Permutation Testing Improves Bayesian Network Learning

Ioannis Tsamardinos and Giorgos Borboudakis

Computer Science Department, University of Crete and
Institute of Computer Science, Foundation for Research and Technology, Hellas

Abstract. We are taking a peek "under the hood" of constraint-based learning of graphical models such as Bayesian Networks. This mainstream approach to learning is founded on performing statistical tests of conditional independence. In all prior work however, the tests employed for categorical data are only asymptotically-correct, i.e., they converge to the exact p-value in the sample limit. In this paper we present, evaluate, and compare exact tests, based on standard, adjustable, and semi-parametric Monte-Carlo permutation testing procedures appropriate for small sample sizes. It is demonstrated that (a) permutation testing is calibrated, i.e, the actual Type I error matches the significance level α set by the user; this is not the case with asymptotic tests, (b) permutation testing leads to more robust structural learning, and (c) permutation testing allows learning networks from multiple datasets sharing a common underlying structure but different distribution functions (e.g. continuous vs. discrete); we name this problem the *Bayesian Network Meta-Analysis* problem. In contrast, asymptotic tests may lead to erratic learning behavior in this task (error increasing with total sample-size). The semi-parametric permutation procedure we propose is a reasonable approximation of the basic procedure using 5000 permutations, while being only 10-20 times slower than the asymptotic tests for small sample sizes. Thus, this test should be practical in most graphical learning problems and could substitute asymptotic tests. The conclusions of our studies have ramifications for learning not only Bayesian Networks but other graphical models too and for related causal-based variable selection algorithms, such as HITON. The code is available at mensxmachina.org.

1 Introduction

Graphical models such as Bayesian Networks (BNs) are often at the heart of decision support systems and employed for prediction and diagnosis [4]. Graphical-model theory has also led to successful variable selection algorithms [3]. In addition, most causal discovery methods induce some type of a graphical model from data representing various types of causal relations among variables. This family of models includes, among others, (static, dynamic, and causal) Bayesian Networks (BNs) [13], Partially Directed Acyclic Graphs (PDAGs), Maximal Ancestral Graphs (MAGs), Partially Oriented Ancestral Graphs (PAGs) [12], and

J.L. Balcázar et al. (Eds.): ECML PKDD 2010, Part III, LNAI 6323, pp. 322–337, 2010.

Pairwise Causal Graphs (PCG) [16]. Often, these models represent a set of conditional dependencies and independencies that hold in the data distribution.

A major approach to learning such models from data is called the *constraint-based approach*: a number of tests of conditional independence are performed on the data whose results (dependence or independence) constrain the possible models fitting the data. A suitable test-strategy can then converge to a single model or all models equally fitting the data and thus are statistically indistinguishable (also called Markov Equivalent networks). Examples include the PC [13], the Fast Causal Inference (FCI) [13], and the recently introduced cSAT+ algorithm [16] that learn a PDAG, a PAG, and a PCG respectively. The constraint-based approach has also been employed for variable selection with excellent results [3].

In this paper, we take a closer look into the main operation that makes all of the above algorithm "tick": the statistical test of conditional independence. We denote as $T(X;Y|\mathbf{Z})$ the test of independence of variables X with Y given variables $\mathbf{Z}$. We denote by $Ind(X;Y|\mathbf{Z})$ and $Dep(X;Y|\mathbf{Z})$ the actual independence and dependence. $T(X;Y|\mathbf{Z})$ returns a p-value denoted by $p_{XY|\mathbf{Z}}$ corresponding to the probability of obtaining a test statistic equal to or more extreme than the observed test statistic on the data, given the hypothesis $Ind(X;Y|\mathbf{Z})$ holds. Typically, given a significance level a, the algorithm rejects $Ind(X;Y|\mathbf{Z})$ (accepts $Dep(X;Y|\mathbf{Z})$) if $p_{XY|\mathbf{Z}} \leq a$ and accepts $Ind(X;Y|\mathbf{Z})$ otherwise.

While foundational, testing conditional independence in the context of graphical model learning has not been studied in depth. In particular, for nominal categorical data two tests have been employed, namely the Pearson's χ^2 test and the likelihood ratio (LR) test [14,13,17]. Both of them are *asymptotic* tests, i.e., the returned p-value is approximate and converges to the true value in the sample limit. Statisticians have long warned that the approximation is often poor in many circumstances, particularly when the sample size is low or the probabilities of the distribution are extreme (close to 0 or 1). However, prior research has not been able to fully characterize these cases (see 9.8.4. [1]) to allow automatic detection of a poor approximation.

Ideally, one would prefer to use exact tests of independence. Unfortunately, in the general case such tests have a high computational overhead that prohibits their use in the context of learning large graphical models. In addition, they require highly specialized software that is often proprietary [10]. To address the problem, we advocate and study easy-to-implement, Monte-Carlo permutation tests. These tests are exact in the sense that $E(\hat{p}) = p$, where p is the true p-value of the test and $\hat{p}$ the value returned by the test. We develop and compare three procedures (a) a standard Monte-Carlo simulation, (b) an adjustable permutation method, and (c) a semi-parametric permutation method. Similar techniques have been successfully developed and employed for attribute selection and pruning in Decision Trees [5], relation learning settings [11,9], rule learning and model selection [8]. We demonstrate empirically that in terms of efficiency:

- The adjustable and the semi-parametric procedures require on average about 450 and 100 permutations respectively to achieve the same learning performance in BNs as the basic procedure using 5000 permutations.

– The semi-parametric procedure is only 10 to 20 times slower than an asymptotic test for small samples sizes (< 500).

The efficiency results show that simple permutation procedures could replace the asymptotic tests without a prohibitive efficiency cost in many practical situations. In terms of the learning-performance benefits, we show that:

– Permutation procedures are more effective in distinguishing between dependence and independence than the asymptotic tests for small sample sizes.
– Permutation procedures are calibrated, i.e, their actual Type I error matches the significance level α; this is not the case with the asymptotic tests.
– Permutation procedures lead to more robust BN structural learning.
– Perhaps most importantly, exact tests allow the development of algorithms that are able to learn from multiple datasets non-identically distributed. In contrast, asymptotic tests with the heuristics lead to erratic behavior where the error rate increases as the available data increase.

