

Contents lists available at [SciVerse ScienceDirect](#)

International Journal of Approximate Reasoning

journal homepage: www.elsevier.com/locate/ijar

Conservative independence-based causal structure learning in absence of adjacency faithfulness

Jan Lemeire^{*}, Stijn Meganck, Francesco Cartella, Tingting Liu

Vrije Universiteit Brussel, ETRO Dept., Pleinlaan 2, B-1050 Brussels, Belgium

Interdisciplinary Institute for Broadband Technology (IBBT), FMI Dept., Gaston Crommenlaan 8 (Box 102), B-9050 Ghent, Belgium

ARTICLE INFO

Article history:

Received 7 September 2011

Received in revised form 12 June 2012

Accepted 12 June 2012

Available online xxx

Keywords:

Causality

Bayesian networks

Structure learning

Faithfulness

ABSTRACT

This paper presents an extension to the Conservative PC algorithm which is able to detect violations of adjacency faithfulness under causal sufficiency and triangle faithfulness. Violations can be characterized by pseudo-independent relations and equivalent edges, both generating a pattern of conditional independencies that cannot be modeled faithfully. Both cases lead to uncertainty about specific parts of the skeleton of the causal graph. These ambiguities are modeled by an f -pattern. We prove that our Adjacency Conservative PC algorithm is able to correctly learn the f -pattern. We argue that the solution also applies for the finite sample case if we accept that only strong edges can be identified. Experiments based on simulations and the ALARM benchmark model show that the rate of false edge removals is significantly reduced, at the expense of uncertainty on the skeleton and a higher sensitivity for accidental correlations.

© 2012 Elsevier Inc. All rights reserved.

1. Introduction

Independence-based algorithms for learning the causal structure from data rely on the Conditional Independencies (CIs) entailed by the system's causal structure. The *causal Markov condition* gives the CIs that follow from a causal structure that is represented by a Directed Acyclic Graph (DAG): every variable is independent of its non-effects conditional on its direct causes. Current algorithms rely on a form of, possibly relaxed version of, faithfulness. *Causal faithfulness* states that no other CIs appear in the system's probability distribution than those entailed by the causal Markov condition. Faithfulness is therefore a very 'convenient' property: all CIs tell us something about the causal structure. Under faithfulness, the Markov equivalence class can be learned, which is the set of all DAGs representing the CIs of the system. Violation of faithfulness means that there are non-Markovian CIs, i.e., CIs not following from the Markov condition.

Unfaithfulness holds only for specific parameterizations of the conditional probability distributions of Bayesian networks. Meek [1] showed that for discrete Bayesian networks, it leads to non-trivial constraints which are polynomials in the parameters. Spirtes et al. [2, p. 41] showed the same result for linear models. The validity of causal faithfulness is supported by the 'Lebesgue measure zero argument' [3], which states that the chance of randomly picking a parameterization of a Bayesian network which results in non-Markovian CIs has measure zero. But in near-to-unfaithful situations probability distributions are close to unfaithful distributions, such that a test for independence which has to rely on a finite sample will not be able to identify the dependencies correctly. The Lebesgue measure zero argument does not hold here, since the ϵ -regions around unfaithful situations do not have Lebesgue measure zero. Moreover, recently the validity of causal faithfulness has been challenged based on the argument that patterns (here: the parameter constraints leading to unfaithfulness) are not rare in nature [4].

^{*} Corresponding author at: Vrije Universiteit Brussel, ETRO Dept., Pleinlaan 2, B-1050 Brussels, Belgium. Tel.: +32 26291679.

E-mail addresses: jan.lemeire@vub.ac.be (J. Lemeire), stijn.meganck@vub.ac.be (S. Meganck), francesco.cartella@vub.ac.be (F. Cartella), tingting.liu@vub.ac.be (T. Liu).

Zhang and Spirtes [5] showed that only in cases of *triangle unfaithfulness* violations of faithfulness are *undetectable*. This happens when the true probability distribution is not faithful to the true causal DAG, but is nonetheless faithful to some other DAG. In those cases, the CIs do not give enough evidence to learn the correct DAG. We will therefore have to rely on triangle faithfulness. Triangle faithfulness will be discussed in more detail in the Section 2.4. Then, violations of faithfulness are detectable in the sense that the true probability distribution is not faithful to any DAG. It means that there exist several DAGs that each explains a subset of the CIs. This leads to additional ambiguities about the DAG. Currently, these ambiguities are detected for a part of faithfulness violations, namely orientation unfaithfulness [6]. In this paper we extend the treatment to adjacency unfaithfulness.

Ramsey et al. [6] showed that we only need adjacency faithfulness and orientation faithfulness to learn the correct equivalence class. *Adjacency faithfulness* states that any two adjacent variables do not become independent when conditioned on some other (possible empty) set of variables. It is necessary to recover the correct skeleton of the true DAG. *Orientation faithfulness* is necessary for finding the correct orientations. Ramsey et al. [6] extended the well-known PC algorithm to detect violations of orientation faithfulness. Violations lead to specific ambiguous parts of the DAG, in which no decision on the orientation can be taken. The modified PC algorithm is given in the next section. In this paper we apply the same idea for handling violations of adjacency faithfulness. Under triangle faithfulness, they can be identified by two patterns: *pseudo-independent relations* and *equivalent edges*. These patterns will lead to parts of the model in which no decision can be taken on the correct skeleton.

The following section recalls the important aspects of independence-based causal structure learning and faithfulness. Sections 3 and 4 discuss the two cases in which adjacency faithfulness is violated. That they comprise almost all cases of adjacency unfaithfulness is proven in Section 5. Section 6 analyzes the ambiguities following from adjacency unfaithfulness and defines the equivalence class. Section 7 discusses how violations can be treated in the finite sample case. Based on this analysis, our ACPC algorithm is presented and proven to be correct in Section 8. Related work is discussed in Section 9. Finally, the experimental results are presented in Section 10.

2. Independence-based causal structure learning

We first review the necessary state-of-the-art.

2.1. DAGs and independencies

A graphical causal model consists of a Directed Acyclic Graph (DAG) in which the edges represent the direct causal relations among the variables. A direct cause of a variable, denoted as a parent in the graph, is a cause that produces the state of the effect variable not through some other observed variables. The direct causes or parents affect the mechanism responsible for generating the value of the effect variable. The mechanism for setting X_i can be described by the Conditional Probability Distribution (CPD) $P(X_i | \text{Parents}(X_i))$ with $\text{Parents}(X_i)$ denoting the parents of X_i . The CPDs taken together describe the whole system under study, namely by the Joint Probability Distribution (JPD) $P(X_1, \dots, X_n)$ with X_1, \dots, X_n the known variables. The CPDs form a *factorization* of the JPD:

$$P(X_1, \dots, X_n) = \prod_i^n P(X_i | \text{Parents}(X_i)). \quad (1)$$

A Bayesian network describes any (not necessarily the causal one) factorization of a JPD according to the above equation. It consists of a Directed Acyclic Graph (DAG), describing the parents of each variable and a set of Conditional Probability Distributions (CPDs), one for each variable. A graphical causal model is therefore a special Bayesian network; one for which the DAG describes the direct causal relations.

We review some graph terminology. Two variables are called *adjacent* in a graph if there is an edge in the graph that connects both variables. A *path* is a set of consecutive edges (independent of the direction) without visiting a variable more than once. A *collider* on a path is a variable for which both edges of the path of which it is an endpoint, point towards it. X is called an ancestor of Y if there is a directed path from X to Y (having all its edges pointing towards Y). Y is called a descendant of X .

One can view the DAG as describing qualitative properties about the system, namely its ‘structure’, whereas the CPDs describe the quantitative properties. An important property is that the DAG entails (conditional) independencies. These independencies will form the basis of identifying the system’s causal structure from pure observational data. The independencies are given by the Markov condition: any variable is independent from all its non-descendants conditioned on its parents.

Besides Markov, the *graphoid properties* describe the general relations among conditional independence statements [7]. If two variables X and Y are independent (dependent) in P we write $X \perp\!\!\!\perp_P Y$ ($X \not\perp\!\!\!\perp_P Y$). Since in the text there can be no confusion about the distribution we omit the subscript P . Conditional independence (dependence) is written as $X \perp\!\!\!\perp Y | Z$ ($X \not\perp\!\!\!\perp Y | Z$).

Properties 1 (Graphoids). Let P be a distribution over a set of variables \mathbf{V} . The independencies that follow from P must satisfy the following independent conditions that are defined for all disjoint subsets \mathbf{X} , \mathbf{Y} , \mathbf{Z} and \mathbf{W} .

Symmetry: $\mathbf{X} \perp\!\!\!\perp \mathbf{Y} | \mathbf{Z} \Leftrightarrow \mathbf{Y} \perp\!\!\!\perp \mathbf{X} | \mathbf{Z}$

Decomposition: $\mathbf{X} \perp\!\!\!\perp \mathbf{Y}, \mathbf{W} | \mathbf{Z} \Rightarrow \mathbf{X} \perp\!\!\!\perp \mathbf{Y} | \mathbf{Z} \ \& \ \mathbf{X} \perp\!\!\!\perp \mathbf{W} | \mathbf{Z}$

Weak Union: $\mathbf{X} \perp\!\!\!\perp \mathbf{Y}, \mathbf{W} | \mathbf{Z} \Rightarrow \mathbf{X} \perp\!\!\!\perp \mathbf{Y} | \mathbf{Z}, \mathbf{W}$

Contraction: $\mathbf{X} \perp\!\!\!\perp \mathbf{Y} | \mathbf{Z} \ \& \ \mathbf{X} \perp\!\!\!\perp \mathbf{W} | \mathbf{Z}, \mathbf{Y} \Rightarrow \mathbf{X} \perp\!\!\!\perp \mathbf{Y}, \mathbf{W} | \mathbf{Z}$

If P is strictly positive (has no zero values), the following condition holds as well:

Intersection: $\mathbf{X} \perp\!\!\!\perp \mathbf{Y} | \mathbf{Z}, \mathbf{W} \ \& \ \mathbf{X} \perp\!\!\!\perp \mathbf{W} | \mathbf{Y}, \mathbf{Z} \Rightarrow \mathbf{X} \perp\!\!\!\perp \mathbf{Y}, \mathbf{W} | \mathbf{Z}$

Note that single stochastic variables are denoted by capital letters, sets of variables by boldface capital letters.

The Markov conditions together with the graphoid properties entail a set of independencies which depends on the given DAG. This set is given by *d-separation*. Two disjoint sets of variables \mathbf{X} and \mathbf{Y} are *d-separated* by \mathbf{Z} if every path between variables in \mathbf{X} and variables in \mathbf{Y} is blocked by \mathbf{Z} . A path is said to be blocked by \mathbf{Z} if it contains a collider $\rightarrow \cdot \leftarrow$ whose descendants are not in \mathbf{Z} or a non-collider $\rightarrow \cdot \rightarrow$ or $\leftarrow \cdot \rightarrow$ or $\leftarrow \cdot \leftarrow$ that is in \mathbf{Z} [8]. We call a non-blocked path an *active path*. Two variables that are not *d-separated* are called *d-connected*. We say that a conditioning variable *closes a path* if it turns an active path into a blocked one (a non-collider on the path). We say that a variable *opens a path* if it turns a blocked path into an active one (a collider on the path or a descendant of a collider). We call a DAG Markovian for a distribution if all CIs following from the Markov condition (given by *d-separation*) are present in the distribution (i.e., it is an I-MAP).

