The Intersection-Validation Method for Evaluating Bayesian Network Structure Learning Without Ground Truth

Jussi Viinikka

Helsinki August 19, 2019

Master's thesis

UNIVERSITY OF HELSINKI
Department of Computer Science

HELSINGIN YLIOPISTO — HELSINGFORS UNIVERSITET — UNIVERSITY OF HELSINKI

Tiedekunta — Fakultet — Faculty	Laitos — Institution — Department
Faculty of Science	Department of Computer Science

Tekijä — Författare — Author
Jussi Viinikka

Työn nimi — Arbetets titel — Title
The Intersection-Validation Method for Evaluating Bayesian Network Structure Learning Without Ground Truth

Oppiaine — Läroämne — Subject
Computer Science

Työn — Arbetets art — Level	Aika — Datum — Month and year	Sivumäärä — Sidoantal — Number of pages
Master's thesis	August 19, 2019	65 pages + 14 appendices

Tiivistelmä — Referat — Abstract

Structure learning algorithms for Bayesian networks are typically evaluated by examining how accurately they recover the correct structure, given data sampled from a benchmark network. A popular metric for the evaluation is the *structural Hamming distance*. For real-world data there is no *ground truth* to compare the learned structures against. Thus, to use such data, one has been limited to evaluating the algorithms' predictive performance on separate test data or via cross-validation. The predictive performance, however, depends on the parameters of the network, for which some fixed values can be used or which can be marginalized over to obtain the posterior predictive distribution using some parameter prior. Predictive performance therefore has an intricate relationship to structural accuracy – the two do not always perfectly mirror each other.

We present *intersection-validation*, a method for evaluating structure learning without ground truth. The input to the method is a dataset and a set of compared algorithms. First, a partial structure, called *the agreement graph*, is constructed consisting of the features that the algorithms agree on given the dataset. Then, the algorithms are evaluated against the agreement graph on subsamples of the data, using a variant of the structural Hamming distance. To test the method's validity we define a set of algorithms that return a score maximizing structure using various scoring functions in combination with an exact search algorithm. Given data sampled from benchmark networks, we compare the results of the method to those obtained through direct evaluation against the ground truth structure. Specifically, we consider whether the rankings for the algorithms determined by the distances measured using the two methods conform with each other, and whether there is a strong positive correlation between the two distances.

We find that across the experiments the method gives a correct ranking for two algorithms (relative to each other) with an accuracy of approximately 0.9, including when the method is applied onto a set of only two algorithms. The Pearson correlations between the distances are fairly strong but vary to a great extent, depending on the benchmark network, the amount of data given as input to intersection-validation and the sample size at which the distances are measured. We also attempt to predict when the method produces accurate results from information available in situations where the method would be used in practice, namely, without knowledge of the ground truth. The results from these experiments indicate that although some predictors can be found they do not have the same strength in all instances of use of the method. Finally, to illustrate the uses for the method we apply it on a number of real-world datasets in order to study the effect of structure priors on learning.

ACM Computing Classification System (CCS):
Mathematics of computing → Bayesian networks
Computing methodologies → Learning in probabilistic graphical models

Avainsanat — Nyckelord — Keywords
Bayesian networks, structure learning

Säilytyspaikka — Förvaringsställe — Where deposited

Muita tietoja — övriga uppgifter — Additional information

Instructors: Ralf Eggeling and Mikko Koivisto

Contents

1 Introduction

A joint probability distribution over some variables of interest can be used to answer any probabilistic query on them. If the sets of possible values for the variables are discrete and finite – as they will be in this thesis – the distribution can naively be represented as a table with a row for each configuration of the variables and the corresponding probability (Figure 1). The size of the table grows exponentially with the number of variables, making storing it require large amounts of memory, answering queries on it slow and estimating the probabilities in it from data difficult.

Many observed distributions, however, approximately satisfy a number of conditional independence assertions over the variables (Koller and Friedman 2009; p. 8). If a factorization of a distribution is found such that it implies the set of conditional independencies in the distribution, the number of individual probabilities required to specify it might be dramatically decreased.

Graphical models – a class of statistical models – encode these independencies into a *structure*. The structure is a graph where the nodes and edges correspond to variables and statistical dependencies between them, respectively (Lauritzen 1996; p. 28). Bayesian network is a type of graphical model where the structure is a simple, *directed and acyclic graph* (DAG, Pearl 1985). The factorization that a Bayesian network implies can be read directly from its structure: it is the product of the local distributions of each variable conditional on its parents in the graph.

Each local distribution is specified by a *conditional probability table* (CPT) where the entries are the probabilities for the variable to take some value given a configuration of the conditioning variables. The form of the CPTs is determined by the structure and the number of possible values, *the arities*, of the variables. The structure together with the corresponding CPTs determine the distribution that the Bayesian network represents (Figure 2).

A	B	C	D	E	$P(A, B, C, D, E)$
0	0	0	0	0	0.04536
1	0	0	0	0	0.04374
		⋮			⋮
1	1	1	1	1	0.01176

$$\left\{ \begin{array}{c} E \perp\!\!\!\perp A|C \\ E \perp\!\!\!\perp B|C \\ E \perp\!\!\!\perp D|C \\ D \perp\!\!\!\perp A, E|\{B, C\} \\ C \perp\!\!\!\perp B|A \end{array} \right\}$$

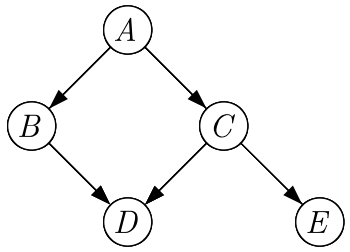
Figure 1: A joint distribution and the set of conditional independencies it satisfies.

A	$P(B\|A)$
0	0.8
1	0.1

$P(A)$
0.3

A	$P(C\|A)$
0	0.6
1	0.8

B	C	$P(D\|B,C)$
0	0	0.1
1	0	0.4
0	1	0.5
1	1	0.7



C	$P(E\|C)$
0	0.1
1	0.7

Figure 2: A Bayesian network consist of a structure (in the middle), and the corresponding CPTs. Here they represent the distribution in Figure 1 through the implied factorization $P(A)\,P(B|A)\,P(C|A)\,P(D|B,C)\,P(E|C)$.

Devising algorithms that learn Bayesian networks or other statistical models from data is a typical problem in machine learning research. For Bayesian networks the learning happens in two consecutive steps, for two different learning targets. First, a structure is learned. Second, a distribution is estimated conditional on the structure. For discrete valued Bayesian networks estimating the distribution given a structure reduces to learning the individual probabilities, *the parameters*, of the CPTs.

Solutions to the learning problem rely on methods to validate the algorithms and their output, i.e. to assess the algorithms' performance. Validating Bayesian network learning therefore concerns evaluating the performance of a given algorithm in learning a well fitting structure or distribution for some set of data. The focus of the analysis can either be on the absolute performance of a given algorithm or comparing the relative performances among a set of them. Here the term *algorithm* is used somewhat informally and can mean, depending on the learning target, either the procedure used to learn a structure or the combination of structure learning and parameter estimation. In practice, only the first step is solved by a procedure described as an algorithm; the latter step is fitting the parameters of a statistical model whose form has already been determined.

For performing accurate probabilistic inference with the model a well fitting distribution suffices. Learning such a distribution is possible, even if the structure is in some ways inaccurate. Specifically, any *complete* DAG (i.e., every node pair is connected by an edge) is able to host any distribution. Such a model would, however, tell nothing about the interrelations between the variables and inference on

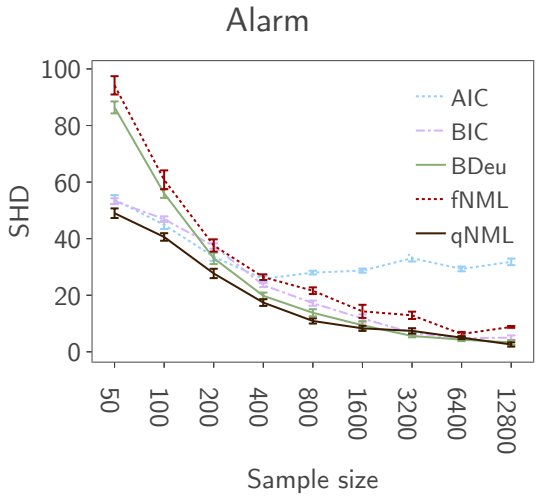


Figure 3: Evaluating structure recovery among 5 compared algorithms with data sampled from the benchmark network `Alarm`. Structural Hamming distance is used as evaluation metric.

the model would be slow. Though spurious edges then might not weaken a model's predictive performance, except the query running times, they by definition are a problem if the goal is to discover the correct structure. With limited training data, missing edges might even improve the predictive performance of a model, due to a given node's each parameter being estimated based on exponentially bigger subsets of the training data as edges to the node are removed. This phenomenon, called *data fragmentation*, might then give reason to favor sparser structures than the true one, even if it were known a priori.

Sometimes the interest is in *knowledge discovery*: understanding the domain underlying the observed distribution, rather than prediction. Rudimentary insights into the interrelations among the model variables can be gained by considering, for example, the pairwise correlations among the variables. However, a much more complete understanding can be formed by observing the direct and indirect dependencies encoded into the structure of a graphical model. With such interest one is therefore specifically concerned with learning a structure that closely approximates the "correct structure", as opposed to learning a well fitting distribution.

Algorithms are often evaluated by measuring how closely they recover a *ground truth model* given data sampled from it (Figure 3). We might for example take a fully specified Bayesian network as a starting point, sample data from it, learn a Bayesian network on the data using some evaluated algorithm, and finally measure

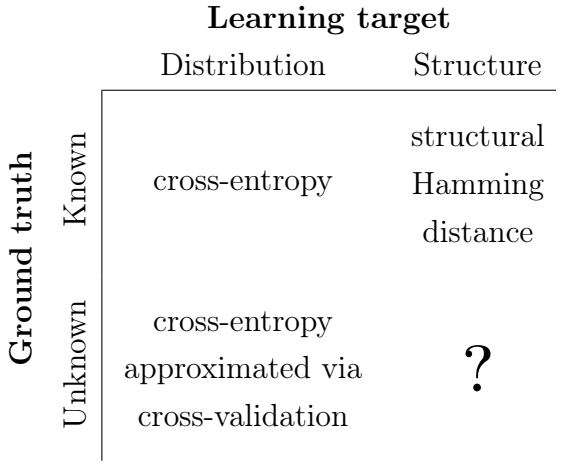


Figure 4: Approaches to evaluating Bayesian network learning.

the distance between the learned model and the ground truth using some metric. The structures of Bayesian networks are discrete objects for which variants of *the Hamming distance* are convenient metrics (e.g., Tsamardinos et al. 2006, Perrier et al. 2008). To evaluate how closely a learned distribution matches the ground truth distribution, we can calculate *cross-entropy* between the two. By learning on a sequence of datasets of increasing size we can also examine *the statistical efficiency* of an algorithm as an estimator of a learning target, that is how fast as a function of dataset size the algorithm's output approaches the correct structure or distribution. Due to the different convergence rates of different algorithms comparing a set of them in this way might reveal some algorithm to fare better than others starting from a certain dataset size, even if it performed worse on smaller sample sizes.

For real-world data there is no ground truth model to compare against. Even then, indirect comparison to the unknown ground truth distribution – assuming it exists – is possible by considering the predictive performance of a learned model on separate test data. The performance of an algorithm in terms of learning well fitting distributions can consequently be evaluated in such situations by *cross-validation*. In contrast, there does not seem to exist any established method for evaluating structure learning indirectly, without the ground truth model (Figure 4).

The thesis presents *intersection-validation*, a method for evaluating structure learning that does not rely on knowing the ground truth structure. The method first constructs a partial structure, *the agreement graph*, consisting of the features that a set of input algorithms agree on given a dataset. The agreement graph then func-

tions as a proxy for the ground truth structure against which the algorithms are evaluated on subsamples of the data. Ideally, the agreement graph consists of the features easiest to learn in the unknown true structure, while difficult enough to differentiate the algorithms on smaller datasets. The hypothesis is that the conclusions that can be drawn about the performance of the algorithms – such as the ranking of them for a given dataset size – are the same that one could draw if the ground truth was known.

The hypothesis is tested empirically through a number of experiments using data sampled from benchmark Bayesian networks and a set of structure discovery algorithms. The approach allows for evaluating the algorithms' performances using both the ground truth and intersection-validation methods and thus finally comparing the two. Strong positive correlation between the two evaluation methods is interpreted as supportive evidence for the hypothesis. Finally, experiments are conducted to analyze under what conditions intersection-validation seems to work. Specifically, the attributes of the agreement graph are considered as potential predictors of strong correlation between the intersection-validation and ground truth based results.

The thesis expands the experiments in a previously published conference paper on the intersection-validation method (Viinikka et al. 2018). Compared to the paper, both the number of input algorithms and benchmark networks are greater. Also, due to improved computational resources, the structure learning is conducted in less restricted search spaces, that is, the algorithms learn potentially more complex models. The types of experiments conducted to study various aspects of the results of the method include some additions as well. For example, the question whether the rankings the method yields for the input algorithms conform to those obtained through ground truth based evaluation is considered as a separate criterion for the method's validity. Additionally, another conference paper where the method was employed is summarized in relevant parts as an example use case for the method (Eggeling et al. 2019).

After the introduction, the thesis continues in section 2 with more formal treatment of concepts relevant to understanding intersection-validation and to interpreting the results obtained when using the method. These include Bayesian networks, learning them from data, and evaluating learning. Intersection-validation is presented in detail in section 3, after which its validity is tested through a number of experiments in section 4. The example use case for the method is presented in section 5, and the thesis ends with some concluding remarks in section 6.

2 Preliminaries

This section formalizes the concepts and sets the used notation related to the definition and properties of Bayesian networks touched upon in the introductory section. First some mathematical tools will be introduced to support the more analytical parts of the thesis, found later in this section and in section 3. Then learning Bayesian networks from data and evaluating learning is covered insofar as is necessary for understanding the experiments conducted and interpreting the results in section 4.

2.1 Mathematical notation and basic tools

Before treating the subject matter of the thesis, we specify some general notation that will be used in various places throughout the work. Also some basic mathematical tools will be presented. The notation specific to a certain topic will be given in conjunction with the topic itself, however.

For sets we clarify two notations. First, the set of positive integers up to (and including) some number k we will write as $[k]$, instead of some more explicit form such as $\{1, 2, \ldots, k\}$. Second, the notation $\{A_i\}_i$ we will use to denote the set of all A_i, where i is in some exhaustive index set. For example, for $i \in [n]$, $j \in [q_i]$ and $k \in [r_i]$ the notation $\{\theta_{ijk}\}_{ijk}$ should be understood as $\{\theta_{ijk} : i \in [n], j \in [q_i], k \in [r_i]\}$. In the example, for brevity, we also wrote ijk instead i, j, k, and we will use similar contractions when there is no risk of misunderstanding. The notation also allows for some of the subscripts to be fixed. Continuing the previous example, the notation $\{\theta_{ijk}\}_k$ thus equals $\{\theta_{ijk} : k \in [r_i]\}$.

For the indicator function we will use the Iverson bracket notation $[\cdot]$, which maps true propositions to the integer 1 and false ones to 0 (see, e.g., Graham et al. 1994; p. 24). For example $[x = x]$ maps to 1 and $[1 = 0]$ to 0.

The uppercase P will denote a probability measure, mapping events in the underlying probability space to their probabilities in the same. The lowercase p denotes a probability mass or density function, depending on the context. When we need to be explicit about the corresponding random variable we will write $p(X)$. The notation $p(x)$ is a shorthand for the value of the probability mass or density function when the corresponding variable is clear from context.

We are interested in the converging behaviour of various estimators, as the amount

of input data grows. Specifically, the mode of of convergence we will be referring to is the following (given for example in Gut 2009; p. 147).

Definition 1. A sequence of random variables $X_1, X_2, \ldots$ *converges in probability* to a random variable X, denoted as $X_N \xrightarrow{P} X$, if for all $\epsilon > 0$ it holds that

$$\lim_{N\to\infty} P\left(|X_N - X| > \epsilon\right) = 0.$$

With the above definition, we can state the weak law of large numbers (Gut 2009; p. 162).

Theorem 1 (The weak law of large numbers)**.** *For independent and identically distributed random variables $Y_1, Y_2, \ldots$ with finite first moment $\mathbb{E}Y$ it holds that*

$$\frac{Y_1 + Y_2 + \ldots Y_N}{N} \xrightarrow{P} \mathbb{E}Y.$$

The law of large numbers can be used to approximate expectations of various functions of random variables using the so called *Monte-Carlo methods* (see, e.g., Murphy 2013; p. 52). Let X then be a discrete random variable, taking values in some set S. The Monte Carlo approximation of the expectation of the random variable under some transformation f (assuming the expectation is finite) is given by

$$\mathbb{E}f\left(X\right) = \sum_{x\in S} p\left(x\right) f\left(x\right) \approx \frac{1}{N}\sum_{i=1}^{N} f\left(x_i\right),$$

where $X_i \sim p(X)$.

We will also use two variants of correlation for measuring the relationship between two variables. The first normalizes the covariance between the two variables to the interval $[-1, 1]$ (see, e.g., Murphy 2013; p. 45).

Definition 2. The *Pearson correlation coefficient* between random variables X and Y is defined as

$$\rho_{X,Y} := \frac{\mathrm{cov}\left(X, Y\right)}{\sigma_X \sigma_Y},$$

where $\sigma_X = \sqrt{\mathrm{var}\left(X\right)}$ is the standard deviation of X (similarly for Y), and where cov is the covariance.

The second variant we use when the interest is in the ranks of the variables (the highest value, the 2nd highest value and so on, in some dataset) and not on their absolute values. The definition only changes from the previous one by considering the corresponding rank variables r_X and r_Y.

Definition 3. The *Spearman correlation coefficient* between the rank variables r_X and r_Y is defined as

$$\rho_{r_X,r_Y} := \frac{\operatorname{cov}(r_X, r_Y)}{\sigma_{r_X}\sigma_{r_Y}}.$$

2.2 Bayesian networks

A Bayesian network models a joint distribution over a set of n random variables $\mathcal{X}$ through a structure and a set of local distributions. The structure is a DAG encoded by pair $G = (V, E)$, where V is a set of nodes and $E \subseteq (V \times V)$ is a set of edges. The set of nodes comprises the integers $[n]$. For brevity we denote edge $(u, v) \in E$ as uv. Each node $v \in V$ corresponds to a random variable $X_v \in \mathcal{X}$, taking values in S_v. The parents of node v in G are denoted by $G_v = \{u : uv \in E\}$ and the set of possible configurations for the parents by S_{G_v}. The number of parents for a node is called its *indegree*. The DAG factorizes the joint distribution as product of local distributions:

$$p(\mathcal{X}) = \prod_v p(X_v|(X_u)_{u\ \in G_v}). \tag{1}$$

The structure encodes the conditional independences between the variables in $\mathcal{X}$. Using the terms variable and node interchangeably, the *local Markov property* for Bayesian networks states that a variable is independent of its non-descendants given its parents. Non-descendants of a given node is the set of nodes left when the node itself, its parents and all the nodes to which there is a directed path from the node are removed from the full set of nodes. Intuitively the property states that given variable X's parents' values no additional information regarding some non-descendant variable Y is obtained by knowing the value of the variable X itself. The full set of conditional independences that the graph encodes, *the global Markov independences*, can be found through a graphical criterion called *d-separation* (Pearl 1988).

Though the factorization that a given structure implies is unique it is possible for different factorizations, and thus different structures, to imply the same set of conditional independences. Structures that imply the same set of conditional independences are called *Markov equivalent* and the set of them constitute a *Markov equivalence class*.

Theorem 2 (Verma and Pearl 1992)**.** *Two structures are equivalent if and only if they have the same* skeleton *and the same set of* v-structures.

Skeleton is the structure disregarding edge directions and a v-structure is any set of three nodes where two are parents of the third and the parents are not connected by an edge. Structures from the same equivalence class can host the same set of distributions. Given observational data such structures therefore are indistinguishable. If the edge directions are interpreted causally, however, some structures from the same Markov equivalence class capture the causal mechanism underlying an observed distribution better than others. In such sense Markov equivalent structures thus are not equivalent.

Completed partially directed acyclic graph (CPDAG) is a representation of a Markov equivalence class (Chickering 2002). A CPDAG has the same skeleton as any structure in the equivalence class. The direction of an edge is specified according to the equivalent structures if they all agree on it, otherwise it is left undirected (Figure 5). The encoding $G = (V, E)$ for DAGs extends to accommodate CPDAGs by including both directions in E for every undirected edge.

One way to obtain a CPDAG for an input DAG is to enumerate all equivalent structures and direct the edges accordingly. More efficient algorithms can be developed on the idea of identifying edges that are compelled to have the same direction as in the input DAG in all equivalent structures (Chickering 2002). As per theorem 2, edges in a v-structure are such *compelled edges*. Additionally, the acyclicity constraint in DAGs also compels the direction of certain edges.

Some additional notation needs to be introduced, to extend the discussion to the parameters of Bayesian networks. We will assume the codomains of the variables are finite and discrete, so we can write $|S_v| = |\{x_{vk}\}_k| = r_v$ and $|S_{G_v}| = |\{w_{vj}\}_j| = q_v$ for some $r_v \in \mathbb{N}$ and $q_v = \prod_{u \in G_v} r_u$. Without loss of generality we will further assume $S_v = [r_v]$. The full set of parameters required to specify the local distributions then is $\theta = \{\theta_{ijk}\}_{ijk}$, where $\theta_{ijk} = p(X_i = x_{ik} | (X_u)_{u \in G_i} = w_{ij})$ and which can be represented as a set of CPTs (Figure 2). The structure together with the parameters define the distribution $p(\mathcal{X})$ that the Bayesian network induces over the variables. Conversely, if equation (1) holds for a pair (G, p) on a set of variables $\mathcal{X}$ where G is a DAG and p is a distribution then it is a Bayesian network.

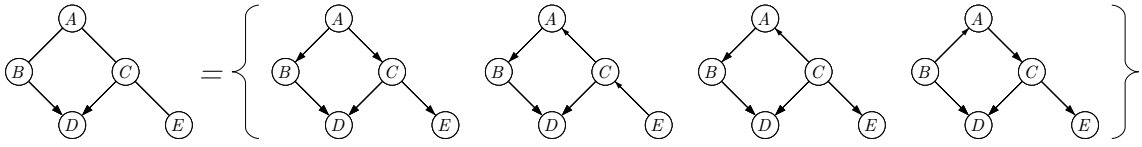


Figure 5: CPDAG, on the left, represents a Markov equivalence class, on the right

Sampling data from a Bayesian network proceeds by sampling a value for each variable in some topological order specified by the DAG. A topological order on $\mathcal{X}$ given by DAG G is any such order where for all $uv \in E$, X_u comes before X_v. The order ensures that the conditional distribution of a variable given its parents in G is fixed by the time it is to be sampled from by first assigning values to the parents. The sampling process described above is known as *forward sampling* (Jensen et al. 1995).

2.3 Training data

Learning Bayesian networks from data relies on a set of assumptions. For data the assumptions in the thesis are that the samples are *independent and identically distributed* (i.i.d.), and that the data is *complete*. The assumption of identical distribution in essence states, that the data contains information specific to the distribution that the learned model is to represent. The completeness assumption rules out the possibility of missing values for some variables in some samples, and specifically the existence of latent variables. A practical result from the independence and completeness assumptions is simplified mathematics, e.g., closed form expressions for parameter estimates. A repeated application of the forward sampling technique presented in the previous subsection yields a set of i.i.d. samples.

Throughout the thesis we assume the dataset D is a $N \times n$ matrix where N is the number of samples and n is their dimension (i.e., $|\mathcal{X}| = n$). We denote by D_i the i^{th} column of D corresponding to the random variable X_i and by D_V the columns corresponding to some subset $V \subset \mathcal{X}$ of the variables. The notation D_{i,G_i} is used for $D_{i \cup G_i}$, $D_{i,G_i=j}$ denotes the entries of column i on the rows on which G_i takes the j^{th} configuration and $D_{i=k,G_i=j}$ adds a further constraint for the column i to take its k^{th} value.

2.4 Parameter estimation

With data satisfying the assumptions discussed, learning a Bayesian network from it is a two-step process where first a structure is learned, after which the structure dependent CPT parameters are estimated. As the likelihood term $p(D|\theta, G)$ is relevant for both steps it is more convenient, though, to explain the steps in reverse order.

With the i.i.d. assumption the likelihood of data is the product of the likelihoods of the samples the data comprises. Assuming thus some structure G, the likelihood for a full dataset D given parameters θ_G consistent with G is

$$p\left(D|\theta_G\right) = \prod_i^n \prod_j^{q_i} \prod_k^{r_i} \theta_{ijk}^{N_{ijk}} \tag{2}$$

where $N_{ijk} = |D_{i=k,G_i=j}|$ is the number of samples where the i^{th} variable takes on the k^{th} value with its parents in the j^{th} configuration in the data. In this setting, the counts $\{N_{ijk}\}_k$ follow a multinomial distribution with parameters $\{\theta_{ijk}\}_k$. The likelihood thus is a product of $\sum_i q_i$ likelihood functions for counts following independent multinomial distributions. The set of counts $\{N_{ijk}\}_{ijk}$ is a *sufficient statistic* for θ_G. *The maximum likelihood estimates* (MLE) for the parameters are obtained by dividing the counts corresponding to a single multinomial distribution (i.e., CPT row) by their sum:

$$\hat{\theta}_{ijk}\left(D_{i,G_i}; G\right) = \frac{N_{ijk}}{\sum_{k'=1}^{r_i} N_{ijk'}}. \tag{3}$$

When D and G are clear from the context we will not explicitly refer to them. For parent configurations not appearing in the data – a situation occurring when not all values of a given variable participating in some parent set are observed – the parameters $\{\theta_{ijk}\}_k$ can be set arbitrarily as it will not change the likelihood.