We name the problem of learning a BN from multiple datasets assumed to be sharing the same structure, defined over the same variables, and obtained under similar experimental and sampling conditions *Bayesian Network Meta-Analysis* learning (BNMA). This is because it generalizes statistical meta-analysis: the latter aims to induce a single dependency relation from multiple similar studies, while the former aims to learning a complete BNs (a set of dependencies and independencies). This situation may occur when different studies measure the same variables but on different scales or using different methods or equipments. For example, in one study *Smoking* maybe taking a dichotomous No/Yes value, while in another the values No/Light/Regular/Heavy smoker. In different psychology studies the level of depression may be measured using different questionnaires and methods. In different gene-expression micro-array experiments, gene expression values from different experiments cannot be easily translated to a common scale for various technical reasons [7]. In all these cases, one could not pool all the data together in a single dataset without the risk of losing information or performing an inappropriate data transformation. Techniques for BNMA learning have been recently introduced independently by our group [18] and Tillman et. al. [14]. Permutation tests allow these techniques to be applicable in practical cases where each dataset has low sample size. We suspect exact testing to be important in other BN learning-related procedures that depend on the exact value of the p-values returned. Such a procedure is a technique for controlling the False Discovery Rate of the identified edges in a BN [19].

2 Asymptotic Tests of Independence

We now consider a test $T(X; Y|\mathbf{Z})$ and denote by $N_{xy\mathbf{z}}$ the number of samples where $X = x$, $Y = y$, and $\mathbf{Z} = \mathbf{z}$ in the data, where $\mathbf{z}$ is a vector of values of $\mathbf{Z}$ in case there are more than one variable in the conditioning set. We denote with $N_{x+\mathbf{z}} = \sum_y N_{xy\mathbf{z}}$ and similarly for $N_{+y\mathbf{z}}$, N_{xy+}, $N_{++\mathbf{z}}$, and $N_{+++} = N$ the total

sample size. Finally, we denote with $|X|$ the size of the domain of variable X and with $|\mathbf{Z}|$ the number of joint states of the variables in $\mathbf{Z}$. Assuming $Ind(X;Y|\mathbf{Z})$ the expected number of samples where $X = x$, $Y = y$, and $\mathbf{Z} = \mathbf{z}$ is:

$$E_{xy\mathbf{z}} = \frac{N_{x+\mathbf{z}} \cdot N_{+y\mathbf{z}}}{N_{++\mathbf{z}}}$$

when the actual observed number is $N_{xy\mathbf{z}}$. The overall discrepancy between these two quantities in all cells of the contingency tables is captured by the following two test statistics:

$$\mathcal{X}^2 = \sum_{x,y,\mathbf{z}} \frac{(N_{xy\mathbf{z}} - E_{xy\mathbf{z}})^2}{E_{xy\mathbf{z}}} \qquad \mathcal{G} = 2 \sum_{x,y,\mathbf{z}} N_{xy\mathbf{z}} \ln \frac{N_{xy\mathbf{z}}}{E_{xy\mathbf{z}}}$$

Whenever $E_{xy\mathbf{z}}$ is zero the corresponding term in the summation is defined as zero as well. Both of these statistics are *asymptotically* distributed as χ^2_{df} with $df = (|X|-1)(|Y|-1)|\mathbf{Z}|$ degrees of freedom. Using the $\mathcal{X}^2$ as a test statistic leads to the Pearson's χ^2 test of independence, while using the $\mathcal{G}$ leads to a likelihood ratio test of independence, also called a G-test [1]. The $p_{XY|\mathbf{Z}}$ is calculated as $1 - F(S_o)$ where F is the cumulative distribution function of χ^2_{df} and S_o the observed value of the statistic (either the $\mathcal{X}^2$ or $\mathcal{G}$) in the data.

Let us denote with $m = \#\{N_{xy\mathbf{z}}\}$ the number of counts to calculate. When all variables are ternary then for $T(X;Y|Z_1,Z_2)$ we get $m = 3^4$. Thus, the average number of samples to estimate each count drops exponentially with the number of variables to condition. This reduces the statistical power and increases the number of cells with zero counts. In other words, with a large enough conditioning set, any $T(X;Y|\mathbf{Z})$ will return a high $p_{XY|\mathbf{Z}}$ with high probability leading the algorithm to accept $Ind(X;Y|\mathbf{Z})$.

Two heuristic solutions have appeared in the literature. First, some algorithms (PC, MMPC) [13,17] do not perform $T(X;Y|\mathbf{Z})$ if they determine there are not enough samples to achieve large enough power. The algorithms require that the average sample per count N/m is at least π, where π is a user defined parameter. Typical values for π are 10 [13], 5 [17], 0 [15] (in chronological order). We call this the *heuristic power rule*.

Second, several of the zero counts $N_{xy\mathbf{z}} = 0$ that appear in the contingency tables are heuristically declared as "structural zeros", i.e., they are judged to stem from structural constraints of the problem and not as random events. A structural zero for example, would appear if X is "taking contraceptives", Y measures "osteoporosis" and Z is gender. We expect that $N_{Yes,y,Male}$ to be zero. Since structural zeros are not free to vary, the degrees of freedom of the test should be adjusted. Spirtes et. al. [13] consider as structural any zero that appears in the contingency tables. They subtract one from df for every such zero, which may actually lead to negative degrees of freedom. In [17] we present a different heuristic where we consider as structural zero any case $N_{xy\mathbf{z}} = 0$ and also either of the marginals are $N_{+y\mathbf{z}} = 0$ or $N_{x+\mathbf{z}} = 0$. For example, if $N_{+y\mathbf{z}} = 0$, then we consider y as a structurally forbidden value for Y when $\mathbf{Z} = \mathbf{z}$ and we reduce the degrees of freedom by $|X| - 1$ (as if we had one column less in

the contingency table where $\mathbf{Z} = \mathbf{z}$). We call the latter method the *degrees of freedom adjustment* heuristic. There have been several pieces of work examining the appropriateness of the approximate tests [1] and several attempts and rules to characterize cases of poor approximation. However, a full characterization is still lacking. The alternative is to use exact tests of independence that is presented next.