Recall that a distribution P is faithful to a DAG G if no conditional independence holds in P unless entailed by the Markov condition on G . Besides faithfulness, *minimality* (MIN) is also a basic condition: elimination of an edge leads to a Bayesian network which violates the Markov condition. Formally written it means that

$$\forall X, Y \in \mathbf{V} \text{ which are adjacent in } G : X \not\perp\!\!\!\perp Y | OthPa(X - Y), \quad (2)$$

where \mathbf{V} is the set of all variables and $OthPa(X - Y)$ of edge $X - Y$ is defined as $Parents(Y) \setminus X$ if X is parent of Y , otherwise it is $Parents(X) \setminus Y$. *OthPa* is short for ‘other parents’. This is a local minimality criterion. We require that every conditioning variable is necessary in the CPD. Omission of a variable breaks the correctness of the factorization.

2.2. The PC algorithm

Based on faithfulness, a straightforward algorithm can be designed that retrieves the causal structure based on the CIs [2]. The causal graph is constructed in two phases. The first phase, called *adjacency search*, learns the undirected graph. The second phase tries to orient the edges. The result of the algorithm is a set of observationally indistinguishable models, the so-called *Markov equivalence class*, and is proved to be correct for distributions that are faithful to some DAG. Note that the information provided by the CIs can be combined with expert knowledge [9].

The construction of the undirected graph, the first phase, is based on the property that two nodes are adjacent if they remain dependent by conditioning on every set of nodes that does not include either node. The algorithm starts with a completely connected, undirected graph. There is an edge between every pair of nodes. It then removes edges for each independency that is found. The number of nodes in the conditioning set is gradually increased up to a certain maximal number, called the *depth* of the search. If for every pair A, B , one should test on all subsets of $\mathbf{V} \setminus \{A, B\}$, this would result in an exponential number of tests to perform. One can however limit the number of subsets to be tested. In order to detect adjacency, it is sufficient to test for conditional independencies of A and B given subsets of variables adjacent to A and subsets of variables adjacent to B that are on undirected paths between A and B . It relies on the following property [2].

Lemma 1. If a distribution P over variables \mathbf{V} is faithful to a Bayesian network with DAG G , and $X, Y \in \mathbf{V}$, then: If X and Y are *d-separated* by any subset of $\mathbf{V} \setminus \{X, Y\}$, then they are *d-separated* either by $Parents(X)$ or $Parents(Y)$.

The orientation phase is based on the identification of v-structures¹ $\rightarrow \cdot \leftarrow$, for which two nodes are independent, but become dependent conditional on a third node. In that case, both nodes are causes of the third node and are oriented towards the third node. Recall that for all three other possible orientations of the three nodes, $\rightarrow \cdot \rightarrow$ or $\leftarrow \cdot \rightarrow$ or $\leftarrow \cdot \leftarrow$, the opposite is true, the two nodes are initially dependent, but become independent by conditioning on the third node. We call these three structures *Markov chains*.

The Markov equivalence class is represented by a Partially Directed Acyclic Graph (PDAG) in which some edges are not oriented. They can be oriented in any direction as long as no v-structures are created nor cycles. It follows that all DAGs of the class contain the same v-structures.

¹ We call a v-structure an unshielded triple (three nodes connected by exactly two edges) which is oriented as a collider.

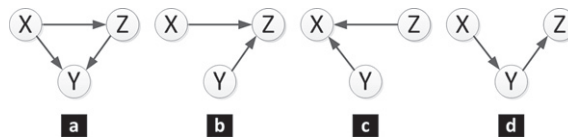


Fig. 1. Triangle (a) and the three faithful models following from triangle unfaithfulness: $X \perp\!\!\!\perp Y$ (b, TRUFF1), $Y \perp\!\!\!\perp Z$ (c, TRUFF2) and $X \perp\!\!\!\perp Z \mid Y$ (d, TRUFF3).

2.3. The conservative PC algorithm

It is proven by Ramsey et al. [6] that Faithfulness is too strict for the correctness of the PC algorithm. For the first phase we only need Adjacency Faithfulness, for the second phase we only need Orientation Faithfulness.

Definition 1 (*Adjacency faithfulness condition*). Given a set of variables \mathbf{V} whose causal structure can be represented by a DAG G , if two variables X, Y are adjacent in G , then they are dependent conditioned on any subset of $\mathbf{V} \setminus \{X, Y\}$.

Given three nodes X, Y, Z in a graph G , the triple $\langle X, Y, Z \rangle$ is called an *unshielded triple* if X and Z are adjacent to Y but X is not adjacent to Z .

Definition 2 (*Orientation faithfulness condition*). Given a set of variables \mathbf{V} whose causal structure can be represented by a DAG G , let $\langle X, Y, Z \rangle$ be any unshielded triple in G :

- If $X \rightarrow Y \leftarrow Z$ (case of a v-structure), then X and Z are dependent given any subset of $\mathbf{V} \setminus \{X, Z\}$ that contains Y .
- otherwise (case of a Markov chain), X and Z are dependent conditional given any subset of $\mathbf{V} \setminus \{X, Z\}$ that does not contain Y .

In this way, faithfulness is decomposed into three disjoint components: Adjacency Faithfulness, Orientation Faithfulness and a third one consisting of the remaining CI statements. One could say that the first is about variables at distance one in the true graph, where the *distance* between two nodes in a graph is the length of their shortest path. Adjacency Faithfulness requires that every d -connection between two adjacent variables (having distance one) is paralleled with a dependence. It ensures that the right adjacencies are found. The second component, Orientation Faithfulness, says that for variables at distance two a d -connection is paralleled with a dependence except for d -connections following from conditioning on descendants of colliders. This property ensures that every unshielded triple can be qualified as being a v-structure or a Markov chain. The third component is about dependencies between variables “at further distance”. Such dependencies are not needed for correctness of the PC algorithm.

Ramsey et al. [6] extended the PC algorithm to capture violations of orientation faithfulness. The basic case of violation is when for unshielded collider $\langle X, Y, Z \rangle$ $X \perp\!\!\!\perp Z$ and $X \perp\!\!\!\perp Z \mid Y$. In such cases, the orientation rules do not apply. The orientation remains unknown. We remain with 4 possible structures. The equivalence class is extended by this additional ambiguity. The unshielded triple $\langle X, Y, Z \rangle$ is indicated as being an *unfaithful triple*.

The Conservative PC algorithm (CPC) is given in Algorithm 1. $Adj(G, X)$ denotes the set of nodes adjacent to X in graph G . Orientation Faithfulness is tested by checking for each unshielded triple that one of both sets of dependencies of orientation faithfulness hold: either the dependencies following from a v-structure, either the dependencies following from a Markov chain. Step 3 (S3) consists of extensions to the original PC algorithm in which Orientation-Faithfulness is tested. Edges of an unshielded triple are not oriented if a failure is detected but are indicated as unfaithful, as shown in Fig. 3(a). An *e-pattern* is a partially-oriented DAG in which some triples are denoted as unfaithful. An unfaithful triple is a triple which we cannot qualify as being a v-structure or a Markov chain. An e-pattern thus describes a set of DAGs which will be larger than the Markov equivalence class. The algorithm returns a correct e-pattern under Causal Sufficiency (there are no common latent causes) and Adjacency-Faithfulness. We will also assume Causal Sufficiency in the following.

2.4. Undetectable violation of faithfulness

An *undetectable violation of faithfulness* happens when the true probability distribution is not faithful to the true causal DAG, but is nonetheless faithful to some other DAG. In such cases, an equivalence class is learned which does not contain the true causal DAG. Since the (wrong) DAGs in this class are faithful, there is no evidence in the form of CIs to refute these. Undetectable violations of faithfulness only happen by violations of the *triangle faithfulness condition* [5]. It states that given a set of variables \mathbf{V} whose true causal DAG is G , let X, Y, Z be any three variables that form a triangle in G , then:

1. If Y is a non-collider on the path $\langle X, Y, Z \rangle$, then X, Z are dependent conditional on any subset of $\mathbf{V} \setminus \{X, Z\}$ that does not include Y .
2. If Y is a collider on the path $\langle X, Y, Z \rangle$, then X, Z are dependent conditional on any subset of $\mathbf{V} \setminus \{X, Z\}$ that includes Y .

Algorithm 1. The CPC algorithm

S1 Start with the complete undirected graph U on the set of variables V .

Phase I **Adjacency search.**

S2 $n = 0$;

repeat

For each pair of variables A and B that are adjacent in (the current) U , check through the subsets of $Adj(U, A) \setminus \{B\}$ and the subsets of $Adj(U, B) \setminus \{A\}$ that have exactly n variables. For all such subsets S check independency $A \perp\!\!\!\perp B | S$. If independent, remove the edge between A and B in U ;

$n = n + 1$;

until for each ordered pair of adjacent variables A and B , $Adj(U, A) \setminus \{B\}$ has less than n elements.

Phase II **Orientation.**

S3 Let G be the undirected graph resulting from step S2. For each unshielded triple (A, B, C) in G , check all subsets of A 's potential parents (nodes that are adjacent to A but are not A 's children) and of C 's potential parents:

- (a) If B is NOT in any such set conditional on which A and C are independent, orient the triple as a collider: $A \rightarrow B \leftarrow C$;
- (b) If B is in all such sets conditional on which A and C are independent, leave $A - B - C$ as it is, i.e., a non-collider;
- (c) Otherwise, mark the triple as “unfaithful” by underlining the triple, $A - \underline{B} - C$.

S4 Execute the orientation rules given in [1], but not on unfaithful triples.

To illustrate triangle unfaithfulness (TRUFF), consider the triangle shown in Fig. 1(a). There are three ways to violate triangle faithfulness for this DAG:

- (TRUFF1) $X \perp\!\!\!\perp Y$ gives faithful model $X \rightarrow Z \leftarrow Y$, shown in Fig. 1(b);
- (TRUFF2) $Y \perp\!\!\!\perp Z$ gives faithful model $Y \rightarrow X \leftarrow Z$, shown in Fig. 1(c);
- (TRUFF3) $X \perp\!\!\!\perp Z | Y$ gives faithful model $X \rightarrow Y \rightarrow Z$, shown in Fig. 1(d).

Based on Zhang and Spirtes [5]’s theorem that under triangle faithfulness all cases of unfaithfulness can be detected, we assume triangle faithfulness and investigate the other cases of adjacency unfaithfulness. We start by discussing the two major cases in which adjacency faithfulness is violated: Pseudo-Independent Relations (PIRs) and information equivalences. Then, in Section 5 we prove that both cases cover almost all cases of adjacency faithfulness violation. There is only one other, special case that can happen.

3. Pseudo-independent relations (PIRs)

Consider causal structure $X \rightarrow Y \leftarrow Z$ with additionally $X \perp\!\!\!\perp Y$. For Minimality (Eq. 2), at least $X \not\perp\!\!\!\perp Y | Z$ should hold. X contains information about Y which only becomes apparent when conditioned on Z . Because of the marginal independence $X \perp\!\!\!\perp Y$, we call $X \rightarrow Y$ a Pseudo-Independent Relation. This entails a specific parameterization of $P(Y|X, Z)$ and $P(Z)$:

$$P(Y|x) = P(Y) \quad \forall x \tag{3}$$

$$\Leftrightarrow \sum_z P(Y|x, z).P(z) = P(Y) \quad \forall x \tag{4}$$

The PC algorithm errs on $X \rightarrow Y$, it removes the edge by the marginal independence $X \perp\!\!\!\perp Y$. The algorithm would only return an edge between Y and Z . The DAG is clearly not Markovian. This error can be detected by testing for dependency $X \not\perp\!\!\!\perp Y | Z$ which would reveal that X and Y might be adjacent. On the other hand, the CI pattern $X \perp\!\!\!\perp Y$ and $X \not\perp\!\!\!\perp Y | Z$ is equally well represented by the graph $X \rightarrow Z \leftarrow Y$. To conclude, we end up with two equivalent v-structures.

