improvement in Hamming distance over each competitor.

5. Conclusions

In this work we have introduced a novel scoring function for learning the structure of Markov networks. The BJP score computes the posterior probability of independence structures by considering the joint probability distribution of the collection of Markov blankets of the structures. The score computes the posterior of each Markov blanket progressively, using information from other blankets as evidence. The blanket posteriors of variables with fewer neighbors is computed first, and then this information is used as evidence for computing the posteriors for variables with bigger blankets. Thus, BJP can be useful to improve the data efficiency for problems with complex networks, where the topology exhibits irregularities, such as social and biological networks. In the experiments, BJP scoring proved that it can improve the sample complexity compared to the state-of-the-art competitors. The score is tested by using exhaustive search for low-dimensional problems and by using a heuristic hill-climbing mechanism for higher-dimensional problems. The results show that BJP produces more accurate structures than the state-of-the-art competitors when data are scarce.

We will guide our future work toward the design of more effective optimization methods, since the hill-climbing optimization has two inherent disadvantages: i) by only flipping one edge per step it scales slowly with the number of variables of the domain n, ii) it is prone to getting stuck in local optima. Moreover, we consider that the properties of BJP score have considerable potential for both further theoretical development, and applications.

6. Acknowledgements

This work was supported by Consejo Nacional de Investigaciones Científicas y Técnicas (CONICET) [PIP 2013 117], Universidad Nacional del Litoral (UNL) [CAI+D 2011 548] and Agencia Nacional de Promoción Científica y Tecnológica (ANPCyT) [PICT 2014 2627] and [PICT-2012-2731].

Appendices

A. Correctness of BJP

This appendix shows the proof of Theorem 1, concerned about the correctness of our method for computing the posterior of MN structures.

Theorem 1. *Let G be an undirected independence structure of a positive graph-isomorph distribution $P(X_V)$. The BJP scoring function of G is "correct" in the sense that the posterior probability that computes is equivalent to the posterior probability of a MN structure.*

Proof of Theorem 1. In the formulation of the BJP score, the joint distribution of the blankets of G is calculated by computing the probabilities of conditional independence and dependence assertions contained in the blanket of each variable of the domain. This proof follows by demonstrating that all the members and non-members of each blanket are unequivocally determined in (12), and therefore, that the joint posterior over these dependences and independences is equivalent to the posterior of the blankets.

From [28, Definition 2], the *Markov blanket closure* is a set of independence and dependence assertions that are formally proven to correctly determine a MN structure. This set is obtained by determining the blanket of each variable $X_i \in X_V$ with the following set of conditional independence and dependence assertions:

$$\Big\{\langle X_i \perp X_j | B^{X_i}\rangle \ : \ X_j \notin B^{X_i}\Big\} \quad \bigcup \quad \Big\{\langle X_i \not\perp X_j | B^{X_i} \setminus \{X_j\}\rangle \ : \ X_j \in B^{X_i}\Big\}.$$

Clearly, this is exactly the same set used by BJP in (12) to compute the posterior of the blanket of each variable of the domain. Since this set determines all members and non-members of a blanket, the posterior of this set of assertions is equivalent to the posterior of the blanket. Then, we demonstrate that such probabilities are correctly estimated by (13) and (14). We proceed by discussing their correctness separately for independence and dependence assertions.

i) **For independence assertions:** Equation (13) computes the probability of independence between a variable and a non-adjacent variable, conditioned on its blanket, given the previously computed blankets and the dataset D. In this equation, for the case when $i < k$, which indexes over the variables for which the blanket posterior is not already computed, the posterior of the independence assertion $\langle \psi_i \perp \psi_k | B^{\psi_i} \rangle$ must be computed from data. It is performed by using the Bayesian statistical test of [35], that has been proven to be statistically consistent, since its mean square error tends to 0 as the dataset size tends to infinity. For the case when $i > k$, which indexes over the variables for which the blanket posterior is already computed, the independence assertion is inferred as 1, since its independence is determined by the blanket of ψ_k, which is in the evidence $\left\{B^{\psi_j}\right\}_{j=0}^{i-1}$. By definition in (12), this case applies to all the variables $\psi_k \notin B^{\psi_i}$ (i.e., all the variables that are not connected to ψ_i). We argue the correctness for this inference by considering an intuitive equivalence commonly used by constraint-based approaches to perform independence tests that involve smaller number of variables [3, p. 980]. If two variables X_i and X_k are not neighbors in G, then by applying the local Markov property of (3) once for each, we have that $\langle X_i \perp X_k | B^{X_i} \rangle$ and $\langle X_i \perp X_k | B^{X_k} \rangle$ hold. Therefore, the inference made is correct.