3 Permutation Tests of Independence

The first exact test for categorical data to appear has been Fisher's exact test [1] that treats the special case in 2×2 tables (i.e., $T(X;Y|\emptyset)$ with $|X| = |Y| = 2$). However, generalizing exact testing in the general case of $i \times j \times k$ (conditional test with unrestricted sizes of domains for all variables) has been proven a difficult task. A mainstream approach to exact testing is called the *exact conditional approach* that considers the row and column marginals in each table $N_{x+\mathbf{z}}$, $N_{+y\mathbf{z}}$, and $N_{++\mathbf{z}}$ as fixed [2,1,8]. The distribution of the test statistic under the null hypothesis is then calculated conditioned on these marginals. Specifically, to calculate the exact p-value one needs to calculate $P(S_o \geq S|Ind(X;Y|\mathbf{Z}))$, where S_o is the observed test statistic. This in turn implies identifying all contingency tables with the same marginals and whose test statistic is larger or equal to the observed one. The number of possible tables with the same marginals quickly explodes: "a 4×4 table ... with a 100 observations can have about 7×10^9 such tables" [2]. Various computational methods have appeared that attempt to do the computations implicitly without enumeration of all tables, some of which have led to the development of the StatXact package [10]. These methods however, are still relatively slow, hard to implement, and not freely available.

In this section, we present intuitive, easy-to-implement, and relatively efficient permutation testing procedures. Notice that, each table (where $\mathbf{Z} = \mathbf{z}$) with the same marginals as the observed table can be produced by permuting the values of X or Y of the samples (while retaining $\mathbf{Z} = \mathbf{z}$). For example, for binary variables X, Y, Z, suppose we have the observations $\langle 0,1,0\rangle$ and $\langle 1,0,0\rangle$ giving $N_{0+0} = N_{1+0} = N_{+00} = N_{+10} = 1$. Permuting the two values of Y between the only two observations provides the permuted data $\langle 0,0,0\rangle$ and $\langle 1,1,0\rangle$ with the same marginals. Under the null hypothesis of independence, this is justified as follows: since X and Y are assumed independent given $\mathbf{Z}$, any such permutation has the same probability of being observed.

Calculating all such possible permutations is equivalent to enumerating all possible tables with the same marginals. However, one can sample from the space of all possible permutations (tables) randomly to estimate $\hat{P}(S_o \geq S|Ind(X;Y|\mathbf{Z}))$. Such methods are called Monte Carlo Permutation methods [6]. We denote with $D_0 = \{\langle x, y, \mathbf{z}\rangle\}_{j=1}^{N}$ the unpermuted, observed data. We obtain permuted data D_i, $i > 0$ as follows: for each possible value $\mathbf{z}$ of $\mathbf{Z}$, randomly permute the values of Y in D_0 *only* among the cases where $\mathbf{Z} = \mathbf{z}$ (i.e., ensuring all marginals remain the same). We denote with $S(D_i)$ the value of the statistic (either $\mathcal{X}^2$ or the $\mathcal{G}$ statistic) on the data D_i. The basic procedure is shown in Algorithm 1.

Algorithm 1. Basic Permutation $T(X;Y|\mathbf{Z})$

Input: Data $D_0 = \{\langle x, y, \mathbf{z}\rangle\}_{j=1}^{N}$, Test Statistic S, number of permutations B
Output: $\hat{p}_{XY|\mathbf{Z}}$
for $i = 1, \ldots, B$ **do**
  Randomly permute data to create D_i; calculate $S(D_i)$
end
return $\hat{p}_{XY|\mathbf{Z}} = \#\{S(D_0) \leq S(D_i), i = 1, \ldots B\}/B$

The procedure requires the computation of $S(D_i)$ an additional B times compared to the asymptotic test, so as given it is at least that many times slower. One obvious optimization to the procedure is to notice that the values $E_{xy\mathbf{z}}$ depend only on the marginals that remain the same across all datasets (observed and permuted) and so can be computed only once.

A sufficient number of permutations B seems to range between 1000 to 5000 which makes the procedure quite costly for learning large graphical models (we used 5000 in our experiments). To improve the computational requirements, we design an adjustable procedure that may stop early the computation of more permuted statistics. The procedure infers whether the current approximation $\hat{p}_{XY|\mathbf{Z}}$ is sufficiently close to the true $p_{XY|\mathbf{Z}}$ to make a decision at significance level a, i.e., to determine whether $p_{XY|\mathbf{Z}} \leq a$ or not.

First, we implement a heuristic rule based on asymptotic tests in an effort to completely avoid permutation testing in easy-to-determine cases. Assuming a typical range for the significance level to be between 0.01 and 0.1, then if a conservative asymptotic test (in our experiments the $\mathcal{G}$ test) returns a relatively much lower p-value (lower than 0.001 in our experiments) we immediately accept dependence. Similarly, if a liberal asymptotic test ($\mathcal{X}^2$ with the *df* heuristic adjustment) returns a high p-value (larger than 0.5), we accept independence.

Rule 1 : if $p_{\mathcal{G}} < 0.001$ return Dep., else if $p_{\mathcal{X}^2} > 0.5$ return Ind.

If the rule does not apply, we begin permutation testing. To bound the error of approximation at the current iteration b, we proceed as follows. We define a Bernoulli trial $X_i = I(S(D_0) \leq S(D_i))$, i.e., the event of a random permutation obtaining a larger statistic than the observed. The probability of success $P(X_i = 1)$ is equal to the exact p-value of the test by definition. Thus, $\sum_{i=1}^{b} X_i$ follows a Binomial(B, $p_{XY|\mathbf{Z}}$). For relatively large b and non-extreme p-values we can approximate this distribution with a normal distribution $\mathrm{N}(\mu', \sigma')$, where $\mu' = b \cdot p_{XY|\mathbf{Z}}$ and $\sigma'^2 = b \cdot p_{XY|\mathbf{Z}} \cdot (1 - p_{XY|\mathbf{Z}})$. Thus, $\hat{p}_{XY|\mathbf{Z}} = \sum_{i=1}^{b} X_i/b$ follows $\mathrm{N}(\mu, \sigma)$, where $\mu = p_{XY|\mathbf{Z}}$ and $\sigma^2 = p_{XY|\mathbf{Z}} \cdot (1 - p_{XY|\mathbf{Z}})/b$. Based on this approximation, we find the confidence interval $p_{XY|\mathbf{Z}} = \hat{p}_{XY|\mathbf{Z}} \pm \epsilon$, $\epsilon > 0$. The maximum magnitude of ϵ called $r(b)$, where b is the current iteration, is obtained for $\sigma^2 = 1/4$ and p-value=0.5 . Then, with probability δ (exercise 3.45 at [1]):

$$r(b) = \frac{\Phi^{-1}(\frac{1+\delta}{2})}{2 \cdot \sqrt{b}}$$

Algorithm 2. Adjustable Permutation $T(X;Y|\mathbf{Z})$

Input: Data $D_0 = \{\langle x, y, \mathbf{z}\rangle\}_{j=1}^{N}$, Test Statistic S, maximum number of permutations B, significance level a
Output: Independent/Dependent

