A Constraint-Based Algorithm For Causal Discovery with Cycles, Latent Variables & Selection Bias

Eric V. Strobl

Pittsburgh, PA

Abstract

Causal processes in nature may contain cycles, and real datasets may violate causal sufficiency as well as contain selection bias. No constraint-based causal discovery algorithm can currently handle cycles, latent variables and selection bias (CLS) simultaneously. I therefore introduce an algorithm called Cyclic Causal Inference (CCI) that makes sound inferences with a conditional independence oracle under CLS, provided that we can represent the cyclic causal process as a non-recursive linear structural equation model with independent errors. Empirical results show that CCI outperforms CCD in the cyclic case as well as rivals FCI and RFCI in the acyclic case.

1. The Problem

Scientists often infer causation using data collected from randomized controlled experiments. However, randomized experiments can be slow, non-generalizable, unethical or expensive. Consider for example trying to discover the causes of a human illness, a common scenario in modern medical science. Performing interventions with possibly harmful consequences on humans is unethical, so scientists often perform experiments on animals instead knowing full well that the causal relationships discovered in animals may not generalize to humans. Moreover, many possible causes for an illness often exist, so scientists typically perform numerous animal experiments in order to discover the causes. A lengthy trial and error process therefore ensues at considerable financial expense. We would ideally like to speed up this scientific process by discovering causation directly from human observational data, which we can more easily acquire.

The need for faster causal discovery has motivated many to develop algorithms for inferring causation from observational data. The PC algorithm, for example, represents one the earliest algorithms for inferring causation using i.i.d. data collected from an underlying acyclic causal process (Spirtes et al., 2000). PC actually falls within a wider class of causal discovery algorithms called constraint-based (CB) algorithms which utilize a conditional independence (CI) oracle, or a CI test in the finite sample case, to re-construct the underlying causal graph. The FCI algorithm is another example of a CB algorithm which extends PC to handle latent variables and selection bias (Spirtes et al., 2000, Zhang, 2008). Yet another CB algorithm called CCD cannot handle latent variables (and perhaps selection bias) like FCI, but CCD can infer cyclic causal structure provided that all causal relations are linear (Richardson, 1996, Richardson and Spirtes, 1999). Many other CB algorithms exist, and most of these methods come with some guarantee of soundness in the sense that their outputs are provably correct with a CI oracle.

The aforementioned CB algorithms and many other non-CB algorithms have been successful in inferring causation under their respective assumptions. However, causal processes and datasets encountered in practice may not satisfy the assumptions of the algorithms. In particular, many causal processes are known to contain feedback loops (Sachs et al., 2005), and datasets may contain latent variables as well as some degree of selection bias (Spirtes et al., 1995). Few algorithms can handle cycles, latent variables and selection bias (CLS) simultaneously, so scientists often must unwillingly apply other methods knowing that the outputs may introduce unwanted bias (Sachs et al., 2005, Mooij and Heskes, 2013). Solving the problem of causal discovery under CLS would therefore provide a much needed basis for justifying the output of causal discovery algorithms when run on real data.

A few investigators have devised non-CB based solutions for the problem of causal discovery under CLS. Hyttinen et al. (2013) introduced the first approach, where CI constraints are fed into a SAT solver which then outputs a graph consistent with the constraints. However, the method can be slow because the SAT solver does not construct efficient test schedules like CB algorithms. Strobl (2017) provided a different solution in the Gaussian case, provided that the cyclic causal process can be decomposed into a set of acyclic ones. The method uses both conditional independence testing and mixture modeling, but the mixture modeling inhibits a straightforward extension of the method to the non-parametric setting even in the linear case. Existing solutions to the problem of causal discovery under CLS thus fall short in either efficiency or generalizability.

The purpose of this paper is to introduce the first CB algorithm that is sound under CLS. The method is efficient because it constructs small test schedules, and it is generalizable to the non-parametric setting because the algorithm only requires a sound CI test. We introduce the new CB algorithm as follows. We first provide provide background material on causal discovery without cycles in Sections 2 through 4. We then review causal discovery with cycles in Section 5. Section 6 introduces the new notion of a maximal almost ancestral graph (MAAG) for summarizing cyclic graphs with latent variables and selection bias. Next, Section 7 contains an overview of the proposed algorithm, while Section 8 outlines an algorithm trace. Section 9 subsequently lists the details of the proposed CB algorithm. Experimental results are included in Section 10. We finally conclude the paper in Section 11. Most of the proofs are located in the Appendix.

2. Graph Terminology

Let italicized capital letters such as A denote a single variable and bolded as well as italicized capital letters such as $\boldsymbol{A}$ denote a set of variables (unless specified otherwise). We will also use the terms "variables" and "vertices" interchangeably.

A graph $\mathbb{G} = (\boldsymbol{X}, \mathcal{E})$ consists of a set of vertices $\boldsymbol{X} = \{X_1, \dots, X_p\}$ and a set of edges $\mathcal{E}$ between each pair of vertices. The edge set $\mathcal{E}$ may contain the following six edge types: $\rightarrow$ (directed), $\leftrightarrow$ (bidirected), — (undirected), $\circ\!\!\rightarrow$ (partially directed), $\circ\!\!-$ (partially undirected) and $\circ\!\!-\!\!\circ$ (nondirected). Notice that these six edges utilize three types of endpoints including *tails*, *arrowheads*, and *circles*.

We call a graph containing only directed edges as a *directed graph*. We will only consider directed graphs without self-loops in this paper. On the other hand, a *mixed graph* contains directed, bidirected and undirected edges. We say that X_i and X_j are *adjacent* in a graph, if they are connected by an edge independent of the edge's type. An *(undirected) path* Π between X_i and X_j is a set of consecutive edges (also independent of their type) connecting the variables such that no vertex is visited more than once. A *directed path* from X_i to X_j is a set of consecutive directed edges from X_i to X_j in the direction of the arrowheads. A *cycle* occurs when a path exists from X_i to X_j, and X_j and X_i are adjacent. More specifically, a directed path from X_i to X_j forms a *directed cycle* with the directed edge $X_j \rightarrow X_i$ and an *almost* *directed cycle* with the bidirected edge $X_j \leftrightarrow X_i$. We call a directed graph a *directed acyclic graph* (DAG), if it does not contain directed cycles.

Three vertices $\{X_i, X_j, X_k\}$ form an *unshielded triple*, if X_i and X_j are adjacent, X_j and X_k are adjacent, but X_i and X_k are not adjacent. On the other hand, the three vertices form a *triangle* when X_i and X_k are also adjacent. We call a nonendpoint vertex X_j on a path Π a *collider* on Π, if both the edges immediately preceding and succeeding the vertex have an arrowhead at X_j. Likewise, we refer to a nonendpoint vertex X_j on Π which is not a collider as a *non-collider*. Finally, an unshielded triple involving $\{X_i, X_j, X_k\}$ is more specifically called a *v-structure*, if X_j is a collider on the subpath $\langle X_i, X_j, X_k \rangle$.

We say that X_i is an *ancestor* of X_j (and X_j is a *descendant* of X_i) if and only if there exists a directed path from X_i to X_j or $X_i = X_j$. We write $X_i \in \text{Anc}(X_j)$ to mean X_i is an ancestor of X_j and $X_j \in \text{Des}(X_i)$ to mean X_j is a descendant of X_i. We also apply the definitions of an ancestor and descendant to a set of vertices $\boldsymbol{Y} \subseteq \boldsymbol{X}$ as follows:

$$\begin{aligned}
\text{Anc}(\boldsymbol{Y}) &= \{X_i | X_i \in \text{Anc}(X_j) \text{ for some } X_j \in \boldsymbol{Y}\},\\
\text{Des}(\boldsymbol{Y}) &= \{X_i | X_i \in \text{Des}(X_j) \text{ for some } X_j \in \boldsymbol{Y}\}.
\end{aligned}$$

3. Causal & Probabilistic Interpretations of DAGs

We will interpret DAGs in a causal fashion (Spirtes et al., 2000, Pearl, 2009). To do this, we consider a stochastic causal process with a distribution $\mathbb{P}$ over $\boldsymbol{X}$ that satisfies the *Markov property*. A distribution satisfies the Markov property if it admits a density that "factorizes according to the DAG" as follows:

$$f(\boldsymbol{X}) = \prod_{i=1}^{p} f(X_i | \text{Pa}(X_i)). \tag{1}$$

We can in turn relate the above equation to a graphical criterion called d-connection. Specifically, if $\mathbb{G}$ is a directed graph in which $\boldsymbol{A}$, $\boldsymbol{B}$ and $\boldsymbol{C}$ are disjoint sets of vertices in $\boldsymbol{X}$, then $\boldsymbol{A}$ and $\boldsymbol{B}$ are *d-connected* by $\boldsymbol{C}$ in the directed graph $\mathbb{G}$ if and only if there exists an *active or d-connecting path* Π between some vertex in $\boldsymbol{A}$ and some vertex in $\boldsymbol{B}$ given $\boldsymbol{C}$. An active path between $\boldsymbol{A}$ and $\boldsymbol{B}$ given $\boldsymbol{C}$ refers to an undirected path Π between some vertex in $\boldsymbol{A}$ and some vertex in $\boldsymbol{B}$ such that, for any collider X_i on Π, a descendant of X_i is in $\boldsymbol{C}$ and no non-collider on Π is in $\boldsymbol{C}$. A path is *inactive* when it is not active. Now $\boldsymbol{A}$ and $\boldsymbol{B}$ are *d-separated* by $\boldsymbol{C}$ in $\mathbb{G}$ if and only if they are not d-connected by $\boldsymbol{C}$ in $\mathbb{G}$. For shorthand, we will write $\boldsymbol{A} \perp\!\!\!\perp_d \boldsymbol{B}|\boldsymbol{C}$ and $\boldsymbol{A} \not\perp\!\!\!\perp_d \boldsymbol{B}|\boldsymbol{C}$ when $\boldsymbol{A}$ and $\boldsymbol{B}$ are d-separated or d-connected given $\boldsymbol{C}$, respectively. The conditioning set $\boldsymbol{C}$ is called a *minimal separating set* if and only if $\boldsymbol{A} \perp\!\!\!\perp_d \boldsymbol{B}|\boldsymbol{C}$ but $\boldsymbol{A}$ and $\boldsymbol{B}$ are d-connected given any proper subset of $\boldsymbol{C}$.

If we have $\boldsymbol{A} \perp\!\!\!\perp_d \boldsymbol{B}|\boldsymbol{C}$, then $\boldsymbol{A}$ and $\boldsymbol{B}$ are conditionally independent given $\boldsymbol{C}$, denoted as $\boldsymbol{A} \perp\!\!\!\perp \boldsymbol{B}|\boldsymbol{C}$, in any joint density factorizing according to (1) (Lauritzen et al., 1990); we refer to this property as the *global directed Markov property*. We also refer to the converse of the global directed Markov property as *d-separation faithfulness*; that is, if $\boldsymbol{A} \perp\!\!\!\perp \boldsymbol{B}|\boldsymbol{C}$, then $\boldsymbol{A}$ and $\boldsymbol{B}$ are d-separated given $\boldsymbol{C}$. One can in fact show that the factorization in (1) and the global directed Markov property are equivalent, so long as the distribution over $\boldsymbol{X}$ admits a density (Lauritzen et al., 1990). We will only consider distributions which admit densities in this report, so we will use the terms "distribution" and "density" interchangeably from here on out.

4. Ancestral Graphs for DAGs

We can associate a directed graph $\mathbb{G}$ with a mixed graph $\mathbb{G}'$ with arbitrary edges as follows. For any directed graph $\mathbb{G}$ with vertices $\boldsymbol{X}$, we consider the partition $\boldsymbol{X} = \boldsymbol{O} \cup \boldsymbol{L} \cup \boldsymbol{S}$, where $\boldsymbol{O}$, $\boldsymbol{L}$ and $\boldsymbol{S}$ are non-overlapping sets of observable, latent and selection variables, respectively. We then consider a mixed graph $\mathbb{G}'$ over $\boldsymbol{O}$, where the arrowheads and tails have the following interpretations. If we have the arrowhead $O_i *\!\!\rightarrow O_j$, where the asterisk is a meta-symbol denoting either a tail or an arrowhead, then we say that O_j is not an ancestor of $O_i \cup \boldsymbol{S}$ in $\mathbb{G}$. On the other hand, if we have the tail $O_i *\!\!- O_j$ then we say that O_j is an ancestor of $O_i \cup \boldsymbol{S}$ in $\mathbb{G}$. Let $\mathrm{Anc}(X_i)$ denote the ancestors of X_i in $\mathbb{G}$. Obviously then, any $\mathbb{G}'$ constructed from a directed graph cannot have a *directed cycle*, where $O_i \rightarrow O_j$ in $\mathbb{G}'$ and $O_j \in \mathrm{Anc}(O_i \cup \boldsymbol{S})$ in $\mathbb{G}$. Similarly, $\mathbb{G}'$ cannot have an *almost directed cycle*, where $O_i \leftrightarrow O_j$ is in $\mathbb{G}'$ and $O_j \in \mathrm{Anc}(O_i \cup \boldsymbol{S})$ in $\mathbb{G}$.

One can also show that, if $\mathbb{G}$ is acyclic, then any mixed graph constructed from $\mathbb{G}$ cannot have a undirected edge $O_i - O_j$ with incoming arrowheads at O_i or O_j (Richardson and Spirtes, 2000). We therefore find it useful to consider a subclass of mixed graphs called *ancestral graphs*:

Definition 1. *(Ancestral Graphs) A mixed graph $\mathbb{G}'$ is more specifically called an ancestral graph if and only if $\mathbb{G}'$ satisfies the following three properties:*

1. There is no directed cycle.

2. There is no almost directed cycle.

3. For any undirected edge $O_i - O_j$, O_i and O_j have no incoming arrowheads.

Observe that every mixed graph of a DAG is an ancestral graph.

A *maximal ancestral graph* (MAG) is an ancestral graph where every missing edge corresponds to a conditional independence relation. One can transform a DAG $\mathbb{G}$ into a MAG $\mathbb{G}'$ as follows. First, for any pair of vertices $\{O_i, O_j\}$, make them adjacent in $\mathbb{G}'$ if and only if there is an *inducing path* between O_i and O_j in $\mathbb{G}$. We define an inducing path as follows:

Definition 2. *(Inducing Path) A path Π between O_i and O_j in $\mathbb{G}$ is called an inducing path if and only if every collider on Π is an ancestor of $\{O_i, O_j\} \cup \boldsymbol{S}$, and every non-collider on Π (except for the endpoints) is in $\boldsymbol{L}$.*

Note that two observables O_i and O_j are connected by an inducing path if and only if they are d-connected given any $\boldsymbol{W} \subseteq \boldsymbol{O} \setminus \{O_i, O_j\}$ as well as $\boldsymbol{S}$ (Spirtes et al., 2000). Then, for each adjacency $O_i \ast\!\!-\!\!\ast\, O_j$ in $\mathbb{G}'$, we have the following edge interpretations:

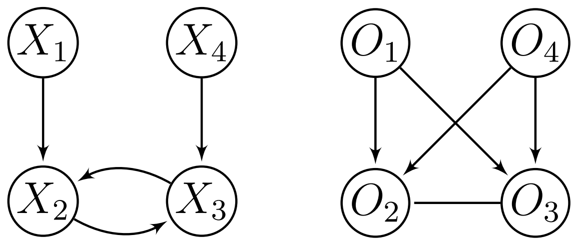
1. If we have $O_i \ast\!\!\rightarrow O_j$, then $O_j \notin \mathrm{Anc}(O_i \cup \boldsymbol{S})$ in $\mathbb{G}$.

2. If we have $O_i \ast\!\!-\!\!- O_j$, then $O_j \in \mathrm{Anc}(O_i \cup \boldsymbol{S})$ in $\mathbb{G}$.

The MAG of a DAG is therefore a kind of marginal graph that does not contain the latent or selection variables, but does contain information about the ancestral relations between the observable and selection variables in the DAG. The MAG also has the same d-separation relations as the DAG, specifically among the observable variables conditional on the selection variables (Spirtes and Richardson, 1996).

5. Directed Cyclic Graphs as Equilibriated Causal Processes

We now allow cycles in a directed graph. Multiple different causal representations of a directed cyclic graph exist in the literature. Examples include dynamic Bayesian networks (Dagum et al., 1995), structural equation models with feedback (Spirtes, 1995), chain graphs (Lauritzen and Richardson, 2002) and mixtures of DAGs (Strobl, 2017). See (Strobl, 2017) for a discussion of



(a) (b)

Figure 1: (a) The graph $\mathbb{G}$ associated with the SEM-IE in Equation (2). (b) The associated MAAG.

each of their strengths and weaknesses. In this report, we will only consider structural equation models with feedback.

Recall that the density $f(\boldsymbol{X})$ associated with a DAG $\mathbb{G}$ obeys the global Markov property. However, the density may not obey the global Markov property if $\mathbb{G}$ contains cycles. We therefore must impose certain assumptions on $\mathbb{P}$ such that its density does obey the property.