The obvious drawback is that the MLE parameters overfit the data, and thus compromise the predictive performance of the model. To overcome this a prior $p\left(\theta\right)$ is applied to obtain the posterior density

$$p\left(\theta_G|D\right) \propto p\left(D|\theta_G\right) p\left(\theta_G\right) = \left[\prod_i^n \prod_j^{q_i} \prod_k^{r_i} \theta_{ijk}^{N_{ijk}}\right] p\left(\{\theta_{ijk}\}_{ijk}\right).$$

To simplify the prior, we will assume that the parameters for different variables are independent of each other (*global parameter independence*) and that the parameters for different parent configurations for a given variable are independent of each other (*local parameter independence*) (Spiegelhalter and Lauritzen 1990). With the two assumptions the posterior factorizes as

$$p\left(\theta_G|D\right) \propto \underbrace{\prod_i^n \left[\prod_j^{q_i} \prod_k^{r_i} \theta_{ijk}^{N_{ijk}} p\left(\{\theta_{ijk}\}_{jk}\right)\right]}_{\text{global independence}} = \prod_i^n \underbrace{\prod_j^{q_i} \left[\prod_k^{r_i} \theta_{ijk}^{N_{ijk}} p\left(\{\theta_{ijk}\}_{k}\right)\right]}_{\text{local independence}}.$$

Dirichlet distribution is a conjugate prior for the likelihood function of a multinomial distribution. The probability density function for a vector x in the n-dimensional

unit simplex in $\mathbb{R}^n$ (i.e. $x_i \geq 0$ for all i and $\sum_i x_i = 1$) following a Dirichlet distribution is defined as (Ng et al. 2011; p. 38)

$$\text{Dirichlet}\left(x;\alpha\right) := \frac{\prod_i \Gamma\left(\alpha_i\right)}{\Gamma\left(\sum_i \alpha_i\right)} \prod_i x_i^{\alpha_i - 1},$$

where Γ is a generalization of the factorial known as gamma function, given by the integral $\Gamma\left(t\right) = \int_0^\infty x^{t-1} e^{-x} dx$, $t > 0$ (Olver et al. 2010; p. 136).

Using a Dirichlet prior for each component likelihood function in the overall likelihood (equation 2), the posterior simplifies to:

$$\begin{aligned} p\left(\theta_G | D\right) &\propto \prod_{i=1}^{n} \prod_{j=1}^{q_i} \left[\prod_{k=1}^{r_i} \theta_{ijk}^{N_{ijk}} \prod_{k'=1}^{r_i} \theta^{\alpha_{ijk'} - 1} \right] \\ &= \prod_{i=1}^{n} \prod_{j=1}^{q_i} \left[\prod_{k=1}^{r_i} \theta_{ijk}^{N_{ijk} + \alpha_{ijk} - 1} \right] \\ &\propto \prod_{i=1}^{n} \prod_{j=1}^{q_i} \text{Dirichlet}\left(\left\{\theta_{ij}\right\}_k ; \left\{N_{ijk} + \alpha_{ijk}\right\}_k\right). \end{aligned} \tag{4}$$

The parameters could now be set to maximize equation 4 to obtain their *maximum aposteriori* (MAP) estimates. Alternatively, as is done in the experiments in this thesis, they can be set to their expected value under the posterior Dirichlet distribution, i.e. the *mean posterior estimate* (Ng et al. 2011; p. 39):

$$\theta_{ijk}^{\text{MP}} = \frac{N_{ijk} + \alpha_{ijk}}{\sum_{k'=1}^{r_i} \left(N_{ijk'} + \alpha_{ijk'}\right)}.$$

The distribution induced by a Bayesian network with these parameters coincides with the predictive distribution $p\left(d|D, G, \alpha\right)$ where α is the set of hyperparameters (Koller and Friedman 2009; p. 748).

2.5 Structure learning

Structure learning algorithms, principally, are either *constraint-based* or *score-based*, though *hybrid* algorithms also exists combining ideas from the first two. In constraint-based approaches (e.g., Pearl and Verma 1991, Verma and Pearl 1992, Meek 1995, Spirtes et al. 2000, Cheng et al. 2002) the data is analyzed using frequentist hypothesis tests to discover conditional independencies between the variables. Once the independencies are found they are mapped to a structure that encodes them. In score-based approaches a hypothesis space of possible structures is searched to

find high scoring ones, given a function that scores each candidate structure based on its goodness of fit to the data. Tsamardinos et al. (2006) present an example of a hybrid method, which first finds a skeleton using constraint-based learning and then orients the edges by running a score-based search conditional on the skeleton

Score-based structure learning can be seen as solving an optimization or model selection problem, where the optimality criterion is determined by the used score. Depending on the search algorithm used, either a locally (e.g., Cooper and Herskovits 1992, Heckerman et al. 1995, Teyssier and Koller 2005) or globally optimal structure is found (e.g., Koivisto and Sood 2004, Cussens 2011, Yuan and Malone 2013). The former are called *heuristic* and the latter *exact* search algorithms.

Assuming the hypothesis space is the space of possible DAGs, the size of it grows super-exponentially with the number of variables in the joint distribution. This is easy to see, as there are $2^{\binom{n}{2}}$ undirected graphs on n nodes and each of them has at least one causal ordering corresponding to a DAG. Consequently, the score-based approach to learning Bayesian network structures is NP-hard, with both running time and memory usage being exponential in the amount of variables in the worst case (Chickering 1996). Beyond the worst-case and best-case bounds the analytic time complexities of the state-of-the-art exact structure discovery algorithms are not well understood (Malone et al. 2014), but in practice they limit the application of the algorithms to problem instances where the number of variables is in the tens (Beek and Hoffmann 2015). Using heuristic algorithms much larger problems can be solved, but these algorithms are not able to provide any guarantee on the quality of the structures they discover (Malone et al. 2014). Often, to make the problem more tractable, the hypothesis space is limited by considering only structures whose *maximum indegree* is below some limit.

Using an exact search algorithm with a scoring function S given a dataset D, the optimization problem of score-based structure learning can be expressed as finding some structure

$$G^* \in \arg\max_G S(G, D). \tag{5}$$

The function maximizing structure may not be unique. For example, some scoring functions satisfy the property of *score equivalence*, by which structures from the same Markov equivalence class are given equal scores. The property is desirable, as given observational data the correct structure is identifiable only up to its equivalence class, giving no reason to prefer one equivalent structure over another. Also, in some cases, structures from different equivalence classes might be score maximizing

ones.

The globally optimal structure of equation 5 can be found using a search algorithm that casts the problem as, for example, dynamic programming, finding a shortest path or integer linear programming. As the practical differences among the algorithms are limited to running times and memory requirements, and as neither is a relevant parameter to the method presented in the thesis, the search algorithms will not, however, be presented in any detail. Next we will thus concentrate on the component that directly influences the output of the used structure learning algorithms – the scoring function.

2.6 Scoring functions

Decomposability as a property of scoring functions states that the score assigned to a structure is the sum of the scores of each local structure, that is, node and its parents (Heckerman et al. 1995). Denoting the local structure of node v as $G_{l(v)} = (\{v, G_v\}, \{(u, v) : u \in G_v\})$ a decomposable score takes the form

$$S(G, D) = \sum_v S(G_{l(v)}, D_{v,G_v}).$$

Decomposability is an important property as it simplifies the search through the hypothesis space speeding it up significantly. All of the considered scores therefore are decomposable.

2.6.1 Bayesian Dirichlet family of scores

A family of scoring functions can be derived by adopting a Bayesian approach to structure learning. The posterior probability of a structure given data is given by the Bayes' theorem as

$$p(G|D) = \frac{p(D|G)p(G)}{p(D)} \propto p(D|G)p(G).$$

The denominator is identical for all structures so it can be ignored when comparing them. Taking the logarithm of the last expression yields the *Bayesian score* (Heckerman et al. 1995):

$$S^{\mathrm{B}}(G, D) := \log p(D|G) + \log p(G) \tag{6}$$

The $P(G)$ term specifies the prior probability given for a structure and can be used to, for example, explicitly favor sparse structures. The $P(D|G)$ term is the *marginal*

likelihood as it is calculated by integrating over the set of all possible parameters Θ_G consistent with G:

$$p\left(D|G\right) = \int_{\Theta_G} p\left(D|\theta_G, G\right) p\left(\theta_G|G\right) d\theta_G.$$

With the assumptions of globally and locally independent parameters with Dirichlet priors parameterized by the set $\alpha = \{\alpha_{ijk}\}_{ijk}$, the integral decomposes as (Buntine 1991, Heckerman et al. 1995)

$$p\left(D|G,\alpha\right) = \prod_{i=1}^{n}\prod_{j=1}^{q_i} p\left(D_{i,G_i=j};\alpha\right) = \prod_{i=1}^{n}\prod_{j=1}^{q_i}\int p\left(D_{i,G_i=j}|\theta_{ij}\right) p\left(\theta_{ij};\alpha\right) d\theta_{ij}.$$

The integral has a closed form solution the logarithm of which yields the *Bayesian Dirichlet score* (BD):

$$S^{\mathrm{BD}}\left(G, D;\alpha\right) := \sum_{i=1}^{n}\log\prod_{j=1}^{q_i}\left[\frac{\Gamma\left(\alpha_{ij}\right)}{\Gamma\left(N_{ij}+\alpha_{ij}\right)}\prod_{k=1}^{r_i}\frac{\Gamma\left(N_{ijk}+\alpha_{ijk}\right)}{\Gamma\left(\alpha_{ijk}\right)}\right], \tag{7}$$

where $N_{ij} = \sum_k N_{ijk}$ and $\alpha_{ij} = \sum_k \alpha_{ijk}$ (Heckerman et al. 1995). The BD score depends on the Dirichlet hyperparameters. It thus defines a class of scoring functions within the Bayesian framework.

To use a score from the BD family one needs to specify the Dirichlet hyperparameters for each structure in the hypothesis space. Clearly, this cannot be done arbitrarily due to the sheer number of parameters for most problem instances of realistic size. To set the parameters systematically a principled approach is to first specify a set of desirable properties for a scoring function and then set the parameters so as to satisfy those properties, if possible. We have argued for decomposability and score equivalence to be such properties.

To achieve decomposability it suffices to set the hyperparameters so, that the prior probability distribution for the parameters of a variable given a set of parents is the same in every structure where the set of parents is the same. The distribution should thus only depend on local structure, that is, it should satisfy *parameter modularity* (Heckerman et al. 1995). This is achieved simply by specifying the set $\{\alpha_{ijk}\}_{jk}$ identically for every structure where variable i has some fixed set of parents.

In order for a BD score to satisfy score equivalence we first need a prior probability distribution p over the entire probability space. Score equivalence is then achieved by selecting an *equivalent sample size* (ESS) α, quantifying how many samples our prior knowledge is equivalent to, and setting

$$\alpha_{ijk} = \alpha \cdot p\left(x_{ik}, w_{ij}\right),$$

where x_{ik} is the k^{th} value of variable i and w_{ij} is the j^{th} configuration of the variable's parents. Setting the parameters in this fashion clearly satisfies parameter modularity, so the resulting *Bayesian Dirichlet equivalent* (BDe) score is also decomposable (Heckerman et al. 1995).

When there is no reason to use an informed prior p in the above procedure, an uninformative uniform prior should be used instead. The hyperparameters are then set as

$$\alpha_{ijk} = \frac{\alpha}{q_i \cdot r_i} \tag{8}$$

where q_i is the number of parent configurations and r_i is the number of possible values or arity of variable i. Finally, the score defined by the logarithm of equation 7 and the hyperparameters specified in equation 8 can be seen to depend on only one parameter, the equivalent sample size, and is called the *Bayesian Dirichlet equivalent uniform* (BDeu, Buntine 1991).

2.6.2 Penalized log-likelihoods

One might think of defining the scoring function so, that the score maximizing structure is the one that maximizes the likelihood of observing the data. This is achieved by mapping an input structure to the likelihood of the data under the maximum likelihood parameters $\hat{\theta}_G$ (equation 3) consistent with the structure. For convenience of computation the logarithmic likelihood is used to arrive at the *log-likelihood score*:

$$S^{\text{LL}}(G, D) := \log p\left(D|\hat{\theta}_G, G\right) = \sum_{i=1}^{n}\sum_{j=1}^{q_i}\sum_{k=1}^{r_i} N_{ijk} \log \hat{\theta}_{ijk}$$

Clearly, the log-likelihood score is decomposable. The problem with the score is explained by two observations. First, for any edge in a structure of a Bayesian network it is trivial to set the parameters so that the dependencies implied by the edge do not appear in the distribution induced by the Bayesian network. In other words, a complete DAG as a structure for a Bayesian network can host any set of independencies and thus any distribution by setting the parameters accordingly. This implies that the likelihood score can never be decreased by introducing additional edges to the structure. Thus for any pair of DAGs $G = (V, E)$ and $G' = (V, E')$ where $E \subset E'$ it holds that

$$S^{\text{LL}}(G, D) \leq S^{\text{LL}}(G', D). \tag{9}$$

Second, given a finite sample $D = \{d_i\}_i$ of N observations from a distribution p satisfying some conditional independence, the empirical distribution $p_N(A) = \frac{1}{N}\sum_i [d_i \in A]$ almost never satisfies the independence exactly. The implication of this is that the inequality 9 is almost always strict, and thus structure learning based on the likelihood score is almost guaranteed to output complete DAGs. In essence, the score overfits the data.

One solution to the problem is to introduce various model complexity penalizing terms to the likelihood score. The resulting *penalized log-likelihood scores* take the form

$$S(G, D) = S^{\mathrm{LL}}(G, D) - \Delta(G, D),$$

where $\Delta(G, D)$ is the function specific penalty term. Intuitively, after a certain level of complexity, the penalty term should become dominant and the overall score start to decrease even if the likelihood term continues increasing with more edges added. If also the penalty term decomposes, these functions are called *decomposable penalized log-likelihood* scores. A simple implementation of the idea is to directly penalize by the amount of free parameters of the model, that is, to use the *Akaike information criterion* (Akaike 1974) whose local penalty term is thus given by

$$\Delta_i^{\mathrm{AIC}}\left(G_{l(i)}\right) = q_i\left(r_i - 1\right)).$$

The previously introduced BD-score also yields a specific penalized log-likelihood score. Asymptotically it evaluates to (Koller and Friedman 2009; p. 801)

$$\lim_{N\to\infty} S^{\mathrm{BD}}(G, D) = S^{\mathrm{BD}}(G, D) - \frac{\log N}{2}\sum_{i=1}^{n} q_i\left(r_i - 1\right) + O(1).$$

Unlike the other considered BD-score variants, the asymptotic approximation thus does not depend on the hyperparameters and, ignoring the constant term, defines a penalized log-likelihood score. The score is called *the Bayesian information criterion* (BIC, Schwarz 1978), whose local penalty term for discrete valued Bayesian networks is given by

$$\Delta_i^{\mathrm{BIC}}\left(G_{l(i)}, D\right) = \frac{\log N}{2} q_i\left(r_i - 1\right).$$

The BIC score, clearly, is decomposable.

Also the BDeu-score has an interpretation as a penalized log-likelihood score, where the local penalty is given by (Silander et al. 2008)

$$\Delta_i^{\mathrm{BDeu}}\left(G_{l(i)}, D\right) = \sum_{j}^{q_i}\sum_{k}^{r_i} \log \frac{p\left(D_{ijk}|D_{ij}\right)}{p\left(D_{ijk}|D_{ij}, \alpha_{ij}\right)}.$$

2.6.3 Scores based on the normalized maximum likelihood

As final alternatives, we consider two scoring functions derived from information theory. The *factorized normalized maximum likelihood* (fNML) score is based on the *normalized maximum likelihood* (NML) distribution given by

$$p_{\mathrm{NML}}(D|G) := \frac{p\left(D|\hat{\theta}(D;G)\right)}{\sum_{D'} p\left(D'|\hat{\theta}(D';G)\right)}$$

which has a *minimum description length* justification (Rissanen 1978, Grünwald 2007). The sum in the denominator is over all $N \times n$ data matrices and the log of it is called *the parametric complexity* or *regret*. NML is score equivalent but not decomposable, and the sum makes the score impractical. To achieve decomposability, fNML uses the NML distribution for the column partitions of D to obtain the score as a sum of 1-dimensional p^{NML}-codes (Silander et al. 2008):

$$\begin{aligned} S^{\mathrm{fNML}}(G,D) &:= \sum_{i=1}^{n}\sum_{j=1}^{q_i} \log p^{\mathrm{NML}}(D_{i,G_i=j}|G) \\ &= \sum_{i=1}^{n}\sum_{j=1}^{q_i} \log \frac{p\left(D_{i,G_{i=j}}|\hat{\theta}\left(D_{i,G_{i=j}};G\right)\right)}{\sum_{D'_i} p\left(D'_i|\hat{\theta}(D'_i;G)\right)}. \end{aligned}$$

Now the sum in the denominator is over the multinomial vectors $D'_i \in [r_i]^{|D_{i,G_i=j}|}$, that is, vectors of length $|D_{i,G_i=j}|$ with each component taking values in $[r_i]$. The sum in the case of a multinomial vector is given by

$$\mathcal{C}_n^r = \sum_{k_1+k_2+\ldots k_r=n} \frac{n!}{k_1!k_2!\ldots k_r!} \prod_{j=1}^{r} \left(\frac{k_j}{n}\right)^{k_j}$$

where n denotes the length of the vector, r is the number of possible values for the multinomial variable, and the sum is over the possible counts for the different values in the vector. There is a linear time algorithm for computing $\mathcal{C}_n^r$ (Kontkanen and Myllymäki 2007) and accurate approximations for large n exist (Kontkanen et al. 2003), making the score usable in practice. The score can be viewed as an alternative definition for the marginal likelihood or a penalized log-likelihood score where the local penalty is given by

$$\Delta_i^{\mathrm{fNML}}(G,D) = \sum_{j=1}^{q_i} \log \mathcal{C}_{N_{ij}}^{r_i}.$$

The *quotient normalized maximum likelihood* (qNML) is closely related to fNML but it is score equivalent (Silander et al. 2018). The score, defined as

$$S^{\mathrm{qNML}}(G, D) := \sum_{i=1}^{n} \log \frac{p^{\mathrm{NML}}(D_{i,G_i}|G)}{p^{\mathrm{NML}}(D_{G_i}|G)},$$

does not partition the data columns corresponding to each variable by the configurations of the parent variables as in fNML. Instead, the relevant variables in some set S are collapsed together to form a single variable with $\prod_{X_i \in S} r_i$ different values, which is then modeled with a 1-dimensional P^{NML}-code. As a penalized log-likelihood score its local penalty term is given by

$$\Delta_i^{\mathrm{qNML}}(G, D) = \log \frac{\mathcal{C}_N^{r_i q_i}}{\mathcal{C}_N^{q_i}}.$$

2.7 Evaluating learning

The approaches to evaluating Bayesian network learning, as they were summarized in Figure 4, can be viewed along two dimensions: whether they are applied on the structure or the distribution and whether they require knowing the model underlying the data or not. From the approaches considered, *structural Hamming distance* is a metric (Tsamardinos et al. 2006), *cross-entropy* is a divergence (Heckerman et al. 1995, Kullback and Leibler 1951), and *cross-validation* is a general technique for evaluating how well a statistical analysis generalizes to unseen data (Russell and Norvig 2009; p. 708).

The intuitive notion of a distance, or a metric, has a well defined mathematical meaning (see, e.g., Arkhangel'skiĭ and Fedorchuk 1990; p. 20).

Definition 4. A non-negative function $d : X \times X \to \mathbb{R}$ is considered a *metric* on X if for all $x, y, z \in X$ it satisfies the following conditions:

1. $d(x, y) = 0 \Leftrightarrow x = y$ (identity of indiscernibles),
2. $d(x, y) = d(y, x)$ (symmetry axiom),
3. $d(x, z) \leq d(x, y) + d(y, z)$ (triangle axiom).

By relaxing one or more of the above conditions functions can be defined, which, though not metrics in the strict sense, can still be useful in analyzing the extent of similarity between two objects. In this thesis the interest lies in comparing a set of

learned Bayesian networks in terms of how close they are to some true model or a proxy of it. The question does not depend on the condition of symmetry for the applied evaluation function, as by convention the learned model can always take the place of the first (or the second) operand. Also the question does not require comparing the learned models directly against each other, so the triangle axiom need not be required.

Knowing the ground truth Bayesian network and the possibility to draw infinite amounts of data from it make the conceptual framework of evaluating the statistical efficiency of learning algorithms straightforward. Given a structure learning algorithm and some parameter estimates the evaluation of them for some fixed dataset size consists of (1) drawing a number of datasets of the chosen size, (2) learning a Bayesian network on each of the datasets, (3) measuring the distance between the learned models and the ground truth, and (4) reporting the results. Analyzing the statistical efficiency of learning then consists of repeating the above process for varying data sizes. In the first step a number of datasets are drawn to offset the effect of sampling bias; in the last step the mean distance and possible other summary statistics, such as *standard errors of the means* are reported.

2.7.1 Cross-entropy

A Bayesian network defined over a set of variables induces a distribution over the possible configurations the variables can assume. Let $\mathcal{X}$ be the set of variables and $S_{\mathcal{X}}$ the set of possible configurations over them. How closely the induced distribution q matches some target distribution p defined over the same set of variables can be measured with the *Kullback–Leibler divergence* (KL-divergence, Kullback and Leibler 1951) defined as

$$d_{\mathrm{KL}}(p \parallel q) := \sum_{i \in S_{\mathcal{X}}} p(i) \log \frac{p(i)}{q(i)}. \tag{10}$$

KL-divergence is not a proper distance metric as generally it does not satisfy the symmetry and triangle axioms, but it is commonly used to analyze the similarity between distributions.

KL-divergence decomposes as

$$d_{\mathrm{KL}}(p \parallel q) = \sum_{i \in S_{\mathcal{X}}} p(i) \log p(i) - \sum_{i \in S_{\mathcal{X}}} p(i) \log q(i) = -H(p) + H(p, q)$$

where $H(p)$ is the *entropy* of p and $H(p, q)$ is the *cross-entropy* between p and q. For a given target distribution p the entropy term is constant so to choose the closest

approximation of it in some set of candidate distributions it suffices to analyze the cross-entropy terms.

In most situations explicit summation over $S_{\mathcal{X}}$ is not feasible as the number of different configurations is exponential in the number of variables. If also p is induced by a Bayesian network the summation can exploit the independencies encoded into the network structures to reduce the amount of summed terms. In the case where the structures and thus the set of parameters are identical (with possibly differing values) cross-entropy takes the form (Heckerman et al. 1995)

$$H\left(p,q\right)=-\sum_{i=1}^{n}\sum_{j=1}^{q_i}\sum_{k=1}^{r_i}p\left(x_{ik},w_{ij}\right)\log q\left(x_{ik}|w_{ij}\right), \tag{11}$$

which can be evaluated efficiently depending on the complexity of the structure. If the structures are different, the above equation is not applicable. But, as noted, any complete DAG can host any distribution on the set of variables it is defined on. It is therefore possible to embed both of the induced distributions on to the same complete DAG in order to apply equation 11. A complete DAG, however, does not encode any independencies so the embedding does not provide computational benefits compared to evaluating equation 10 explicitly. An optimal solution can be derived by embedding the two distributions onto a consensus DAG that encodes the maximum amount of the independencies of the two DAGs without introducing spurious ones. Finding such a DAG, however, is NP-hard, although heuristic algorithms exist (Peña 2011).

A practical alternative to exact calculation of cross-entropy is to use Monte Carlo approximation. The approximation is of the form

$$H(p,q)\approx-\frac{1}{N}\sum_{j=1}^{N}\log q(x_j)$$

where x_j are random samples from p. Here, to satisfy the assumption of finite first moment required by the law of large numbers (Theorem 1, p. 7), we will assume that the probabilities $q(x)$ are positive for all x. In a setting where the target distribution is known this asymptotic result is not just a theoretical curiosity, as arbitrarily large numbers of samples can be drawn from the distribution.

2.7.2 Structural Hamming distance

The structure of a Bayesian network as noted is a graph. To quantify the distance between two graphs defined on the same set of nodes (e.g., a learned one and the

ground truth), an intuitive measure is to calculate the number of editing operations required to transform one into the other, or the so called *edit distance*. A less ambiguous alternative formulation is to count the number of node pairs that differ with regard to the edge type between them in the operand graphs. Among the graph types considered in the thesis, a node pair (u, v) can either be connected by an edge from u to v, from v to u, have both edges in which case the node pair is said to be connected by a bidirected edge, or not be connected by an edge. If two graphs differ with respect to $d \in \mathbb{N}$ node pairs the distance between them would then be d. This type of a discrete distance measure is called a Hamming distance and was originally formulated on strings (Hamming 1950).

Hamming distance between two different DAGs in the same Markov equivalence class is non-zero. As noted, such structures encode the same conditional independencies and thus are able to host the same set of distributions – they are statistically indistinguishable given observational data. A desirable property for a distance measure on structures without causal interpretation therefore is that the distance between equivalent structures is zero. Structural Hamming distance (SHD) is a metric defined on CPDAGs (Figure 5), so clearly it satisfies the property (Tsamardinos et al. 2006).

Using SHD as a metric, any difference between any node pair in the operand CPDAGs yields an additional distance of one. There can be justified reasons to regard some specific type of difference more serious than another, in which case the metric can be adjusted to yield a greater additional distance from the more serious mismatch. For example, it can be argued that a difference in the graph skeleton is more consequential than a difference in edge orientation (Perrier et al. 2008). In this thesis such adjustments however are not done.

The above formulation is still ambiguous in that it does not specify whether the node pairs considered are ordered or unordered. To have a convenient formalization of the Hamming distance for graphs that is well aligned with the notation we have used thus far we will first introduce an auxiliary definition.

Definition 5. The *type* of an ordered node pair (u, v) given a set of edges E is given

by the function f_E defined as

$$f_E(u,v) := \begin{cases} \text{bidirected} & \text{if } uv \in E \text{ and } vu \in E, \\ \text{forward} & \text{if } uv \in E \text{ and } vu \notin E, \\ \text{backward} & \text{if } uv \notin E \text{ and } vu \in E, \\ \text{non-adjacent} & \text{if } uv \notin E \text{ and } vu \notin E. \end{cases}$$

Now the Hamming distance can be defined using the definition 5 and the indicator function.