Let $\delta = 0.99$
Apply **Rule 1**
for $b = 1, \ldots, B$ **do**
  Randomly permute data to create D_b; calculate $S(D_b)$
  Apply **Rule 2**
  Apply **Rule 3**
end
if $\hat{p}_{XY|\mathbf{Z}} > a$ **then**
  return *Independent*
else
  return *Dependent*
end

Algorithm 3. Semi-Parametric (fitted) Permutation $T(X;Y|\mathbf{Z})$

Input: Data $D_0 = \{\langle x, y, \mathbf{z}\rangle\}_{j=1}^{N}$, Test Statistic S, number of permutations B
Output: $\hat{p}_{XY|\mathbf{Z}}$

for $i = 1, \ldots, B$ **do**
  Randomly permute data to create D_i; calculate $S(D_i)$
end
$df = \overline{S(D_i)}$
return $\hat{p}_{XY|\mathbf{Z}} = 1 - F(S(D_0))$, where F is the cumulative distribution of χ^2_{df}

The adjustable procedure periodically checks whether the following rule applies to the current approximation of the p-value $\hat{p}_{XY|\mathbf{Z}}(b)$ at iteration b:

Rule 2: if $\hat{p}_{XY|\mathbf{Z}}(b) + r(b) \leq a$ return Dep., else if $a \leq \hat{p}_{XY|\mathbf{Z}}(b) - r(b)$ return Ind.

We additionally implement an idea described in [6] that prematurely aborts further permutations based on worst-case reasoning. Specifically, if assuming all the remaining permutations give $S(D_0) \leq S(D_i)$ and our estimate $\hat{p}_{XY|\mathbf{Z}}$ would still be greater than a, we can immediately determine independence. Similarly, if assuming all remaining permutations turn out $S(D_0) > S(D_i)$ and our $\hat{p}_{XY|\mathbf{Z}}$ would still be less than a, we can immediately determine dependence. Given these observations, a lower bound LB of $\hat{p}_{XY|\mathbf{Z}}$ at any iteration b during the algorithm is: $LB(b) = \#\{S(D_0) \leq S(D_i), i = 1, \ldots b\}/B$ where B is the maximum allowed number of permutations. Similarly, we find an upper bound UB as $UB(b) = 1 - \#\{S(D_0) > S(D_i), i = 1, \ldots b\}/B$ and so, the final early-stopping rule implemented is:

Rule 3 : if $UB(b) < a$ return Dep., else if $LB(b) > a$ return Ind.

We formalize the procedure in Algorithm 2. Finally, we consider a semi-parametric approach often used in resampling (permutation and bootstrapping) methods to reduce the number of permutations required. When the distribution of the test statistic is suspected to follow a specific parametric form of unknown parameters it is not necessary to perform a larger number of permutations to identify the shape of the distribution. Only a relatively small number of permutations are performed to estimate the parameters. In our case, we assume that the distribution is well-approximated by a χ^2_{df} distribution with a single unknown parameter, the degrees of freedom df. In preliminary experiments we determine that a reasonable number of permutations required to estimate df is about 100. The maximum likelihood estimate of df given samples from a χ^2_{df} is the sample mean. Once the df has been approximated, the test calculates the p-value as in the asymptotic tests. The procedure is presented in Algorithm 3.

4 Distinguishing between Dependence and Independence

In this section we empirically evaluate the ability of the tests to distinguish between dependence and independence situations. We consider the following two network structures $A \leftarrow C \rightarrow B$ and $A \rightarrow C \leftarrow B$ because they are heavily involved in the basic reasoning operations performed by typical network algorithms such as the PC. In the former structure it holds that $Dep(A; B|\emptyset)$ and $Ind(A; B|C)$. The latter independence would lead network algorithms to remove the edge $A - B$ in the graph. For the latter structure it holds that $Ind(A; B|\emptyset)$ and $Dep(A; B|C)$. This subgraph is called a v-structure [13] and allows network algorithms to orient edges. For each structure we perform the unconditional test $T(A; B|\emptyset)$ and the conditional $T(A; B|C)$. We consider the following cases for the sizes of the domains of the variables A, B, and C respectively: $2 \times 2 \times k$, $k = 2, 4, 8$ and $4 \times 4 \times k$, $k = 4, 16, 32$. For each such case, we randomly sample with uniform probability the parameters of the network 20 times. For each parametrization of the networks and for each sample size in the set $\{20, 40, 60, 80, 100, 150, 200, 300, 500\}$ we randomly create 25 datasets from the distribution of the network. The total number of tests to perform is 2 (networks) $\times$ 2 (types of tests) $\times$ 6 (domain definitions) $\times$ 9 (sample sizes) $\times$ 20 (parameterizations) $\times$ 25 samplings = 108K tests per method to evaluate.

The methods under study are: asymptotic tests with and without the heuristic adjustment based on the G and the $\mathcal{X}^2$ test statistic, denoted by AG, AX, AGh, and AXh, A denoting an asymptotic test, the second letter referring to the test statistic used, and h referring to the employment of the heuristic; the basic and the fitted permutations based on the two statistics, denoted with PGB, PXB, PGF, PXF, P denoting a permutation based test, the second referring to the test statistic, and the last letter to the basic or fitted procedure. The adjustable procedure is not included in this set of experiments because it returns a binary decision, not a p-value. Thus, there is a total of 8 methods to evaluate.

