Establishing Markov Equivalence in Cyclic Directed Graphs

Tom Claassen[1] **Joris M. Mooij**[2]

[1]Institute for Computing and Information Sciences, Radboud University, Nijmegen, Netherlands
[2]Korteweg-deVries Institute, University of Amsterdam, Amsterdam, Netherlands

Abstract

We present a new, efficient procedure to establish Markov equivalence between directed graphs that may or may not contain cycles under the *d*-separation criterion. It is based on the Cyclic Equivalence Theorem (CET) in the seminal works on cyclic models by Thomas Richardson in the mid '90s, but now rephrased from an ancestral perspective. The resulting characterization leads to a procedure for establishing Markov equivalence between graphs that no longer requires explicit tests for *d*-separation, leading to a significantly reduced algorithmic complexity. The conceptually simplified characterization may help to reinvigorate theoretical research towards sound and complete cyclic discovery in the presence of latent confounders.

1 INTRODUCTION

Discovering causal relations from observational and experimental data is one of the key goals in many research areas. Developing principled, automated causal discovery methods has been an active area of research within the machine learning community, which has resulted a wide variety of algorithms and techniques. Two of the main challenges here are handling the impact of unobserved confounders, and the possible presence of feedback mechanisms or cycles in the system under investigation. Both have a long history in the field: in this article we solely focus on the latter.

Building on earlier work by Spirtes (1994, 1995) on (linear) cyclic directed models that obey the global directed Markov property (see section 2.1, below), Richardson (1996b) introduced the Cyclic Causal Discovery (CCD) algorithm that was able to infer a sound cyclic causal model from independence constraints on data. It was based on the so-called Cyclic Equivalence Theorem (Richardson, 1997) that characterized Markov equivalence between cyclic directed graphs.

Strangely enough, after this promising start progress in cyclic directed models slowly ground to a halt, even though many challenges remained: the CCD output was certainly not complete, and could not account for latent confounders.

In the mean time theory and methods for acyclic causal discovery took flight, where, for example Zhang (2008) managed to extend FCI to a provably sound and complete algorithm under latent confounders and selection bias.

And even to this day fundamental progress continues to be made: recently several new and faster algorithms and characterizations for establishing Markov equivalence between maximal ancestral graphs (graphical independence models closed under marginalization and conditioning) have been developed (Hu and Evans, 2020; Wienöbst et al., 2022; Claassen and Bucur, 2022), ultimately bringing it down to linear complexity for sparse graphs. However, despite a widely acknowledged need to handle feedback cycles in learning algorithms for real world causal discovery, major steps towards that goal have been few and far between.

A promising attempt to extend CCD to the case of unobserved confounders was made by Strobl (2018), but though the resulting CCI algorithm was sound, it was by no means complete, foregoing on key FCI elements like discriminating paths and selection bias, and the output was not guaranteed to uniquely identify the Markov equivalence class.

Fundamentally different approaches to cyclic causal discovery have also been developed: for example, Lacerda et al. (2008) employs independent component analysis, Mooij et al. (2011); Mooij and Heskes (2013) proposed likelihood-based structure learning approaches for additive noise models, Hyttinen et al. (2012) exploits experiments to build a complete model, and Rothenhäusler et al. (2015) builds on information from unknown shift interventions to reconstruct the underlying cyclic causal graph.

On another front, Forré and Mooij (2018) showed that for *nonlinear* causal models with cycles and confounders, the

Accepted for the 39th Conference on Uncertainty in Artificial Intelligence (UAI 2023).

usual d-separation criterion needs to be replaced with their σ-separation criterion (see also section 3 in the supplement). More recently, Mooij and Claassen (2020) showed that vanilla FCI was in fact already sound and complete for these nonlinear cyclic models. However, it does not account for the peculiarities encountered when handling *linear* cyclic models, as in Figure 1.

For linear or discrete cyclic causal models, σ-separation is too weak, as the stronger d-separation applies. Perhaps surprisingly, this significantly complicates the causal structure analysis. But even in nonlinear systems we often consider linear approximations, which means in practice we may expect to encounter similar complications there as well. In section 3 in the supplement we summarize some results from the literature under which cyclic causal models are known to satisfy the stronger d-separation criterion. For the current paper it suffices to know that we focus on d-separation equivalence between cyclic directed graphs with no unobserved confounders, which, for the important class of systems where the global directed Markov condition in combination with its corresponding faithfulness assumption holds, also implies Markov equivalence.

Part of the reason for the slow progress on cyclic models that satisfy the d-separation criterion may be that the associated theoretical machinery developed to characterize Markov equivalence is quite imposing, which may make further extensions towards confounders seem an overly daunting task.

In this article we find things may not be quite as bad as perhaps once feared. We show, for example, that establishing Markov equivalence between directed graphs becomes more intuitive when viewed from an *ancestral* perspective, leading to a simplified characterization and an efficient algorithm that greatly speeds up identification. Although this is of course but a small step, we hope that it may inspire renewed investigation into full-fledged cyclic causal discovery in the presence of latent confounders and selection bias.

In the rest of the article, section 2 introduces the necessary tools to handle cyclic directed graphs, section 3 describes an alternative, ancestral formulation of the CET, section 4 shows how to infer a graphical characterization of the Markov equivalence class without the need for d-separation tests, and section 5 demonstrates the remarkable efficiency of the resulting procedure compared to current state of the art. Detailed proofs for all lemmas and theorems, as well as some additional experimental results are provided in the accompanying supplement.

2 CYCLIC DIRECTED GRAPHS

In this section we start with a few standard graphical model definitions, and then continue with some perhaps less familiar terminology and results specific to cyclic graphs.

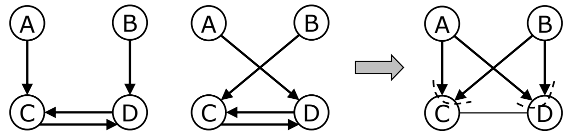

Figure 1: Two different cyclic graphs (left) that together form the only two members of the Markov equivalence class on the right, where the dashed lines signal two *virtual* v-structures (see §3.1). For linear/discrete models conditioning on C would make A and B dependent, but conditioning on $\{C, D\}$ would not.

2.1 GRAPH NOTATIONS AND TERMINOLOGY

Throughout this article we use capital letters for vertices/variables, boldface capitals to indicate sets, and calligraphic letters to indicate graphs or distributions.

