Local Causal Discovery for Statistically Efficient Causal Inference

Mátyás Schubert
University of Amsterdam

Tom Claassen
Radboud University Nijmegen

Sara Magliacane
Saarland University &
University of Amsterdam

Abstract

Causal discovery methods can identify valid adjustment sets for causal effect estimation for a pair of *target variables*, even when the underlying causal graph is unknown. Global causal discovery methods focus on learning the whole causal graph and therefore enable the recovery of *optimal adjustment sets*, i.e., sets with the lowest asymptotic variance, but they quickly become computationally prohibitive as the number of variables grows. Local causal discovery methods offer a more scalable alternative by focusing on the local neighborhood of the target variables, but are restricted to statistically suboptimal adjustment sets. In this work, we propose Local Optimal Adjustments Discovery (LOAD), a sound and complete causal discovery approach that combines the computational efficiency of local methods with the statistical optimality of global methods. First, LOAD identifies the causal relation between the targets and tests if the causal effect is identifiable by using only local information. If it is identifiable, it finds the possible descendants of the treatment and infers the optimal adjustment set as the parents of the outcome in a modified forbidden projection. Otherwise, it returns the locally valid parent adjustment sets. In our experiments on synthetic and realistic data LOAD outperforms global methods in scalability, while providing more accurate effect estimation than local methods.

1 INTRODUCTION

Estimating causal effects (Pearl, 2009) is an essential task in science and decision making. A popular method for causal effect estimation is covariate adjustment based on the underlying causal graph. However, often the causal graph is unknown. In this case we can learn it using causal discovery (Glymour et al., 2019). We can then use the learned graph to read off valid adjustment sets (Perković et al., 2018). Global causal discovery methods, i.e., algorithms that learn the full graph, are generally not scalable to large number of variables due to computational demand (Mokhtarian et al., 2021), making them inapplicable in many practical scenarios.

On the other hand, if we only care about estimating causal effects between a pair of target variables, then recovering the full causal graph over all variables might be unnecessary and inefficient. Under the assumption of causal sufficiency, i.e., no unobserved confounders or selection bias, local causal discovery methods estimate the causal effect between a pair of targets by discovering only local information about the causal graph around the treatment (Wang et al., 2014; Gupta et al., 2023) and identifying the parent adjustment set. Recent extensions consider also the causally insufficient case (Xie et al., 2024; Ling et al., 2025). Other approaches recover only coarse-grained information on ancestral relations that are enough to identify valid adjustment sets (Watson and Silva, 2022; Maasch et al., 2024).

While these methods can effectively reduce computational demands for finding valid adjustment sets, these sets can be suboptimal in terms of asymptotic variance (Henckel et al., 2022). Moreover, they require additional assumptions about ancestral relationships. Schubert et al. (2025) show that definite non-ancestors of the target variables are not required to discover the optimal adjustment set. They propose SNAP, which progressively identifies these definite non-ancestors and prunes them from the causal discovery process, improving the computational efficiency of the method. However, SNAP still requires discovering fine-grained causal relations over the remaining, potentially large, number of variables, making it potentially inefficient.

In this paper, we combine the advantages of global and local methods, and propose Local Optimal Adjustments Discovery (LOAD), a method to identify the optimal

Proceedings of the 29th International Conference on Artificial Intelligence and Statistics (AISTATS) 2026, Tangier, Morocco. PMLR: Volume 300.

adjustment set using only local information around variables. LOAD employs local causal discovery to cheaply identify information over the local neighborhoods of the target variables and their siblings. Then, using only this local information, it identifies the type of causal relation between the targets and determines whether the causal effect is identifiable. If it is identifiable, it then finds the possible descendants of the treatment and infers the optimal adjustment set as the parents of the outcome in a variant of the forbidden projection (Witte et al., 2020). Otherwise, it returns the locally valid parent adjustment sets (Maathuis et al., 2009). Our contributions are:

- We develop a method to determine the identifiability of a causal effect from local information.
- We propose Local Optimal Adjustments Discovery (LOAD), a sound and complete method to identify the optimal adjustment set, using only local information around variables.
- We evaluate LOAD on both simulated and realistic data, showing that it can recover high-quality adjustment sets with low computational costs.

2 PRELIMINARIES

We consider graphs $G = (\mathbf{V}, \mathbf{E})$ with nodes $\mathbf{V}$ and edges $\mathbf{E} \subseteq \mathbf{V} \times \mathbf{V}$. We denote undirected edges as $X - Y$ and directed edges as $X \to Y$. An undirected graph contains only undirected edges, while a directed graph contains only directed edges. A mixed graph can contain both directed and undirected edges.

If two nodes are connected by an edge, then we say that they are *adjacent*, and denote the set of nodes adjacent to a node X as $Adj_G(X)$. If two nodes are connected by an undirected edge $X - Y$ then we say that they are siblings and denote the set of siblings of X as $Sib_G(X)$. If $X \to Y$, we say that X is a parent of Y and Y is a child of X, and denote the set of parents and children of X as $Pa_G(X)$ and $Ch_G(X)$ respectively. In directed graphs, the Markov blanket $MB(X)$ of X consists of its parents, children and the parents of its children.

A path between two nodes X and Y is a sequence of distinct adjacent nodes starting with X and ending with Y. If every edge on a path between X and Y is directed towards Y then it is a directed path from X to Y. A directed acyclic graph (DAG) is a graph with only directed edges and no cycles, i.e., no directed paths from a node to itself. If there is a directed path from X to Y in a DAG G, then we say that X is an ancestor of Y and Y is a descendant of X, and we denote the set of ancestors and descendants of X as $An_G(X)$ and $De_G(X)$ respectively. By convention, we consider all nodes to be their own ancestors and descendants. We extend the definitions of siblings, parents, children, Markov blanket, ancestors, and descendants to sets of nodes by taking the union of the respective sets, e.g., the parents of variables $\mathbf{S}$ are $Pa_G(\mathbf{S}) = \cup_{X \in \mathbf{S}} Pa_G(X)$.

The data generating process of an observational distribution p over variables $\mathbf{V}$ can be described by a causal DAG D such that if a variable X has a direct causal effect on another variable Y then $X \to Y$. As standard in causal discovery, we assume that the distribution p is Markov and faithful to D (Spirtes et al., 2000), which implies that every conditional independence $X \perp\!\!\!\perp Y | \mathbf{S}$ in p is equivalent to a d-separation relation of the same form in D. Additionally, we assume causal sufficiency, i.e., no latent confounders or selection bias.

Constraint-based causal discovery methods leverage these assumptions to identify causal relations from conditional independence (CI) tests. In general, we cannot fully identify the true causal graph D from CI tests, as multiple DAGs imply the same set of conditional independences. This set of graphs is called the Markov equivalence class (MEC) of D and we can represent it by a mixed graph, the *completed partially directed acyclic graph (CPDAG)*. A CPDAG will have a directed edge $X \to Y$ if all DAGs in the MEC have this directed edge, otherwise if some DAGs have $X \to Y$ and others $Y \to X$, it will have an undirected edge $X - Y$.

We say that X is a possible ancestor of Y, denoted as $X \in PossAn_G(Y)$, and that Y is a possible descendant of X, denoted as $Y \in PossDe_G(X)$ in a CPDAG G when X is an ancestor of Y in at least one DAG in the MEC represented by G. Equivalently, X is a possible ancestor of Y, iff there exists a *possibly directed path* from X to Y, i.e., a path that does not contain directed edges from Y to X. If there is no such path, then X is a definite non-ancestor of Y in G. If X is an ancestor of Y in all DAGs in a MEC, then X is a definite ancestor Y. However, this does not necessarily mean that there is a directed path from X to Y in the corresponding completed partially directed acyclic graph (CPDAG) (Roumpelaki et al., 2016). If there is a directed path from X to Y in the CPDAG then we say that X is an explicit ancestor of Y. We denote the set of explicit ancestors of X as $ExplAn_G(X)$. Reversely, we define the explicit descendants of X as the nodes with a directed path starting from X in the CPDAG G.

If all DAGs in the MEC share a valid adjustment set, then we say that the causal effect is identifiable. Amenability provides a sufficient and necessary graphical condition for the causal effect of X on Y to be identifiable from the CPDAG G (Perkovic, 2020).

Definition 2.1 (Amenability (Perković et al., 2015))**.** *For any two distinct nodes X and Y in a CPDAG G, G is amenable relative to (X, Y) if every possibly directed path from X to Y starts with a directed edge out of X.*

If the causal effect of a treatment X on an outcome Y is identifiable, there might be multiple valid adjustment sets to estimate it. Perković et al. (2018) show that no valid adjustments contain the *forbidden nodes*, $Forb_G(X,Y) = PossDe_G(PossCn_G(X,Y)) \cup X$, where $PossCn_G(X,Y)$ are the nodes on possibly directed paths from X to Y, excluding X. Henckel et al. (2022) and Rotnitzky and Smucler (2020) define the optimal adjustment set as the set with the lowest asymptotic variance across all valid adjustment sets, and provide graphical criteria to identify it.

We take inspiration from (Witte et al., 2020), who formulate the optimal adjustment sets as the parents of the outcome in the *forbidden projection* in a general class of graphs, maxPDAGs, which extends CPDAGs to settings with additional background knowledge. In our case, we focus only on CPDAGs and consider a single treatment and outcome for which the graph is amenable, so Def. 17 in (Witte et al., 2020) together with Prop. 19 and Lem. 20, simplifies to:

Definition 2.2 (Simplified forbidden projection)**.** *Let G be a CPDAG with nodes $\mathbf{V}$, and let $X, Y \in \mathbf{V}$ such that G is amenable relative to (X,Y). Define $\mathbf{F} = Forb_G(X,Y) \setminus \{X,Y\}$. The forbidden projection $\tilde{G}^{X,Y}$ of G is a graph with nodes $\mathbf{V} \setminus \mathbf{F}$ and edges as follows. For distinct nodes $W_i, W_j \in \mathbf{V} \setminus \mathbf{F}$,*

1. *$\tilde{G}^{X,Y}$ contains a directed edge $W_i \to W_j$ if and only if G contains a directed path $W_i \to \cdots \to W_j$ on which all non-endpoint nodes are in $\mathbf{F}$,*
2. *$\tilde{G}^{X,Y}$ contains an undirected edge $W_i - W_j$ if and only if G contains $W_i - W_j$.*

Based on this definition, we can now define the optimal adjustment set as the parents of the outcome in the forbidden projection, excluding the treatment.

Definition 2.3 (Optimal adjustment set (Witte et al., 2020))**.** *For treatment X and outcome Y in an amenable CPDAG G with $\tilde{G}^{X,Y}$ as the forbidden projection, the optimal adjustment set $Oset_G(X,Y)$ relative to X and Y in G is given by*

$$Oset_G(X,Y) = Pa_{\tilde{G}^{X,Y}}(Y) \setminus \{X\}.$$

As we will see in Lem. 4.4, this definition is particularly useful for our method, since we can show that we can slightly modify the projection to include all possible descendants of the treatment, which we can easily estimate from local information, while also reducing the size of the projected graph.

In case the causal effect is not identifiable, we can still estimate a set of possible causal effects by considering all possible DAGs in the MEC of the discovered CPDAG. While this is computationally infeasible for large graphs, Maathuis et al. (2009) showed that local information around the treatment is enough to identify the set of possible *locally valid parent adjustment sets* that can be used to estimate this set of possible effects. Intuitively, a candidate parent adjustment set of an outcome Y contains its identified parents and a subset of its undirected siblings $\mathbf{S} \subseteq Sib_G(Y)$. Then, this candidate parent set $Pa_G(Y) \cup \mathbf{S}$ is locally valid, if after orienting $S - Y$ as $S \to Y$ for all $S \in \mathbf{S}$, no new v-structures are created in G, which would imply additional orientations and contradict the fact that they are siblings in the CPDAG. This is ensured when $S \in \mathbf{S}$ are adjacent to all other candidate parents $Pa_G(Y) \cup \mathbf{S}$, because then all new colliders are shielded.

3 RELATED WORK

We focus on estimating the causal effect of a pair of target variables in a computationally and statistically efficient way, when the causal graph is unknown. A standard way to achieve this is to learn the causal graph over all available variables using standard causal discovery methods (Glymour et al., 2019), e.g., PC (Spirtes et al., 2000), and then find the optimal adjustment set (Henckel et al., 2022) through a graphical characterization. We call this approach *global causal discovery*. The issue with this approach is that global causal discovery is an inherently computationally expensive task, and even methods that reduce the computational complexity, e.g., MARVEL (Mokhtarian et al., 2021), still struggle with hundreds of variables. Moreover, it is usually unnecessary to recover a complete graph only to estimate a causal effect between a pair of variables.

An alternative approach, *local causal discovery*, instead estimates the graph only in the neighborhood of a target variable, which is inherently a computationally easier task, but it is limited to suboptimal adjustment sets, thus potentially providing a less accurate estimate of the causal effect between targets.

Most work in local causal discovery is limited to a single target variable and the effects of its neighbors (Gao and Ji, 2015). The adjustment sets are also restricted to the neighborhood of the target, e.g., the locally valid parent adjustment sets. In particular, MB-by-MB (Wang et al., 2014) employs sequential Markov blanket discovery starting from the target until the orientations of all edges adjacent to the target are determined. Local Discovery using Eager Collider Checks (LDECC) (Gupta et al., 2023) starts by identifying the Markov blanket of the target and then performs CI tests similarly to PC with additional checks to efficiently determine the relation of adjacent variables. LDECC provides complementary computational benefits to the other local causal discovery methods. However, as we show in App. A.1, LDECC and other methods using LocalPC,

e.g. (Xie et al., 2024; Wang et al., 2014; Li et al., 2025), are not sound, since they might misidentify a non-adjacent spouse as adjacent, due to only testing for separating sets among the neighbors. We show that this can be fixed by using the Markov Blanket to test for separating sets. Moreover, as we show in App. A.2, LDECC is not complete, since it may not orient all orientable edges around the treatment in some cases.

In contrast to previous local methods, Local Discovery by Partitioning (LDP) (Maasch et al., 2024) discovers adjustment sets that are not restricted to the neighborhood of the target. LDP learns partitions of the nodes in terms of their relation to the target pair. While more scalable than global causal discovery methods, LDP still potentially learns sub-optimal adjustment sets and assumes that we know which target causes the other. Moreover, as we show in App. A.3, when an identifiable effect between the target pair is zero, LDP might misreport the effect as not identifiable.

Two recent approaches focus on learning adjustment sets in the more general setting, when we do not know how the target variables are causally related, while learning only parts of the causal graph. The Confounder Blanket Learner (CBL) (Watson and Silva, 2022) learns ancestral relations and adjustment sets for the causal effect between the target variables, but requires that all other variables are non-descendants of the targets. SNAP (Schubert et al., 2025) does not have assumptions on the underlying causal graph and recovers the optimal adjustment set without discovering the full CPDAG by progressively identifying and removing definite non-ancestors of the target variables. While more scalable than global causal discovery methods, SNAP still requires causal discovery over all possible ancestors of the targets, which is unnecessary for learning optimal adjustment sets and is potentially still a computational bottleneck to scaling to larger graphs.

In this paper, we combine the best of both worlds in terms of global and local causal discovery: similar to global methods, we aim to learn optimal adjustment sets, which allow us to have statistically efficient causal effect estimation, while in the spirit of local methods, we only want to estimate the necessary causal information for this task, thus being computationally efficient.

4 METHOD

We first discuss how to extend local causal discovery algorithms to the case in which we do not know the causal relations between two targets X and Y. Then, we introduce Local Optimal Adjustments Discovery (LOAD), a sound and complete method that identifies the optimal adjustment set for the causal effect between the pair of targets in a computationally efficient way.

Algorithm 1 LocalRelate

Require: Targets $X, Y \in \mathbf{V}$ and variables $\mathbf{V}$
Ensure: Causal relation between X and Y, G_X, G_Y
1: $Relation(X,Y) \leftarrow Relation(Y,X) \leftarrow DefNonAn$
2: $G_X \leftarrow$ MB-by-MB$(X, \mathbf{V})$
3: $G_Y \leftarrow$ MB-by-MB$(Y, \mathbf{V})$
4: **if** IsExplAn(X, Y, G_X) **then** ▷ Alg. 8
5: $Relation(X,Y) \leftarrow ExplAn$
6: **else if** IsExplAn(Y, X, G_Y) **then** ▷ Alg. 8
7: $Relation(Y,X) \leftarrow ExplAn$
8: **else** ▷ Cannot determine causal relation
9: **if** IsPossAn(X, Y, G_X) **then** ▷ Alg. 9
10: $Relation(X,Y) \leftarrow PossAn$
11: **if** IsPossAn(Y, X, G_Y) **then** ▷ Alg. 9
12: $Relation(Y,X) \leftarrow PossAn$
13: **return** $Relation, G_X, G_Y$

4.1 Identifying relations between targets

While local causal discovery methods usually assume that we known the causal relation between a pair of targets X and Y, in this section we show how we can extend them to the setting in which this relation is unknown. We leverage results by Fang et al. (2022) on how to test explicit and possible ancestry (and conversely, definite non-ancestry) with local information:

Theorem 4.1 (Thm. 3 in (Fang et al., 2022))**.** *For any two distinct nodes X and Y in a CPDAG G, $X \in ExplAn_G(Y)$ iff $X \not\perp\!\!\!\perp Y | Pa_G(X) \cup Sib_G(X)$.*

Theorem 4.2 (Thm. 2 in (Fang et al., 2022))**.** *For any two distinct nodes X and Y in a CPDAG G, X is a definite non-ancestor of Y iff $X \perp\!\!\!\perp Y | Pa_G(X)$. Otherwise, X is a possible ancestor of Y.*

The local information returned by methods such as MB-by-MB (Wang et al., 2014) is the set of parents, children and siblings, thus we can use it to identify the relation between targets. We show an implementation in Alg. 1, which first runs MB-by-MB on each target and then uses Thm. 4.1 and Thm. 4.2 implemented by Alg. 8 and Alg. 9 with a few optimizations to test explicit or possible ancestry using only local information.

The output of the Algorithm are $Relation(X,Y)$ and $Relation(Y,X)$, which describe if the relation between X and Y is definite non-ancestral ($DefNonAn$), i.e., they do not cause each other in any DAG in the MEC, if one is an explicit ancestor of the other ($ExplAn$) or if either is a possible ancestor of the other ($PossAn$). In this last case, the possible ancestry is potentially true in both directions. We show that our method is sound and complete in recovering these relations.

Lemma 4.1. *Alg. 1 is sound and complete in finding*

definite non-ancestral, possible ancestral and explicit ancestral relations between a pair of targets.

We provide a proof in App. D.2. Besides identifying the causal relations, the algorithm also provides local structures around both targets that can be used to enumerate the locally valid parent adjustment sets (Maathuis et al., 2009), as described in Alg. 10.

4.2 Local Optimal Adjustments Discovery

Local Optimal Adjustments Discovery (LOAD) is a sound and complete method to identify the optimal adjustment set for a target pair using only local information. LOAD leverages standard local causal discovery methods to collect subgraphs G_V around nodes V identifying the parents, children and siblings of V. In our implementation we use MB-by-MB (Wang et al., 2014) extended with caching, as described in Alg. 11, but LOAD is agnostic to the local causal discovery method and we evaluate using an alternative method, CMB (Gao and Ji, 2015) in App. H.4. We describe the pseudocode of LOAD in Alg. 3.

LOAD starts in Step 1 by determining the causal relation between the two targets by employing Alg. 1. If a variable is a definite non-ancestor of the other, then its causal effect on the other is trivially identifiable as zero. If one of the variables is a possible ancestor of the other, we prove a necessary but not sufficient condition for the causal effect to be identifiable.