i) **For dependence assertions:**

A similar argument can be given for the case of the dependence assertions. Equation (14) computes the probability of dependence between a variable and an adjacent variable conditioned on its remaining neighbors, given the previously computed blankets and the dataset D. Again, for the case when $i < k$, which indexes over the variables for which the blanket posterior is not already computed, the posterior of the dependence assertion must be computed from data. For the case when $i > k$, which indexes over the variables for which the blanket posterior is already computed, the dependence assertion is inferred as 1, since its dependence is deter-

mined by the blanket of ψ_k, which is again in the evidence $\left\{B^{\psi_j}\right\}_{j=0}^{i-1}$. By definition in (12), this case applies to all the variables $\psi_k \in B^{\psi_i}$ (i.e., all the variables that are connected to ψ_i). Clearly, if two variables X_i and X_k are neighbors in G, there are no sets separating them in the graph. Therefore, the dependence assertion inferred is true.

□

B. Impact of different orderings for blankets

This appendix shows two simulations that illustrate the convenience of the proposed ordering, that sorts the variables by their blanket sizes in ascending order. The first simulation illustrates how the sample complexity of statistical tests grows with the size of the conditioning set. The second simulation shows the sample complexity of the BJP score, computed for the underlying structure of data (i.e., the graph of Figure 1). For the graph of Figure 1, a MN random distribution has been generated, and then a synthetic dataset D has been sampled from the distribution with a Gibbs sampler. For more details about how we generated our synthetic data, see Section 4.

For the first simulation, the posterior probabilities of two independence assertions $t_1 = \langle X_1 \perp X_2 | X_0 \rangle$ and $t_2 = \langle X_1 \perp X_2 | X_0, X_3 \rangle$ were computed from D with the Bayesian statistical test. Both assertions are correct in the graph of Figure 1, and also must be present in the synthetic dataset generated. In the left plot of Figure 9 the trends of the log posterior probabilities of t_1 and t_2 are shown, computed from data for increasing dataset sizes $D = \{250, 500, 1000, 2000, 4000, 8000, 40000, 70000, 100000\}$. The log of the threshold 0.5 is drawn in a dashed line, to show the convergence of the probabilities. Although t_1 and t_2 are equivalent, t_2 has two variables in the conditioning set, and clearly requires higher amounts of data to converge to $\log(1) = 0$.

For the second simulation, we computed the BJP score using the following orderings of the variables:

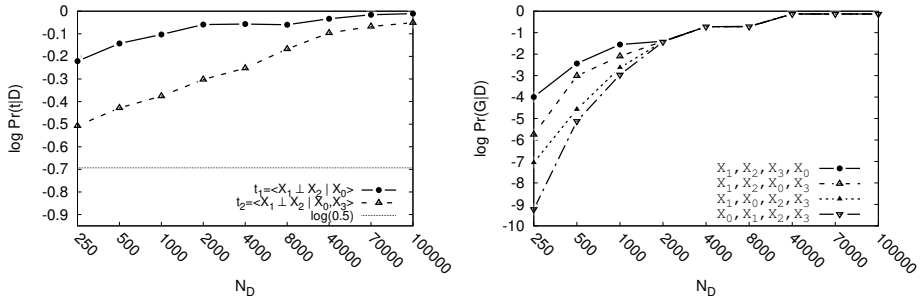


Figure 9: Simulation with data sampled from the hub structure of Figure 1. Left: the posterior of two equivalent independence assertions tested on data, with different conditioning sizes. Right: the BJP score computed for different arbitrary orderings.

(i) X_1, X_2, X_3, X_0 (optimal ordering, when sorting the blankets by their size in ascending order).

(ii) X_1, X_2, X_0, X_3 (sub-optimal).

(iii) X_1, X_0, X_2, X_3 (sub-optimal).

(iv) X_0, X_1, X_2, X_3 (worst ordering).