A *directed graph* (DG) $\mathcal{G}$ is an ordered pair $\langle \mathbf{V}, \mathbf{E} \rangle$, where $\mathbf{V}$ is a set of vertices (nodes), and $\mathbf{E}$ is a set of directed edges (arcs) between vertices. Two nodes in $\mathcal{G}$ are *adjacent* if they are connected by an edge, two edges are *adjacent* if they share a node. A *path* in the graph $\mathcal{G}$ is a sequence of adjacent edges where each consecutive pair along the path is adjacent in $\mathcal{G}$ and each node occurs at most once, or just a single node (a *trivial* path). A *directed path* $X_0 \longrightarrow X_1 \longrightarrow .. \longrightarrow X_k$ is a path where each pair of consecutive nodes is connected by an arc $X_i \longrightarrow X_{i+1}$ in $\mathcal{G}$. A *cycle* is a directed path $X_0 \longrightarrow .. \longrightarrow X_k$ together with an edge $X_k \longrightarrow X_0$. A directed graph with no cycles is called a *directed acyclic graph* (DAG). If $X \longrightarrow Y$ in $\mathcal{G}$ then X is called a *parent* of Y, and Y a *child* of X. Similarly, if there is a directed path from X to Y in $\mathcal{G}$ then X is an *ancestor* of Y, and Y a *descendant* of X. We use $pa_{\mathcal{G}}(X)$ to denote the set of parents of X in graph $\mathcal{G}$. Idem $ch_{\mathcal{G}}(X)$, $an_{\mathcal{G}}(X)$ and $de_{\mathcal{G}}(X)$ for the sets of children, ancestors, and descendants of X in $\mathcal{G}$, with natural extensions to sets, e.g. $pa_{\mathcal{G}}(\mathbf{X}) : \{V : \exists X \in \mathbf{X}, V \in pa_{\mathcal{G}}(X)\}$. A node Z is a *collider* on a path $\langle .., X, Z, Y, .. \rangle$ if the subpath is of the form $X \longrightarrow Z \longleftarrow Y$, otherwise it is a *noncollider*. A triple of nodes $\langle .., X, Z, Y, .. \rangle$ on a path is said to be *unshielded* if X and Y are not adjacent in $\mathcal{G}$. An unshielded collider $X \longrightarrow Z \longleftarrow Y$ is known as a *v-structure*.

A *DG model* is an ordered pair $\langle \mathcal{G}, \mathcal{P} \rangle$ where $\mathcal{G}$ is a (cyclic or acyclic) directed graph and $\mathcal{P}$ is a probability distribution over the vertices (variables) in $\mathcal{G}$. The *global directed Markov property* links the structure of the graph $\mathcal{G}$ to probabilistic independences in $\mathcal{P}$ via the d-separation criterion: for disjoint sets of vertices $\mathbf{X}, \mathbf{Y}, \mathbf{Z}$ in a graph $\mathcal{G}$, $\mathbf{X}$ is *d-connected* to $\mathbf{Y}$ given $\mathbf{Z}$ iff there is an $X \in \mathbf{X}$ and $Y \in \mathbf{Y}$ such that there is a path π between X and Y on which every noncollider is not in $\mathbf{Z}$, and every collider on π is an ancestor of $\mathbf{Z}$; otherwise $\mathbf{X}$ and $\mathbf{Y}$ are said to be *d-separated* given $\mathbf{Z}$. Two graphs $\mathcal{G}_1$ and $\mathcal{G}_2$ are said to be *d-separation equivalent* iff every d-separation in $\mathcal{G}_1$ also holds in $\mathcal{G}_2$ and v.v. For more details on graphical causal models, see (Koller and

Friedman, 2009; Spirtes et al., 2000; Pearl, 2009; Bongers et al., 2021). In section 3 in the supplement, we provide more details on Markov properties in structural causal models, and describe some concrete classes of models for which the *d*-separation criterion applies.

2.2 FEATURES OF CYCLIC GRAPHS

Next we will introduce a few properties and definitions that are specific to directed graphs with cycles.

Definition 1 *In a directed graph $\mathcal{G}$ over set of vertices* $\mathbf{V}$*, a subset* $\mathbf{S} \subseteq \mathbf{V}$ *is a **strongly connected component (SCC)** of $\mathcal{G}$ iff* $\mathbf{S}$ *is a maximal set of vertices where every vertex is reachable via a directed path in $\mathcal{G}$ from every other vertex in* $\mathbf{S}$.

In cyclic graphs the presence of arcs into directed cycles can create dependencies that behave like additional induced edges:

Definition 2 *In a graph $\mathcal{G}$, two nodes A and B are said to be **virtually adjacent** iff there is no edge between A and B in $\mathcal{G}$, but A and B have a common child C which is an ancestor of A or B.*

Two nodes connected by a virtual edge cannot be *d*-separated by any set of nodes, and therefore appear like they are connected by an edge. In (Richardson, 1997) virtual edges were also called *p(seudo)-adjacent*.

These induced virtual edges can also be part of paths we have to consider, giving rise to the generalized concept of an itinerary:

Definition 3 *In a graph $\mathcal{G}$, a sequence of vertices $\langle X_0, ..., X_{n+1} \rangle$ where all neighbouring nodes in the sequence are (virtually) adjacent in the graph is said to be an **itinerary**. If none of the nodes on the itinerary are (virtually) adjacent to each other except for the ones that occur consecutively on it then the itinerary is said to be **uncovered**, otherwise it is said to be **covered**.*

Virtual edges can also appear in regular (non)collider triples, leading to the generalized notion of (non)conductors:

Definition 4 *In a graph $\mathcal{G}$, a triple $\langle A, B, C \rangle$ forms a **conductor** if $\langle A, B, C \rangle$ is an itinerary, and B is an ancestor of A and/or C. If $\langle A, B, C \rangle$ is an itinerary, but B is NOT an ancestor of A or C, then $\langle A, B, C \rangle$ is a **nonconductor**. A (non)conductor $\langle A, B, C \rangle$ is said to be **unshielded** if A and C are not (virtually) adjacent, otherwise it is **shielded**.*

In some case we can actually detect the presence of some induced edge, although we can never be sure which one:

Definition 5 *In a graph $\mathcal{G}$ a nonconductor triple $\langle A, B, C \rangle$ is a **perfect nonconductor** if B is also a descendant of a common child of A and C. If not, then $\langle A, B, C \rangle$ is an **imperfect nonconductor**.*

Key notion here is that for unshielded perfect nonconductors conditioning on a set that includes B *always* creates a dependence between A and C, whereas unshielded imperfect nonconductors do create a dependence when conditioning on B, but not for *every* set containing B. This is impossible in acyclic graphs and is therefore a hallmark for the presence of cycles. See the two virtual *v*-structures in Figure 1 for an example.

