Independence-based Causal Structure Learning in Absence of Faithfulness.

Jan Lemeire, Meganck Stijn

Independence-based Causal Structure Learning in Absence of Faithfulness.

Jan Lemeire — Stijn Meganck

ETRO Department, Vrije Universiteit Brussel
Pleinlaan 2, B-1050 Brussel, Belgium
jan.lemeire@vub.ac.be
stijn.meganck@vub.ac.be

RÉSUMÉ. *On présente des extensions pour l'algorithme PC Conservative qui sont capable de détecter des violations de la Fidélité Adjacente en assumant Suffisance Causale et la Fidélité Triangulaire. Les violations sont caractérisées par des relations pseudo-indépendantes et des liens équivalents, générant des indépendances conditionnelles qui ne peuvent pas être modélisés fidèlement. Les deux cas résultent dans de l'incertitude sur des parties du réseau bayésien. Ceci est modèlisé par un 'f-pattern'. Notre algorithme PC Très Conservative est capable d'apprendre correctement le 'f-pattern' d'un système. Des expériences basées sur des simulations montrent que le taux de faux suppressions de liens est réduit de façon significative, mais créant en même temps une augmentation de l'incertitude sur le squelette et de la sensibilité pour des corrélations accidentelles.*

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. This is modeled by an f-pattern. We proved that our Very Conservative PC algorithm is able to correctly learn the f-pattern. Experiments based on simulations 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.*

MOTS-CLÉS : *Apprentissage, structure de réseaux bayésiens,réseaux bayésiens,fidélité*

KEYWORDS: *causal structure learning, faithfulness, causal models*

1. Introduction

Independence-based learning algorithms mainly rely on a reliable independence test (called the oracle) and causal faithfulness, which states that all Conditional Independencies (CIs) are Markovian, i.e. following from the system's causal structure (Spirtes *et al.*, 1993). The validity of causal faithfulness is supported by the Lebesgue measure zero argument (Meek, 1995b), which says that the chance of randomly picking a parameterization resulting in non-Markovian CIs has measure zero. But, in practice, the oracle must test for CI on finite samples. The oracle cannot be expected to correctly detect weak dependencies. This happens in near-to-unfaithful situations - probability distributions can come infinitely close to unfaithful distributions. The Lebesgue measure zero argument does not hold here, since the ϵ-regions around unfaithful situations, in which $DS(X;Y|\mathbf{Z}) < \epsilon$ for $X \not\perp\!\!\!\perp Y|\mathbf{Z}$, do not have Lebesgue measure zero. $DS(X;Y|\mathbf{Z})$ denotes the *dependency strength* of X and Y when conditioned on $\mathbf{Z}$. For linearly-related continuous variables it is defined as the partial correlation. We therefore cannot rely on faithfulness.

We will show that near-to-unfaithful cases can be handled in a similar way as unfaithful cases. Except in cases of triangle unfaithfulness, violations of faithfulness are *detectable* (Zhang *et al.*, 2007), in the sense that the true probability distribution is not faithful to any DAG. The Conservative PC algorithm relies on a weaker form of faithfulness, namely *adjacency faithfulness*, and is able to detect violations of *orientation faithfulness* (Ramsey *et al.*, 2006). When a violation is detected, no orientation is made, but the unfaithful part is marked as a part in which no decision is taken. In this paper we do the same for violations of adjacency faithfulness, which can be identified, under triangle faithfulness, by two patterns : *potential 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. Therefore we called our algorithm the Very Conservative PC algorithm (VCPC).

The next section recalls the essentials about the learning algorithms and the assumptions. Section 3 determines the patterns of adjacency faithfulness violations and, in Section 4, our extensions to the learning algorithm are proposed. Finally, the improvement of the learning efficacy is demonstrated experimentally in Section 5.