Spirtes (1995) proposed the following assumptions on $\mathbb{P}$. We say that a distribution $\mathbb{P}$ obeys a *structural equation model with independent errors* (SEM-IE) with respect to $\mathbb{G}$ if and only if we can describe $\boldsymbol{X}$ as $X_i = g_i(\text{Pa}(X_i), \varepsilon_i)$ for all $X_i \in \boldsymbol{X}$ such that X_i is $\sigma(\text{Pa}(X_i), \varepsilon_i)$ measurable and $\varepsilon_i \in \boldsymbol{\varepsilon}$ (Evans, 2016). Here, we have a set of jointly independent errors $\boldsymbol{\varepsilon}$, and $\sigma(Y)$ refers to the sigma-algebra generated by the random variable Y. An example of an SEM-IE is illustrated below with an associated directed graph drawn in Figure 1a:

$$\begin{aligned} X_1 &= \varepsilon_1, \\ X_2 &= B_{21}X_1 + B_{23}X_3 + \varepsilon_2, \\ X_3 &= B_{34}X_4 + B_{32}X_2 + \varepsilon_3, \\ X_4 &= \varepsilon_4, \end{aligned} \tag{2}$$

where $\boldsymbol{\varepsilon}$ denotes a set of jointly independent standard Gaussian error terms, and B is a 4 by 4 coefficient matrix. Notice that the structural equations in (2) are linear structural equations.

We can simulate data from an SEM-IE using the *fixed point method* (Fisher, 1970). The fixed point method involves two steps per sample. We first sample the error terms according to their independent distributions and initialize $\boldsymbol{X}$ to some values. Next, we apply the structural equations iteratively until the values of the random variables converge to values which satisfy the structural equations; in other words, the values converge almost surely to a fixed point.[1] Note that the values of the random variables may not necessarily converge to a fixed point all of the time for every set of structural equations and error distributions, but we will only consider those structural equations and error distributions which do satisfy this property. We call the distribution reached at the fixed points as the *equilibrium distribution*.

Spirtes (1995) proved the following regarding *linear SEM-IEs*, or SEM-IEs with linear structural equations:

Theorem 1. *The equilibrium distribution $\mathbb{P}$ of a linear SEM-IE satisfies the global directed Markov property with respect to the SEM-IE's directed graph $\mathbb{G}$ (acyclic or cyclic).*

The above theorem provided a basis from which Richardson started constructing the Cyclic Causal Discovery (CCD) algorithm (Richardson, 1996, Richardson and Spirtes, 1999) for causal discovery with feedback.

6. Almost Ancestral Graphs for Directed Graphs

Recall that an ancestral graph satisfies the three properties listed in Definition 1. We now define an *almost ancestral graph* (AAG) which only satisfies the first two conditions of an ancestral graph. The following result should be obvious:

Proposition 1. *Any mixed graph $\mathbb{G}'$ constructed from a directed graph $\mathbb{G}$ (cyclic or acyclic) over $\boldsymbol{O}$ is an AAG.*

Proof. No directed cycle and no almost directed cycle can exist in $\mathbb{G}'$ because that would imply that there exists a vertex O_j which is simultaneously both an ancestor of $O_i \cup \boldsymbol{S}$ and not an ancestor of $O_i \cup \boldsymbol{S}$ in $\mathbb{G}$. $\square$

Now an almost ancestral graph is said to *maximal* when an edge exists between any two vertices O_i and O_j if and only if there exists an inducing path between O_i and O_j. Note that a maximal almost ancestral graph (MAAG) $\mathbb{G}'$

[1] We can perform the fixed point method more efficiently in the linear case by first representing the structural equations in matrix format: $\boldsymbol{X} = B\boldsymbol{X} + \boldsymbol{\varepsilon}$. Then, after drawing the values of $\boldsymbol{\varepsilon}$, we can obtain the values of $\boldsymbol{X}$ by solving the following system of equations: $\boldsymbol{X} = (\mathbb{I} - B)^{-1}\boldsymbol{\varepsilon}$, where $\mathbb{I}$ denotes the identity matrix.

does not necessarily preserve the d-separation relations between the variables in $\boldsymbol{O}$ given $\boldsymbol{S}$ in a directed graph $\mathbb{G}$, even though $\mathbb{G}'$ does do so when $\mathbb{G}$ is acyclic (Spirtes and Richardson, 1996). We provide an example of an MAAG in Figure 1b, where $\boldsymbol{X} = \boldsymbol{O}$ because $\boldsymbol{L} = \emptyset$ and $\boldsymbol{S} = \emptyset$.

7. Overview of the Proposed Algorithm

We now introduce a new CB algorithm called Cyclic Causal Inference (CCI) for discovering causal relations under CLS. We first provide a bird's eye view of the algorithm (for more details, see Section 9). We will assume that the reader is familiar with past CB algorithms including PC, FCI, RFCI and CCD; for a brief review of each, see Section 12 in the Appendix.

We have summarized CCI in Algorithm 1. The algorithm first performs FCI's skeleton discovery procedure in Step 1 (see Algorithm 5), which discovers a graph where each adjacency between any two observables O_i and O_j corresponds to an inducing path even in the cyclic case (see Lemma 4 of Section 13 for a proof of this statement). The algorithm then orients v-structures in Step 2 using FCI's v-structure discovery procedure (Algorithm 4).

Step 3 of CCI checks for additional long range d-separation relations. Recall that the PC algorithm only checks for short range d-separation relations via v-structures. However, two cyclic directed graphs may agree locally on d-separation relations, but disagree on d-separation relations between distant variables, even if they do not contain any latent variables or selection bias (Richardson, 1994). As a result, Step 3 allows the algorithm to orient additional edges by checking for additional d-separation relations.

Step 4 of CCI discovers non-minimal d-separating sets which were not discovered in Step 1. Recall that, if we have $O_i {*\!\!\rightarrow} O_j \leftarrow\!\!* O_k$ with O_i and O_k non-adjacent, then every set d-separating O_i and O_k does not contain O_j. However, in the cyclic case, O_i and O_k can be d-separated given a set that contains O_j. It turns out that we can infer additional properties about the MAAG, if we find d-separating sets which contain O_j. Step 4 of Algorithm 1 therefore discovers these additional non-minimal d-separating sets using Algorithm 2. Steps 5 and 6 in turn utilize the d-separating sets discovered in Steps 1 and 4 in order to orient additional edges. Finally, Step 7 applies the 7 orientation rules described in Section 9. CCI thus ultimately outputs a *partially oriented MAAG*, or an MAAG with tails, arrowheads and unspecified endpoints denoted by circles.

Data: CI oracle
Result: $\widehat{\mathbb{G}}$

1 Run FCI's skeleton discovery procedure (Algorithm 5).
2 Run FCI's v-structure orientation procedure (Algorithm 4).
3 For any triple of vertices $\langle O_i, O_k, O_j \rangle$ such that we have $O_k \circ\!\!-\!\!* O_i$, if there is a set in $\text{Sep}(O_i, O_j)$ discovered in Step 1 such that $O_k \notin \text{Sep}(O_i, O_j)$, $O_i \not\perp\!\!\!\perp_d O_k | \text{Sep}(O_i, O_j) \cup \boldsymbol{S}$ and $O_j \not\perp\!\!\!\perp_d O_k | \text{Sep}(O_i, O_j) \cup \boldsymbol{S}$, then orient $O_k \circ\!\!-\!\!* O_i$ as $O_k \leftarrow\!\!* O_i$.
4 Find additional non-minimal d-separating sets using Algorithm 2.
5 Find all quadruples of vertices $\langle O_i, O_j, O_k, O_l \rangle$ such that O_i and O_k non-adjacent, $O_i *\!\!\rightarrow O_l \leftarrow\!\!* O_k$, and $O_i \perp\!\!\!\perp_d O_k | \boldsymbol{W} \cup \boldsymbol{S}$ with $O_j \in \boldsymbol{W}$ and $\boldsymbol{W} \subseteq \boldsymbol{O} \setminus \{O_i, O_k\}$. If $O_l \notin \boldsymbol{W} = \text{Sep}(O_i, O_k)$ as discovered in Step 1, then orient $O_j *\!\!-\!\!\circ O_l$ as $O_j *\!\!\rightarrow O_l$. If we also have $O_i *\!\!\rightarrow O_j \leftarrow\!\!* O_k$ and $O_l \in \boldsymbol{W} = \text{SupSep}(O_i, O_j, O_k)$ as discovered in Step 4, then orient $O_j *\!\!-\!\!\circ O_l$ as $O_j *\!\!- O_l$.
6 If we have $O_i \perp\!\!\!\perp_d O_k | \boldsymbol{W} \cup \boldsymbol{S}$ for some $\boldsymbol{W} \subseteq \boldsymbol{O} \setminus \{O_i, O_k\}$ with $O_j \in \boldsymbol{W}$ but we have $O_i \not\perp\!\!\!\perp_d O_k | O_l \cup \boldsymbol{W} \cup \boldsymbol{S}$, then orient $O_l \circ\!\!-\!\!* O_j$ as $O_l \leftarrow\!\!* O_j$.
7 Execute orientation rules 1-7.

Algorithm 1: Cyclic Causal Inference (CCI)

Now Algorithm 1 is sound due to the following theorem:

Theorem 2. *Consider a DAG or a linear SEM-IE with directed cyclic graph $\mathbb{G}$. If d-separation faithfulness holds, then CCI outputs a partially oriented MAAG of $\mathbb{G}$.*

8. Algorithm Trace

We now illustrate a sample run of the CCI algorithm with a CI oracle. Consider the directed graph in Figure 2a containing just one latent variable L_1 with MAAG in Figure 2b. CCI proceeds as follows:

Step 1: Discovers the skeleton shown in Figure 2c.

Step 2: Adds two arrowheads onto O_2 yielding Figure 2d because $O_1 \perp\!\!\!\perp_d O_5$.

Step 3: Does not orient any endpoints.

Step 4: Discovers the additional d-separating set $O_1 \perp\!\!\!\perp_d O_5 | \{O_2, O_3\}$.

Step 5: Does not orient any endpoints.

Step 6: Does not orient any endpoints.

Step 7: Rule 1 orients $O_2 \circ\!\!\rightarrow\!\! * O_4$ as $O_2 \rightarrow\!\! * O_4$ and $O_2 \circ\!\!\rightarrow\!\! * O_3$ as $O_2 \rightarrow\!\! * O_3$. Then Rule 4 orients $O_1 \circ\!\!- * O_3$ as $O_1 \leftarrow\!\! * O_3$ and $O_4 \circ\!\!- * O_5$ as $O_4 \leftarrow\!\! * O_5$. Next, Rule 1 again fires twice to orient $O_3 \circ\!\!-\!\!\circ O_4$ as $O_3 - O_4$. Finally Rule 3 also fires twice to orient $O_2 *\!\!-\!\!\circ O_4$ and $O_2 *\!\!-\!\!\circ O_3$ as $O_2 *\!\!- O_4$ and $O_2 *\!\!- O_3$, respectively. The orientation rules in Step 7 therefore yield Figure 2e.

Now we would expect CCD to output a pretty good partially oriented MAAG, given that the directed graph contains only one latent variable and no selection variables. However, CCD outputs the graph in Figure 2e. The output contains one error (O_1 is not an ancestor of O_2) and eight un-oriented endpoints which were oriented by CCI.

9. Algorithm Details

We present the details of CCI. We claim that statements made herein hold for both cyclic and acyclic directed graphs, unless indicated otherwise. Most of the proofs are located in Section 13.

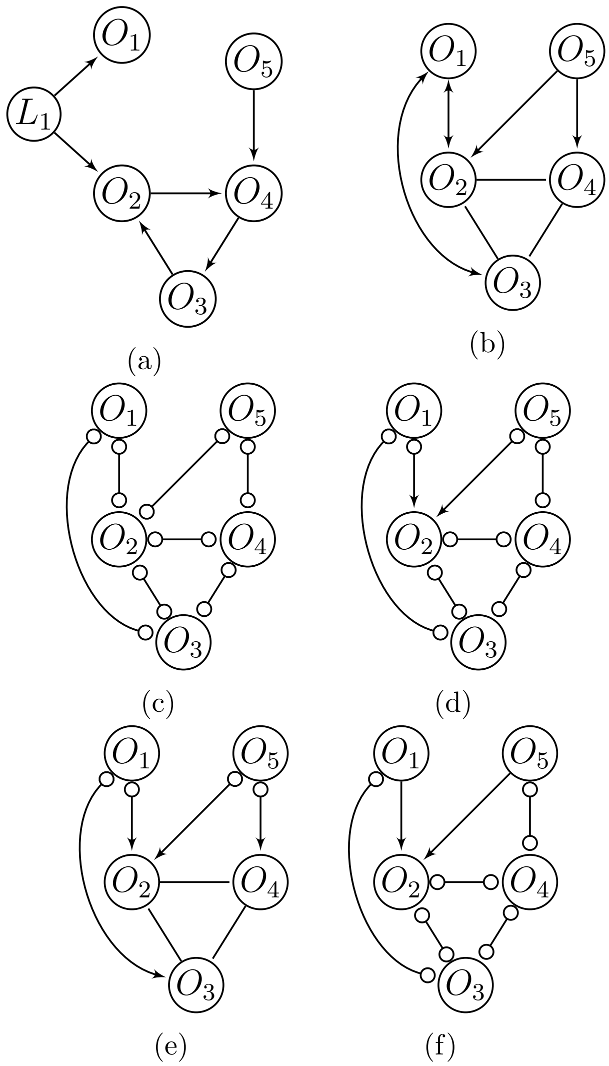


Figure 2: A sample run of CCI. The ground truth directed graph is illustrated in (a) with corresponding MAAG in (b). Step 1 of CCI outputs (c), Step 2 (d), and Step 7 the final output (e). In contrast, CCD outputs (f).

9.1. Step 1: Skeleton Discovery

We first discover the skeleton of an MAAG by consulting a CI oracle. The following result demonstrates that we can discover the skeleton of an MAAG, if we can search over all possible separating sets:

Lemma 1. *There exists an inducing path between O_i and O_j if and only if O_i and O_j are d-connected given $\boldsymbol{W} \cup \boldsymbol{S}$ for all possible subsets $\boldsymbol{W} \subseteq \boldsymbol{O} \setminus \{O_i, O_j\}$.*

Searching over all separating sets is however inefficient. We therefore consider the following sets instead:

Definition 3. *(D-SEP Set) We say that $O_k \in \text{D-SEP}(O_i, O_j)$ in a directed graph $\mathbb{G}$ if and only if there exists a sequence of observables $\Pi = \langle O_i, \ldots, O_k \rangle$ in $\text{Anc}(\{O_i, O_j\} \cup \boldsymbol{S})$ such that, for any subpath $\langle O_{h-1}, O_h, O_{h+1} \rangle$ on Π, we have an inducing path between O_{h-1} and O_h that is into O_h as well as an inducing path between O_{h+1} and O_h that is into O_h.*

Notice that $\text{D-SEP}(O_i, O_j)$ and $\text{D-SEP}(O_j, O_i)$ may not be equivalent. The D-SEP set is important, because we can use it to discover inducing paths without searching over all possible separating sets:

Lemma 2. *If there does not exist an inducing path between O_i and O_j, then O_i and O_j are d-separated given $\text{D-SEP}(O_i, O_j) \cup \boldsymbol{S}$. Likewise, O_i and O_j are d-separated given $\text{D-SEP}(O_j, O_i) \cup \boldsymbol{S}$.*

The D-SEP sets are however not computable. We therefore consider computable possible d-separating sets, which are supersets of the D-SEP sets:

Definition 4. *(Possible D-Separating Set) We say that $O_k \in \text{PD-SEP}(O_i)$ in any partial oriented mixed graph $\widetilde{\mathbb{G}}$ if and only if there exists a path Π between O_i and O_k in $\widetilde{\mathbb{G}}$ such that, for every subpath $\langle O_{h-1}, O_h, O_{h+1} \rangle$ on Π, either O_h is a v-structure or $\langle O_{h-1}, O_h, O_{h+1} \rangle$ forms a triangle.*

The following lemma shows that we can utilize PD-SEP sets in replace of D-SEP sets:

Lemma 3. *If there does not exist an inducing path between O_i and O_j, then O_i and O_j are d-separated given $\boldsymbol{W} \cup \boldsymbol{S}$ with $\boldsymbol{W} \subseteq \text{PD-SEP}(O_i)$ in the MAAG $\mathbb{G}'$. Likewise, O_i and O_j are d-separated given some $\boldsymbol{W} \cup \boldsymbol{S}$ with $\boldsymbol{W} \subseteq \text{PD-SEP}(O_j)$ in $\mathbb{G}'$.*

Recall that a similar lemma was also proven in the acyclic case (Spirtes et al., 2000). We conclude that the procedure for discovering the skeleton of an MAAG is equivalent to that of an MAG in the acyclic case.

The justification of Step 1 in Algorithm 1 then follows by generalizing the above lemma to $\mathbb{G}''$, the partially oriented mixed graph discovered by PC's skeleton and FCI's v-structure discovery procedures utilized in Step 1:

Lemma 4. *If an inducing path does not exist between O_i and O_j in $\mathbb{G}$, then O_i and O_j are d-separated given $\boldsymbol{W} \cup \boldsymbol{S}$ with $\boldsymbol{W} \subseteq \text{PD-SEP}(O_i)$ in $\mathbb{G}''$. Likewise, O_i and O_j are d-separated given some $\boldsymbol{W} \cup \boldsymbol{S}$ with $\boldsymbol{W} \subseteq \text{PD-SEP}(O_j)$ in $\mathbb{G}''$.*

The above lemma holds because $\text{PD-SEP}(O_i)$ formed using $\mathbb{G}'$ is a subset of $\text{PD-SEP}(O_i)$ formed using $\mathbb{G}''$; likewise for $\text{PD-SEP}(O_j)$.