Lemma 4.2. *For any two distinct nodes X and Y such that $X \in PossAn_G(Y)$ in a CPDAG G, the causal effect of X on Y is identifiable only if X is an explicit ancestor of Y in G.*

We provide the proof in App. D.3. If X is an explicit ancestor of Y, then we call X the "treatment" T and Y the "outcome" O, or viceversa if Y is an explicit ancestor of X. If X is not an explicit ancestor of Y, then its causal effect on Y is not identifiable and LOAD returns the locally valid parent adjustment sets for X.

Even if X is an explicit ancestor of Y, its causal effect on Y might still not be identifiable. We can test this by checking amenability (Perković et al., 2015), but that requires checking all possibly directed paths between X and Y, which potentially requires knowing the whole CPDAG. Instead, we develop a test for amenability that uses only local information around X and its siblings.

Lemma 4.3. *For any two distinct nodes X and Y such that $X \in PossAn_G(Y)$ in a CPDAG G, G is adjustment amenable relative to (X, Y) iff*

$$\forall V \in Sib_G(X) : V \perp\!\!\!\perp Y | Pa_G(V) \cup \{X\}.$$

We provide a proof in App. D.4. We implement this lemma through Alg. 2 and use it in Step 2 of LOAD.

Algorithm 2 LocalAmenTest

Require: Targets X, Y, Sibling V, Graph G
if $V \in Adj_G(Y)$ **then**
  return False
else if $V \perp\!\!\!\perp Y | Pa_G(V) \cup \{X\}$ **then**
  return True
else
  return False

This test requires only knowing a local subgraph of the CPDAG, in particular the parents and adjacencies of each sibling of the treatment. We show that with Lem. 4.2 and Lem. 4.3, LOAD correctly determines the identifiability of the causal effects between the targets.

Corollary 4.1. *LOAD is sound and complete in determining the identifiability of the causal effect between a pair of targets X and Y.*

We provide a proof in App. D.5. Similarly to the previous step, if the causal effect of the treatment on the outcome is not identifiable, then LOAD returns the locally valid parent adjustment sets. If the causal effect is identifiable, then LOAD continues in Step 3 to identify the possible descendants of the treatment using Alg. 9, which we use in a modified forbidden projection (Witte et al., 2020) that instead of marginalizing the forbidden nodes out, projects over the nodes $\mathbf{V} \setminus PossDe_G(T) \cup \{T, O\}$. We show that the optimal adjustment set is still the parents of the outcome in this smaller projection, as in the original projection:

Lemma 4.4. *For treatment T and outcome O in an amenable CPDAG G, let $\mathbf{D} = PossDe_G(T)) \setminus \{T, O\}$ and $G^{T,O}$ be the modified forbidden projection with nodes $\mathbf{V} \setminus \mathbf{D}$. We show that $G^{T,O}$ is a CPDAG and the optimal adjustment set $Oset_G(X, Y)$ is given by*

$$Oset_G(T, O) = Pa_{G^{T,O}}(O) \setminus \{T\}.$$

We provide a proof in App. D.6. In Step 4, LOAD performs local causal discovery on the outcome in this modified forbidden projection by running MB-by-MB only on $\mathbf{V} \setminus PossDe_G(T) \cup \{T, O\}$. The parents of the outcome identified by local discovery are the optimal adjustment set. We show that LOAD is sound and complete in discovering optimal adjustment sets.

Theorem 4.3. *LOAD is sound and complete in finding optimal adjustment sets for a pair of target variables.*

We provide the proof in App. D.7. We show our implementation of LOAD in Alg. 3. LOAD returns three sets of variables. *Relation* variable holds the causal relation between the targets, while $IsIdent$ indicates whether a causal effect is identifiable or not. If the causal effect is identifiable, then $AdjSets$ contains the

Algorithm 3 LOAD

Require: Targets $X, Y \in \mathbf{V}$ and variables $\mathbf{V}$
Ensure: Causal relation of X and Y in $Relation$, if the effects of X on Y and Y on X are identifiable in $IsIdent$, and either optimal or locally valid parent adjustment set in $AdjSet$ based on identifiability.

1: $IsIdent_{X\to Y} \leftarrow$ False, $IsIdent_{Y\to X} \leftarrow$ False
2: $AdjSets_{X\to Y} \leftarrow \emptyset, AdjSets_{Y\to X} \leftarrow \emptyset$
▷ **Step 1:** Determine causal relations between targets
3: $Relation, G_X, G_Y \leftarrow$ LocalRelate$(X, Y, \mathbf{V})$ ▷ Alg. 1
4: **if** $Relation(X, Y) = ExplAn$ **then**
5: T, O $\leftarrow X, Y$
6: **else if** $Relation(Y, X) = ExplAn$ **then**
7: T, O $\leftarrow Y, X$
8: **else** ▷ Cannot determine treatment and outcome
9: **if** $Relation(X, Y) = PossAn$ **then**
10: $AdjSets_{X\to Y} \leftarrow$ LocalValidSets(X, Y, G_X)
11: **if** $Relation(Y, X) = PossAn$ **then**
12: $AdjSets_{Y\to X} \leftarrow$ LocalValidSets(Y, X, G_Y)
13: **if** $Relation(X, Y) = DefNonAn$ **then**
14: $IsIdent_{X\to Y} \leftarrow$ True
15: **if** $Relation(Y, X) = DefNonAn$ **then**
16: $IsIdent_{Y\to X} \leftarrow$ True
17: **return** $Relation, IsIdent, AdjSets$
18: $IsIdent_{O\to T} \leftarrow$ True ▷ Zero effect in reverse dir.
▷ **Step 2:** Test identifiability of treatment on outcome
19: **for** $V \in Sib_{G_T}(T)$ **do**
20: $G_V \leftarrow$ MB-by-MB$(V, \mathbf{V})$
21: **if not** LocalAmenTest(T, O, V, G_V) **then**
22: $AdjSets_{T\to O} \leftarrow$ LocalValidSets(T, O, G_T)
23: **return** $Relation, IsIdent, AdjSets$
24: $IsIdent_{T\to O} \leftarrow$ True
▷ **Step 3:** Find possible descendants of treatment
25: $PossDe(T) \leftarrow \{T, O\}$
26: **for** $V \in \mathbf{V} \setminus \{T, O\}$ **do**
27: **if** IsPossAn(T, V, G_T) **then**
28: $PossDe(T) \leftarrow PossDe(T) \cup \{V\}$
▷ **Step 4:** Identify $Oset$ via mod. forbidden projection
29: $G_O^{T,O} \leftarrow$ MB-by-MB$(O, \mathbf{V} \setminus PossDe(T) \cup \{T, O\})$
30: $Oset(T, O) \leftarrow Pa_{G_O^{T,O}}(O) \setminus \{T\}$
31: $AdjSets_{T\to O} \leftarrow \{Oset(T, O)\}$
32: **return** $Relation, IsIdent, AdjSets$

optimal adjustment set, which is trivially the empty set for a zero causal effect. Otherwise, it contains the locally valid parent adjustment sets for targets that are explicit or possible ancestors of the other. We discuss further optimizations in App. E, including substituting MB-by-MB in line 29 with just Markov blanket discovery, which did not significantly improve performance.

Computational Complexity The computational complexity of LOAD in terms of CI tests is mainly determined by the local causal discovery algorithm it uses. Given a local causal discovery algorithm with complexity $\mathcal{O}(L)$, the worst-case complexity of LOAD is given by applying the local algorithm on all variables, adding up to $\mathcal{O}(L \times |\mathbf{V}|)$, plus performing $\mathcal{O}(|\mathbf{V}|)$ tests to determine causal relations using Alg. 8 and Alg. 9, dominated by the previous term. However, we can re-implement most local methods, so that they cache and re-use Markov blankets and local structures from their earlier runs, which allows us to only run a Markov blanket discovery and local structure learning once per variable, reducing the worst-case complexity to $\mathcal{O}(L)$.

In our implementation, we use the MB-by-MB algorithm modified to cache and reuse Markov blankets and local structures from earlier runs in Alg. 11. The complexity of MB-by-MB depends on the complexity of the algorithm for finding Markov blankets, which is a well-known task in literature. The worst-case complexity for forward search in Markov blanket discovery is $\mathcal{O}(|MB_{\max}| \times |\mathbf{V}|)$, where $|MB_{\max}|$ is the size of the largest Markov blanket (Tsamardinos et al., 2003). For learning local structures over the Markov Blankets, we then use PC (Spirtes et al., 2000) which has a worst-case complexity of $\mathcal{O}(|MB_{\max}|^{d_{\max}+2})$, where $d_{\max}$ is the maximum degree over all nodes.

In the worst-case, MB-by-MB runs both subroutines for each node in the graph, resulting in a total complexity for our implementation of $\mathcal{O}(|MB_{\max}| \times |\mathbf{V}|^2 + |MB_{\max}|^{d_{\max}+2} \times |\mathbf{V}|))$. On the other hand, when applied to the whole graph, the PC algorithm has a worst-case complexity of $\mathcal{O}(|\mathbf{V}|^{d_{\max}+2})$. Since typically $|MB_{\max}| \ll |\mathbf{V}|$ this showcases the benefits of local causal discovery and our method. This means that LOAD typically has a much lower complexity compared to running a global method such as PC to obtain a full CPDAG and then collecting adjustments sets using IDA-like methods (Maathuis et al., 2009), which we also demonstrate empirically.

5 EMPIRICAL EVALUATION

We evaluate the performance of LOAD on synthetic and realistic semi-synthetic data from bnlearn (Scutari, 2010). We compare with global algorithms PC (Spirtes et al., 2000) and MARVEL (Mokhtarian et al., 2021), the targeted causal discovery method SNAP(∞) (Schubert et al., 2025), as well as the extended versions of local algorithms MB-by-MB (Wang et al., 2014), LDECC (Gupta et al., 2023) and LDP (Maasch et al., 2024) based on our Alg. 1. We describe the extension in App. B and refer to the extended methods as MB-by-MB$^+$, LDECC$^+$ and LDP$^+$. In this section, we focus on the setting in which the causal relation between the two targets is not known. In App. H.6, we present results for the scenario where the treatment-outcome relation is known, and evaluate the original versions of MB-by-MB, LDECC and LDP, showing similar results. We publish our code at `https://github.com/matyasch/load`.

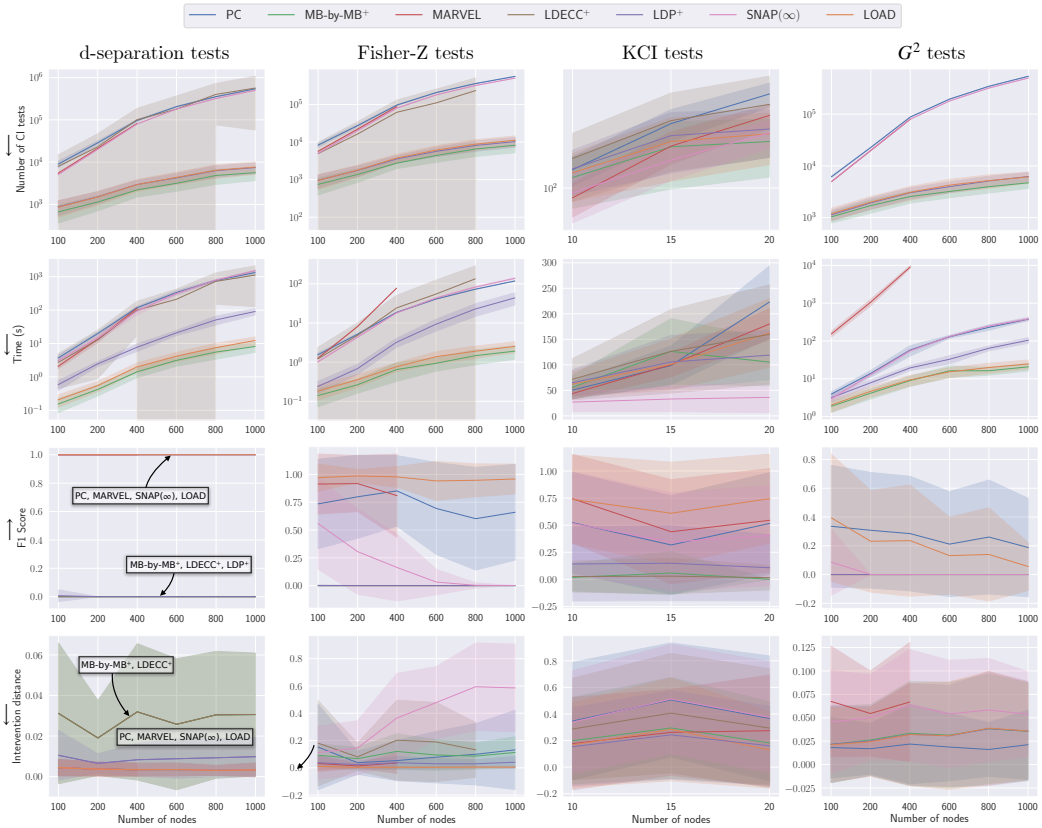


Figure 1: Results over number of nodes with $n_{\mathbf{D}} = 10000$ ($n_{\mathbf{D}} = 1000$ for KCI), $\overline{d} = 2$, $d_{\max} = 10$ and targets such that one is an explicit ancestor of the other. The shadow area denotes the the standard deviation. We could not run MARVEL on more than 400 nodes in any setting, LDECC$^+$ on 1000 nodes with Fisher-Z tests and LDECC$^+$ on any number of nodes with G^2 tests due to memory issues. We report all numbers in detail in App. G.

Synthetic data. We evaluate LOAD using $n_{\mathbf{D}} = 10000$ data samples generated according to randomly generated Erdős–Rényi graphs with varying number of nodes $n_{\mathbf{V}}$, expected degree of $\overline{d} = 2$ and maximum degree of $d_{\max} = 10$. We sample the pairs of targets such that one is an explicit ancestor of the other, but we do not provide this background knowledge about the causal relation between the targets to the algorithms. We evaluate algorithms using three types of data and corresponding conditional independence (CI) tests: oracle d-separation tests, Fisher-Z tests on linear Gaussian data, and G^2 tests on binary data. We also evaluate all algorithms using the non-parametric KCI tests (Zhang et al., 2011) using $n_{\mathbf{D}} = 1000$ data samples of linear Gaussian data. For all CI tests we use a significance level $\alpha = 0.01$. For linear Gaussian data, we sample edge weights from $[-3, -0.5] \cup [0.5, 3]$ with standard Gaussian noises. For the binary data, we generate a conditional probability table according to the graph structure with probabilities sampled uniformly. We report the average results over 100 seeds with the best 5 and worst 5 results removed for each method to better show the general trends. We provide further details of the experimental setup in App. F

Realistic data. We also evaluate all baselines on realistic data generated according to the MAGIC-NIAB and ANDES networks from bnlearn (Scutari, 2010) with ground truth graphs shown in App. F.1. Similarly to the fully synthetic setting, we sample the target pairs such that one is an explicit ancestor of the other. For MAGIC-NIAB ($n_{\mathbf{V}} = 44$ and $\overline{d} = 3$) we simulate linear Gaussian data AND run each algorithm using Fisher-Z tests with significance level $\alpha = 0.01$. For ANDES ($n_{\mathbf{V}} = 223$ and $\overline{d} = 3.03$) we generate binary data and use G^2 tests.

Metrics. We compare methods over three metrics: number of CI tests (which is a hardware and implementation independent measure of computational efficiency), F1 score on the optimal adjustment sets and the intervention distance. If both the output of an algorithm and the true CPDAG indicates that the optimal adjustment set does not exist, then we consider the F1 score to be 1. If the output of an algorithm and the true CPDAG does not agree whether the optimal

adjustment set should exist, then we consider the F1 score to be 0. For MB-by-MB$^+$, LDECC$^+$ and LDP$^+$ we use the highest F1 score reached by the adjustment sets they return. As the recovered adjustment sets should be ultimately used for causal effect estimation, we also report the intervention distance (Schubert et al., 2025), defined for a single target pair as

$$\frac{1}{2}\sum_{T,T'\in\{(X,Y),(Y,X)\}}\frac{1}{|\hat{\Theta}_{T\to T'}|}\sum_{\hat{\theta}\in\hat{\Theta}_{T\to T'}}\left|\theta^*_{T\to T'}-\hat{\theta}\right|,$$

where $\theta^*_{T\to T'}$ is the true causal effect of T on T' and $\hat{\Theta}_{T\to T'}$ is the set of estimated causal effects. For all methods, if their output indicates that T is a definite non-ancestor of T', either based on the output CPDAG, the returned relation or missing adjustment sets, we consider its estimated causal effect $\hat{\Theta}_{T\to T'}$ to be $\{0\}$. For MB-by-MB$^+$, LDECC$^+$, LDP$^+$ and LOAD, we use the adjustment sets they return to estimate the causal effects $\hat{\Theta}_{T\to T'}$. For PC, MARVEL and SNAP, if their output CPDAG indicates that the causal effect is identifiable, then we use the optimal adjustment set according to the CPDAG for the causal effect estimation. If the output CPDAG indicates that the causal effect is not identifiable, then we use the locally valid parent adjustments, which results in a set of possible causal effects $\hat{\Theta}_{T\to T'}$. Following Gradu et al. (2022), we use 10000 newly generated synthetic samples for causal effect estimation to avoid *double dipping*.

Simulated Data Results. Our experimental results in Fig. 1 show that LOAD combines the best of both worlds in terms of local and global causal discovery. In particular, it provides a comparable accuracy in causal effect estimation to the slow but accurate global causal discovery methods, while being only slightly more computationally expensive than the fast but less accurate local causal discovery methods. We could not compare some of the methods in some settings, because they had running time or memory issues. In particular, we could not run MARVEL on more than 400 nodes, LDECC$^+$ on 1000 nodes with Fisher-Z tests and LDECC$^+$ on any number of nodes with G^2 tests.

The first row of Fig. 1 describes the number of CI tests for each method in each of the three settings: d-separation, Fisher-Z tests on linear Gaussian data and G^2 tests on binary data. In all settings, the number of CI performed by LOAD is consistently well below global methods, LDECC$^+$ and SNAP(∞) and slightly above MB-by-MB$^+$ and LDP$^+$. This is expected, since LOAD requires fewer tests than global methods due to not recovering the whole graph, but it requires more CI tests than local methods, since it focuses on a more difficult task, optimal adjustment set discovery.

We report the computation time in the second row of Fig. 1. Our results show similar trends as the number of CI tests, but with LOAD and MB-by-MB$^+$ performing the best, requiring even less computation time than LDP, over all data settings except with KCI tests where SNAP(∞) performs best.

The third row of Fig. 1 describes the F1 score for learning the optimal adjustment set in each setting. This is the main quality metric our method is targeting. Our results using oracle d-separation tests empirically verify that LOAD is sound and complete in this task the same way as global methods and SNAP(∞) are, providing an F1 score of 1. On the other hand, local methods (MB-by-MB, LDECC and LDP) have an F1 score of 0 in terms of recovering the optimal adjustment set, which is expected since this is not their goal. On linear Gaussian data with Fisher-Z tests, LOAD outperforms all baselines and can consistently recover the optimal adjustment set almost perfectly. PC, MARVEL and SNAP(∞) still recover some of the optimal adjustment sets, while as expected local methods are still not able to learn any of the optimal adjustment sets. LOAD similarly outperforms all baselines with KCI tests. All methods struggle on binary data using G^2 tests, as it is inherently more challenging for CI tests than linear Gaussian data, leading to more CI testing errors, as we discuss in App. H.1. In this setting, PC performs best followed by LOAD, while other methods, including MARVEL and SNAP(∞), cannot recover the optimal adjustment set at all.

The fourth row of Fig. 1 shows the intervention distance results. Intervention distance provides a complementary metric for the quality of the optimal adjustment set discovery, since it focuses on the downstream task of causal effect estimation using the discovered (optimal) adjustment sets. The theoretical advantages of the optimal adjustment set are showcased when measuring the intervention distance on results for oracle d-separation CI tests. Here, methods that can recover the optimal adjustment set, including LOAD, achieve not only the lowest intervention distance, but also the lowest variance. On linear Gaussian data, LOAD outperforms all baselines in intervention distance, with LDP following closely. On binary data, PC has the lowest intervention distance, while LOAD is the second best with local methods MB-by-MB$^+$ and LDP$^+$.