The right plot of Figure 9 shows the BJP score when using each of these orderings, for increasing datasets sizes $D = \{250, 500, 1000, 2000, 4000, 8000, 40000, 70000, 100000\}$. Clearly, the optimal ordering (X_1, X_2, X_3, X_0) shows the best sample complexity, and the ordering (X_0, X_1, X_2, X_3) shows the worst sample complexity. As it can be seen, the ordering used greatly affects the score when data is scarce ($D < 2000$). For dataset sizes greater than 1000 data points, the BJP score is the same for any order. It illustrates how the independence assumption between blankets affects the data efficiency. For small dataset sizes, an optimal ordering for computing the blankets joint posterior is expected to improve the sample complexity of those methods that assume independence between blankets.

C. Bayesian statistical test of conditional independence

This appendix describes briefly the Bayesian statistical test of conditional independence [35], and explains how to adapt it for discrete variables. The Bayesian test allows us to query a conditional independence between two random variables X_i and X_j, given a conditioning set X_Z, in a training dataset D. The statistical test works by comparing the posterior probability of two statistical models: the independent model M_{CI}, and the dependent model $M_{\neg CI}$.

The posterior probability of the independent model is computed from D as follows:

$$P(M_{CI} \mid D) = 1/\left(1 + \frac{1 - P(M_{CI})}{P(M_{CI})} \cdot \frac{P(D \mid M_{\neg CI})}{P(D \mid M_{CI})}\right), \tag{16}$$

where $P(M_{CI})$ denotes the a priori probability of the independent model, $P(D \mid M_{CI})$ is the data likelihood of the independent model, and $P(D \mid M_{\neg CI})$ is the data likelihood of the dependent model. The posterior probability of the dependent model is simply obtained by $P(M_{\neg CI} \mid D) = 1 - P(M_{CI} \mid D)$.

For computing the above formula, it is required to compute $P(D \mid M_{CI})$ and $P(D \mid M_{\neg CI})$. For discrete domains, the data likelihood of the independent model can be computed by the product of each of the "slices" of X_Z (that is, each possible complete assignment or configuration of X_Z), because it is assumed that the data is disjoint and independent for each slice. By denoting as K the number of slices, the data likelihood of the independent model is computed by

$$P(D \mid M_{CI}) = \prod_{k=1}^{K} P(D^k \mid M_{CI}^k) = \prod_{k=1}^{K} g_k, \tag{17}$$

where D^k is the subset of D corresponding to the slice k, and g_k is the likelihood in slice k, computed as

$$g_k = P(D^k \mid M_{CI}^k) = \left(\frac{\Gamma(\alpha)}{\Gamma(\alpha + M)} \prod_{i=1}^{I} \frac{\Gamma(\alpha_i + c_i)}{\Gamma(\alpha_i)}\right)\left(\frac{\Gamma(\beta)}{\Gamma(\beta + M)} \prod_{j=1}^{J} \frac{\Gamma(\beta_j + c_j)}{\Gamma(\beta_j)}\right). \tag{18}$$

This equation corresponds to the use of two independent Dirichlet priors. The α and β values are hyper-parameters, and c_i, c_j are the counts of variables X_i and

X_j in D^K. The hyper-parameters α and β are obtained by summing over all the hyper-parameters α_i, and β_j, respectively. The cardinalities of X_i and X_j are I and J respectively. The gamma function Γ is defined as $\Gamma(x) = \int_0^{+\infty} e^{-t} y^{x-1}\, dt$. When x is a non-negative integer, $\Gamma(x+1) = x!$.

For the dependent model, the data likelihood is more complex. It consists of a sum over all the possible values of independence and dependence for the slices of the conditioning set. As described in [39], it can be computed as

$$P(D \mid M_{\neg CI}) = \frac{\prod_{k=1}^{K} p_k g_k + q_k h_k - \prod_{k=1}^{K} p_k g_k}{P(M_{\neg CI})}, \tag{19}$$

where g_k is computed with (18), $p_k = P(M_I^k) = P(M_{CI}^{1/K})$ is the prior probability of the independent model in the slice k, $q_k = P(M_I^k) = 1 - p_k$ is the prior probability of the dependent model in the slice k, and h_k is the data likelihood of the model for the slice k, computed as

$$h_k = P(D^k \mid M_{\neg CI}^k) = \frac{\Gamma(\gamma)}{\Gamma(\gamma + M)} \prod_{i=1}^{I} \prod_{j=1}^{J} \frac{\Gamma(\gamma_{ij} + c_{ij})}{\Gamma(\gamma_{ij})}. \tag{20}$$

The values γ and γ_{ij} are hyper-parameters, and c_{ij} are the frequencies of variables X_i and X_j in D^K. The hyper-parameter γ is obtained by summing over all the hyper-parameters γ_{ij}.