Finally, as pièce de résistance, we have some patterns that introduce a nonlocality aspect:

Definition 6 *If $\langle X_0, ..., X_{n+1} \rangle$ is a sequence of vertices such that, each consecutive triple along the (uncovered) itinerary is a conductor, all nodes $\{X_1, .., X_n\}$ are ancestors of each other, but not ancestors of either X_0 or X_{n+1}, then the triples $\langle X_0, X_1, X_2 \rangle$ and $\langle X_{n-1}, X_n, X_{n+1} \rangle$ are **mutually exclusive (m.e.) conductors w.r.t. an (uncovered) itinerary**.*

An example is depicted in Figure 2. As a result, graphs that have identical *d*-separation relations locally everywhere in the graph can still differ regarding a *d*-separation between nodes that are arbitrarily far apart in the graph (something that is impossible in the acyclic case).

2.3 THE CYCLIC EQUIVALENCE THEOREM

With the features introduced in the previous section Richardson (1997) established the following characterization:

Cyclic Equivalence Theorem (CET): Two directed graphs $\mathcal{G}_1$ and $\mathcal{G}_2$ over vertices $\mathbf{V}$ are *d*-separation equivalent iff

(i) they have the same (virtual) adjacencies,

(ii).a they have the same unshielded conductors,

(ii).b they have the same unshielded perfect nonconductors,

(iii) two triples $\langle A, B, C \rangle$ and $\langle X, Y, Z \rangle$ are mutually exclusive conductors on some uncovered itinerary $P = \langle A, B, C, .., X, Y, Z \rangle$ in $\mathcal{G}_1$ iff they are also m.e. conductors on some uncovered itinerary in $\mathcal{G}_2$,

(iv) if $\langle A, X, B \rangle$ and $\langle A, Y, B \rangle$ are unshielded imperfect nonconductors in $\mathcal{G}_1$ and $\mathcal{G}_2$, then X is an ancestor of Y in $\mathcal{G}_1$ iff X is an ancestor of Y in $\mathcal{G}_2$,

(v) if $\langle A, B, C \rangle$ and $\langle X, Y, Z \rangle$ are m.e. conductors on an uncovered itinerary $P = \langle A, B, C, .., X, Y, Z \rangle$, and $\langle A, M, Z \rangle$ is an unshielded imperfect nonconductor (in $\mathcal{G}_1$ and $\mathcal{G}_2$), then M is a descendant of B in $\mathcal{G}_1$ iff M is a descendant of B in $\mathcal{G}_2$.

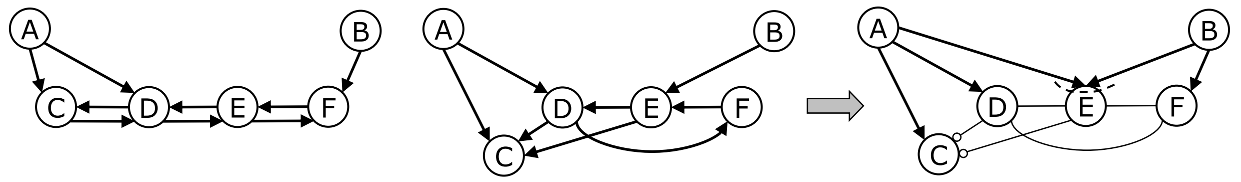

Figure 2: Two Markov equivalent graphs (left) with $\langle A, D, F\rangle$ and $\langle D, F, B\rangle$ a pair of m.e. conductors on uncovered itinerary $\langle A, D, F, B\rangle$; (right) corresponding (maximally informative) CPAG

2.4 CYCLIC PAGS

To characterize the (d-separation) Markov equivalence class of a cyclic directed graph $\mathcal{G}$, denoted $MEC(\mathcal{G})$, Richardson (1996c) described an algorithm that created a set of exhaustive lists of instances in the graph matching one of the individual rules in the CET, above. Establishing Markov equivalence then boils down to comparing the lists constructed for each.

Later on, Richardson (1996b) introduced a more intuitive graphical representation in the form of a (cyclic) partial ancestral graph that also captured enough elements to uniquely identify the equivalence class of a directed graph:

Definition 7 *A graph $\mathcal{P}$ is a* ***partial ancestral graph (PAG)*** *for directed (a)cyclic graph $\mathcal{G}$ with vertex set $\mathbf{V}$, iff*

(i) there is an edge between vertices A and B iff A and B are d-connected given any subset $\mathbf{W} \subseteq \mathbf{V} \setminus \{A, B\}$,

(ii) If $A \longrightarrow\!\!\ast\, B$ is in $\mathcal{P}$, then in every graph in $MEC(\mathcal{G})$, A is ancestor of B

(iii) If $A \ast\!\!\rightarrow B$ is in $\mathcal{P}$, then in every graph in $MEC(\mathcal{G})$, B is NOT an ancestor of A

(iv) if $A \ast\!\!-\!\underline{\ast\, B\, \ast}\!-\!\!\ast\, C$ in $\mathcal{P}$, then B is ancestor of A and/or C in every $\mathcal{G}' \in MEC(\mathcal{G})$,

(v) if $A \longrightarrow \underset{- - -}{B} \longleftarrow C$ in $\mathcal{P}$, then B is NOT a descendant of a common child of A and C in every $\mathcal{G}' \in MEC(\mathcal{G})$,

(vi) any remaining edge mark not oriented in the above ways obtains a circle mark $\circ\!\!-\!\!\ast$ in $\mathcal{P}$.

We use the term ***cyclic PAG (CPAG)*** *of a graph $\mathcal{G}$ to denote a PAG $\mathcal{P}$ that captures invariant ancestral relations shared by all and only the collection of graphs $\{\mathcal{G}'\}$ in the Markov equivalence class of $\mathcal{G}$.*

In these rules the asterisk $\ast\!\!-$ mark on an edge is used as a meta symbol that represents any of the other marks $\{-, >, \circ\}$. The solid underlining in rule (iv), indicating that the middle node is *not* a collider between the other two, is superfluous and therefore often omitted from the graph $\mathcal{P}$. The dashed underlining in rule (v), however, *is* essential, and unique to cyclic graphs, and appears in the virtual v-structures introduced in §3.1. See Figure 2 for an example CPAG.

The CPAG has the same purpose and interpretation as the familiar PAG output by the well-known FCI algorithm (Spirtes et al. (2000); Zhang (2008)), including circle marks $X \circ\!\!-\!\!\ast\, Y$ from rule (vi) to explicitly denote 'not determined'. This can be either because the implied ancestral relation is not invariant between all members in the Markov equivalence class of $\mathcal{G}$, i.e. there are some graphs where X is an ancestor of Y and some where it is not ('can't know'), or because the relation *is* invariant but we have not determined what it is yet ('don't know'). As a result, a graph $\mathcal{G}$ can correspond to different CPAGs $\mathcal{P}$ that differ in level of completeness. In this paper we are not concerned with obtaining the (unique) *maximally informative* CPAG, but instead settle for any *Markov complete* CPAG that represents a unique (d-separation) Markov equivalence class.