9.2. Steps 2 & 3: Short and Long Range Non-Ancestral Relations

We orient endpoints during v-structure discovery with the following lemma:

Lemma 5. *Consider a set $\boldsymbol{W} \subseteq \boldsymbol{O} \setminus \{O_i, O_j\}$. Now suppose that O_i and O_k are d-connected given $\boldsymbol{W} \cup \boldsymbol{S}$, and that O_j and O_k are d-connected given $\boldsymbol{W} \cup \boldsymbol{S}$. If O_i and O_j are d-separated given $\boldsymbol{W} \cup \boldsymbol{S}$ such that $O_k \notin \boldsymbol{W}$, then O_k is not an ancestor of $\{O_i, O_j\} \cup \boldsymbol{S}$.*

Recall that, if there exists an inducing path between O_i and O_k as well as an inducing path between O_j and O_k, then O_i and O_k are d-connected given $\boldsymbol{W} \cup \boldsymbol{S}$ and O_j and O_k are d-connected given $\boldsymbol{W} \cup \boldsymbol{S}$ by Lemma 1. Moreover, if O_i and O_j are d-separated given $\boldsymbol{W} \cup \boldsymbol{S}$, then an inducing path does not exist between O_i and O_j. This means that, if we are dealing with the partially oriented MAAG in Figure 3a, and O_i and O_j are d-separated given $\boldsymbol{W} \cup \boldsymbol{S}$ such that $O_k \notin \boldsymbol{W}$, then we orient the endpoints in Figure 3a as the arrowheads in Figure 3b. Lemma 5 therefore justifies the v-structure discovery procedure in Step 2 of CCI for short range non-ancestral relations.

Now Lemma 5 also justifies Step 3 of CCI, because O_i and O_k as well as O_j and O_k need not be adjacent in the underlying MAAG. In fact, O_k may be located far from O_i and O_j. CCI utilizes such long range relations because two cyclic directed graphs may agree "locally" on d-separation relations, but disagree on some d-separation relations between distant variables (Richardson, 1994).

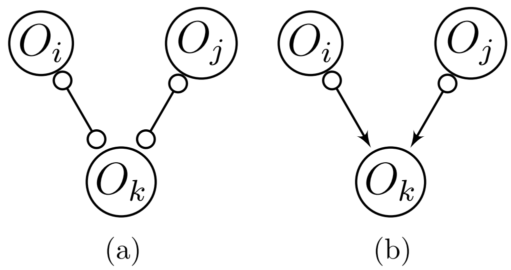


Figure 3: An unshielded triple in (a) oriented to a v-structure in (b) after Step 2 of CCI.

9.3. Step 4: Discovering Non-Minimal D-Separating Sets

The algorithm now utilizes the graph from Step 3 in order to find additional d-separating sets. The skeleton discovery phase of CCI finds minimal d-separating sets, but non-minimal d-separating sets can also inform the algorithm about the underlying cyclic causal graph. Recall that, if we have $O_i {*\!\!\rightarrow} O_j \leftarrow\!\!* O_k$ in the acyclic case, then O_i and O_j are d-connected given $O_j \cup \boldsymbol{W} \cup \boldsymbol{S}$ for any $\boldsymbol{W} \subseteq \boldsymbol{O} \setminus \{O_i, O_j\}$ (Spirtes and Richardson, 1996). The same fact is not true however in the cyclic case. Consider for example the graph in Figure 1a. Here, we know that X_1 and X_4 are d-separated given $\emptyset$, but they are also d-separated given $\{X_2, X_3\}$. Moreover, we have $O_1 \rightarrow O_3 \leftarrow O_4$ in the corresponding MAAG in Figure 1b.

The above example motivates us to search for additional d-separating sets, which will prove to be important for orientating additional circle endpoints as evidenced in the next section. From Lemma 3, we already know that some subset of PD-SEP(O_i) and some subset of PD-SEP(O_k) d-separate O_i and O_k when we additionally condition on $\boldsymbol{S}$. We therefore search for the additional d-separating sets by testing all subsets of PD-SEP(O_i) as well as those of PD-SEP(O_k).

We summarize the details of Step 4 in Algorithm 2. The sub-procedure specifically works as follows. For each v-structure $O_i {*\!\!\rightarrow} O_j \leftarrow\!\!* O_k$, the algorithm determines whether O_i and O_k are d-separated given specific supersets of the minimal separating set $\text{Sep}(O_i, O_k)$. In particular, Algorithm 2 forms the sets $\boldsymbol{T} = \text{Sep}(O_i, O_k) \cup O_j \cup \boldsymbol{W}$ where $\boldsymbol{W} \subseteq \text{PD-SEP}(O_i) \setminus \{\text{Sep}(O_i, O_k) \cup \{O_j, O_k\}\}$ in line 7. The algorithm then consults the CI oracle with $\boldsymbol{T} \cup \boldsymbol{S}$ in line 8. Finally, like the skeleton discovery phase, the algorithm first consults the CI oracle with the smallest subsets and then progresses to larger subsets until such a separating set $\boldsymbol{T} \cup \boldsymbol{S}$ is found or all subsets of $\text{PD-SEP}(O_i) \setminus \{\text{Sep}(O_i, O_k) \cup \{O_j, O_k\}\}$ have been exhausted.

Data: $\widehat{\mathcal{G}}$, CI oracle
Result: $\widehat{\mathcal{G}}$, SupSep

$m = 0$
repeat
    repeat
        select the ordered triple $\langle O_i, O_j, O_k \rangle$ with the v-structure $O_i {*\!\!\rightarrow} O_j \leftarrow\!\!{*} O_k$ such that $|\text{PD-SEP}(O_i)| \geq m$
        repeat
            select a subset $\boldsymbol{W} \subseteq \text{PD-SEP}(O_i) \setminus \{\text{Sep}(O_i, O_k) \cup \{O_j, O_k\}\}$ with m vertices
            $\boldsymbol{T} = \boldsymbol{W} \cup \text{Sep}(O_i, O_k) \cup O_j$
            if O_i and O_k are d-separated given $\boldsymbol{T} \cup \boldsymbol{S}$, then record the set $\boldsymbol{T}$ in $\text{SupSep}(O_i, O_j, O_k)$
        until *all subsets* $\boldsymbol{W} \subseteq \text{PD-SEP}(O_i) \setminus \{\text{Sep}(O_i, O_k) \cup \{O_j, O_k\}\}$ *have been considered or a d-separating set of* O_i *and* O_k *has been recorded in* $\text{SupSep}(O_i, O_j, O_k)$;
    until *all triples* $\langle O_i, O_j, O_k \rangle$ *with the v-structure* $O_i {*\!\!\rightarrow} O_j \leftarrow\!\!{*} O_k$ *and* $|\text{PD-SEP}(O_i)| \geq m$ *have been selected*;
until *all ordered triples* $\langle O_i, O_j, O_k \rangle$ *with the v-structure* $O_i {*\!\!\rightarrow} O_j \leftarrow\!\!{*} O_k$ *have* $|\text{PD-SEP}(O_i)| < m$;

Algorithm 2: Step 4 of CCI

9.4. Step 5: Orienting with Non-Minimal D-Separating Sets

The following lemma justifies Step 5 which utilizes the non-minimal d-separating sets discovered in the previous step:

Lemma 6. *Consider a quadruple of vertices $\langle O_i, O_j, O_k, O_l\rangle$. Suppose that we have:*

1. *O_i and O_k non-adjacent;*
2. *$O_i {*\!\!\rightarrow} O_l \leftarrow\!\!* O_k$;*
3. *O_i and O_k are d-separated given some $\boldsymbol{W} \cup \boldsymbol{S}$ with $O_j \in \boldsymbol{W}$ and $\boldsymbol{W} \subseteq \boldsymbol{O} \setminus \{O_i, O_k\}$;*
4. *$O_j {*\!\!-\!\!\circ} O_l$*

If $O_l \notin \boldsymbol{W} = \mathrm{Sep}(O_i, O_k)$, then we have $O_j {*\!\!\rightarrow} O_l$. If $O_l \in \boldsymbol{W} = \mathrm{SupSep}(O_i, O_j, O_k)$ and in addition we have $O_i {*\!\!\rightarrow} O_j \leftarrow\!\!* O_k$, then we have $O_j {*\!\!-} O_l$.

Notice that the above lemma utilizes $\mathrm{SupSep}(O_i, O_j, O_k)$ as discovered in Step 4.

9.5. Step 6: Long Range Ancestral Relations

We can justify Step 6 with the following result:

Lemma 7. *If O_i and O_k are d-separated given $\boldsymbol{W} \cup \boldsymbol{S}$, where $\boldsymbol{W} \subseteq \boldsymbol{O} \setminus \{O_i, O_k\}$, and $\boldsymbol{Q} \subseteq \mathrm{Anc}(\{O_i, O_k\} \cup \boldsymbol{W} \cup \boldsymbol{S}) \setminus \{O_i, O_k\}$, then O_i and O_k are also d-separated given $\boldsymbol{Q} \cup \boldsymbol{W} \cup \boldsymbol{S}$.*

Notice that the above lemma allows us to infer long range ancestral relations because all variables in $\boldsymbol{Q}$ are ancestors of $\{O_i, O_k\} \cup \boldsymbol{W} \cup \boldsymbol{S}$. Step 6 of Algorithm 1 then follows by the contrapositive of Lemma 7:

Corollary 1. *Assume that O_i and O_k are d-separated by $\boldsymbol{W} \cup \boldsymbol{S}$ with $O_j \in \boldsymbol{W}$ and $\boldsymbol{W} \subseteq \boldsymbol{O} \setminus \{O_i, O_k\}$, but O_i and O_k are d-connected by $O_l \cup \boldsymbol{W} \cup \boldsymbol{S}$. Then, O_l is not an ancestor of $O_j \cup \boldsymbol{S}$.*

9.6. Step 7: Orientation Rules

We will now describe the orientation rules in Step 7. Notice that the orientation rules are always applied after Step 6 and therefore also after v-structure discovery. This ordering implies that, if O_i and O_j are non-adjacent and we have $O_i {*\!\!-\!\!*} O_k {*\!\!-\!\!*} O_j$, but we do not have $O_i {*\!\!\rightarrow} O_k \leftarrow\!\!* O_j$, then $O_k \in \mathrm{Sep}(O_i, O_j)$; this follows because, if $O_k \notin \mathrm{Sep}(O_i, O_j)$, then we would have $O_i {*\!\!\rightarrow} O_k \leftarrow\!\!* O_j$ by v-structure discovery.

9.6.1. First to Third Orientation Rules

Lemma 5 allows us to infer non-ancestral relations. The following lemma allows us to infer ancestral relations:

Lemma 8. *Suppose that there is a set $\boldsymbol{W} \setminus \{O_i, O_k\}$ and every proper subset $\boldsymbol{V} \subset \boldsymbol{W}$ d-connects O_i and O_k given $\boldsymbol{V} \cup \boldsymbol{S}$. If O_i and O_k are d-separated given $\boldsymbol{W} \cup \boldsymbol{S}$ where $O_j \in \boldsymbol{W}$, then O_j is an ancestor of $\{O_i, O_k\} \cup \boldsymbol{S}$.*

The above lemma justifies the following orientation rule:

Lemma 9. *If we have $O_i {*}\!\!\rightarrow O_j {\circ}\!\!-\!\!{*}\, O_k$ with O_i and O_k non-adjacent, then orient $O_j {\circ}\!\!-\!\!{*}\, O_k$ as $O_j -\!{*} O_k$.*

Proof. If we have $O_i {*}\!\!\rightarrow O_j {\circ}\!\!-\!\!{*}\, O_k$ with O_i and O_k non-adjacent, then $O_j \in \mathrm{Sep}(O_i, O_k)$ because we have already performed v-structure discovery. By Lemma 8, we know that $O_j \in \mathrm{Anc}(\{O_i, O_k\} \cup \boldsymbol{S})$. We more specifically know that $O_j \in \mathrm{Anc}(O_k)$ because the arrowhead $O_i {*}\!\rightarrow O_j$ implies that $O_j \notin \mathrm{Anc}(O_i \cup \boldsymbol{S})$. $\square$

For example, if we have the structure in Figure 4a, then we can add a undirected edge as in Figure 4b.

We may also add an arrowhead at O_k in Figure 4b provided that some additional conditions are met. The following lemma is central to causal discovery with cycles:

Lemma 10. *If we have $O_i {*}\!\!\rightarrow O_j — O_k$ with O_i and O_k non-adjacent, then $O_i {*}\!\!\rightarrow O_j$ is in a triangle involving O_i, O_j and O_l ($l \neq k$) with $O_j — O_l$ and $O_i {*}\!\!\rightarrow O_l$. Moreover, there exists a sequence of undirected edges between O_l and O_k that does not include O_j.*

The above statement may appear arcane at first glance, but it justifies multiple orientation rules.

The following definitions are useful towards applying Lemma 10:

Definition 5. *(Potentially Undirected Path) A potentially undirected path Π exists between O_i and O_j if and only if all endpoints on Π are tails or circles.*

Definition 6. *(Potential 2-Triangulation) The edge $O_i \, {*}\!-\!{*} \, O_j$ is said to be potentially 2-triangulated w.r.t. O_k if and only if (1) O_i, O_j and another vertex O_l is in a triangle, (2) we have $O_j — O_l$, $O_j {\circ}\!\!- O_l$, $O_j -\!\!{\circ} O_k$ or $O_j {\circ}\!\!-\!\!{\circ} O_l$, (3) we have $O_i {*}\!\!\rightarrow O_l$ or $O_i {*}\!\!-\!\!{\circ} O_l$, and (4) there exists a potentially undirected path between O_l and O_k that does not include O_j.*

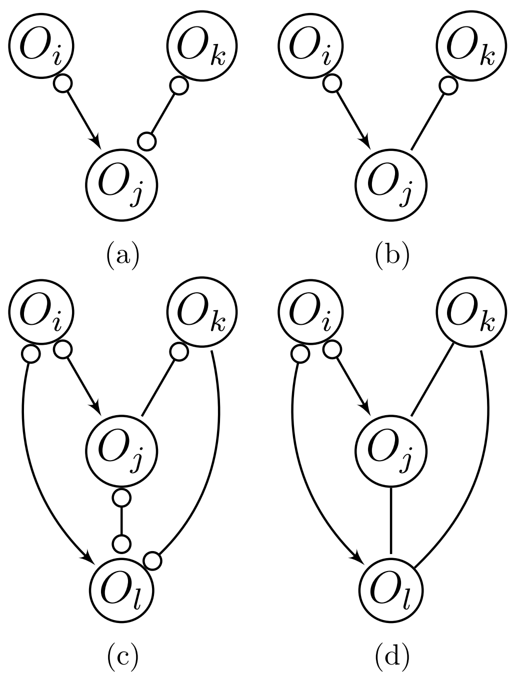


Figure 4: The first part of Rule 1 orients the graph (a) to (b). Note that the edge $O_i {\circ\!\!\rightarrow} O_j$ is potentially 2-triangulated w.r.t. O_k in (c), so the second part of Rule 1 cannot fire. However, if we additionally have $O_j - O_k$, then we can orient 3 endpoints with Rule 3 and obtain (d).

We now have three orientation rules that utilize the concept of potential 2-triangulation:

Lemma 11. *The following orientation rules are sound:*

#1. If we have $O_i {*\!\!\rightarrow} O_j {\circ\!\!-\!\!*}\, O_k$ with O_i and O_k non-adjacent, then orient $O_j {\circ\!\!-\!\!*}\, O_k$ as $O_j -\!\!* O_k$. Furthermore, if $O_i {*\!\!\rightarrow} O_j$ is not potentially 2-triangulated w.r.t. O_k, then orient $O_j {-\!\!\circ} O_k$ as $O_j \rightarrow O_k$.

#2. If we have $O_i -\!\!* O_j {\circ\!\!-\!\!*}\, O_k$ with O_i and O_k nonadjacent, and $O_j {\circ\!\!-\!\!*}\, O_k$ is not potentially 2-triangulated w.r.t. O_i, then orient $O_j {\circ\!\!-\!\!*}\, O_k$ as $O_j -\!\!* O_k$.

#3. Suppose that we have $O_i {*\!\!\rightarrow} O_j - O_k$ with O_i and O_k nonadjacent, and $O_i {*\!\!\rightarrow} O_j$ is potentially 2-triangulated w.r.t. O_k. If $O_i {*\!\!\rightarrow} O_j$ can be potentially 2-triangulated w.r.t. O_k using only one vertex O_l in the triangle involve $\{O_i, O_j, O_l\}$, then orient $O_i {*\!\!-\!\!\circ} O_l$ as $O_i {*\!\!\rightarrow} O_l$, $O_j {\circ\!\!-\!\!*} O_l$ as $O_j -\!\!* O_l$ and/or $O_j {*\!\!-\!\!\circ} O_l$ as $O_j {*\!\!-} O_l$. Next, if there exists only one potentially undirected path $\Pi_{O_l O_k}$ between O_l and O_k, then substitute all circle endpoints on $\Pi_{O_l O_k}$ with tail endpoints.