Ablations. The accuracy of LOAD in estimating the optimal adjustment set depends on the quality of the local neighborhoods estimated by the local causal discovery subroutine. In App. H.2 we show that the F1 score of LOAD decreases slightly with more structural errors in the estimated local neighborhoods due to lower sample sizes. Notably, even with an average of almost one erroneous node in every local neighborhood for 500

samples of linear Gaussian data with Fisher-Z CI tests, the F1 score and intervention distance of LOAD is still better than most baselines. Additionally, we also study how errors in earlier steps of LOAD propagate to later steps in App. H.3. As expected, the precision and recall scores of the first two steps are generally higher than the later steps. Interestingly, LOAD has a very high precision for Step 2 and better recall than for Step 1, meaning that it almost never predicts that the causal effect is identifiable when it is not.

Our implementation of LOAD utilizes MB-by-MB for local causal discovery, which employs the Grow-Shrink algorithm for Markov blanket discovery. We perform an ablation where we implement LOAD with alternative modules for local causal discovery in App. H.4 and Markov blanket discovery in App. H.5. Our results for replacing MB-by-MB with the CMB (Gao and Ji, 2015) algorithm are presented in Fig. 9 and show that MB-by-MB outperforms CMB in every metric except for computation time with both d-separation and Fisher-Z CI tests. We show results for replacing the Grow-Shrink algorithm with Total-Conditioning (Pellet and Elisseeff, 2008b) in Fig. 10 and with scored-based S^2TMB algorithm (Gao and Ji, 2017) in Fig. 11. While Total-Conditioning performs fewer CI tests and achieves comparable intervention distances, it requires much more computation time. Additionally, it fails to recover the optimal adjustment set on binary data. S^2TMB takes a prohibitively long time for more than a few dozen nodes and achieves worse F1 scores and intervention distances than Grow-Shrink. MB-by-MB, LDECC and LDP assume background knowledge of the treatment-outcome relation. Thus, we also consider the scenario where this is provided to them in App. H.6. We compare with LOAD*, a variant of LOAD that can also leverage this information by skipping Step 1 of the algorithm. Our results show the same trends as in Fig. 1, with LDP improving on its computational performance and LOAD* surpassing PC on binary data in F1 score and intervention distances.

We also report results on identifiable target pairs for $n_{\mathbf{V}} \leq 500$ (App. H.7), where the overall patterns remain consistent. The main distinction is again in the binary setting, where the F1 score and intervention distance of LOAD is on par with PC.

We evaluate the robustness of LOAD to different parameters of the synthetic data generation and experiment $n_{\mathbf{V}} = 200$ with various numbers of data samples in App. H.8, and expected degrees for the synthetic graphs in App. H.9. As expected, with larger numbers of samples, PC, MARVEL, SNAP(∞) and LOAD improve in F1 score and all methods improve in intervention distance. The number of samples does not seem to affect the computation time for Fisher Z. For G^2 the computation time and tests increase with the number of samples, reducing the gap between local methods and LOAD with PC and SNAP(∞). As expected, a larger average degree requires more CI tests and computation time, especially for PC and LDECC$^+$, while the intervention distance and the F1 score worsen.

Table 1: Results for realistic data. The best result for each metric is indicated in **bold**. We did not run LDECC$^+$ due to memory issues.

	CI tests $\times 10^3$	F1 of Oset	Int. Dist.
	MAGIC-NIAB		
PC	14.94 ± 0.34	0.32 ± 0.43	0.016 ± 0.035
MB-by-MB$^+$	2.36 ± 1.65	0.03 ± 0.07	0.007 ± 0.007
MARVEL	9.66 ± 4.87	0.60 ± 0.42	0.011 ± 0.015
LDP$^+$	2.49 ± 1.64	0.14 ± 0.35	0.009 ± 0.010
SNAP(∞)	$\mathbf{2.08 \pm 1.64}$	0.30 ± 0.29	0.010 ± 0.014
LOAD	5.31 ± 6.79	$\mathbf{0.62 \pm 0.43}$	$\mathbf{0.006 \pm 0.006}$
	ANDES		
PC	69.29 ± 2.80	0.00 ± 0.00	0.005 ± 0.005
MB-by-MB$^+$	$\mathbf{2.77 \pm 1.13}$	0.00 ± 0.00	0.004 ± 0.003
MARVEL	24.75 ± 0.00	0.00 ± 0.00	0.013 ± 0.032
LDP$^+$	2.86 ± 1.12	0.00 ± 0.00	0.004 ± 0.003
SNAP(∞)	24.75 ± 0.00	0.00 ± 0.00	0.013 ± 0.032
LOAD	3.01 ± 1.20	$\mathbf{0.05 \pm 0.15}$	$\mathbf{0.002 \pm 0.001}$

Realistic Data Results. We report our results for the MAGIC-NIAB and ANDES networks in Tab. 1. Similarly to synthetic data, the number of CI tests performed by LOAD is between global and local methods. The quality of the recovered adjustment sets by LOAD is consistently as one of the best among all methods. Notably, LOAD achieves the only non-zero F1 score on the ANDES network, which poses a significant challenge for all algorithms due to generating binary data, as discussed in App. H.1. Furthermore, LOAD achieves the lowest intervention distance in both settings.

6 CONCLUSIONS

We propose LOAD, a sound and complete local method to identify optimal adjustment sets to estimate the causal effect of a pair of target variables. Unlike previous local discovery methods, LOAD can determine the treatment-outcome relation of the targets, and if the causal effect is identifiable. We show that LOAD is more computationally efficient than global methods, while retaining their statistical efficiency for causal effect estimation. As future work, we plan to extend LOAD to the causally insufficient setting, for which there are no unique optimal adjustment sets (Smucler et al., 2022), which will require defining graphical criteria to identify candidate adjustments.

Acknowledgements

We thank SURF (www.surf.nl) for the support in using the National Supercomputer Snellius. We would like to thank Roel Hulsman for valuable discussions.

References

Steen A. Andersson, David Madigan, and Michael D. Perlman. A characterization of Markov equivalence classes for acyclic digraphs. *The Annals of Statistics*, 25(2):505 – 541, 1997. doi: 10.1214/aos/1031833662. URL https://doi.org/10.1214/aos/1031833662.

Gabor Csardi and Tamas Nepusz. The igraph software package for complex network research. *InterJournal*, Complex Systems:1695, 2006.

Zhuangyan Fang, Yue Liu, Zhi Geng, Shengyu Zhu, and Yangbo He. A local method for identifying causal relations under markov equivalence. *Artificial Intelligence*, 305:103669, 2022. ISSN 0004-3702. doi: https://doi.org/10.1016/j.artint.2022.103669. URL https://www.sciencedirect.com/science/article/pii/S0004370222000091.

Tian Gao and Qiang Ji. Local causal discovery of direct causes and effects. *Advances in Neural Information Processing Systems*, 28, 2015.

Tian Gao and Qiang Ji. Efficient score-based markov blanket discovery. *International Journal of Approximate Reasoning*, 80:277–293, 2017. ISSN 0888-613X. doi: https://doi.org/10.1016/j.ijar.2016.09.009. URL https://www.sciencedirect.com/science/article/pii/S0888613X1630161X.

Clark Glymour, Kun Zhang, and Peter Spirtes. Review of causal discovery methods based on graphical models. *Frontiers in Genetics*, 10, 2019. ISSN 1664-8021. doi: 10.3389/fgene.2019.00524.

Paula Gradu, Tijana Zrnic, Yixin Wang, and Michael Jordan. Valid inference after causal discovery. In *NeurIPS 2022 Workshop on Causality for Real-world Impact*, 2022.

Shantanu Gupta, David Childers, and Zachary Chase Lipton. Local causal discovery for estimating causal effects. In *Conference on Causal Learning and Reasoning*, pages 408–447. PMLR, 2023.

Aric Hagberg, Pieter Swart, and Daniel S Chult. Exploring network structure, dynamics, and function using networkx. Technical report, Los Alamos National Lab.(LANL), Los Alamos, NM (United States), 2008.

Leonard Henckel, Emilija Perković, and Marloes H Maathuis. Graphical criteria for efficient total effect estimation via adjustment in causal linear models. *Journal of the Royal Statistical Society Series B: Statistical Methodology*, 84(2):579–599, 2022.

Markus Kalisch, Martin Mächler, Diego Colombo, Marloes H. Maathuis, and Peter Bühlmann. Causal inference using graphical models with the R package pcalg. *Journal of Statistical Software*, 47(11):1–26, 2012. doi: 10.18637/jss.v047.i11.

Zheng Li, Zeyu Liu, Feng Xie, Hao Zhang, Chunchen LIU, and Zhi Geng. Local identifying causal relations in the presence of latent variables. In *Forty-second International Conference on Machine Learning*, 2025. URL https://openreview.net/forum?id=O6q2BHK1BL.

Zhaolong Ling, Jiale Yu, Yiwen Zhang, Debo Cheng, Peng Zhou, Xingyu Wu, Bingbing Jiang, and Kui Yu. Local causal discovery without causal sufficiency. *Proceedings of the AAAI Conference on Artificial Intelligence*, 39(18):18737–18745, Apr. 2025. doi: 10.1609/aaai.v39i18.34062. URL https://ojs.aaai.org/index.php/AAAI/article/view/34062.

Jacqueline R. M. A. Maasch, Weishen Pan, Shantanu Gupta, Volodymyr Kuleshov, Kyra Gan, and Fei Wang. Local discovery by partitioning: Polynomial-time causal discovery around exposure-outcome pairs. In *The 40th Conference on Uncertainty in Artificial Intelligence*, 2024.

Marloes H. Maathuis and Diego Colombo. A generalized back-door criterion. *The Annals of Statistics*, 43(3):1060 – 1088, 2015. doi: 10.1214/14-AOS1295. URL https://doi.org/10.1214/14-AOS1295.

Marloes H Maathuis, Markus Kalisch, and Peter Bühlmann. Estimating high-dimensional intervention effects from observational data. *The Annals of Statistics*, 37(6A):3133–3164, 2009.

Dimitris Margaritis and Sebastian Thrun. Bayesian network induction via local neighborhoods. *Advances in neural information processing systems*, 12, 1999.

Christopher Meek. Causal inference and causal explanation with background knowledge. In *Proceedings of the Eleventh conference on Uncertainty in artificial intelligence*, pages 403–410, 1995.

Ehsan Mokhtarian, Sina Akbari, AmirEmad Ghassami, and Negar Kiyavash. A recursive markov boundary-based approach to causal structure learning. In *The KDD'21 Workshop on Causal Discovery*, pages 26–54. PMLR, 2021.

Ehsan Mokhtarian, Sepehr Elahi, Sina Akbari, and Negar Kiyavash. Recursive causal discovery. *arXiv preprint arXiv:2403.09300*, 2024.

Judea Pearl. *Causality*. Cambridge university press, 2009.

Jean-Philippe Pellet and André Elisseeff. Finding latent causes in causal networks: an efficient approach based on markov blankets. In D. Koller, D. Schuurmans, Y. Bengio, and L. Bottou, editors, *Advances in Neural Information Processing Systems*, volume 21. Curran Associates, Inc., 2008a. URL https://proceedings.neurips.cc/paper_files/paper/2008/file/e2230b853516e7b05d79744fbd4c9c13-Paper.pdf.

Jean-Philippe Pellet and André Elisseeff. Using markov blankets for causal structure learning. *Journal of Machine Learning Research*, 9(7), 2008b.

Emilija Perkovic. Identifying causal effects in maximally oriented partially directed acyclic graphs. In *Conference on Uncertainty in Artificial Intelligence*, pages 530–539. PMLR, 2020.

Emilija Perković, Johannes Textor, Markus Kalisch, and Marloes H Maathuis. A complete generalized adjustment criterion. In *Uncertainty in Artificial Intelligence-Proceedings of the Thirty-First Conference (2015)*, pages 682–691. AUAI Press, 2015.

Emilija Perković, Johannes Textor, Markus Kalisch, and Marloes H. Maathuis. Complete graphical characterization and construction of adjustment sets in markov equivalence classes of ancestral graphs. *Journal of Machine Learning Research*, 18(220):1–62, 2018. URL http://jmlr.org/papers/v18/16-319.html.

Andrea Rotnitzky and Ezequiel Smucler. Efficient adjustment sets for population average causal treatment effect estimation in graphical models. *Journal of Machine Learning Research*, 21(188):1–86, 2020. URL http://jmlr.org/papers/v21/19-1026.html.

Anna Roumpelaki, Giorgos Borboudakis, Sofia Triantafillou, and Ioannis Tsamardinos. Marginal causal consistency in constraint-based causal learning. In *Causation: Foundation to Application Workshop, UAI*, 2016.

Mátyás Schubert, Tom Claassen, and Sara Magliacane. Snap: Sequential non-ancestor pruning for targeted causal effect estimation with an unknown graph. In Yingzhen Li, Stephan Mandt, Shipra Agrawal, and Emtiyaz Khan, editors, *Proceedings of The 28th International Conference on Artificial Intelligence and Statistics*, volume 258 of *Proceedings of Machine Learning Research*, pages 3340–3348. PMLR, 03–05 May 2025. URL https://proceedings.mlr.press/v258/schubert25a.html.

Marco Scutari. Learning bayesian networks with the bnlearn R package. *Journal of Statistical Software*, 35(3):1–22, 2010. doi: 10.18637/jss.v035.i03.

Ezequiel Smucler, Facundo Sapienza, and Andrea Rotnitzky. Efficient adjustment sets in causal graphical models with hidden variables. *Biometrika*, 109(1): 49–65, 2022.

Peter Spirtes, Clark N Glymour, and Richard Scheines. *Causation, prediction, and search*. MIT press, 2000.

Johannes Textor, Benito van der Zander, Mark S Gilthorpe, Maciej Liśkiewicz, and George TH Ellison. Robust causal inference using directed acyclic graphs: the r package 'dagitty'. *International Journal of Epidemiology*, 45(6):1887–1894, 2016. doi: 10.1093/ije/dyw341.

Ioannis Tsamardinos, Constantin F Aliferis, Alexander R Statnikov, and Er Statnikov. Algorithms for large scale markov blanket discovery. In *FLAIRS*, volume 2, pages 376–81, 2003.

Changzhang Wang, You Zhou, Qiang Zhao, and Zhi Geng. Discovering and orienting the edges connected to a target variable in a dag via a sequential local learning approach. *Computational statistics & data analysis*, 77:252–266, 2014.

David S Watson and Ricardo Silva. Causal discovery under a confounder blanket. In *Uncertainty in Artificial Intelligence*, pages 2096–2106. PMLR, 2022.

Janine Witte, Leonard Henckel, Marloes H. Maathuis, and Vanessa Didelez. On efficient adjustment in causal graphs. *Journal of Machine Learning Research*, 21(246):1–45, 2020. URL http://jmlr.org/papers/v21/20-175.html.

Feng Xie, Zheng Li, Peng Wu, Yan Zeng, Chunchen Liu, and Zhi Geng. Local causal structure learning in the presence of latent variables. In Ruslan Salakhutdinov, Zico Kolter, Katherine Heller, Adrian Weller, Nuria Oliver, Jonathan Scarlett, and Felix Berkenkamp, editors, *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pages 54511–54530. PMLR, 21–27 Jul 2024. URL https://proceedings.mlr.press/v235/xie24f.html.

Kui Yu, Xianjie Guo, Lin Liu, Jiuyong Li, Hao Wang, Zhaolong Ling, and Xindong Wu. Causality-based feature selection: Methods and evaluations. *ACM Comput. Surv.*, 53(5), September 2020. ISSN 0360-0300. doi: 10.1145/3409382. URL https://doi.org/10.1145/3409382.

Jiji Zhang. On the completeness of orientation rules for causal discovery in the presence of latent confounders and selection bias. *Artificial Intelligence*, 172(16):1873–1896, 2008. ISSN 0004-3702. doi: https://doi.org/10.1016/j.artint.2008.08.001. URL https://www.sciencedirect.com/science/article/pii/S0004370208001008.

K Zhang, J Peters, D Janzing, and B Schölkopf. Kernel-based conditional independence test and application

in causal discovery. In *27th Conference on Uncertainty in Artificial Intelligence (UAI 2011)*, pages 804–813. AUAI Press, 2011.

Yujia Zheng, Biwei Huang, Wei Chen, Joseph Ramsey, Mingming Gong, Ruichu Cai, Shohei Shimizu, Peter Spirtes, and Kun Zhang. Causal-learn: Causal discovery in python. *Journal of Machine Learning Research*, 25(60):1–8, 2024.

Checklist

1. For all models and algorithms presented, check if you include:
   (a) A clear description of the mathematical setting, assumptions, algorithm, and/or model. [Yes]
   (b) An analysis of the properties and complexity (time, space, sample size) of any algorithm. [Yes]
   (c) (Optional) Anonymized source code, with specification of all dependencies, including external libraries. [Yes]
2. For any theoretical claim, check if you include:
   (a) Statements of the full set of assumptions of all theoretical results. [Yes]]
   (b) Complete proofs of all theoretical results. [Yes]
   (c) Clear explanations of any assumptions. [Yes]
3. For all figures and tables that present empirical results, check if you include:
   (a) The code, data, and instructions needed to reproduce the main experimental results (either in the supplemental material or as a URL). [Yes]
   (b) All the training details (e.g., data splits, hyperparameters, how they were chosen). [Yes]
   (c) A clear definition of the specific measure or statistics and error bars (e.g., with respect to the random seed after running experiments multiple times). [Yes]
   (d) A description of the computing infrastructure used. (e.g., type of GPUs, internal cluster, or cloud provider). [Yes]
4. If you are using existing assets (e.g., code, data, models) or curating/releasing new assets, check if you include:
   (a) Citations of the creator if your work uses existing assets. [Yes]
   (b) The license information of the assets, if applicable. [Yes]
   (c) New assets either in the supplemental material or as a URL, if applicable. [Not Applicable]
   (d) Information about consent from data providers/curators. [Not Applicable]
   (e) Discussion of sensible content if applicable, e.g., personally identifiable information or offensive content. [Not Applicable]
5. If you used crowdsourcing or conducted research with human subjects, check if you include:
   (a) The full text of instructions given to participants and screenshots. [Not Applicable]
   (b) Descriptions of potential participant risks, with links to Institutional Review Board (IRB) approvals if applicable. [Not Applicable]
   (c) The estimated hourly wage paid to participants and the total amount spent on participant compensation. [Not Applicable]

Local Causal Discovery for Statistically Efficient Causal Inference Supplementary Materials

A ISSUES IN LOCAL CAUSAL DISCOVERY

In this section we detail the issues we found in various local causal discovery methods. In App. A.1 we discuss LocalPC, a subroutine implemented by several local discovery algorithm, and show that LocalPC is not sound in terms of the edges it returns. Then, in App. A.2 and App. A.3 we consider issues specific to the LDECC and LDP algorithms.

A.1 Issues with LocalPC

A key component of several local discovery algorithms is to distinguish adjacent variables from spouses (or co-parents) in the Markov blanket of a variable. Gupta et al. (2023) implement this subroutine for LDECC by following the logic of the PC algorithm as shown in Alg. 4. Xie et al. (2024) also "apply the logic of the PC algorithm to identify the adjacent edges" over the Markov blanket in their MMB-by-MMB algorithm, an extension of the MB-by-MB algorithm (Wang et al., 2014) to causally insufficient settings the same way. Similarly, Li et al. (2025) "apply the logic of the PC algorithm to identify the skeleton" over the Markov blanket in their LocICR algorithm, which aims to identify causal relationships from local information.