Definition 6. The *Hamming distance* between graphs $G = (V, E)$ and $G' = (V, E')$ is given by

$$d_{\mathrm{H}}(G, G') := \frac{1}{2} \sum_{(u,v) \in V \times V} \left[f_E(u,v) \neq f_{E'}(u,v) \right].$$

The multiplier is necessary as without it a difference between an unordered node pair in the compared graphs would be counted twice, both for (u, v) and (v, u) for some fixed $u, v \in V$. We will use the Hamming distance, as defined above, exclusively on CPDAGs, and therefore denote it as d_{SHD}.

2.7.3 Approximating cross-entropy via cross-validation

If there is only a finite set of data, but the ground truth model is not known, neither SHD nor cross-entropy can be evaluated. As an alternative, one can divide the dataset into parts for training and validating, learn a model on the training part and evaluate the learned model based on how well it predicts the validation data (i.e., how high probability it gives to it). Equivalently, the mean log probability that a learned distribution q assigns to the validation data is computed as

$$\frac{1}{N} \sum_{i=1}^{N} \log q(x_i).$$

As can be seen, the expression is identical to that of Monte Carlo approximation of cross-entropy, except for a change of sign. Approximating cross-entropy between a distribution and some hypothetically existing true distribution underlying a dataset is thus possible based solely on the data. However, in this setting the amount of data is limited, so the accuracy of the approximation is too. Also, the learning only happens on one part of the data – the information in the validation set is not exploited.

Cross-validation is a technique by which the approximation can be made more accurate. Instead of splitting the data once into training and validation sets, the idea is to make a number of systematic splits, so that each data sample is used for learning at least once. Then, for each split, the test measure is computed, and finally the average of the results is reported. If the interest is in evaluating the performance of some algorithm given a dataset of N samples and the full data consists of $M > N$ samples there are $\binom{M}{N}$ ways in which the full data can be partitioned into the learning and testing parts. Instead of approximating the test measure at a chosen fixed amount of training data, a more common framework is to use as much as possible of the available data for training. Taken to extreme, each split would then consists of one sample for testing and the remainder for training. This is known as the *leave-one-out cross-validation*. As each learning process might be computationally expensive, in practice some simpler scheme is used. A common choice is k-fold cross-validation for some integer k. Using it, the data is split into k equally sized disjoint parts, each of which are used as a testing set once, thus defining k different splits of the data.

Analyzing the statistical efficiency of a learning algorithm in a setting where only a real-world dataset is available differs in two ways from the process described for the situation where the ground truth was known. Firstly, as the ground truth model is not available, the construction of datasets cannot be based on sampling from it. What is possible, however, is to take subsamples of the original dataset. Secondly, the distance measure cannot be based on directly comparing a learned model against the unknown ground truth model. We have seen a way to approximate cross-entropy via cross-validation. Intersection-validation aims to specify an analogous method for structural measures.

3 Intersection-Validation

Algorithms that learn Bayesian network structures from data are developed based on properties that ensure they perform well in some theoretical sense. When evaluating and comparing their performance, however, empirical studies are the norm. Basing the studies on data sampled from benchmark Bayesian networks gives the possibility to directly evaluate how well the different algorithms reconstruct the correct structure, using metrics such as the structural Hamming distance. If real-world data is preferred, one has been limited to evaluating the predictive performance of

the structures by assigning some fixed parameter values to the CPTs or marginalizing over the parameters using some prior. Intersection-validation is – to the best knowledge of the author – the first published method to extend evaluation specific to structure learning to situations where the correct structure is not known.

The input to the method is at least two structure learning algorithms and a dataset. First, *an agreement graph* is constructed from the features common to the structures discovered by the algorithms, given the full dataset. The agreement graph does not fit into existing graph formalisms, so the concept of *partial graph* is defined. Then, using the agreement graph as a proxy for the ground truth structure, the algorithms are evaluated against it on subsamples of the data. The metric applied is a variant of the Hamming distance defined for partial graphs, *the partial Hamming distance.*

Intuitively, the agreement graph consists of the features easiest to learn in the hypothetical ground truth, yet difficult enough to differentiate the algorithms on smaller sample sizes. The hope is that the conclusions that can be drawn about the performance of the algorithms using the method are the same one could draw if the ground truth was known. At a given sample size the conclusions can be for example the ranking of the algorithms or the magnitude of differences in the obtained distances to the target structure.

3.1 Agreement graph

The construction of the agreement graph corresponding to a dataset D and a set of algorithms $\{A_i\}_i$ starts with running each of the algorithms on the dataset. Denoting the CPDAG corresponding to the structure discovered by algorithm A_i on dataset D as $G_i = A_i(D)$, the first step produces a set of structures

$$\{A_i(D)\}_i = \{G_i\}_i = \{(V, E_i)\}_i.$$

The agreement graph, on the discovered structures, is defined through an intersection operation (Figure 6). However, defining the agreement graph for graphs G_i and G_j by intersecting the sets of edges to arrive at $G = G_i \cap G_j = (V, E_i \cap E_j)$ is unsatisfactory. First, if some node pair (u, v) is non-adjacent in G_i, but connected by an edge in G_j, it would be included as a non-adjacent node pair in G. Second, if the node pair is of the type forward in G_i and bidirected in G_j, it would be included as a forward node pair in G (since $\{(u, v)\} \cap \{(u, v), (v, u)\} = \{(u, v)\}$). Both of the results are misleading, since clearly the operands did not agree on the type of

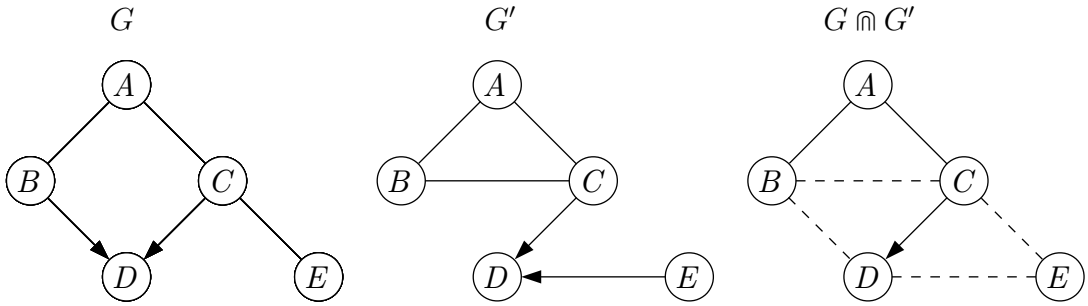


Figure 6: CPDAGs G, G' and their strict intersection. Dashed lines indicate node pairs not in the intersection.

connection between u and v. To overcome the above problem the notions of partial graph and strict intersection are defined.

Definition 7. A *partial graph* is a pair (S, E), where $S \subseteq V \times V$ is a set of node pairs and $E \subseteq S$ is a set of edges. The inclusion of an ordered node pair in S implies the inclusion of the node pair in the opposite order as well, that is, $(u, v) \in S \Leftrightarrow (v, u) \in S$.

Definition 8. A *strict intersection* of partial graphs $P = (S, E)$ and $P' = (S', E')$ is the partial graph $P \Cap P' = (I, I \cap E \cap E')$, where $I \subseteq S \cap S'$ is the set of node pairs with the same type in P and P'.

An ordinary graph, such as a CPDAG, is a special case of partial graph, where $S = V \times V$. For CPDAGs $G = (S, E)$ and $G' = (S', E')$ the initially considered intersection thus results in $G \cap G' = (I, I \cap E \cap E')$, where $I = S \cap S'$, in contrast to the strict intersection in which only a subset of the common node pairs are included. Both intersections are associate and commutative so the operations can be executed in any pairwise order among a set of operand graphs, allowing for the following definition:

Definition 9. The *agreement graph* corresponding to a dataset D and a set of algorithms $\{A_i\}_i$ is the partial graph $G_0 = \bigCap_{i=1}^{k} A_i(D)$.

3.2 Partial Hamming distance

The agreement graph is a partial graph. Also, any CPDAG learned by one of the compared algorithms is a (special case of) partial graph. To measure the distance between them, a metric on partial graphs thus has to be defined. Partial Hamming

distance is a straightforward extension of Hamming distance for partial graphs. It is defined as the number of unordered node pairs that differ with respect to the type of connection between them in the two operand partial graphs.

Definition 10. The *partial Hamming distance* between partial graphs $P = (S, E)$ and $P' = (S', E')$ is given by

$$d_{\mathrm{PHD}}(P, P') := \frac{1}{2} \sum_{(u,v) \in S \cap S'} [f_E(u, v) \neq f_{E'}(u, v)]$$

where f_E is a function mapping an ordered node pair to its type in E (definition 5, p. 22).

From the assumption $d_{\mathrm{PHD}}(P, P') = 0$, it does not generally follow that $P = P'$, since the distance only considers node pairs common to both operands. The partial Hamming distance as defined thus violates the identity of indiscernibles, and therefore is not a metric (Definition 4, p. 19). However, with the additional constraint of a fixed set of node pairs, on which the operands are defined, PHD is a metric.

Proposition 3. *Partial Hamming distance is a metric in the set of partial graphs on a fixed set of node pairs.*

Proof. Clearly, PHD is non-negative. From the definition we also directly see that $d_{\mathrm{PHD}}(P, P') = 0 \Leftrightarrow P = P'$, and the symmetry of PHD follows from the symmetry of the inequality relation. To prove that PHD satisfies the triangle axiom, we specify the partial graphs $\mathcal{P} = (S, P)$, $\mathcal{Q} = (S, Q)$, and $\mathcal{R} = (S, R)$ and the sets

$$\begin{aligned} A &= \{(u, v) : (u, v) \in S, f_P(u, v) \neq f_R(u, v)\}, \\ B &= \{(u, v) : (u, v) \in S, f_P(u, v) \neq f_Q(u, v)\}, \\ C &= \{(u, v) : (u, v) \in S, f_Q(u, v) \neq f_R(u, v)\}. \end{aligned}$$

For all $(u, v) \in A$, we have that if $(u, v) \notin B$ then $(u, v) \in C$, and if $(u, v) \notin C$ then $(u, v) \in B$. It holds thus that $A \subset B \cup C$, and therefore

$$d_{\mathrm{PHD}}(\mathcal{P}, \mathcal{R}) = \frac{1}{2}|A| \leq \frac{1}{2}(|B| + |C|) = d_{\mathrm{PHD}}(\mathcal{P}, \mathcal{Q}) + d_{\mathrm{PHD}}(\mathcal{Q}, \mathcal{R}).$$

□

3.3 Evaluation pipeline

To evaluate the performance of an algorithm (among a set of them) at a given sample size using intersection-validation we first run the algorithm on subsamples

of the chosen size of the original input data. Then the PHDs between the output structures and the agreement graph are measured and the results are reported. The process is repeated for varying sample sizes to analyze the statistical efficiency of the algorithm, that is, to obtain a "learning curve" representation of the results.

The details of the above process need not be fixed. Here, however, we present the way the method was used in the experiments, with the hope that the choices made represent a good baseline for the general case.

Assuming thus that D is a dataset of size N, $\mathcal{A} = \{A_i, \ldots, A_k\}$ is a set of algorithms, and $G_0 = \bigcap_{i=1}^{k} A_i(D)$ is the agreement graph of $\mathcal{A}$ on D, the evaluation of the algorithms at some sample size $s < N$ follows the following process. First, we draw $r = 10$ independent subsamples of size s uniformly at random, without replacement from D, denoted as $D_1, \ldots, D_r$. Next, we compute the CPDAG structures, $G_{ij} = A_i(D_j)$, and their distances to the agreement graph, $d_{ij} = d_{\mathrm{PHD}}(G_0, G_{ij})$, for each algorithm ($i \in [k]$) and subsampled dataset ($j \in [r]$). Finally, for each algorithm the mean distance

$$\mu_i^r(s) := \frac{1}{r} \sum_{j=1}^{r} d_{\mathrm{PHD}}(G_0, G_{ij}) \tag{12}$$

and the standard error of the mean

$$\sigma_{\mu_i^r(s)} := \frac{\sigma}{\sqrt{s}} = \frac{\sqrt{\frac{\sum_j \left(d_{ij} - \mu_i^r(s)\right)^2}{s-1}}}{\sqrt{s}},$$

where σ is the population standard deviation, are reported. The process is then repeated for subsample sizes following the pattern $N/2^i$ for $i = 1, \ldots, m$ so that $N/2^m \leq 100$.

3.4 Asymptotic consistency

The rationale presented thus far for the intersection-validation method has been informal. In the asymptotic case, assuming the method is applied onto asymptotically consistent algorithms, it, however, has a simple theoretical justification.

Let $\mathcal{B}$ then be a set of Bayesian networks defined on a fixed set of variables, and θ a map from them to some set R. Further, let $T^{(N)}$ be a function from a dataset of N samples to an element in R. The sequence of functions $\left(T^{(1)}, T^{(2)}, \ldots, T^{(N)}\right)$ is called a *consistent estimator* of θ if for all $(G, p) \in \mathcal{B}$ the sequence converges in

probability (definition 1, p. 7) to $\theta\left(G, p\right)$ as N tends to infinity, assuming the data consists of independent draws from p.

Convergence in probability was defined in terms of real valued random variables. A structure learning algorithm given input data of size N, sampled from a ground truth distribution, can be viewed as a random variable, whose codomain is the space of CPDAGs. To consider the convergence of such a random variable, we should first thus encode the possible CPDAGs with some unique numbers. Doing the encoding implicitly, a structure learning algorithm A_i is a consistent estimator of the ground truth CPDAG C^*, or just *consistent*, if

$$A_i\left(D^{(N)}\right) \xrightarrow{P} C^*,$$

where $N = \left|D^{(N)}\right|$. A scoring function is called consistent if the score maximizing structure converges in probability to a member of the correct equivalence class as $N \to \infty$. All of the scoring functions considered in the thesis, except for AIC, are consistent.

Let D be a dataset of s samples, and $\mu_i(s)$ the expected value of $d_{\mathrm{SHD}}(A_i(D), C^*)$. We call the mean distance in formula 12 the *intersection-validation estimator* for the expected SHD. Its consistency follows from the consistency of the input algorithms.

Theorem 4. *Let $A_1, A_2, \ldots, A_k$ be consistent learning algorithms. Let $i \in [k]$ and $s \in \mathbb{N}$. Then $(\mu_i^N(s))_{N=s}^{\infty}$ is a consistent estimator for $\mu_i(s)$.*

Proof. Let ε and δ be positive numbers. To prove the theorem we should show that for all $i \in [k]$ and $s \in \mathbb{N}$, there is some N_0 such that for $N > N_0$ the following holds

$$P\left(\left|\mu_i^N\left(s\right) - \mu_i\left(s\right)\right| > \varepsilon\right) < \delta.$$

Let C^* be the ground truth CPDAG and let p be the ground truth distribution. The proof (1) first shows that the probability of the agreement graph A_0 equaling C^* can be made arbitrarily close to 1, and specifically larger than $1 - \delta$, by learning the operand structures on a large enough dataset from p. Then (2) assuming the agreement graph equals the ground truth CPDAG, we show that the subsampling distribution converges to the ground truth distribution from which the claim follows.

1. From the consistency of the algorithms it directly follows that there is N' such that for all $i \in [k]$

$$P\left(A_i\left(D^{(N')}\right) = C^*\right) > 1 - \frac{\delta}{2k},$$

where $D^{N'}$ is a dataset of N' samples. In other words, we can choose such dataset size that independently for each algorithm the probability of the algorithm not finding the correct CPDAG is less than (or equal to) $\frac{\delta}{2k}$.

Using Boole's inequality, we can then bound the probability of the event that at least one algorithm does not learn the correct CPDAG as

$$P\left(\cup_{i=1}^{k}\left\{A_i\left(D^{(N')}\right) \neq C^*\right\}\right) \leq \sum_{i=1}^{k} P\left(A_i\left(D^{(N')}\right) \neq C^*\right) \leq k \cdot \frac{\delta}{2k} = \frac{\delta}{2}.$$

Equivalently, the probability of the agreement graph equaling the ground truth CPDAG is now lower bounded as

$$P\left(A_0 = C^*\right) > 1 - \frac{\delta}{2}.$$

2. Assume then that the agreement graph equals the ground truth CPDAG. Now it holds that $d_{\text{PHD}}\left(A_0, A_i\left(D\right)\right) = d_{\text{SHD}}\left(C^*, A_i\left(D\right)\right)$. With the assumption $A_0 = C^*$ we can thus treat the two metrics as equal, when measuring distances to the agreement graph. Let $D = (X_1, X_2, \ldots, X_N)$ be a dataset of size N, where $X_i \sim p$ are random samples from the ground truth distribution. For a fixed algorithm A_i and subsample size s define

$$Y_{j,N} := d_{\text{SHD}}\left(C^*, A_i\left(D_j\right)\right),$$

where $D_j \subset D$ is a random variable corresponding to the j^{th} subsample from D. To prove the theorem, it is now enough to show that for the empirical mean of the random variables $Y_{j,N}$, $j = 1, \ldots, N$ it holds that

$$\frac{1}{N}\sum_{j=1}^{N} Y_{j,N} \xrightarrow{P} \mathbb{E}\left[Y_{1,N} \mid F_N\right] \xrightarrow{P} \mu_i\left(s\right),$$

where

$$F_N\left(x\right) = \frac{\left|\{i : X_i = x\}\right|}{N}$$

is the empirical distribution of X.

The first convergence follows from the law of large numbers, as the variables $Y_{j,N}$ are i.i.d. given F_N. The second convergence follows from

$$F_N\left(x\right) \xrightarrow{P} P\left(X_1 = x\right),$$

for all x, from which it follows that for each $z = (x_1, \ldots, x_s)$ the conditional probability $P\left(D_1 = z \mid F_N\right)$ converges to probability $P\left(D = z\right)$. □

4 Experimental results

Intersection-validation is intended to be used as a replacement for SHD based evaluation in situations where the ground truth structure is not known. Ideally, any conclusions one could draw concerning the performance of a structure learning algorithm among a set of them when the ground truth is known, one should be able to draw using intersection-validation when the ground truth is not known. Intersection-validation thus, as a method, is evaluated by considering the following question:

Are the results of the method similar to those obtainable with ground truth?

To explore the question, an experiment pipeline is specified, where intersection-validation based PHD results are compared to ground truth based SHD results. In order to conduct the experiments, we sample datasets from the benchmark Bayesian networks `Sachs` (Sachs et al. 2005), `Child` (Spiegelhalter et al. 1993), `Insurance` (Binder et al. 1997), `Water` (Jensen et al. 1989), `Alarm` (Beinlich et al. 1989) and `Barley-Fungal` (Kristensen and Rasmussen 2002), whose basic properties are given in Table 1. The networks were downloaded from `www.bnlearn.com/bnrepository`, except `Barley-Fungal` which was downloaded from `repo.bayesfusion.com`. From each benchmark Bayesian network we sampled independent datasets of sizes 50×2^i where $i \in \{0, \ldots, 8\}$, repeating the process 10 times for a total of 90 datasets per network.

The compared algorithms were formed by combining each of the scoring functions AIC, BIC, BDeu (with ess 1), fNML and qNML to an exact search algorithm. The local scores were calculated with the software `bene` (Silander and Myllymäki 2006) and the search was performed with the integer linear programming based software `Gobnilp` (Cussens 2011, Barlett and Cussens 2013).

Table 1: The benchmark networks used in the experiments.

Name	Nodes	Edges	Max Indegree	Parameters
`Sachs`	11	17	3	178
`Barley-Fungal`	15	19	4	43007
`Child`	20	25	2	230
`Insurance`	27	52	3	984
`Water`	32	66	5	10083
`Alarm`	37	46	4	509

The differences among the compared algorithms are thereby limited to the differences between the scoring functions themselves. Specifying the algorithms in this manner does not eliminate all random variation from the results. It does, however, rule out one layer of uncertainty that would arise from the use of heuristic algorithms. The choice of algorithms also makes the results in principle replicable.

4.1 Evaluation with established methods

To evaluate intersection-validation, we must first obtain a point of reference by evaluating the considered algorithms against the ground truth structure using the SHD as a metric. Also, by additionally evaluating the algorithms using the other two established methods, namely cross-entropy and its approximation obtained through cross-validation, we can consider the relationships among them. An obvious preliminary question, for example, is to what extent cross-validation can be used to assess the performance of structure learning algorithms when the ground truth is not known.

We therefore computed the quantities at each considered sample size and averaged the results over the ten independent datasets. In order to compute cross-entropy and its approximation through cross-validation we equipped the learned structures with mean posterior estimates of the parameters, using equivalent sample size of 1, given the data that was used to learn each structure. Then the cross entropies between the ground truth distribution and the learned distributions were approximated using the Monte Carlo method discussed previously. The log probability of each dataset was approximated with 10-fold cross-validation. The sign of the log probability is changed to make the numbers conform to those of cross-entropy. Figure 7 presents the results for all of the three measures.

It can be seen clearly, that the networks pose different levels of difficulty in terms of structure learning, although in none of the networks is 12,800 samples sufficient for all of the scores to find a structure from the correct equivalence class. For `Sachs` and `Child` this observation, however, is only a consequence of the AIC score not being consistent – all the other scores find a correct structure (i.e., zero SHD) already by 6,400 samples. The choice of the number of independent datasets to use for each sample size has a direct effect on the standard error of the mean, represented by the vertical bars in the plots, with more repetitions yielding smaller standard errors.

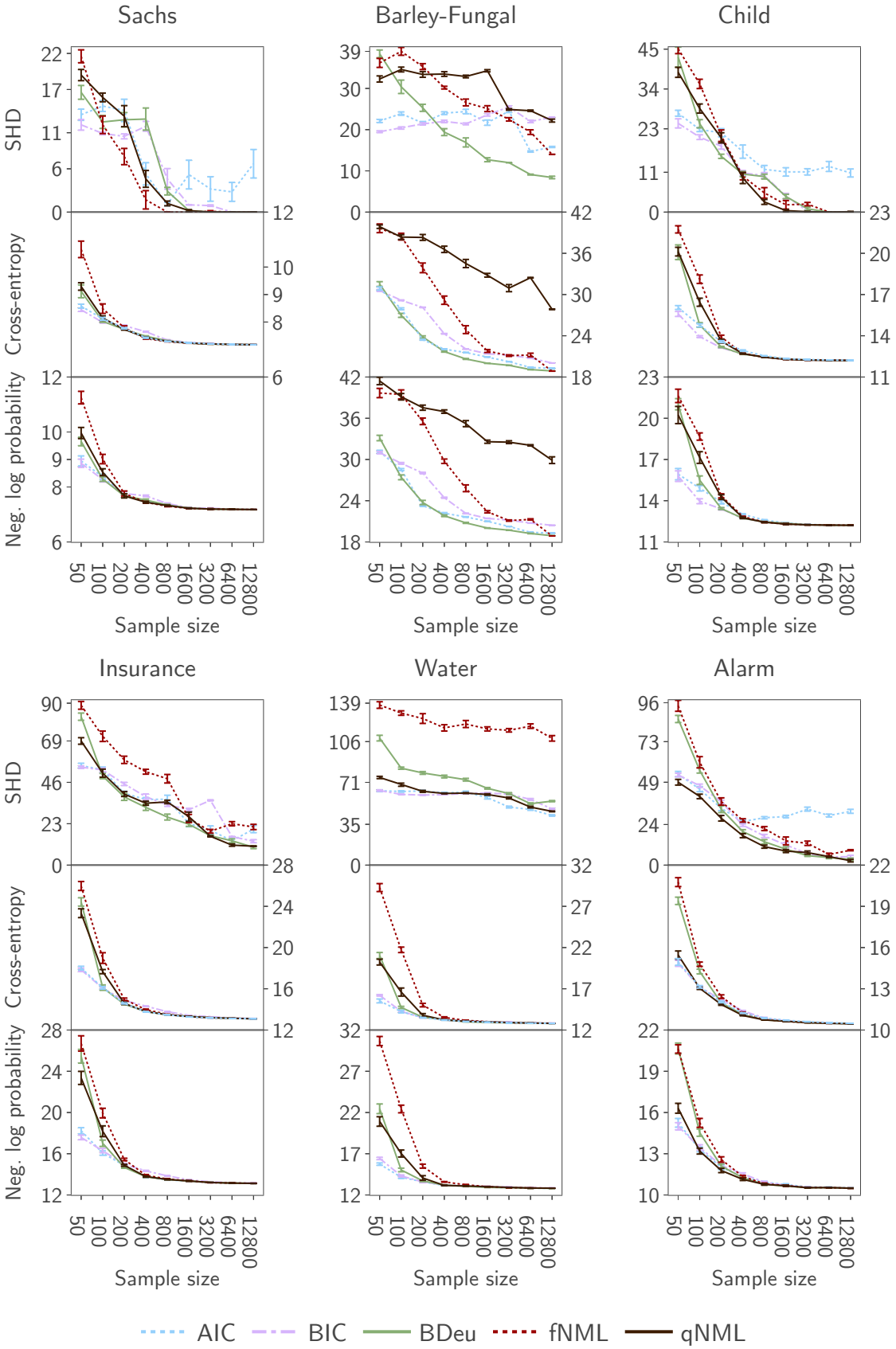


Figure 7: Mean values for structural Hamming distance, cross-entropy and negative log probability, where the last quantity is computed with 10-fold cross-validation.