2.5 CPAG-FROM-GRAPH ALGORITHM

Using the CPAG definition above we now describe an algorithm by Richardson (1996a) that takes as input a (possibly cyclic) directed graph $\mathcal{G}$ and outputs a CPAG $\mathcal{P}$ such that two graphs $\mathcal{G}_1$ and $\mathcal{G}_2$ are Markov equivalent iff the algorithm outputs the same CPAG for both. In other words, the algorithm is *d-separation complete*.

(a) form the complete undirected graph $\mathcal{P}$ with all circle edges $\circ\!\!-\!\!\circ$, and then for every edge $A \circ\!\!-\!\!\circ B$ in $\mathcal{P}$, if A is d-separated from B given $\mathbf{C} = An(\{A, B\}) \setminus \{A, B\}$ then remove edge $A \circ\!\!-\!\!\circ B$ from $\mathcal{P}$ and record $\mathbf{C}$ in $Sepset(A, B)$ and $Sepset(B, A)$,

(b) for each unshielded triple $A \ast\!\!-\!\!\ast B \ast\!\!-\!\!\ast C$ in $\mathcal{P}$, orient $A \longrightarrow B \longleftarrow C$ if $B \notin Sepset(A, C)$,

(c) for each triple $\langle A, X, Y\rangle$ such that $X \ast\!\!-\!\!\ast Y$ in $\mathcal{P}$, A is not adjacent to X or Y in $\mathcal{P}$, $X \notin Sepset(A, Y)$, then if A and X are d-connected given $Sepset(A, Y)$, then orient $X \longleftarrow Y$,

(d) for each unshielded triple $A \longrightarrow B \longleftarrow C$ in $\mathcal{P}$, if A and C are d-separated given a set $\mathbf{R}$, then orient $A \longrightarrow \underset{- - -}{B} \longleftarrow C$ in $\mathcal{P}$ and record $\mathbf{R}$ in $SupSepset\langle A, B, C\rangle$ (and $SupSepset\langle C, B, A\rangle$),

(e) for each quadruple $\langle A, B, C, D\rangle$, if $A \longrightarrow \underset{- - -}{B} \longleftarrow C$ in $\mathcal{P}$, $A \longrightarrow D \longleftarrow C$ or $A \longrightarrow \underset{- - -}{D} \longleftarrow C$ in $\mathcal{P}$, B and D are adjacent in $\mathcal{P}$, then if $D \in SupSepset\langle A,B,C\rangle$ then orient $B \ast\!\!-\!\!\!- D$, otherwise orient $B \longrightarrow D$ in $\mathcal{P}$,

(f) for each quadruple $\langle A,B,C,D\rangle$, if $A \longrightarrow \underset{- - - -}{B} \longleftarrow C$ in $\mathcal{P}$, and D is not adjacent to both A and C in $\mathcal{P}$, then if A and C are d-connected given $SupSepset\langle A,B,C\rangle \cup \{D\}$, then orient $B \longrightarrow D$.

The algorithm has complexity $O(N^7)$, and is d-separation complete:

Theorem 2 in (Richardson, 1996a): For two graphs $\mathcal{G}_1$ and $\mathcal{G}_2$, the CPAG-from-Graph algorithm outputs corresponding CPAGs $\mathcal{P}_1$ and $\mathcal{P}_2$ that are identical iff $\mathcal{G}_1$ and $\mathcal{G}_2$ are d-separation equivalent.

Actually, the theorem was formulated for the CCD algorithm (Richardson, 1996b) for obtaining a CPAG from (oracle) independence information, but the two are so similar that the proof automatically carries over to the CPAG-from-Graph algorithm. The algorithm is an improvement by a factor $O(N^2)$ on the earlier list-based Cyclic Classification algorithm in (Richardson, 1996c, §5.4).

3 AN ANCESTRAL PERSPECTIVE ON THE CET

On reflection of the characterization of Markov equivalence between cyclic graphs obtained, one may note that the rather daunting definitions and terminology in the CET seem to contrast quite sharply with the apparent simplicity of the actual invariant features contained in the CPAG. At the same time complicated again by the fact that some of these 'invariant features' like edges in the CPAG are not actually invariant in the underlying graph at all.

Furthermore, there is no clear match from some rules in the CET to specific invariant features in the CPAG. In particular the 'mutually exclusive conductors on an uncovered itinerary'[1] in rule CET-(iii) are never explicitly recorded, even though they can of course be inferred from the CPAG afterwards.

A natural question, inspired by the familiar DAG-MAG-PAG triad for acyclic graphs, would be whether it might make sense to also consider an intermediate ancestral stage for cyclic graphs.

In this section we answer that question with an emphatic: yes! We first introduce the CMAG as the cyclic analogue to the (acyclic) maximal ancestral graph (Richardson and Spirtes, 2002), and rephrase the CET in terms of ancestral graphs. This results in a simplified set of rules that each are in direct correspondence with invariant features in the CPAG. In the next section we will show that this approach also leads to an efficient procedure to establish Markov equivalence that no longer needs to rely on d-separation tests.

[1]Actually this term is a bit of a misnomer, as the two conductors need not be mutually exclusive when there is an induced virtual edge along the uncovered itinerary connecting the two.

3.1 INTRODUCING THE CMAG

In keeping with the spirit of regular (acyclic) maximal ancestral graphs, we will conceptually define a cyclic MAG as:

Definition 8 *The* ***cyclic maximal ancestral graph (CMAG)*** *$\mathcal{M}$ corresponding to (cyclic) directed graph $\mathcal{G}$ over set of vertices $\mathbf{V}$ is a graph where:*

(i) there is an edge between every distinct pair of vertices $\{X,Y\}$ iff they cannot be d-separated by any subset of $\mathbf{V} \setminus \{X,Y\}$ in $\mathcal{G}$,

(ii) there is a tail mark $X \!-\!\!-\!\!\ast\, Y$ at vertex X on the edge to Y iff there exists a directed path from X to Y in $\mathcal{G}$, otherwise there is an arrowhead mark $X \leftarrow\!\!\ast\, Y$,

(iii) every v-structure $X \longrightarrow Z \longleftarrow Y$ in $\mathcal{M}$ where Z is not a descendant of a common child of X and Y in $\mathcal{G}$ obtains a dashed underline $X \longrightarrow \underset{- - - -}{Z} \longleftarrow Y$.

The 'dashed-underlined' triples in a CMAG are referred to as ***virtual v-structures****.*

With this definition, a CPAG becomes a straightforward collection of invariant edges and edge marks (rather than 'ancestral relations') shared by all and only the CMAGs corresponding to graphs in the same Markov equivalence class.