A special case is given a pseudo-independent model [10] in which all three variables are pairwise independent but become dependent when conditioned on the third variable. Consider three binary variables X, Y, Z with causal structure $X \rightarrow Z \leftarrow Y$. Consider that X and Z are fair coin tosses ($P(X = 0) = 0.5$ and $P(Z = 0) = 0.5$) and influence Y according to a logical XOR relation, i.e.,

$$Y = X \oplus Z.$$

This yields $X \perp\!\!\!\perp Y$ and $Y \perp\!\!\!\perp Z$. We end up with three equivalent v-structures.

Please cite this article in press as: J. Lemeire et al., Conservative independence-based causal structure learning in absence of adjacency faithfulness, Int. J. Approx. Reason (2012), <http://dx.doi.org/10.1016/j.ijar.2012.06.004>

More generally, we define a PIR as follows. A strict d -separation, denoted as $[X \perp Y | Z]$, is a d -separation which gives a d -connection for any proper subset of Z .

Definition 3 (*Pseudo-independent relation*). Given G and P compatible with G for which MIN holds and either $X \rightarrow Y$ or $X \leftarrow Y$ in the correct causal structure. Edge $X - Y$ is called a Pseudo-Independent Relation (PIR) if X and Y are independent conditional on a minimal subset $W \subseteq V \setminus (\{X, Y\})$ that d -separates X and Y in $G \setminus X - Y$:

$$X \perp\!\!\!\perp Y | W \text{ and } [X \perp_{G \setminus X - Y} Y | W]. \quad (5)$$

Note that with $\perp_{G \setminus X - Y}$ we indicate the graph in which the d -separation holds. By minimality, we have $X \perp\!\!\!\perp Y | \text{OthPa}(X - Y)$. So there is at least one set that renders X and Y dependent. We call a set S a *depset* of the PIR if $[X \perp\!\!\!\perp Y | W, S]$.

The following theorem will be used in the proof of Theorem 3.

Theorem 1. Two variables X and Y of a PIR $X \rightarrow Y$ can only become dependent when conditioned on variables that have an active path to Y via a variable of a depset or variables that have a separate active path to both X and Y .

Proof. Dependence can only occur for variables that are d -connected to either X or Y (expressed by weak union). When a conditioning variable A is a descendant of Y , we have $A \perp\!\!\!\perp X | Y$ since A has no separate active path to X . Together with $X \perp\!\!\!\perp Y$ it follows that $X \perp\!\!\!\perp A$ (contraction). By weak union, $X \perp\!\!\!\perp Y | A$ follows. The same holds for variables that are d -connected to X . \square

4. Information equivalences

Basically, PIRs generate non-Markovian *marginal* independencies and information equivalences generate non-Markovian *conditional* independencies. Here we consider the second.

The typical case of information equivalence happens in the presence of *deterministic* causal relations. Consider variables X, Y, Z to be connected by the causal chain $X \rightarrow Y \rightarrow Z$ with a deterministic relation between Y and X given by function $Y = f(X)$. Due to this function, Y is conditionally independent of Z given X :

$$Y \perp\!\!\!\perp Z | X.$$

By the functional dependence, X contains all information about Y . So Y is independent from any other variable given X . Deterministic relations generate additional independences beyond the Markov condition. These CIs are given by an extension of d -separation, called D -separation [11].

The PC and CPC algorithms fail in such cases. In the example, we have $Y \perp\!\!\!\perp Z$ and $Y \perp\!\!\!\perp Z | X$ which falsely suggest that X separates Y from Z and that edge $Y \rightarrow Z$ can be removed. Moreover, $X \perp\!\!\!\perp Z | Y$ (following from d -separation) suggests that there is no edge between X and Z either. Removal of both edges results in a non-Markovian DAG in which neither X nor Y is related to Z although that both contain information about Z .

Both conditional independence statements together with $X \perp\!\!\!\perp Z$ and $Y \perp\!\!\!\perp Z$ cannot be faithfully represented by any DAG. The true DAG is not faithful due to the independence following from the function. On the other hand is the following DAG also Markovian but not faithful: $Y \leftarrow X \rightarrow Z$. Both independencies imply a violation of the Intersection Condition (see the Graphoid Properties in Section 2.1). We call X and Y *information equivalent* with respect to Z , since both variables contain the same information about Z . By the information equivalence, we end up with two equivalent structures: $X \rightarrow Y \rightarrow Z$ and $Y \leftarrow X \rightarrow Z$. Either X or Y can be connected to Z . We call $X \rightarrow Z$ and $Y \rightarrow Z$ *equivalent edges*, since both equally well represent the dependencies.

This is the case if we know the causal ordering: X is the cause of Y and both X and Y are causes of Z . The ambiguity lies in which one is the direct cause of Z : X or Y . In absence of information about the orientations, the arrows can be reversed except that no v -structures are permitted; $X \rightarrow Y \leftarrow Z$ and $Y \rightarrow X \leftarrow Z$ are prohibited. We end up with 6 possible causal structures.

More generally, we define an information equivalence as follows:

Definition 4 (*Information equivalence*). Two sets of variables X and Y are called **information equivalent** with respect to Z , the *reference variable*, if there exists a subset $W \subseteq V \setminus (X \cup Y \cup \{Z\})$ for which

- $X \perp\!\!\!\perp Z | W$ and $Y \perp\!\!\!\perp Z | W$
- $Y \perp\!\!\!\perp X | W$
- $X \perp\!\!\!\perp Y | W$

A *conditional information equivalence* is one for which the equivalence only holds for a non-empty conditioning set W .

It is shown in [12] that information equivalences happen by a larger class than just deterministic relations. They appear by parameterizations that lead to so-called *equivalent partitions*. We review the theory about equivalent partitioning and recall the basic theorem in Appendix B.

5. Violation of adjacency faithfulness

The theorem of this section proves that PIRs and information equivalences are the main cases of detectable violations of adjacency unfaithfulness, besides a third type that can be identified by what we call 2-1 CI patterns.

Definition 5 (2-1 CI pattern). We call a 2-1 CI pattern a tuple of variables X, Y, Z and disjoint sets $\mathbf{U}, \mathbf{W} \subset \mathbf{V} \setminus \{X, Y, Z\}$, where \mathbf{U} is not empty, for which the following statements hold in the distribution:

- $[X \perp\!\!\!\perp Y | Z, \mathbf{W}]$; and
- $[Z \perp\!\!\!\perp Y | X, \mathbf{U}, \mathbf{W}]$; and
- $X \not\perp\!\!\!\perp Y | Z, \mathbf{U}, \mathbf{W}$.

A strict independency, denoted as $[X \perp\!\!\!\perp Y | Z]$, is a CI for which no proper subset of the conditioning set leads to an independence. The pattern of CIs resembles the pattern of an information equivalence, X gets independent from Y by Z and Z gets independent from Y by X . The difference lies in the asymmetry regarding \mathbf{U} . For the second independence you need to condition on some other variables \mathbf{U} but conditioning on \mathbf{U} does not render X independent from Y .

We will analyze 2-1 CI patterns more in detail after having characterized detectable violations of adjacency faithfulness.

5.1. Characterization

Theorem 2. Under triangle faithfulness and minimality, violations of adjacency faithfulness by strict independencies are limited to pseudo-independent relations, information equivalences and 2-1 CI patterns.

The proof is given in [Appendix A](#).

Informally, the proof is as follows. Adjacency unfaithfulness happens for each independence of two adjacent variables X and Y . By minimality, there is a conditional dependency for each edge $X - Y$; namely when conditioned on all other parents. But when not conditioned on all other parents there might be an independence. This results in a PIR. On the other hand, a conditional independence might show up when conditioned on some other variables. Since X and Y are adjacent, not all paths between X and Y can be cut and hence there is a path from each variable of the conditioning set to X (Y) that crosses Y (X). It follows that each variable of the conditioning set can be separated by X (Y) by conditioning on Y (X). This leads to an information equivalence or a 2-1 CI pattern.

Note that the theorem is restricted to strict independencies $[X \perp\!\!\!\perp Y | Z]$ since a non-strict independency means that a violation exists with fewer conditioning variables. This can mean either of the following:

- A subset $\mathbf{U} \subset \mathbf{Z}$ of the conditioning set is not responsible for the violation, hence it is not necessary to consider the subset to analyze the unfaithful case. We therefore only have to consider $\mathbf{Z} \setminus \mathbf{U}$. On the other hand, it is possible that a proper subset \mathbf{W} of \mathbf{Z} which includes \mathbf{U} ($\mathbf{U} \subset \mathbf{W} \subset \mathbf{Z}$) results in a violation. This would be given by another strict independency and should therefore be analyzed separately.
- There is a variable which, given strict independency $[X \perp\!\!\!\perp Y | Z_1]$, restores the dependency $X \not\perp\!\!\!\perp Y | Z_1, Z_2$ and another variable deleting the restoration by $X \perp\!\!\!\perp Y | Z_1, Z_2, Z_3$. For instance when Z_2 opens a path between X and Y but Z_3 closes the path again. Both variables are unnecessary to characterize the unfaithful case.

5.2. 2-1 CI patterns

2-1 CI patterns cannot be represented faithfully. To analyze this let us assume that \mathbf{U} has one element U and assume that \mathbf{W} is empty. The first independence ($[X \perp\!\!\!\perp Y | Z]$) suggests that the path connecting X and Y goes via Z , while the second independence ($[Z \perp\!\!\!\perp Y | X, U]$) suggests that the path from Z to Y goes via X and there is a path via U . The question is thus whether X or Z is adjacent to Y . An example of adjacency of X and Y is shown in [Fig. 2\(b\)](#). An example of adjacency of Z and Y is (a). Note that in these examples we assume that there is no other variable that separates the three variables, that is, we are certain that some of the three variables are adjacent.

The question is to what extent the different models that lead to 2-1 CI patterns are identifiable and, if not identifiable, which ambiguities follow. The following analysis, although incomplete, shows that most cases are identifiable and only for very specific distributions are there equivalent models which are indistinguishable based on CI information. Based on this analysis we will exclude 2-1 CI patterns as a source of ambiguity.

Models (a), (b) and (c) of [Fig. 2](#) are typical examples of structures that for a specific parameterization generate $[X \perp\!\!\!\perp Y | Z]$ and $[Z \perp\!\!\!\perp Y | X, U]$. For model (a) also CI $X \perp\!\!\!\perp Y | Z, U$ holds (by d -separation). This CI is however not necessarily present in model (b). If it is present we have a normal symmetric information equivalence which will result in an equivalence of both models as we will show in [Section 6.4](#). Edges $X \rightarrow Y$ and $Z \rightarrow Y$ are interchangeable. On the other hand, if $X \not\perp\!\!\!\perp Y | Z, U$ then model (a) is not Markovian. Model (a) cannot lead to a 2-1 CI pattern. U should open a path between X and Y to render them dependent. This is true for model (c). Thus, a 2-1 CI pattern could come from models (b) or (c). Both cases are, however, not equivalent in general. In model (b) X can be d -separated from U and in model (c) Z can be d -separated from U . Identification

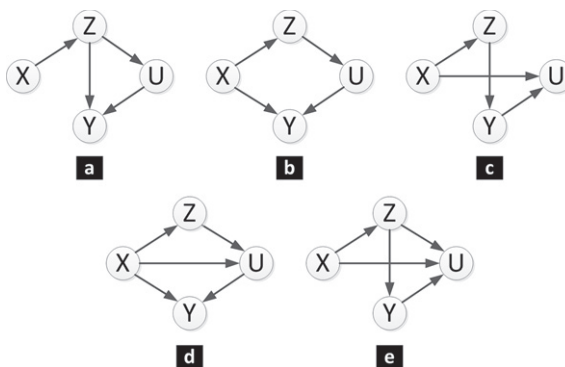


Fig. 2. Five structures of which only (b) and (c) could generate a 2-1 CI pattern under triangle faithfulness.

is not certain if we add an edge in both models so that the d -separation does not hold any more. This results in models (d) and (e) respectively. However, by assuming triangle faithfulness, these cases can be excluded. Under triangle faithfulness, $X \perp\!\!\!\perp Y|Z$ must hold for (d) (otherwise TRUFF3) and $Z \perp\!\!\!\perp Y|X, U$ must hold for (e) (otherwise TRUFF1). To conclude, for these basic cases, 2-1 CI patterns seem identifiable.