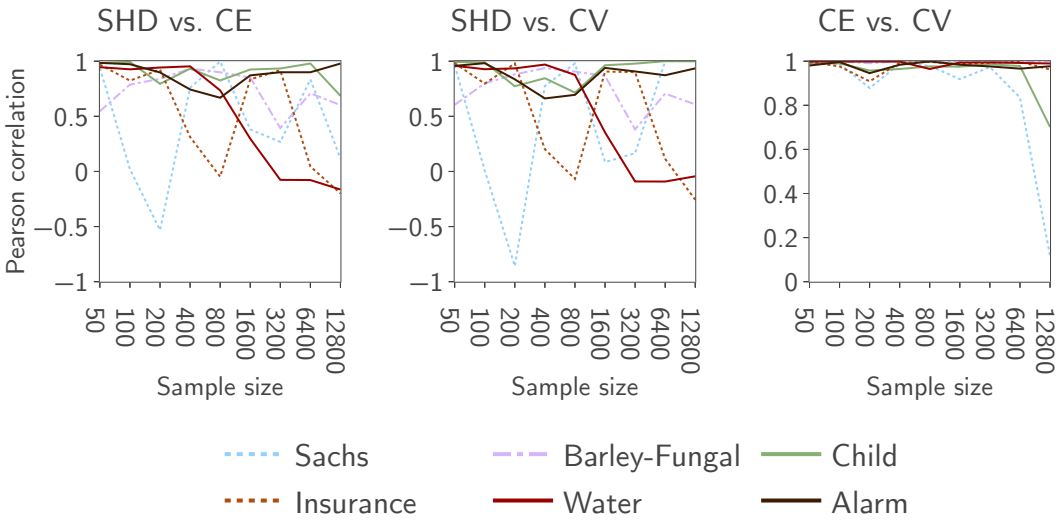


Figure 8: Correlations between SHD, CE and CV.

By visually comparing the cross-entropy results to the cross-validation ones it is clear that the latter approximates cross-entropy accurately even on small sample sizes. Comparing the SHD and the cross-entropy results reveals, perhaps less obviously, that the relative performance of a structure learning algorithm in finding well predicting structures does not always seem to imply equal performance in terms of accuracy of the structures. Computing the correlations between the SHD and cross-entropy values as shown in Figure 8 confirm these observations. In a setting without ground truth one cannot thus rely on cross-validation if the interest lies in evaluating the structure learning performance of a set of algorithms.

4.2 Performance of intersection-validation

To apply intersection-validation, a choice has to be made regarding the size of the input data. When the method is used in a "real life" setting this question does not arise, as there is only a certain amount of data available. In the set up of the experiments, where we have data sampled at various dataset sizes, and a generative model from which we could draw data in arbitrarily large amounts, some number needs to be fixed. We call the chosen amount of data the *intersection point*, as it determines the point on the scale of considered sample sizes at which we apply the strict intersection operation on the learned CPDAGs to obtain the agreement graph.

With a fixed intersection point N, the method produces PHD results between the

agreement graph learned on N samples and structures learned on fewer than N samples. In the asymptotic case, when applying the method on a set of consistent algorithms, the PHD results converge to the SHD between the ground truth and the learned structures (Theorem 4, p. 29). Intuitively, one would then expect the PHD results to reflect the corresponding SHD results more accurately as the intersection point is increased, even with finite data sizes. Conversely, a question of interest becomes how low can we set the intersection point and still expect to obtain meaningful results?

To consider the similarity between the PHD and the SHD results, we must specify how we measure similarity. Since we are using intersection-validation to evaluate the relative performance of a set of algorithms, a simple criterion for the similarity is to compare the rankings that the PHD and SHD measurements determine for the algorithms. A second, stronger, criterion for similarity is to consider the magnitudes of relative differences between the algorithms at the various sample sizes. Similarity in the latter sense would then imply similarity in the former sense as well.

4.2.1 Visual similarity to structural Hamming distance

To begin analysing the results, we will first, however, rely on visual inspection to compare the SHD and PHD values. We thus set the intersection point $s_{\Cap}$ to 1,600. Then for each sample size $s < s_{\Cap}$ we computed the mean PHD values and compared them with the corresponding mean SHD values. To be clear, the PHD values are computed by first averaging the PHDs measured on 10 subsamples of size s from a full dataset whose size is given by the intersection point. This process is repeated for each of the 10 independent full datasets, and finally the mean of the means is reported. The standard errors of the means in the SHD case are based on the 10 independent SHD measurements, whereas for PHD they are calculated from the 10 independent mean PHD measurements. The results are plotted side by in Figure 9.

We can make an immediate observation: the results look strikingly similar. The PHD results do not reproduce the SHD ones perfectly, however, as some differences can also be seen, most noticeably in the `Barley-Fungal` network. A somewhat trivial observation is, that the absolute PHD values tend to be smaller than the SHD ones. This is a direct consequence of the agreement graph consisting of only some subset of the total node pairs, even if strictly speaking it does not guarantee lower number of mismatching node pairs. For example, the mean PHD to the underlying agreement graph at sample size 800 using BIC on `Sachs` data is greater than the corresponding

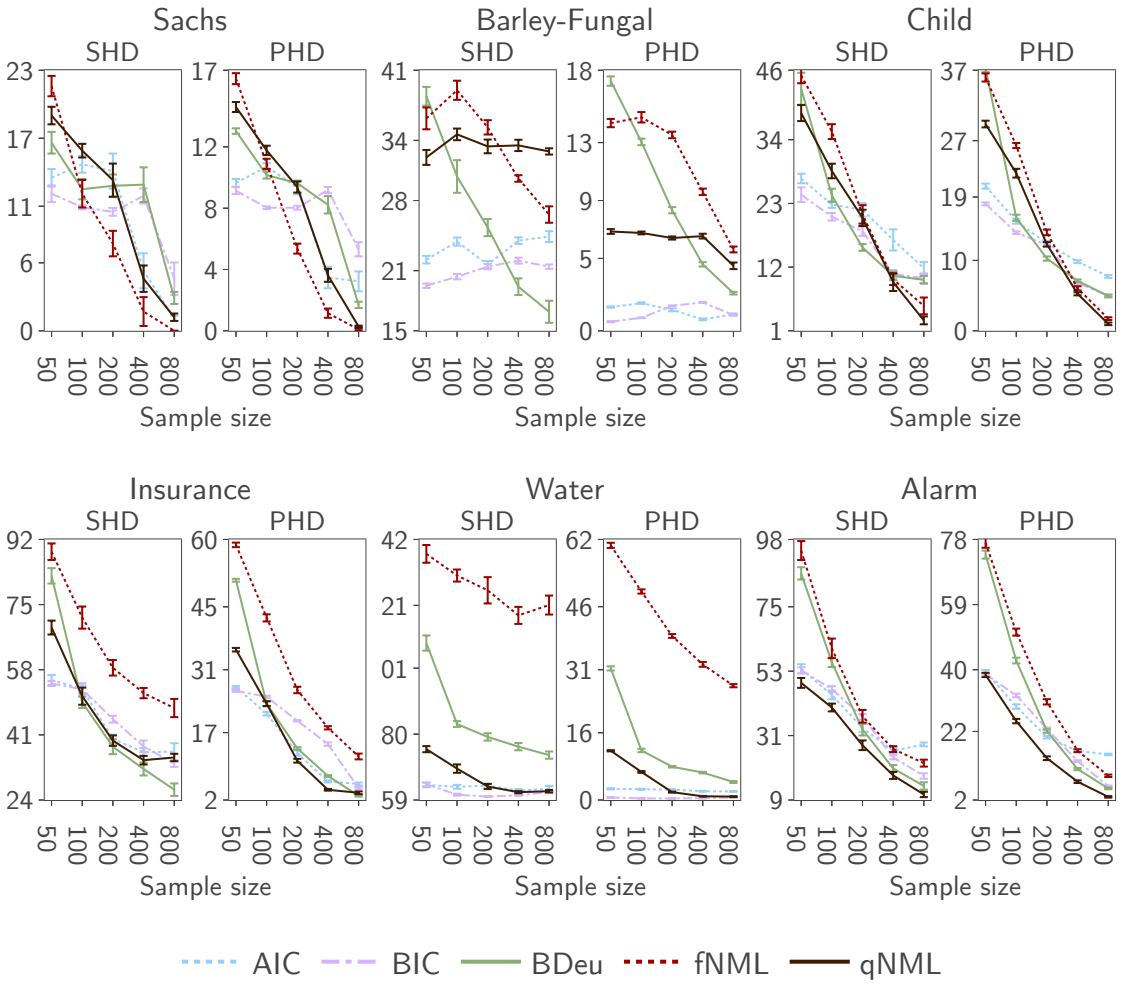


Figure 9: Comparison of ground truth based SHD results (left) to intersection-validation based PHD results (right), with intersection point set to 1600.

mean SHD to the ground truth structure.

It is not clear how to best compare the curves, to consider how close the PHD curves are to the SHD ones. Though having the disadvantage of subjectivity, visual inspection permits us to freely concentrate on any aspect of the curves deemed important by any particular use case for the method. The aspects could be the ones discussed, the ranking and the magnitudes of differences, or for example the convergence rates of the algorithms. In such sense, visual inspection is the most information preserving method for comparing the results.

Visual inspection does not, however, assign a number on the similarity between the results. To be able to objectively analyze how the similarity varies as a function of the inputs to the method we need to quantify it. For example, it seems intersection

validation works better for data sampled from `Sachs` than from `Barley-Fungal`. Though the difference, if sufficiently pronounced as it is between the two networks, might be clearly visible in the graphical representation, it will be difficult to see subtle ones, let alone compare such differences against each other.

4.2.2 Similarity of rankings

To quantify the similarity between the SHD and PHD results, we will begin by considering the question of rankings given by the two metrics. To compare the rankings we will use two measures: Spearman (rank) correlation, and the ratio of matching pairwise rankings. Specifically, for each intersection point $s_{\Cap}$ we have a set of pairs

$$\mathrm{r}_{\mathrm{SHD}}(A_i, s), \mathrm{r}_{\mathrm{PHD}}(A_i, s|s_{\Cap})$$

where $\mathrm{r}_{\mathrm{SHD}}(A_i, s)$ is the SHD based ranking for algorithm A_i at sample size s, and where $s < s_{\Cap}$. The notation for the PHD based ranking is only different in that it conditions the rank on the intersection point. Due to the setup of the experiments we cannot directly establish a correspondence between SHD and PHD values. We can, however, form meaningful pairs from the mean distances. Therefore, the rankings are determined by the mean measurements obtained as previously explained. The correlation is then computed from this set.

For the ratio of matching pairwise rankings, we iterate over every combination (A_i, A_j) of pairs of algorithms and count the number of times the equivalence

$$(\mathrm{r}_{\mathrm{SHD}}(A_i, s) < \mathrm{r}_{\mathrm{SHD}}(A_j, s)) \Leftrightarrow (\mathrm{r}_{\mathrm{PHD}}(A_i, s|s_{\Cap}) < \mathrm{r}_{\mathrm{PHD}}(A_j, s|s_{\Cap}))$$

holds true, divided by the total number of observations. Observation here refers to the computation of the rankings at some sample size for a pair of algorithms, given the intersection point. The results presented in Figure 10 show that the measure is largely equivalent with the Spearman rank correlation, with both showing moderate to strong results, as hinted by the results in Figure 7. The overall ratio of matching pairwise rankings is 0.875. One should note, however, that the rankings might be somewhat arbitrary in cases where the mean distance values are so close to each other that the standard errors overlap. Moreover, the SHD and PHD values at a given sample size should be understood as data dependent random variables. Whether the mean is a meaningful statistic depends on the shape of the distribution. Though not presented here in any detail, a preliminary analysis shows the standard deviations in the empirical distributions are moderate, so focusing on the means seems reasonable.

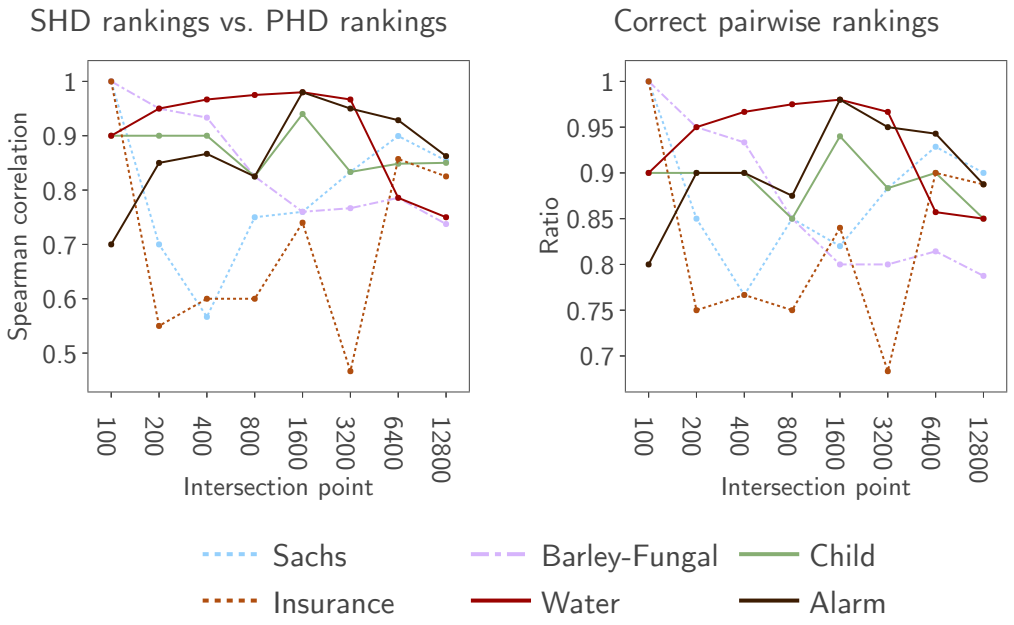


Figure 10: Spearman correlation between SHD and PHD rankings (left) and ratio of matching pairwise SHD and PHD rankings (right).

4.2.3 Correlations and root-mean-square error

Next we will consider three performance measures given intersection point and sample size. The quantities based on which we compute the measures are therefore the sets of pairs

$$\bar{d}_{\mathrm{SHD}}\left(A_i|s\right), \bar{d}_{\mathrm{PHD}}\left(A_i|s, s_{\Cap}\right),$$

where $\bar{d}_{\mathrm{SHD}}\left(A_i|s\right)$ is the mean SHD for algorithm A_i at sample size s and $\bar{d}_{\mathrm{PHD}}\left(A_i|s, s_{\Cap}\right)$ is the mean of the mean PHDs measured for algorithm A_i at sample size s when the intersection point is set to $s_{\Cap}$. In each set there is 5 such pairs, corresponding to each of the considered scoring functions.

First performance measure is the Spearman correlation for the rankings determined by the above quantities. The only difference to the previous analysis is that in addition to the intersection point we also condition on sample size.

Second is the Pearson correlation between the mean distances. For the Pearson correlation we could not condition the results only on the intersection points as we did for the ranking based correlations, as there is a nearly monotonic relationship between both of the distance metrics and the sample size (see Figure 9). Calculating

the Pearson correlation per intersection point would then allow the sample size to have a strong confounding effect on the metrics, and consequently invalidate the obtained correlation coefficient.

Third measure is the *root-mean-square error* (RMSE) obtained by fitting a least squares regression line to the sets of mean distance pairs (i.e., points in $\mathbb{R}^2$). The procedure is equivalent to finding the scaling and translation of the PHD values that minimizes the squared error between them and the corresponding SHD values.

Finally, to obtain some baseline upon which we expect intersection-validation to improve, we computed the same performance measures based on the pairs

$$\bar{d}_{\mathrm{SHD}}\left(A_i|s\right), \bar{d}_{\mathrm{CE}}\left(A_i|s,\right),$$

where the second quantity is the mean cross-entropy estimate we computed with cross-validation earlier (Figure 7). The quantity, similarly to PHD, does not rely on knowing the ground truth. Correlations above and RMSE below the baseline consequently indicate improvement in our ability to evaluate structure learning without ground truth.

The results in Figure 11 show that, though in general the correlations are moderately strong, for some of the networks for certain sample sizes they suddenly drop. On closer inspection a reason for most of these ocurrences can be observed in the SHD curves, where the drops in correlations correspond to the distances measured for the various algorithms coming very close to each other. Further, in cases where all of the distances are close to each other both the Spearman and Pearson correlations drop (`Sachs` at sample sizes 100, 200; `Child` at 200; `Insurance` at 1600). In cases where all the distances except for one algorithm are close to each other the Spearman correlation drops while the Pearson correlation remains strong (`Insurance` at 100; `Water` at 800; `Alarm` at 3200, 6400).

The only clear exception to the pattern is `Barley-Fungal`, where the drops in correlations are due to intersection-validation producing unarguably wrong results. Also, the correlations for the network decrease with higher sample sizes, in contrast to a weak pattern indicating the opposite trend in the other networks. The last observation seems explainable by the slow convergence of the SHD results for the network (Figure 7). With higher sample sizes `Barley-Fungal` should see similar improvement in the correlations as the other networks.

The strong performances across all the networks at the smallest sample size might mostly be indicative of the varying amount of penalty that the scores impose on

complexity. Without sufficient data, structure discovery for each algorithm is largely guesswork, where the complexity penalty determines the number of edges, and consequently the expected distance to the correct structure. At large sample sizes many of the algorithms start converging on the ground truth CPDAG, inflating the Pearson correlation if there is one outlier algorithm, as there seems to be for each network except `Barley-Fungal` and `Insurance`. The sample sizes in between the extremes seem the most relevant for assessing the performance of intersection-validation expected in practice. Though it is there that the method performs the weakeast, the results still seem strong enough to make the method useful.

Importantly, intersection-validation seems to mostly improve on the baseline. In places where the baseline performs better the difference seem moderate. At no sample size in any of the networks do the correlations between the SHD and PHD results fall below zero, whereas for the baseline this happens for `Sachs`, `Insurance` and `Water`. Intersection-validation thus seems more reliable method for evaluating the structure learning performance of a set of algorithms than trying to infer the same from the algorithms' predictive performance.

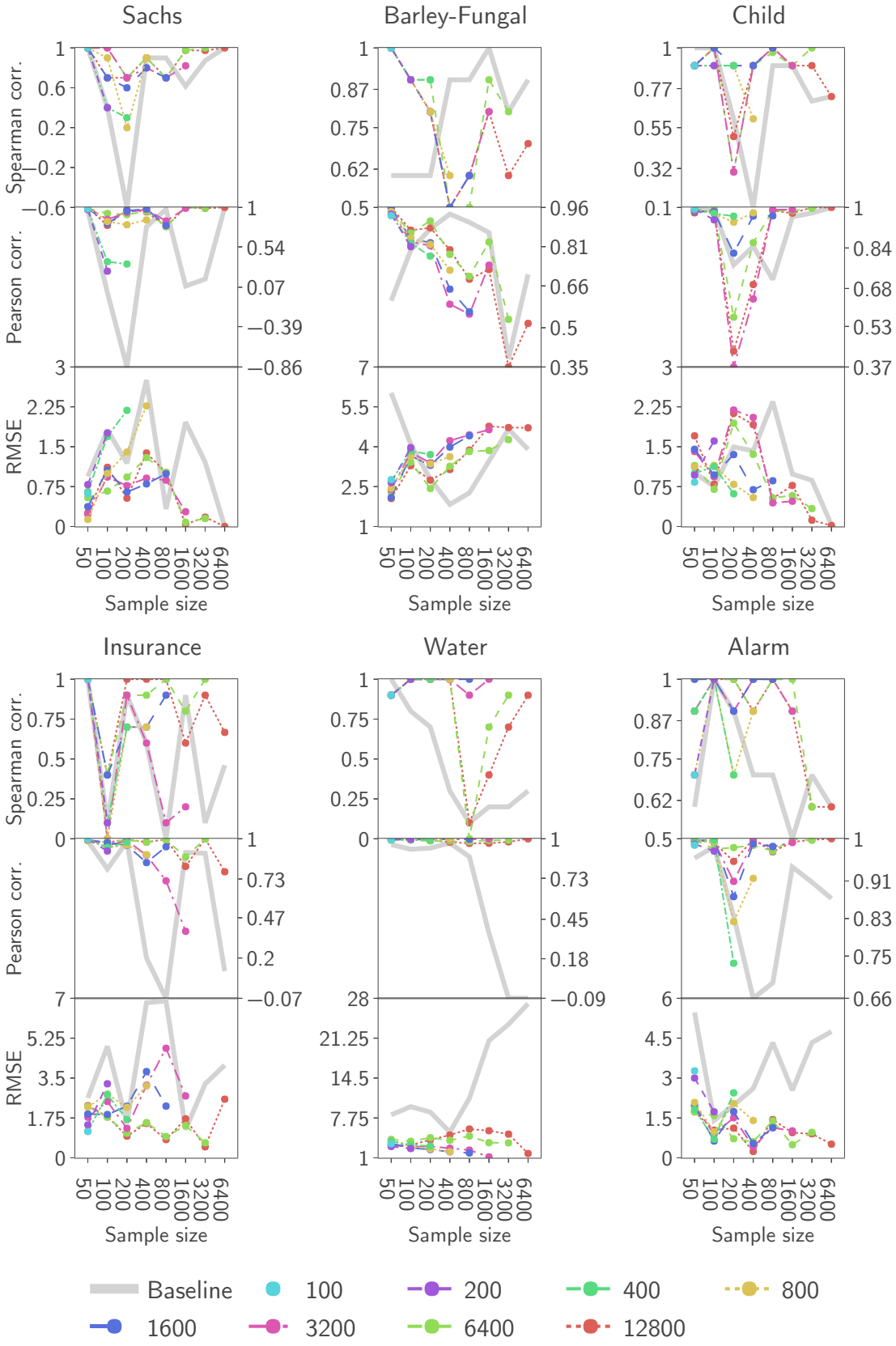


Figure 11: Three performance measures (Y-axis) given sample size (X-axis) and intersection-point (legend) to evaluate similarity of SHD and PHD results: Spearman rank correlation, Pearson correlation and RMSE for least squares regression from PHD to SHD. Baseline shows the same between SHD and cross-validated CE.

4.3 Quality of the agreement graphs

Intersection-validation effectively is based on using the agreement graph as a proxy for the ground truth structure. It would seem reasonable then, that the performance of the method depends on the quality of the agreement graphs. To gain an initial understanding into how the agreement graphs develop as the intersection point is increased, Figure 12 visualizes the CPDAG and the agreement graph at three different intersection points for the benchmark network `Barley-Fungal`. There are 10 independent datasets for each sample size, each producing a possibly differing agreement graph. The visualized graphs correspond to the datasets that had been assigned the index number 1. The dashed lines indicate, as in the minimal example seen earlier (Figure 6), node pairs excluded from the agreement graph.

Defining the *agreement graph size* as the ratio between the number of node pairs included in it and the total amount of node pairs, we can see the size grow – i.e., the number of dashed lines decrease – with the intersection point, corresponding to a greater agreement among the compared algorithms. The number of *connected node pairs*, that is node pairs connected by any type of an edge, does not seem to grow very fast, however. In the agreement graph corresponding to intersection point 800 there is only a single edge, and only 5 when the intersection point is set to 12,800. We can also see some differences between the agreement graphs and the ground truth: at intersection point 800 the algorithms agree not to have an edge between the nodes `areal` and `lt22`, while the ground truth does have an edge from the former to the latter; at 12,800 the algorithms agree to have an edge between `udbrsv` and `lt22`, while there is no such edge in the ground truth.

To begin analyzing the observations systematically Figure 13 plots the sizes of the agreement graphs and the numbers of connected node pairs as a function of the intersection point for each network. As a further analysis, also the mean pairwise distances between the agreement graph operand CPDAGs is reported. We can observe, that the `Barley-Fungal` network seems to be the most difficult to agree on for the algorithms. Even if the size of the agreement graph grows with every increase of the intersection point, it is only at 6,400 samples that the algorithms start finding edges to agree on.

Assuming we know the ground truth structure, we can also consider the types of errors in the agreement graphs. An error is defined as a difference in the edge type between a pair of nodes in the agreement graph and the ground truth CPDAG, among the node pairs included in the agreement graph. The results presented in

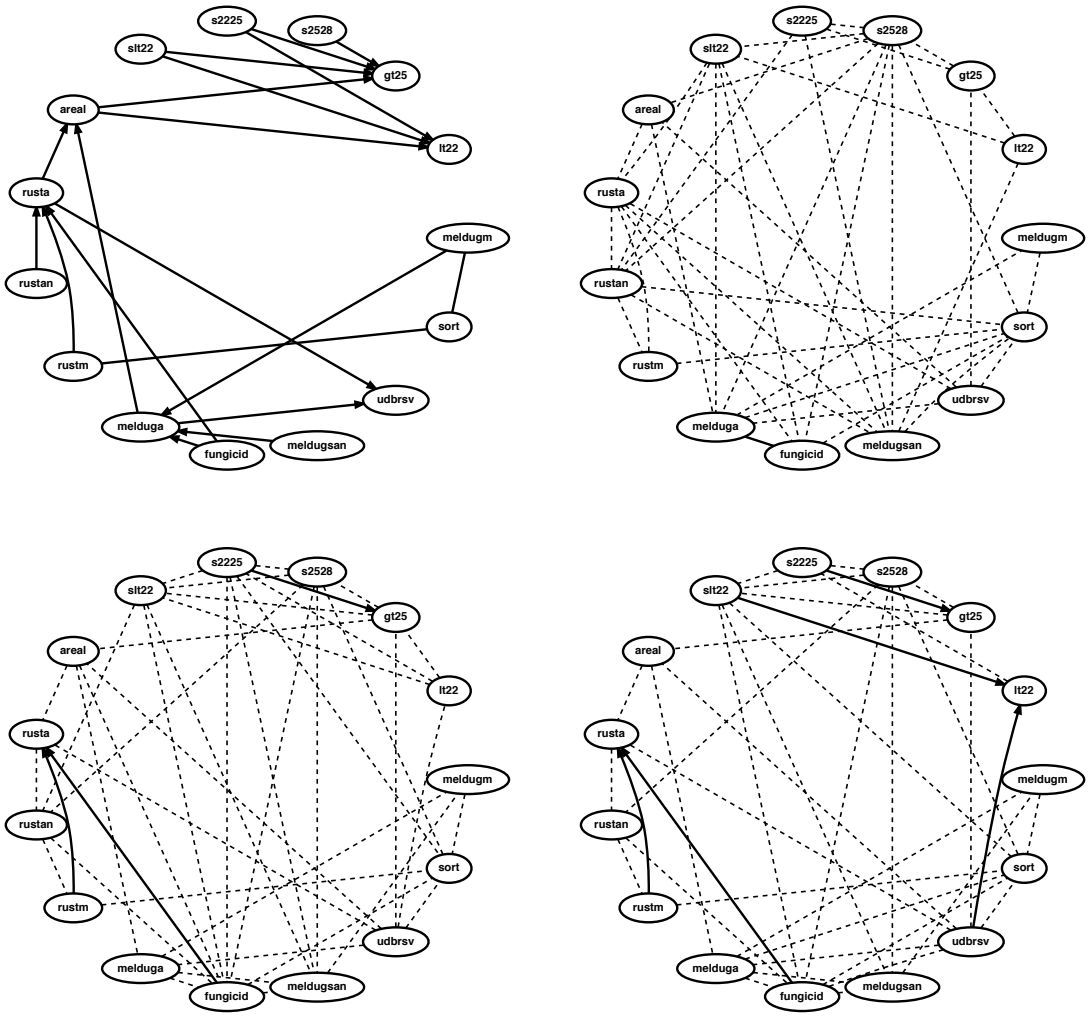


Figure 12: From left to right, top to bottom: `Barley-Fungal` ground truth CPDAG and agreement graphs at intersection points 800, 6,400 and 12,800 for first of the 10 datasets.