Proof. The following arguments correspond to their associated orientation rule:

#1. The first part follows from Lemma 8. The second part follows by the contrapositive of Lemma 10.

#2. Suppose that we have $O_i - O_j$ and $O_j \leftarrow\!\!* O_k$. But this would contradict Lemma 10. Suppose instead that we had $O_i \rightarrow O_j$ and $O_j \leftarrow\!\!* O_k$. But the arrowheads give rise to another contradiction because we know that $O_j \in \text{Sep}(O_i, O_k)$, so $O_j \in \text{Anc}(\{O_i, O_k\} \cup \boldsymbol{S})$ by Lemma 8.

#3. Follows directly from Lemma 10.

□

For example, $O_i {\circ\!\!\rightarrow} O_j$ is not potentially triangulated in Figure 4a (there are only three variables), so we may orient the endpoint $O_j {-\!\!\circ} O_k$ as $O_j \rightarrow O_k$ according to the first orientation rule. On the other hand, $O_i {\circ\!\!\rightarrow} O_j$ is potentially triangulated w.r.t O_k in Figure 4c, so we cannot orient the circle endpoint at O_k as an arrowhead. However, if we additionally have $O_j - O_k$, then we can apply Rule 3 to orient $O_j {\circ\!\!-\!\!\circ} O_l$ as $O_j - O_l$ and $O_l {\circ\!\!-} O_k$ as $O_l - O_k$ to ultimately obtain Figure 4d.

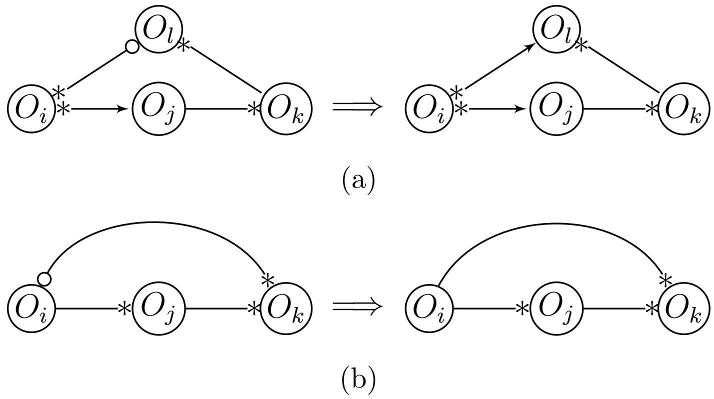


Figure 5: Examples of Rules 4 and 5 in (a) and (b), respectively.

9.6.2. Fourth & Fifth Orientation Rules

Lemma 12. *The following orientation rules are sound:*

- #4. *If $O_i {*}\!\!\rightarrow O_j \,{-}{*} O_k$, there exists a path $\Pi = \langle O_k, \cdots, O_i \rangle$ with at least $n \geq 3$ vertices such that we have $O_h \,{-}{*} O_{h+1}$ for all $1 \leq h \leq n-1$ except for only one index l where we have $O_l {\circ}\!{-}{*}\, O_{l+1}$, then orient $O_l {\circ}\!{-}{*}\, O_{l+1}$ as $O_l \leftarrow\!{*} O_{l+1}$.*

- #5. *If we have the sequence of vertices $\langle O_1, \ldots, O_n \rangle$ such that $O_i \,{-}{*} O_{i+1}$ with $1 \leq i \leq n-1$, and we have $O_1 {\circ}\!{-}{*}\, O_n$, then orient $O_1 {\circ}\!{-}{*}\, O_n$ as $O_1 \,{-}{*} O_n$.*

Proof. The following arguments correspond to their associated orientation rule:

- #4. Suppose for a contradiction that we had $O_l \,{-}{*} O_{l+1}$. But then O_j is an ancestor of $O_i \cup \boldsymbol{S}$ by transitivity of the tails.

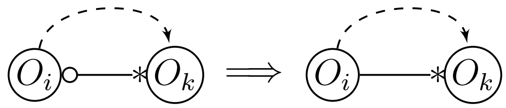
- #5. Follows by transitivity of the tail.

□

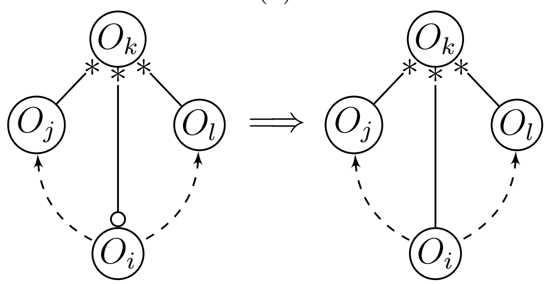
We provide examples of Rules 4 and 5 in Figures 5a and 5b, respectively.

9.6.3. Sixth & Seventh Orientation Rules

The CCI algorithm has 2 more orientation rules which require successive applications of the first orientation rule. We first require the following definition:



(a)



(b)

Figure 6: Rules 6 and 7 in (a) and (b), respectively. Dotted lines indicate non-potentially 2-triangulated paths.

Definition 7. *(Non-Potentially 2-Triangulated Path) A path* $\Pi = \langle O_1, \ldots, O_n \rangle$ *is said to be non-potentially 2-triangulated if the following conditions hold:*

1. *If* $n \geq 3$*, then the vertices* O_{i-1} *and* O_{i+1} *are non-adjacent for every* $2 \leq i \leq n-1$ *(i.e., every consecutive triple is non-adjacent), and* $O_{i-1} \ast\!\!-\!\!\ast O_i \ast\!\!-\!\!\ast O_{i+1}$ *is a non-v-structure for every* $2 \leq i \leq n-1$.

2. *If* $n \geq 4$*, then the vertices* $O_i \ast\!\!\leftrightarrow\!\!\ast O_{i+1}$ *are not potentially 2-triangulated w.r.t.* O_{i+2} *for every* $1 \leq i \leq n-3$.

The following orientation rules utilize the above definition:

Lemma 13. *The following orientation rules are sound:*

\#6. *If we have* $O_k \ast\!\!-\!\!\circ O_i$*, there exists a non-potentially 2-triangulated path* $\Pi = \langle O_i, O_j, O_l, \ldots O_k \rangle$ *such that* $O_k \ast\!\!-\!\!\circ O_i$ *is not potentially 2-triangulated w.r.t.* O_j*, and* $O_j \ast\!\!\leftrightarrow\!\!\ast O_i \ast\!\!\leftrightarrow\!\!\ast O_k$ *is a non-v-structure, then orient* $O_k \ast\!\!-\!\!\circ O_i$ *as* $O_k \ast\!\!- O_i$ *(Figure 6a).*

\#7. *Suppose we have* $O_i \circ\!\!\rightarrow\!\!\ast O_k$*,* $O_j \rightarrow\!\!\ast O_k \ast\!\!- O_l$*, a non-potentially 2-triangulated path* Π_1 *from* O_i *to* O_j*, and a non-potentially 2-triangulated path* Π_2 *from* O_i *to* O_l*. Let* O_m *be a vertex adjacent to* O_i *on* Π_1 *(*O_m *could be* O_j*), and let* O_n *be the vertex adjacent to* O_i *on* Π_2 *(*O_n *could be* O_l*).*

If further $O_m \ast\!\!-\!\!\ast\, O_i \ast\!\!-\!\!\ast\, O_n$ is a non-v-structure and $O_i \circ\!\!-\!\!\ast\, O_k$ is not potentially 2-triangulated w.r.t. both O_n and O_m, then orient $O_i \circ\!\!-\!\!\ast\, O_k$ as $O_i - \!\!\ast O_k$ (Figure 6b).

Proof. The following arguments apply to their corresponding orientation rules:

#6. Suppose for a contradiction that we have $O_i \leftarrow\!\!\ast O_k$. Then we can iteratively apply the first orientation rule on Π until the transitivity of the added tails contradicts the arrowhead at O_i.

#7. Suppose for a contradiction that we have $O_i \leftarrow\!\!\ast O_k$. Then we iteratively apply the first orientation rule along Π_1 or Π_2 (or both). In any case, $O_i \in \text{Anc}(O_k \cup \boldsymbol{S})$ by transitivity of the added tails which contradicts the arrowhead at O_i.

□

10. Experiments

We now report the empirical results.

10.1. Synthetic Data

We generated 1000 random Gaussian directed cyclic graphs (directed graphs with at least one cycle) with an expected neighborhood size $\mathbb{E}(N) = 2$ and $p = 20$ vertices using the following procedure. First, we generated a random adjacency matrix B with independent realizations of Bernoulli($\mathbb{E}(N)/(2p - 2)$) random variables in the off-diagonal entries. We then replaced the non-zero entries in B with independent realizations of Uniform($[-1, -0.1] \cup [0.1, 1]$) random variables. We can interpret a nonzero entry B_{ij} as an edge from X_i to X_j with coefficient B_{ij} in the following linear model:

$$X_i = \sum_{r=1}^{p} B_{ir} X_r + \varepsilon_i, \tag{3}$$

for $i = 1, \ldots, p$ where $\varepsilon_1, ..., \varepsilon_p$ are mutually independent $\mathcal{N}(0, 1)$ random variables. The variables $X_1, \ldots,\, X_p$ then have a multivariate Gaussian distribution with mean vector 0 and covariance matrix $\Sigma = (\mathbb{I} - B)^{-1}(\mathbb{I} - B)^{-T}$, where $\mathbb{I}$ is the $p \times p$ identity matrix.

We similarly generated 1000 random Gaussian DAGs with the same parameters but created each random adjacency matrix B with independent realizations of Bernoulli($\mathbb{E}(N)/(p-1)$) random variables in the lower triangular and off-diagonal entries (Colombo et al., 2012).

We introduced latent and selection variables into each DCG and DAG as follows. We first randomly selected a set of 0-3 latent common causes $\boldsymbol{L}$ without replacement. We then selected a set of 0-3 selection variables $\boldsymbol{S}$ without replacement from the set of vertices $\boldsymbol{X} \setminus \boldsymbol{L}$ with at least two parents.

We ultimately created datasets with sample sizes of 500, 1000, 5000, 10000, 50000 and 100000 for each of the 1000 DCGs and each of the 1000 DAGs. We therefore generated a total of $1000 \times 6 \times 2 = 12000$ datasets.

10.2. Algorithms

We compared the following four CB algorithms. We also list each algorithm's assumptions:

1. CCI: acyclic or cyclic with linear SEM-IE
2. FCI: acyclic
3. RFCI: acyclic
4. CCD: acyclic or cyclic with linear SEM-IE, no latent variables[2]

All algorithms additionally assume d-separation faithfulness. Only CCI remains sound under CLS. We ran all algorithms using Fisher's z-test with α set to 1E-2 for sample sizes 500 and 1000, 1E-3 for 5000 and 10000, and 1E-4 otherwise. Recall that we require decreasing p-values with increasing sample sizes in order to ensure consistency (Kalisch and Bühlmann, 2007, Colombo et al., 2012).

10.3. Metrics

We assessed the algorithms using the structural Hamming distance (SHD) (Tsamardinos et al., 2006) to the corrected oracle graphs. We construct the corrected oracle graph as follows. First, we run an algorithm with a CI oracle to obtain the oracle graph. Then, we replace any incorrect arrowhead with a

[2]CCD cannot handle selection bias as proposed in (Richardson and Spirtes, 1999), but the algorithm may be able to if we modify the proofs.

tail and vice versa. For example, if we have $O_i {*\!\!\rightarrow} O_j$ in the oracle graph, but $O_j \in \text{Anc}(O_i \cup \boldsymbol{S})$, then we replace $O_i {*\!\!\rightarrow} O_j$ with $O_i {*\!\!-} O_j$. An algorithm which is sound will always output an oracle graph which does not require correction, if the algorithm's assumptions are satisfied.

10.4. Cyclic Case

We first compared CCI to CCD in the cyclic case. Here, we hope that CCI will outperform CCD, since CCD cannot handle latent common causes. We have summarized the results in Figure 7. We unexpectedly found that CCD outperformed CCI by a significant margin across the largest four of the 6 sample sizes (Figure 7a; min t = -2.94, p = 3.34E-3). CCD also completed in a much shorter time frame than CCI (Figure 7b). Recall however that CCI makes more long range inferences than CCD by applying multiple orientation rules. We therefore also analyzed the performance of CCI with the orientation rules removed, denoted as CCI minus OR (CCI–OR); this comparison pits CCI against CCD on more fair grounds, because CCD does not have orientation rules. Here, we found that CCI–OR outperformed CCD across all sample sizes (max t = -26.64, p<2.2E-16; Figure 7c). We also added the orientation rules of CCI to CCD, which we call CCD plus OR (CCD+OR). CCI again outperformed CCD+OR across all sample sizes (max t = -13.13, p<2.2E-16; Figure 7d). We conclude that CCI outperforms CCD once we account for the orientation rules.

10.5. Acyclic Case

We next compared CCI to FCI and RFCI in the acyclic case. Here, we expect CCI to perform worse than FCI and RFCI on average, because CCI does not assume acyclicity. However, we hope that CCI will not underperform by a large margin.

We have summarized the SHD results in Figure 8a and the timing results in Figure 8b. CCI recovered acyclic causal graphs less accurately than FCI by a significant margin with sample sizes $\leq$ 10000 (min t = 4.23, p = 2.59E-5). We found no statistically significant difference at larger sample sizes ($p > 0.05/6$). CCI was also outperformed by RFCI with sample sizes between 1000 to 10000 (min t = 2.78, p = 5.50E-3). The effect sizes were nonetheless very small; CCI had mean SHDs at most 0.91 points greater than FCI and RFCI across all sample sizes. We conclude that CCI underperforms FCI and RFCI in the acyclic cause but only by a negligible margin.

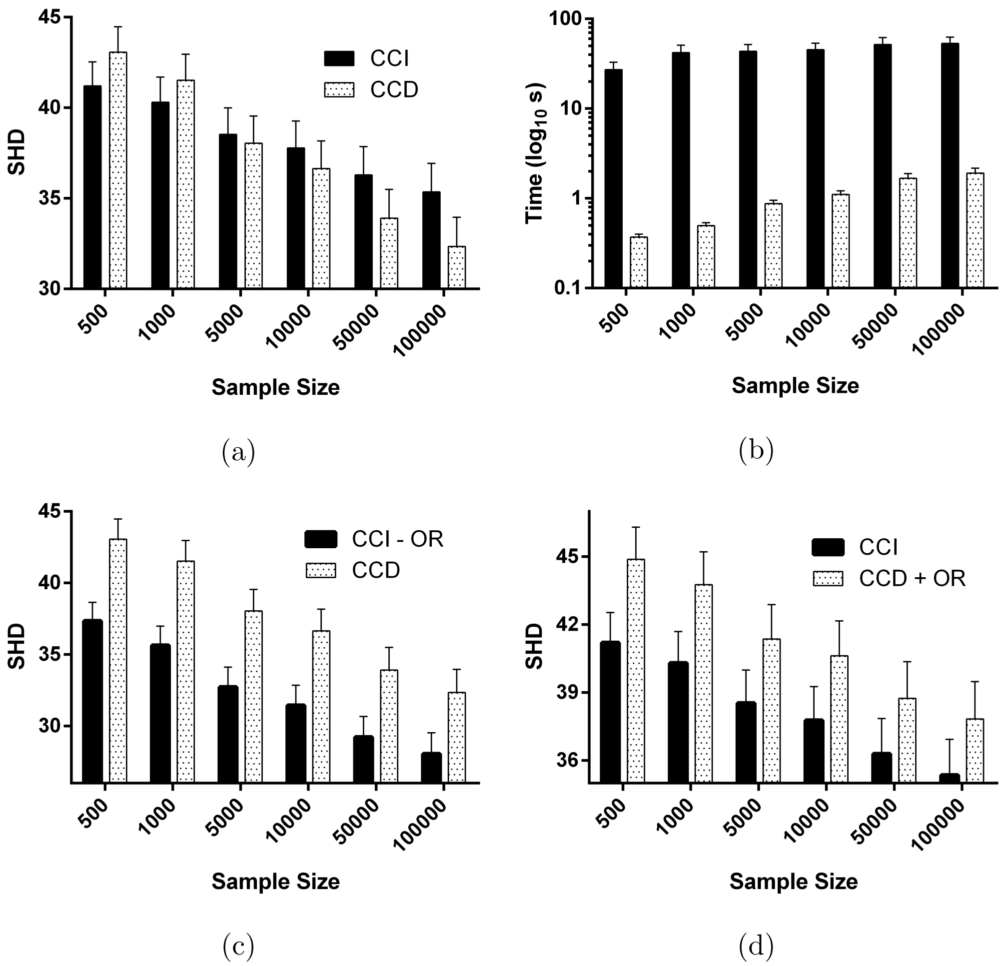


Figure 7: CCI versus CCD on recovering cyclic graphs with latent variables and selection bias. Smaller SHD is better; error bars always denote 95% confidence intervals of the mean. (a) CCD outperforms CCI on sample sizes >1000. (b) CCD also has shorter running times than CCI. However, CCI–OR outperforms CCD in (c). CCI similarly outperforms CCD+OR in (d).