6. Ambiguities following from adjacency unfaithfulness

Now that we know that adjacency unfaithfulness comes from PIRs and information equivalences, the question is which ambiguities follow from them. In this section we define the equivalence class of graphs. When adjacency faithfulness holds, the only ambiguities lie in the orientation of some edges. The Markov equivalence class and the e-pattern (in absence of orientation faithfulness) are expressed by a partially-oriented acyclic graph. In absence of adjacency faithfulness, additional ambiguities follow.

6.1. Adjacency ambiguities

We will prove that ambiguities following from adjacency unfaithfulness are *local*, in the sense that some edges can be replaced by some other edges. We say that edges or sets of edges that are equivalent. The equivalence class will contain graphs in which some edges are replaced by others. We, however, have to exclude cases in which one of the equivalent edge sets creates a v-structure that is not present if another edge set is chosen. Since v-structures have a complete different CI-pattern, the graph will not be equivalent with respect to the CIs it represents. Hence, differences in v-structures make the ambiguity identifiable unless we deal with unfaithful triples. As discussed in Section 2.3, the CIs of an unfaithful triple do not reveal whether it is a Markov chain or a v-structure.

Definition 6 (Equivalent edge sets). Edge sets $\{E_1 \dots E_k\}$ are called *equivalent edge sets* if and only if for all DAGs G :

$$\begin{aligned} & \exists i \in [1..k] : G \cup E_i \text{ is Markovian for } P \\ \Rightarrow & \forall j \in [1..k] \text{ for which } G \cup E_j \text{ gives the same orientation as } G \cup E_i \text{ for all faithful triples:} \\ & G \cup E_j \text{ is Markovian for } P \end{aligned} \tag{6}$$

and there exists for each $i \in [1..k]$ at least one graph G which is not Markovian but for which $G \cup E_i$ is Markovian.

Note that the condition trivially holds when G is Markovian or when $G \cup E_1 \cup \dots \cup E_k$ is not Markovian. By adding edges to a Markovian DAG, it stays Markovian; by deleting edges from a non-Markovian DAG, it stays non-Markovian. The non-trivial cases are when G is not Markovian and $G \cup E_i$ is Markovian for some i .

6.2. Assumptions

To simplify the rest of the discussion, we exclude complex cases of adjacency unfaithfulness which rarely happen. First, for each PIR we assume the existence of a depset with one element.

Assumption 1. If X and Y is a PIR, there exists a $Z \in \text{OthPa}(X - Y)$ such that $X \perp\!\!\!\perp Y|Z$.

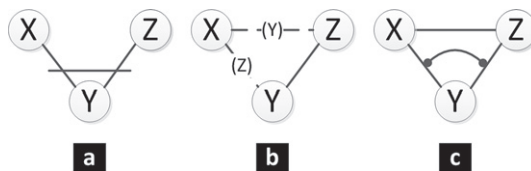


Fig. 3. The three cases of ambiguity: (a) an unfaithful triple by violation of orientation faithfulness for unshielded triple (A, B, C) , (b) PPIRs when $X \perp\!\!\!\perp Y$ in model $X \rightarrow Y \leftarrow Z$ and (c) equivalent edges when $X \perp\!\!\!\perp Y | Z$ in model $X \rightarrow Y \rightarrow Z$.

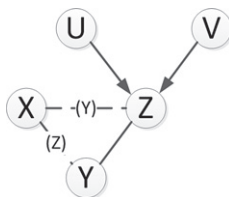


Fig. 4. A PIR-pattern which can be identified by an identifiable v-structure.

Secondly, we exclude context-dependence of information equivalences. We define a *context-independent* information equivalence as an information equivalence given by $X \perp\!\!\!\perp Y | Z, \mathbf{U}$ and $Z \perp\!\!\!\perp Y | X, \mathbf{U}$ for which for all non-empty subsets $\mathbf{W} \subseteq \mathbf{V} \setminus (\mathbf{U} \cup \{X, Y, Z\})$:

$$X \perp\!\!\!\perp Y | Z, \mathbf{U}, \mathbf{W} \Leftrightarrow Z \perp\!\!\!\perp Y | X, \mathbf{U}, \mathbf{W} . \tag{7}$$

Assumption 2. All information equivalences are context-independent.

Finally we assume identifiability of 2-1 CI patterns. As discussed in Section 5.2, 2-1 CI patterns lead to ambiguities only in very special cases.

Assumption 3. A 2-1 CI pattern is identifiable, in the sense that the correct edges between the nodes X, Y and Z of the pattern can be identified by the CIs of the structure.

6.3. Equivalent v-structures

As discussed in Section 3, the CI pattern of a PIR is the same as that of a v-structure. Hence, a PIR leads to two equivalent structures that can explain all CIs. We describe this pattern by connecting variables which are marginally independent but have a depset by a special edge: a Potential PIR (PPIR). A PPIR is written as $X - (Z) - Y$ and graphically denoted by a dashed edge annotated with the depset, as shown in Fig. 3(b). A PPIR can thus be a PIR or be part of a v-structure (in which case the variables are not adjacent). A PIR can be identified from a structure with 2 PPIRs if the orientation of some of the other edges can be identified. Fig. 4 shows a PIR with identifiable v-structure $U \rightarrow Z \leftarrow V$. When considering both equivalent structures, one has incoming edges into Z , the other not. If some of the triples with Z and U/V and X/Y is faithful, one of both equivalent structures can be excluded. The same can be done for identifiable v-structure $U \rightarrow Y \leftarrow V$. Equivalence of v-structures coming from PIRs thus holds up to identifiable v-structures containing Y or Z .

We define *equivalent v-structures* as edge set $\{X \rightarrow Y, Z \rightarrow Y\}$ which is equivalent to edge set $\{X \rightarrow Z, Y \rightarrow Z\}$.

Theorem 3. If $X - Y$ is a PIR with depset containing one variable Z , it generates equivalent v-structures.

The proof is given in Appendix A.

6.4. Information equivalent edges

Next we consider the ambiguities following from information equivalences. The result of an information equivalence is a set of equivalent structures in which one edge can be replaced by another. We call them *equivalent edges*. Equivalent edges are linked with an arc with a bullet at each end, as shown in Fig. 3(c).

Theorem 4. If X and U are information equivalent with respect of Y when conditioned on \mathbf{S} and this information equivalence is context-independent, then edges $X - Y$ and $U - Y$ are equivalent edge sets.

The proof is given in Appendix A.

6.5. F-pattern

We represent the class of equivalent graphs with an *f-pattern*. An *f-pattern* contains a mixture of undirected and directed edges, as well as underlinings for unshielded triples that are unfaithful, edges that are connected to indicate that they are information equivalent and edges which are annotated with their depset (PPIRs). A DAG is represented by an *f-pattern* if every edge is present in the pattern as a normal edge or an equivalent edge or a PPIR. If it is an equivalent edge, no other of the equivalent edges might be present in the DAG. If it is a PPIR, it should be part of a *v-structure* and the other PPIR corresponding to the equivalent *v-structure* might not be present. Furthermore, every directed edge $A \rightarrow B$ in the *f-pattern* is oriented as $A \rightarrow B$ in the DAG, and every unshielded collider in the DAG is either an unshielded collider or a marked unfaithful triple in the *f-pattern*.

Under the three assumptions of Section 6.2 and by Theorems 2, 3 and 4, it follows that an *f-pattern* represents the equivalence class under triangle faithfulness.

7. Finite sample case

One could limit the analysis about adjacency unfaithfulness by only considering the CIs of a distribution. When relying on data for learning the system's causal structure, this is called the infinite sample case since large samples are needed to correctly identify the CIs of the underlying distribution. On the other hand, when only limited data is available, the independence test will make errors. Let us assume that the test is based on estimating the (conditional) Dependency Strength (DS) between two variables and the test uses a threshold for deciding on (in)dependency. The smaller the sample, the more the estimated DS can deviate from the true value. For smaller sample sizes, a higher threshold is then used so that true independencies are not misclassified as dependencies. But this implies that the weaker a dependency is, the more likely it gets misclassified as an independence. This is especially true as the true DS becomes lower than the threshold. The test will only detect dependencies that are sufficiently strong. Misqualified dependencies are the false positives of the independence test. On the other hand, independencies can be misqualified as dependencies; the false negatives of the independence test. We discuss them in turn.

7.1. False positive independence tests

The following three cases should be considered.

Weak edges. An edge $X - Y$ with a small dependency strength of X and Y , denoted as $DS(X; Y)$, can still have a high $DS(X; Y|Z)$ when conditioned on one of the other parents, as is shown by PIRs. A PIR still contains a lot of information despite the marginal independence. Our extensions overcome missing PIRs or quasi-PIRs (edges that look like PIRs due to the finite sample size). On the other hand, if both $DS(X; Y)$ and $DS(X; Y|Z)$ for all $Z \subset OthPa(X - Y)$ are small, we cannot overcome overlooking such edges. Limited data gives limited precision. We call them *weak edges*.

Near-to-unfaithfulness. In general, dependencies with a low DS lead to near-to-unfaithful situations. The sample size is not large enough to distinguish all dependencies. Consider $X \rightarrow Y \rightarrow Z$. Even when $Y \perp\!\!\!\perp Z|X$ holds in the distribution, by the information X has about Y , the dependency strength between Y and Z becomes smaller. The more X has information about Y , the smaller the dependency. Even when Y has a lot of information about Z . As a result the independence test is not able to detect the dependency between Y and Z when conditioned on X . The (erroneous) independence $Y \perp\!\!\!\perp Z|X$ together with $X \perp\!\!\!\perp Z|Y$ results in the CI pattern of an information equivalence. Cases of near-to-unfaithfulness lead to the same patterns of observed conditional independencies and hence the same ambiguities. Our extensions to the learning algorithm will therefore also be able to handle near-to-adjacency-unfaithfulness.

Weakening by conditioning. A third way in which limited samples disrupt the learning is that an increased cardinality of the conditioning set reduces the robustness of most independence tests [2, p. 116]. This is one of the reasons that a limit is put on the depth of the adjacency search of the PC algorithm.² We call this effect *weakening by conditioning*. It results in a CI which does not correspond to a *d-separation* in the true graph.

We conclude the following:

1. Missing weak edges cannot be overcome.
2. Near-to-unfaithfulness may result in the CI pattern of a PIR. This will be resolved by checking for depsets as we do for PIRs.
3. Both, near-to-unfaithfulness resulting in the CI pattern of an information equivalence and CIs coming from weakening by conditioning result in what we call *non-genuine separation set*, that is, the conditioning set of a CI is not a separation set in the true graph.

To overcome the last problem we will develop a test for detecting false separations. During learning, CIs are used to detect that two variables are *d-separated* in the true graph. A cutset is a set of variables that blocks all active paths between X and Y .

² A depth limit may also be placed on PC simply to allow it to finish in reasonable time.