2. Independence-Based Causal Structure Learning

The *Markov condition* gives the CIs that follow from a causal structure : every variable is independent of its non-effects conditional on its direct causes. The causal structure can be represented by a Directed Acyclic Graph (DAG) in which the parents of a variable are its direct causes. d-separation, written as $X \perp Y|\mathbf{Z}$, gives all the CIs following from the Markov condition. *Faithfulness* says that no other CIs appear in the system's probability distribution than those entailed by the Markov condition. It means that all CIs we observe are coming from the causal structure and can be used to infer the causal structure. The PC algorithm, Alg. 1, is an algorithm that can partially

learn the causal structure from observations of a faithful distributions (Spirtes *et al.*, 1993). It consists of 2 steps : the skeleton learning and the orientation of unshielded triples. The first is based on the principle that two adjacent variables cannot become independent when conditioned on some other (possible empty) set of variables. This property is expressed by *Adjacency-Faithfulness* (Ramsey *et al.*, 2006). The second step is based on the recognition of v-structures. The correctness of this step relies on *Orientation-Faithfulness* (Ramsey *et al.*, 2006). The output of the PC algorithm is the *Markov Equivalence Class* : the set of indistinguishable DAGs, all having the same Markovian CIs. The class is represented by a Partially-Directed Acyclic Graph (PDAG) in which some edges are not oriented since their orientation cannot be derived from the CIs ; either orientation implies the same CIs.

All directed edges in a *pattern* corresponding to a Markov Equivalence Class represent direct causal relationships. The main advantage of the causal interpretation of directed edges is that it allows to perform *causal inference*, which is the calculation of the effect of an intervention on some variable. In Pearl (2000) a calculus was derived to answer these causal queries using the JPD and structural properties of the DAG representing the causal relations.

Violations of adjacency faithfulness give rise to other, non-Markovian CIs. The question is whether the correct structure can still be learned. 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. It is proven that this only happens by violations of triangle faithfulness (Zhang *et al.*, 2007). To illustrate triangle unfaithfulness, consider $X \to Y \to Z$ and $X \to Z$. There are 3 ways to violate triangle faithfulness :

– (TRUFF1) $X \perp\!\!\!\perp Z$ gives faithful model $X \to Y \leftarrow Z$

– (TRUFF2) $Y \perp\!\!\!\perp Z$ gives faithful model $Y \to X \leftarrow Z$

– (TRUFF3) $X \perp\!\!\!\perp Y | Z$ gives faithful model $X \to Z \to Y$

The problem with triangle unfaithfulness is that if we want to take such violations into account, we have to question each unshielded collider. Since each unshielded collider can be a case of triangle unfaithfulness instead of a v-structure. The good thing about the results of (Zhang *et al.*, 2007) is that except from triangle unfaithfulness, all other violations are detectable. Moreover, we see that the violations of adjacency faithfulness result in clearly defined patterns and equivalence classes.

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 :

$\forall X, Y \in \boldsymbol{V}$which are adjacent in Bayesian network :

$$X \not\perp\!\!\!\perp Y \mid OthPa(X - Y) \qquad [1]$$

where $OthPa(X - Y)$ 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'.

An important extension to the PC algorithm in order to cope with violations of faithfulness, is the Conservative PC algorithm (CPC). Instead of relying on Orientation-Faithfulness, it tests for violations of it. If a violation is detected, a safety measure is taken : the triple is not oriented. Therefore, the result of the CPC algorithm will less frequently draw wrong conclusions about the causal relationship between variables. This is important, since causal inference queries are dependent on correct causal orientations. $Adj(G, X)$ denotes the set of nodes adjacent to X in graph G. Single stochastic variables are denoted by capital letters, sets of variables by boldface capital letters. Step 3 consists of extensions to the original PC algorithm (Spirtes *et al.*, 1993) in which Orientation-Faithfulness is tested (Ramsey *et al.*, 2006). Edges of an unshielded triple are not oriented if a failure is detected, but are indicated as unfaithful. The algorithm returns a correct e-pattern under Causal Sufficiency (there are no common latent causes) and Adjacency-Faithfulness. An e-pattern is a partially-oriented DAG in which some triples are denoted as unfaithful, see Fig. 1(a).