The hardest cases (smaller ratio of samples over counts-to-estimate) are summarized in Figure 1(a)-(c) that show the AUCs over sample size of all datasets

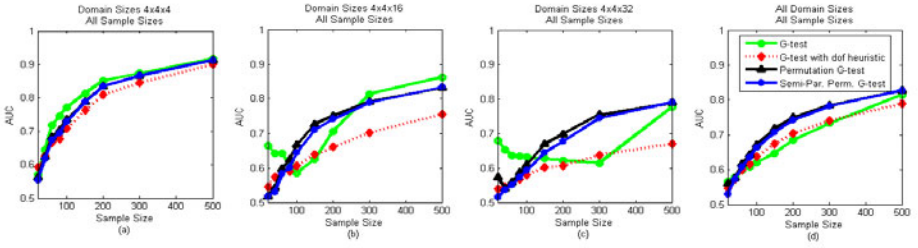

Fig. 1. Distinguishing between $Ind(X;Y|\mathbf{Z})$ **and** $Dep(X;Y|\mathbf{Z})$. (a)-(c) AUC with sample size for the tests based on the G statistic for the $4 \times 4 \times k$ cases. The asymptotic tests have an erratic behavior for the more difficult cases. (d) AUC over sample size for all experiments. The permutation-based tests show statistically significantly ($p < 0.0001$) increased AUC ranging between 1.75% and 3.39% according to the 95% confidence intervals (Table 1).

where the domain counts $4 \times 4 \times k$, $k = 4, 16, 32$. The AUCs were calculated as follows: the p-values returned on all tests performed by a method on a group of tests were ordered and thresholded by the significance level. Taking all possible thresholds gives the ROC curve and the AUC for that method on that group of tests. To avoid clutter in the figures we only present the tests based on the G statistic; the results are similar for the $\mathcal{X}^2$. In figure (a) all tests have a reasonable behavior with increasing performance as sample size increases. As k increases and the sample is split to 16 and then 32 contingency tables (figures (b) and (c)) the behavior of the asymptotic tests becomes erratic and large dips in the curves appear. For the $4 \times 4 \times 32$ case there are 512 counts $N_{xy\mathbf{z}}$ so in the best case on average there is about 1 sample to estimate each count and a large number of expected zero counts. The asymptotic approximations of the tests fail in these cases while the permutation-based procedures are quite robust. Figure 1(d) shows the overall AUCs on all datasets; this includes the easier cases of $2 \times 2 \times k$, $k = 2, 4, 8$ so the average behavior is smoother for all tests. Finally, the AUCs of all methods are presented at Table 1. We also note that, the permutation procedures do not depend as much on the choice of the test statistic (figures omitted for brevity).

Note that, the permutation methods are worse than the asymptotic methods for sample sizes 20-60 and $k = 16, 32$. This is because the number of possible permutations (that maintain the marginal counts) is too low to estimate the distribution of the test statistic. When sample size increases, the number of admissible permutations grows exponentially and the methods quickly improve over the parametric ones. E.g. the percentage of unique values of the statistic in 10000 permutations for the $4 \times 4 \times 32$ case and sample size 20, 40, 100, 150 turns out to be on average 0.01%, 0.06%, 3%, and 12% respectively.

To compare whether the differences between the methods are statistically significant, we used (what else?) a permutation testing procedure. We define AUC_m the AUC returned by a method m on all results and define the statistic $\Sigma_m = AUC_{PGF} - AUC_m$. To generate a single round of permuted data, we

Table 1. AUC: The Area Under the ROC Curve of the testing procedures over all networks, domain sizes, sample sizes, and samplings. CI(%): 95% confidence interval of the difference with PGF Σ_m times 100.

	Asymptotic Tests				Permutation Tests			
	AG	AX	AGh	AXh	PGB	PXB	PGF	PXF
AUC	0.6432	0.6197	0.6548	0.5918	0.6766	0.6749	0.6744	0.6718
CI(%)	[2.85 3.39]	[5.18 5.76]	[1.75 2.16]	[7.91 8.61]	[-0.34 -0.10]	[-0.19 0.07]	0	[0.037 0.16]

permute each pair of corresponding p-values (whether they come from PGF or m) with 50% probability. We estimate the empirical distribution of Σ_m using 10000 such rounds. The PGF test is statistically significantly better ($\Sigma_m > 0$) than all asymptotic tests ($p < 0.0001$) and than PXF ($p < 0.0001$). However, the PGF performs worse than the PGB ($p < 0.0001$) and PXB ($p = 0.19$). To estimate the performance difference between PGF and the other tests, we present the 95% confidence intervals for Σ_m in Table 1. The improvements range from at least 1.75% improvement in AUC over the AGh, to at least 7.91% over the AXh. Based on these results, we forgo the use of tests based on the $\mathcal{X}^2$ statistic in the rest of the paper.

In terms of execution time, the asymptotic tests require about the same time given that the d.f. adjusting heuristic takes time linear to the number of counts computed. The PXB and PGB optimized versions take approximately 500-700 times more than the asymptotic procedures for the sample and domain sizes used in the study. Finally, the PGF takes about only 10-20 times more than the asymptotic tests. Given that the difference in performance between PGB and PGF is less than 0.4% AUC we consider PGF a good trade-off between performance and computational effort spent.

5 Evaluating the Calibration of the Tests

We now evaluate the relation between the actual Type I error when rejecting the null hypothesis at confidence level a ($P(p \leq a|\text{null})$), where p is the p-value returned by the test, and the level a. For a calibrated test these two quantities should be equal, i.e., $P(p \leq a|\text{null}) = a$. Notice that, a test may not be calibrated (e.g., always return half the true p-value) and still achieve a high AUC if the cases of $Dep(X;Y|\mathbf{Z})$ have a lower p-value than the cases of $Ind(X;Y|\mathbf{Z})$.