The 'virtual' in the dashed-underlined *v*-structures from rule (iii) emphasises that they appear like regular *v*-structures in the CMAG, but look and behave differently in the underlying directed graph $\mathcal{G}$. They are a direct consequence of rule (v) in Def. 7, and correspond to unshielded imperfect nonconductors in $\mathcal{G}$, that are unique to cyclic graphs.

3.2 VIRTUAL COLLIDER TRIPLES

Having brought out the CMAG we can make a straightforward mapping from elements in the CET to their ancestral counterpart: (virtual) adjacencies become edges, itineraries become paths, unshielded conductors become standard unshielded noncolliders, unshielded (perfect) nonconductors become v-structures, and unshielded imperfect nonconductors become virtual v-strucutures.

That only leaves the 'mutually exclusive conductors w.r.t. an uncovered itinerary'. For that we note that these only appear in the CPAG as the invariant arcs into a cycle, oriented in step (c) of the CPAG-from-Graph algorithm. In other words, from an ancestral perspective it is not about the conductor triples at the beginning and end of the uncovered itinerary, but only about the first and last edge along the corresponding path in the CMAG.

This brings us to the following definition:

Definition 9 *In a CMAG $\mathcal{M}$ for directed graph $\mathcal{G}$, a quadruple of distinct nodes $\langle X, Z, Z', Y\rangle$ is a **u-structure** if $X \longrightarrow Z$ and $Z' \longleftarrow Y$ are in $\mathcal{M}$, $Z' \in SCC_{\mathcal{G}}(Z)$, and there is an uncovered path $\langle X, Z, .., Z', Y\rangle$ in $\mathcal{M}$ where all intermediate nodes are also in $SCC_{\mathcal{G}}(Z)$.*

The term *u*-structure reflects the fact that it is similar to a *v*-structure, but with the central collider node replaced by an uncovered path through a strongly connected component. Note that SCCs in $\mathcal{G}$ correspond to nodes in (maximal) connected undirected subgraphs in $\mathcal{M}$, as each node in an SCC is ancestor of all other nodes in the same SCC.

There is a straightforward connection between *u*-structures and the 'm.e. conductors w.r.t. an uncovered itinerary' from Definition 6:

Lemma 1 *For a directed graph $\mathcal{G}$ and corresponding CMAG $\mathcal{M}$, there is a u-structure $\langle X, Z, Z', Y\rangle$ in $\mathcal{M}$ iff there is an uncovered itinerary $\pi = \langle X, Z, U, .., U', Z', Y\rangle$ in $\mathcal{G}$, possibly with $Z = U'$ or $U = U'$, where $\langle X, Z, U\rangle$ and $\langle U', Z', Y\rangle$ are a pair of m.e. conductors w.r.t. the uncovered itinerary π in $\mathcal{G}$.*

(For proof details for this and other results in the rest of this article, see supplement.)

Crucially, in the CMAG or CPAG we do not actually record the *u*-structure explicitly. In fact, the only elements of a *u*-structure that need to be oriented in the CPAG are the first and last edge *into* the strongly connected component (cf. step (c) of the CPAG-from-Graph algorithm, §2.5).

As a result, we do not have to identify the full quadruple $\langle X, Z, Z', Y\rangle$ of each *u*-structure, but only if an edge $X - Z$ is part of *some* u-structure pattern. For that, we can rely on the following result:

Lemma 2 *In a CMAG $\mathcal{M}$, a pair of nodes $\langle X, Z\rangle$ is part of a u-structure $\langle X, Z, Z', Y\rangle$ with a node $Y \in \mathbf{Y} \subseteq pa(SCC(Z)) \smallsetminus adj(\{X, Z\})$, iff $X \in pa(Z)$, and X and Y are connected in the undirected subgraph over $((SCC(Z) \smallsetminus adj(X)) \cup \{X, Z\} \cup \mathbf{Y}$.*

This significantly reduces the complexity of establishing Markov equivalence later on, as it means we only need to search over triples rather than quadruples in the CMAG. More importantly, however, is that this result suggests that virtual *v*-structures and *u*-structures can actually be seen as two manifestations of the same invariant element, which in turn will greatly simplify the CET.

Definition 10 *In a CMAG $\mathcal{M}$, a triple of distinct nodes $\langle X, Z, Y\rangle$ is a **virtual collider triple** iff $\langle X, Z, Y\rangle$ is a virtual v-structure, or there is some $Z' \in SCC(Z)$, such that either $\langle X, Z, Z', Y\rangle$ or $\langle X, Z', Z, Y\rangle$ is a u-structure.*

Intuitively, a virtual collider triple $\langle X, Z, Y\rangle$ implies that X and Y are connected by an uncovered itinerary via nodes in $SCC(Z)$ that *identifiably* contains one or more virtual edges. The strongly connected component of Z fulfils the role of collider in $X \longrightarrow SCC(Z) \longleftarrow Y$, and the *virtual* emphasises there is no 'real' collider triple $X \longrightarrow Z \longleftarrow Y$ in the underlying directed graph.

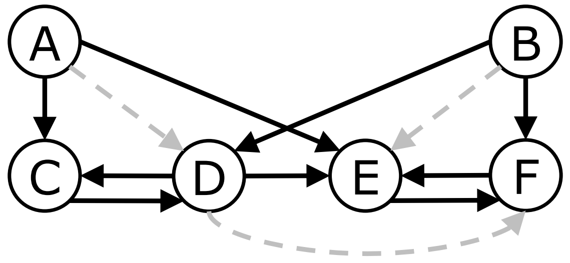

Figure 3: Example CET orientation rule (iv) on virtual collider triples $\langle A, D, B\rangle$ and $\langle A, E, B\rangle$ for invariant edge $D \longrightarrow E$, with virtual edges as dashed grey arcs.

3.3 A NEW CET

We are now ready to restate the Cyclic Equivalence Theorem in terms of CMAGs:

Theorem 1 *Two CMAGs $\mathcal{M}_1$ and $\mathcal{M}_2$ corresponding to cyclic directed graphs $\mathcal{G}_1$ resp. $\mathcal{G}_2$ are Markov equivalent iff*

(i) they have the same skeleton,

(ii) they have the same v-structures,

(iii) they have the same virtual collider triples,

(iv) if $\langle A, B, C\rangle$ and $\langle A, D, C\rangle$ are virtual collider triples, then B is an ancestor of D in $\mathcal{M}_1$ iff B is an ancestor of D in $\mathcal{M}_2$.

Each rule in this ancestral CET can be linked directly to specific invariant elements in the CPAG: rule (i) to the edges in the CPAG, (ii) to all *v*-structures, (iii) to remaining invariant arcs into strongly connected components (incl. under-dashed marks for virtual *v*-structures), and (iv) to invariant edges within or between (identifiable) cycle components.