Figure 14 distinguish between three types of errors: missing edges (relative to the ground truth structure), spurious edges, and incorrect orientations. The last error type refers to node pairs where the compared graphs disagree on the direction of a directed edge, or on whether the edge is directed or bi-directed. We can see that missing edges represent the vast majority of the errors, and that there is in fact very few spurious edges, less than 1 on average for any intersection point for any network.

The above analysis disregards the fact that unconnected node pairs form the ma-

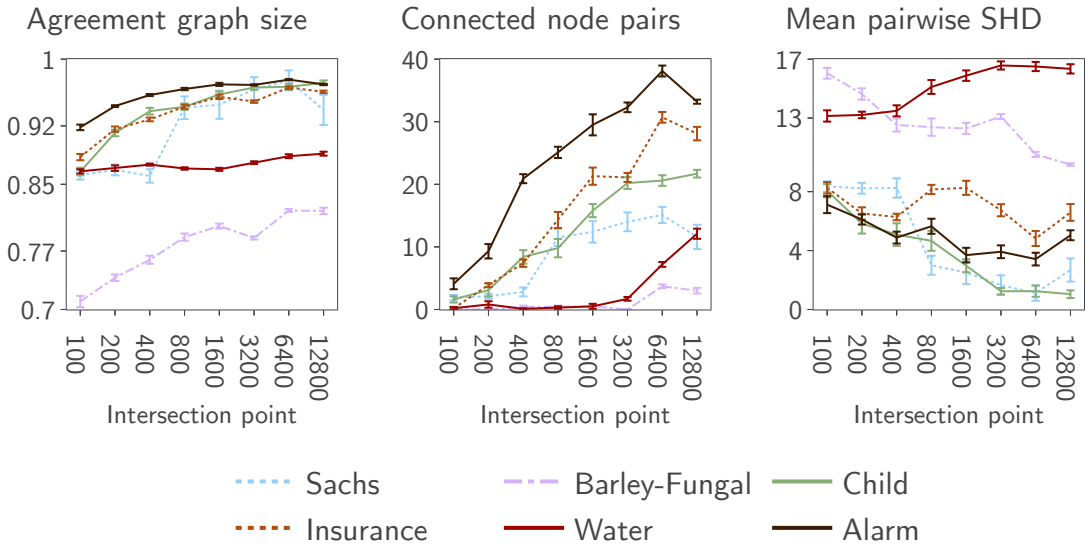


Figure 13: Agreement graph size, number of connected node pairs and mean pairwise SHD of the agreement graph operands.

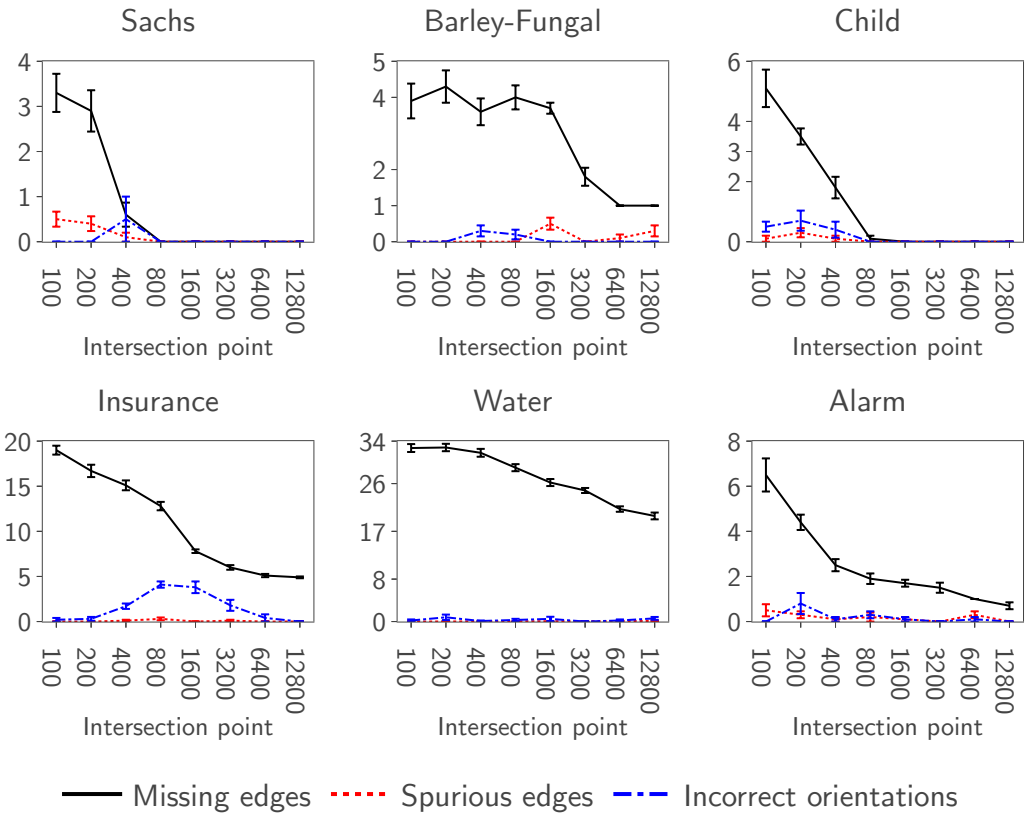


Figure 14: Agreement graph error types.

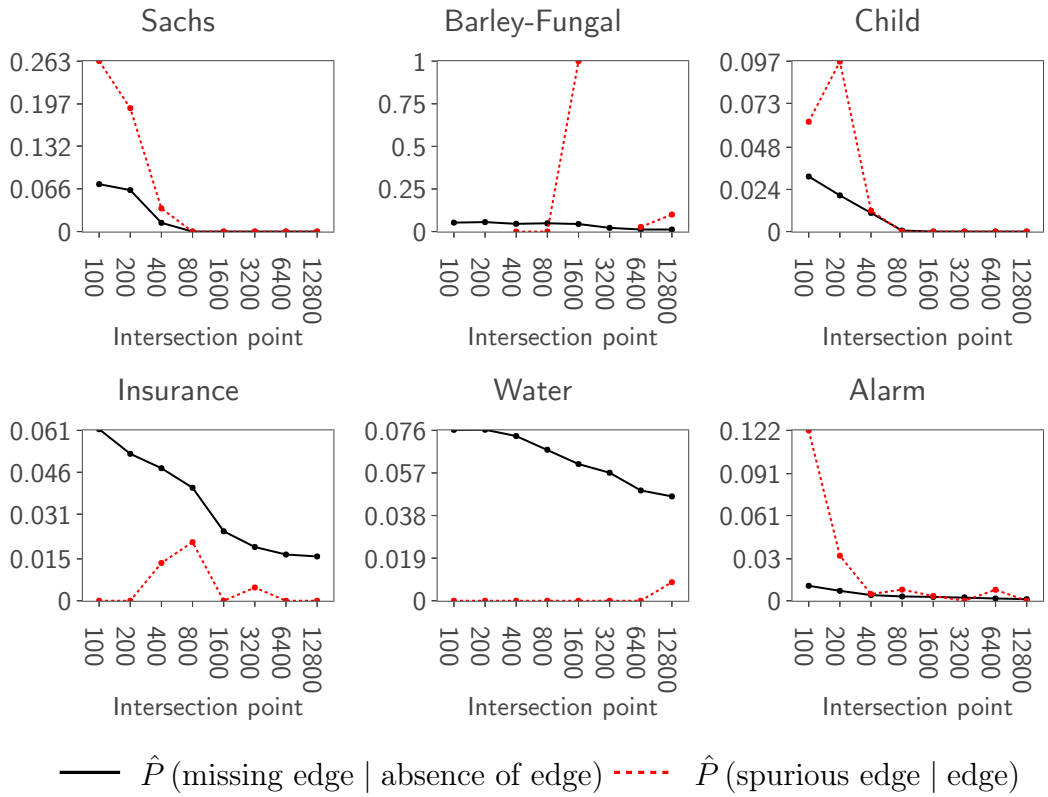


Figure 15: Agreement graph error type empirical probabilities. The missing values for spurious edge probabilities in `Barley-Fungal` are due to 0 connected node pairs in the agreement graphs.

jority both in the ground truth and in the agreement graphs. It is therefore hardly surprising, that they also form the majority of errors. To account for this and to consider the empirical probability that an edge type is indeed incorrectly agreed upon by the algorithms, Figure 15 plots the probability for an absence of an edge to be a missing edge and the probability of an edge to be a spurious one. The empirical probability for each error type is the sum of the incorrectly agreed node pairs of the given type in the agreement graphs, computed at a given intersection point, divided by the total sum of node pairs of the given type in the same.

The probabilities for both error types in general are low. Values above 0.10 are observable only at the lowest one or two intersection points for `Sachs` and `Alarm`, and one particular intersection point for `Barley-Fungal`, where the agreement graph seems to contain an error in its only edge. Across all the networks and intersection points up to 800 samples the probability for an absence of an edge to be a missing edge is 0.04 and the probability for an edge to be a spurious one is 0.02.

The analysis shows that the agreement graphs behave somewhat predictably, with nearly monotonous relationship between their size and the considered intersection points. The probabilites for errors are low, even if the agreement graph sizes are above 0.85 already at 100 samples (except for `Barley-Fungal` which starts at 0.7). The qualities of the agreement graphs therefore do not seem to present any obvious bottlenecks for the performance of intersection validation.

4.4 Performance with error-free agreement graphs of limited size

We would like to consider the relative effects the size and the correctness of the agreement graphs have on the performance of the method. One way to look at the question, is to evaluate the performance using agreement graphs of limited size that do not contain any errors. Repeating the performance evaluation using only those instances of runs of intersection-validation that result in agreement graphs without errors is not reasonable however, due to the small number of such runs (Figure 14). As an alternative we can evaluate the performance after first removing the errors from the agreement graphs. Accordingly, we repeat the analysis performed for Figure 11, with two exceptions:

1. The new agreement graphs are constructed by projecting the ground truth CPDAG onto the original agreement graph node pairs. Denoting the original agreement graph by $G_0 = (S, E_0)$ and the ground truth CPDAG by $G = (V, E)$ the new agreement graph then is defined as the pair $(S, S \cap E)$.

2. The structures compared against the agreement graphs are those computed on the original independent datasets, not on subsamples of the data that the original agreement graph was computed on.

The second exception is introduced to lessen the effect of sampling error on the results, and to focus the analysis on examining how much of the lack of performance might be attributable to the limited size of the agreement graph as opposed to errors in it. The experiment, however, suffers from a disconnect between any dataset and the set of node pairs chosen for the agreement graph. With these reservations, the results presented in Figure 16 show, that even when the size of the agreement graph is large and contains no errors, the performance might be relatively weak (e.g., `Alarm`

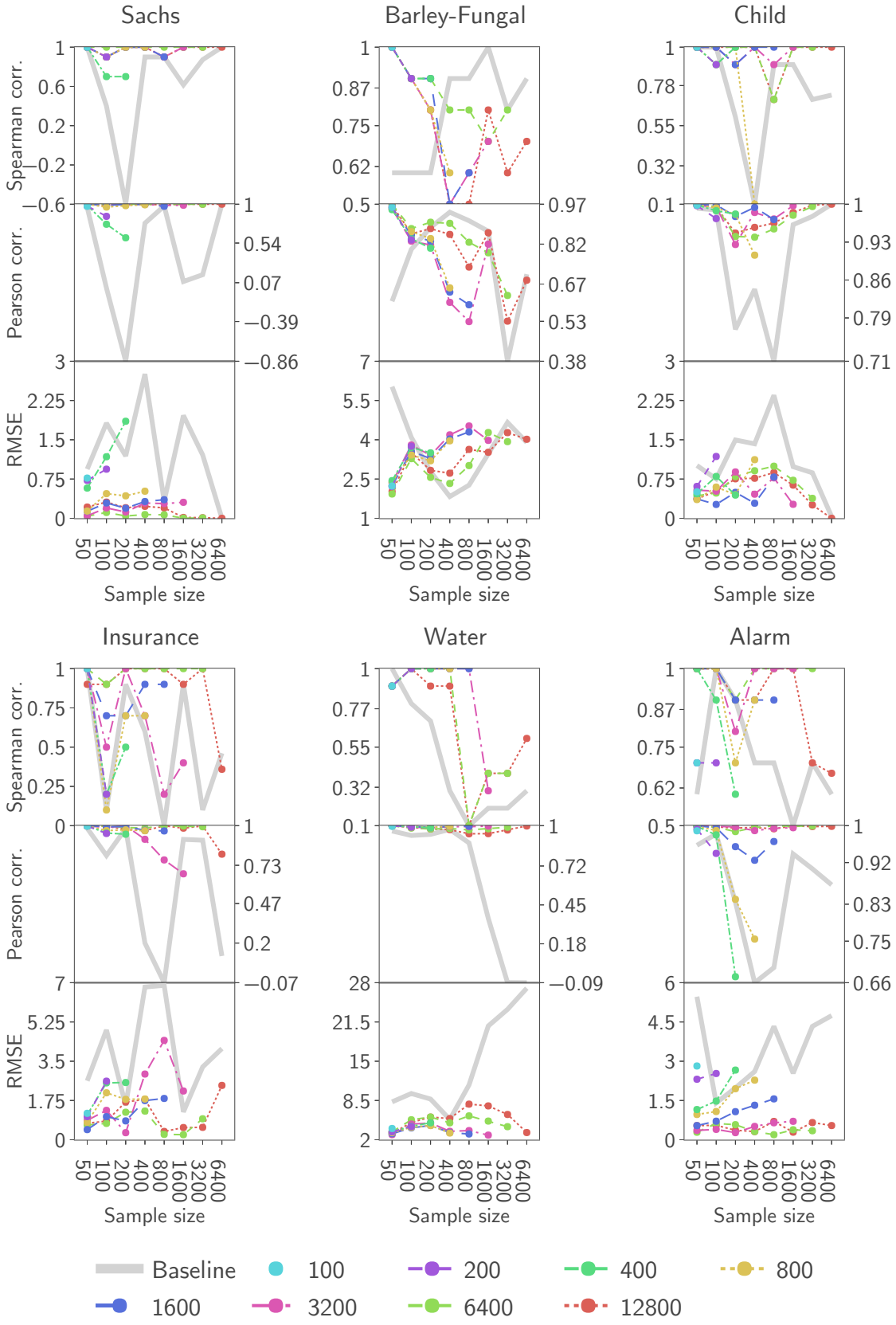


Figure 16: Three performance measures (Y-axis) given sample size (X-axis) and intersection-point (legend) computed identically to Figure 11, but with error-free agreement graphs.

at intersection-point 3200 and sample sizes 200 and 400) due to the node pairs not included in the agreement graph.

4.5 Predicting the performance of the method

As we have seen, intersection-validation performs rather well across the considered datasets. There is, however, some variation to the level of performance. To have confidence in the results when the method is applied on real-world data, we would like to know whether the variation can be explained by statistics computable from the observable data. We have also analyzed the properties of the agreement graphs both against the ground truth structure and in terms of what can be observed when only data is available. Focusing on the latter properties we will now try to predict the performance of intersection-validation.

4.5.1 Agreement graph size as predictor

We saw earlier, that the size of the agreement graph grows (somewhat monotonically for `Child`, `Insurance` and `Alarm` and less so for `Sachs`, `Barley-Fungal` and `Water`) with the intersection point (Figure 13). Also, the Spearman correlations between the SHD and PHD determined rankings over all the sample sizes do not seem to show any clear connection to the intersection-point (Figure 10). If the agreement graph size predicts the Spearman correlation between the PHD and SHD rankings, the predictive power thus does not seem to be strong.

The other two performance statistics – Pearson correlation and RMSE – have been calculated based on the SHD and PHD values averaged over the ten independent datasets, independently for each sample size (Figure 11). These statistics could then be meaningfully predicted only using the mean qualities of the agreement graphs, as each of the independent datasets produces a possibly differing agreement graph. Alternatively, to pair the performance statistics with the qualities of a single agreement graph, we can recompute the statistics based on the mean SHDs and mean PHDs obtained through a single run of intersection-validation.

We choose the latter alternative. The relevant elements in the analysis thus are

1. a "full dataset" of size s_0 given as input to intersection-validation and the associated agreement graph;

2. mean PHD between structures learned on 10 subsamples of size $s < s_0$ from the full dataset and the agreement graph;

3. mean SHD between structures learned on 10 independent datasets of size s sampled from the ground truth network, and the ground truth.

We will then consider whether some quality of the agreement graph (1) predicts a performance measure computed between (2) and (3). For $s_0 = 100$ the size of the set of possible values for s is $|\{50\}| = 1$, for $s_0 = 200$ the size is $|\{50, 100\}| = 2$ and so on, for a total of 36 combinations of intersection point and sample size, at which the performance measure can be computed. As there are 10 independent datasets of size s_0 we can perform the evaluation for a total of $10 \times 36 = 360$ times. Each evaluation outcome is finally paired with a quality of the agreement graph, whose predictive power we are interested in, to produce a scatter plot.

The analysis can be run for any combination of agreement graph quality (size, number of connected node pairs and mean pairwise operand distance) and performance measure (Pearson and Spearman correlations, RMSE). Figure 17 presents the results for predicting the Pearson correlation based on the agreement graph size, as it seemed to yield the strongest results out of the possible combinations. The results have a strong dependence on the sample size. To make more sense out of the scatterplot, the different samples sizes are coded with symbols of different shape and color.

It can be seen that, for some combinations of benchmark network and sample size, the agreement graph size seems to predict the method's performance rather well. This observation holds for example for `Insurance` at sample size 1600, and for `Alarm` at sample size 200. For some other combinations there does not seem to be any easily observable connection between the two quantities. To further quantify the analysis we can, again, compute the Pearson correlation between the agreement graph size and the performance measure at the various sample sizes. The results presented in Table 2 confirm the observation: though the size of the agreement graph does seem to predict the performance in many instances, in a lot of cases it also fails or even correlates negatively with the method's performance.

4.5.2 Mean absolute deviation as predictor

Previously we observed, that if the SHD values for the algorithms are close to each other the correlations between them and the corresponding PHD values degrade.

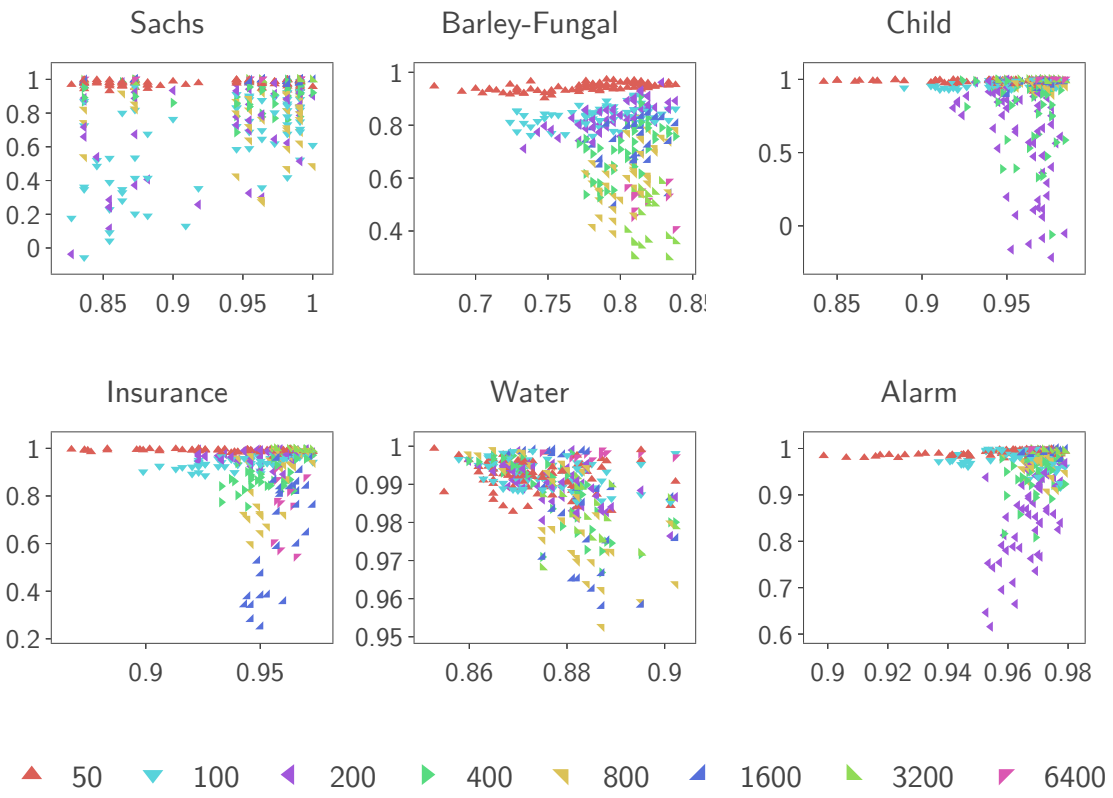


Figure 17: Pearson correlation between mean SHD and PHD (Y-axis) for varying agreement graph sizes (X-axis) at different sample sizes (legend).

If the same happens with the observable PHD values, we can hope to attach more confidence in the intersection-validation results when the values are more varied. Denoting the set of 5 mean PHD values, corresponding to each algorithm, of the

Table 2: The Pearson correlation between the agreement graph size and the Pearson correlation between the mean SHD and PHD values at different sample sizes. Correlations with absolute value of at least 0.7 in bold.

	50	100	200	400	800	1600	3200	6400
`Sachs`	0.19	0.59	0.49	0.00	-0.05	-0.18	0.05	0.43
`Barley-Fungal`	0.57	0.42	**0.70**	0.56	0.55	0.09	0.04	-0.18
`Child`	-0.25	0.35	-0.50	-0.32	-0.07	0.24	0.03	-0.03
`Insurance`	-0.43	0.67	0.17	0.61	0.56	**0.72**	0.00	0.05
`Water`	-0.14	-0.33	-0.65	**-0.75**	**-0.73**	-0.49	-0.17	-0.08
`Alarm`	0.63	-0.14	0.62	0.27	-0.06	0.44	-0.17	-0.37

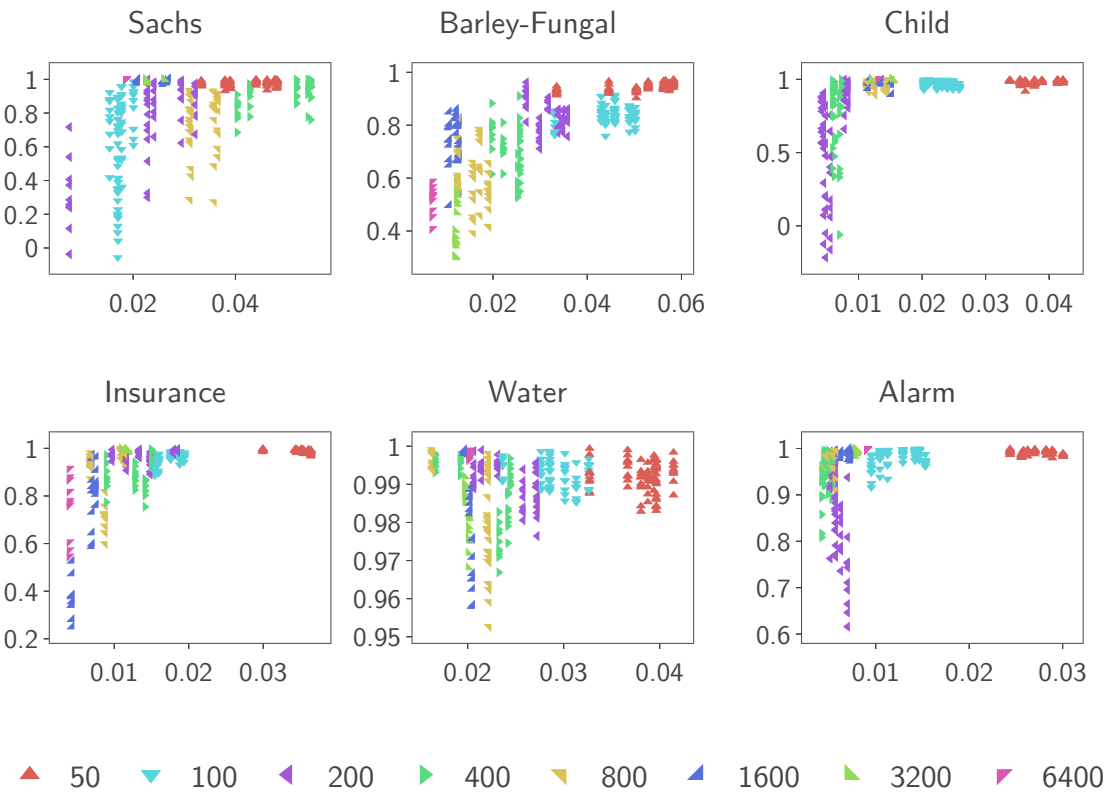


Figure 18: Pearson correlation between mean SHD and PHD (Y-axis) against mean absolute deviation between the PHDs divided by total amount of node pairs (X-axis) at different sample sizes (markers).

point (2) above as $\{d_i\}_{i=1}^5$ and the mean of them as $\bar{d}$, *the mean absolute deviation* (MAD) of the set is

$$\frac{1}{5}\sum_{i=1}^{5}\left|d_i - \bar{d}\right|.$$

Plotting the MAD (normalized with the total number of node pairs) against the Pearson correlation in Figure 18 shows, that there is a clear connection between the two quantities. Even though, for some of the networks, high correlation coefficients can be observed even when the MAD is close to 0, for all of the networks high MAD values (> 0.03) imply strong correlation.