The statistical test returns true when $P(M_{CI} \mid D) > P(M_{\neg CI} \mid D)$ and false otherwise. We recommend to implement the above formulas in the logarithmic space, for avoiding arithmetic underflow. In this work, our implementation uses the same hyper-parameter values as used in previous works [39, 28], which are: $\gamma_{ij} = 1, \alpha_i = 1, \beta_j = 1$ y $P(M_{CI}) = 0.5$).

References

[1] J. Pearl, Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference, Morgan Kaufmann Publishers, Inc., 1988.

[2] S. L. Lauritzen, Lectures in contingency tables, 2nd Edition, University of Aalborg Press, Aalborg, Denmark, 1982.

[3] D. Koller, N. Friedman, Probabilistic Graphical Models: Principles and Techniques, MIT Press, 2009.

[4] S. Li, Markov random field modeling in image analysis, Springer-Verlag New York, Inc., Secaucus, NJ, USA, 2001.

[5] W. Hwang, J. Kim, Markov network-based unified classifier for face recognition, IEEE Transactions on Image Processing 24 (11) (2015) 4263–4275.

[6] F. Peng, J. Lu, Y. Wang, R. Yi-Da Xu, C. Ma, J. Yang, N-dimensional markov random field prior for cold-start recommendation, Neurocomputing 191 (2016) 187–199.

[7] Y. Li, S. A. Pearl, S. A. Jackson, Gene networks in plant biology: approaches in reconstruction and analysis, Trends in plant science 20 (10) (2015) 664–675.

[8] M. Schmidt, K. Murphy, G. Fung, R. Rosales, Structure learning in random fields for heart motion abnormality detection, in: Computer Vision and Pattern Recognition, 2008. CVPR 2008. IEEE Conference on, 2008, pp. 1 –8. doi:10.1109/CVPR.2008.4587367.

[9] Y.-W. Wan, G. I. Allen, Y. Baker, E. Yang, P. Ravikumar, Z. Liu, M. Y.-W. Wan, Package xmrf.

[10] P. Larrañaga, J. Lozano, Estimation of Distribution Algorithms. A New Tool for Evolutionary Computation, Kluwer Pubs, 2002.

[11] S. Shakya, R. Santana, J. Lozano, A markovianity based optimisation algorithm, Genetic Programming and Evolvable Machines 13 (2) (2012) 159–195.

[12] D. Lowd, J. Davis, Improving markov network structure learning using decision trees, Journal of Machine Learning Research 15 (2014) 501–532.

[13] J. Van Haaren, J. Davis, Markov network structure learning: A randomized feature generation approach, in: Proceedings of the Twenty-Sixth AAAI Conference on Artificial Intelligence, 2012.

[14] J. Davis, P. Domingos, Bottom-up learning of Markov network structure, in: Proceedings of the 27th International Conference on Machine Learning (ICML-10), 2010, pp. 271–278.

[15] S. Lee, V. Ganapathi, D. Koller, Efficient structure learning of Markov networks using L1-regularization, in: NIPS, 2006.

[16] J. Van Haaren, J. Davis, M. Lappenschaar, A. Hommersom, Exploring disease interactions using markov networks, in: Workshops at the Twenty-Seventh AAAI Conference on Artificial Intelligence, 2013.

[17] G. Claeskens, E. Pircalabelu, L. Waldorp, Constructing graphical models via the focused information criterion, in: Modeling and Stochastic Learning for Forecasting in High Dimensions, Springer, 2015, pp. 55–78.

[18] H. Nyman, J. Pensar, T. Koski, J. Corander, Context-specific independence in graphical log-linear models, Computational Statistics (2014) 1–20.