Figure 2 shows $P(p \leq a|\text{null})$ (Type I error) vs. the confidence level a. $P(p \leq a|\text{null})$ is estimated as the proportion of cases where $p \leq a$ in datasets where $Ind(X;Y|\mathbf{Z})$ holds. Figure 2(a)-(d) focus on the hardest case of $4 \times 4 \times 32$ experiments. With sample size 20 all tests return non-calibrated p-values. At sample size 60 the basic permutation procedure is already well-calibrated. At sample size 150 the semi-parametric procedure catches up. The asymptotic tests fail miserably to return calibrated values even for the largest sample size attempted of 500 cases. In the second row, Figures 2(e)-(g) show the results for the $4 \times 4 \times k$ domain sizes averaged out on all sample sizes. The last sub-figure shows the average overall behavior including the easier cases of $2 \times 2 \times k$. The results for

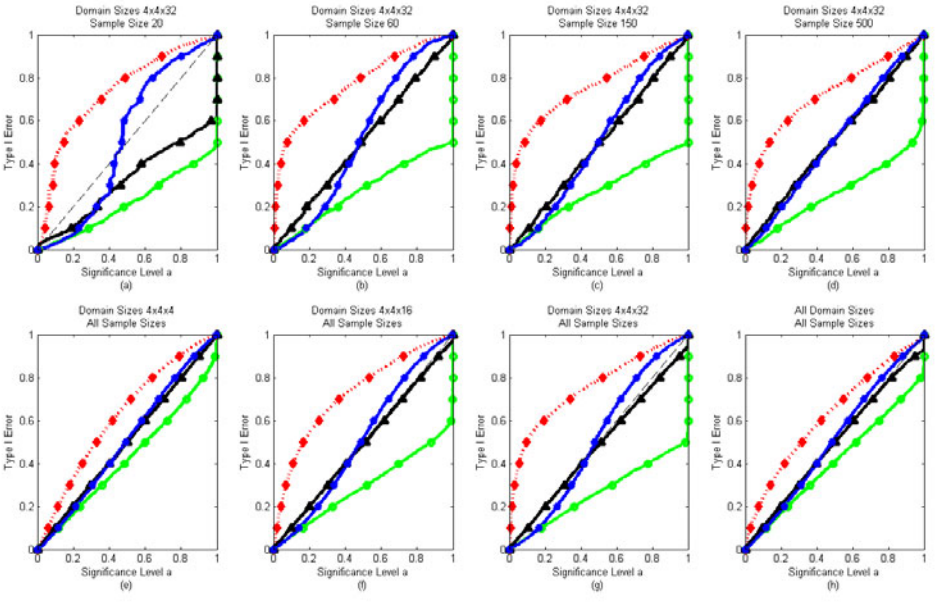

Fig. 2. Calibration Properties of Tests. The subfigures plot the Type I error ($P(p \leq a|\text{null})$) vs. the significance level a for various cases. The legend is the same as in Figure 1. The asymptotic G test with the heuristic (red line) underestimates the true p-value; the asymptotic G test (green line) overestimates it. The permutation procedures are relatively well-calibrated.

the tests based on the $\mathcal{X}^2$ statistic lead to similar conclusions, not shown due to lack of space. In conclusion, when independence holds *the asymptotic G test with the heuristic underestimates the true p-value, while the asymptotic counterpart without the heuristic adjustment overestimates it. The permutation procedures are well-calibrated for the complete range of the significance levels.*

6 Improving Bayesian Network Learning

We now examine whether the improvements in AUC and calibration of the permutation procedures translate to improved learning rates and robustness. We consider four typical Bayesian Networks in the literature used in decision support systems, namely the ALARM, Insurance, Hailfinder, and Child [17]. These have 37, 27, 56, 20 variables and 46, 52, 66, 25 edges respectively. From the distribution of each network and for each sample size in { 50, 75, 100, 125, 150, 175, 200, 250, 300, 500, 700, 1000, 2500, 5000} we sample 20 times simulated data. The prototypical PC algorithm [13] is then employed to attempt to reconstruct the network from the data. We used our implementation of the PC algorithm with one main difference from the original version: the algorithm begins from the empty graph and not the complete one, then adds the edges corresponding to detectable pairwise associations (see [17] for a detailed justification).

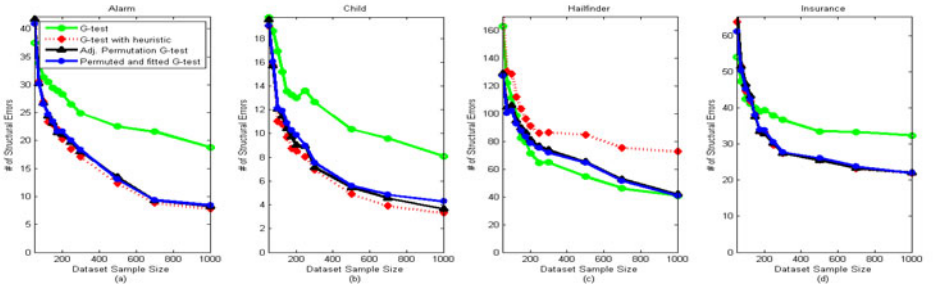

Fig. 3. Learning the Skeleton of Bayesian Networks. The number of structural errors (extra and omitted edges) is presented for each network and procedure. The performance of the asymptotic procedures highly depends on the network. Permutation-based method robustly perform in all networks and sample sizes.

The PC algorithm is executed with different testing procedures to evaluate their efficacy. PC accepts two parameters, the significance level a for the tests and the threshold π in the heuristic power rule that determines which tests to omit. The significance threshold used in all cases is the standard 0.05. In our experiments $\pi = 2$ selected as the value of $\pi \in \{0, \ldots, 15\}$ that optimizes the performance of the asymptotic tests.

Based on the results of Section 4, we focus on tests based only on the G statistic. In addition, we exclude the basic permutation procedure from the evaluation as having a large computational overhead. Instead, we employ the adjustable permutation procedure (PGA) defined in Algorithm 2. In preliminary experiments (not shown) this algorithm was found to be a good approximation of the basic permutation procedure. Thus, overall there are four tests in the evaluation: AG, AGh, PGF, and the PGA.