Algorithm 1 The CPC algorithm

S1 Start with the complete undirected graph U on the set of variables $\boldsymbol{V}$.

Part 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 $Adj(U, A)\backslash\{B\}$ and the subsets of $Adj(U, B)\backslash\{A\}$ that have exactly n variables. For all such subsets $\boldsymbol{S}$ check independency $A \perp\!\!\!\perp B \mid \boldsymbol{S}$. If independent, remove the edge between A and B in U, and record $\boldsymbol{S}$ as $Sepset(A, B)$;

$n = n + 1$;

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

Part II ***Orientation.***

S3 Let G be the undirected graph resulting from step S2. For each unshielded triple $\langle A, B, C\rangle$ 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 partners :

(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 (Meek, 1995a), but not on unfaithful triples.

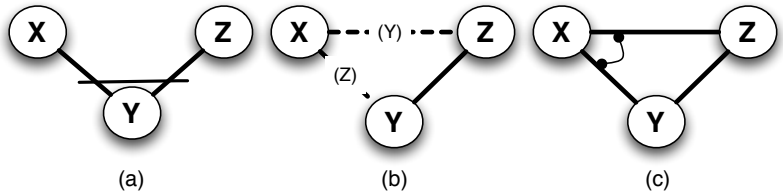

Figure 1. *The three cases of uncertainty : (a) an unfaithful triple by violation of orientation faithfulness for unshielded triple $\langle A, B, C\rangle$, (b) PPIRs when $X \perp\!\!\!\perp Y$ in model $X \rightarrow Y \leftarrow Z$ and (c) equivalent edges when $X \perp\!\!\!\perp Y|Z$ in model $X \rightarrow Y \rightarrow Z$.*

3. Violation of Adjacency Faithfulness

In all other cases than triangle unfaithfulness, there are CIs that make violation of faithfulness detectable. Two patterns of CIs enable the detection of violations of adjacency-faithfulness, in which the CPC algorithm fails. Consider $X \rightarrow Y$, X and Y are adjacent. There are 2 kinds of violations : one in which X and Y are marginally independent ($X \perp\!\!\!\perp Y$) and one in which they become independent when conditioned on some $\boldsymbol{Z}$ ($X \perp\!\!\!\perp Y \mid \boldsymbol{Z}$).

3.1. *Violation by Marginal Independence*

Whenever $X \perp\!\!\!\perp Y$ for some adjacent variables X and Y, we call $X - Y$ a *Pseudo-Independent Relation* (PIR). By Eq. 1, there exists at least one subset of $\boldsymbol{V}$, namely $OthPa(X - Y)$, which turns the independency into a dependency after conditioning. We call any such subset a *depset* of X and Y, written as $depset_{XY}$. The term depset comes from dependency set and is analogous to sepset, defined as a set which separates two variables. A special case of PIRs is defined as *pseudo-independent models* (Xiang *et al.*, 1996), in which three variables are pairwise independent but become dependent when conditioned on the third variable.

Take Z an element of $depset_{XY} \subset OthPa(X - Y)$. Z is related to Y. If Z is also related to X, $X \perp\!\!\!\perp Y$ is a result of triangle unfaithfulness (TRUFF1 or TRUFF2). By excluding this case, $X \rightarrow Y \leftarrow Z$ is an unshielded collider for which there is a $\boldsymbol{U}$ such that $X \perp\!\!\!\perp Z|\boldsymbol{U}$ and $X \not\perp\!\!\!\perp Z|Y, \boldsymbol{U}$[1]. The pattern of CIs implied by an unshielded collider is the same as the PIR-pattern. In other words, a PIR is part of one or more triangles in which it cannot be distinguished from v-structures. A PIR leads to two equivalent structures. In our case of three variables, $X \rightarrow Z \leftarrow Y$ would lead to the same set of independencies if $X - Z$ is also a PIR.

1. This dependency, by which v-structures are identified, appears in most cases, but there are parameterizations of $P(Y|X, Z)$ possible such that $X \perp\!\!\!\perp Z|Y, U$.