By the Markov condition, a cutset results in a CI if X and Y are not adjacent. The CI makes the identification of non-adjacency possible. A cutset is minimal if no proper subset is a cutset. We define a *genuine separation set* of X and Y as a set of which all variables block a different path between X and Y . Omitting a variable from the set opens one or more paths. A genuine separation set should not be a cut set, it is a subset of a minimal cut set if X and Y are not adjacent.

The following theorem gives the conditions to recognize non-genuine separation sets.

Theorem 5. *A set $Z \subseteq V \setminus \{X, Y\}$ is a non-genuine separation set for X and Y in G if for one of the elements U of Z one of the following d -separations hold in G : ($Z' = Z \setminus U$ and $T \subseteq V \setminus (Z \cup \{X, Y\})$)*

1. $U \perp Y | Z'$ or $U \perp X | Z'$;
2. $U \perp Y | Z', T$ and $U \perp X | Z', T$;
3. $[U \perp Y | X, Z'']$ or $[U \perp X | Y, Z'']$ for some $Z'' \subset Z'$;
4. $[U \perp Y | X, Z', T]$ or $[U \perp X | Y, Z', T]$.

If none of the d -separations hold, either Z is a genuine separation set, or there is a $U \in Z$ that forms a triangle or a v -structure with X and Y .

The proof is given in [Appendix A](#).

The conditions are in terms of d -separations. Non-genuine separation sets can thus be identified by the CIs that correspond to the d -separations. This will form the basis of our learning algorithm. If a strict conditional independence $[X \perp Y | Z]$ is found, the CIs following from the 4 conditions are tested to make sure a genuine separation set is found. Note that condition (1) is not possible under the graphoid axioms. Given $U \perp Y | Z'$ and $X \perp Y | Z', U$, from the contraction property follows $X \perp Y | Z'$ which contradicts the given strict d -separation. Condition (1) is, however useful in the finite sample case. From condition (2) it follows that T separates X and Y . An example is model $X \rightarrow T \rightarrow Y$ with $T \rightarrow Z$. We should therefore consider T as separation set and not Z . The CIs corresponding to conditions (3) and (4) result in the presence of equivalent edges, as discussed previously.

7.2. False negative independence tests

In the former sections, the infinite sample case, we studied non-Markovian independencies (where faithfulness would expect a dependency). We showed how such independencies lead to ambiguities. On the other hand, we assumed that all Markovian CIs were measured correctly. In the finite sample case we cannot assume the latter. *Accidental dependencies* will occur. They can give rise to the following situations:

1. An accidental marginal dependency $X \not\perp Y$ where there is no active path between X and Y ($X \perp Y$).
2. An accidental conditional dependency $X \not\perp Y | Z$ where X and Y get d -separated by Z ($X \perp Y | Z$).
3. For finding separation sets we will test the conditions of Theorem 5. Those conditions are based on Markovian independencies. However, if these independencies are misqualified, the test might fail and a non-genuine separation set may result in the removal of an edge.
4. An accidental PIR by $[X \not\perp Y | Z]$; Z renders X and Y dependent where there is no active path between X and Y ($X \perp Y$).

For the first two, accidental marginal and conditional dependencies, there is no way to find out that these dependencies are accidental. Moreover, we are very prudent in removing an edge. We require several conditions to hold for a separation set (Theorem 5). Since an accidental dependency does not have a separation set, it is very likely that this edge will remain. In the PC and CPC algorithm, it is more likely that both variables get independent by some other variables. This is confirmed by the experimental results shown in Section 10.2. Therefore our algorithm is conservative: each dependency will be present in the graph. In case 3 there is a combination of (near-to-)adjacency unfaithfulness and an accidental dependency. We do not think there is a straightforward way of detecting this. Finally, to overcome case 4 when testing for PIRs, we add the extra test that $X \not\perp Z | Y$ and $Y \not\perp Z | X$ should hold for identifying Z as a depset of $X - Y$. These dependencies trivially hold for distributions, but not when considering the finite sample case.

8. The adjacency conservative PC algorithm

The results of the previous section resulted in an extension of the CPC algorithm (Section 2.3): the *Adjacency Conservative PC algorithm* (ACPC). The algorithm adds to CPC the rules of S2' to S2 (Algorithm 2) and replaces Part II with Part II' (Algorithm 4). In S2'IIa we choose threshold $s = 2$ to limit the complexity of the learned graphs. The conditions of Theorem 5 are used for the procedure given by Algorithm 3. Besides recording the sepsets for pairs of variables, it will also record depsets. The algorithm returns an f -pattern. When we speak about adjacencies, these special edges are also considered. Non-equivalent and non-PPIR edges are called normal edges. An implementation of the algorithm can be found on <http://parallel.vub.ac.be/acpc>.

Algorithm 2. ACPC algorithm S2'

[I] Before testing whether $X \perp\!\!\!\perp Y | \mathbf{S}$ holds, check the following:

- a** When $X - Y$ is a PPIR, add $depset_{XY}$ to \mathbf{S} .
- b** Skip the test if $X - Y$ has an equivalent edge $X - Z$ or $Y - Z$ and Z is a member of \mathbf{S} .

[II] If the independence test returns $X \perp\!\!\!\perp Y | \mathbf{S}$, do the following before removing the edge:

- a** If \mathbf{S} has less than s elements, look for a T in $Adj(X) \cup Adj(Y)$ for which $X \perp\!\!\!\perp Y | T, \mathbf{S}$ and $X \perp\!\!\!\perp T | Y, \mathbf{S}$ and $T \perp\!\!\!\perp Y | X, \mathbf{S}$. If such a T exists, do not remove the edge, denote it as a PPIR with $depset T$.
- b** If \mathbf{S} is not empty, test whether it is a genuine sepset by calling Algorithm 3. If not, do not remove the edge.
- c** If during the test, a potential information equivalence was found by an independence following from conditions (3) or (4) from Theorem 5, say by $Z \perp\!\!\!\perp X | Y, \mathbf{U}$, denote $X - Y$ as equivalent to $X - Z$ in the graph.
- d** If the separation of X and Y leads to the separation of a depset from its PPIR (X or Y is a depset from a PPIR to which the other variable belongs): search for another depset among the nodes adjacent to one of both variables of the PPIR. If no such depset can be found, delete the PPIR.

Algorithm 3. Genuine separation set test

To test whether set \mathbf{S} genuinely separates X from Y : test for all $Z \in \mathbf{S}$ all conditional independencies following from the d -separations of Theorem 5.

If $X - Y$ is denoted as a PPIR, add the depset to all tests.

For conditions 2 and 4, check only variables T which are adjacent to X , Y or Z and which are genuine separation variables (call this procedure recursively but without checking conditions (2) and (4)).

If one test returns independence, return false.

If condition 3 or 4 returns independence, denote X , Y and Z as potential information equivalences.

If after all tests, no independencies are found, return true.

Theorem 6 (Correctness of ACPC). *Consider a graph G , a JPD P generated by G and $I(P)$, the set of CIs of P : if minimality, causal sufficiency, Assumption 1 and triangle-faithfulness hold for P , the algorithm will, based on $I(P)$, return an f -pattern describing a set of DAGs that includes G . The algorithm is not trivial; it does not always return the set of all DAGs.*

Algorithm 4. ACPC algorithm Part II'

Part II' **Orientation.**

- Perform all of the following steps until no more edges can be oriented:
 - Remove the PPIRs from G ;
 - Perform S3 of Algorithm 1, except that unshielded triples containing an equivalent edge are not considered;
 - Perform S4 from the original algorithm on non-equivalent edges;
 - Add the PPIR edges back into G ;
- S5 Go through all PPIRs. Look for triangles consisting of normal edges and PPIRs in which for each PPIR the opposite variable in the triangle is a depset.
 - a** If the triangle contains two normal edges which form a v-structure, remove the PPIR.
 - b** If the triangle only contains one normal edge which is directed, direct the PPIR that contains the node to which the arrow of the normal edge is pointing, label the PPIR as a PIR and remove the other PPIR from the graph.
 - c** For all oriented edges $D \rightarrow A$ in G for which only A belongs to the triangle, check whether A and D form a faithful triple and a v-structure with one of the two other nodes of the triangle (as in S3 of CPC). When testing the unshielded triple $\langle B, A, D \rangle$, add $depset_{AB}$ to the conditioning set of the independence tests. If the CIS of a v-structure are found, orient the two triangle edges containing A towards A and delete the third triangle edge if it is a PPIR.

The algorithm is not complete as it does not resolve all possible PIRs and information equivalences. First, we do not identify 2-1 CI patterns which are in general identifiable (see discussion in Section 5.1) Second, we do not explicitly test for context-dependent information equivalences. They are also identifiable in general, but the results of the independence test are however not reliable as we are testing with many conditioning variables. Keeping the equivalent edges of context-dependent information equivalences does not create errors, just a larger equivalence class.

9. Related work

The learning algorithms considered here are based on the evidence given by the observed conditional independencies. Score-based algorithms rely on a global score of how well the model fits the data [13–15]. Those algorithms do not explicitly depend on faithfulness. As shown in the experiments section, they give better results in case of violations. The advantage of independence-based algorithms is that they explicitly provide *evidence* for the resulting graph. The choice for connecting or not connecting two variables is backed by a set of dependencies and independencies. Secondly, by our extensions, an equivalence set is returned in the form of a pattern which indicates the ambiguities which cannot be resolved by the given evidence.

In the realm of independence-based learning algorithms, our work is a natural continuation of the work of Spirtes et al. [2] and Ramsey et al. [6]. Since the invention of the PC algorithm, several improvements were made for making the algorithm more reliable. The necessary path condition (NPC) algorithm [16] was introduced as a robust extension for the PC algorithm. It states that for each strict conditional independence $[X \perp\!\!\!\perp Y | Z]$ there must exist a path between X (Y) and each $U \in Z$ not crossing Y (X). This is similar to the notion of genuine separation set for which we defined a set of sufficient conditions (Section 7.1). The idea of NPC directly relates to the notion of information equivalence which is based on the idea of separating X and U by Y . This is possible if there is no path between X and U not going via Y . The NPC introduces the concept of ambiguous edges, which is defined as an edge whose presence depends on the absence of another. In NPC, these ambiguous regions are resolved by including a minimal number of ambiguous edges in order to satisfy a maximal number of independence relations. We showed that ambiguous regions correspond to the equivalent edges. Instead of forcing them into a DAG structure, we model the ambiguity explicitly by an f -pattern. Note that our set of conditions goes beyond checking for information equivalence, conditions (1) and (2) of Theorem 5 allow for identifying independencies following from ‘weakening by conditioning’. However, the conditions of Theorem 5 ensure that there is an active path between X and Y when there is a dependency between both. In that sense does the theorem provide the ‘necessary conditions for a path’.

Other work in limiting the separation sets to be considered rely on similar ideas, see for instance [17, 18]. In this paper, we provide a more complete analysis in terms of adjacency unfaithfulness and identification of all possible ambiguities. Next, both J. Abellan and Moral [17] and Cano et al. [18] take into account the measure of their strength to solve ambiguities. This is surely possible when considering that most equivalence come from near-to-unfaithfulness situation. In cases of real information equivalences, however, the information that equivalent variables contain about the target is the same.

Information equivalences lead to the existence of multiple, equivalent Markov boundaries. This has recently been studied by Statnikov and Aliferis [19].

10. Experimental results

To illustrate the adequacy of our extensions, simulations were performed on linear Gaussian and binary models. Experiments were performed on 100 randomly selected DAGs with d nodes and $d * 3/2$ edges where d will be selected between 10 and 50. For each such graph, a random structural equation model was constructed by selecting random edge coefficients uniformly from $[0.1, 1] \cup [-1, -0.1]$ and the variance of the disturbance terms was chosen randomly from $[0.01, 1]$. A random data set is simulated for each of the models to base the learning on. A significance level $\alpha = 0.05$ is used for each independence test which is based on Fisher’s Z transformation of partial correlation. Similar experiments were performed with Bayesian networks defined over a set of binary variables and randomly chosen conditional probabilities. The Chi-Square test was used as independence test with a significance level of 0.05. We also did tests with the ALARM benchmark model [20].