Repeating the analysis with Spearman correlation (Figure 19) for the ranks of the algorithms reveals a similar pattern, though not as clearly. The only exception is in the instances of high MAD for `Alarm`, where the correlation is not as high as the pattern would seem to indicate, though still mostly above 0.6.

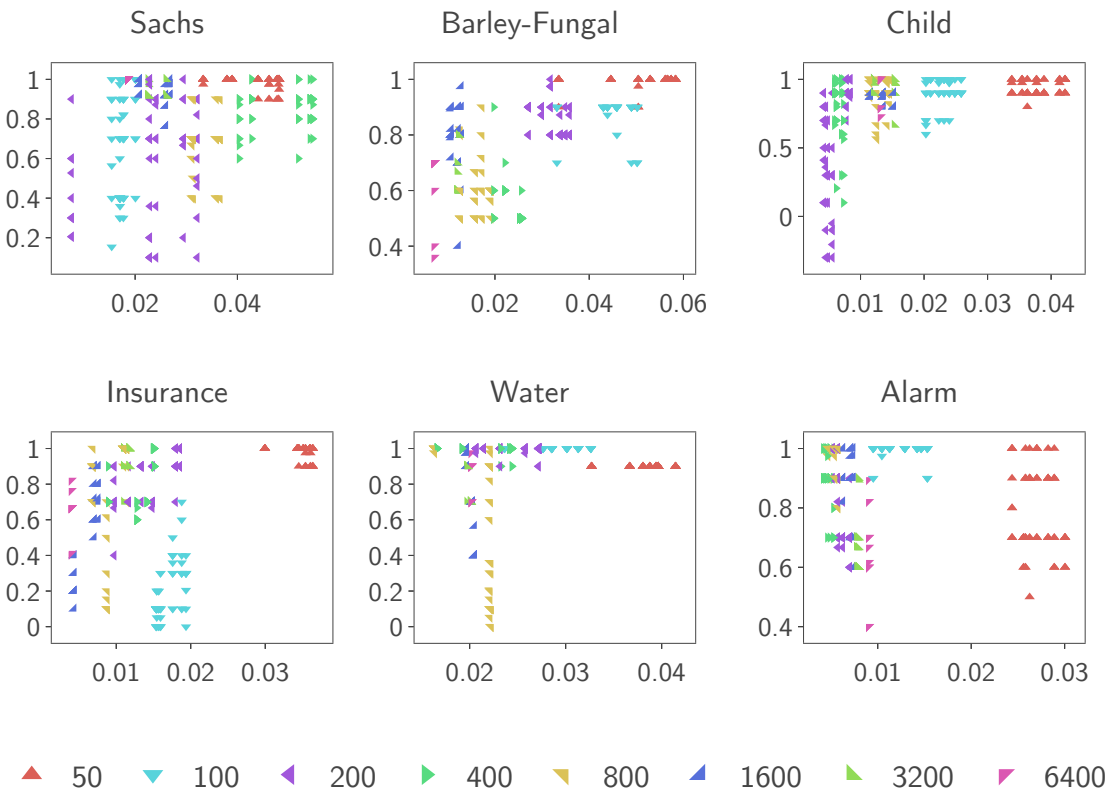


Figure 19: Spearman correlation between the ranks determined by the mean SHD and PHD (Y-axis) against mean absolute deviation between the PHDs divided by total amount of node pairs (X-axis) at different sample sizes (markers).

4.6 Pairwise comparisons

The experiments presented thus far have been based on using intersection-validation with the full set of considered scoring functions: AIC, BIC, BDeu, fNML and qNML. We have examined the effect the amount of data has on the performance of the method. In an analogous manner, we can evaluate the performance of intersection-validation when the amount of compared algorithms is limited. In the extreme case there is only two algorithms to compare. Out of the five scoring functions we can form ten pairs. Given an intersection point and a sample size we now have only two measurements of mean SHD and mean PHD. Evaluating the performance of intersection-validation in this setting with Pearson correlation therefore is not possible, because of zero standard deviations for the two correlates. We can, however, still calculate the fraction of matching rankings as determined by the two metrics.

Computing the ratio of matching rankings over all the parameters (the benchmark

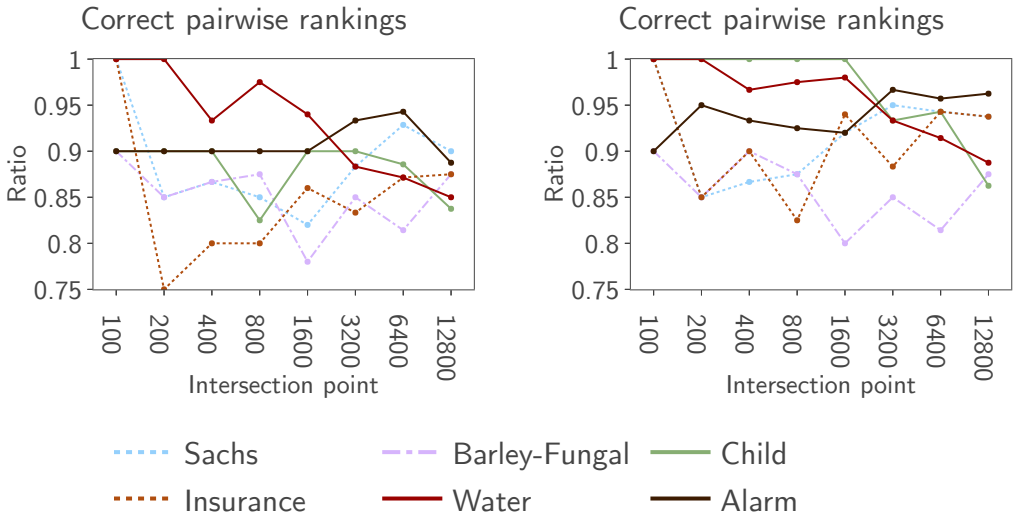


Figure 20: Ratio of matching SHD and PHD rankings when applying intersection-validation on pairs of scores. Plot on the left only considers the mean values for the metrics, while plot on the right ranks scores with overlapping standard errors of the means equally.

Bayesian networks, the pairs of scores, the intersection points and sample sizes), we find it to be as high as 0.880, a slight improvement over the 0.875 obtained when intersection-validation was applied on the full set of scores. When multiple scores were under analysis it was difficult to take the standard errors of the means into account when determining the rankings, except by noting that when they overlap they make the results more uncertain. In the case of compared score pairs the situation is somewhat clearer. To analyze how many of the ranking mismatches between SHD and PHD are in fact due to the scores being assigned very similar distances, we will next define the two metrics to produce identical rankings also in the case when the compared scores have overlapping standard errors given by both of the metrics. After the redefinition the accuracy improves to 0.920. The total ratio can be examined in finer resolution in multiple ways. Figure 20 plots the ratio per benchmark network and intersection point, similarly to Figure 10, for both of the discussed variants.

To consider the effect the specific scores have on the ratios we will now summarize the results on a per score pair basis in Table 3. The results indicate that there is no particular pair of scores, among the considered ones, using which intersection-validation would fail to reliably predict their SHD-based ranking. The accuracy

Table 3: Ratio of correct pairwise mean-based rankings (upper triangular) and when overlapping standard errors of the means are taken into account (lower triangular). Extreme values for both variants are in bold.

	AIC	BIC	BDeu	fNML	qNML
AIC		0.824	0.819	0.917	0.870
BIC	0.875		**0.810**	**0.963**	0.870
BDeu	0.870	**0.843**		0.917	0.829
fNML	0.949	**0.981**	0.963		0.958
qNML	0.944	0.917	0.880	0.972	

of the predictions varies, from 0.810 (0.843, when standard errors are taken into account) for the pair BIC-BDeu to 0.963 (0.981) for BIC-fNML. The differences in the accuracies seem attributable to the differences in the scores' performances in the SHD-based evaluation.

5 Case study: Structure priors

In the presented experiments, we have used scores mostly based on alternative ways to define the marginal likelihood component in the equation 6, ignoring the prior. The prior has in general received relatively little attention, motivating the need for the second paper that the thesis builds on (Eggeling et al. 2019). This section summarizes the paper as an example use case for intersection-validation.

5.1 Modular priors and search space penalty

In the paper four different sparsity promoting priors are considered and compared to the uniform prior, that is, to not setting a prior. Three of the priors are modular, meaning that they decompose, with some normalizing constant, according to the factorization

$$P(G) \propto \prod_{i=1}^{n} \rho_i(G_i),$$

where ρ_i are functions from the subsets of $[n] \setminus \{i\}$ (i.e., from the possible sets of parents for variable i) to non-negative reals. The modular priors considered in the paper are presented in Table 4.

Table 4: The uniform prior and three sparsity promoting modular priors. *Fair* is non-parametric, *Edge* is parameterized with p and *Data* with τ.

Name	Factor	Notes	Reference
Unif	1	Uniform over DAGs	–
Edge	$(p/(1-p))^{\lvert G_i\rvert}$	Edge probability p	Heckerman et al. (1995)
Fair	$1/\binom{n-1}{\lvert G_i\rvert}$	Balances indegrees	Friedman and Koller (2003)
Data	$\exp\bigl(-(1+\tau)^{\lvert G_i\rvert}\ln N\bigr)$	N is the data size	Pensar et al. (2016)

The names of the priors reflect the main ideas behind them: *Unif* is the usual uniform prior assigning equal probability to each DAG; *Edge* is equivalent to a random graph model where each edge is created with probability p, discarding graphs with directed cycles; *Fair* aims to allocate equal total probability for all indegrees; *Data* depends on the size of data.

Additionally, the paper presents a procedure that can be used to implement a sparsity promoting prior, even when a software used for structure discovery does not allow for setting any explicitly. The procedure, called the *search space penalty* (SSP), only depends on the possibility to set the maximum indegree for the search and that the score is returned together with the structure – requirements customarily satisfied by existing software.

Each possible value of maximum indegree specifies a space of DAGs. Let $\mathcal{G}_n^d$ denote such a space on DAGs of n nodes with a maximum indegree of d. Clearly, the spaces (i.e., sets) of increasing indegree are nested, with each successive set being a superset of the previous:

$$\mathcal{G}_n^0 \subset \mathcal{G}_n^1 \subset \cdots \subset \mathcal{G}_n^{n-1} = \mathcal{G}_n.$$

Let $d\,(G)$ be the maximum indegree of DAG G. The search space penalty is a non-modular prior to promote sparseness, by which the prior probability assigned to a structure is inversely proportional to the size of the smallest search space that it is a member of, that is, by setting

$$P_{\mathrm{SSP}}\,(G) \propto \frac{1}{\left|\mathcal{G}_n^{d(g)}\right|}.$$

To find a score maximizing structure under the prior, one can first find the score maximizing structures under the uniform prior separately for each indegree. The

SSP prior is then applied as a postprocessing step, by multiplying the scores of each found structure with the prior, and finally selecting the one with the highest resulting score. The normalizing constant factor in the prior, naturally, can be ignored if the interest is only in finding the highest scoring structure and not in the final score itself. The process, for some scoring function f, can be formalized in three simple steps as follows.

Algorithm *Search Space Penalization*

Step 1. For each indegree $d = 0, 1, \ldots, n-1$, let
$\hat{G}(d) \in \arg\max \left\{ f(G) : G \in \mathcal{G}_n^d \right\}$.
"Find a score maximizing DAG for each search space."

Step 2. Let
$\hat{d} \in \arg\max \left\{ f(\hat{G}(d))/|\mathcal{G}_n^d| : d = 0, 1, \ldots, n-1 \right\}$.
"Find a score maximizing indegree after applying SSP prior."

Step 3. Output $\hat{G}(\hat{d})$.

The output is the DAG G that maximizes $P_{\mathrm{SSP}}(G)\, f(G)$. In the first step one could limit the maximum indegree to some tractable number. Running the first step for each of the indegrees up to the selected maximum one results in longer computations than searching through a single search space. The running times, however, are in practice dominated by the largest indegree. The additional cost of running the algorithm, compared to search in a single space, thus seems acceptable. Also, the size of the search space, required in step 2, can be calculated efficiently with a recurrence formula provided in the paper.

5.2 Empirical studies

The main results of the paper are based on evaluating benchmark Bayesian network recovery, given datasets sampled from them. The benchmark networks are the same that we have seen before: `Alarm`, `Insurance`, `Child` and `Water`. Additionally, the paper includes experiments using the intersection-validation method on various real-world datasets. The datasets, preprocessed by Malone et al. (2018) and originating from the UCI machine learning repository (Dua and Graff 2017), represent measurements of various real-world quantities, from Turkish university course evaluation data, to hypothyroid disease data. The results for both experiments are shown in Figure 21, for the BDeu baseline score. The conclusions from the latter

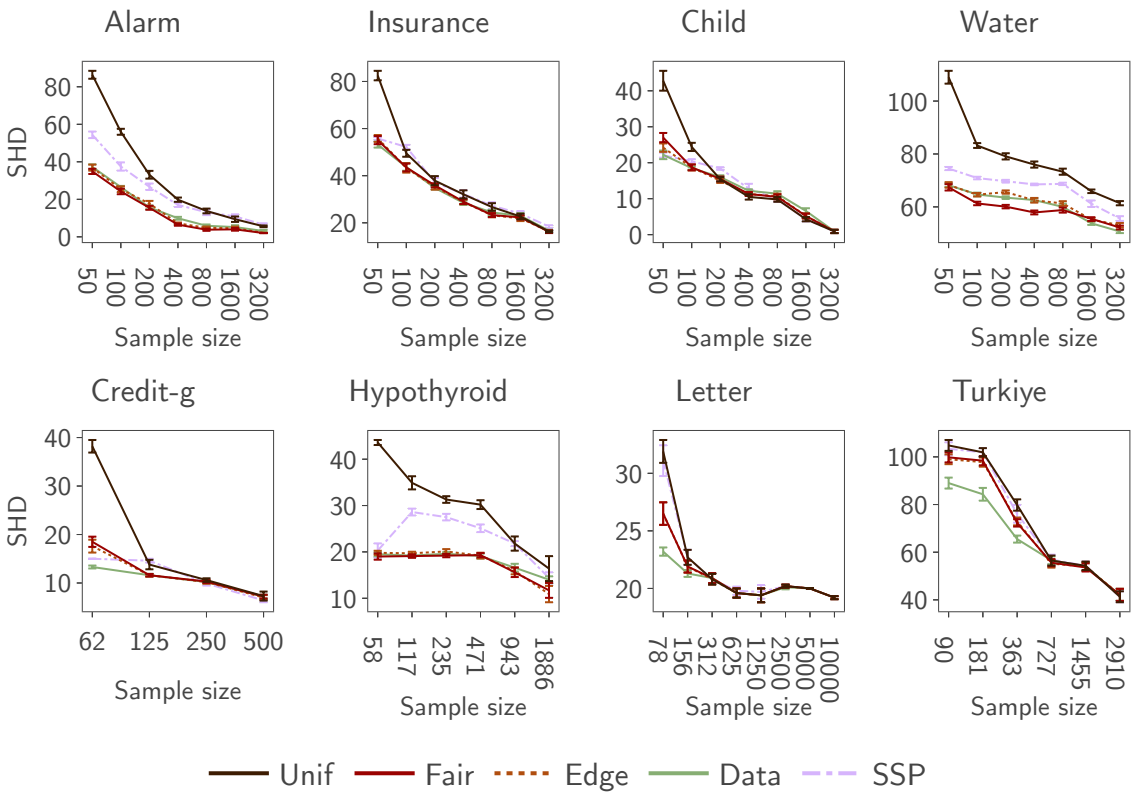


Figure 21: The effect of various priors on exact structure learning using BDeu score. Top row shows benchmark network recovery, while bottom row shows intersection-validation applied on a number of real-world datasets.

experiments confirm those of the former ones: the priors have a significant improving effect on structure learning, especially on low sample sizes, and the strength of the effect depends on the scoring function.

In the case of real-world data there is no ground truth structure, so we cannot measure how close the intersection-validation results are to the correct ones. However, the fact, that the conclusions drawn from using both synthetic and real data are the same, can be regarded as weak supportive evidence for the validity of intersection-validation.

6 Conclusions and discussion

The thesis has introduced intersection-validation, a method for evaluating the performance of algorithms that learn Bayesian network structures from data, without a

ground truth to compare the learned structures against. As a proxy for the ground truth, the method constructs an agreement graph from the features common to the structures learned by the evaluated input algorithms on an input dataset. Then the algorithms are evaluated against the agreement graph, given subsamples of the input data.

The empirical experiments tested the validity of the method by using it on data generated from ground truth networks. The approach allowed comparing the method's results to those obtained by directly evaluating the algorithms against the correct structure. The results provide moderately strong evidence for the method's validity, both for ranking the input algorithms correctly, and for estimating the (relative) distances between their outputs and the ground truth, at some predetermined sizes of subsampled data.

We considered five different scoring functions to specify the algorithms and data from six benchmark Bayesian networks. The benchmark networks, from which the experiment data were sampled, were chosen to represent different learning scenarios, where the properties of the networks vary considerably. The thesis has also included an experiment to study the performance of the method when there are only two input algorithms. Further, the experiments in the paper in which the method was first presented were based on yet another set of algorithms (Viinikka et al. 2018). Though no amount of experiments will provide irrefutable evidence that the method works in any theoretical sense, the positive results from these experiments would seem to indicate that the method can be applied with some confidence in a variety of settings.

Specifically, we have seen, that across the experiments the method gives a correct ranking for two algorithms (relative to each other) with an accuracy of approximately 0.9, including when the method is applied onto a set of only two algorithms. The Pearson correlations between the SHDs and PHDs measured for the set of algorithms vary a lot, depending on the benchmark network, intersection point and sample size at which the quantities are measured. In general, they seem fairly strong, and for the most part improve when the method is given increasing amounts of data as input. The results seem to exhibit three sample size regimes, where at the lowest sample size the method performs well, after which the performance degrades, until finally improving again. We compared the results to the baseline obtained by computing the correlations between the SHDs and the cross-entropies approximated via cross-validation – another evaluation approach not relying on knowing the ground truth.

The results revealed that intersection-validation mostly improves on the baseline and never fails as bad as the baseline on occasions does.

We have also attempted to predict the performance of the method based on quantities observable when the method is applied in practice. The size of the agreement graph (defined as the fraction of node pairs in it out of the total) is a straightforward and intuitive potential predictor. Though in some cases it does predict the Pearson correlation, in many other cases it does not. Additionally, we examined whether greater differences in the PHDs measured for the the compared algorithms allow for greater confidence in the results, and found the answer to be affirmative.

The method can, however, also give misleading results, even when the measured distances differ greatly. As there are no theoretical guarantees as to when the method works, and predicting the method's failure or success from observable data does not seem to be easy, one might then want to consider alternatives.

An obvious alternative, when evaluating the structure learning performance of an algorithm, is to rely on studies using benchmark Bayesian networks. The problem with the approach is that it leads to disproportionate importance given to the performance evaluated on specific networks, as there are not many such networks easily available. To avoid the problem of scarcity, one can sample Bayesian networks synthetically (Melançon et al. 2001, Ide and Cozman 2002). In both approaches the problem, however, remains that the data sampled from the networks is synthetic, even if in the first alternative the network might have been created to model some real-world phenomenon. The ultimate interest in developing structure discovery algorithms, however, remains in their performance given real-world data – despite the inherent difficulty of defining such performance. Evaluating them on synthetic data thus might yield misleading results.

The method as presented relies on the structural Hamming distance, and the extension presented in the thesis, the partial Hamming distance. Any conclusions regarding the performance of intersection-validation thus are subject to the used metrics. Though popular, the SHD might not be without problems. For one, it has a very local view to the differences in the graphs between which the distance is measured. Equal importance is given to the differences in every part of the operand graphs, though changes between certain node pairs could be argued in some senses to be more consequential than others. The *structural intervention distance*, which examines the causal implications of the differences between the graphs, for example, has a more global approach (Peters and Bühlmann 2015). It is, however, the local

and decomposable nature of the SHD that allowed for the metric's straightforward extension to the partial graphs that the method is fundamentally based on. Some future directions for the study of intersection-validation could be found by examining whether some other metrics are similarly extendable to be used with the method.

Asymptotically, with reasonable assumptions, intersection-validation produces results identical to ground truth based evaluation. A fundamental open question remains whether guarantees on the accuracy of the results can be found assuming finite data.

References

H. Akaike. A new look at the statistical model identification. *IEEE Transactions on Automatic Control*, 19(6):716–723, December 1974.

A. V. Arkhangel'skiĭ and V. V. Fedorchuk. *The Basic Concepts and Constructions of General Topology*. Springer Berlin Heidelberg, Berlin, Heidelberg, 1990.

M. Barlett and J. Cussens. Advances in Bayesian network learning using integer programming. In *Proceedings of the Twenty-Ninth Conference on Uncertainty in Artificial Intelligence*, UAI'13, pages 182–191, Arlington, Virginia, United States, 2013. AUAI Press.

P. Beek and H.-F. Hoffmann. Machine learning of Bayesian networks using constraint programming. In *Proceedings of the 21st International Conference on Principles and Practice of Constraint Programming - Volume 9255*, pages 429–445, New York, NY, USA, 2015. Springer-Verlag New York, Inc.

I. A. Beinlich, H. J. Suermondt, R. M. Chavez, and G. F. Cooper. The ALARM monitoring system: A case study with two probabilistic inference techniques for belief networks. In J. Hunter, J. Cookson, and J. Wyatt, editors, *AIME 89*, pages 247–256, Berlin, Heidelberg, 1989. Springer Berlin Heidelberg.

J. Binder, D. Koller, S. Russell, and K. Kanazawa. Adaptive probabilistic networks with hidden variables. *Machine Learning*, 29(2):213–244, Nov 1997.

W. Buntine. Theory refinement on Bayesian networks. In *Proceedings of the Seventh Conference on Uncertainty in Artificial Intelligence*, UAI'91, pages 52–60, San Francisco, CA, USA, 1991. Morgan Kaufmann Publishers Inc.

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

D. M. Chickering. *Learning Bayesian Networks is NP-Complete*, pages 121–130. Springer New York, New York, NY, 1996.

D. M. Chickering. Learning equivalence classes of Bayesian-network structures. *J. Mach. Learn. Res.*, 2:445–498, Mar. 2002.

G. F. Cooper and E. Herskovits. A Bayesian method for the induction of probabilistic networks from data. *Mach. Learn.*, 9(4):309–347, Oct. 1992.

J. Cussens. Bayesian network learning with cutting planes. In *Proceedings of the Twenty-Seventh Conference on Uncertainty in Artificial Intelligence*, UAI'11, pages 153–160, Arlington, Virginia, United States, 2011. AUAI Press.

D. Dua and C. Graff. UCI machine learning repository, 2017. URL `http://archive.ics.uci.edu/ml`.

R. Eggeling, J. Viinikka, A. Vuoksenmaa, and M. Koivisto. On structure priors for learning Bayesian networks. In K. Chaudhuri and M. Sugiyama, editors, *Proceedings of Machine Learning Research*, volume 89 of *Proceedings of Machine Learning Research*, pages 1687–1695. PMLR, 16–18 Apr 2019.

N. Friedman and D. Koller. Being Bayesian about network structure. a Bayesian approach to structure discovery in Bayesian networks. *Machine Learning*, 50(1): 95–125, 2003.

R. L. Graham, D. E. Knuth, and O. Patashnik. *Concrete Mathematics: A Foundation for Computer Science*. Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA, 2nd edition, 1994.

P. D. Grünwald. *The Minimum Description Length Principle (Adaptive Computation and Machine Learning)*. The MIT Press, 2007.

A. Gut. *An Intermediate Course in Probability*. Springer Publishing Company, Incorporated, 2nd edition, 2009.

R. Hamming. Error Detecting and Error Correcting Codes. *Bell System Techincal Journal*, 29:147–160, 1950.

D. Heckerman, D. Geiger, and D. M. Chickering. Learning Bayesian networks: The combination of knowledge and statistical data. *Machine Learning*, 20(3):197–243, 1995.

J. S. Ide and F. G. Cozman. Random generation of Bayesian networks. In *Proceedings of the 16th Brazilian Symposium on Artificial Intelligence: Advances in Artificial Intelligence*, SBIA '02, pages 366–375, Berlin, Heidelberg, 2002. Springer-Verlag.

C. S. Jensen, U. Kjærulff, and A. Kong. Blocking Gibbs sampling in very large probabilistic expert systems. *International Journal of Human-Computer Studies*, 42:647–666, 1995.

F. V. Jensen, U. Kjærulff, K. G. Olesen, and J. Pedersen. Et forprojekt til et ekspertsystem for drift af spildevandsrensning (an expert system for control of waste water treatment—a pilot project). *Technical Report*, 1989.

M. Koivisto and K. Sood. Exact Bayesian structure discovery in Bayesian networks. *J. Mach. Learn. Res.*, 5:549–573, Dec. 2004.

D. Koller and N. Friedman. *Probabilistic Graphical Models: Principles and Techniques - Adaptive Computation and Machine Learning*. The MIT Press, 2009.

P. Kontkanen and P. Myllymäki. A linear-time algorithm for computing the multinomial stochastic complexity. *Inf. Process. Lett.*, 103(6):227–233, Sept. 2007.

P. Kontkanen, W. Buntine, P. Myllymäki, J. Rissanen, and H. Tirri. Efficient computation of stochastic complexity. In *Proceedings of the Ninth International Conference on Artificial Intelligence and Statistics*, pages 233–238, 2003.