We describe this pattern by connecting variables which are mutual 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. 1(b). A PPIR can thus be a PIR or come from a v-structure.

For not overloading the rest of the discussion we assume that for each PIR there exists a depset with one element.

Assumption 1 *If X and Y are adjacent, and $X \perp\!\!\!\perp Y$, there exists a $Z \in OthPa(X - Y)$ such that $X \not\perp\!\!\!\perp Y|Z$.*

It must be noted that the outer variables in a v-structure, X and Z in the example, do not always become dependent by conditioning on the collider, as $X \not\perp\!\!\!\perp Z|Y$ in the example.

3.2. *Violation by a non-Markovian Conditional Independence*

A second violation of adjacency faithfulness happens when $X \not\perp\!\!\!\perp Y|depset_{XY}$ and there is a set $\mathbf{Z}$ for which $X \perp\!\!\!\perp Y \mid \mathbf{Z} \cup depset_{XY}$ where for each $Z \in \mathbf{Z} : X \not\perp\!\!\!\perp Y \mid \mathbf{Z} \setminus Z$. Based on this independence, the PC algorithm would wrongly remove the edge between X and Y. To detect that this is a wrong decision we have to look how the elements of $\mathbf{Z}$ relate to X and Y. Four cases should be considered.

3.2.1. *Case (1) Information Equivalence.*

Variable $Z \in \mathbf{Z}$ becomes independent from X or Y when conditioned on the other variable : $X \perp\!\!\!\perp Z \mid Y$ or $Y \perp\!\!\!\perp Z \mid X$. This happens for example when Z is connected to X or Y, but not to both. It follows that Z can be d-separated from either X or Y by conditioning on the other variable.

Deterministic relations lead to such cases. Consider the structure $X \rightarrow Y \rightarrow Z$ and the functional relation $Y = f(Z)$. Two conditional independencies follow :

$$X \perp\!\!\!\perp Z \mid Y \ \ \& \ \ X \perp\!\!\!\perp Y \mid Z. \qquad [2]$$

We call Y and Z information equivalent with respect to X (Lemeire, 2007). The first equation comes from the Markov condition, the second is implied by the functional relation : Y is completely determined by Z, so Z has all information about Y. Knowing Z therefore renders Y irrelevant for X. Information equivalences happen when there are deterministic relations, but also under weaker conditions (Lemeire, 2007). The fundamental property is that Y and Z contain the same information about X. By knowing one variable, everything is said about the target variable. By absence of additional information, structures $X \rightarrow Y \rightarrow Z$ and $X \rightarrow Z \leftarrow Y$ are equivalent. Y or Z should be connected to X to explain the dependencies, but we do not have a clue which one it is. We call Y and Z equivalent for X.

3.2.2. *Case (2) Violation of Triangle Faithfulness*

When Z is adjacent to both variables and $X \perp\!\!\!\perp Y \mid Z$, we have the third case of triangle unfaithfulness (TRUFF3).

3.2.3. *Case (3) Z connected to X and Y.*

In this case, Z is d-connected to both variables, but not adjacent to both. It means that there exists a set $\boldsymbol{U}$ which d-separates Z from X or Y. Independency $Z \perp\!\!\!\perp Y | X \cup \boldsymbol{U}$ or $Z \perp\!\!\!\perp X | Y \cup \boldsymbol{U}$ will be used to identify this case.

Take the model in which Z is adjacent to X and connected to Y through U. In that case, the CPC algorithm will first delete edge $X - Y$. Since at that point Y is still connected to Z, all observed dependencies are explained. But with independency $Z \perp\!\!\!\perp Y | X \cup U$, deletion of edge $Z - Y$ results in a model which cannot explain the dependencies. We end up with two equivalent structures : one with edge $X - Y$, the other with edge $Z - Y$. Both models explain all dependencies. But the first cannot explain $X \perp\!\!\!\perp Y \mid Z$, the second cannot explain $Z \perp\!\!\!\perp Y | X \cup U$.