The structural errors are defined as the number of extra and omitted edges and are shown in Figure 3. The adjustable and semi-parametric permutation procedures are always close to the maximum accuracy achieved by any method in all four networks. The permutation-based methods perform similarly in terms of structural errors. In terms of computations, the number of permutations required for the adjustable procedure is on average about 450 while the semi-parametric procedure performs a fixed number of only 100 permutations. *The semi-parametric permutation test performs surprisingly well; the results show that permutation-based tests can be practical for graphical model learning.*

When it comes to the asymptotic tests, it is easy to observe that their performance highly depends on the network structure. The AGh test outperforms the AG test in three out of the four networks. However, the reverse relation holds in the Hailfinder network. This is explained due to the existence of several variables with a large domain and many adjacencies: AGh in general underestimates the true p-value and so the large-domain variables initially obtain many adjacencies at the first iteration of the algorithm. Once this happens, it becomes impossible to perform tests involving these variables due to the power rule, which eventually leads to a large number of false positives. The AG tests never includes edges

with these variables and so there are many fewer false positives. E.g., for sample size 500 on Hailfinder the average number of structural errors for nodes 3 and 40 (domain size 11) and its neighbors (19 nodes) is 61 and 31 for the AGh and the AG respectively. For all other 37 nodes it is 23 and 23.

In contrast, the permutation procedures are highly robust over all networks. Over all networks and sample sizes, the PGF procedure statistically significantly outperforms the asymptotic ones (single-sided, 10000 permutations $p < 0.0001$) in terms of the number of structural errors. However, we would like to note that the learning performance of an algorithm depends on the sparseness of the graph (ratio between edges and non-edges) and the cost associated with a false positive (extra edge) and a false negative (omitted edges). The number of structural errors in particular penalizes both types of errors by the same amount. A more complete evaluation should optimize the threshold a of a test relative to the performance measure. Given this discussion, *the important conclusion from this set of experiments in our opinion is the robustness of the permutation tests to the different characteristics of the networks relative to asymptotic tests.*

7 Bayesian Network Meta-analysis

In this set of experiments, we demonstrate the importance of exact testing procedures to the BNMA problem. Recall that in such tasks cases, one can not pool all the data together in a single dataset. Fortunately, we can overcome this problem. Most constraint-based algorithms that learn graphical models only require the ability to perform conditional tests. A straightforward approach that employs all datasets is to perform the test $T_i(X; Y|\mathbf{Z})$ individually in each available dataset D_i obtaining the p-values $\{p_i\}$. Fisher's Inverse χ^2 test can then be used to compute a combined statistic $S = -2\sum \log p_i$. S follows a χ^2 distribution with $2q$ degrees of freedom, where q is the number of datasets contributing data to the test, from which we can obtain the combined p-value p^* for the test $T(X; Y|\mathbf{Z})$ on all datasets (see also [7,14] for other methods).

When combining multiple datasets, each dataset may not have enough sample to perform the test, but their combination could have. So, we generalized the heuristic power rule to perform a test when $\pi \leq \sum N_i/m_i$ over all datasets i where N_i is the sample size of D_i and m_i the number of parameters estimated by the specific test (e.g., m_i may be different from m_j if a variable takes 3 possible values in one dataset and 4 in the other). When all datasets have the same number of parameters for the test, the rule results in testing whether $\pi \leq N/m$ as in the single-dataset case. Again, the value of π is set to 2.

To evaluate the performance of the PC on multiple-datasets we employed the same four networks as in Section 6. For each network we decide on the number of datasets to feed the PC in the set $\{1, 2, 3, 4, 5, 7, 10, 15, 20\}$ and the size of these datasets within the set $\{50, 100, 300, 500\}$. For each case (4 networks $\times$ 9 dataset-collection size $\times$ 4 sample sizes) we sample 10 times from the distribution of the network. All reported results are averaged over the 10 samplings (repeats of an experiment). We then execute the PC algorithm equipped with Fisher's Inverse

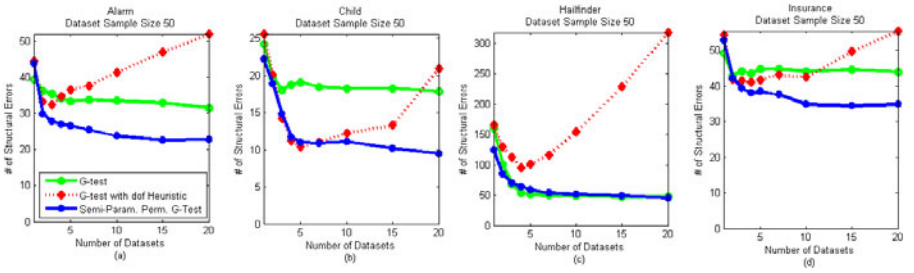

Fig. 4. Learning the Skeleton of Bayesian Networks from Multiple i.i.d. Datasets: the effect of increasing number of datasets. The permutation procedure dominates performance. The G-test with the heuristic adjustment has counter-intuitive behavior showing more errors with total available sample.

χ^2 test to combine p-values from the different datasets, and the AG, AGh, and PGF to compute the p-values on each individual dataset, giving rise to three versions of the algorithm. Obviously, in this set of experiments the datasets fed to the PC could be pooled together; the results are to validate the methods in the simplest case of i.i.d. datasets.

The results for sample size 50 are shown in Figures 4(a)-(d) where the number of structural errors is shown over the number of datasets combined. An interesting phenomenon is that the AGh exhibits decreasing performance with available datasets (and total sample size)! This is explained considering the lack of calibration of the test demonstrated at Section 5. The p-values of the test are underestimated to be closer to zero. When several of them are combined together in the statistic $S = -2\sum \log p_i$ the errors accumulate and falsely provide confidence that dependency can be accepted. For example, consider combining four exact p-values all equal to 0.2 . This gives $S = 12.8755$ and combined $p* = 0.1162$. If the approximate p-values are computed instead as half the exact values (i.e., as 0.1) then $S = 18.4207$ and the combined $p* = 0.0183$, i.e, 6 times lower. Thus, as the number of datasets and p-values that are combined each time increases, the probability of accepting dependence also increases.