K. Kristensen and I. A. Rasmussen. The use of a Bayesian network in the design of a decision support system for growing malting barley without use of pesticides. *Computers and Electronics in Agriculture*, 33(3):197 – 217, 2002.

S. Kullback and R. A. Leibler. On information and sufficiency. *Ann. Math. Statist.*, 22(1):79–86, 03 1951.

S. L. Lauritzen. *Graphical Models*. Oxford University Press, 1996.

B. Malone, K. Kangas, M. Järvisalo, M. Koivisto, and P. Myllymäki. Predicting the hardness of learning Bayesian networks. In *Proceedings of the Twenty-Eighth AAAI Conference on Artificial Intelligence*, AAAI'14, pages 2460–2466. AAAI Press, 2014.

B. Malone, K. Kangas, M. Järvisalo, M. Koivisto, and P. Myllymäki. Empirical hardness of finding optimal Bayesian network structures: Algorithm selection and runtime prediction. *Mach. Learn.*, 107(1):247–283, Jan. 2018.

C. Meek. Causal inference and causal explanation with background knowledge. In *Proceedings of the Eleventh Conference on Uncertainty in Artificial Intelligence*, UAI'95, pages 403–410, San Francisco, CA, USA, 1995. Morgan Kaufmann Publishers Inc.

G. Melançon, I. Dutour, and M. Bousquet-Mélou. Random generation of directed acyclic graphs. *Electronic Notes in Discrete Mathematics*, 10:202–207, 2001.

K. P. Murphy. *Machine learning: a probabilistic perspective*. MIT Press, Cambridge, Mass. [u.a.], 2013.

K. Ng, G. Tian, and M. Tang. *Dirichlet and Related Distributions: Theory, Methods and Applications*. Wiley Series in Probability and Statistics. Wiley, 2011.

F. W. J. Olver, D. W. Lozier, R. F. Boisvert, and C. W. Clark. *The NIST Handbook of Mathematical Functions*. Cambridge Univ. Press, 2010.

J. Pearl. Bayesian networks: A model of self-activated memory for evidential reasoning. In *Proceedings of the 7th Conference of the Cognitive Science Society, University of California, Irvine*, pages 329–334, 1985.

J. Pearl. *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 1988.

J. Pearl and T. Verma. A theory of inferred causation. In *Proceedings of the Second International Conference on Principles of Knowledge Representation and Reasoning*, KR'91, pages 441–452, San Francisco, CA, USA, 1991. Morgan Kaufmann Publishers Inc.

J. M. Peña. Finding consensus Bayesian network structures. *Journal of Artificial Intelligence Research*, 42:661–687, 2011.

J. Pensar, H. Nyman, J. Lintusaari, and J. Corander. The role of local partial independence in learning of Bayesian networks. *Int. J. Approx. Reasoning*, 69(C): 91–105, Feb. 2016.

E. Perrier, S. Imoto, and S. Miyano. Finding optimal Bayesian network given a super-structure. *Journal of Machine Learning Research*, 9(Oct):2251–2286, 2008.

J. Peters and P. Bühlmann. Structural intervention distance for evaluating causal graphs. *Neural Computation*, 27(3):771–799, Mar. 2015.

J. Rissanen. Modeling by shortest data description. *Automatica*, 14:465–471, 1978.

S. Russell and P. Norvig. *Artificial Intelligence: A Modern Approach*. Prentice Hall Press, Upper Saddle River, NJ, USA, 3rd edition, 2009.

K. Sachs, O. Perez, D. Pe'er, D. A. Lauffenburger, and G. P. Nolan. Causal protein-signaling networks derived from multiparameter single-cell data. *Science*, 308 (5721):523–529, 2005.

G. Schwarz. Estimating the dimension of a model. *The Annals of Statistics*, 6(2): 461–464, 03 1978.

T. Silander and P. Myllymäki. A simple approach for finding the globally optimal Bayesian network structure. In *Proceedings of the Twenty-Second Conference on Uncertainty in Artificial Intelligence*, UAI'06, pages 445–452, Arlington, Virginia, United States, 2006. AUAI Press.

T. Silander, T. Roos, P. Kontkanen, and P. Myllymäki. Factorized normalized maximum likelihood criterion for learning Bayesian network structures. In M. Jaeger and T. D. Nielsen, editors, *Proceedings of the Fourth European Workshop on Probabilistic Graphical Models*, pages 257–264, 2008.

T. Silander, J. Leppä-aho, E. Jääsaari, and T. Roos. Quotient normalized maximum likelihood criterion for learning Bayesian network structures. In A. Storkey and F. Perez-Cruz, editors, *Proceedings of the Twenty-First International Conference on Artificial Intelligence and Statistics*, volume 84 of *Proceedings of Machine Learning Research*, pages 948–957, Playa Blanca, Lanzarote, Canary Islands, 09–11 Apr 2018. PMLR.

D. J. Spiegelhalter and S. L. Lauritzen. Sequential updating of conditional probabilities on directed graphical structures. *Networks*, 20:579–605, 1990.

D. J. Spiegelhalter, A. P. Dawid, S. L. Lauritzen, and R. G. Cowell. Bayesian analysis in expert systems. *Statist. Sci.*, 8(3):219–247, 08 1993.

P. Spirtes, C. Glymour, and R. Scheines. *Causation, Prediction, and Search*. MIT press, 2nd edition, 2000.

M. Teyssier and D. Koller. Ordering-based search: A simple and effective algorithm for learning Bayesian networks. *Proceedings of the 21st Conference on Uncertainty in Artificial Intelligence, UAI 2005*, 2005.

I. Tsamardinos, L. E. Brown, and C. F. Aliferis. The max-min hill-climbing Bayesian network structure learning algorithm. *Machine Learning*, 65(1):31–78, 2006.

T. Verma and J. Pearl. An algorithm for deciding if a set of observed independencies has a causal explanation. In *Proceedings of the Eighth International Conference on Uncertainty in Artificial Intelligence*, UAI'92, pages 323–330, San Francisco, CA, USA, 1992. Morgan Kaufmann Publishers Inc.

J. Viinikka, R. Eggeling, and M. Koivisto. Intersection-validation: A method for evaluating structure learning without ground truth. In A. Storkey and F. Perez-Cruz, editors, *Proceedings of the Twenty-First International Conference on Artificial Intelligence and Statistics*, volume 84 of *Proceedings of Machine Learning Research*, pages 1570–1578, Playa Blanca, Lanzarote, Canary Islands, 09–11 Apr 2018. PMLR.

C. Yuan and B. Malone. Learning optimal Bayesian networks: A shortest path perspective. *J. Artif. Int. Res.*, 48(1):23–65, Oct. 2013.

Appendix 1. SHD vs. PHD comparisons for all intersection-points

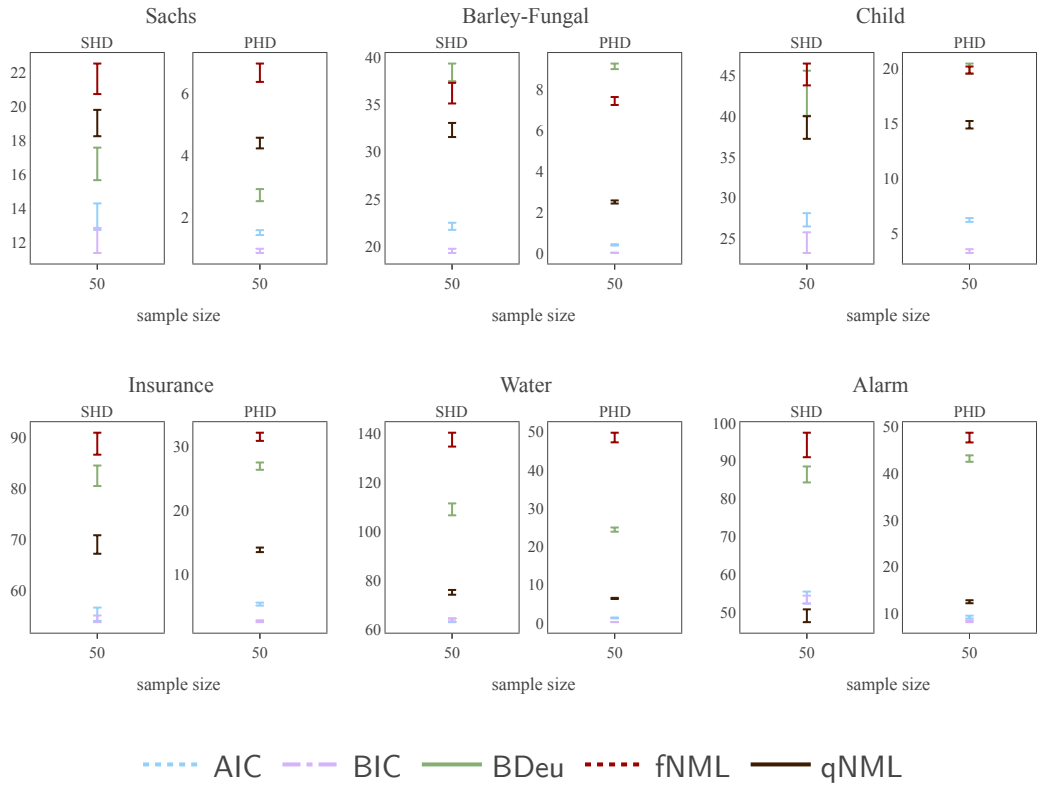


Figure 22: SHD (left) vs. PHD (right), with intersection point set to 100.

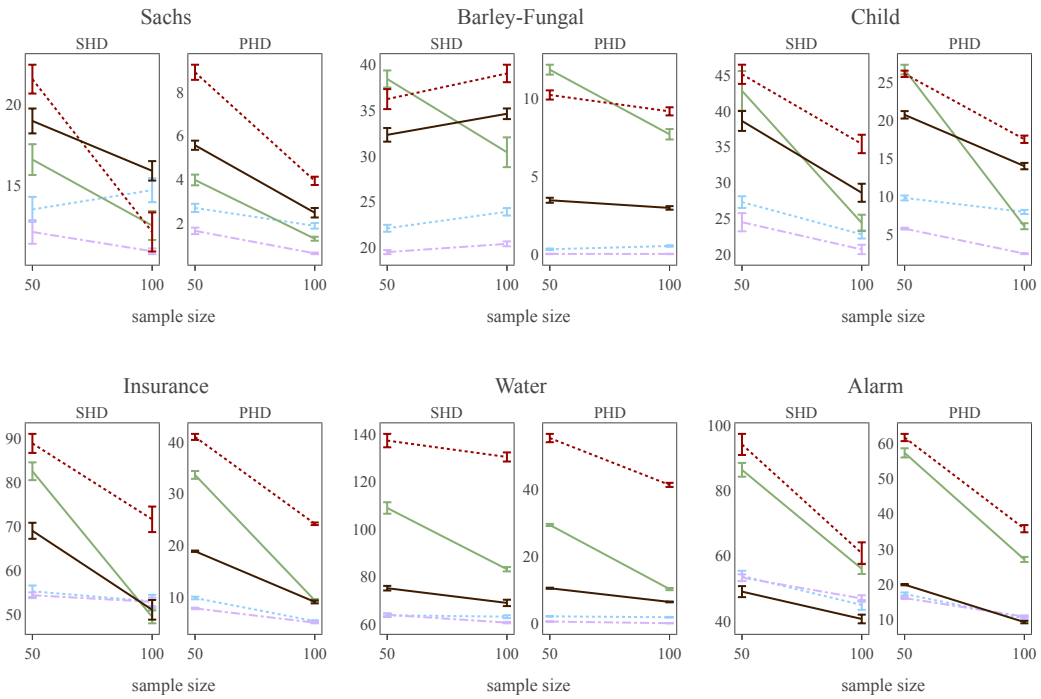


Figure 23: SHD (left) vs. PHD (right), with intersection point set to 200.

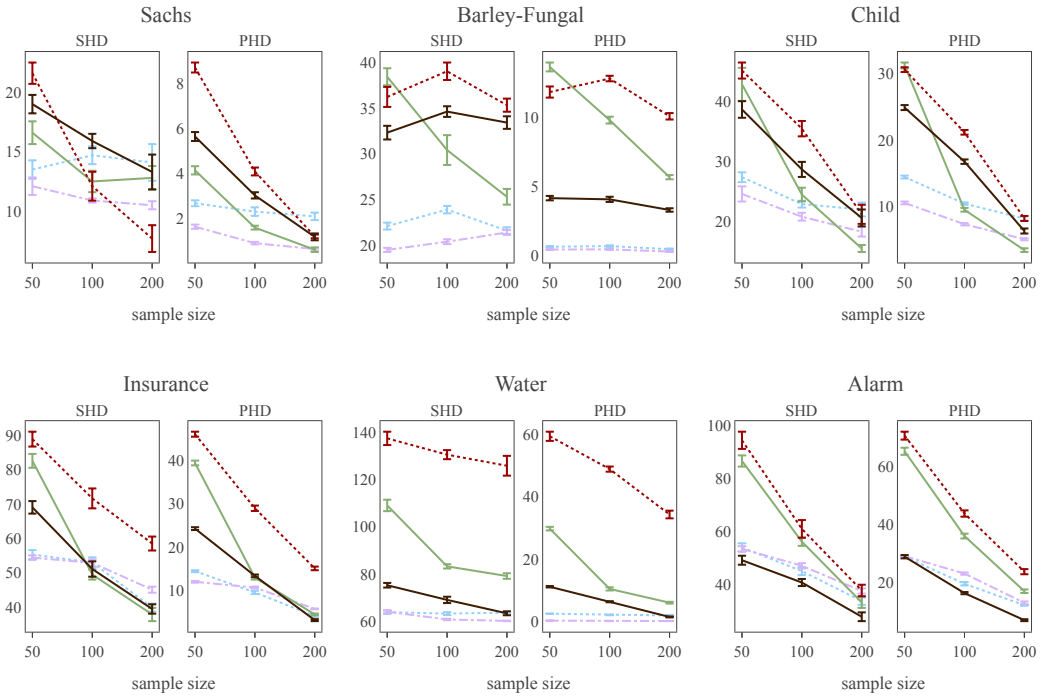


Figure 24: SHD (left) vs. PHD (right), with intersection point set to 400.

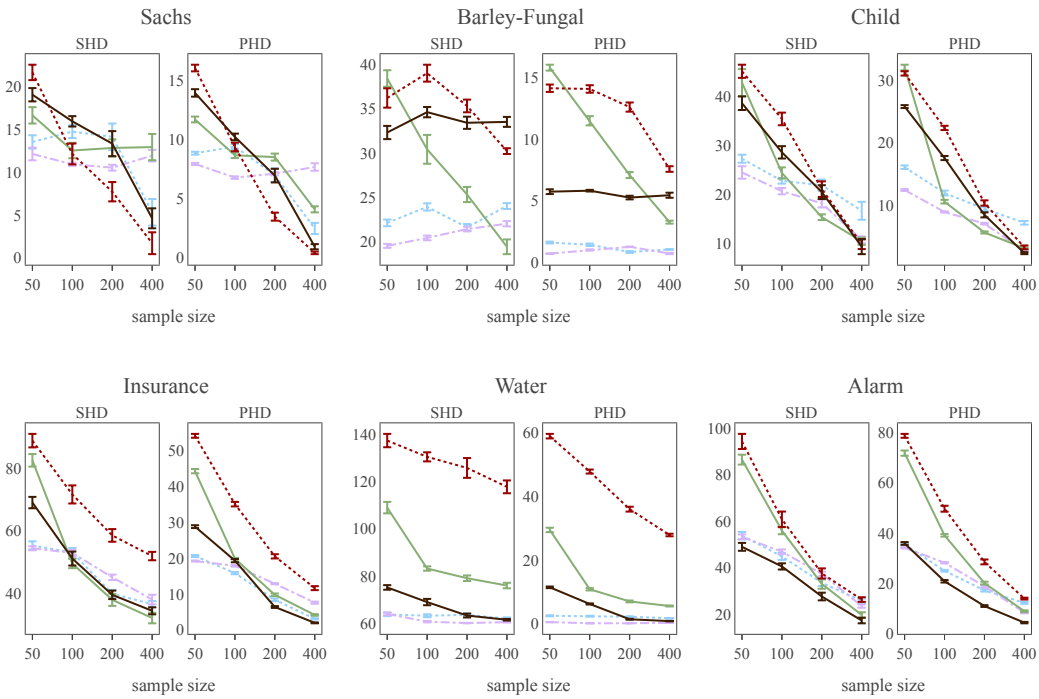


Figure 25: SHD (left) vs. PHD (right), with intersection point set to 800.

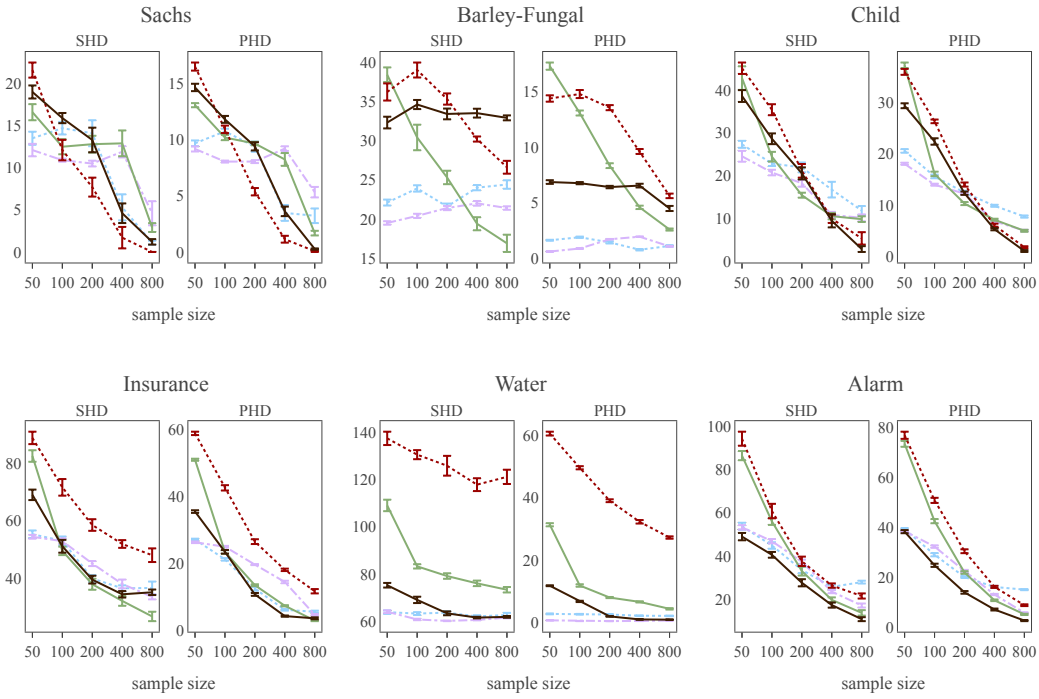


Figure 26: SHD (left) vs. PHD (right), with intersection point set to 1600.

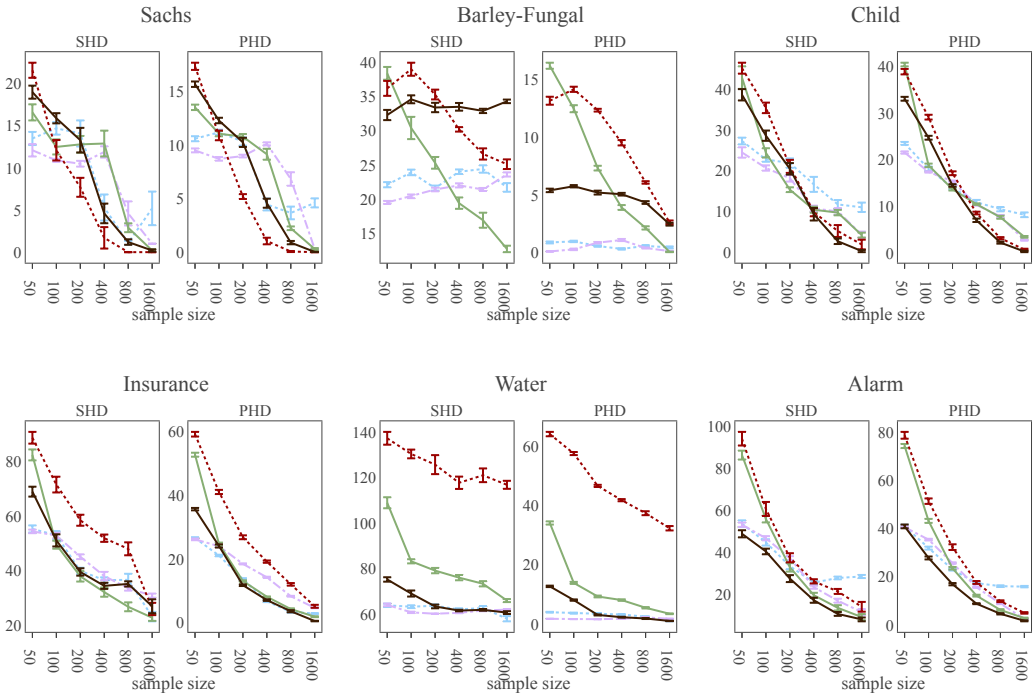


Figure 27: SHD (left) vs. PHD (right), with intersection point set to 3200.

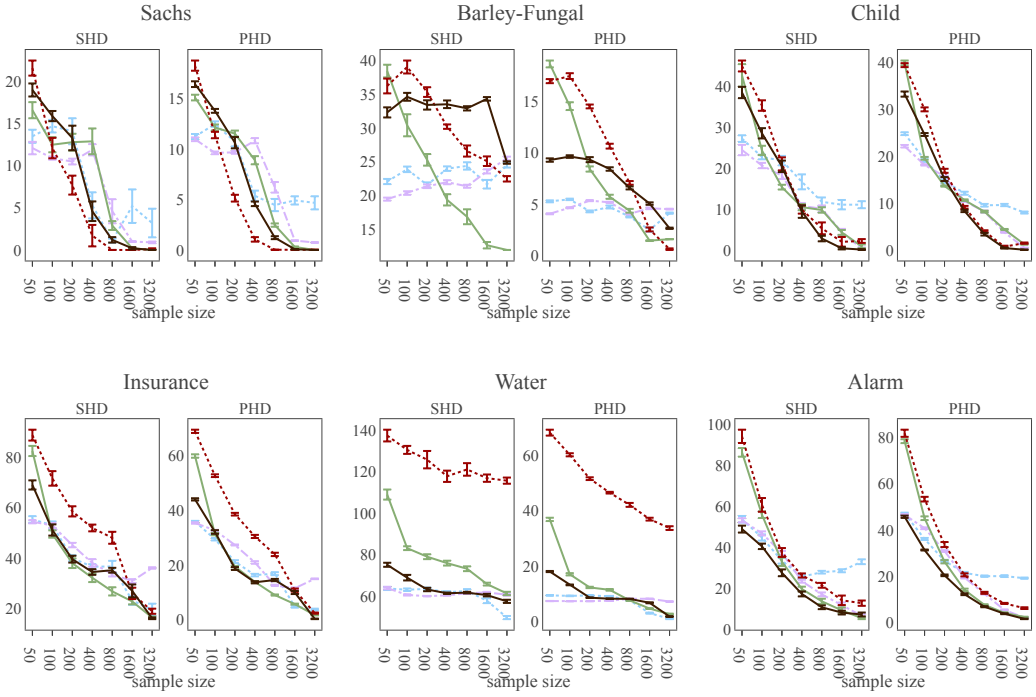


Figure 28: SHD (left) vs. PHD (right), with intersection point set to 6400.

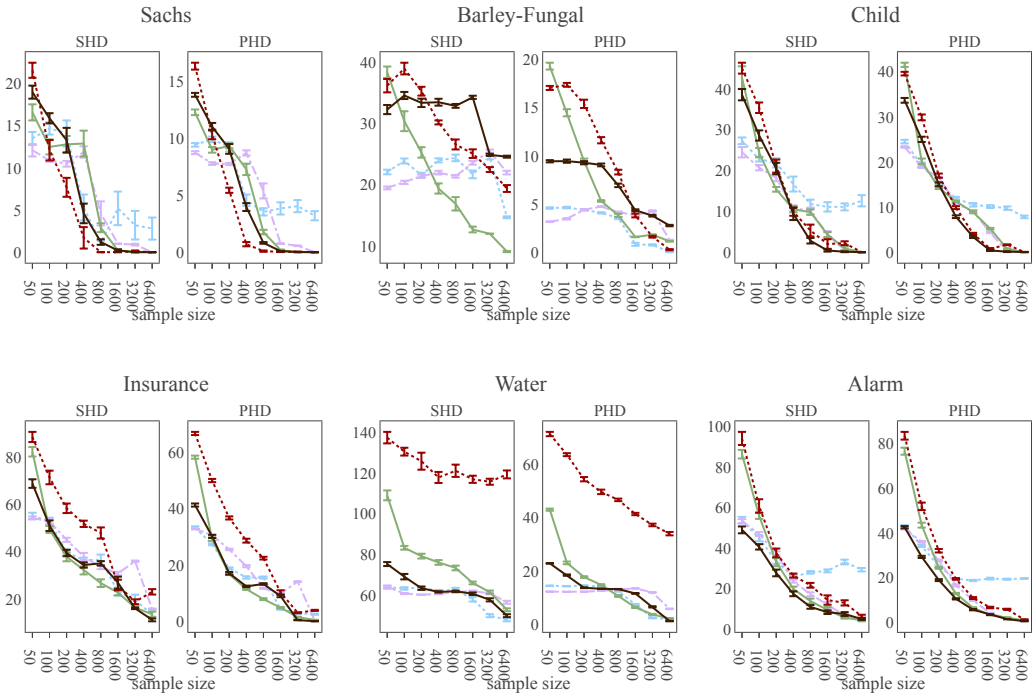


Figure 29: SHD (left) vs. PHD (right), with intersection point set to 12800.