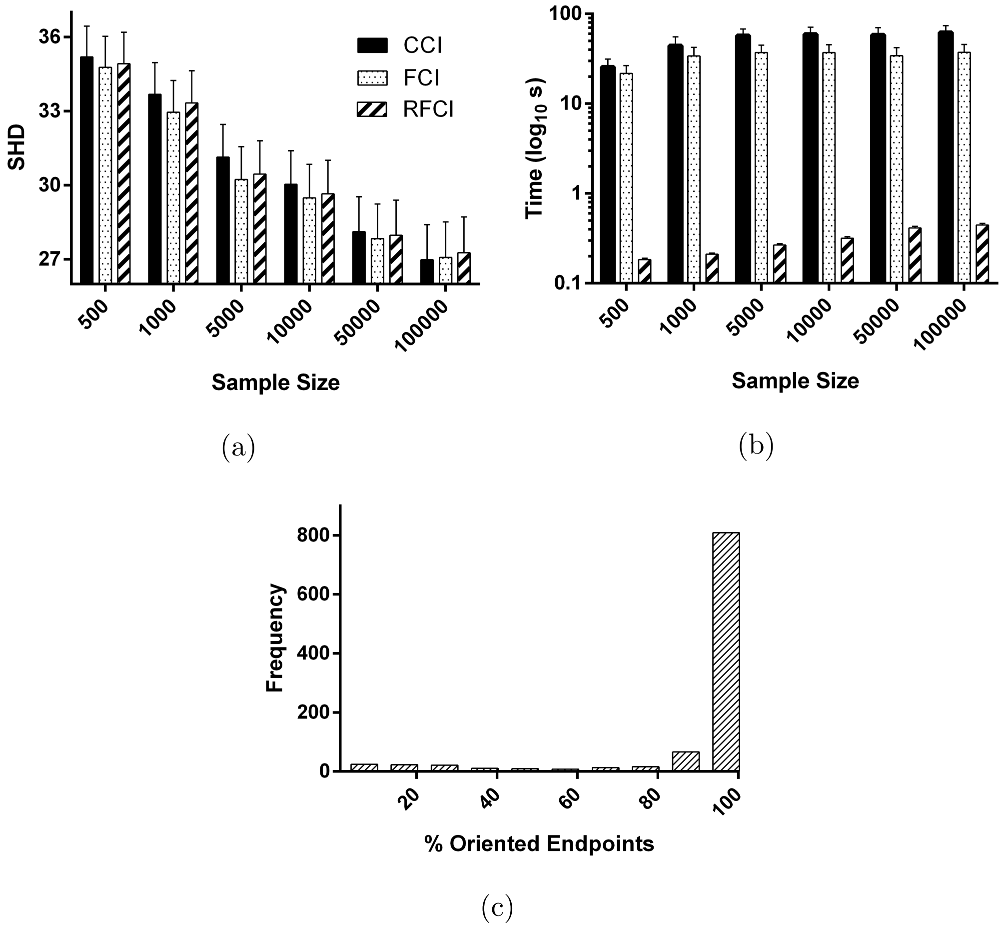
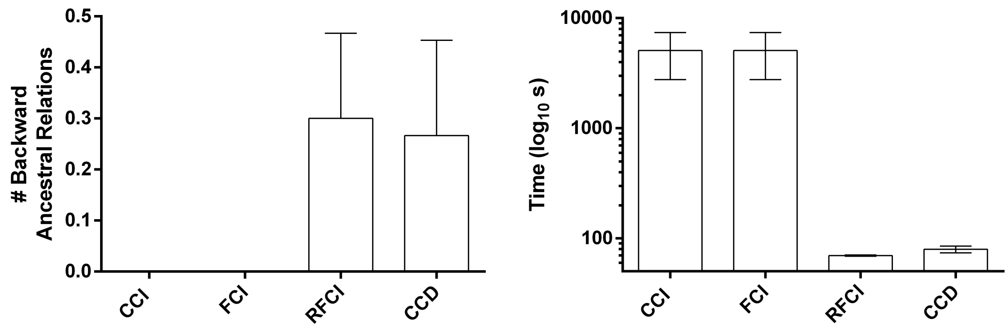


Figure 8: CCI versus FCI and RFCI in recovering acyclic graphs with latent variables and selection bias. (a) FCI and RFCI outperform CCI by a slight margin on sample sizes ≤ 10000. (b) CCI takes slightly longer to complete than FCI. (c) CCI orients the majority of endpoints oriented by FCI.



(a) (b)

Figure 9: Results of algorithms on real data. (a) FCI and CCI perform the best because they do not discover any backward ancestral relations. (b) However, both of these algorithms take much longer to complete than the others.

We also sought to answer the follow question: how many edges does CCI orient compared to FCI in the acyclic case? It is impossible for CCI to orient 100% of the endpoints oriented by FCI, because FCI assumes acyclicity whereas CCI does not. We would however ideally like CCI to orient most of the endpoints oriented by FCI in the acyclic case.

In order to answer the question, we ran the CCI and FCI algorithms on 1000 random DAGs with a CI oracle. CCI oriented 89.98% (SE: 0.74%) of the endpoints oriented by FCI on average. Moreover, the histogram of percentages had a heavy left skew (Figure 8c), so the median of CCI vs FCI was 100%. We conclude that CCI orients the majority of endpoints oriented by FCI in the acyclic case.

10.6. Real Data

We finally ran the same algorithms using the nonparametric CI test called RCoT (Strobl et al., 2017) at $\alpha = 0.01$ on a publicly available longitudinal dataset from the Framingham Heart Study (Mahmood et al., 2014), where scientists measured a variety of clinical variables related to cardiac health. The dataset contains 28 variables, 3 waves and 2008 samples after performing list-wise deletion. All but 2 variables were measured in all 3 waves.

Note that we do not have access to a gold standard solution set in this case. We can however develop an approximate solution set by utilizing time information because we cannot have ancestral relations directed backwards in time. A variable in wave b thus cannot be an ancestor of a variable in wave $a < b$. In terms of a partially oriented MAAG, this means that any edge between wave a and wave b with both a tail and an arrowhead at a vertex in wave b is incorrect. We thus evaluated the algorithms using the average number of incorrect ancestral relations directed backwards in time.

We have summarized the results in Figure 9a averaged over 30 bootstrapped datasets. Notice that FCI and CCI outperform CCD by a significant margin because FCI and CCI do not make any errors (t=-2.80, p=8.90E-3). Moreover, RFCI performs the worst, highlighting the price one must pay for speed (Figure 9b). We conclude that accounting for latent variables allows for more accurate causal discovery on this dataset.

11. Conclusion

This report introduced an algorithm called CCI for performing causal discovery with CLS provided that we can represent the cyclic causal process as a linear SEM-IE. As far as I am aware, CCI is the most general CB algorithm proposed to date. The experimental results in the previous section highlight the superior or comparable performance of CCI when compared to previous algorithms that do not allow selection bias, latent variables and/or cycles.

References

References

D. Colombo, M. Maathius, M. Kalisch, and T. Richardson. Learning high-dimensional directed acyclic graphs with latent and selection variables. *Annals of Statistics*, 40(1):294–321, Apr. 2012. doi: 10.1214/11-AOS940. URL http://projecteuclid.org/euclid.aos/1333567191.

P. Dagum, A. Galper, E. Horvitz, and A. Seiver. Uncertain reasoning and forecasting. *International Journal of Forecasting*, 11:73–87, 1995.

R. J. Evans. Graphs for margins of bayesian networks. *Scandinavian Journal of Statistics*, 43(3):625–648, 2016.

F. M. Fisher. A correspondence principle for simultaneous equation models. *Econometrica*, 38(1):73–92, 1970. URL https://EconPapers.repec.org/RePEc:ecm:emetrp:v:38:y:1970:i:1:p:73-92.

A. Hyttinen, P. O. Hoyer, F. Eberhardt, and M. Järvisalo. Discovering cyclic causal models with latent variables: A general sat-based procedure. In *Proceedings of the Twenty-Ninth Conference on Uncertainty in Artificial Intelligence, UAI 2013, Bellevue, WA, USA, August 11-15*, 2013. URL https://dslpitt.org/uai/displayArticleDetails.jsp?mmnu=1&smnu=2&article_id=2391&proceeding_id=29.

M. Kalisch and P. Bühlmann. Estimating high-dimensional directed acyclic graphs with the pc-algorithm. *J. Mach. Learn. Res.*, 8:613–636, May 2007. ISSN 1532-4435. URL http://dl.acm.org/citation.cfm?id=1248659.1248681.

S. L. Lauritzen and T. S. Richardson. Chain graph models and their causal interpretations. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 64(3):321–348, 2002. doi: 10.1111/1467-9868.00340. URL https://rss.onlinelibrary.wiley.com/doi/abs/10.1111/1467-9868.00340.

S. L. Lauritzen, A. P. Dawid, B. N. Larsen, and H. G. Leimer. Independence Properties of Directed Markov Fields. *Networks*, 20(5):491–505, Aug. 1990. doi: 10.1002/net.3230200503. URL http://dx.doi.org/10.1002/net.3230200503.

S. S. Mahmood, D. Levy, R. S. Vasan, and T. J. Wang. The framingham heart study and the epidemiology of cardiovascular disease: a historical perspective. *The Lancet*, 383(9921):999 – 1008, 2014. ISSN 0140-6736. doi: http://doi.org/10.1016/S0140-6736(13)61752-3. URL http://www.sciencedirect.com/science/article/pii/S0140673613617523.

C. Meek. Causal inference and causal explanation with background knowledge. In *Proceedings of the Eleventh Conference on Uncertainty in Artificial Intelligence*, UAI'95, pages 403–410, San Francisco, CA, USA, 1995. Morgan Kaufmann Publishers Inc. ISBN 1-55860-385-9. URL http://dl.acm.org/citation.cfm?id=2074158.2074204.

J. M. Mooij and T. Heskes. Cyclic causal discovery from continuous equilibrium data. In A. Nicholson and P. Smyth, editors, *Proceedings of the 29th Annual Conference on Uncertainty in Artificial Intelligence (UAI-13)*, pages 431–439. AUAI Press, 2013. URL http://auai.org/uai2013/prints/papers/23.pdf.

J. Pearl. *Causality: Models, Reasoning and Inference*. Cambridge University Press, New York, NY, USA, 2nd edition, 2009. ISBN 052189560X, 9780521895606.

T. Richardson. Properties of cyclic graphical models. Master's thesis, Carnegie Mellon University, 1994.

T. Richardson. A discovery algorithm for directed cyclic graphs. In *Proceedings of the Twelfth International Conference on Uncertainty in Artificial Intelligence*, UAI'96, pages 454–461, San Francisco, CA, USA, 1996. Morgan Kaufmann Publishers Inc. ISBN 1-55860-412-X. URL http://dl.acm.org/citation.cfm?id=2074284.2074338.

T. Richardson and P. Spirtes. Automated causal discovery under linear feedback. In *Computation, Causation, and Discovery*, pages 253–302. AAAI Press, Menlo Park, CA, 1999.

T. Richardson and P. Spirtes. Ancestral graph markov models. *Annals of Statistics*, 30:2002, 2000.

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

P. Spirtes. Directed cyclic graphical representations of feedback models. In *Proceedings of the Eleventh Conference on Uncertainty in Artificial Intelligence*, UAI'95, pages 491–498, San Francisco, CA, USA, 1995. Morgan Kaufmann Publishers Inc. ISBN 1-55860-385-9. URL http://dl.acm.org/citation.cfm?id=2074158.2074214.

P. Spirtes and T. Richardson. A polynomial time algorithm for determining dag equivalence in the presence of latent variables and selection bias. In *Proceedings of the 6th International Workshop on Artificial Intelligence and Statistics*, 1996.

P. Spirtes, C. Meek, and T. Richardson. Causal inference in the presence of latent variables and selection bias. In *Proceedings of the Eleventh Conference on Uncertainty in Artificial Intelligence*, UAI'95, pages 499–506, San Francisco, CA, USA, 1995. Morgan Kaufmann Publishers Inc. ISBN 1-55860-385-9. URL http://dl.acm.org/citation.cfm?id=2074158.2074215.

P. Spirtes, C. Meek, and T. Richardson. An algorithm for causal inference in the presence of latent variables and selection bias. In *Computation, Causation, and Discovery*, pages 211–252. AAAI Press, Menlo Park, CA, 1999.

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

E. Strobl. Causal discovery under non-stationary feedback. July 2017. URL http://d-scholarship.pitt.edu/32790/.

E. V. Strobl, K. Zhang, and S. Visweswaran. Approximate Kernel-based Conditional Independence Tests for Fast Non-Parametric Causal Discovery. 2017. URL http://arxiv.org/abs/1702.03877.

I. Tsamardinos, L. E. Brown, and C. F. Aliferis. The max-min hill-climbing bayesian network structure learning algorithm. *Mach. Learn.*, 65(1):31–78, Oct. 2006. ISSN 0885-6125. doi: 10.1007/s10994-006-6889-7. URL http://dx.doi.org/10.1007/s10994-006-6889-7.

J. Zhang. On the completeness of orientation rules for causal discovery in the presence of latent confounders and selection bias. *Artif. Intell.*, 172(16-17): 1873–1896, Nov. 2008. ISSN 0004-3702. doi: 10.1016/j.artint.2008.08.001. URL http://dx.doi.org/10.1016/j.artint.2008.08.001.

12. Appendix: Algorithms

We will utilize ideas developed for the PC, FCI, RFCI and CCD algorithms in order to construct CCI. We therefore briefly review PC, FCI, RFCI and CCD in the next four subsections.

12.1. The PC Algorithm

The PC algorithm considers the following problem: assume that $\mathbb{P}$ is d-separation faithful to an unknown DAG $\mathbb{G}$. Then, given oracle information about the conditional independencies between any pair of variables X_i and X_j given any $\boldsymbol{W} \subseteq \boldsymbol{X} \setminus \{X_i, X_j\}$ in $\mathbb{P}$, reconstruct as much of the underlying DAG as possible. The PC algorithm ultimately accomplishes this goal by reconstructing the DAG up to its *Markov equivalence class*, or the set of

DAGs with the same conditional dependence and independence relations between variables in $\boldsymbol{X}$ (Spirtes et al., 2000, Meek, 1995).

The PC algorithm represents the Markov equivalence class of DAGs using a *completed partially directed acyclic graph* (CPDAG). A *partially directed acyclic graph* (PDAG) is a graph with both directed and undirected edges. A PDAG is *completed* when the following conditions hold: (1) every directed edge also exists in every DAG belonging to the Markov equivalence class of the DAG, and (2) there exists a DAG with $X_i \rightarrow X_j$ and a DAG with $X_i \leftarrow X_j$ in the Markov equivalence class for every undirected edge $X_i - X_j$. Each edge in the CPDAG also has the following interpretation:

(i) An edge (directed or undirected) is absent between two vertices X_i and X_j if and only if there exists some $\boldsymbol{W} \subseteq \boldsymbol{X} \setminus \{X_i, X_j\}$ such that $X_i \perp\!\!\!\perp X_j | \boldsymbol{W}$.

(ii) If there exists a directed edge from X_i to X_j, then $X_i \in \text{Pa}(X_j)$.

The PC algorithm learns the CPDAG through a three step procedure. First, the algorithm initializes a fully connected undirected graph and then determines the presence or absence of each undirected edge using the following fact: under d-separation faithfulness, X_i and X_j are non-adjacent if and only if X_i and X_j are conditionally independent given some subset of $\text{Pa}(X_i) \setminus X_j$ or some subset of $\text{Pa}(X_j) \setminus X_i$. Note that PC cannot differentiate between the parents and children of a vertex from its neighbors using an undirected graph. Thus, PC tests whether X_i and X_j are conditionally independent given all subsets of $\text{Adj}(X_i) \setminus X_j$ and all subsets of $\text{Adj}(X_j) \setminus X_i$, where $\text{Adj}(X_i)$ denotes the vertices adjacent to X_i in $\mathbb{G}$ (a superset of $\text{Pa}(X_i)$), in order to determine the final adjacencies; we refer to this sub-procedure of PC as *skeleton discovery* and list the pseudocode in Algorithm 3. The PC algorithm therefore removes the edge between X_i and X_j during skeleton discovery if such a conditional independence is found.

Step 2 of the PC algorithm orients unshielded triples to v-structures $X_i \rightarrow X_j \leftarrow X_k$ if X_j is not in the set of variables which rendered X_i and X_k conditionally independent in the skeleton discovery phase of the algorithm. The final step of the PC algorithm involves the repetitive application of three orientation rules to replace as many tails as possible with arrowheads (Meek, 1995).