Algorithm 4 LocalPC

Require: Target X, Markov blanket $MB(X)$
1: $Adj(X) \leftarrow MB(X)$
2: $s \leftarrow 0$
3: **while** $Adj(X) > s$ **do**
4: **for** $V \in Adj(X)$ **do**
5: **for** $\mathbf{S} \subseteq Adj(X) \setminus \{V\}$ s.t. $|\mathbf{S}| = s$ **do**
6: **if** $X \perp\!\!\!\perp V | \mathbf{S}$ **then**
7: $Adj(X) \leftarrow Adj(X) \setminus \{V\}$
8: **break**
9: $s \leftarrow s + 1$
10: **return** $Adj(X)$

However, the LocalPC algorithm is not sound, as it can **incorrectly** identify non-adjacent spouses in the Markov blanket as adjacent. Ex. A.1 shows a CPDAG/PAG an example such a scenario.

Example A.1. *Consider the CPDAG and PAG on Fig. 2 with treatment X and outcome Y. In both graphs, X and Y are non-adjacent. However, the LocalPC algorithm in Alg. 4 with input X incorrectly identifies Y as adjacent, as shown below.*

1. *LocalPC correctly finds $X \perp\!\!\!\perp V_2$ at $s = 0$.*
2. *LocalPC removes V_2 from $Adj(X)$.*
3. *After this, V_2 is never considered for conditioning at line 5.*
4. *Thus, LocalPC never finds $X \perp\!\!\!\perp Y | \{V_1, V_2\}$.*
5. *LocalPC **incorrectly** leaves $Y \in Adj(X)$.*

The core issue of the LocalPC algorithm in Alg. 4 is that when it tries to separate X from some V, it only considers the still remaining adjacencies of X for separating sets, and not the adjacencies of V. This is in contrast with the true logic of the PC algorithm, which would also consider subsets of the adjacencies of V if the adjacencies of X could not separate the two. Finding separating sets among subsets of the adjacencies, which are a superset of the parents, of both variables is possible due to the Markov condition and Lemma 3.3.9 in (Spirtes et al., 2000), which we restate in Lem. A.1.

(a) Example CPDAG

(b) Example PAG

Figure 2: An example CPDAG and PAG for Ex. A.1 where the LocalPC algorithm in Alg. 4 incorrectly identifies Y as adjacent to X because it never tests $X \perp\!\!\!\perp Y | \{V_1, V_2\}$

Lemma A.1 (Lemma 3.3.9 in (Spirtes et al., 2000))**.** *In a DAG G, if Y is not a descendant of X, and X and Y are not adjacent, then X and Y are d-separated by $Pa(X)$.*

Crucially, Lem. A.1 does not apply when Y is a descendant of X, which is exactly the case in Ex. A.1. Instead of LocalPC, local algorithms should implement Lem. A.2, which appeared as part of the Grow-Shrink algorithm in (Margaritis and Thrun, 1999), discussed in (Pellet and Elisseeff, 2008a), stated as a consequence of Lemma 1 in (Wang et al., 2014), and also stated as Theorem 1 in (Xie et al., 2024) and as Proposition 1 in (Li et al., 2025).

Lemma A.2. *In a DAG/MAG G, for X and $Y \in MB(X)$, Y is adjacent to X if and only if*

$$\forall \mathbf{S} \subsetneq MB(X) \setminus \{Y\} : X \not\perp\!\!\!\perp Y | \mathbf{S}.$$

A.1.1 Solution

In practice, LocalPC in Alg. 4 can be fixed by replacing $Adj(X)$ with $MB(X)$ at lines 3 and 5.

A.1.2 Code to Reproduce Ex. A.1

The following code snippet reproduces Ex. A.1 for LDECC using the public implementation from Gupta et al. (2023). The assertion in the final line fails because node 4 (Y in the example) is returned as a child of node 0 (X in the example).

```
import networkx as nx
import pandas as pd
import numpy as np

from causal_discovery.ldecc import LDECCAlgorithm

dag = nx.DiGraph([(0, 1), (0, 3), (1, 4), (2, 1), (2, 4), (4, 3)])
data = np.zeros((2,len(dag.nodes)))
ldecc = LDECCAlgorithm(
    treatment_node=0,
    outcome_node=4,
    alpha=0.05,
    use_ci_oracle=True,
    graph_true=dag
)
result = ldecc.run(pd.DataFrame(data))
assert result['tmt_children'] == {1,3}
```

The following code snippet implements Ex. A.1 for MMB-by-MMB using the public implementation[1] from Xie

[1]GitHub repository: `https://github.com/fengxie009/MMB-by-MMB`.

et al. (2024). Similarly to LDECC, the assertion in the final line fails because node 4 (Y in the example) is returned as adjacent to node 0 (X in the example).

```
import networkx as nx
import numpy as np

import MMB_by_MMB
import Utils.MMB_TC_Z as tc

dag = nx.DiGraph([(0, 1), (0, 3), (1, 4), (2, 1), (2, 4), (4, 3)])
data = np.zeros((2, len(dag.nodes)))
ci_test = lambda x, y, S, data, alpha: (
    nx.is_d_separator(dag, x, y, set(S)),
    float(nx.is_d_separator(dag, x, y, set(S))),
)
tc.CI_Test = ci_test
MMB_by_MMB.CI_Test = ci_test
mmb_by_mmb = MMB_by_MMB.MMB_by_MMB(
    Data=data,
    target=0,
    alpha=0.05,
    p=len(dag.nodes),
    maxK=max([val for _, val in dag.degree()]),
)
result = mmb_by_mmb.mmb_by_mmb()
assert result[3].tolist() == [1]
```

In the case of LocICR, even though Y is considered adjacent by the algorithm, the returned causal relation is correct.

A.2 Issues with LDECC beyond LocalPC

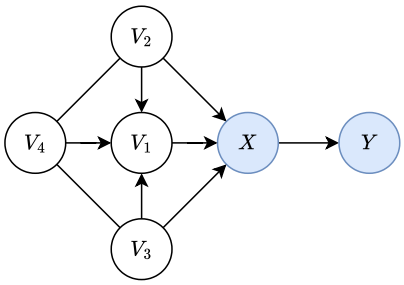
In the previous section we showed that LDECC might erroneously consider spouses as adjacent due to the LocalPC algorithm. In this section we show that even if the LocalPC algorithm is fixed or replaced, LDECC is still not complete, as it might not orient all adjacent edges to the target, even though they are oriented in the true CPDAG.

In particular, Theorem 5 in (Gupta et al., 2023) aims to show that "every orientable neighbor of the treatment X will get oriented correctly by LDECC". However, Ex. A.2 shows a CPDAG where LDECC fails to orient an orientable parent of the treatment.

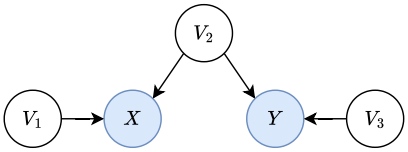
Example A.2. *Consider the CPDAG on Fig. 3a with treatment X and outcome Y. In this CPDAG, all parents of X are orientable. However, LDECC fails to orient V_1 as a parent, as shown below.*

1. *LDECC correctly identifies that Markov blanket of X as $MB(X) = \{Y, V_1, V_2, V_3\}$.*
2. *LDECC correctly identifies the adjacencies of X as $Adj(X) = \{Y, V_1, V_2, V_3\}$.*
3. *LDECC correctly finds $Y \perp\!\!\!\perp V_{1,2,3,4} | X$.*
4. *LDECC correctly finds $V_2 \perp\!\!\!\perp V_3 | V_4$.*
   (a) *The condition at line 7 is satisfied, correctly identifying V_2 and V_3 as parents.*
   (b) *The condition at line 19 is satisfied, correctly identifying Y as a children.*
5. *No other independence exists and LDECC **incorrectly** leaves V_1 as unoriented.*

Running PC on Ex. A.2 would orient $V_1 \to X$ by first orienting the v-structure $V_2 \to V_1 \leftarrow V_3$, then applying Meek rule 3 to orient $V_4 \to V_1$ and finally applying Meek rule 1 to orient $V_1 \to X$. Crucially, in the v-structure $V_2 \to V_1 \leftarrow V_3$ that originally starts the propagation of these orientation rules, both V_2 and V_3 are adjacent to X and hence cannot be separated from it. LDECC fails on Ex. A.2 because it does not contain a condition to handle this scenario. In particular, this structure contradicts the proof of Theorem 5 for Meek rule 1 in (Gupta et al., 2023), which states that both V_2 and V_3 can be separated from X with separating sets that should include V_1.



(a) CPDAG for Ex. A.2



(b) DAG/CPDAG for Ex. A.3

Figure 3: (a): Example CPDAG for Ex. A.2, where LDECC fails to orient $V_1 \to X$ and identify V_1 as a parent of X. (b): Example DAG/CPDAG for Ex. A.3 where LDP fails to identify a valid adjustment set for treatment X and outcome Y, even though the causal effect of 0 and a valid adjustment set of V_2 is identifiable.

A.2.1 Code to Reproduce Ex. A.2

Ex. A.2 can be reproduced using the public implementation[2] of LDECC from Gupta et al. (2023) as follows, where the assertion in the final line fails because node 2 (V_1 in the example) is not recognized as a parent of node 0 (X in the example) by LDECC.

```
import networkx as nx
import pandas as pd
import numpy as np

from causal_discovery.ldecc import LDECCAlgorithm

dag = nx.DiGraph([(0,1), (2,0), (3,0), (4,0), (3,2), (4,2), (5,2), (5,3), (5,4)])
data = np.zeros((2,len(dag.nodes)))
ldecc = LDECCAlgorithm(
    treatment_node=0,
    outcome_node=1,
    alpha=0.05,
    use_ci_oracle=True,
    graph_true=dag
)
result = ldecc.run(pd.DataFrame(data))
assert result['tmt_parents'] == {2,3,4}
```

A.3 Issues with LDP

Local Discovery by Partitioning (LDP) is a causal discovery algorithm that identifies a valid adjustment set for a treatment-outcome pair by partitioning other variables according to their causal relationships to this pair (Maasch et al., 2024). LDP requires the following assumptions to hold to be able to discover a valid backdoor adjustment set for the causal effect of treatment X on outcome Y:

1. X and Y are marginally dependent.
2. Y is not an ancestor of X.
3. There exists a non-descendant of Y that is marginally dependent on Y, but marginally independent of X.
4. There exists a non-descendant of X whose causal effect on Y is *fully mediated* by X, shares no confounder with Y, and is marginally independent of every non-descendant of X that lie on active backdoor paths between X and Y.

[2]GitHub repository: `https://github.com/acmi-lab/local-causal-discovery`.

Maasch et al. (2024) state that "the causal effect of X on Y can be of arbitrary strength or **null**". Thus, as long as the conditions above hold X does not need to be an ancestor of Y (although it is then not clear what *fully mediated* means in the fourth condition). However, in this case, Ex. A.3 shows a causal DAG with corresponding CPDAG from which a valid backdoor adjustment set and the true causal effect of 0 is identifiable, but LDP incorrectly raises a warning that a valid adjustment set is not identifiable.

Example A.3. *Consider the causal DAG, with identical CPDAG, in Fig. 3b with treatment X and outcome Y. The DAG satisfies all conditions listed above, as*

1. *X and Y are marginally dependent.*
2. *Y is not an ancestor of X.*
3. *V_3 is a non-descendant of Y that is marginally dependent on Y, but marginally independent of X.*
4. *V_1 is a non-descendant of X whose causal effect on Y is fully mediated by X, shares no confounder with Y, and is marginally independent of V_2, the only non-descendant of X that lie on active backdoor paths between X and Y.*

Furthermore, we can identify a valid adjustment set $\{V_2\}$ and a causal effect of 0 from the CPDAG. However, LDP fails to identify any valid backdoor adjustment sets as shown below.

Step 1 correctly identifies $\mathbf{Z}_8 = \emptyset$
Step 2 correctly identifies $\mathbf{Z}_4 = \{V_3\}$.
Step 3 ***incorrectly*** *identifies $\mathbf{Z}_{5,7} = \emptyset$ instead of $\mathbf{Z}_{5,7} = \{V_1\}$, because $Y \perp\!\!\!\perp V_1$.*
Step 4 correctly identifies $\mathbf{Z}_{POST} = \emptyset$.
Step 5 correctly identifies $\mathbf{Z}_{MIX} = \emptyset$.
Step 6 is skipped because $\mathbf{Z}_{MIX} = \emptyset$, ***incorrectly*** *resulting in $\mathbf{Z}_1 = \mathbf{Z}_{1,5} = \emptyset$.*
Step 7 is skipped because $\mathbf{Z}_1 = \mathbf{Z}_{1,5} = \emptyset$ ***incorrectly*** *resulting in $\mathbf{Z}_5 = \emptyset$.*
Step 8 ***incorrectly*** *concludes that a valid adjustment set is not identifiable because $\mathbf{Z}_5 = \emptyset$.*

LDP fails on Ex. A.3 because all non-descendants of the treatment that satisfy condition 4 are marginally independent of the outcome, violating the conditions of Step 3. Thus, LDP can be easily fixed by extending condition 4 to also require that at least one non-descendant satisfying all existing conditions is also marginally dependent on the outcome.

A.3.1 Code to Reproduce Ex. A.3

Ex. A.3 can be reproduced using the public implementation[3] of LDP from Maasch et al. (2024) as follows, where the assertion in the final line fails due to node 2 (V_2 in Ex. A.2) not getting recognized as a valid adjustment set for treatment node 1 (X in Ex. A.2) and outcome node 3 (Y in Ex. A.2) by LDP.

```
import numpy as np
import networkx as nx
import pandas as pd

from ldp import LDP

dag = nx.DiGraph([(0, 1), (2, 1), (2, 3), (4, 3)])
data = np.zeros((2, len(dag.nodes)))
ldp = LDP(data=pd.DataFrame(data), independence_test="oracle")
ldp.test = lambda x, y, S: float(nx.is_d_separator(dag, x, y, set(S) if S else set()))
res = ldp.partition_z(exposure=1, outcome=3)
assert res["Z1"] != []
```

[3]GitHub repository: `https://github.com/jmaasch/ldp`.

B EXTENDED LOCAL CAUSAL DISCOVERY ALGORITHMS

In this section we implement the extensions to local causal discovery algorithms outlined in Sec. 4.1. We show MB-by-MB$^+$ in Alg. 5, LDECC$^+$ in Alg. 6 and LDP$^+$ in Alg. 7. In general, all three algorithms use Alg. 1 to determine ancestral relations and then return adjustment sets according to possible treatment-outcome relationships. MB-by-MB$^+$ and LDECC$^+$ use MB-by-MB and LDECC respectively to find local information and return the locally valid parent sets for each target determined to be an explicit or a possible ancestor of the other.

Algorithm 5 MB-by-MB$^+$

Require: Targets $X, Y \in \mathbf{V}$ and variables $\mathbf{V}$
Ensure: Causal relation of X and Y in *Relation* and the locally valid parent adjustment sets of (possible) treatments among X and Y.

$AdjSets_{X\to Y} \leftarrow \emptyset, AdjSets_{Y\to X} \leftarrow \emptyset$
▷ **Step 1:** Determine ancestral relationships
$LocalRelate \leftarrow$ LocalRelate$(X, Y, \mathbf{V})$ ▷ Alg. 1
if $Relation(X, Y) = ExplAn$ **then**
    $AdjSets_{X\to Y} \leftarrow$ LocalValidSets(X, Y, G_X)
else if $Relation(Y, X) = ExplAn$ **then**
    $AdjSets_{Y\to X} \leftarrow$ LocalValidSets(Y, X, G_Y)
else ▷ Cannot determine treatment and outcome
    if $Relation(X, Y) = PossAn$ **then**
        $AdjSets_{X\to Y} \leftarrow$ LocalValidSets(X, Y, G_X)
    if $Relation(Y, X) = PossAn$ **then**
        $AdjSets_{Y\to X} \leftarrow$ LocalValidSets(Y, X, G_Y)
return $Relation, AdjSets$

Algorithm 6 LDECC$^+$

Require: Targets $X, Y \in \mathbf{V}$ and variables $\mathbf{V}$
Ensure: Causal relation of X and Y in *Relation* and the locally valid parent adjustment sets of (possible) treatments among X and Y.

$AdjSets_{X\to Y} \leftarrow \emptyset, AdjSets_{Y\to X} \leftarrow \emptyset$
▷ **Step 1:** Determine ancestral relationships
$LocalRelate \leftarrow$ LocalRelate$(X, Y, \mathbf{V})$ ▷ Alg. 1, but MB-by-MB replaced by LDECC at lines 2 and 3.
if $Relation(X, Y) = ExplAn$ **then**
    $AdjSets_{X\to Y} \leftarrow$ LocalValidSets(X, Y, G_X)
else if $Relation(Y, X) = ExplAn$ **then**
    $AdjSets_{Y\to X} \leftarrow$ LocalValidSets(Y, X, G_Y)
else ▷ Cannot determine treatment and outcome
    if $Relation(X, Y) = PossAn$ **then**
        $AdjSets_{X\to Y} \leftarrow$ LocalValidSets(X, Y, G_X)
    if $Relation(Y, X) = PossAn$ **then**
        $AdjSets_{Y\to X} \leftarrow$ LocalValidSets(Y, X, G_Y)
return $Relation, AdjSets$

LDP$^+$ uses MB-by-MB to find local information and uses LDP to determine a valid adjustment set for each target determined to be an explicit or a possible ancestor of the other.

Algorithm 7 LDP$^+$

Require: Targets $X, Y \in \mathbf{V}$ and variables $\mathbf{V}$
Ensure: Causal relation of X and Y in *Relation* and the causal partitions determined by LDP for the approriate (possible) treatment-outcome pairs.

$AdjSets_{X\to Y} \leftarrow \emptyset, AdjSets_{Y\to X} \leftarrow \emptyset$
▷ **Step 1:** Determine ancestral relationships
$LocalRelate \leftarrow \text{LocalRelate}(X, Y, \mathbf{V})$ ▷ Alg. 1
if $Relation(X, Y) = ExplAn$ **then**
    $AdjSets_{X\to Y} \leftarrow \text{LDP}(X, Y)$
else if $Relation(Y, X) = ExplAn$ **then**
    $AdjSets_{Y\to X} \leftarrow \text{LDP}(Y, X)$
else ▷ Cannot determine treatment and outcome
    if $Relation(X, Y) = PossAn$ **then**
        $AdjSets_{X\to Y} \leftarrow \text{LDP}(X, Y)$
    if $Relation(Y, X) = PossAn$ **then**
        $AdjSets_{Y\to X} \leftarrow \text{LDP}(Y, X)$
return $Relation, AdjSets$

C ADDITIONAL ALGORITHMS

This section provides pseudocode for all subroutines implemented by LOAD. Alg. 9 implements the negation of Thm. 4.2, i.e., that if $X \not\perp\!\!\!\perp Y | Pa_G(X)$, then X is a possible ancestor of Y. Before testing this dependence, the algorithm checks whether possible ancestry can already be determined from the local information around X, when Y is part of it. Alg. 8 implements Thm. 4.1, and also checks the local information around X before testing the corresponding dependence.

Algorithm 8 IsExplAn

Require: Nodes X, Y and graph G
if $Y \in Ch_G(X)$ **then**
  return True
else if $Y \in Pa_G(X) \cup Sib_G(X)$ **then**
  return False
else if $X \not\perp\!\!\!\perp Y | Pa_G(X) \cup Sib_G(X)$ **then**
  return True
else
  return False

Algorithm 9 IsPossAn

Require: Nodes X, Y and graph G
if $Y \in Ch_G(X) \cup Sib_G(X)$ **then**
  return True
else if $Y \in Pa_G(X)$ **then**
  return False
else if $X \not\perp\!\!\!\perp Y | Pa_G(X)$ **then**
  return True
else
  return False

Alg. 10 implements a slight variation of Algorithm 3 in (Maathuis et al., 2009). In particular, it returns all locally valid parent adjustment sets (instead of the estimated causal effects using them).