10.1. Adjacency unfaithfulness

First, we test the occurrence of unfaithfulness and classify the cases in which they appear. It should be noted that these are cases of near-to-unfaithfulness. Since the parameters of the distribution are chosen at random, the probability of exact unfaithfulness can be considered zero (see Introduction). For each edge $X - Y$, strict independence of X and Y was tested by conditioning on all possible subsets of the other variables. We limited the size of the conditioning set to 2.

Table 1 shows the percentage of edges for which adjacency faithfulness holds. First for a fixed number of 20 nodes (and 30 edges) and an increasing sample size. Then for sample size of 500 and number of nodes from 10 to 50. All percentages come from an averaging over 10 experiments.

As expected, a larger sample size leads to less unfaithfulness. However, even for the highest sample size, about 10% of the edges are unfaithful. The ALARM model is less susceptible to unfaithfulness than random models. Finally, more nodes – a measure for the complexity of the model – do not lead to less adjacency faithfulness.

Next, each violation (strict independence by a subset of other nodes) was classified as either resulting from a weak edge, a violation of conditions (1) or (2) from Theorem 5 on genuine separation, a PIR, an information equivalence or triangle unfaithfulness. Note that for one edge several violations could be found. The classification was performed based on testing for patterns of conditional independencies based on the data, not by checking the pattern in the model itself. As independence tests on finite samples can fail, some cases were not classified and would result in a false removal of the edge since not identified by the algorithm. The number of such cases was very low. Mostly just 1 or 2 cases appeared in each set of 10

Table 1

Percentage of edges for which adjacency faithfulness holds, for varying sample size (with 20 nodes and 30 edges) and for varying number of nodes (with sample size of 500).

Sample size	100	200	500	1000	2000	10000
Random model	16	26	43	64	69	85
ALARM	23	41	70	79	90	93

Number of nodes	10	20	30	40	50
Random model	42	39	41	44	40

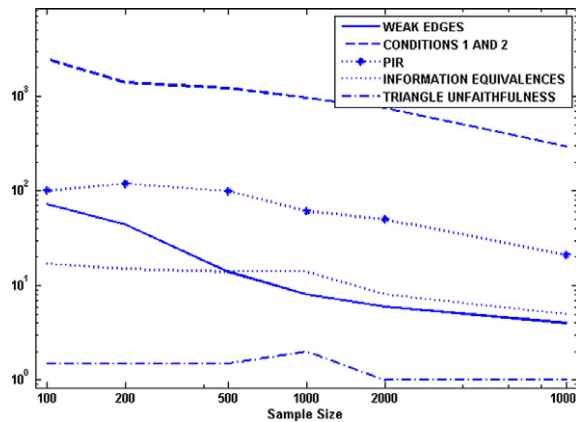


Fig. 5. Classification of adjacency unfaithfulness for random models with 20 nodes and 30 edges and increasing sample size. Note that both axes have a logarithmic scale.

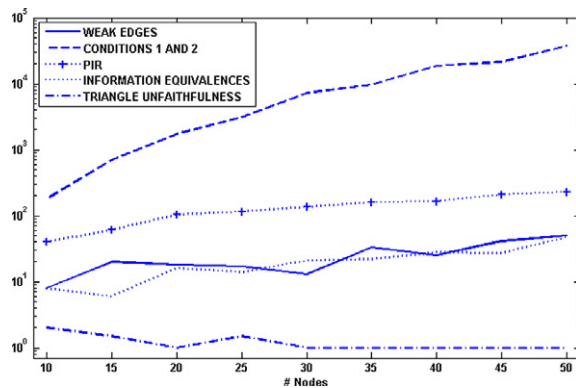


Fig. 6. Classification of adjacency unfaithfulness for random models with a sample size of 500 and nodes from 10 to 50. Note that the Y-axis has a logarithmic scale.

experiments, with a maximum of 14 cases for the experiments with a sample size of 100. The results for random models and the ALARM model are shown in Figs. 5, 6 and 7. Note that the figures show the counts of unfaithful cases, whereas the table shows the percentages of the opposite, namely faithfulness. The numbers denote the sum of cases identified over 10 experiments. As expected, most types appear less with larger samples. But they appear more for larger models, except for triangles. But this is due to a lower chance to have a triangle in the larger networks.

10.2. Learning accuracy

To compare the learning capabilities, the PC, CPC and ACPC algorithms were applied with depth 2 on the data and for ACPC only single variables were considered for the T -sets in Algorithm 3. We also compared the results with the learning accuracy of the score-based GES algorithm [21]. We took the TETRAD implementation of the algorithms (See TETRAD project: www.phil.cmu.edu/projects/tetrad/). At <http://parallel.vub.ac.be/acpc> you can find an implementation of ACPC which is compatible with TETRAD. The website also provides illustrative examples and some of the datasets of the experiments.

The output graph of each algorithm was compared to the Markov equivalence class (MEC) of the true DAG. Table 2 shows the outcomes averaged over all experiments and relative to the number of nodes (percentages). Correct edges are the edges of the MEC of the true graph that appear as normal edges in the f-pattern. PPIRs and equivalent edges in the f-pattern

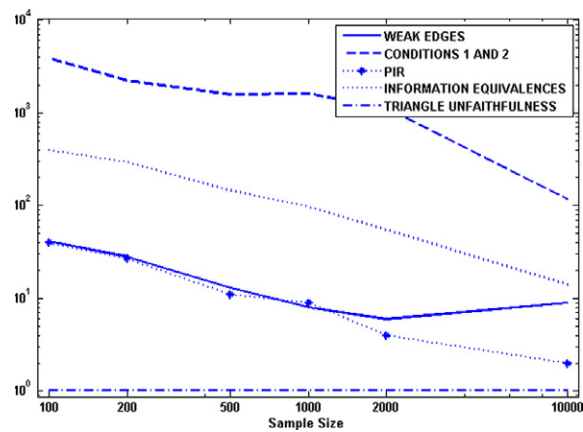


Fig. 7. Classification of adjacency unfaithfulness for the ALARM model and increasing sample size. Note that both axes have a logarithmic scale.

Table 2

Accuracy results of the 4 algorithms, in percentages.

	PC	CPC	GES	ACPC
<i>Edges</i>				
Correct	76.7	76.2	87.0	77.9
Ambiguous	–	–	–	74.4
False negatives	23.2	23.8	12.9	8.1
Weak	3.7	4.5	2.2	3.6
False positives	4.1	4.3	13.2	11.8
Accidental correlations	2.9	3.1	6.2	9.3
<i>Orientations</i>				
Correct	25.3	29.8	54.6	36.6
Ambiguous	3.6	16.5	12.7	47.6
Wrong	19.9	2.1	9.2	3.7
False positives	15.1	2.9	10.6	3.9

are counted as ambiguous edges. False negative edges are edges in the MEC that do not appear in the f-pattern, not as a normal edge and not as an ambiguous edge. Weak edges are false negatives whose nodes are marginally independent and independent conditional on the other parents. False positive edges appear as normal edges in the f-pattern, but not in the MEC. If the nodes of a false positive are not d -connected in the MEC, they are classified as accidental correlations.

We first show the results of experiments performed on 100 randomly selected DAGs with d nodes and $3/2 * d$ edges, where d is randomly chosen between 5 and 25. The data size is 1000.

The learning performance of the orientation is evaluated by looking at edges appearing in both the MEC and the f-pattern. Edges having the same orientations in both are counted as correct orientations, when not oriented in both or only in f-pattern as ambiguous. Wrong orientations appear as oriented in both, but in the opposite direction. False positives are arrowheads appearing in the f-pattern but not in the MEC.

The results show that the difference between the PC and CPC lies clearly in the reduction of the false positive arrowheads, whereas the performance on adjacencies is similar (the adjacency phases are the same). GES clearly outperforms PC and CPC regarding adjacencies. The reason is that GES is less susceptible for unfaithfulness. The ACPC algorithm clearly reduces the number of false negative edges, when compared to the three algorithms. If we consider that weak edges cannot be identified, the performance gain is even more drastic. By subtracting the number weak edges from the false negatives, the number of (false negatives minus weak edges) drops from 19.5%/19.3% for PC/CPC to 10.7% for GES and to 4.5% for ACPC.

The accuracy increase of ACPC regarding false negatives is at the expense of ambiguous edges and more false positives. The latter can be explained by *accidental correlations*. Accidental correlations lead to false negative independence tests - the oracle qualifies a Markovian CI as dependent due to accidentally-correlated data. ACPC is conservative about dependencies; it will not remove edges if there is no alternative path to explain a dependency. Take nodes that are not d -connected in the true graph, but are accidentally correlated. If this accidental correlation is above the threshold, the oracle will qualify it as a dependency. In the following steps, when conditioning happens on other variables, the weakening-by-conditioning effect will bring the measured dependency strength below the threshold and remove the 'accidental' edge. This happens with PC and CPC as can be seen in the table.

Finally, it is worth noting that the standard deviation for the false positive and negative edges is almost as high as the average, which points to a high performance fluctuation from one experiment to another.

Figs. 8–10 show the influence of sample size and number of nodes. We only consider the correctness of the presence of edges, not their orientations. The numbers are percentages relative to the true number of edges. As expected, the true positives increase with increasing sample size and the false negatives decrease. On the other hand, the number of ambiguous

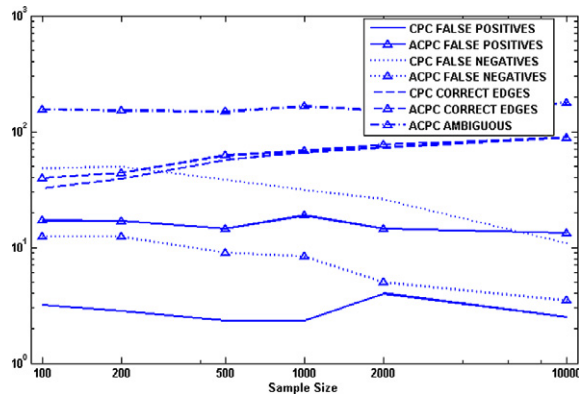


Fig. 8. Learning accuracy for random models with 20 nodes and 30 edges and increasing sample size. Note that both axes have a logarithmic scale.

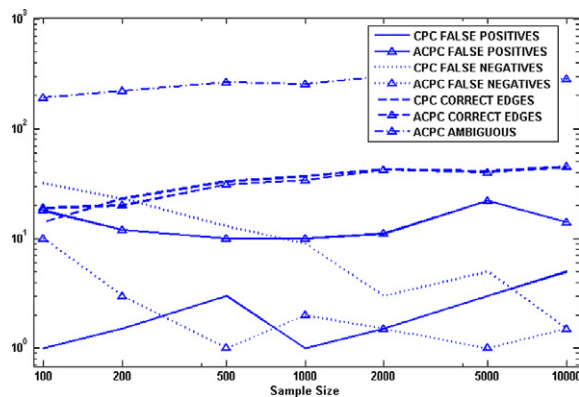


Fig. 9. Learning accuracy for the ALARM model and increasing sample size. Note that both axes have a logarithmic scale.