Comparing to the original CET in §2.3, we can see that the ancestral formulation greatly simplifies the Markov equivalence characterization, leading to two fewer rules and only requiring (collider) triples.

An interesting observation is that in the acyclic case going from DAGs to MAGs (to allow for unobserved confounders) implied going from '(unshielded) collider triples' (v-structures) to 'collider triples with order' in the characterization of Markov equivalence between graphs (Ali et al., 2009; Claassen and Bucur, 2022). Given that analogy we conjecture that for the cyclic case allowing for latent confounders can similarly be accomplished by extending to '(virtual) collider triples with order'.

4 ESTABLISHING MARKOV EQUIVALENCE FOR CYCLIC GRAPHS

We now show that with the intermediate CMAG representation we can derive a consistent CPAG that uniquely defines the equivalence class of a cyclic directed graph without the need for any d-separation tests. The resulting algorithm is extremely fast, and allows to determine Markov equivalence between graphs by directly comparing the output CPAGs.

4.1 OBTAINING THE CMAG

To capture the first rule of the new CET, we need to obtain the skeleton of the CPAG. To avoid the d-separation tests in step (a) of the CPAG-from-Graph algorithm in §2.5, we can use the following result:

Lemma 3 *In a CMAG $\mathcal{M}$ corresponding to directed graph $\mathcal{G}$, two variables X and Y are adjacent, iff X and Y are (virtually) adjacent in $\mathcal{G}$.*

It implies we can read off the CMAG skeleton directly from the graph $\mathcal{G}$, by starting from the skeleton of $\mathcal{G}$, and adding an edge between X and Y for every v-structure $X \longrightarrow Z \longleftarrow Y$ in $\mathcal{G}$ with $Y \in SCC(Z)$.

It does mean that we first need to partition the vertices in the graph into the set of strongly connected components. This can be achieved in time linear in the number of vertices and edges $O(Nd)$ using e.g. Tarjan's algorithm (Tarjan, 1972)[2].

Subsequent orientations of edges in $\mathcal{M}$ follow orientations in $\mathcal{G}$, where edges between nodes in the same SCC become undirected edges, signifying they are all ancestor of each other. Induced edges between nodes in the same cycle also become undirected, and induced edges by a triple $X \longrightarrow Z \longleftarrow Y$ in $\mathcal{G}$ with $X \notin SCC(Z)$ become $X \longrightarrow Y$.

Alternatively, we can process each node X in $\mathcal{G}$ in turn, and draw undirected edges between all of its parents in the same cycle as X (incl. X) in $\mathcal{M}$, and add arcs from all remaining parents into the first set of parents (again incl. X), which is what we do in Algorithm 1, below.

Algorithm 1 Cyclic-Graph-to-CMAG

```
Input: directed cyclic graph $\mathcal{G}$ over nodes $\mathbf{V}$
Output: CMAG $\mathcal{M}$, $SCC$s,
$SCC \leftarrow Get\_StronglyConnComps(\mathcal{G})$
part 1: CMAG rules (i) + (ii)
for all $X \in \mathbf{V}$ do
    $\mathbf{Z} \leftarrow pa_{\mathcal{G}}(X)$
    $\mathbf{Z}_{cyc} \leftarrow \mathbf{Z} \cap SCC(X)$
    $\mathbf{Z}_{acy} \leftarrow \mathbf{Z} \setminus \mathbf{Z}_{cyc}$
    add all arcs $\mathbf{Z}_{acy} \longrightarrow \mathbf{Z}_{cyc} \cup \{X\}$ to $\mathcal{M}$
    add all undirected edges $\mathbf{Z}_{cyc} \text{---} \mathbf{Z}_{cyc} \cup \{X\}$ to $\mathcal{M}$
end for
part 2: CMAG rule (iii)
for all $X \in \mathbf{V} : |SCC(X)| \geq 2$ do
    $\mathbf{Z} \leftarrow pa_{\mathcal{M}}(X)$
    for all non-adjacent pairs $\{Z_i, Z_j\} \subseteq \mathbf{Z}$ do
        if $\{Z_i, Z_j\} \nsubseteq adj_{\mathcal{G}}(X)$ then
            if $X \notin de_{\mathcal{G}}(ch_{\mathcal{G}}(Z_i) \cap ch_{\mathcal{G}}(Z_j))$ then
                mark virtual v-structure $\langle Z_i, X, Z_j \rangle$ in $\mathcal{M}$
    end for
end for
```

The second part of Algorithm 1 simply involves checking all v-structures in $\mathcal{M}$ with central collider node in a non-trivial SCC, and with at least one virtual edge in $\mathcal{G}$. Here we use the matrix of ancestral relations, constructed when identifying the SCCs at the start of the algorithm, to reduce the 'descendant of' check in the second 'if'-clause to constant time per node.

4.2 CONSTRUCTING THE CPAG

Before we can go on to construct the CPAG from the CMAG $\mathcal{M}$ obtained above, we still need to recognise the virtual collider triples corresponding to so-called u-structures. These are not marked explicitly in the CMAG (contrary to virtual v-structures), but they *are* needed to orient certain invariant edges in the CPAG corresponding to rules (iii) and (iv) in Theorem 1. Fortunately, for that we can rely on Lemma 2, where the fact that we only need to consider straightforward 'connected undirected subgraphs' means the complexity of this step scales linearly with the number of edges in the subgraph.

It also means that, in the construction of the CPAG, to cover invariant arcs from u-structures, we only need to consider edges $X \longrightarrow Z$ in $\mathcal{M}$ that are not yet oriented in $\mathcal{P}$, and where Z is part of a nontrivial SCC (size $|SCC(Z)| \geq 2$), and the $\mathbf{Y}$ in Lemma 2 are all other parents of $SCC(Z)$ that are not adjacent to X and/or Z in $\mathcal{M}$. Note that the arcs oriented thusly were previously captured by the exhaustive search in step (c) of the CPAG-from-Graph algorithm in

[2]Actually, we use a modified version of Tarjan's algorithm that also tracks ancestral relations in one go. For details on this and all other algorithms used in the paper, see source code available at `https://github.com/tomc-ghub/CET_uai2023`

section 2.5.

Finally, note that rule (iv) of our new CET in Theorem 1 applies to all virtual collider triples, but that in the construction of the CPAG, similar to the original CPAG-from-Graph algorithm, we only need to consider cases where at least one of them is a virtual *v*-structure, as the case for two *u*-structures will be superfluous as follows from the original CET in §2.3.

We can now bring these steps together in Algorithm 2.