Data: CI oracle
Result: $\widehat{\mathbb{G}}$, Sep, $\mathcal{M}$

Form a complete graph $\widehat{\mathbb{G}}$ on $\boldsymbol{O}$ with vertices $\circ\!\!-\!\!\circ$
$l \leftarrow -1$
repeat
  Let $l = l + 1$
  repeat
    forall *vertices in* $\widehat{\mathbb{G}}$ **do**
      Compute $\mathrm{Adj}(O_i)$
    end
    Select a new ordered pair of vertices (O_i, O_j) that are adjacent in $\widehat{\mathbb{G}}$ and satisfy $|\mathrm{Adj}(O_i) \setminus O_j| \geq l$
    repeat
      Choose a new set $\boldsymbol{W} \subseteq \mathrm{Adj}(O_i) \setminus O_j$ with $|\boldsymbol{W}| = l$
      if $O_i \perp\!\!\!\perp O_j | \boldsymbol{W} \cup \boldsymbol{S}$ **then**
        Delete the edge $O_i \circ\!\!-\!\!\circ O_j$ from $\widehat{\mathbb{G}}$
        Let $\mathrm{Sep}(O_i, O_j) = \mathrm{Sep}(O_j, O_i) = \boldsymbol{W}$
      end
    until O_i *and* O_j *are no longer adjacent in* $\widehat{\mathbb{G}}$ *or all* $\boldsymbol{W} \subseteq \mathrm{Adj}(O_i) \setminus O_j$ *with* $|\boldsymbol{W}| = l$ *have been considered*;
  until *all ordered pairs of adjacent vertices* (O_i, O_j) *in* $\widehat{\mathbb{G}}$ *with* $|\mathrm{Adj}(O_i) \setminus O_j| \geq l$ *have been considered*;
until *all pairs of adjacent vertices* (O_i, O_j) *in* $\widehat{\mathbb{G}}$ *satisfy* $|\mathrm{Adj}(O_i) \setminus O_j| \leq l$;
Form a list $\mathcal{M}$ of all unshielded triples $\langle O_k, \cdot, O_m \rangle$ (i.e., the middle vertex is left unspecified) in $\widehat{\mathbb{G}}$ with $k < m$

Algorithm 3: PC's skeleton discovery procedure

12.2. The FCI Algorithm

The FCI algorithm considers the following problem: assume that the distribution of $\boldsymbol{X} = \boldsymbol{O} \cup \boldsymbol{L} \cup \boldsymbol{S}$ is d-separation faithful to an unknown DAG. Then, given oracle information about the conditional independencies between any pair of variables O_i and O_j given any $\boldsymbol{W} \subseteq \boldsymbol{O} \setminus \{O_i, O_j\}$ as well as $\boldsymbol{S}$, reconstruct as much information about the underlying DAG as possible (Spirtes et al., 2000). The FCI algorithm ultimately accomplishes this goal by reconstructing a MAG up to its Markov equivalence class, or the set of MAGs with the same conditional dependence and independence relations between variables in $\boldsymbol{O}$ given $\boldsymbol{S}$ (Zhang, 2008).

The FCI algorithm represents the Markov equivalence class of MAGs using a *completed partial maximal ancestral graph* (CPMAG).[3] A *partial maximal ancestral graph* (PMAG) is nothing more than a MAG with possibly some circle endpoints. A PMAG is *completed* (and hence a CPMAG) when the following conditions hold: (1) every tail and arrowhead also exists in every MAG belonging to the Markov equivalence class of the MAG, and (2) there exists a MAG with a tail and a MAG with an arrowhead in the Markov equivalence class for every circle endpoint. Each edge in the CPMAG also has the following interpretations:

(i) An edge is absent between two vertices O_i and O_j if and only if there exists some $\boldsymbol{W} \subseteq \boldsymbol{O} \setminus \{O_i, O_j\}$ such that $O_i \perp\!\!\!\perp O_j | \boldsymbol{W} \cup \boldsymbol{S}$. That is, an edge is absent if and only if there does not exist an inducing path between O_i and O_j.

(ii) If an edge between O_i and O_j has an arrowhead at O_j, then $O_j \notin \text{Anc}(O_i \cup \boldsymbol{S})$.

(iii) If an edge between O_i and O_j has a tail at O_j, then $O_j \in \text{Anc}(O_i \cup \boldsymbol{S})$.

The FCI algorithm learns the CPMAG through a three step procedure involving skeleton discovery, v-structure orientation and orientation rule application. The skeleton discovery procedure involves running PC's skeleton discovery procedure, orienting v-structures using Algorithm 4, and then re-performing skeleton discovery using possible d-separating sets (see Definition

[3]The CPMAG is also known as a partial ancestral graph (PAG). However, we will use the term CPMAG in order to mimic the use of the term CPDAG.

4) constructed after the v-structure discovery process. FCI then orients v-structures again using Algorithm 4 on the final skeleton. The third step of FCI involves the repetitive application of 10 orientation rules (Zhang, 2008).

Data: $\widehat{\mathbb{G}}$, Sep, $\mathcal{M}$
Result: $\widehat{\mathbb{G}}$

```
forall elements ⟨O_i, O_j, O_k⟩ in M do
    if O_j ∉ Sep(O_i, O_k) then
        Orient O_i *–o O_j o–* O_k as O_i *→ O_j ←* O_k in Ĝ
    end
end
```

Algorithm 4: Orienting v-structures

12.3. The RFCI Algorithm

Discovering inducing paths can require large possible d-separating sets, so the FCI algorithm often takes too long to complete. The RFCI algorithm (Colombo et al., 2012) resolves this problem by recovering a graph where the presence and absence of an edge have the following modified interpretations:

(i) The absence of an edge between two vertices O_i and O_j implies that there exists some $\boldsymbol{W} \subseteq \boldsymbol{O} \setminus \{O_i, O_j\}$ such that $O_i \perp\!\!\!\perp O_j | \boldsymbol{W} \cup \boldsymbol{S}$.

(ii) The presence of an edge between two vertices O_i and O_j implies that $O_i \not\perp\!\!\!\perp O_j | \boldsymbol{W} \cup \boldsymbol{S}$ for all $\boldsymbol{W} \subseteq \text{Adj}(O_i) \setminus O_j$ and for all $\boldsymbol{W} \subseteq \text{Adj}(O_j) \setminus O_i$. Here $\text{Adj}(O_i)$ denotes the set of vertices adjacent to O_i in RFCI's graph.

We encourage the reader to compare these edge interpretations to the edge interpretations of FCI's CPMAG.

The RFCI algorithm learns its graph (not necessarily a CPMAG) also through a three step process. The algorithm performs skeleton discovery using PC skeleton discovery procedure (Algorithm 3). RFCI then orients v-structures using Algorithm 6. Notice that Algorithm 6 requires more steps than Algorithm 4 used in FCI because an inducing path may not exist between any two adjacent vertices after only running PC's skeleton discovery procedure. RFCI must therefore check for additional conditional dependence relations in order to infer the non-ancestral relations. RFCI finally repetitively applies the 10 orientation rules of FCI in the last step with some

Data: $\widehat{\mathbb{G}}$, Sep
Result: $\widehat{\mathbb{G}}$, Sep, $\mathcal{M}$

forall *vertices* O_i *in* $\widehat{\mathbb{G}}$ **do**
    Compute PD-SEP(O_i)
    forall *vertices* $O_j \in \mathrm{Adj}(O_i)$ **do**
        Let $l = -1$
        repeat
            Let $l = l + 1$
            repeat
                Choose a (new) set $\boldsymbol{W} \subseteq \text{PD-SEP}(O_i) \setminus O_j$ with $|\boldsymbol{W}| = l$
                if $O_i \perp\!\!\!\perp O_j | \boldsymbol{W} \cup \boldsymbol{S}$ **then**
                    Delete edge $O_i \ast\!\!-\!\!\ast O_j$ in $\widehat{\mathbb{G}}$
                    Let $\mathrm{Sep}(O_i, O_j) = \mathrm{Sep}(O_j, O_i) = \boldsymbol{W}$
                end
            until O_i *and* O_j *are no longer adjacent in* $\widehat{\mathbb{G}}$ *or all* $\boldsymbol{W} \subseteq \text{PD-SEP}(O_i) \setminus O_j$ *with* $|\boldsymbol{W}| = l$ *have been considered*;
        until O_i *and* O_j *are no longer adjacent in* $\widehat{\mathbb{G}}$ *or* $|\text{PD-SEP}(O_i) \setminus O_j| < l$;
    end
end
Reorient all edges in $\widehat{\mathbb{G}}$ as $\circ\!\!-\!\!\circ$
Form a list $\mathcal{M}$ of all unshielded triples $\langle O_k, \cdot, O_m \rangle$ in $\widehat{\mathbb{G}}$ with $k < m$

Algorithm 5: Obtaining the final skeleton in the FCI algorithm modifications to the fourth orientation rule (see (Colombo et al., 2012) for further details).

12.4. The CCD Algorithm

The CCD algorithm considers the following problem: assume that $\mathbb{P}$ is d-separation faithful to an unknown *possibly cyclic* directed graph $\mathbb{G}$. Then, given oracle information about the conditional independencies between any pair of variables X_i and X_j given any $\boldsymbol{W} \subseteq \boldsymbol{X} \setminus \{X_i, X_j\}$ in $\mathbb{P}$, output a partial oriented MAAG (see Section 6 for a definition) of the underlying directed graph (Richardson, 1996, Richardson and Spirtes, 1999). Notice that CCD does not consider latent or selection variables.

The CCD algorithm involves six steps. The first step corresponds to skeleton discovery and is analogous to PC's procedure (Algorithm 3). CCD also orients v-structures like PC. The algorithm then however checks for certain long-range d-separation relations in its third step in order to infer additional non-ancestral relations. The fourth step proceeds similarly (but not exactly) to CCI's Step 4 by discovering additional non-minimal d-separating sets. Finally, the fifth and sixth steps of CCD utilize the aforementioned non-minimal d-separating sets in order to orient additional endpoints. Note that CCD does not apply orientation rules.

13. Appendix: Proofs

In the arguments to follow, I will always consider a directed graph (cyclic or acyclic) with vertices $\boldsymbol{X} = \boldsymbol{O} \cup \boldsymbol{L} \cup \boldsymbol{S}$, where $\boldsymbol{O}, \boldsymbol{L}$ and $\boldsymbol{S}$ are disjoint sets.

13.1. Utility Lemmas

Lemma 14. *(Lemma 2.5 in Colombo et al., 2011) Suppose that X_i and X_j are not in $\boldsymbol{W} \subseteq \boldsymbol{X} \setminus \{X_i, X_j\}$, there is a sequence σ of distinct vertices in $\boldsymbol{X}$ from X_i to X_j, and there is a set $\mathcal{T}$ of paths such that:*

1. *for each pair of adjacent vertices X_v and X_w in σ, there is a unique path in $\mathcal{T}$ that d-connects X_v and X_w given $\boldsymbol{W}$;*

2. *if a vertex X_q in σ is in $\boldsymbol{W}$, then the paths in $\mathcal{T}$ that contain X_q as an endpoint collide at X_q;*

3. *if for three vertices X_v, X_w and X_q occurring in that order in σ, the d-connecting paths in $\mathcal{T}$ between X_v and X_w, and between X_w and X_q collide at X_w, then X_w has a descendant in $\boldsymbol{W}$.*

Data: Initial skeleton $\widehat{\mathbb{G}}$, Sep, $\mathcal{M}$
Result: $\widehat{\mathbb{G}}$, Sep

Let $\mathcal{L}$ denote an empty list
while $\mathcal{M}$ *is non-empty* **do**
 Choose an unshielded triple $\langle O_i, O_j, O_k \rangle$ from $\mathcal{M}$
 if $O_i \perp\!\!\!\perp O_j | \mathrm{Sep}(O_i, O_k) \cup \boldsymbol{S}$ *and* $O_j \perp\!\!\!\perp O_k | \mathrm{Sep}(O_i, O_k) \cup \boldsymbol{S}$ **then**
  Add $\langle O_i, O_j, O_k \rangle$ to $\mathcal{L}$
 end
 else
  for $r \in \{i, k\}$ **do**
   if $O_r \perp\!\!\!\perp O_j | (\mathrm{Sep}(O_i, O_k) \setminus O_j) \cup \boldsymbol{S}$ **then**
    Find a minimal separating set $\boldsymbol{W} \subseteq \mathrm{Sep}(O_i, O_k)$ for O_r and O_j
    Let $\mathrm{Sep}(O_r, O_j) = \mathrm{Sep}(O_j, O_r) = \boldsymbol{W}$
    Add all triples $\langle O_{\min(r,j)}, \cdot, O_{\max(r,j)} \rangle$ that form a triangle in $\widehat{\mathbb{G}}$ into $\mathcal{M}$
    Delete from $\mathcal{M}$ and $\mathcal{L}$ all triples containing (O_r, O_j) : $\langle O_r, O_j, \cdot \rangle$, $\langle O_j, O_r, \cdot \rangle$, $\langle \cdot, O_j, O_r \rangle$ and $\langle \cdot, O_r, O_j \rangle$
    Delete edge $O_r \ast\!\!-\!\!\ast O_j$ in $\widehat{\mathbb{G}}$
   end
  end
 end
 Remove $\langle O_i, O_j, O_k \rangle$ from $\mathcal{M}$
end
forall *elements* $\langle O_i, O_j, O_k \rangle$ *of* $\mathcal{L}$ **do**
 if $O_j \notin \mathrm{Sep}(O_i, O_k)$ *and both* $O_i \ast\!\!-\!\!\ast O_j$ *and* $O_j \ast\!\!-\!\!\ast O_k$ *are present in* $\widehat{\mathbb{G}}$ **then**
  Orient $O_i \ast\!\!-\!\!\circ O_j \circ\!\!-\!\!\ast O_k$ as $O_i \ast\!\!\rightarrow O_j \leftarrow\!\!\ast O_k$ in $\widehat{\mathbb{G}}$
 end
end

Algorithm 6: Orienting v-structures in the RFCI algorithm

Then there is a path $\Pi_{X_iX_j}$ in $\mathbb{G}$ that d-connects X_i and X_j given $\boldsymbol{W}$. In addition, if all of the edges in all of the paths in $\mathcal{T}$ that contain X_i are into (out of) X_i, then $\Pi_{X_iX_j}$ is into (out of) X_i, and similarly for X_j.

Lemma 15. *Consider a directed graph with vertices O_i and O_j as well as a set of vertices $\boldsymbol{R}$ such that $O_i, O_j \notin \boldsymbol{R}$. Suppose that there is a set $\boldsymbol{W} \setminus \{O_i, O_j\}$ such that $\boldsymbol{R} \subseteq \boldsymbol{W}$ and every proper subset $\boldsymbol{V} \subset \boldsymbol{W}$ where $\boldsymbol{R} \subseteq \boldsymbol{V}$ d-connects O_i and O_j given $\boldsymbol{V} \cup \boldsymbol{S}$. If O_i and O_j are d-separated given $\boldsymbol{W} \cup \boldsymbol{S}$ where $O_k \in \boldsymbol{W}$, then O_k is an ancestor of $\{O_i, O_j\} \cup \boldsymbol{R} \cup \boldsymbol{S}$.*

Proof. We will prove the claim by contrapositive. That is, we will prove the following statement: suppose that there is a set $\boldsymbol{W} \setminus \{O_i, O_j\}$ and every proper subset $\boldsymbol{V} \subset \boldsymbol{W}$ where $\boldsymbol{R} \subseteq \boldsymbol{V}$ d-connects O_i and O_j given $\boldsymbol{V} \cup \boldsymbol{S}$. If O_k is not an ancestor of $\{O_i, O_j\} \cup \boldsymbol{R} \cup \boldsymbol{S}$, then O_i and O_j are d-connected given $\boldsymbol{W} \cup \boldsymbol{S}$ where $O_k \in \boldsymbol{W}$.