Appendix 2. Predicting performance with all combinations of performance measure and predictor

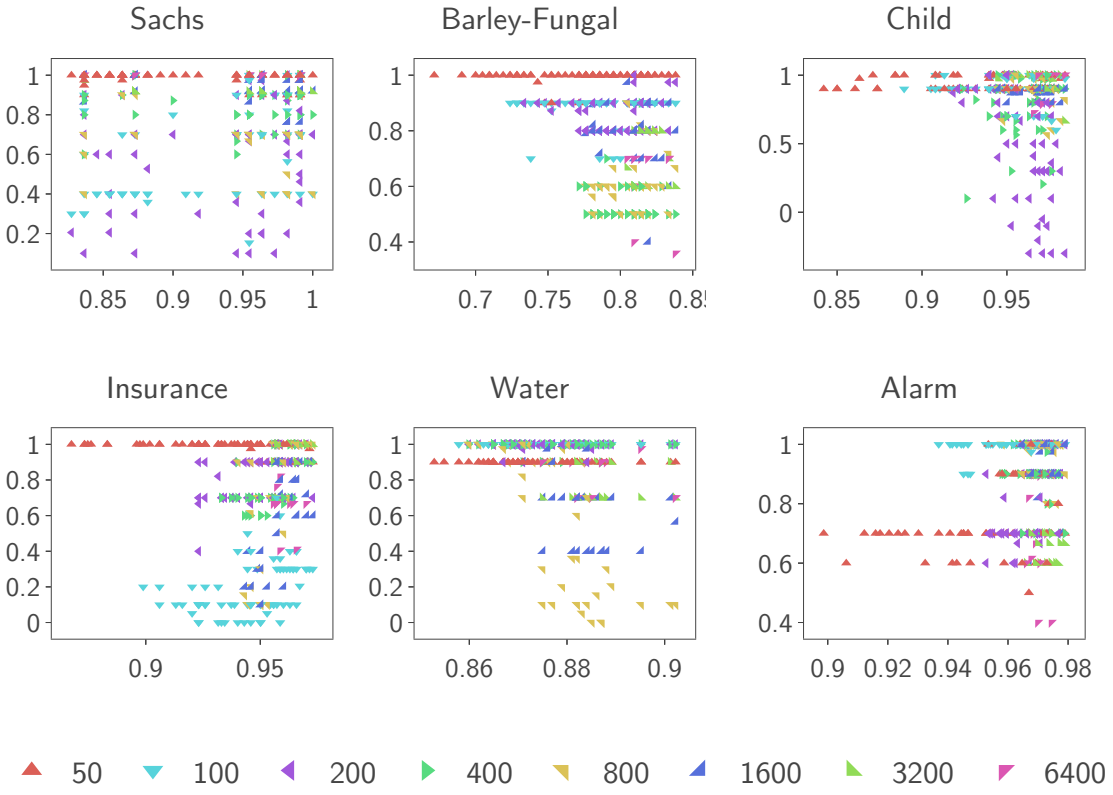


Figure 30: Spearman correlation (Y-axis), size of agreement graph (X-axis)

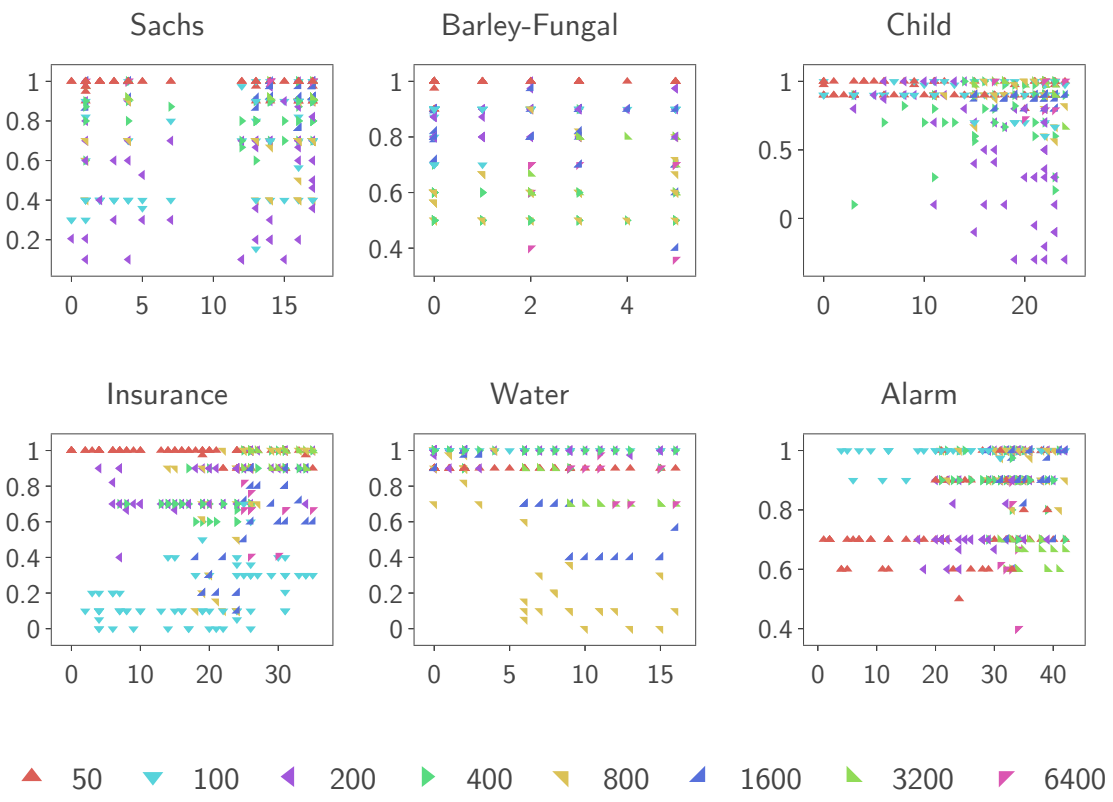


Figure 31: Spearman correlation (Y-axis), connected node pairs (X-axis)

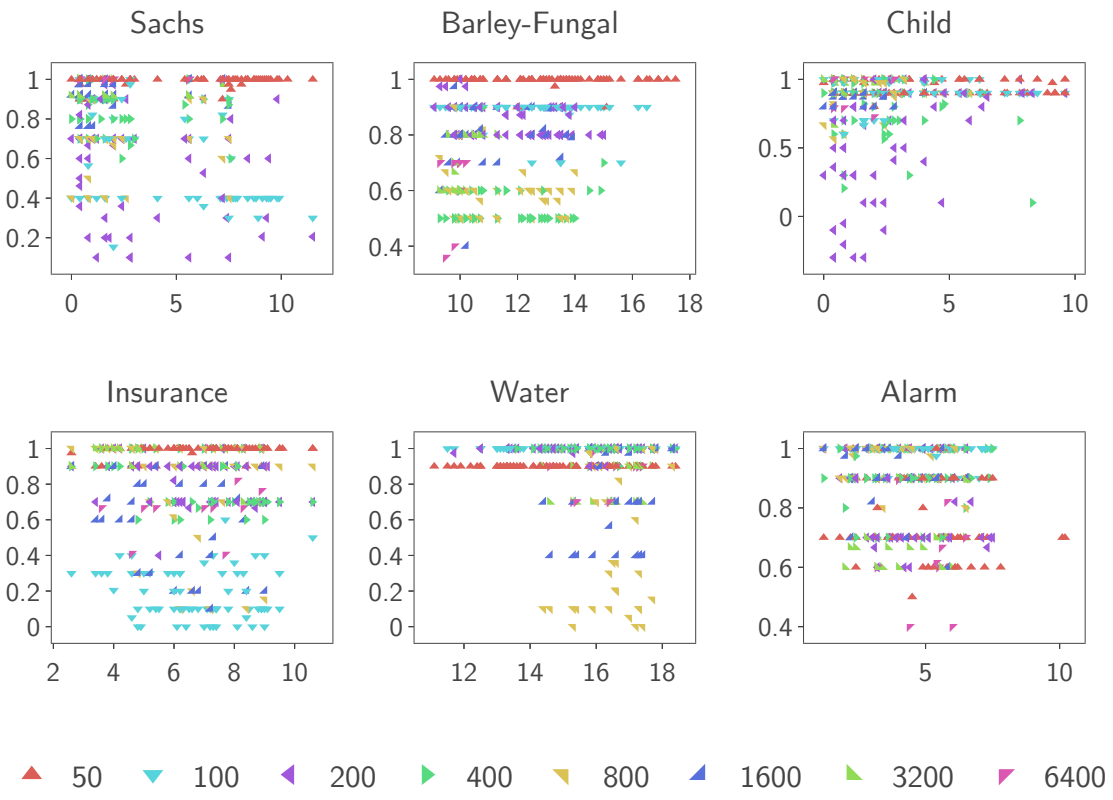


Figure 32: Spearman correlation (Y-axis), mean pairwise agreement graph operand distance (X-axis)

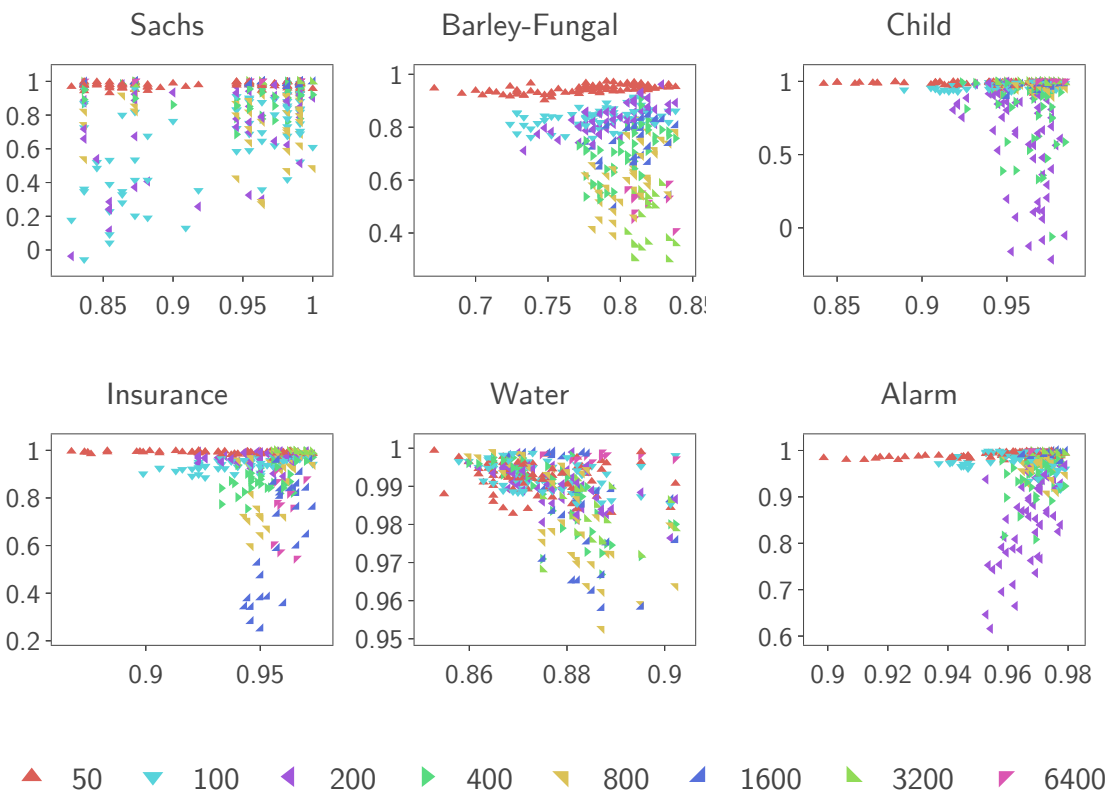


Figure 33: Pearson correlation (Y-axis), size of agreement graph (X-axis)

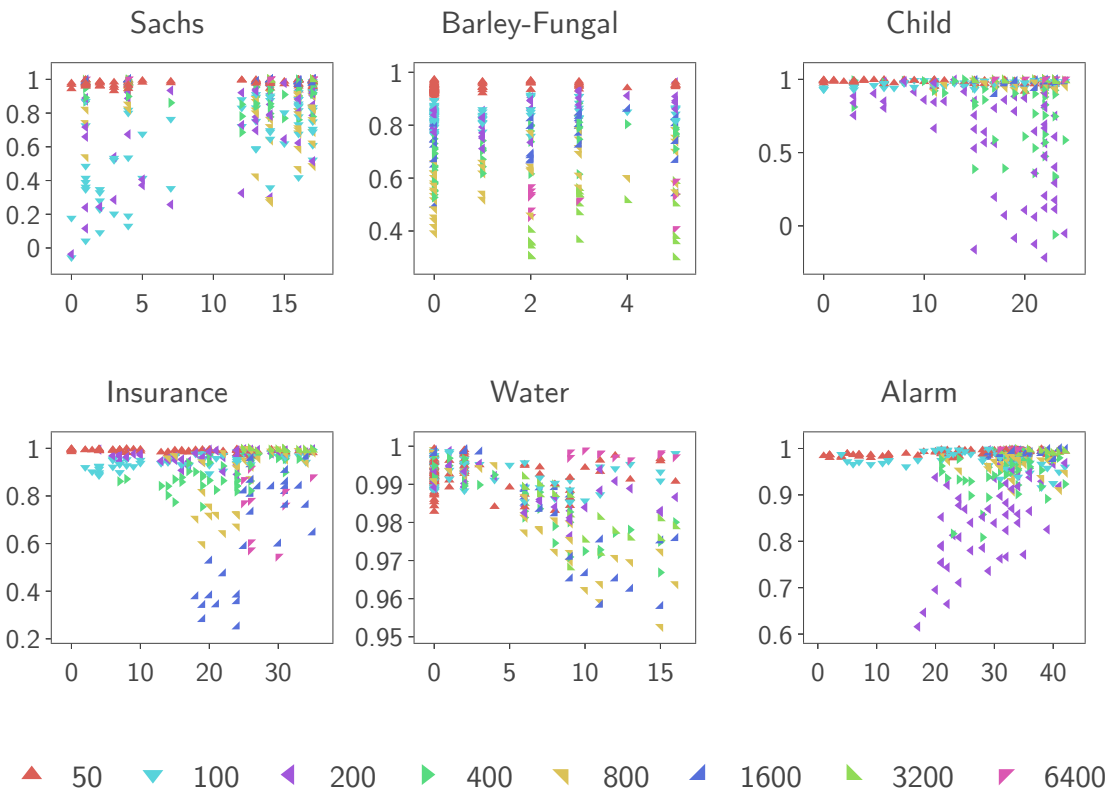


Figure 34: Pearson correlation (Y-axis), connected node pairs (X-axis)

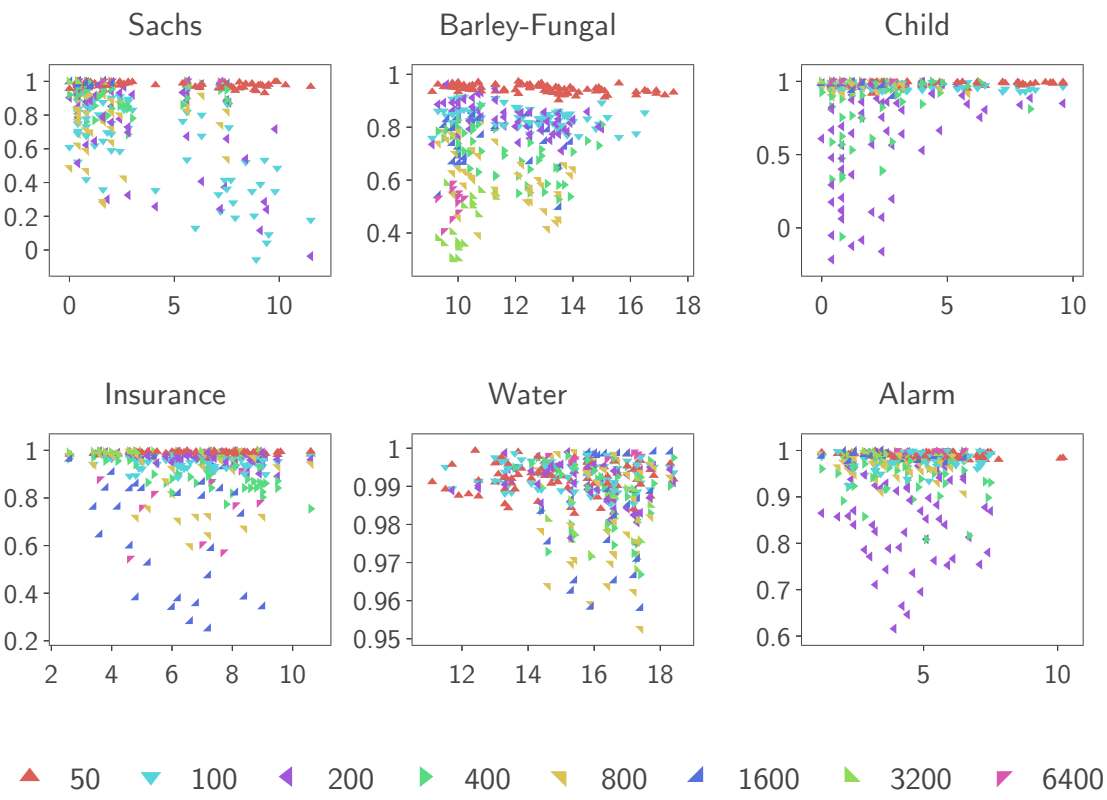


Figure 35: Pearson correlation (Y-axis), mean pairwise agreement graph operand distance (X-axis)

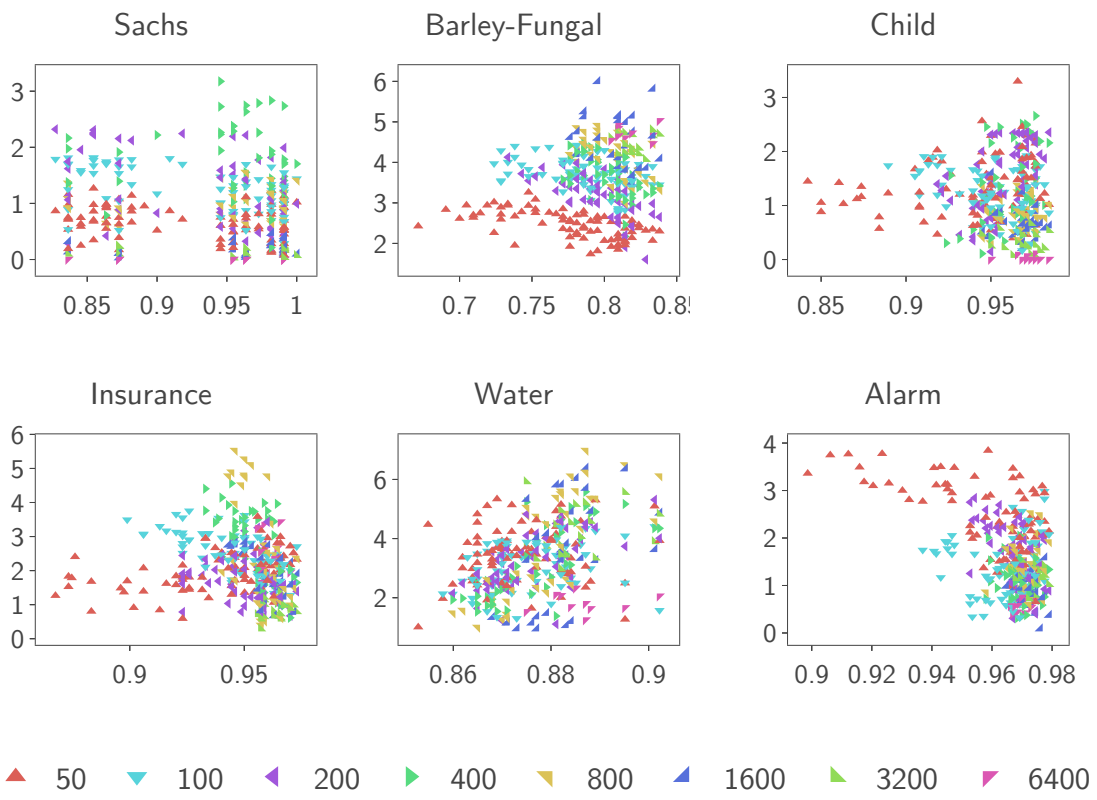


Figure 36: RMSE (Y-axis), size of agreement graph (X-axis)

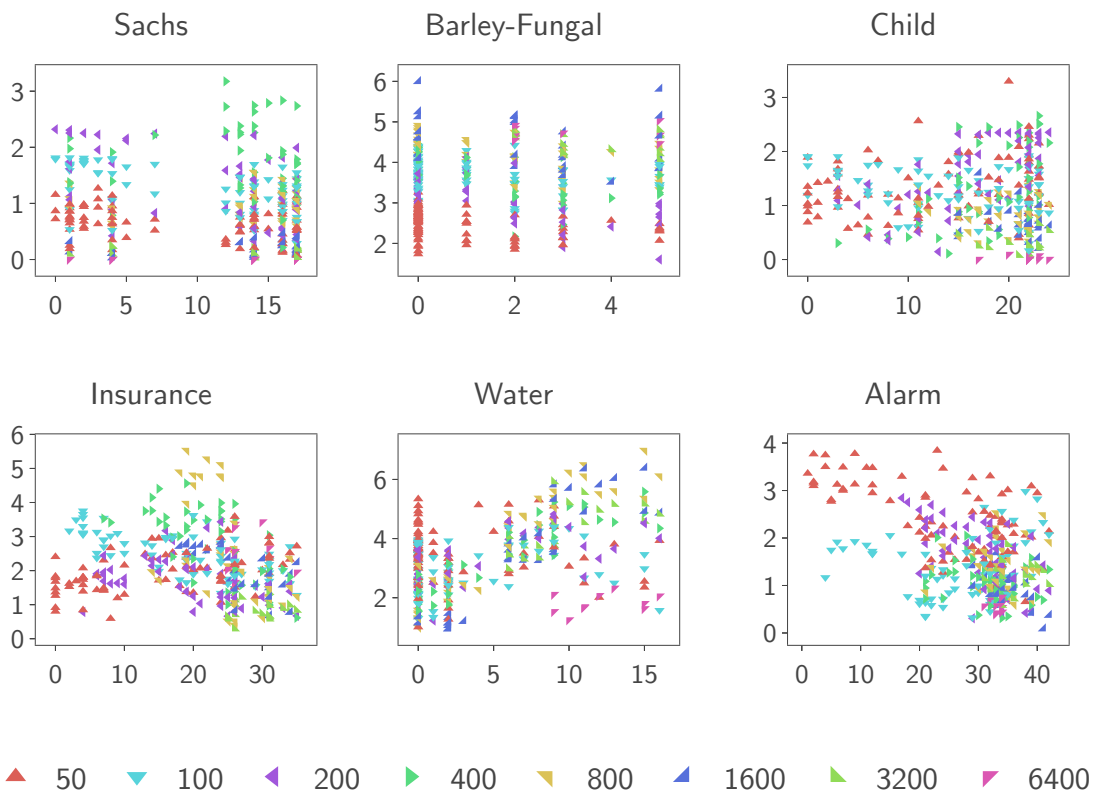


Figure 37: RMSE (Y-axis), connected node pairs (X-axis)

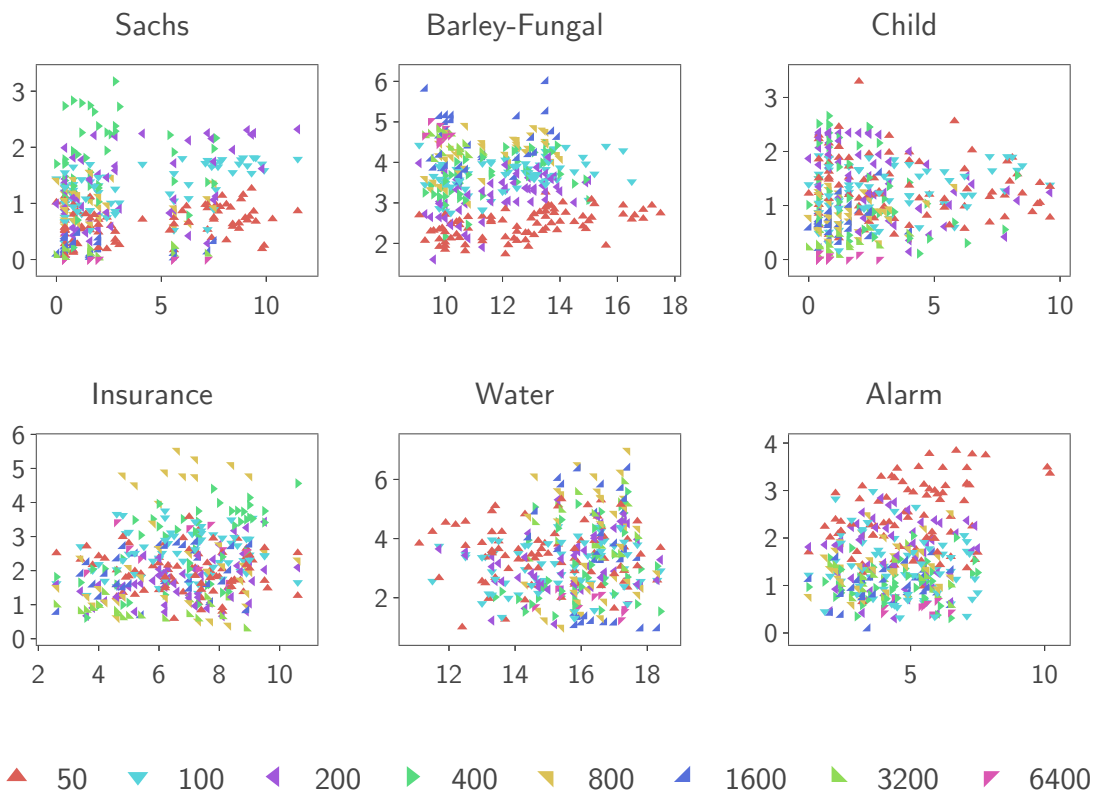


Figure 38: RMSE (Y-axis), mean pairwise agreement graph operand distance (X-axis)