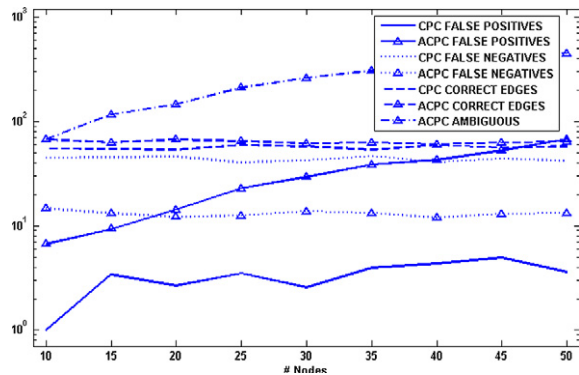


Fig. 10. Learning accuracy for random models with a sample size of 500 and nodes from 10 to 50. Note that the Y-axis has a logarithmic scale.

edges and false positives (due to accidental dependencies) remain relatively constant. An increasing number of nodes (and edges) greatly affects the number of ambiguous edges and false positives of the ACPC algorithm. The other numbers stay relatively stable.

11. Conclusions

We cannot rely on adjacency faithfulness when constructing reliable independence-based learning algorithms. We showed that under triangle faithfulness, violations mainly lead to two types of ambiguities: equivalent v-structures (PPIRs) and equivalent edges. The first coming from pseudo-independent relations, the second from information equivalences.

Based on specific CI patterns, a set of DAGs, modeled by an f -pattern, can be identified that are indistinguishable from the perspective of the CIs. Just like the Conservative PC algorithm detects and treats failures of orientation-faithfulness, our Adjacency Conservative PC algorithm detects violations of adjacency-faithfulness. Our algorithm is conservative in the sense that it ensures that every (conditional) dependence that is detected between two variables has a d -connection in all DAGs of the equivalence class. ACPC is therefore highly susceptible for accidental dependencies. In the finite sample case, weak conditional dependencies can be wrongly classified as CIs by the oracle. This leads to near-to-unfaithful cases, weakening by conditioning and weak edges. The two first are treated, missing weak edges should be accepted. Since weak edges and triangle unfaithfulness cannot be detected, we believe that this analysis shows the natural bounds of what can reliably be learned under causal sufficiency from conditional independence information alone.

Acknowledgements. We want to thank Alexander Statnikov, Angelos Armen and Sofia Triantafillou for their valuable comments. We would like to thank the blind reviewers in helping to structure our exposition and clarify our ideas. This research was partly funded by the Prognostics for Optimal Maintenance (POM) project (grant nr. 100031; www.pom-sbo.org) which is financially supported by the Institute for the Promotion of Innovation through Science and Technology in Flanders (IWT-Vlaanderen).

Appendix A. Proofs

Here we give the proofs of the important theorems. We also repeat the theorems themselves.

Theorem 2. *Under triangle faithfulness and minimality, violations of adjacency faithfulness by strict independencies are limited to pseudo-independent relations, information equivalences and 2-1 CI patterns.*

Proof. Consider X and Y adjacent in G , the true causal graph. If X and Y are marginally independent, $X - Y$ is a PIR; there exists a depset (by minimality). If $[X \perp\!\!\!\perp Y | Z]$, then we have the following possibilities for all $Z \in \mathbf{Z}$:

- Z must be dependent on both X and Y (to entail the independence). If Z has an active path to one, let's say X , then the path to Y goes via X . Then X d -separates Z and Y . We have $[X \perp\!\!\!\perp Y | Z]$ and $Y \perp\!\!\!\perp Z | X$.
 - If conditioning set \mathbf{Z} contains only Z , we have an unconditional information equivalence.
 - If $Y \perp\!\!\!\perp Z | X$, $Z \setminus Z$, then we have a conditional information equivalence.
 - Otherwise, if $Y \not\perp\!\!\!\perp Z | X$, $Z \setminus Z$ we have a 2-1 CI pattern.
- If Z is adjacent to both, because of triangle faithfulness, the case in which Z is a collider on the path $\langle X, Z, Y \rangle$ (case TRUFF3) is excluded. Z is not a collider on the path $\langle X, Z, Y \rangle$. Then we have a v -structure in X or Y . Say it is in Y . It means that Z is a parent of Y . By MIN it follows that there is a depset. $X - Y$ is a PIR.
- As a last case, Z has at least one separate active path to both X and Y , and since it is not a triangle at least one of both paths contains another variable. Say U on the paths connecting Y and Z . By edge $X - Y$, Z has two paths to both X and Y . It means that Z can be d -separated from Y by $[Z \perp\!\!\!\perp Y | X, U]$, unless we have v -structure $Y \rightarrow X \leftarrow Z$. Then Z is another parent of $X - Y$. By minimality there must be another parent that renders X and Y dependent. The edge is a PIR. If the strict d -separation holds, we have a conditional information equivalence if also $X \perp\!\!\!\perp Y | U, Z$. Otherwise we end up with a 2-1 CI pattern. \square

Theorem 3. *If $X - Y$ is a PIR with depset containing one variable Z , it generates equivalent v -structures $X \rightarrow Y \leftarrow Z$ and $X \rightarrow Z \leftarrow Y$.*

Proof. We denote the first edge set as $struct_1$ and the second as $struct_2$. We have to prove (I) that when $G \in \mathcal{G}$ is not Markovian and $G \cup struct_1$ is Markovian that also $G \cup struct_2$ is Markovian (see note under definition) if G has no incoming edge into Y or Z and (II) that at least one such G exists.

- (I) Take $A \not\perp\!\!\!\perp B | \mathbf{S}$ not represented in G , but represented in $G \cup struct_1$. First consider that the active path goes through $Z \rightarrow Y$. Since there might be no identifiable incoming arrow into Y or Z , the active path has subpath $U \leftarrow Z \rightarrow Y \leftarrow V$. Then, $G \cup struct_2$ also generates an active path which is based on subpath $U \leftarrow Z \leftarrow Y \leftarrow V$. When the active path contains $X \rightarrow Y$, then $X \not\perp\!\!\!\perp Y | \mathbf{S}$ must hold. Since otherwise $X \perp\!\!\!\perp Y | \mathbf{S}$ and $A \perp\!\!\!\perp Y | X, \mathbf{S}$ (consider that the path from A to B arrives at X first, the independence expresses that the edge is needed) would give $A \perp\!\!\!\perp Y | \mathbf{S}$ by contraction. By the dependency it follows, by Theorem 1, that a depset (or a set of variables d -connected to it) must be presented in \mathbf{S} . These variables open the path of A to B in $G \cup struct_2$ since unblocking v -structure $X \rightarrow Z \leftarrow Y$.
- (II) We have dependency $X \not\perp\!\!\!\perp Y | Z$. So for a Markovian graph H , there must exist an active path between X and Y when conditioned on Z . We construct a graph H which contains $X - Y$ but no other active path between X and Y when conditioned on Z . We ensure that all dependencies of the distribution are represented in the graph. This is possible under the given restriction of no incoming edge since we can always ensure that there is a path from A to X and from Y to B . Hence H is Markovian. Then, by construction, $G = H \setminus X - Y$ is not Markovian (not representing $X \not\perp\!\!\!\perp Y | Z'$) and $G \cup X - Y$ is. \square

Theorem 4. *If X and U are information equivalent with respect of Y when conditioned on S and this information equivalence is context-independent, then edges $X - Y$ and $U - Y$ are equivalent edge sets.*

Proof. We have to prove (I) that when G is not Markovian and $G \cup X - Y$ is Markovian that also $G \cup U - Y$ is Markovian (see note under definition) and (II) that at least one such G exists. (I) Take $A \perp\!\!\!\perp B | S$ not represented in G , but represented in $G \cup X - Y$. For this, (a) an active path must exist from A to B via $X - Y$ or (b) $X - Y$ opens a collider. First assume (a) and assume that the path starting in A first reaches X and then Y before leading to B . We have to prove that under the given conditions there is also an active path in $G \cup U - Y$ if it does not create new v-structures. We have that $A \perp\!\!\!\perp Y | S$, otherwise the path via $X - Y$ would not have been necessary for a d -connection. We prove that necessarily $A \perp\!\!\!\perp U | S$ (see below) and since the graph is Markovian there is an active path from A to U . Since $U - Y$ is not creating new v-structures, we have an active path from A to Y . The concatenation of the path from A to Y and from Y to B is thus an active path (if Y is a collider on the path via $X \rightarrow Y$ it is also a collider on the path via $U \rightarrow Y$).

Proof of $A \perp\!\!\!\perp U | S$. We show that if A would be independent from U , the independence $A \perp\!\!\!\perp Y | S$ would follow. By the information equivalence we have $\kappa_Y(U) = \kappa_Y(X)$. From $\kappa_Y(U) = f(U)$ follows that $A \perp\!\!\!\perp \kappa_Y(U) | U, S$. The three following deductions are based on the contraction property:

$$X \perp\!\!\!\perp Y | \kappa_Y(X), S \ \& \ A \perp\!\!\!\perp Y | X, \kappa_Y(X), S \Rightarrow A \perp\!\!\!\perp Y | \kappa_Y(X), S \tag{8}$$

$$A \perp\!\!\!\perp U | S \ \& \ A \perp\!\!\!\perp \kappa_Y(U) | U, S \Rightarrow A \perp\!\!\!\perp \kappa_Y(U) | S \tag{9}$$

$$A \perp\!\!\!\perp \kappa_Y(U) | S \ \& \ A \perp\!\!\!\perp \kappa_Y(X) | k, S \Rightarrow A \perp\!\!\!\perp Y | S \tag{10}$$

(b) Now we consider the case in which X is a collider (or a descendant of a collider) on the path between A and B (e.g., $A \rightarrow X \leftarrow B$ and $X \rightarrow Y$). Similar to the proof above, we have that $A \perp\!\!\!\perp U | S$ and $B \perp\!\!\!\perp U | S$, to there must active paths between A and U , and between B and U . But both paths cannot create an active path from A to B through U , since G is not Markovian. This means that either Z is a collider or is a descendant of another collider. This ensures that edge $U - Y$ opens a collider.

(II) There is at least one dependency, namely $X \perp\!\!\!\perp Y | Z'$. So for a Markovian graph H , there must exist an active path between X and Y when conditioned on Z' . We construct a graph H which contains $X - Y$ but no other active path between X and Y when conditioned on Z' . We ensure that all dependencies of the distribution are represented in the graph. This is possible under the given restriction since we can always ensure that there is a path from A to X and from Y to B . Hence H is Markovian. Then, by construction, $G = H \setminus X - Y$ is not Markovian (not representing $X \perp\!\!\!\perp Y | Z'$) and $G \cup X - Y$ is. \square

Theorem 5. *A set $Z \subseteq V \setminus \{X, Y\}$ is a non-genuine separation set for X and Y in G if for one of the elements U of Z one of the following d -separations hold in G : ($Z' = Z \setminus U$ and $T \subseteq V \setminus (Z \cup \{X, Y\})$)*

1. $U \perp\!\!\!\perp Y | Z'$ or $U \perp\!\!\!\perp X | Z'$;
2. $U \perp\!\!\!\perp Y | Z', T$ and $U \perp\!\!\!\perp X | Z', T$;
3. $[U \perp\!\!\!\perp Y | X, Z'']$ or $[U \perp\!\!\!\perp X | Y, Z'']$ for some $Z'' \subset Z'$;
4. $[U \perp\!\!\!\perp Y | X, Z', T]$ or $[U \perp\!\!\!\perp X | Y, Z', T]$.

If none of the d -separations hold, either Z is a genuine separation set, or there is a $U \in Z$ that forms a triangle or a v-structure with X and Y .

Proof. To be a genuine separation set, all elements of Z must lie on a separate path between X and Y , and all paths between X and Y must be blocked by Z . With path we mean an active path in terms of a d -connection. The conditions happen when Z is not a minimal cut set. Condition (1) or (2) holds when U is not connected to X or Y with a separate path. Condition (3) or (4) happens when U is d -connected to Y via X (by the strictness).