Intuitively this algorithm checks if considering a subset of the siblings $\mathbf{S}$ as parents of T in addition to $Pa_G(T)$ would create new v-structures and hence orientations, which would contradict the fact that they are actually siblings and not parents in the original CPDAG. A simple way to avoid this is to only add new candidate parents that are adjacent to the currently considered parents.

Algorithm 10 LocalValidSets (slight variation of Algorithm 3 in (Maathuis et al., 2009) that returns all valid sets instead of causal effects)

Require: Treatment T, Outcome O and graph G
$ValidSets \leftarrow \emptyset$
for $\mathbf{S} \subseteq Sib_G(T) \setminus \{O\}$ **do**
  ▷ Ensure that any new candidate parent $S \in \mathbf{S}$ is adjacent to all candidate parents $Pa_G(T) \cup \mathbf{S} \setminus \{S\}$.
  $Valid \leftarrow$ True
  for $S \in \mathbf{S}$ **do**
    if $(Pa_G(T) \cup \mathbf{S} \setminus \{S\}) \not\subseteq Adj_G(S)$ **then**
      $Valid \leftarrow$ False
      break
  if $Valid$ **then**
    $ValidSets \leftarrow ValidSets \cup \{Pa_G(T) \cup \mathbf{S}\}$
return $ValidSets$

Alg. 11 implements the MB-by-MB algorithm by Wang et al. (2014). MB-by-MB is a local causal discovery algorithm that identifies the parents, children and siblings of a target variable by sequentially discovering the Markov blankets of variables until some stopping criteria is met. Similarly to the original paper, in this algorithm we use the notation of $MB^+(X) = MB(X) \cup \{X\}$.

We allow MB-by-MB to cache and reuse already identified separating sets, Markov blankets and local structures by previous runs of the algorithm, as these might be identified during earlier runs of MB-by-MB on different targets. This allows us to avoid unnecessarily identifying the same relationships multiple times. When the cached objects are empty, the algorithm behaves exactly the same as the original algorithm.

The local structure L_X of X is learned at line 31 by running the skeleton step of the PC algorithm (Spirtes et al., 2000) over $MB^+(X)$ and then orient v-structures. While this structure might contain erroneous edges due to latent confounding, Wang et al. (2014) prove that the edges connected to X and the v-structures containing X in

L_X are correct and thus safe to add to the learned graph over all considered variables G at line 33.

Algorithm 11 Slight variation of MB-by-MB (Wang et al., 2014) with caching

Require: Target $T \in \mathbf{V}$, variables $\mathbf{V}$, cached Markov blankets MB and cached local structures L
1: **procedure** ORIENT-UNDIRECTED-EDGES(G)
2: **for** $(a \to b - c) \in G$ **do**
3: **if** exists a separating set for (a, c) and $b \in sepset(a, c)$ **then**
4: Orient $b \to c$ in G
5: **for** $(a \to b \to c - a) \in G$ **do**
6: Orient $a \to c$ in G
7: **for** $a - b, a - c \to b$ and $a - d \to b \in G$ **do**
8: **if** exists a separating set for (c, d) and $a \in sepset(c, d)$ **then**
9: Orient $a \to b$ in G
10: **return** G
11: Take cached separating sets *sepset*, Markov blankets MB and local structures L from previous runs of MB-by-MB.
12: DoneList $= \emptyset$
13: WaitList $= [T]$
14: $G = \{\mathbf{V}, \emptyset\}$
15: **repeat**
16: $X \leftarrow$ take a node from the head of WaitList
17: **if** $MB(X) \in MB$ **then**
18: Use cached $MB(X)$ from an earlier run of MB-by-MB
19: **else**
▷ Design choice: We use Grow-Shrink for Markov blanket discovery instead of IAMB in (Wang et al., 2014)
20: Find $MB(X)$ using the Grow-Shrink Markov blanket algorithm (Figure 2 in (Margaritis and Thrun, 1999))
21: Add $MB(X)$ to cached MB
22: Add $[MB(X) \setminus$ DoneList $\setminus$ WaitList$]$ to the tail of WaitList
23: Add X to DoneList
24: **if** $L_X \in L$ **then**
25: Use cached L_X from an earlier run of MB-by-MB
26: **else if** $MB^+(X) \subseteq MB^+(X')$ for some $X' \in$ DoneList **then**
27: Set L_X equal to the substructure of $L_{X'}$ over $MB^+(X)$
28: **else if** $MB(X) \subseteq$ DoneList **then**
29: Set L_X equal to the substructure of G over $MB^+(X)$
30: **else**
▷ Design choice: we run the PC skeleton search and orient v-structures instead of IC in (Wang et al., 2014)
31: Learn L_X from observed data of $MB^+(X)$ using PC (Spirtes et al., 2000) and updating cached *sepset*
32: Add L_X to cached L
33: Put the edges connected to X and the v-structures containing X in L_X to G
34: Call ORIENT-UNDIRECTED-EDGES(G) to orient undirected edges in G
35: Remove all nodes from WaitList whose paths to T in G are blocked by directed edges
36: **until** WaitList is empty
37: **return** G, MB, L

D PROOFS

D.1 Useful Results from Literature

We use the following result for the correctness of the MB-by-MB algorithms from (Wang et al., 2014) in our proof for Lem. 4.1 in App. D.2.

Theorem D.1 (Theorem 3 in (Wang et al., 2014))**.** *Suppose that a causal network is causal sufficient and faithful to a probability distribution and that all conditional independencies are correctly checked. Then the MB-by-MB algorithm can correctly discover the edges connected to the target node T. Further these edges can be correctly oriented as the partially directed network representing the Markov equivalence class of the underlying global causal network.*

This result implies that after MB-by-MB, all the nodes adjacent to a target will have their edges oriented as they would be in the underlying CPDAG. In other words, this result correctly identifies the parents, children and siblings (i.e., nodes connected by undirected edges) of each target.

Following Perković et al. (2018) we also define an unshielded path as a path, where each consecutive triple X, Y, Z is unshielded, i.e., X is not adjacent to Z, and we use the following result from (Zhang, 2008) in our proofs for Lem. 4.2 in App. D.3 and Lem. 4.3 in App. D.4.

Lemma D.1 (Lemma B.1 in (Zhang, 2008))**.** *Let X and Y be distinct nodes in a CPDAG G. If p is a possibly directed path from X to Y in G, then some subsequence of p forms an unshielded possibly directed path from X to Y in G.*

We use the following results from (Andersson et al., 1997) in our proofs for Lem. D.2, Lem. D.4 and Lem. 4.3 in App. D.4.

Theorem D.2 (Theorem 4.1 in (Andersson et al., 1997))**.** *A graph $G = (\mathbf{V}, \mathbf{E})$ is equal to the CPDAG for some DAG D if and only if G satisfies the following four conditions.*

1. *G is a chain graph.*
2. *For every chain component τ of G, G_τ is chordal.*
3. *The structure $X \to Y - Z$ does not occur as an induced subgraph of G.*
4. *Every directed edge $X \to Y \in G$ is strongly protected in G.*

In our proofs, we only use items 1 and 3 in Thm. D.2, while for the terms used in the other items, we refer to the original paper (Andersson et al., 1997). As discussed by Andersson et al. (1997), item 1 implies that any CPDAG is a chain graph, i.e. it may have both directed and undirected edges but it may contain *no partially directed cycles*, which are formed by a possibly causal path from X to Y with a directed path from Y to X. Item 3 implies that no CPDAGs can contain the the structure $X \to Y - Z$.

We use the following results by Maathuis and Colombo (2015) in our proof for Lem. 4.3.

Corollary D.1 (Corollary 4.2 in (Maathuis and Colombo, 2015))**.** *Let X and Y be two distinct vertices in a CPDAG C. Let $C_{\underline{X}}$ be the graph obtained from C by removing all directed edges out of X. Then there exists a generalized back-door set relative to (X, Y) and C if and only if $Y \in Pa_C(X)$ and $Y \notin PossDe_{C_{\underline{X}}}(X)$. Moreover, if such a generalized back-door set exists, then $Pa_C(X)$ is such a set.*

A generalized back-door set relative to (X, Y) is a valid adjustment set that allows for the estimation of the causal effect of X on Y. The formal definition of a generalized back-door set is given by Definition 3.7 in (Maathuis and Colombo, 2015). Cor. D.1 states that if a generalized back-door set exists relative to (X, Y) in a CPDAG C, (which is the case when the causal effect of X on Y is identifiable in C), then the definite parents $Pa_C(X)$ are such a valid adjustment set.

D.2 Proof of Lem. 4.1

We introduce the following Lemmas to show the soundness and completeness of Alg. 8 and Alg. 9, which we use in the proofs of Lem. 4.1.

Lemma D.2. *Alg. 8 is sound and complete in finding explicit ancestral relations between a pair of targets.*

Proof. Consider target pair X and Y in a CPDAG G. We need to prove that IsExplAn(X, Y) described in Alg. 8 returns True iff X is an explicit ancestor of Y in G, i.e., X has a directed path to Y in the CPDAG.

We consider first the case in which X and Y are adjacent. In this case, X is an explicit ancestor of Y iff Y is a definite child of X. The "if" direction is trivial: if Y is a child of X, then there exists a directed path from X to Y, thus X is an explicit ancestor of Y (lines 1 and 2).

For the "only if" direction, if X is an explicit ancestor of Y, then by definition there exists a directed path from X to Y. Then, an edge between X and Y has to be oriented as $X \to Y$, otherwise there would be a partially directed cycle in G, which is forbidden in CPDAGs as shown by Thm. D.2. This means that if X and Y is adjacent, but Y is not a child of X, i.e., Y is a parent or a sibling of X, then X cannot be an explicit ancestor of Y (lines 3 and 4).

In case X and Y are not adjacent, then Alg. 8 implements Thm. 4.1 by Fang et al. (2022) at lines 5 to 8, which is a sound and complete criterion for determining explicit ancestry. □

Lemma D.3. *Alg. 9 is sound and complete in finding possible ancestral relations between a pair of targets.*

Proof. Consider target pair X and Y in a CPDAG G. We need to prove that IsPossAn(X, Y) described in Alg. 9 returns True iff X is a possible ancestor of Y in G, i.e., there is a possibly directed path from X to Y.

In case X and Y are adjacent, then X is a possible ancestor of Y iff Y is a child or sibling of X. The "if" direction is trivial: if Y is a child or sibling of X, then there is a possibly directed path from X to Y, thus by definition X is a possible ancestor of Y (lines 1 and 2).

For the "only if" direction, if X is a possible ancestor of Y, then Y cannot be a definite parent of X, since otherwise Y would also be a definite ancestor, i.e., an ancestor of X in every DAG represented by G. By acyclicity this means that in none of these DAGs X can be an ancestor of Y, thus X would not be a possible ancestor of Y by definition (lines 3 and 4).

In case X and Y are not adjacent, then Alg. 9 implements Thm. 4.2 by Fang et al. (2022) at lines 5 to 8, which is a sound and complete criterion for determining explicit ancestry. □

We can now use these results to prove that the algorithm for identifying the relations between targets (Alg. 1) is sound and complete in its outputs.

Lemma 4.1. *Alg. 1 is sound and complete in finding definite non-ancestral, possible ancestral and explicit ancestral relations between a pair of targets.*

Proof. At the start of the algorithm, the estimated causal relations for the target pair X and Y are set as definite non-ancestor as default in both directions, denoted as $DefNonAn$ (line 1). We then use MB-by-MB to discover the local information around both X and Y (lines 2-3).

By Thm. D.1, this local information correctly identifies the parents, children and siblings of each of the targets in the CPDAG. This information is enough to use the previous lemmas and algorithms to correctly identify the explicit ancestry in each direction (lines 4 to 7) with Alg. 8, which we show is sound and complete in Lem. D.2, and possible ancestry in each direction (lines 8 to 12) with Alg. 9, which we show is sound and complete in Lem. D.3. While for explicit ancestry between X and Y only one direction can hold at the same time, and the other direction defaults to definite non-ancestor, for possible ancestry both directions are possible at the same time (e.g., when X and Y are connected by an undirected path), so we test them both in any case (lines 8 to 12).

Finally, assume that a target, say X, is a definite non-ancestor of the other, say Y. By definition, X is a definite non-ancestor of Y when it is not a possible ancestor of Y. By definition, if X is not a possible ancestor of Y, then it is also not an explicit ancestor of Y. Furthermore, if Y is an explicit ancestor of X, then X is not a possible ancestor of Y. Since Alg. 8 and Alg. 9 are sound and complete for testing explicit and possible ancestry, X is a definite non-ancestor of Y iff Alg. 1: (i) finds that Y is an explicit ancestor of X, or (ii) if it finds that X is neither an explicit nor a possible ancestor of Y, in which case it never updates the ancestral relation of X on Y, and correctly returns definite non-ancestry. □

D.3 Proof of Lem. 4.2

Lemma 4.2. *For any two distinct nodes X and Y such that $X \in PossAn_G(Y)$ in a CPDAG G, the causal effect of X on Y is identifiable only if X is an explicit ancestor of Y in G.*

Proof. We prove this by contradiction. Assume that X is a possible but not an explicit ancestor of Y, so there are only possibly directed, but no directed paths from X to Y. Let p be the shortest possibly directed path from X to Y in G. According to Lem. D.1, there is a subsequence of p that is an unshielded possibly directed path from X to Y. However, since p is the shortest possibly directed path from X to Y, the only subsequence of p that could be a possibly directed path from X to Y is p itself. Hence, by Lem. D.1, p is an unshielded possibly directed path from X to Y. If p is unshielded, then there cannot be any colliders in p, otherwise there would be an unshielded collider $V \to V' \leftarrow V''$ in p, which would also be directed as such in the CPDAG G, and thus p would not be a possibly directed path in G since the edge $V' \leftarrow V''$ is directed backwards. Thus, every node on p has to be a non-collider in every DAG in the MEC represented by G.

However, since p is not a "fully" directed path in G, there is both a DAG D_1 in the MEC in which it is directed from X to Y, but also another DAG D_2 in the MEC in which it is not directed X to Y, but still open when the conditioning set is empty. Consequently, in D_1 no valid adjustment sets for X and Y can contain a node on p as it is a causal path from X to Y. However, in D_2 all valid adjustments set have to contain at least one node on p, as it is an otherwise open, but not causal path from X to Y. Hence, there is no shared set of nodes that is a valid adjustment set in all DAGs in the MEC of G and therefore the causal effect of X on Y is not identifiable in G. □

D.4 Proof of Lem. 4.3

We introduce the following lemma to aid us in the proof of Lem. 4.3.

Lemma D.4. *For any two distinct nodes X and Y such that $X \in PossAn_G(Y)$ in a CPDAG G, if G is adjustment amenable relative to X and Y, i.e., every possibly directed path from X to Y starts with a directed edge out of X, then for all $V \in Sib_G(X)$, $Y \notin Adj_G(V)$.*

Proof. Consider any $V \in Sib_G(X)$, i.e., $X - V$. The edges $V - Y$ or $V \to Y$ cannot be in G, since then there would be a possibly directed path from X to Y starting with V as $X - V - Y$ or $X - V \to Y$, which would make G not adjustment amenable relative to X and Y by definition, since the first edge is not directed out of X.

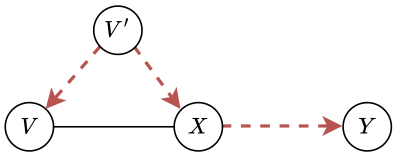
Since $X \in PossAn_G(Y)$, there is a a possibly directed path from X to Y. Then, if $V \leftarrow Y$ would be in G, there would be a partially directed cycle in G as $X \to \cdots Y \to V - X$, which is forbidden in CPDAGs as shown by Thm. D.2. Thus, V and Y cannot be adjacent. □

Lemma 4.3. *For any two distinct nodes X and Y such that $X \in PossAn_G(Y)$ in a CPDAG G, G is adjustment amenable relative to (X, Y) iff*

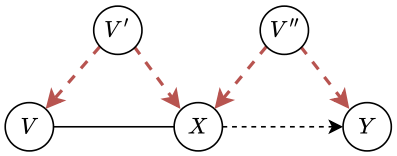
$$\forall V \in Sib_G(X) : V \perp\!\!\!\perp Y | Pa_G(V) \cup \{X\}.$$

Proof. We first consider the "if" direction and assume that $V \perp\!\!\!\perp Y | Pa_G(V) \cup \{X\}, \forall V \in Sib_G(X)$. We will now show that all possibly directed paths from X to Y start with a definite child of X, which implies that they start with a directed edge out of X, i.e., they are amenable relative to (X, Y) according to Perković et al. (2015). All nodes that are adjacent to X are either siblings, parents or children of X. Since a possibly directed path from X to Y cannot start with an edge into X, which happens if the first node on the path is a parent of X, we just need to show that the first node on all of this path cannot be a sibling of X.

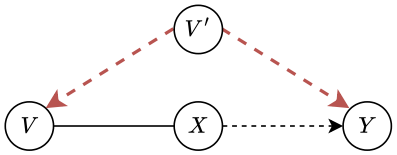
Consider any $V \in Sib_G(X)$. Assuming the faithfulness condition, V and Y are not adjacent, since they can be d-separated. If there are open paths between V and Y, conditioning on the definite parents $Pa_G(V)$ blocks paths that start with an edge into V, since by definition the definite parents cannot be colliders on those paths. If any other open path exists, i.e., starting either with an undirected or a directed edge out of V, since we assume it has to be blocked to have d-separation, and it cannot be blocked by $Pa_G(V)$, then it has to be blocked by X. This means X is a node on all of those paths. By definition, the possibly directed paths from V to Y are a subset of these paths, so they all contain X. This implies that no possibly directed path from X to Y goes through any sibling V (since then X would have to appear twice on the path). Since they also cannot go through a parent of



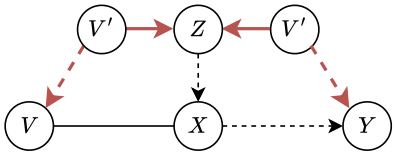
(a) Example path for case 1a.



(b) Example path for case 1b.



(c) Example path for case 2a.



(d) Example path for case 2b.

Figure 4: Example paths, shown in red, for the different cases considered in the "only if" direction of Lem. 4.3. The dashed arrows indicate that there can be a longer, possibly directed path between the two nodes instead of a direct edge. (a): A path p between V and Y that goes through X such that X is a non-collider on p. (b): A path p between V and Y that goes through X such that X is a collider on p. (c): A path p between V and Y that does not go through X and does not contain a node Z that is both an ancestor of X and a collider on p. (d): A path p between V and Y that does not go through X but contains a node Z that is both an ancestor of X and a collider on p.

X, this means that all possibly directed paths from X to Y start with a child of X. Thus, all possibly directed paths from X to Y start with an edge out X. By definition, this means that G is amenable relative to (X, Y).

For the other "only if" direction, we now assume that G is adjustment amenable relative to (X, Y), i.e., all possibly directed paths from X to Y start with a directed edge out of X. Consider any $V \in Sib_G(X)$. By Lem. D.4, V and Y are not adjacent and thus they can be d-separated, which implies a conditional independence given the appropriate separating set, since we assume the causal Markov assumption.

We now show that $Pa_G(V) \cup \{X\}$ blocks all paths between V and Y. To do this, we consider the following (exhaustive) cases of possible paths between V and Y and possible DAGs in the MEC represented by G that they can appear in, and show that in all of these cases the paths are blocked by $Pa_G(V) \cup \{X\}$ in the DAGs:

1. Any path p between V and Y that goes through X and
   (a) any DAG D in the MEC represented by G in which X is a non-collider on p (as shown in Fig. 4a);
   (b) any DAG D in the MEC represented by G in which X is a collider on p (as shown in Fig. 4b);
2. Any path p between V and Y that does not go through X and
   (a) any DAG D in the MEC represented by G in which p does not contain a node Z that is both an ancestor of X and a collider on p (as shown in Fig. 4c);
   (b) any DAG D in the MEC represented by G in which p contains a node Z that is both an ancestor of X and a collider on p (as shown in Fig. 4d);

Case 1a. Consider any path p between V and Y that goes through X in any DAG D in the MEC represented by G where X is a non-collider on p, such as the one shown in Fig. 4a. Then, p is blocked by conditioning on X regardless of whether we also condition on $Pa_G(V)$ or not. Thus, for any path p between V and Y that goes through X in any DAG D in the MEC represented by G in which X is a non-collider on p, p is blocked by $Pa_G(V) \cup \{X\}$.