Let $\boldsymbol{W}^* = \text{Anc}(\{O_i, O_j\} \cup \boldsymbol{R} \cup \boldsymbol{S}) \cap \boldsymbol{W}$. Note that $\boldsymbol{W}^*$ is a proper subset of $\boldsymbol{W}$ because $\boldsymbol{W}^*$ is a subset of $\boldsymbol{W} \setminus O_k$, so O_i and O_j must be d-connected given $\boldsymbol{W}^* \cup \boldsymbol{S}$ by a path Π by assumption. By the definition of a d-connecting path, we know that every element in Π must be an ancestor of O_i, O_j, $\boldsymbol{R}$, $\boldsymbol{S}$ or $\boldsymbol{W}^*$ (or some union). Moreover, because $\boldsymbol{W}^* = \text{Anc}(\{O_i, O_j\} \cup \boldsymbol{R} \cup \boldsymbol{S}) \cap \boldsymbol{W}$, every element in $\boldsymbol{W}^*$ is an ancestor of $\{O_i, O_j\} \cup \boldsymbol{R} \cup \boldsymbol{S}$. Thus every element on the path Π is an ancestor of $\{O_i, O_j\} \cup \boldsymbol{R} \cup \boldsymbol{S}$. Since $\boldsymbol{W}^* \subset \boldsymbol{W}$, the only way in which Π could fail to d-connect O_i and O_j given $\boldsymbol{W} \cup \boldsymbol{S}$ would be if some element of $\boldsymbol{W} \setminus \boldsymbol{W}^*$ were located on Π. But neither O_k nor any element in $\boldsymbol{W} \setminus \boldsymbol{W}^*$ is an ancestor of $\{O_i, O_j\} \cup \boldsymbol{R} \cup \boldsymbol{S}$, so it follows that no vertex in $\boldsymbol{W} \setminus \boldsymbol{W}^*$ lies on Π. We conclude that O_i and O_j are d-connected given $\boldsymbol{W} \cup \boldsymbol{S}$. $\square$

13.2. Step 1: Skeleton Discovery

Lemma 1. *There exists an inducing path between O_i and O_j if and only if O_i and O_j are d-connected given $\boldsymbol{W} \cup \boldsymbol{S}$ for all possible subsets $\boldsymbol{W} \subseteq \boldsymbol{O} \setminus \{O_i, O_j\}$.*

Proof. I first prove the forward direction. Consider any set $\boldsymbol{W} \subseteq \boldsymbol{O} \setminus \{O_i, O_j\}$. Suppose there exists an inducing path Π between O_i and O_j. We have two situations:

1. There exists a collider C_1 on Π that is an ancestor of O_i via a directed path $C_1 \rightsquigarrow O_i$ but not an ancestor of $\boldsymbol{W} \cup \boldsymbol{S}$. Let C_1 more specifically

be such a collider on Π closest to O_j. Now one of the following two conditions will hold:

(a) There also exists a collider C_2 on Π that is an ancestor of O_j via a directed path $C_2 \rightsquigarrow O_j$ but not an ancestor of $\boldsymbol{W} \cup \boldsymbol{S}$. Let C_2 more specifically denote such a collider which is closest to C_1 on Π (if two such colliders are equidistant from C_1, then choose one arbitrarily). Let $\Pi_{C_1C_2}$ denote the part of the inducing path between C_1 and C_2. Recall that every non-collider on $\Pi_{C_1C_2}$ is a member of $\boldsymbol{L}$ because Π is an inducing path. Moreover, every collider on $\Pi_{C_1C_2}$ is an ancestor $\boldsymbol{W} \cup \boldsymbol{S}$ by construction. Then the path $\mathcal{T} = \{O_i \leftsquigarrow C_1, \Pi_{C_1C_2}, C_2 \rightsquigarrow O_j\}$ is a d-connecting path by invoking Lemma 14 with $\mathcal{T}$.

(b) There does not exist a collider C_2 on Π that is an ancestor of O_j via a directed path $C_2 \rightsquigarrow O_j$ and not an ancestor of $\boldsymbol{W} \cup \boldsymbol{S}$. It follows that all colliders on Π are ancestors of $O_i \cup \boldsymbol{W} \cup \boldsymbol{S}$. More specifically, all of the colliders on $\Pi_{O_jC_1}$ are ancestors of $\boldsymbol{W} \cup \boldsymbol{S}$ by construction. Recall also that every non-collider on $\Pi_{O_jC_1}$ is a member of $\boldsymbol{L}$ because Π is an inducing path. We conclude that the path $\mathcal{T} = \{\Pi_{O_jC_1}, C_1 \rightsquigarrow O_i\}$ is a d-connecting path by invoking Lemma 14 with $\mathcal{T}$.

2. There does not exist a collider C_1 on Π that is an ancestor of O_i via a directed path $C_1 \rightsquigarrow O_i$ and not an ancestor of $\boldsymbol{W} \cup \boldsymbol{S}$. This implies that all colliders on Π are ancestrs of $O_j \cup \boldsymbol{W} \cup \boldsymbol{S}$. Let $\Pi_{O_iC_3}$ correspond to the part of the inducing path between O_i and C_3, where C_3 corresponds to the collider closest to O_i that is an ancestor of O_j via a directed path $C_3 \rightsquigarrow O_j$ but not an ancestor of $\boldsymbol{W} \cup \boldsymbol{S}$; if we do not encounter such a collider, then set $C_3 = O_j$. Notice then that all colliders on $\Pi_{O_iC_3}$ are ancestors of $\boldsymbol{W} \cup \boldsymbol{S}$. Recall also that every non-collider on $\Pi_{O_iC_3}$ is a member of $\boldsymbol{L}$ because Π is an inducing path. Thus the path $\mathcal{T} = \{\Pi_{O_iC_3}, C_3 \rightsquigarrow O_j\}$ is a d-connecting path by invoking Lemma 14 with $\mathcal{T}$.

For the backward direction, assume O_i and O_j are d-connecting given $\boldsymbol{W} \cup \boldsymbol{S}$ for all possible subsets $\boldsymbol{W} \subseteq \boldsymbol{O} \setminus \{O_i, O_j\}$. Then O_i and O_j are d-connected given $((\text{Anc}(\{O_i, O_j\} \cup \boldsymbol{S}) \cap \boldsymbol{O}) \cup \boldsymbol{S}) \setminus \{O_i, O_j\}$. The backward direction follows by invoking Lemma 8 in (Spirtes et al., 1999) whose argument remains unchanged even for a cyclic directed graph. $\square$

Lemma 2. *If there does not exist an inducing path between O_i and O_j, then O_i and O_j are d-separated given* $\text{D-SEP}(O_i, O_j) \cup \boldsymbol{S}$. *Likewise, O_i and O_j are d-separated given* $\text{D-SEP}(O_j, O_i) \cup \boldsymbol{S}$.

Proof. We will prove this by contradiction. Assume that we have $O_i \not\perp\!\!\!\perp_d O_j | \text{D-SEP}(O_i, O_j) \cup \boldsymbol{S}$. If there does not exist an inducing path between O_i and O_j, then there exists some $\boldsymbol{W} \subseteq \boldsymbol{O} \setminus \{O_i, O_j\}$ such that $O_i \perp\!\!\!\perp_d O_j | \boldsymbol{W} \cup \boldsymbol{S}$ by Lemma 1. Let Π correspond to the path d-connecting O_i and O_j given $\text{D-SEP}(O_i, O_j) \cup \boldsymbol{S}$.

We have two conditions:

1. Suppose that every vertex in $\boldsymbol{O}$ on Π is a collider on Π. This implies that all non-colliders on Π must be in $\boldsymbol{L} \cup \boldsymbol{S}$. But no non-collider on Π can be in $\boldsymbol{S}$ because Π would be inactive in that case. Thus all non-colliders on Π must more specifically be in $\boldsymbol{L}$. Now recall that we assumed that $O_i \not\perp\!\!\!\perp_d O_j | \text{D-SEP}(O_i, O_j) \cup \boldsymbol{S}$, so every collider on Π (including those in $\boldsymbol{O}$) must be an ancestor of $\text{D-SEP}(O_i, O_j) \cup \boldsymbol{S}$ and hence also an ancestor of $\{O_i, O_j\} \cup \boldsymbol{S}$. The above facts imply that there exists an inducing path between O_i and O_j; contradiction.

2. Suppose that there exists at least one vertex in $\boldsymbol{O}$ on Π that is a non-collider. Let O_k denote the first such vertex on Π closest to O_i. Note that every vertex on Π is an ancestor of $\{O_i, O_j\} \cup \text{D-SEP}(O_i, O_j) \cup \boldsymbol{S}$ by the definition of d-connection and hence an ancestor of $\{O_i, O_j\} \cup \boldsymbol{S}$. This implies that O_k is an ancestor of $\{O_i, O_j\} \cup \boldsymbol{S}$.

   We will show that $O_k \in \text{D-SEP}(O_i, O_j)$ in order to arrive at the contradiction that Π does not d-connect O_i and O_j given $\text{D-SEP}(O_i, O_j) \cup \boldsymbol{S}$. Consider the subpath $\Pi_{O_i O_k}$. Let $\langle C_1, \dots, C_m \rangle$ denote the possibly empty sequence of colliders on $\Pi_{O_i O_k}$ which are ancestors of $\text{D-SEP}(O_i, O_j)$ but not $\boldsymbol{S}$. Also let C_n denote an arbitrary collider in $\langle C_1, \dots, C_m \rangle$. Notice that there is a directed path $C_n \rightsquigarrow O_n$ with $O_n \in \text{D-SEP}(O_i, O_j)$. Let F_n denote the first observable on $C_n \rightsquigarrow O_n$ which may be O_n if no other observable lies on $C_n \rightsquigarrow O_n$.

   We will show that there exists an inducing path between F_n and F_{n+1}, where F_{n+1} corresponds to the first observable on $C_{n+1} \rightsquigarrow O_{n+1}$. First note that $F_n, F_{n+1} \notin \text{Anc}(\boldsymbol{S})$ because $C_n, C_{n+1} \notin \text{Anc}(\boldsymbol{S})$. Consider the path Φ_n constructed by concatenating the paths $C_n \rightsquigarrow F_n$, $\Pi_{C_n C_{n+1}}$ and $C_{n+1} \rightsquigarrow F_{n+1}$. Notice that, by construction, the only observables in Φ_n lie on $\Pi_{C_n C_{n+1}}$. Moreover, every observable on $\Pi_{C_n C_{n+1}}$ is a collider because O_k is the first observable that is a non-collider on Π; this implies that only a latent or a selection variable on $\Pi_{C_n C_{n+1}}$ can be a non-collider. But no selection variable is also a non-collider on $\Pi_{C_n C_{n+1}}$ because Π d-connects O_i and O_j given D-SEP$(O_i, O_j) \cup \boldsymbol{S}$. We conclude that only a latent variable can be a non-collider on $\Pi_{C_n C_{n+1}}$. Next, every collider on $\Pi_{C_n C_{n+1}}$ is an ancestor of $\boldsymbol{S}$ by construction of $\langle C_1, \ldots, C_m \rangle$. We have shown that all colliders on Φ_n are ancestors of $\boldsymbol{S}$ and all non-colliders on Φ_n are in $\boldsymbol{L}$. This implies that Φ_n is an inducing path between F_n and F_{n+1}; specifically one that is into F_n and into F_{n+1} by construction.

We will now tie up the endpoints. We can also concatenate the paths $\Pi_{O_i C_1}$ and $C_1 \rightsquigarrow F_1$ in order to form an inducing path Φ_0 between O_i and F_1 that is into F_1. Similarly, we can concatenate the paths $\Pi_{O_k C_m}$ and $C_m \rightsquigarrow F_m$ in order to form an inducing path Φ_m between O_k and F_m that is into F_m.

We have constructed a sequence of vertices $\langle O_i \equiv F_0, F_1, \ldots, F_m, F_{m+1} \equiv O_k \rangle$, where each vertex is an ancestor of $\{O_i, O_j\} \cup \boldsymbol{S}$ and any given F_l is connected to F_{l-1} by an inducing path into F_l and to F_{l+1} by an inducing path also into F_l. Hence, $O_k \in$ D-SEP(O_i, O_j). But this implies that Π does not d-connect O_i and O_j given D-SEP$(O_i, O_j) \cup \boldsymbol{S}$ because O_k is a non-collider on Π; contradiction.

We have shown that, if there does not exist an inducing path between O_i and O_j, then $O_i \perp\!\!\!\perp_d O_j |$D-SEP$(O_i, O_j) \cup \boldsymbol{S}$. Now $O_i \perp\!\!\!\perp_d O_j |$D-SEP$(O_i, O_j) \cup \boldsymbol{S} \implies O_j \perp\!\!\!\perp_d O_i |$D-SEP$(O_j, O_i) \cup \boldsymbol{S}$ because i and j are arbitrary indices. Moreover, $O_j \perp\!\!\!\perp_d O_i |$D-SEP$(O_j, O_i) \cup \boldsymbol{S} \implies O_i \perp\!\!\!\perp_d O_j |$D-SEP$(O_j, O_i) \cup \boldsymbol{S}$ because $O_j \perp\!\!\!\perp_d O_i |$D-SEP$(O_j, O_i) \cup \boldsymbol{S}$ if and only if $O_i \perp\!\!\!\perp_d O_j |$D-SEP$(O_j, O_i) \cup \boldsymbol{S}$ by symmetry of d-separation. We conclude that, if there does not exist an inducing path between O_i and O_j, then we also have $O_i \perp\!\!\!\perp_d O_j |$D-SEP$(O_j, O_i) \cup \boldsymbol{S}$.

□

Lemma 3. *If an inducing path does not exist between O_i and O_j in $\mathbb{G}$, then O_i and O_j are d-separated given $\boldsymbol{W} \cup \boldsymbol{S}$ with $\boldsymbol{W} \subseteq$ PD-SEP(O_i) in the MAAG $\mathbb{G}'$. Likewise, O_i and O_j are d-separated given some $\boldsymbol{W} \cup \boldsymbol{S}$ with $\boldsymbol{W} \subseteq$ PD-SEP(O_j) in $\mathbb{G}'$.*

Proof. It suffices to show that D-SEP$(O_i, O_j) \subseteq$ PD-SEP(O_i) by Lemma 2. The argument will hold analogously for D-SEP$(O_j, O_i) \subseteq$ PD-SEP(O_j). If $O_k \in$ D-SEP(O_i, O_j), then there exists a sequence of observables Π_{O_i,O_k} between O_i and O_k such that an inducing path exists between any two consecutive observables $\langle O_h, O_{h+1}\rangle$ in Π_{O_i,O_k}. Thus there also exists a path Π'_{O_i,O_k} between O_i and O_k in $\mathbb{G}'$ whose vertices involve all and only the vertices in Π_{O_i,O_k}. We also know that, in every consecutive triplet $\langle O_{h-1}, O_h, O_{h+1}\rangle$, the inducing path from O_{h-1} to O_h is into O_h, and the inducing path from O_{h+1} to O_h is also into O_h; hence, O_h is a collider in $\mathbb{G}$. We now need to show that any triplet $\langle O_{h-1}, O_h, O_{h+1}\rangle$ on Π'_{O_i,O_k} is a v-structure in $\mathbb{G}'$ or a triangle in $\mathbb{G}'$. We have two situations:

1. Suppose that the collider $O_h \not\in \text{Anc}(\{O_{h-1}, O_{h+1}\} \cup \boldsymbol{S})$. Then, the path between O_{h-1} and O_h and then between O_h and O_{h+1} is not an inducing path. Hence, O_h lies in an unshielded triple involving $\langle O_{h-1}, O_h, O_{h+1}\rangle$ on Π'_{O_i,O_k}. If O_h lies in the unshielded triple, then O_h more specifically lies in a v-structure because $O_h \not\in \text{Anc}(\{O_{h-1}, O_{h+1}\} \cup \boldsymbol{S})$ by assumption.

2. Suppose that $O_h \in \text{Anc}(\{O_{h-1}, O_{h+1}\} \cup \boldsymbol{S})$. Then there exists an inducing path between O_{h-1} and O_{h+1}, so O_h is in a triangle on Π'_{O_i,O_k}.

□

Lemma 4. *If an inducing path does not exist between O_i and O_j in $\mathbb{G}$, then O_i and O_j are d-separated given $\boldsymbol{W} \cup \boldsymbol{S}$ with $\boldsymbol{W} \subseteq$ PD-SEP(O_i) in $\mathbb{G}''$. Likewise, O_i and O_j are d-separated given some $\boldsymbol{W} \cup \boldsymbol{S}$ with $\boldsymbol{W} \subseteq$ PD-SEP(O_j) in $\mathbb{G}''$.*

Proof. In light of Lemma 3, it suffices to show that PD-SEP(O_i) formed using the MAAG $\mathbb{G}'$ is a subset of PD-SEP(O_i) formed using $\mathbb{G}''$. Recall that all edges in $\mathbb{G}'$ are also in $\mathbb{G}''$. Hence, all triangles in $\mathbb{G}'$ are also triangles in $\mathbb{G}''$. We now need to show that all v-structures in $\mathbb{G}'$ are also v-structures in $\mathbb{G}''$ or are triangles in $\mathbb{G}''$. Let $\langle O_{h-1}, O_h, O_{h+1}\rangle$ denote an arbitrary v-structure in $\mathbb{G}'$. The edge between O_{h-1} and O_h as well as the edge between O_h and O_{h+1} must be in $\mathbb{G}''$, because again all edges in $\mathbb{G}'$ are also in $\mathbb{G}''$. We have two cases:

1. An edge exists between O_{h-1} and O_{h+1} in $\mathbb{G}''$. Then the triple $\langle O_{h-1}, O_h, O_{h+1}\rangle$ forms a triangle in $\mathbb{G}''$.

2. An edge does not exist between O_{h-1} and O_{h+1} in $\mathbb{G}''$. Recall that $\langle O_{h-1}, O_h, O_{h+1}\rangle$ is a v-structure in $\mathbb{G}'$, so $O_h \notin \text{Anc}(\{O_{h-1}, O_{h+1}\} \cup \boldsymbol{S})$. Note that PC's skeleton discovery procedure only discovers minimal separating sets so, if we have $O_{h-1} \perp\!\!\!\perp_d O_{h+1} | \boldsymbol{W} \cup \boldsymbol{S}$ with $\boldsymbol{W} \subseteq \boldsymbol{O} \setminus \{O_{h-1}, O_{h+1}\}$ and $O_h \in \boldsymbol{W}$, then $O_h \in \text{Anc}(\{O_{h-1}, O_{h+1}\} \cup \boldsymbol{S})$ by Lemma 15 with $\boldsymbol{R} = \emptyset$; but this contradicts the fact that $O_h \notin \text{Anc}(\{O_{h-1}, O_{h+1}\} \cup \boldsymbol{S})$. Hence $O_h \notin \boldsymbol{W}$, so $\langle O_{h-1}, O_h, O_{h+1}\rangle$ is also a v-structure in $\mathbb{G}''$.