Algorithm 2 Graph-to-CPAG

```
Input: directed cyclic graph $\mathcal{G}$ over nodes $\mathbf{V}$,
Output: CPAG $\mathcal{P}$,
$(\mathcal{M}, SCC) \leftarrow$ Cyclic-Graph-to-CMAG($\mathcal{G}$)
part 1: new-CET rules (i)-(iii)
$\mathcal{P} \leftarrow$ skeleton of $\mathcal{M}$ with all $\circ\!\!-\!\!\circ$ edges
$\mathcal{P} \leftarrow$ copy all (virtual) $v$-structures from $\mathcal{M}$
for all $X \circ\!\!-\!\!\circ Z$ in $\mathcal{P}$, $X \longrightarrow Z$ in $\mathcal{M}$, $|SCC(Z)| \geq 2$ do
  if $\exists \langle X, Z, Z', Y \rangle$ as $u$-structure in $\mathcal{M}$ then
    orient $X \longrightarrow Z$ in $\mathcal{P}$          {Lemma 2}
end for
part 2: new-CET rule (iv)
for all virtual $v$-structures $\langle X, Z, Y \rangle$ in $\mathcal{P}$ do
  for all not fully oriented edges $Z *\!\!-\!\!* W$ in $\mathcal{P}$ do
    if $\langle X, W, Y \rangle$ is virtual collider triple then
      copy edge $Z *\!\!-\!\!* W$ from $\mathcal{M}$ to $\mathcal{P}$
    end if
  end for
end for
```

In practice we already copy invariant features to the CPAG while constructing the CMAG to improve efficiency. Note that the final output CPAG is *d*-separation complete, but *not* guaranteed to be identical to the CPAG from the original CPAG-from-Graph algorithm. This is because step (c) there contained an exhaustive search that also oriented certain arcs that are sound but not needed for the CET, but could also be obtained from subsequent implied orientation rules, similar to the PC/FCI algorithm. Similarly, the new algorithm can orient some edges in the last step that are not guaranteed to be found by the previous version. Therefore the CPAGs from the two algorithms cannot be compared directly against each other to establish Markov equivalence. However the main result remains the same:

Theorem 2 *For two different cyclic directed graphs $\mathcal{G}_1$ and $\mathcal{G}_2$, let $\mathcal{P}_1$ and $\mathcal{P}_2$ be the corresponding CPAGs output by algorithm 2. Then $\mathcal{G}_1$ is Markov equivalent to $\mathcal{G}_2$ iff $\mathcal{P}_1 = \mathcal{P}_2$.*

4.3 COMPUTATIONAL COMPLEXITY

The scaling behaviour of Algorithm 2 depends primarily on the number of vertices N and average node degree d corresponding to $N * d$ edges in the graph.

The first part of algorithm 1 requires order $O(N + N * d)$ steps to find the strongly connected components, followed by a loop over N vertices comparing d^2 parents, so overall $O(N * d^2)$. Similarly, the second part considers d^2 parents for N nodes for $O(N * d^2)$ (provided the $Get\_StronglyConnComps$ step also tracks the ancestral matrix for constant-time descendant checks). Subsequent steps initializing the skeleton and (virtual) v-structures are also $O(N * d^2)$. Next, for the *u*-structures we may need to loop over $O(N * d)$ edges and establish connectedness in a subgraph over at most N nodes, which can be done in order $O(N * d)$ steps (similar to the SCC procedure) leading to overall $O(N^2 * d^2)$. Finally we need to loop over $N * d^2$ virtual *v*-structures, considering links to d other edges, while testing for connectedness order $O(N * d)$, giving a total of $O(N^2 * d^3)$.

So overall worst case complexity scales with $O(N^2 * d^3)$, or $O(N^5)$ for arbitrary density, which is a significant improvement over the $O(N^7)$ achieved by the current state-of-the-art CPAG-from-Graph algorithm.

In practice, even for large graphs there is typically only a relatively small number of cases to consider in the final steps, and so for both procedures the actual scaling behaviour is usually much better than this worst-case bound suggests, as evidenced by the next section.

5 EXPERIMENTAL EVALUATION

In order to evaluate the performance of the CPAG-from-graph procedure as a function of size and density of the graph we generate collections of random directed cyclic graphs and track both average and worst-case performance in terms of number of elementary operations and time.

Note that in generating the random cyclic graphs we introduced a few parameters to be able to tweak the number and type of cycles included, as for increasing size and density truly random cyclic graphs quickly tend to collapse into the 'one big cycle' type, avoiding most of the intricacies from CET rules (iv) and (v) that relate to invariate edges between cycles; see section 1.1 in the supplement for details.

5.1 SCALING BEHAVIOUR

Figure 4 shows the results for the two CPAG-from-graph procedures. As expected, the scaling behaviour of the new procedure in Algorithm 2 is much more benign. As a result, for graphs of $N = 200$ nodes with density $d = 3.0$, the latter requires only about 0.05 sec. on average to construct the CPAG, whereas the original version takes about 78 sec.: a speed-up by 3 orders of magnitude.

In the supplement we see that the original CPAG-from-Graph procedure spends the vast majority of its time in the

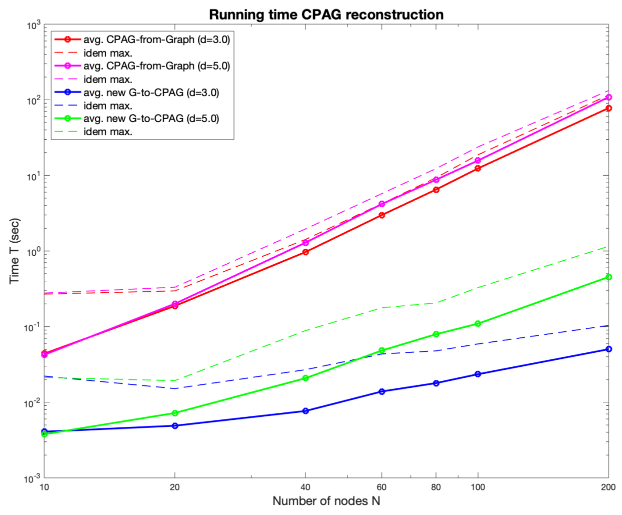

Figure 4: Log-log plot depicting scaling behaviour of original (red/magenta) and new CPAG algorithms (blue/green), as a function of size of the graph N, for two different densities $d \in \{3.0, 5.0\}$. Solid lines indicate average performance over 100 instances, dashed lines the worst case encountered.

expensive d-separation searches in stage (a) and (c), whereas for sparse graphs the new Graph-to-CPAG version spends roughly equal amounts in each phase. For denser graphs, the final stage in the latter starts to dominate, as expected from the complexity analysis in section 4.3.