The last part is about what happens when none of the conditions are met. Conditions (1) or (2) guarantee that all elements of Z are connected with X and Y via separate paths. Next for a cut set, all paths between X and Y must be cut. If there would be an uncut path via another node, this node should be added to Z to form a cut set. The remaining case is when X and Y are adjacent. Then, unless U forms a triangle with X and Y , there exists a subset T which separates U from Y (or X). U can then be d -separated from Y given X, T and Z' (condition (4)) unless they form a v-structure ($U \rightarrow X \leftarrow Y$). \square

Theorem 6 (Correctness of ACPC). *Consider a graph G , a JPD P generated by G and $I(P)$, the set of CIs of P : if minimality, causal sufficiency, Assumption 1 and triangle-faithfulness hold for P , the algorithm will, based on $I(P)$, return an f -pattern describing a set of DAGs that includes G . The algorithm is not trivial; it does not always return the set of all DAGs.*

Proof. (1) Assume adjacency faithfulness:

Given adjacency faithfulness, the only difference with CPC during the adjacency search is that in step S2'[II]a for each true v-structure $X \rightarrow Z \leftarrow Y$, such that $X \perp\!\!\!\perp Y$, a PPIR $X - (Z) - Y$ is added. During the first part of the orientation phase these PPIRs are temporarily removed to discover the v-structures. As a result in step S5a, the PPIR is removed

and the correct structure is found. The correctness as well as the non-triviality follows then from the correctness and non-triviality of CPC, proven by [6].

(2) No adjacency faithfulness:

We have to prove that no edge is deleted based on non-Markovian CIs, and that no mistakes are made during orientation.

(2.1) No missing edges

(A) Assume $X - Y$ in correct graph and $X \perp\!\!\!\perp Y$:

In step S2'[II]a, the algorithm looks for a variable T such that $X \perp\!\!\!\perp Y | T$. The existence of T follows from Minimality. Therefore the edge $X - Y$ will be replaced by a PPIR. Now, we show that this PPIR is not removed from the graph, which can only happen when there is a v-structure. If a PPIR would be removed based on the existence of a v-structure $X \rightarrow Z \leftarrow Y$ for some Z , then this indicates that the triangle faithfulness assumption is not satisfied. The removal of a PPIR in this case is an immediate consequence of the triangle faithfulness assumption which dictates that the direction of one of the arcs in the triangle imposes a v-structure. We do not orient v-structures containing equivalent edges, since a v-structure based on an equivalent edge which is not in the true graph could lead to erroneous deletion of a PIR when the equivalent edge appears in a triangle with the PIR.

(B) Assume $X - Y$ in correct graph and $X \perp\!\!\!\perp Y | \mathcal{S}$, with $\mathcal{S} \neq \emptyset$:

theorem 2 showed that this only happens for PIRs, information equivalences and 2-1 CI patterns. All three of them are identified in the algorithm and $X - Y$ is not removed.

(C) Non-triviality is a direct consequence of the deletion of an edge $X - Y$ not present in G if for $\forall \mathcal{S} \subseteq \{X, Y\}$ the independency $X \perp\!\!\!\perp Y | \mathcal{S}$ holds and \mathcal{S} with is a genuine sepset.

(2.2) Correct conservative orientation:

(a) Orientation in the first step of the ACPC orientation phase (II') is only based on non-equivalent edges and non-PPIRs. So the correctness of CPC proves the correctness of these orientation steps in our algorithm.

(b) We do not orient any edges based on equivalent edges. c) Both S5b and S5c trigger when there is a known orientation of a normal edge inside a triangle with (a) PPIR(s) (S5b) or when the orientation of an edge in such a triangle can be inferred (S5c). A direct consequence of triangle faithfulness is that there is a v-structure at the node of the triple which has an incoming arrow. So the correctness of orientation follows from triangle faithfulness. \square

Appendix B. Information equivalence

Information equivalences happen by a larger class than just deterministic relations. They appear by parameterizations that lead to so-called *equivalent partitions*. We will give the definition and main theorem here.

If two variables X and Z are dependent, implied by $P(Z|X) \neq P(Z)$, the conditional distribution of one variable differs for at least two values of the conditioning variable:

$$X \perp\!\!\!\perp Y \Leftrightarrow \exists x_1, x_2 \in X_{dom} : P(Z|x_1) \neq P(Z | x_2). \tag{11}$$

The information a variable contains about the other lies in the differences in the conditional distributions. Values for which this distribution is the same contain the same information. To study this, we partition the domain of the variable with respect to the conditional distribution.

Definition 7 (Z-partition). The domain of X , denoted by X_{dom} , can be partitioned into disjoint subsets X_{dom}^k for which $P(Z | x)$ is the same for all $x \in X_{dom}^k$. We call this the $Z - partition$ of X_{dom} . We denote $\kappa_Z(X)$ as the index of the subset.

Consequently, the conditional distribution depends solely on the index of the Z -partition:

$$P(Z|X) = P(Z|\kappa_Z(X)) \tag{12}$$

$$X \perp\!\!\!\perp Z | \kappa_Z(X) \tag{13}$$

Fig. 11 shows the Z -partition of Z_{dom} and the related conditional distributions of Z . $P(Z)$ is also shown, it is the weighted sum of the conditional distributions: $P(Z) = \sum_x P(Z | x) \cdot P(x)$.

Now we define an equivalent partition.

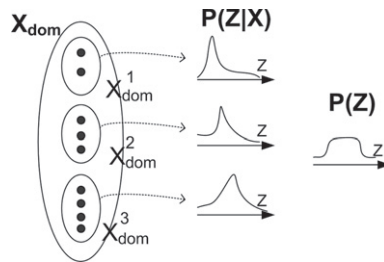


Fig. 11. Z-partition of the domain of X into subsets exhibiting the same conditional distribution.

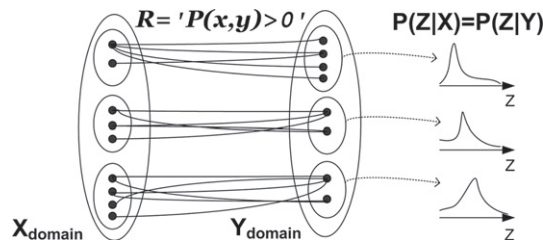


Fig. 12. Variables X and Y are information equivalent for Z . $P(x, y)$ is only strictly positive for values that affect $P(Z)$ similarly. These values are related by relation R .

Definition 8. A relation $\mathfrak{R} \subset X \times Y$ (where the \times operator denotes the Cartesian product) defines an **equivalent partition** in Y_{dom} to a partition of X_{dom} if:

1. $\forall x_1$ and $x_2 \in X_{dom}$ that do not belong to the same partition: $\forall y_1 \in Y_{dom}$ with $x_1 \mathfrak{R} y_1$, it must be that $\neg(x_2 \mathfrak{R} y_1)$.
2. For all subsets X_{dom}^k of the partition: $\exists x_1 \in X_{dom}^k, \exists y_1 \in Y_{dom} : x_1 \mathfrak{R} y_1$.

For an equivalent partition, every partition X_{dom}^k corresponds to a partition Y_{dom}^l . If an element of Y_{dom} is related to an element of a partition of X_{dom} , it is not related to an element of another partition, and each partition of X_{dom} has at least one element that is related to a partition of Y_{dom} . Fig. 12 shows an example of an equivalent partition. No y is related to X -values belonging to different partitions and for every partition, there is at least one related Y -value. Note that a function, for which every x -value has a related Y -value, defines an equivalent partition on Y_{dom} for every partition of X .

The following theorem links violations of the intersection condition, or information equivalences, to equivalent partitions.

Theorem 7. For $X, Y, Z \in \mathbf{V}$, if $X \perp\!\!\!\perp Z$ and $Y \perp\!\!\!\perp Z | X$, then $X \perp\!\!\!\perp Z | Y \Leftrightarrow$ the relation $x \mathfrak{R} y$ defined by $P(x, y) > 0$, with $x \in X_{dom}$ and $y \in Y_{dom}$, defines an equivalent partition in Y_{dom} to the Z-partition of X_{dom} .

The proof is given in [12, p. 100]. Note that Eq. (12) and (13) also hold with respect to Y if we number the corresponding subsets by the same indices.

References

- [1] C. Meek, Causal inference and causal explanation with background knowledge, in: Proceedings of UAI-1995, pp. 403–441.
- [2] P. Spirtes, C. Glymour, R. Scheines, Causation, Prediction, and Search, 2nd ed., Springer-Verlag, 1993.
- [3] C. Meek, Strong completeness and faithfulness in Bayesian networks, in: Proceedings of UAI-1995, pp. 411–418.
- [4] J. Lemeire, D. Janzing, Replacing causal faithfulness with algorithmic independence of conditionals, Minds and Machines (2012).
- [5] J. Zhang, P. Spirtes, Detection of unfaithfulness and robust causal inference, in: Proceedings of the LSE-Pitt Conference: Confirmation, Induction and Science, London, 2007.
- [6] J. Ramsey, J. Zhang, P. Spirtes, Adjacency-faithfulness and conservative CI inference, in: Proceedings of UAI-2006, pp. 401–408.
- [7] J. Pearl, Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference, Morgan Kaufman Publishers, San Mateo, CA, 1988.
- [8] J. Pearl, Causality, Models, Reasoning, and Inference, Cambridge University Press, 2000.
- [9] L.M. de Campos, J.G. Castellano, Bayesian network learning algorithms using structural restrictions, International Journal of Approximate Reasoning 45 (2007) 233–254.
- [10] Y. Xiang, S.K. Wong, N. Cercone, Critical remarks on single link search in learning belief networks, in: Proceedings of UAI 1996, pp. 564–571.
- [11] D. Geiger, T. Verma, J. Pearl, Identifying independence in Bayesian networks, Networks 20 (1990) 507–534.
- [12] J. Lemeire, Learning Causal Models of Multivariate Systems and the Value of it for the Performance Modeling of Computer Programs, Ph.D. thesis, Vrije Universiteit Brussel, 2007.
- [13] D. Heckerman, D. Geiger, D. Chickering, Learning Bayesian Networks: the Combination of Knowledge and Statistical Data, Technical Report MSR-TR-94-09, Microsoft Research, 1994.
- [14] E. Herskovits, Computer-Based Probabilistic Network Construction, Ph.D. thesis, Medical information sciences, Stanford University, CA, 1991.
- [15] J. Suzuki, Learning Bayesian belief networks based on the MDL principle: An efficient algorithm using the branch and bound technique, in: Proceedings of the International Conference on Machine Learning, Bally, Italy, 1996.
- [16] H. Steck, V. Tresp, Bayesian belief networks for data mining, in: Proceedings of the 2nd Workshop on Data Mining und Data Warehousing, 1999.
- [17] M.G.-O.J. Abellan, S. Moral, Some variations on the PC algorithm, in: Proceedings of the European Workshop on Probabilistic Graphical Models (PGM), 2006.

- [18] A. Cano, M. Gomez-Olmedo, S. Moral, A score based ranking of the edges for the PC algorithm, in: Proceedings of the European Workshop on Probabilistic Graphical Models (PGM), 2008, pp. 41–48.
- [19] A. Statnikov, C.F. Aliferis, Analysis and computational dissection of molecular signature multiplicity, PLoS Computational Biology (2010).
- [20] R.C.I.A. Beinlich, H.J. Suermondt, G. Cooper, The Alarm monitoring system: A case study with two probabilistic inference techniques for belief networks, in: European Conference on Artificial Intelligence in Medicine, 1989.
- [21] D.M. Chickering, Optimal structure identification with greedy search, Journal of Machine Learning Research (2002).