□

13.3. Steps 3 & 4: Short and Long Range Non-Ancestral Relations

Lemma 16. *If O_i is an ancestor of $O_j \cup \boldsymbol{S}$, O_j and some vertex O_k are d-separated given $\boldsymbol{W} \cup \boldsymbol{S}$ with $\boldsymbol{W} \subseteq \boldsymbol{O} \setminus \{O_j, O_k\}$, O_i and O_j are d-connected given $\boldsymbol{W} \cup \boldsymbol{S}$, and $O_i \notin \boldsymbol{W}$, then O_i and O_k are d-separated given $\boldsymbol{W} \cup \boldsymbol{S}$.*

Proof. Suppose for a contradiction that O_i and O_k are d-connected given $\boldsymbol{W} \cup \boldsymbol{S}$. There are two cases.

In the first case, suppose that O_i has a descendant in $\boldsymbol{W} \cup \boldsymbol{S}$. Recall however that we have $O_i \notin \boldsymbol{W} \cup \boldsymbol{S}$, so we can merge the d-connecting path $\Pi_{O_j O_i}$ between O_j and O_i and the d-connecting path $\Pi_{O_i O_k}$ between O_i and O_k by invoking Lemma 14 with $\mathcal{T} = \{\Pi_{O_j O_i}, \Pi_{O_i O_k}\}$ in order to form a d-connecting path between O_j and O_k given $\boldsymbol{W} \cup \boldsymbol{S}$. We have arrived at a contradiction.

In the second case, suppose that O_i does not have a descendant in $\boldsymbol{W} \cup \boldsymbol{S}$. Recall also that O_i is an ancestor of $O_j \cup \boldsymbol{S}$ by assumption. These two facts imply that there exists a directed path $O_i \rightsquigarrow O_j$ that does not include $\boldsymbol{W} \cup \boldsymbol{S}$; hence the $O_i \rightsquigarrow O_j$ is d-connecting. We can again invoke Lemma 14 with $\mathcal{T} = \{O_j \leftsquigarrow O_i, \Pi_{O_i O_k}\}$ in order to form a d-connecting path between O_j and O_k given $\boldsymbol{W} \cup \boldsymbol{S}$. We have thus arrived at another contradiction.

We have exhausted all possibilities and therefore conclude that O_i and O_k are in fact d-separated given $\boldsymbol{W} \cup \boldsymbol{S}$. □

We can write the contrapositive of the above lemma as follows:

Corollary 2. *Let $\boldsymbol{W} \subseteq \boldsymbol{O} \setminus \{O_j, O_k\}$. If O_i and O_j are d-connected given $\boldsymbol{W} \cup \boldsymbol{S}$, O_k and O_i are d-connected given $\boldsymbol{W} \cup \boldsymbol{S}$, O_k and O_j are d-separated given $\boldsymbol{W} \cup \boldsymbol{S}$, and $O_i \notin \boldsymbol{W}$, then O_i is not an ancestor of $O_j \cup \boldsymbol{S}$.*

Lemma 5. *Consider a set* $\boldsymbol{W} \subseteq \boldsymbol{O} \setminus \{O_i, O_j\}$*. Now suppose that* O_i *and* O_k *are d-connected given* $\boldsymbol{W} \cup \boldsymbol{S}$*, and that* O_j *and* O_k *are d-connected given* $\boldsymbol{W} \cup \boldsymbol{S}$*. If* O_i *and* O_j *are d-separated given* $\boldsymbol{W} \cup \boldsymbol{S}$ *such that* $O_k \notin \boldsymbol{W}$*, then* O_k *is not an ancestor of* $\{O_i, O_j\} \cup \boldsymbol{S}$*.*

Proof. Follows by applying Corollary 2 twice with O_i and O_k d-connected and with O_j and O_k d-connected. □

13.4. Step 5: Orienting with Non-Minimal D-Separating Sets

Lemma 6. *Consider a quadruple of vertices* $\langle O_i, O_j, O_k, O_l \rangle$*. Suppose that we have:*

1. O_i *and* O_k *non-adjacent.*
2. $O_i {*\!\!\rightarrow} O_l \leftarrow\!\!{*} O_k$*.*
3. O_i *and* O_k *are d-separated given some* $\boldsymbol{W} \cup \boldsymbol{S}$ *with* $O_j \in \boldsymbol{W}$ *and* $\boldsymbol{W} \subseteq \boldsymbol{O} \setminus \{O_i, O_k\}$*;*
4. $O_j \,{*\!\!-\!\!\circ}\, O_l$*.*

If $O_l \notin \boldsymbol{W} = \mathrm{Sep}(O_i, O_k)$*, then we have* $O_j {*\!\!\rightarrow} O_l$*. If* $O_i {*\!\!\rightarrow} O_j \leftarrow\!\!{*} O_k$ *and* $O_l \in \boldsymbol{W} = \mathrm{SupSep}(O_i, O_j, O_k)$*, then we have* $O_j \,{*\!\!-}\, O_l$*.*

Proof. We prove the first conclusion by contrapositive. Assume that we have $O_j \,{*\!\!-}\, O_l$. Now suppose for a contradiction that $O_l \notin \boldsymbol{W}$ (but $O_j \in \boldsymbol{W}$). Note that $O_j \cup \boldsymbol{S}$ contains at least one descendant of O_l because $O_l \in \mathrm{Anc}(O_j \cup \boldsymbol{S})$. With Lemma 14, we can use the d-connecting path between O_i and O_l given $\boldsymbol{W} \cup \boldsymbol{S}$ as well as the d-connecting path between O_k and O_l given $\boldsymbol{W} \cup \boldsymbol{S}$ to form a d-connecting path between O_i and O_k given $\boldsymbol{W} \cup \boldsymbol{S}$ irrespective of whether or not the paths collide at O_l; this contradicts the fact that O_i and O_k are d-separated given $\boldsymbol{W} \cup \boldsymbol{S}$.

For the second conclusion, assume that we have $O_l \in \boldsymbol{W}$. We know from Lemma 15 with $\boldsymbol{R} = O_j \cup \mathrm{Sep}(O_i, O_k)$ that O_l is an ancestor of $\{O_i, O_j, O_k\} \cup \mathrm{Sep}(O_i, O_k) \cup \boldsymbol{S}$. Recall that every member of $\mathrm{Sep}(O_i, O_k)$ is an ancestor of $\{O_i, O_k\} \cup \boldsymbol{S}$ by setting $\boldsymbol{R} = \emptyset$. Hence, O_l is more specifically an ancestor of $\{O_i, O_j, O_k\} \cup \boldsymbol{S}$. Now since we have $O_i {*\!\!\rightarrow} O_l \leftarrow\!\!{*} O_k$, we can also claim that we have $O_l \in \mathrm{Anc}(O_j)$. Hence, we have $O_j \,{*\!\!-}\, O_l$. □

13.5. Step 6: Long Range Ancestral Relations

Lemma 7. *If O_i and O_k are d-separated given $\boldsymbol{W} \cup \boldsymbol{S}$, where $\boldsymbol{W} \subseteq \boldsymbol{O} \setminus \{O_i, O_k\}$, and $\boldsymbol{Q} \subseteq \text{Anc}(\{O_i, O_k\} \cup \boldsymbol{W} \cup \boldsymbol{S}) \setminus \{O_i, O_k\}$, then O_i and O_k are also d-separated given $\boldsymbol{Q} \cup \boldsymbol{W} \cup \boldsymbol{S}$.*

Proof. We will prove this by contrapositive. Suppose that there is a path $\Pi_{O_iO_k}$ which d-connects O_i and O_k given some $\boldsymbol{Q} \cup \boldsymbol{W} \cup \boldsymbol{S}$. Then every vertex on $\Pi_{O_iO_k}$ is an ancestor of $\{O_i, O_k\} \cup \boldsymbol{Q} \cup \boldsymbol{W} \cup \boldsymbol{S}$ by the definition of a d-connecting path. Since $\boldsymbol{Q} \subseteq \text{Anc}(\{O_i, O_k\} \cup \boldsymbol{W} \cup \boldsymbol{S}) \setminus \{O_i, O_k\}$, every vertex on $\Pi_{O_iO_k}$ must more specifically be an ancestor of $\{O_i, O_k\} \cup \boldsymbol{W} \cup \boldsymbol{S}$.

Let O_a denote the collider furthest from O_i on $\Pi_{O_iO_k}$ which is an ancestor of $O_i \cup \boldsymbol{S}$ and not in $\boldsymbol{W} \cup \boldsymbol{S}$ (or O_i if no such collider exists). Similarly, let O_b denote the first collider after O_a on $\Pi_{O_iO_k}$ which is an ancestor of $O_k \cup \boldsymbol{S}$ and not in $\boldsymbol{W} \cup \boldsymbol{S}$ (or O_k if no such collider exists). The directed path $\Pi_{O_aO_i}$ from O_a to $O_i \cup \boldsymbol{S}$, and the directed path $\Pi_{O_bO_k}$ from O_b to $O_k \cup \boldsymbol{S}$ are d-connecting given $\boldsymbol{W} \cup \boldsymbol{S}$, since no vertices on the path $\Pi_{O_aO_i}$ or $\Pi_{O_bO_k}$ are in $\boldsymbol{W} \cup \boldsymbol{S}$. The subpath of $\Pi_{O_aO_b}$ between O_a and O_b on $\Pi_{O_iO_k}$ is also d-connecting given $\boldsymbol{W} \cup \boldsymbol{S}$ because every collider is an ancestor of $\boldsymbol{W} \cup \boldsymbol{S}$, and every non-collider is in $\boldsymbol{L}$. Lemma 14 implies that we can take $\mathcal{T} = \{\Pi_{O_aO_i}, \Pi_{O_aO_b}, \Pi_{O_bO_k}\}$ to form a d-connecting path between O_i and O_k given $\boldsymbol{W} \cup \boldsymbol{S}$. □

13.6. Step 7: Orientation Rules

Lemma 8. *Suppose that there is a set $\boldsymbol{W} \setminus \{O_i, O_j\}$ and every proper subset $\boldsymbol{V} \subset \boldsymbol{W}$ d-connects O_i and O_j given $\boldsymbol{V} \cup \boldsymbol{S}$. If O_i and O_j are d-separated given $\boldsymbol{W} \cup \boldsymbol{S}$ where $O_k \in \boldsymbol{W}$, then O_k is an ancestor of $\{O_i, O_j\} \cup \boldsymbol{S}$.*

Proof. This is a special case of Lemma 15 with $\boldsymbol{R} = \emptyset$. □

Lemma 10. *If we have $O_i {*\!\!\rightarrow} O_j — O_k$ with O_i and O_k non-adjacent, then $O_i {*\!\!\rightarrow} O_j$ is in a triangle involving O_i, O_j and O_l ($l \neq k$) with $O_j — O_l$ and $O_i {*\!\!\rightarrow} O_l$. Moreover, there exists a sequence of undirected edges between O_l and O_k that does not include O_j.*

Proof. Note that O_j or O_k (or both) cannot be ancestors of $\boldsymbol{S}$ because this would contradict the arrowhead at O_j. Therefore, O_j is an ancestor of O_k, and O_k is an ancestor of O_j, so there is a cycle involving O_j and O_k. Since we have an arrowhead at O_j, there must be an inducing path $\Pi_{O_iO_j}$ between O_i and O_j that is either out of O_j or into O_j:

1. Suppose that $\Pi_{O_iO_j}$ is out of O_j. Every vertex on $\Pi_{O_iO_j}$ is an ancestor of $\{O_i, O_j\} \cup \boldsymbol{S}$ by the definition of an inducing path. Thus, $O_j \in \text{Anc}(\{O_i, O_j\} \cup \boldsymbol{S})$. Recall that we also have the arrowhead $O_i * \!\!\rightarrow O_j$, so we more specifically have the obvious relation $O_j \in \text{Anc}(O_j)$. Let C_1 denote the collider closest to O_j on $\Pi_{O_iO_j}$. Such a collider must exist or else $O_j \in \text{Anc}(O_i)$ which contradicts the arrowhead $O_i * \!\!\rightarrow O_j$. Since $\Pi_{O_iO_j}$ is an inducing path, we must have $C_1 \in \text{Anc}(\{O_i, O_j\} \cup \boldsymbol{S})$. However, C_1 cannot be an ancestor of $O_i \cup \boldsymbol{S}$ because that would imply that we have $O_j \in \text{Anc}(O_i \cup \boldsymbol{S})$. We therefore more specifically have $C_1 \in \text{Anc}(O_j)$. Let $C_1 \rightsquigarrow O_j$ denote a directed path to O_j. We have two scenarios:

   (a) $C_1 \rightsquigarrow O_j$ contains a member of $\boldsymbol{O}$ besides O_j. Denote that member of $\boldsymbol{O}$ closest to C_1 as O_l (note that we may have $C_1 = O_l$). Then $\Pi_{O_iC_1}$, the part of $\Pi_{O_iO_j}$ between O_i and C_1, as well as $C_1 \rightsquigarrow O_l$ together form an inducing path between O_i and O_l (every non-collider on $C_1 \rightsquigarrow O_l$ is in $\boldsymbol{L}$ by construction). Moreover, we must have $O_i * \!\!\rightarrow O_l$ because $O_l \notin \text{Anc}(O_i \cup \boldsymbol{S})$ by construction. There also exists an inducing path $\Pi_{O_jO_l}$ between O_j and O_l because all non-colliders on $\Pi_{O_jO_l}$ are in $\boldsymbol{L}$. We more specifically must have $O_j - O_l$ because $O_j \in \text{Anc}(O_l)$ and $O_l \in \text{Anc}(O_j)$ by construction. Finally, there exists a sequence of undirected edges to O_k because every member of $\boldsymbol{O}$ on $C_1 \rightsquigarrow O_j$ between O_l and O_k is an ancestor of O_k and O_k is an ancestor of them.

   (b) $C_1 \rightsquigarrow O_j$ does not contain a member of $\boldsymbol{O}$ besides O_j. But then $\Pi_{O_iC_1}$ as well as $C_1 \rightsquigarrow O_j$ form an inducing path because every non-collider on $C_1 \rightsquigarrow O_j$ must be in $\boldsymbol{L}$. Hence, there exists an inducing path between O_i and O_j that is into O_j. See below for the continuation of the argument.

2. Suppose that $\Pi_{O_iO_j}$ is into O_j. We also know that there is an inducing path between O_j and O_k. Furthermore, there exists a directed path from O_k to O_j by the first paragraph. Hence, there exists an inducing path $\Pi_{O_jO_l}$ between some variable O_l (O_l is in the cycle involving O_j and O_k with possibly $l = k$) and O_j which is into O_j. Suppose $l = k$; but this would imply that O_i and O_k are adjacent in the MAAG, since $\Pi_{O_iO_j}$ and $\Pi_{O_jO_k}$ would together form an inducing path between O_i and O_k (the collider O_j is an ancestor of O_k). Hence, the inducing

path must involve O_j and some other observable O_l where $l \neq k$. Call this inducing path $\Pi_{O_j O_l}$. Note that the path $\{\Pi_{O_i O_j}, \Pi_{O_j O_l}\}$ is an inducing path between O_i and O_l because O_j is an ancestor of O_l. Thus $O_i {*\!\!\rightarrow} O_j$ is in a triangle involving O_i, O_j and O_l.

Finally recall that O_l is a member of a cycle involving O_j and O_k. Hence O_l is an ancestor of O_j and O_j is an ancestor of O_l. Now O_l is also not an ancestor of $\boldsymbol{S}$ because otherwise both O_j and O_k would also be ancestors of $\boldsymbol{S}$. Next suppose for a contradiction that O_l is an ancestor of O_i. Then O_j must be an ancestor of O_i which contradicts the arrowhead $O_i {*\!\!\rightarrow} O_j$.

□

13.7. Main Result

Theorem 2. *(Soundness) Consider a DAG or a linear SEM-IE with directed cyclic graph $\mathbb{G}$. If d-separation faithfulness holds, then CCI outputs a partially oriented MAAG of $\mathbb{G}$.*

Proof. Under d-separation faithfulness, O_i and O_j are d-separated given $\boldsymbol{W} \subseteq \boldsymbol{O} \setminus \{O_i, O_j\}$ if and only if $O_i \perp\!\!\!\perp O_j | \{\boldsymbol{W} \cup \boldsymbol{S}\}$. Hence, we may use the terms d-separation and conditional independence as well as d-connection and conditional dependence interchangeably.

Lemma 1 implies that an inducing path exists in a maximal ancestral graph if and only if O_i and O_j are conditionally independent given all possible subsets of $\boldsymbol{O} \setminus \{O_i, O_j\}$ as well as $\boldsymbol{S}$. Lemmas 2 and 3 imply that we can discover the inducing paths using subsets of PD-SEP(O_i) and PD-SEP(O_j). Hence, Step 1 of CCI is sound.

We can justify Steps 2 and 3 by invoking Lemma 5. Correctness of Step 5 follows by by Lemma 6, and Step 6 by the contrapositive of Lemma 7. Finally, correctness of the orientation rules follows by invoking Lemmas 11, 12 and 13 for orientation rules 1-3, 4-5 and 6-7, respectively. □