Finally, note that for both algorithms there is not much difference between average and worst-case scaling behaviour in the collection of randomly sampled graphs (around $1.5 - 2.0$ times more expensive for both versions), and both stay well below their theoretical worst-case limits. The reason is that, in order to reach the dreaded 'worst case' scenario, the graphs require very specific configurations that are extremely unlikely to occur in truly random graphs. As a result, a reassuring message of Figure 4 is that in practice the challenge of handling even (very) large cyclic directed graphs is likely to remain feasible in practice, despite the quite imposing theoretical worst-case limit.

6 DISCUSSION

We presented a new, ancestral perspective on the Cyclic Equivalence Theorem for directed graphs that resulted in a fast and efficient procedure to obtain the CPAG from an arbitrary directed graph.

The resulting CPAGs can be compared directly to establish Markov equivalence between cyclic directed graphs, but so far we made no attempt to derive *all* invariant features shared by all (and only) the CMAGs in the same equivalence class. In other words, we did not yet aim for the *maximally informative* CPAG. As a result, not all identifiable cycles are guaranteed to appear in an easily recognisable form. Squeezing out all available information would likely entail a set of additional orientation propagation rules, similar to augmented FCI in (Zhang, 2008).

The obtained efficiency of the Graph-to-CPAG procedure in algorithm 2 also means it is fast enough to be a viable route for extending score-based greedy equivalence search algorithms like GES (Chickering, 2002) towards cyclic graphs, similar to recent extensions for acyclic graphs in the presence of confounders (Claassen and Bucur, 2022).

However, we consider the most promising aspect of our results the significantly reduced conceptual complexity provided by the ancestral perspective. The new ancestral CET is notably simpler than the original version, and suggests a natural extension to cyclic models with confounders, analogous to that for MAGs.

Finally, the CMAG under d-separation treats strongly connected components more similar to the nonlinear case under σ-separation (Mooij and Claassen, 2020), which suggests they may be merged to handle arbitrary cyclic relationships in the near future. We hope this may encourage researchers to renew work towards extending available constraint-based algorithms towards sound and complete causal discovery in the presence of confounders, cycles, and selection bias.

References

Ali, R. A., Richardson, T. S., and Spirtes, P. (2009). Markov equivalence for ancestral graphs. The Annals of Statistics, 37(5B):2808–2837.

Bongers, S., Forré, P., Peters, J., and Mooij, J. M. (2021). Foundations of structural causal models with cycles and latent variables. Annals of Statistics, 49(5):2885–2915.

Chickering, D. M. (2002). Optimal structure identification with greedy search. Journal of machine learning research, 3(Nov):507–554.

Claassen, T. and Bucur, I. G. (2022). Greedy equivalence search in the presence of latent confounders. In Uncertainty in Artificial Intelligence, pages 443–452. PMLR.

Forré, P. and Mooij, J. M. (2018). Constraint-based causal discovery for non-linear structural causal models with cycles and latent confounders. In Proceedings of the 34th Annual Conference on Uncertainty in Artificial Intelligence (UAI-18).

Hu, Z. and Evans, R. (2020). Faster algorithms for markov equivalence. In Conference on Uncertainty in Artificial Intelligence, pages 739–748. PMLR.

Hyttinen, A., Eberhardt, F., and Hoyer, P. (2012). Learning linear cyclic causal models with latent variables. Journal of Machine Learning Research, 13:3387–3439.

Koller, D. and Friedman, N. (2009). Probabilistic graphical models: principles and techniques. MIT press.

Lacerda, G., Spirtes, P., Ramsey, J., and Hoyer, P. O. (2008). Discovering cyclic causal models by independent components analysis. In Proceedings of the 24th Conference on Uncertainty in Artificial Intelligence (UAI-08).

Mooij, J. M. and Claassen, T. (2020). Constraint-based causal discovery using partial ancestral graphs in the presence of cycles. In Conference on Uncertainty in Artificial Intelligence, pages 1159–1168. PMLR.

Mooij, J. M. and Heskes, T. (2013). Cyclic causal discovery from continuous equilibrium data. In Nicholson, A. and Smyth, P., editors, Proceedings of the 29th Annual Conference on Uncertainty in Artificial Intelligence (UAI-13), pages 431–439. AUAI Press.

Mooij, J. M., Janzing, D., Heskes, T., and Schölkopf, B. (2011). On causal discovery with cyclic additive noise models. In Shawe-Taylor, J., Zemel, R., Bartlett, P., Pereira, F., and Weinberger, K., editors, Advances in Neural Information Processing Systems 24 (NIPS*2011), pages 639–647.

Pearl, J. (2009). Causality: Models, Reasoning and Inference. Cambridge University Press.

Richardson, T. (1996a). Discovering cyclic causal structure. Technical Report CMU-PHIL-68, Carnegie Mellon University.

Richardson, T. (1996b). A discovery algorithm for directed cyclic graphs. In Proceedings of the Twelfth international conference on Uncertainty in Artificial Intelligence (UAI-96), pages 454–461.

Richardson, T. (1996c). A polynomial-time algorithm for deciding Markov equivalence of directed cyclic graphical models. In Proceedings of the Twelfth international conference on Uncertainty in artificial intelligence, pages 462–469.

Richardson, T. (1997). A characterization of Markov equivalence for directed cyclic graphs. International Journal of Approximate Reasoning, 17(2-3):107–162.

Richardson, T. S. and Spirtes, P. (2002). Ancestral graph Markov models. The Annals of Statistics, 30(4):962–1030.

Rothenhäusler, D., Heinze, C., Peters, J., and Meinshausen, N. (2015). BACKSHIFT: Learning causal cyclic graphs from unknown shift interventions. In Advances in Neural Information Processing Systems 28 (NIPS 2015), pages 1513–1521.

Spirtes, P. (1994). Conditional independence in directed cyclic graphical models for feedback. Technical Report CMU-PHIL-54, Carnegie Mellon University.

Spirtes, P. (1995). Directed cyclic graphical representations of feedback models. In Proceedings of the Eleventh Conference on Uncertainty in Artificial Intelligence (UAI-95), pages 499–506.

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

Strobl, E. V. (2018). A constraint-based algorithm for causal discovery with cycles, latent variables and selection bias. International Journal of Data Science and Analytics, 8:33–56.

Tarjan, R. (1972). Depth-first search and linear graph algorithms. SIAM journal on computing, 1(2):146–160.

Wienöbst, M., Bannach, M., and Liśkiewicz, M. (2022). A new constructive criterion for markov equivalence of mags. In Uncertainty in Artificial Intelligence, pages 2107–2116. PMLR.

Zhang, J. (2008). On the completeness of orientation rules for causal discovery in the presence of latent confounders and selection bias. Artificial Intelligence, 172(16-17):1873–1896.