Case 1b. Consider any path p between V and Y that goes through X in any DAG D in the MEC represented by G where X is a collider on p, such as the one shown in Fig. 4b. Since X is a collider on p in D then conditioning on X can potentially unblock p. In this case, we have to show that additionally conditioning on the definite parents $Pa_G(V)$ blocks p in D. If X is a collider on p in D and there are no other colliders on p, then the sub-path between X and Y on p in D must be a path between X and Y that starts with an edge into X. Since G is adjustment amenable relative to (X, Y) then, by Theorem 3.6 in (Perkovic, 2020), the causal effect of X on

Y is identifiable in G. In this case, by Cor. D.1, the set of definite parents $Pa_G(X)$ of X blocks all paths between X and Y starting with an edge into X in D, including p.

We can show that this also implies that $Pa_G(V)$ also blocks all paths between X and Y starting with an edge into X in D, including p, because $Pa_G(X) \subseteq Pa_G(V)$. To show this, consider any $P \in Pa_G(X)$. P and V must be adjacent in G, since the unshielded structure $P \to X - V$ cannot appear in a CPDAG, as shown by Thm. D.2. Furthermore, P cannot be a child of V, because then $V \to P \to X - V$ would create a partially directed cycle which is forbidden in CPDAGs, as shown by Thm. D.2. For the same reason, P also cannot be an sibling of V, as $P \to X - V - P$ also creates a partially directed cycle. This means that every $P \in Pa_G(X)$ has to also be a parent of V. Thus, for any path p between V and Y that go through X and any DAG D in the MEC represented by G in which X is a collider on p, p is blocked by $Pa_G(V) \cup \{X\}$.

Case 2. Before considering cases 2a and 2b, we first show that any path p between V and Y that does not go through X is blocked by $Pa_G(V)$ in G (and thus in all DAGs D in the MEC represented by G). We will then also have to show that this still holds even if we add X to the conditioning set.

The amenability of the causal effect of X on Y means that all possibly directed paths from X to Y start with an edge out of X. This implies that all possibly directed paths from V to Y have to go through X, otherwise there would exist a possibly directed path from X to Y through V starting with an undirected edge as $X - V \cdots Y$ in G. Thus, all paths between V and Y that do not go through X, including p, are not possibly directed paths from V to Y.

Now, consider G with all possibly directed paths from V to Y removed, denoted as G', which allows us to focus on the other, not possibly directed, paths, including p. Since by construction there are no possibly directed paths from V to Y in G', then V is a definite non-ancestor of Y in G'. Then, Thm. 4.2 states that $V \perp\!\!\!\perp Y | Pa_{G'}(V)$, i.e., all remaining paths between V and Y, including p, are blocked by $Pa_{G'}(V)$ in G'. Since the incoming edges into V in G and G' are the same, because we only removed the possibly directed paths from V to Y, which all start with an outgoing or undirected edge out of V, then by definition the definite parents of V are the same, i.e., $Pa_{G'}(V) = Pa_G(V)$. Hence, all paths that are not possibly directed from V to Y in G, including p, are blocked by $Pa_G(V)$. We now show that adding X to this conditioning set still keeps p blocked by considering the two cases in which p contains or does not contain a node Z that is both an ancestor of X and a collider on p.

Case 2a. Consider any DAG D in the MEC represented by G in which p does not contain a node Z that is both an ancestor of X and a collider on p, such as the one shown in Fig. 4c. Then, additionally conditioning on X does not open p. Thus, for any path p between V and Y that does not go through X and any DAG D in the MEC represented by G in which p does not contain a node Z that is both an ancestor of X and a collider on p, p is blocked by $Pa_G(V) \cup \{X\}$.

Case 2b. Consider any DAG D in the MEC represented by G in which p contains a node Z that is both an ancestor of X and a collider on p, such as the one shown in Fig. 4d. Then, additionally conditioning on X might open up p. In the following, we show that this does not happen. Since Z is an ancestor of X in the DAG D, it is a possible ancestor of X in the CPDAG G and there is a possibly directed path from Z to X in G. Then, Z cannot be a descendant of V in D, because then there would be a possibly directed path from V to Z in G, and thus a partially directed cycle $V \cdots Z \cdots X - V$ in G, which is forbidden in CPDAGs as shown by Thm. D.2. If Z cannot be a descendant of V in any D, then V is a definite non-ancestor of Z in G. Then, by Thm. 4.2, $V \perp\!\!\!\perp Z | Pa_G(V)$. This implies that the sub-path of p between V and Z is already blocked by $Pa_G(V)$ in G, and also by $Pa_G(V) \cup \{X\}$. Since we just need a subpath of p to be blocked by $Pa_G(V) \cup \{X\}$ for the whole path p to be blocked, this means that p is blocked by $Pa_G(V) \cup \{X\}$. Thus, for any path p between V and Y that does not go through X and any DAG D in the MEC represented by G in which p contains a node Z that is both an ancestor of X and a collider on p, p is blocked by $Pa_G(V) \cup \{X\}$.

In summary, $Pa_G(V) \cup \{X\}$ blocks all paths between V and Y in G. By the Causal Markov assumption, this means the conditional independence $V \perp\!\!\!\perp Y | Pa_G(V) \cup \{X\}$ holds for any sibling V of X. □

D.5 Proof of Cor. 4.1

Corollary 4.1. *LOAD is sound and complete in determining the identifiability of the causal effect between a pair of targets X and Y.*

Proof. LOAD begins by identifying the causal relations between the target pair using Alg. 1 (line 3). By Lem. 4.1, the identified causal relations are sound and complete for definite non-ancestry, possible ancestry and explicit ancestry. If no explicit ancestral relation is found in either direction, then by Lem. 4.2, the causal effect is not identifiable and LOAD correctly returns this, as well as the locally valid parent adjustment sets (lines 8 to 17).

If an explicit ancestral relation is found in either direction, then LOAD tests amenability (lines 19 to 23) by employing Alg. 2 for each sibling of the treatment (i.e., the target that is identified as the explicit ancestor of the other). Thm. D.1 ensures that the local information used for testing amenability is correct. The test of amenability fails if a sibling is found to be adjacent to the outcome, shown to be correct by Lem. D.4, or if Lem. 4.3 does not hold, in which case LOAD again returns the locally valid parent adjustment sets.

Thus, LOAD returns that the causal effect is identifiable if and only if the CPDAG is amenable relative to the target pair, i.e., if the causal effect between the targets is identifiable. □

D.6 Proof of Lem. 4.4

We first report a definition of a general projection for CPDAGs with single treatment and outcome based on a set $\mathbf{F}$, that is inspired by Def. 17 by Witte et al. (2020), but it focuses on CPDAGs and does not specify the set $\mathbf{F}$.

Definition D.1 (Projection for CPDAGs with single treatment and outcome given $\mathbf{F}$)**.** *Let G be a CPDAG with nodes $\mathbf{V}$, and let $T, O \in \mathbf{V}$, and $\mathbf{F} \subset \mathbf{V} \setminus \{T, O\}$. A projection $\hat{G}^{T,O}$ of G is a graph with nodes $\mathbf{V} \setminus \mathbf{F}$ and edges as follows. For distinct nodes $W_i, W_j \in \mathbf{V} \setminus \mathbf{F}$,*

1. *$\hat{G}^{T,O}$ contains a directed edge $W_i \to W_j$ if and only if G contains a directed path $W_i \to \cdots \to W_j$ on which all non-endpoint nodes are in $\mathbf{F}$,*
2. *$\hat{G}^{T,O}$ contains a bi-directed edge $Wi \leftrightarrow Wj$ if and only if G contains a path, with at least one non-endpoint node, of the form $W_i \leftarrow \cdots \to W_j$ on which all non-endpoints are non-colliders and in $\mathbf{F}$,*
3. *$\hat{G}^{T,O}$ contains an undirected edge $W_i - W_j$ if and only if G contains $W_i - W_j$.*

In our setting with a CPDAG with a single treatment and outcome, the original forbidden projection by Witte et al. (2020) is then the projection with $\mathbf{F} = PossDe_G(PossCn_G(T, O)) \setminus \{T, O\}$, where $PossCn_G(T, O)$ are the nodes on possibly directed paths from T to O, excluding T. If the causal effect between T and O is identifiable, i.e., the CPDAG G is amenable to single treatment and outcome (T, O), Prop. 19 and Lemma 20 in Witte et al. (2020) show that there are no bidirected edges in the projection, so the resulting graph is also a CPDAG. Moreover, the parents of the outcome except the treatment are the optimal adjustment set. We show that if we consider a larger set $\mathbf{D} = PossDe_G(T) \setminus \{T, O\}$ for the projection, the resulting graphs is also still a CPDAG and that the parents of the outcome except the treatment in this graph are still the optimal adjustment set.

Lemma 4.4. *For treatment T and outcome O in an amenable CPDAG G, let $\mathbf{D} = PossDe_G(T)) \setminus \{T, O\}$ and $G^{T,O}$ be the modified forbidden projection with nodes $\mathbf{V} \setminus \mathbf{D}$. We show that $G^{T,O}$ is a CPDAG and the optimal adjustment set $Oset_G(X, Y)$ is given by*

$$Oset_G(T, O) = Pa_{G^{T,O}}(O) \setminus \{T\}.$$

Proof. We start by applying a forbidden projection for treatment T and outcome O on the original amenable CPDAG G and getting a graph $\tilde{G}^{T,O}$ with a node set $\mathcal{V}$. As shown in Prop. 22 by Witte et al. (2020) this is still a CPDAG, but without any possible causal nodes and any possible descendants of O (which includes its siblings), except O itself. The optimal adjustment set are the parents of T except T in $\tilde{G}^{T,O}$. We can further project out the remaining variables in $\tilde{G}^{T,O}$, $\Delta = PossDe_G(T) \setminus PossDe_G(PossCn_G(T, O)) \cup \{T, O\}$. These will be still the remaining possible descendants of T in this new graph, i.e., $PossDe_{\tilde{G}^{T,O}}(T)$, which are definite non-ancestors of O and its possible ancestors, e.g., its parents, in both graphs, since otherwise they would be possible causal nodes in the original graph G and removed in the first projection. As such, following Schubert et al. (2025), they can be safely marginalized out in $G^{T,O}$ without changing any orientation in the graph regarding O, its possible ancestors, e.g., parents, i.e., without changing the parent set of O, and hence the optimal adjustment set.

To prove that $G^{T,O}$ is still a CPDAG, we first show that it does not contain bi-directed edges, and then show that we do not add any other edge to the remaining variables with respect to the forbidden projection $\tilde{G}^{T,O}$.

In order for a bi-directed edge to appear in $G^{T,O}$, there have to exist two nodes $W_i, W_j \notin \mathbf{D}$ such that $W_i \leftarrow \cdots \to W_j$ in G and all non-endpoint nodes on this path are non-colliders and in $\mathbf{D}$. This means that both

W_i and W_j have to be possible descendants of a node in $\mathbf{D}$. By transitivity, possible descendants of nodes in $\mathbf{D}$ are also in $\mathbf{D}$, except for T and O, which are then the only possible candidates for W_i and W_j. On the other hand, there cannot exists such a path $W_i = T \leftarrow \cdots \rightarrow O = W_j$ in G such that all non-endpoint nodes on this path are by definition of $\mathbf{D}$ also possible descendants of T, since this would be a contradiction based on the orientation of the path.

We now show that the modified projection does not add any other directed or undirected edge compared to the original forbidden projection $\tilde{G}^{T,O}$. A fully undirected path or (possibly) directed path from T to other variables $V \in \mathcal{V} \setminus \{\Delta \cup O, T\}$ that passes through some variable $D \in \Delta$ would imply that V is also in Δ by definition of possible descendants, which is a contradiction. Moreover, it is not possible to have a directed path from V to T through some $D \in \Delta$, since this would contradict the fact that Δ are possible descendants of T.

We will therefore consider the case when there is a possibly directed path $p = (V, \ldots, D_1, \ldots, D_k, \ldots, T)$ in $\mathcal{G}$ for $V \in \mathcal{V} \setminus \{\Delta \cup \{O, T\}\}$ where all non-endpoint nodes $D_1, \ldots, D_k$ are in Δ. The subpath p' from $D_1 - D_2 \cdots - D_k - T$ has to undirected, since it cannot be possibly directed from any D_i to T by definition of Δ and it cannot be possibly directed in the other direction by assumption. Additionally, the subpath p'' from $V \ldots D_1$ cannot be completely undirected, otherwise V would be part of Δ. In order for p' to stay undirected, we need that the variable V_j on p'' closest to D_1 that has an orientation is part of a shielded triple $V_i \rightarrow V_j - D_1$ with additionally $V_i - D_1$ or $V_i \rightarrow D_1$, otherwise the orientation will propagate further due to Meek's rules (Meek, 1995), but we cannot use $V_i - D_1$, since this would mean that $V_i \in \Delta$, because it is connected by an undirected path to T, which is a contradiction. So the only possible shield is $V_i \rightarrow D_1$, but then since we have $V_i \rightarrow D_1 - D_2$ this would imply that there needs to be another shield $V_i \rightarrow D_2$, on which we can apply the same argument on each node D_i, until we would get a shield $V_i \rightarrow T$. This means that in the projection we would not need to add any edge from V_i to T. A simplified case is when we only have three variables $V \rightarrow D - T$, which would require $V - T$, which is again a contradiction, or $V \rightarrow T$ which would mean we do not need to add a new edge after the projection. The only possible path from T to other variables $V \in \mathcal{V} \setminus \{\Delta \cup O, T\}$ that passes through some variable $D \in \Delta$ would have D as a collider, which means we can safely remove this variable and still have a valid CPDAG, which means we do not need to add any edge to the original $\tilde{G}^{T,O}$.

□

D.7 Proof of Thm. 4.3

Theorem 4.3. *LOAD is sound and complete in finding optimal adjustment sets for a pair of target variables.*

Proof. By Thm. 3.13 of (Henckel et al., 2022), the optimal adjustment set is identifiable iff the causal effect between T and O is identifiable in the full CPDAG G. Cor. 4.1 shows that LOAD is sound and complete in determining the identifiability of causal effects.

When the causal effect of T on O is identifiable, we show that LOAD returns the correct optimal adjustment set. To construct the estimated optimal adjustment, LOAD identifies the possible descendants of T using Alg. 9 (lines 25 to 28). Since Alg. 9 is sound and complete in finding possible ancestral relations as shown in Lem. D.3, these are exactly $PossDe_G(X)$.

When the causal effect of T on O is identifiable, then, by Lem. 4.4, the modified forbidden projection $G^{T,O} = G((\mathbf{V} \setminus PossDe_G(T)) \cup \{T, O\})$ is also a CPDAG and thus we can perform local causal discovery using MB-by-MB. We perform local causal discovery on the outcome O in the modified forbidden projection $G^{T,O}$ at line 29 to identify the parents of O in the modified forbidden projection $G^{T,O}$ i.e., $Pa_{G^{T,O}}(O)$. Then, as shown by Witte et al. (2020), $Pa_{G^{T,O}}(O) \setminus \{T\}$ is exactly the optimal adjustment set. The correctness of the local information throughout all of these steps is ensured by Thm. D.1. Thus, LOAD is sound and complete in determining the optimal adjustment set. □

E POTENTIAL FURTHER OPTIMIZATIONS

Here we describe a set of potential further optimizations to our method that we did not use in the main paper, due to the limited improvement in our experimental setting. The LocalRelate (Alg. 1) and LOAD (Alg. 3) algorithms run the local causal discovery sub-routine sequentially on several target variables with information cached and shared between different executions as shown by our implementation of MB-by-MB in Alg. 11. Execution time

can be further improved by running local causal discovery on multiple target nodes in parallel with the same shared memory already implemented. In particular, local causal discovery can be run in parallel for the two target variables by Alg. 1 at lines 2 and 3. Furthermore, local causal discovery on all siblings of the treatment in Step 3 of LOAD at line 19 can also be fully parallelized. Note, however, that parallelizing Step 3 can result in performing more CI tests, as the current implementation terminates early when amenability fails.

Another potential optimization involves the last step of LOAD at line 29, which runs local causal discovery around the outcome over the modified forbidden projection. We conjecture that instead of local causal discovery, recovering only the Markov blanket of the outcome already identifies the parents, and thus the optimal adjustment set, without having to orient any edges.

Lemma E.1. *For a treatment T and outcome O in a graph G, the Markov blanket of O in the modified forbidden projection $G^{T,O}$ directly identifies the optimal adjustment as*

$$Oset_G(T, O) = MB_{G^{T,O}}(O) \setminus \{T\}.$$

Proof. In Lem. 4.4 we showed that $Oset_G(T, O) = Pa_{G^{T,O}}(O) \setminus \{T\}$. Thus, we need to show that $Pa_{G^{T,O}}(O) = MB_{G^{T,O}}(O)$. It holds that $Pa_{G^{T,O}}(O) \subseteq MB_{G^{T,O}}(O)$ by definition.

For the other direction, we have to show that $MB_{G^{T,O}}(O) \subseteq Pa_{G^{T,O}}(O)$. Since $G^{T,O}$ is the modified forbidden projection, it does not contain any possible descendants of T. Hence, O has no children or undirected siblings in $G^{T,O}$ and thus $MB_{G^{T,O}}(O)$ also does not contain any children or undirected siblings of O in $G^{T,O}$. Furthermore, if O has no children or undirected siblings in $G^{T,O}$, then it also does not have co-parents in $G^{T,O}$. This means that $MB_{G^{T,O}}(O)$ can only contain the parents of O in $G^{T,O}$, i.e., $MB_{G^{T,O}}(O) \subseteq Pa_{G^{T,O}}(O)$. □

Lem. E.1 allows us to run Markov blanket discovery once on the outcome variable at line 29 instead of possibly on multiple variables through MB-by-MB. On the other hand, in our experiments, the computational gain is very limited, so we did not implement this strategy in the main paper, also because this might make any potential extension to the causally insufficient case more complicated.

F EXPERIMENTAL DETAILS

For our experiments, we used the following libraries: igraph (Csardi and Nepusz, 2006) (GNU GPL version 2 or later), networkx (Hagberg et al., 2008) (3-Clause BSD License), bnlearn (Scutari, 2010) (MIT License), pcalg (Kalisch et al., 2012) (GNU GPL version 2 or later), dagitty (Textor et al., 2016) (GNU GPL), causal-learn (Zheng et al., 2024) (MIT License) and RCD (Mokhtarian et al., 2024) (BSD 2-Clause License). In particular, we used the CI test implementations from the causal-learn package (Zheng et al., 2024).

All experiments were performed on AMD Rome CPUs, using 48 CPU cores and 84 GiB of memory. We let each experiment run for at most 24 hours. If an experiment over 100 seeds did not finish in the given time or did not fit in the given memory, then we do not report any results for it.

F.1 Real-world networks

The MAGIC-NIAB network has 44 nodes, an average degree 3, and is parameterized by linear Gaussian data. We show the network structure of MAGIC-NIAB in Fig. 5. The Andes network has 223 nodes, an average degree 3.03, and is parameterized by binary data. We show the network structure of Andes in Fig. 6.

G EXTENDED RESULTS

We repeat the main results presented in Fig. 1 in tables with numbers shown explicitly for additional clarity, due to several overlapping results. Tab. 2 reports the main results for d-separation, Tab. 3 for Fisher-Z tests and Tab. 4 KCI tests on linear Gaussian data, and Tab. 5 for G^2 tests on binary data.