3.2.4. *Case (4) More complex cases are excluded.*

To simplify the rest of the discussion we only allow simple equivalences. We exclude the following complex, exotic cases in which equivalences can happen.

Assumption 2 *For all independencies of the form $X \perp\!\!\!\perp Y | Z \cup \boldsymbol{U} \cup depset_{XY}$, where $\boldsymbol{U}$ is not empty and with a dependency if one of the elements is removed from $\boldsymbol{U}$:*

$$X \perp\!\!\!\perp Z | Y \cup \boldsymbol{W} \cup depset_{XY} \Rightarrow X \perp\!\!\!\perp Z | Y \cup depset_{XY}, \quad [3]$$

$$X \perp\!\!\!\perp Z | Y \cup \boldsymbol{U'} \cup depset_{XY} \Rightarrow X \perp\!\!\!\perp Z | Y \cup depset_{XY}, \quad [4]$$

with $\boldsymbol{U'} \subset \boldsymbol{U}$.

By Eq. 3, the equivalence of sets is excluded in which $\{X, Y\}$ and $\{A, B\}$ are equivalent for Z. Eq. 4 rules out equivalence under conditioning : X and Y equivalent for Z under conditioning by U. In other words : for every $X \perp\!\!\!\perp Y | Z \cup \boldsymbol{U}$ we only have to test for $X \perp\!\!\!\perp Z | Y$ and $Z \perp\!\!\!\perp Y | X$.

3.2.5. *Equivalent edges*

Both in case (1) and case (3) the dependencies and independencies can be explained by 2 equivalent structures in which one edge is replaced by another. We call those edges *equivalent* edges. Equivalent edges are linked with an arc with a node at each end, as shown in Fig. 1(c). Note that multiple edges can be equivalent.

4. The Very Conservative PC Algorithm

Algorithm 2 VCPC algorithm S2'

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

a When $X - Y$ is a PPIR, add $depset_{XY}$ to $\boldsymbol{S}$.

b If $X - Y$ has an equivalent edge $X - Z$ or $Y - Z$ and Z is a member of $\boldsymbol{S}$, skip the test.

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

a If $\boldsymbol{S}$ is empty, look for a T in $Adj(X) \cup Adj(Y)$ for which $X \not\perp\!\!\!\perp Y \mid T$. If such a T exists, do not remove the edge, denote it as a PPIR with depset T.

b If $\boldsymbol{S}$ is not empty, test for all $Z \in \boldsymbol{S}$ whether $X \perp\!\!\!\perp Z | Y \cup depset_{XY}$ and $Z \perp\!\!\!\perp Y | X \cup depset_{XY}$. If for a Z, one of both independencies hold, do not remove edge $X - Y$ and do the following. Assume the first independency is found (if the second independency holds, just swaps X and Y in the following.). (1) If $X - Z$ has been removed due to d-separation by respectively Y or X, add the edge back to the graph and go to (3). (2) If $X - Z$ has been removed due to some other d-separation, leave edge $X - Y$ in the graph, but do not qualify it as equivalent. (3) If $Y \not\perp\!\!\!\perp Z | depset_{XY}$, denote $X - Y$ as equivalent to $X - Z$ in the graph.

c 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) : find another depset among the nodes adjacent to one of both variables of the PPIR. If no such depset can be found, delete the PPIR.

The Very Conservative PC algorithm (VCPC) adds to CPC the rules of S2' to S2 (Alg. 2) and replaces Part II with Part II' (Alg. 3). Besides recording the sepsets for pairs of variables, we will also record depsets. The algorithm returns an *f-pattern*, which is an e-pattern augmented by edges that are denoted as PPIRs and subsets of edges denoted as equivalent. An oriented PPIR in the pattern is identified as a PIR. When we speak about adjacencies, these special edges are also considered. Non-equivalent and non-PPIR edges are called normal edges.

Theorem 3 *(Correctness of VCPC)*

Consider a graph G, a JPD P generated by G and $I(P)$, the set of CIs of P : if minimality, triangle-faithfulness and assumptions 1 and 2 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.

The proof is given in our upcoming UAI 2010 paper.