[19] J. Pensar, H. Nyman, J. Niiranen, J. Corander, Marginal pseudo-likelihood learning of discrete markov network structures, Bayesian Analysis (2017) 1–21.

[20] P. Spirtes, C. Glymour, R. Scheines, Causation, Prediction, and Search, Adaptive Computation and Machine Learning Series, MIT Press, 2000.

[21] F. Bromberg, D. Margaritis, V. Honavar, Efficient Markov network structure discovery using independence tests, JAIR 35 (2009) 449–485.

[22] C. Aliferis, A. Statnikov, I. Tsamardinos, S. Mani, X. Koutsoukos, Local Causal and Markov Blanket Induction for Causal Discovery and Feature Selection for Classification Part I: Algorithms and Empirical Evaluation, JMLR 11 (2010) 171–234.

[23] F. Schlüter, A survey on independence-based Markov networks learning, Artificial Intelligence Review (2012) 1–25.

[24] S. Della Pietra, V. Della Pietra, J. Lafferty, Inducing Features of Random Fields, IEEE Trans. PAMI. 19 (4) (1997) 380–393.

[25] A. McCallum, Efficiently inducing features of conditional random fields, in: Proceedings of Uncertainty in Artificial Intelligence (UAI), 2003.

[26] V. Ganapathi, D. Vickrey, J. Duchi, D. Koller, Constrained Approximate Maximum Entropy Learning of Markov Random Fields, in: Uncertainty in Artificial Intelligence, 2008, pp. 196–203.

[27] F. Barahona, On the computational complexity of Ising spin glass models, Journal of Physics A: Mathematical and General 15 (10) (1982) 3241–3253.

[28] F. Schlüter, F. Bromberg, A. Edera, The IBMAP approach for Markov network structure learning, Annals of Mathematics and Artificial Intelligence (2014) 1–27.

[29] I. Csiszár, Z. Talata, Consistent estimation of the basic neighborhood of markov random fields, in: Information Theory, 2004. ISIT 2004. Proceedings. International Symposium on, IEEE, 2004, p. 170.

[30] M. Frydenberg, S. L. Lauritzen, Decomposition of maximum likelihood in mixed graphical interaction models, Biometrika (1989) 539–555.

[31] A. P. Dawid, S. L. Lauritzen, Hyper Markov laws in the statistical analysis of decomposable graphical models, The Annals of Statistics (1993) 1272–1317.

[32] J. Hammersley, P. Clifford, Markov fields on finite graphs and lattices.

[33] T. Cover, J. Thomas, Elements of information theory, Wiley-Interscience, New York, NY, USA, 1991.

[34] A. Agresti, Categorical Data Analysis, 2nd Edition, Wiley, 2002.

[35] D. Margaritis, Distribution-Free Learning of Bayesian Network Structure in Continuous Domains, in: Proceedings of AAAI, 2005.

[36] W. Cochran, Some methods of strengthening the common χ tests, Biometrics. (1954) 10:417451.

[37] J. E. Besag, Nearest-neighbour systems and the auto-logistic model for binary data, Journal of the Royal Statistical Society. Series B (Methodological) (1972) 75–83.

[38] I. Tsamardinos, C. Aliferis, A. Statnikov, Algorithms for large scale Markov blanket discovery, in: FLAIRS, 2003.

[39] D. Margaritis, F. Bromberg, Efficient Markov Network Discovery Using Particle Filter, Comp. Intel. 25 (4) (2009) 367–394.

[40] D. Margaritis, S. Thrun, Bayesian network induction via local neighborhoods, in: Proceedings of NIPS, 2000.

[41] T. Silva, L. Zhao, Machine Learning in Complex Networks, Springer International Publishing, 2016.
URL `https://books.google.com.ar/books?id=WdDurQEACAAJ`

[42] M. O. Albertson, The irregularity of a graph, Ars Combinatoria 46 (1997) 219–225.

[43] D. Lowd, A. Rooshenas, The libra toolkit for probabilistic models, arXiv preprint arXiv:1504.00110.

[44] A. Barabasi, E. Bonabeau, Scale-free netwroks, Scientific American.

[45] T. A. Davis, Y. Hu, The university of florida sparse matrix collection, ACM Transactions on Mathematical Software (TOMS) 38 (1) (2011) 1.