On top of the metrics already shown in Fig. 1, the tables also report the computation time of all algorithms over all experiments. In the d-separation setting, the computation times show the same trends as the number of CI tests. However, when using Fisher-Z or G^2 tests, the computation time of MARVEL increases substantially and the gap between the local methods (MB-by-MB, LDECC and LDP) and methods like PC and SNAP(∞) gets

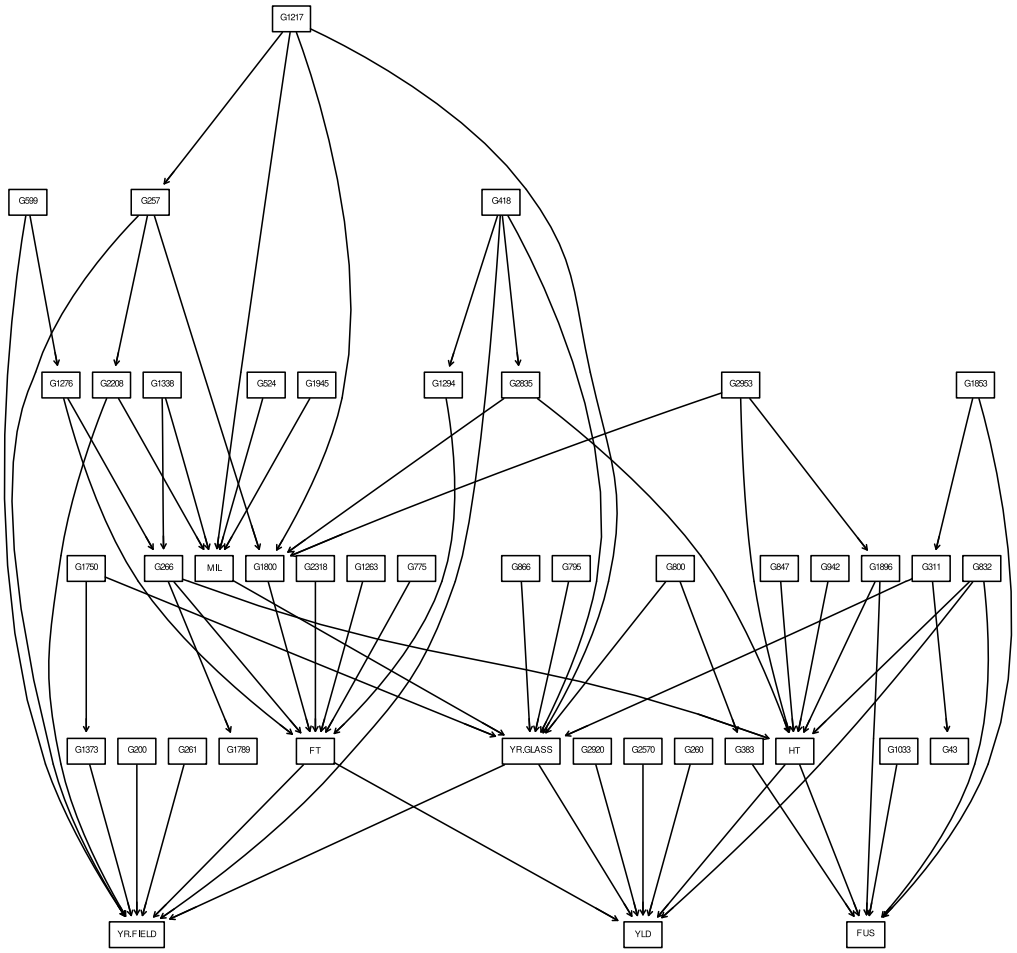


Figure 5: The MAGIC-NIAB network.

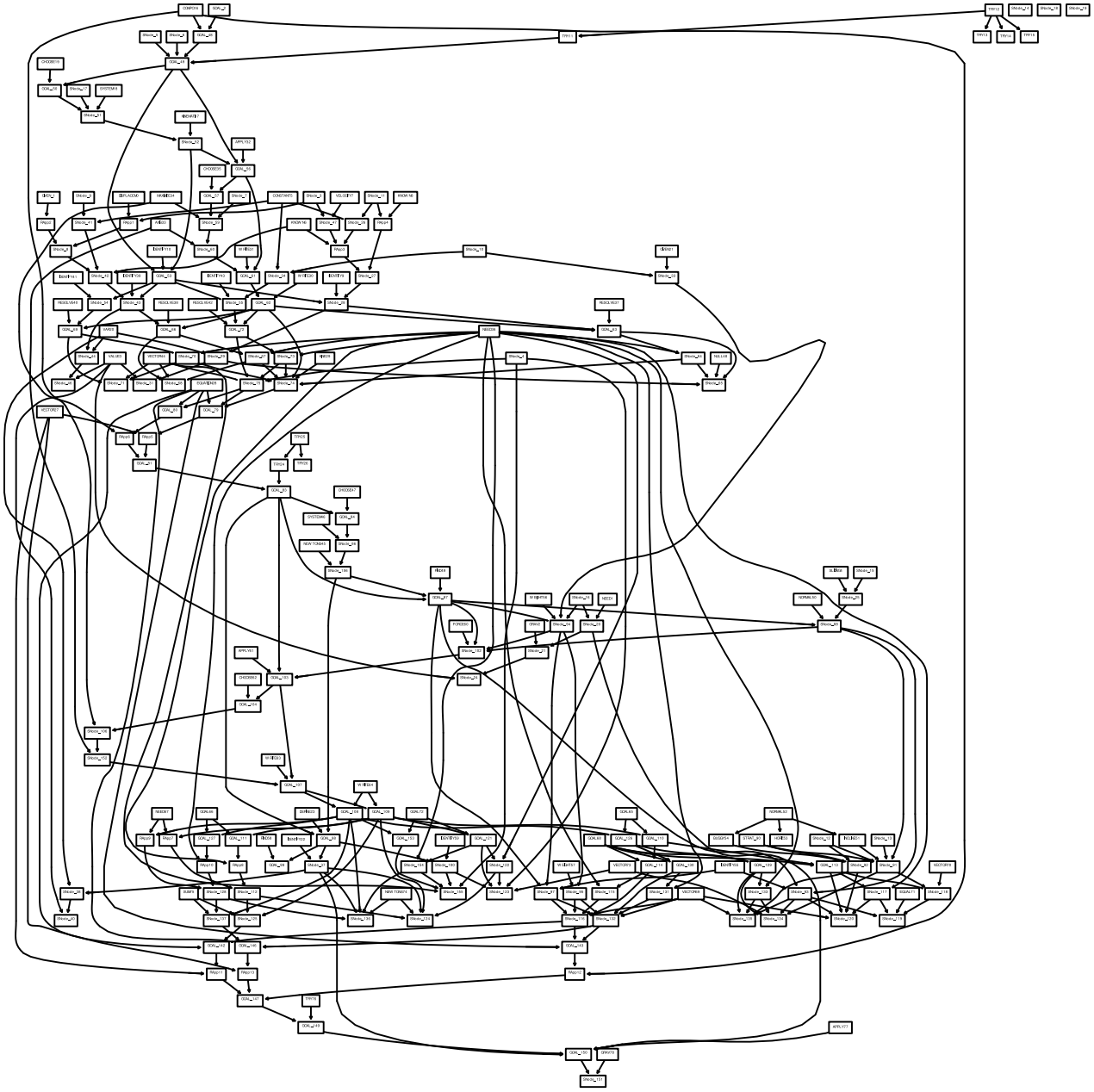


Figure 6: The Andes network.

smaller with the increasing number of nodes. LOAD performs as expected in-between local and global methods for lower numbers of nodes, but is slower than global methods on extremely large graphs. This is due to MARVEL, MB-by-MB$^+$, LDECC$^+$ and LOAD all employing some form of Markov blanket search, which generally perform CI tests with large conditioning set sizes and require more computations.

Table 2: Results over number of nodes with $n_{\mathbf{D}} = 10000$, $\overline{d} = 2$ and $d_{\max} = 10$ and target pairs such that one is an explicit ancestor of the other, for d-separation CI tests.

	d-separation CI tests					
# Nodes	100	200	400	600	800	1000
	Number of CI tests ($\times 10^3$) ↓					
PC	9.00 ± 1.46	28.61 ± 3.04	96.81 ± 4.78	205.14 ± 5.19	353.48 ± 5.51	542.52 ± 6.66
MB-by-MB$^+$	0.66 ± 0.30	1.14 ± 0.43	2.20 ± 0.71	3.16 ± 1.11	4.84 ± 1.86	5.64 ± 1.98
MARVEL	5.48 ± 0.14	21.11 ± 0.37	82.07 ± 0.50	-	-	-
LDECC$^+$	7.95 ± 6.90	23.00 ± 23.59	100.55 ± 86.36	179.14 ± 188.47	400.75 ± 324.45	566.69 ± 508.87
LDP$^+$	0.88 ± 0.34	1.51 ± 0.47	2.96 ± 0.75	4.25 ± 1.15	6.29 ± 1.91	7.45 ± 2.06
SNAP(∞)	5.11 ± 0.18	20.02 ± 0.14	79.98 ± 0.17	179.89 ± 0.21	319.74 ± 0.13	499.67 ± 0.15
LOAD	0.86 ± 0.36	1.49 ± 0.53	2.94 ± 0.92	4.36 ± 1.39	6.41 ± 2.23	7.63 ± 2.47
	Computation Time (s) ↓					
PC	3.67 ± 0.71	19.67 ± 2.03	117.25 ± 11.66	334.99 ± 44.21	771.44 ± 43.63	1334.79 ± 200.84
MB-by-MB$^+$	0.15 ± 0.07	0.42 ± 0.15	1.41 ± 0.46	3.11 ± 1.02	5.53 ± 2.13	8.13 ± 2.68
MARVEL	2.06 ± 0.30	13.23 ± 2.70	113.14 ± 11.77	-	-	-
LDECC$^+$	2.71 ± 2.26	12.87 ± 11.98	99.77 ± 82.67	210.67 ± 212.49	725.48 ± 575.05	1120.81 ± 993.60
LDP$^+$	0.58 ± 0.19	2.44 ± 0.42	7.69 ± 1.95	20.83 ± 3.59	50.77 ± 15.00	91.10 ± 19.53
SNAP(∞)	3.07 ± 0.27	14.42 ± 2.33	93.44 ± 22.16	313.71 ± 40.02	786.31 ± 35.68	1514.65 ± 65.48
LOAD	0.21 ± 0.08	0.55 ± 0.19	1.99 ± 0.61	4.15 ± 1.24	7.48 ± 2.57	12.31 ± 3.46
	F1 score of Oset ↑					
PC	1.00 ± 0.00	1.00 ± 0.00	1.00 ± 0.00	1.00 ± 0.00	1.00 ± 0.00	1.00 ± 0.00
MB-by-MB$^+$	0.00 ± 0.00	0.00 ± 0.00	0.00 ± 0.00	0.00 ± 0.00	0.00 ± 0.00	0.00 ± 0.00
MARVEL	1.00 ± 0.00	1.00 ± 0.00	1.00 ± 0.00	-	-	-
LDECC$^+$	0.00 ± 0.00	0.00 ± 0.00	0.00 ± 0.00	0.00 ± 0.00	0.00 ± 0.00	0.00 ± 0.00
LDP$^+$	0.01 ± 0.04	0.00 ± 0.00	0.00 ± 0.00	0.00 ± 0.00	0.00 ± 0.00	0.00 ± 0.00
SNAP(∞)	1.00 ± 0.00	1.00 ± 0.00	1.00 ± 0.00	1.00 ± 0.00	1.00 ± 0.00	1.00 ± 0.00
LOAD	1.00 ± 0.00	1.00 ± 0.00	1.00 ± 0.00	1.00 ± 0.00	1.00 ± 0.00	1.00 ± 0.00
	Intervention distance ↓					
PC	0.004 ± 0.004	0.004 ± 0.003	0.003 ± 0.003	0.003 ± 0.003	0.003 ± 0.002	0.003 ± 0.003
MB-by-MB$^+$	0.031 ± 0.035	0.019 ± 0.018	0.032 ± 0.034	0.026 ± 0.032	0.030 ± 0.031	0.031 ± 0.030
MARVEL	0.004 ± 0.004	0.004 ± 0.003	0.003 ± 0.003	-	-	-
LDECC$^+$	0.031 ± 0.035	0.019 ± 0.018	0.032 ± 0.034	0.026 ± 0.032	0.030 ± 0.031	0.031 ± 0.030
LDP$^+$	0.011 ± 0.012	0.007 ± 0.005	0.008 ± 0.008	0.009 ± 0.010	0.009 ± 0.009	0.010 ± 0.010
SNAP(∞)	0.004 ± 0.004	0.004 ± 0.003	0.003 ± 0.003	0.003 ± 0.003	0.003 ± 0.002	0.003 ± 0.003
LOAD	0.004 ± 0.004	0.004 ± 0.003	0.003 ± 0.003	0.003 ± 0.003	0.003 ± 0.002	0.003 ± 0.003

H ABLATIONS

H.1 CI Testing Errors

Our results in Fig. 1 and App. G indicate that identifying the optimal adjustment set is much more challenging in the binary data setting compared to linear Gaussian data for all algorithms. This can be attributed to the fact that Fisher-Z tests can more easily compute partial correlation based on the inverse of the sample covariance matrix, which is not possible for the G^2 tests on binary data. Instead, tests for discrete data, such as the G^2 test, have to use the observed counts of samples with a given combination of values for all of the variables in the conditioning set for each test, resulting in higher sample complexity. Intuitively, the larger this set is, the smaller the number of samples for each combination, which will give us less accurate results with lower sample sizes

To show empirically that in our settings the G^2 tests are less accurate than Fisher-Z tests, we perform an ablation

Table 3: Results over number of nodes with $n_{\mathbf{D}} = 10000$, $\overline{d} = 2$ and $d_{\max} = 10$ and target pairs such that one is an explicit ancestor of the other, for Fisher-Z CI tests on linear Gaussian data.

	Fisher-Z CI tests					
# Nodes	100	200	400	600	800	1000
	Number of CI tests ($\times 10^3$) ↓					
PC	8.28 ± 0.95	27.16 ± 1.75	96.02 ± 2.49	206.76 ± 3.04	359.64 ± 3.42	555.47 ± 4.69
MB-by-MB$^+$	0.75 ± 0.35	1.37 ± 0.53	2.75 ± 0.77	4.39 ± 1.51	6.53 ± 2.40	8.20 ± 3.00
MARVEL	5.65 ± 0.16	21.99 ± 0.42	87.64 ± 1.08	-	-	-
LDECC$^+$	4.93 ± 4.86	16.29 ± 18.11	61.89 ± 69.00	109.95 ± 143.96	233.69 ± 273.58	
LDP$^+$	0.96 ± 0.37	1.75 ± 0.55	3.56 ± 0.77	5.64 ± 1.54	8.11 ± 2.44	10.23 ± 3.02
SNAP(∞)	5.01 ± 0.06	19.93 ± 0.04	79.81 ± 0.01	179.70 ± 0.01	319.60 ± 0.00	499.50 ± 0.00
LOAD	0.95 ± 0.40	1.79 ± 0.62	3.77 ± 0.95	6.10 ± 1.83	8.77 ± 2.78	11.15 ± 3.37
	Computation Time (s) ↓					
PC	1.54 ± 0.16	5.08 ± 0.28	18.66 ± 0.43	40.85 ± 0.51	72.80 ± 0.62	117.75 ± 0.93
MB-by-MB$^+$	0.14 ± 0.07	0.26 ± 0.10	0.65 ± 0.32	0.93 ± 0.35	1.46 ± 0.60	1.91 ± 0.72
MARVEL	1.25 ± 0.04	7.97 ± 0.10	77.41 ± 0.40	-	-	-
LDECC$^+$	1.21 ± 1.16	4.76 ± 5.14	24.23 ± 26.15	54.63 ± 69.10	134.30 ± 152.48	-
LDP$^+$	0.24 ± 0.08	0.68 ± 0.19	3.22 ± 0.83	9.29 ± 3.06	22.78 ± 8.02	43.69 ± 15.19
SNAP(∞)	1.02 ± 0.04	4.26 ± 0.08	18.16 ± 0.51	43.38 ± 1.32	82.60 ± 2.82	137.85 ± 5.39
LOAD	0.18 ± 0.08	0.35 ± 0.13	0.76 ± 0.19	1.38 ± 0.46	1.90 ± 0.66	2.52 ± 0.77
	F1 score of Oset ↑					
PC	0.74 ± 0.40	0.80 ± 0.37	0.85 ± 0.32	0.69 ± 0.41	0.60 ± 0.46	0.66 ± 0.43
MB-by-MB$^+$	0.00 ± 0.00	0.00 ± 0.00	0.00 ± 0.00	0.00 ± 0.00	0.00 ± 0.00	0.00 ± 0.00
MARVEL	0.91 ± 0.27	0.92 ± 0.25	0.81 ± 0.37	-	-	-
LDECC$^+$	0.00 ± 0.00	0.00 ± 0.00	0.00 ± 0.00	0.00 ± 0.00	0.00 ± 0.00	-
LDP$^+$	0.00 ± 0.01	0.00 ± 0.00	0.00 ± 0.00	0.00 ± 0.00	0.00 ± 0.00	0.00 ± 0.00
SNAP(∞)	0.56 ± 0.41	0.31 ± 0.38	0.16 ± 0.30	0.03 ± 0.11	0.00 ± 0.02	0.00 ± 0.00
LOAD	0.97 ± 0.12	0.99 ± 0.06	0.98 ± 0.09	0.94 ± 0.18	0.95 ± 0.15	0.96 ± 0.13
	Intervention distance ↓					
PC	0.152 ± 0.313	0.038 ± 0.085	0.053 ± 0.147	0.078 ± 0.188	0.099 ± 0.225	0.131 ± 0.290
MB-by-MB$^+$	0.092 ± 0.081	0.067 ± 0.083	0.120 ± 0.130	0.094 ± 0.112	0.086 ± 0.096	0.113 ± 0.114
MARVEL	0.032 ± 0.057	0.014 ± 0.017	0.036 ± 0.076	-	-	-
LDECC$^+$	0.181 ± 0.312	0.080 ± 0.100	0.202 ± 0.293	0.189 ± 0.290	0.132 ± 0.201	-
LDP$^+$	0.037 ± 0.044	0.023 ± 0.025	0.037 ± 0.045	0.029 ± 0.037	0.029 ± 0.030	0.041 ± 0.042
SNAP(∞)	0.111 ± 0.194	0.145 ± 0.199	0.365 ± 0.327	0.479 ± 0.263	0.594 ± 0.321	0.586 ± 0.321
LOAD	0.010 ± 0.030	0.003 ± 0.003	0.004 ± 0.006	0.004 ± 0.004	0.005 ± 0.007	0.005 ± 0.005

study under the same setting as Fig. 1 with 100, 200 and 400 nodes, where we report the percentage of erroneous CI tests results. Our results in Fig. 7a show that G^2 tests indeed have indeed many more errors than Fisher-Z tests in our data.

H.2 Errors in Local Neighborhoods

The performance of LOAD decreases slightly with more structural errors in the local neighborhoods estimated by the local causal discovery subroutine. We show this empirically in Fig. 7b where we report the average number of erroneous relations in the estimated local neighborhoods (i.e., the size of the symmetric difference between the estimated and the true parents, children and undirected neighbors). We evaluate for both Fisher-Z tests on linear Gaussian data and G^2 tests on binary data, over various samples sizes and $n_{\mathbf{V}} = 200$, $\overline{d} = 2$ and $d_{\max} = 10$. As expected, fewer errors in the local neighborhoods lead to better recovery of the optimal adjustment set (Fig. 7c). Notably, even with an average of almost two erroneous node in every estimation of a local neighborhood for 500 samples with Fisher-Z CI tests, the F1 score (and intervention distance as shown in Fig. 14) of LOAD is still better than most baselines.