Algorithm 3 VCPC 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 from CPC, except :

- that unshielded triples containing an equivalent edge are not considered,

- and when testing for unfaithful triples, do the equivalence check as in S2'II for each CI that is found. Do not take the CI into account if an equivalence is found.

Perform S4 from the original algorithm on non-equivalent edges ;

Add the PPIR edges back 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, verify if there is another depset among the other variables related to the PPIR. If not, 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 triple A, B and D, add $depset_{AB}$ to the conditioning set of the independence tests. If a v-structure is found, orient the two triangle edges containing A towards A and delete the third triangle edge if it is a PPIR.

5. 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 edges, where d is randomly chosen between 5 and 25. For each such graph, a random structural equation model was constructed by selecting edge coefficients randomly 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 of 1000 cases was simulated for each of the models, to which the PC, CPC and VCPC algorithms were applied with depth 2 and significance level $\alpha = 0.05$ for each independence test based on Fisher's Z transformation of partial correlation. The output graph was compared to the Markov equivalence class (MEC) of the true DAG. Similar experiments were performed with Bayesian networks defined over a set of binary va-

riables and randomly chosen conditional probabilities. The Chi-Square test was used as independence test.

The table on the next page 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 are counted as undecided 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 undecided edge. Weak edges are false negatives whose nodes are marginally independent and independent conditional on any other parent. 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 'not connected'.

	PC	CPC	VCPC
Edges			
Correct	76.6	75.6	77.8
Undecided	0.0	0.0	110.9
False negatives	**23.4**	**24.4**	**8.6**
Weak	4.1	4.7	4.1
False positives	**4.7**	**4.5**	**12.3**
Not connected	3.4	3.6	9.3
Orientations			
Correct	25.8	32.0	41.7
Undecided	15.7	39.9	42.1
Wrong	**19.9**	**2.1**	**3.7**
False positives	**15.1**	**2.9**	**3.9**

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 undecided. 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 in the reduction of the false positive arrowheads. The VCPC algorithm clearly reduces the number of false negative edges. 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 drops from 19.3%/19.7% for PC/CPC to 4.5% for VCPC. This drop is at the expense of undecided 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. VCPC 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 accidently 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. But these CIs will not pass the equivalence tests of VCPC. Both variables are not connected, so there is no variable that can d-separate it. Due to its conservativeness, VCPC is more sensitive to accidental correlations.

6. Conclusions

We cannot rely on adjacency faithfulness when constructing learning algorithms that are uniform consistent. We showed that under triangle faithfulness, violations can be detected by two patterns : potential pseudo-independent relations (PPIRs) and equivalent edges. Based on both patterns, a set of DAGs 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 Very Conservative PC algorithm detects violations of adjacency-faithfulness. The experimental results confirm that the Conservative PC algorithm reduces the number of wrong orientations and that our Very Conservative version reduces the number of false edge removals.

7. Bibliographie

Lemeire J., Learning Causal Models of Multivariate Systems and the Value of it for the Performance Modeling of Computer Programs, PhD thesis, Vrije Universiteit Brussel, 2007.

Meek C., « Causal Inference and Causal Explanation with Background Knowledge », *Procs of UAI-1995*, p. 403-41, 1995a.

Meek C., « Strong completeness and faithfulness in Bayesian networks », *Procs of UAI-1995*, p. 411-418, 1995b.

Ramsey J., Zhang J., Spirtes P., « Adjacency-Faithfulness and Conservative Causal Inference », *Procs of UAI-2006*, 2006.

Spirtes P., Glymour C., Scheines R., *Causation, Prediction, and Search*, 2nd edn, Springer Verlag, 1993.

Xiang Y., Wong S. K., Cercone N., « Critical Remarks on Single Link Search in Learning Belief Networks », *Procs of UAI-1996*, p. 564-571, 1996.

Zhang J., Spirtes P., « Detection of Unfaithfulness and Robust Causal Inference », *Procs of the LSE-Pitt Conference : Confirmation, Induction and Science, London*, 2007.