Figures 5(a)-(c) show the performance (total number of structural errors in all four networks) of each algorithm respectively, as the sample size per dataset increases. As expected all algorithms benefit from the increased available size per dataset. In Figure 5(a) we observe that the counter-intuitive behavior of the AGh test is ameliorated as the asymptotic approximations of the p-values become more accurate with increased sample size. Figure 5(c) shows the average over all sample sizes of the total structural errors on all four networks: *the permutation procedure dominates the asymptotic ones in learning quality.*

In a second set of experiments using the same exact settings we simulate non-identically distributed networks. Specifically, for each non-binary variable in a dataset, with probability 10% we collapse pairs of neighboring values to a single one. For example, a variable Smoking taking values No/Light/Regular/Heavy becomes a binary No/Yes variable. The behavior of all algorithms is very similar

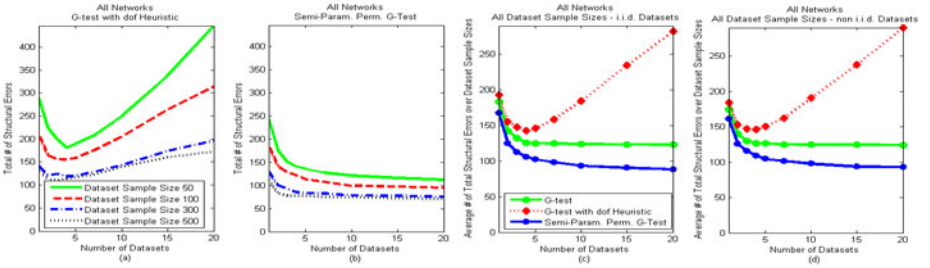

Fig. 5. (a)-(b)**The effect of increasing sample size per dataset**. The counter-intuitive behavior of the G-test with the heuristic adjustment (a) is ameliorated with increased sample size per dataset. The permutation based procedure well-behaves for small and large datasets (b). **Learning the Skeleton from Multiple i.i.d. and non i.i.d. Datasets**. The permutation procedure dominates performance in both cases. The performance for all procedures is similar in the i.i.d. (c) and the non i.i.d. (d) data.

to the results employing i.i.d. datasets, so we select to present only the average results over all networks and sample sizes in Figure 5(d). The maximum difference in average performance between figures (c)(i.i.d.) and (d) (non i.i.d.) cases is between 6-9 total structural errors for each algorithm.

8 Conclusions

Procedures for testing conditional independence are foundational for learning graphical models using constraint-based methods. The tests used in prior work for discrete data are all based on asymptotic approximations that are not robust to small sample sizes. We counter-suggest the use of exact tests based on permutation procedures. We show that the latter are both practical (10 to 20 times slower than asymptotic tests) and provide several benefits in learning behavior. Specifically, we show that (a) permutation testing is calibrated, i.e, the actual Type I error matches the significance level α set by the user; this is not the case with asymptotic tests, (b) permutation testing leads to more robust structural learning, and (c) permutation testing allows learning networks from multiple datasets that cannot be pooled together; in contrast, asymptotic tests may lead to erratic learning behavior in this task (error increasing with total sample-size). The semi-parametric permutation procedure we propose is a reasonable approximation of the basic procedure using 5000 permutations, while being only 10-20 times slower than the asymptotic tests for small sample sizes. While our evaluation considers learning BNs, the conclusions of our studies should be applicable in learning other graphical models and related causal-based variable selection algorithms.

Acknowledgements. The work was partially funded by ELKE, University of Crete contract 2934 and the REACTION GA 248590 EU project.

References

1. Agresti, A.: Categorical Data Analysis, 2nd edn. Wiley Series in Probability and Statistics. Wiley-Interscience, Hoboken (2002)
2. Agresti, A.: A survey of exact inference for contingency tables. Statistical Science 7(1), 131–153 (1992)
3. Aliferis, C.F., et al.: Local causal and markov blanket induction for causal discovery and feature selection for classification part i: Algorithms and empirical evaluation. JMLR 11, 171–234 (2010)
4. Beinlich, I., Suermondt, G., Chavez, R., Cooper, G.: The ALARM monitoring system: A case study with two probabilistic inference techniques for belief networks. Artificial Intelligence in Medicine, 247–256 (1989)
5. Frank, E., Witten, I.H.: Using a permutation test for attribute selection in decision trees. In: ICML, pp. 152–160. Morgan Kaufmann, San Francisco (1998)
6. Good, P.: Permutation, Parametric, and Bootstrap Tests of Hypotheses, 3rd edn. Springer Series in Statistics. Springer, Heidelberg (2004)
7. Hong, F., Breitling, R.: A comparison of meta-analysis methods for detecting differentially expressed genes in microarray experiments. Bioinformatics 24, 374–382 (2008)
8. Jensen, D.: Induction with Randomization Testing: Decision-Oriented Analysis of Large Data Sets. Ph.D. thesis, Washington University (1992)
9. Jensen, D., Neville, J.: Randomization tests for relational learning. Tech. Rep. 03-05, Department of Computer Science, University of Massachusetts Amherst (2003)
10. Mehta, C.P.: Statxact: A statistical package for exact nonparametric inference. The American Statistician 45, 74–75 (1991)
11. Neville, J., Jensen, D., Friedland, L., Hay, M.: Learning relational probability trees. In: 9th ACM SIGKDD (2003)
12. Richardson, T., Spirtes, P.: Ancestral graph markov models. Annals of Statistics 30(4), 962–1030 (2002)
13. Spirtes, P., Glymour, C., Scheines, R.: Causation, Prediction, and Search, 2nd edn. MIT Press, Cambridge (2000)
14. Tillman, R.E.: Structure learning with independent non-identically distributed data. In: 26th International Conference on Machine Learning, ICML 2009 (2009)
15. Tillman, R.E., Danks, D., Glymour, C.: Integrating locally learned causal structures with overlapping variables. In: NIPS (2008)
16. Triantafyllou, S., Tsamardinos, I., Tollis, I.G.: Learning causal structure from overlapping variable sets. In: AI and Statistics (2010)
17. Tsamardinos, I., Brown, L., Aliferis, C.: The Max-Min Hill-Climbing Bayesian Network Structure Learning Algorithm. Machine Learning 65(1), 31–78 (2006)
18. Tsamardinos, I., Triantafyllou, S.: The possibility of integrative causal analysis: Learning from different datasets and studies. Journal of Engineering Intelligent Systems (to appear, 2010)
19. Tsamardinos, I., Brown, L.E.: Bounding the false discovery rate in local bayesian network learning. In: AAAI, pp. 1100–1105 (2008)