Table 4: Results over number of nodes with $n_{\mathbf{D}} = 1000$, $\overline{d} = 2$ and $d_{\max} = 10$ and target pairs such that one is an explicit ancestor of the other, for KCI CI tests on binary data.

	KCI tests		
# Nodes	10	15	20
	Number of CI tests ↓		
PC	134.33 ± 31.11	282.21 ± 70.97	454.97 ± 86.31
MB-by-MB$^+$	118.46 ± 46.16	194.31 ± 95.69	211.46 ± 92.08
MARVEL	85.41 ± 22.67	196.51 ± 49.77	322.07 ± 62.74
LDECC$^+$	160.88 ± 80.08	296.03 ± 164.97	384.71 ± 220.95
LDP$^+$	136.20 ± 47.53	232.32 ± 104.91	258.59 ± 94.33
SNAP(∞)	92.52 ± 35.49	156.89 ± 43.81	242.79 ± 53.92
LOAD	127.82 ± 47.10	211.42 ± 99.53	239.56 ± 93.61
	Computation Time (s) ↓		
PC	50.37 ± 16.94	98.77 ± 38.19	223.26 ± 70.62
MB-by-MB$^+$	56.27 ± 22.72	126.40 ± 64.11	105.36 ± 46.32
MARVEL	43.77 ± 10.80	100.21 ± 23.77	180.03 ± 29.99
LDECC$^+$	71.71 ± 40.32	126.50 ± 81.87	159.61 ± 97.09
LDP$^+$	65.01 ± 25.20	105.60 ± 52.09	119.03 ± 48.46
SNAP(∞)	26.98 ± 18.91	33.02 ± 24.64	36.24 ± 30.05
LOAD	62.11 ± 24.26	101.09 ± 49.99	162.64 ± 67.13
	F1 score of Oset ↑		
PC	0.52 ± 0.46	0.32 ± 0.45	0.52 ± 0.47
MB-by-MB$^+$	0.02 ± 0.13	0.06 ± 0.20	0.00 ± 0.00
MARVEL	0.74 ± 0.41	0.44 ± 0.48	0.55 ± 0.48
LDECC$^+$	0.03 ± 0.14	0.03 ± 0.16	0.02 ± 0.11
LDP$^+$	0.14 ± 0.34	0.15 ± 0.35	0.11 ± 0.31
SNAP(∞)	0.52 ± 0.47	0.35 ± 0.45	0.40 ± 0.46
LOAD	0.74 ± 0.41	0.61 ± 0.47	0.75 ± 0.41
	Intervention distance ↓		
PC	0.35 ± 0.44	0.51 ± 0.43	0.37 ± 0.47
MB-by-MB$^+$	0.20 ± 0.33	0.29 ± 0.38	0.18 ± 0.31
MARVEL	0.18 ± 0.35	0.26 ± 0.32	0.27 ± 0.42
LDECC$^+$	0.29 ± 0.38	0.41 ± 0.45	0.29 ± 0.45
LDP$^+$	0.15 ± 0.29	0.24 ± 0.33	0.16 ± 0.30
SNAP(∞)	0.34 ± 0.39	0.51 ± 0.41	0.38 ± 0.42
LOAD	0.17 ± 0.32	0.28 ± 0.39	0.13 ± 0.29

H.3 Performance of Steps

In this section we study the performance of the different steps of LOAD. In general, the correctness of the first two steps should highly influence the performance of later steps, since if LOAD identifies the wrong causal relations between the targets in Step 1 or make a mistake in identifying if a causal effect is identifiable in Step 2, this will change the flow of the complete algorithm. Similarly Steps 3 and 4 which together identify mediators are crucial to identifying the correct optimal adjustment set. On the other hand, we assume that the last 2 steps are less important in terms of the accuracy of the causal effect estimation because in principle the output could still be potentially a valid set, which would allow the causal effect estimation to still be unbiased, although with a higher variance.

We empirically test this using linear Gaussian data with different sample sizes to study different error rates in each phase. We report the precision and recall scores of each step. In particular, we measure how well Step 1 can recover the causal relation between the targets, Step 2 can determine identifiability, how well Steps 3 can identify possible descendants, and finally how Step 4 can recover the optimal adjustment set. In case LOAD determines the causal relation or the identifiability of the causal effect incorrectly, then later steps automatically have a precision and recall of 0.

Our results in Fig. 8 show that recovering all and only the correct possible descendants of the treatment in Step 3

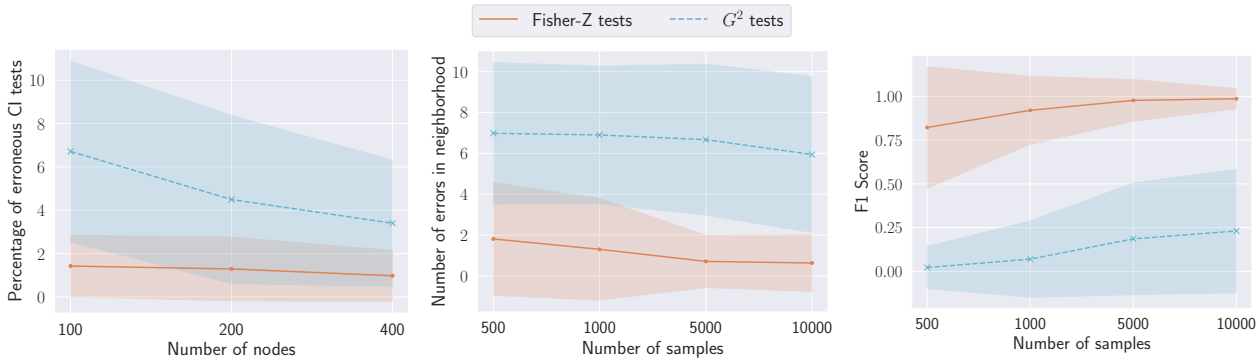
Table 5: Results over number of nodes with $n_{\mathbf{D}} = 10000$, $\overline{d} = 2$ and $d_{\max} = 10$ and target pairs such that one is an explicit ancestor of the other, for G^2 CI tests on binary data.

	G^2 CI tests					
# Nodes	100	200	400	600	800	1000
	Number of CI tests ($\times 10^3$) ↓					
PC	6.15 ± 0.21	22.77 ± 0.37	87.79 ± 0.64	195.23 ± 0.77	345.68 ± 0.90	539.23 ± 1.48
MB-by-MB$^+$	1.04 ± 0.25	1.71 ± 0.49	2.56 ± 0.65	3.20 ± 0.78	3.95 ± 0.95	4.71 ± 1.04
MARVEL	4.95 ± 0.00	19.90 ± 0.00	79.80 ± 0.00	-	-	-
LDECC$^+$	-	-	-	-	-	-
LDP$^+$	1.13 ± 0.25	1.93 ± 0.47	3.03 ± 0.73	3.96 ± 0.93	5.11 ± 1.19	6.23 ± 1.21
SNAP(∞)	4.95 ± 0.00	19.90 ± 0.00	79.80 ± 0.00	179.70 ± 0.00	319.60 ± 0.00	499.50 ± 0.00
LOAD	1.19 ± 0.33	2.00 ± 0.55	3.08 ± 0.91	4.25 ± 1.24	5.19 ± 1.50	6.10 ± 1.47
	Computation Time (s) ↓					
PC	3.93 ± 0.57	13.84 ± 2.75	57.46 ± 17.63	129.28 ± 11.20	229.50 ± 37.73	378.14 ± 24.23
MB-by-MB$^+$	1.87 ± 0.57	4.16 ± 1.26	8.99 ± 2.71	16.22 ± 5.43	16.29 ± 3.78	20.66 ± 4.55
MARVEL	153.59 ± 26.30	1068.25 ± 206.65	9074.14 ± 800.29	-	-	-
LDECC$^+$	-	-	-	-	-	-
LDP$^+$	3.16 ± 0.73	7.85 ± 1.49	19.39 ± 2.91	32.80 ± 8.03	64.22 ± 9.30	104.63 ± 17.39
SNAP(∞)	2.95 ± 0.41	13.02 ± 1.34	55.38 ± 4.29	130.66 ± 6.99	249.79 ± 6.83	382.60 ± 48.54
LOAD	2.03 ± 0.74	4.65 ± 1.38	9.35 ± 2.87	15.24 ± 4.40	19.83 ± 5.83	24.64 ± 6.55
	F1 score of Oset ↑					
PC	0.34 ± 0.42	0.31 ± 0.40	0.28 ± 0.40	0.21 ± 0.36	0.26 ± 0.40	0.19 ± 0.34
MB-by-MB$^+$	0.00 ± 0.00	0.00 ± 0.00	0.00 ± 0.00	0.00 ± 0.00	0.00 ± 0.00	0.00 ± 0.00
MARVEL	0.00 ± 0.00	0.00 ± 0.00	0.00 ± 0.00	-	-	-
LDECC$^+$	-	-	-	-	-	-
LDP$^+$	0.00 ± 0.00	0.00 ± 0.00	0.00 ± 0.00	0.00 ± 0.00	0.00 ± 0.00	0.00 ± 0.00
SNAP(∞)	0.09 ± 0.23	0.00 ± 0.00	0.00 ± 0.00	0.00 ± 0.00	0.00 ± 0.00	0.00 ± 0.00
LOAD	0.39 ± 0.44	0.23 ± 0.35	0.24 ± 0.39	0.13 ± 0.27	0.14 ± 0.32	0.06 ± 0.17
	Intervention distance ↓					
PC	0.018 ± 0.033	0.017 ± 0.027	0.022 ± 0.042	0.019 ± 0.036	0.016 ± 0.029	0.021 ± 0.037
MB-by-MB$^+$	0.022 ± 0.041	0.026 ± 0.038	0.033 ± 0.055	0.031 ± 0.056	0.038 ± 0.058	0.035 ± 0.052
MARVEL	0.067 ± 0.059	0.055 ± 0.045	0.067 ± 0.064	-	-	-
LDECC$^+$	-	-	-	-	-	-
LDP$^+$	0.022 ± 0.041	0.025 ± 0.037	0.032 ± 0.054	0.031 ± 0.054	0.038 ± 0.061	0.036 ± 0.053
SNAP(∞)	0.045 ± 0.055	0.050 ± 0.046	0.063 ± 0.059	0.054 ± 0.057	0.058 ± 0.055	0.054 ± 0.046
LOAD	0.021 ± 0.040	0.024 ± 0.037	0.032 ± 0.055	0.031 ± 0.057	0.039 ± 0.061	0.036 ± 0.052

is the biggest challenge for LOAD. Note, that this is generally not because LOAD determines a wrong causal relation or effect identifiability, as the precision and recall of Steps 1 and 2 are high. In fact, Step 2 achieves a perfect precision at all sample sizes, meaning that it never predicts that a causal effect is identifiable when it is not. Together with the high recall of Step 2, they demonstrate that Alg. 2 can robustly determine identifiability across different sample sizes. Furthermore, the precision and recall of Step 4 is also high, indicating that the recovered parents of the outcome from local causal discovery even under an imperfect forbidden projection can successfully recover optimal adjustment sets.

H.4 Alternative local causal discovery algorithms

We implement LOAD in Alg. 3 with the MB-by-MB local causal discovery subroutine. In this section, we compare this implementation to an alternative where we replace MB-by-MB by the CMB local causal discovery algorithm (Gao and Ji, 2015), using the implementation of Yu et al. (2020). Our comparison between the two implementations is shown in Fig. 9. Our results show that while LOAD with MB-by-MB takes less time with d-separation and Fisher-Z CI tests and performs less CI tests in all data settings. Similarly, CMB performs comparably on binary data, but achieves much worse F1 scores and intervention distances with Fisher-Z CI tests and even d-separation CI tests. As the latter should not be possible, since CMB should be also sound and complete similarly to MB-by-MB, we assume there might be a bug in its implementation.



(a) CI test errors. (b) Neighborhood errors. (c) F1 Score.

Figure 7: Errors in LOAD for Fisher-Z tests on linear Gaussian data and for G^2 tests on binary data. Fig. 7a: Percentage of erroneous CI tests results over various number of nodes, with $n_{\mathbf{D}} = 10000$. Fig. 7b and Fig. 7c: Number of erroneous neighbors estimated by local causal discovery and F1 score of estimating the optimal adjustment set over various numbers of samples and $n_{\mathbf{V}} = 200$. For all figures $\overline{d} = 2$, $d_{\max} = 10$ and target pairs such that one is an explicit ancestor of the other. The shadow area denotes the range of the standard deviation.

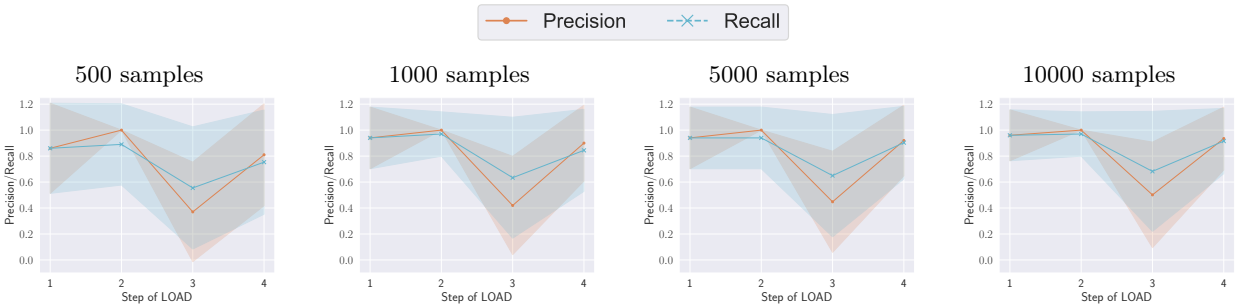


Figure 8: Precision and Recall of the different steps of LOAD in Alg. 3 over different sample sizes, with Fisher-Z tests on linear Gaussian data, $n_{\mathbf{V}} = 200$, $\overline{d} = 2$ and $d_{\max} = 10$ and target pairs such that one is an explicit ancestor of the other. The shadow area denotes the range of the standard deviation.

H.5 Alternative Markov blanket discovery algorithms

For the main results in Fig. 1, we implement LOAD using the MB-by-MB local causal discovery algorithm as a sub-routine, which employs the Grow-Shrink algorithm to find the Markov blankets of variables. In this section, we compare utilizing Grow-Shrink to alternative Markov blanket discovery algorithm employed by MB-by-MB when running LOAD.

In Fig. 10, we show results for replacing Grow-Shrink with Total Conditioning (Pellet and Elisseeff, 2008b). While Total-Conditioning performs fewer CI tests and achieves comparable intervention distances, it requires much more computation time in finite data settings. Furthermore, it completely fails to identify any nodes in the optimal adjustment sets correctly on binary data.

In Fig. 11, we compare to the S^2TMB score based Markov blanket discovery algorithm (Gao and Ji, 2017). We use the implementation of Yu et al. (2020), which can only handle discrete data. Thus, we use CI tests on binary data with 10, 15 and 20 nodes under the same other settings as Fig. 1. Our results show that while S^2TMB performs fewer CI tests than when using Grow-Shrink, it takes much more time. Furthermore, it achieves a worse F1 scores and intervention distances across all graph sizes.

H.6 Providing Treatment-Outcome Relation as Background Knowledge

In this section we evaluate the scenario where the treatment-outcome relationship is known by background knowledge. This scenario fits the requirements of most local causal discovery algorithms in literature, such as

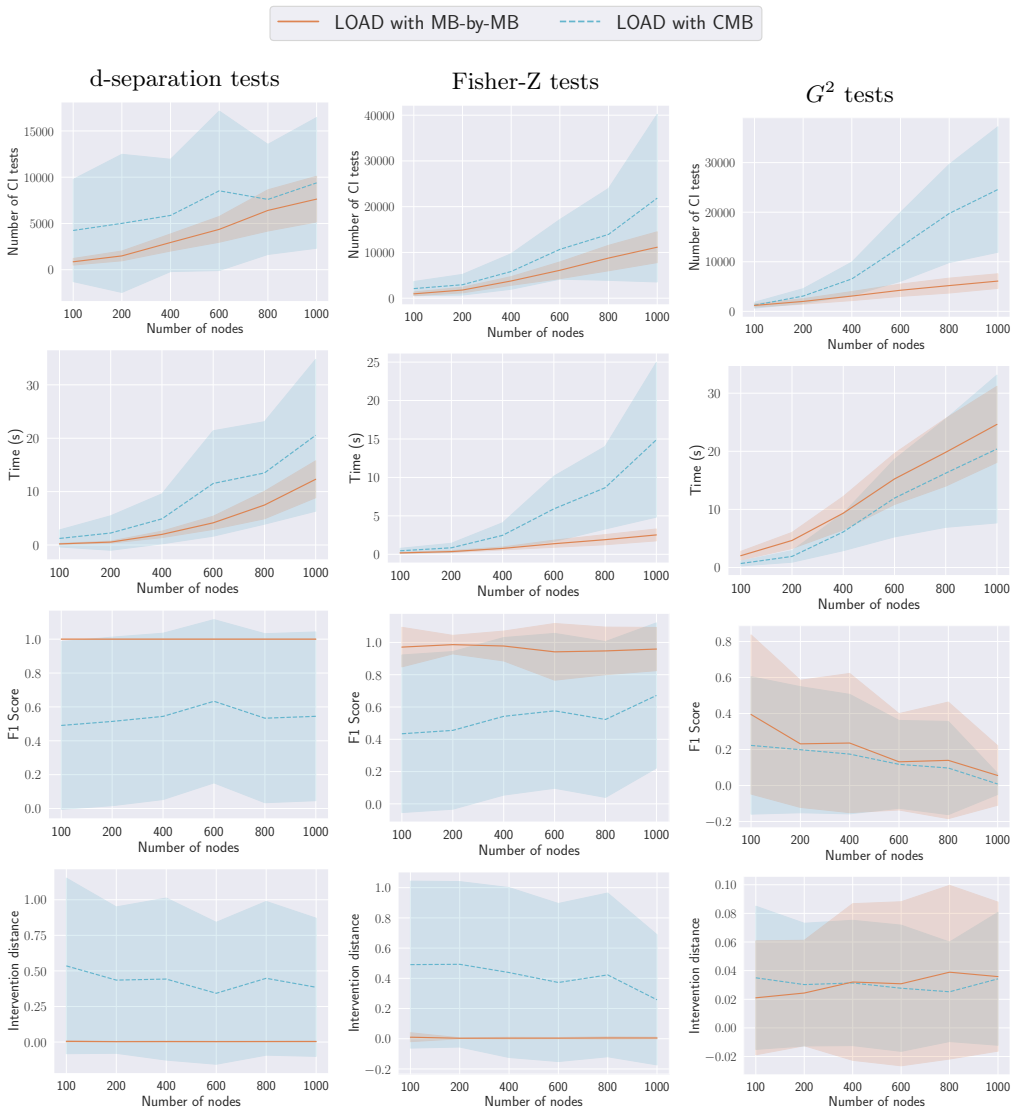


Figure 9: Results comparing LOAD using the MB-by-MB and the CMB algorithms for local causal discovery over various number of nodes, $n_{\mathbf{D}} = 10000$, $\overline{d} = 2$ and $d_{\max} = 10$ and target pairs such that one is an explicit ancestor of the other. The shadow area denotes the range of the standard deviation.

MB-by-MB (Wang et al., 2014), LDECC (Gupta et al., 2023) and LDP (Maasch et al., 2024). Thus, for the experiments in this section, we run the original versions of these algorithms with the background knowledge provided. Furthermore, we introduce a version of LOAD that can also utilize this background knowledge, denoted as LOAD*, which simply skips the first step of LOAD for determining ancestral relationships. Global algorithms, such as PC, MARVEL and SNAP remain unchanged.

Otherwise, we use the same data setup as in the main paper, i.e., $n_{\mathbf{D}} = 10000$ data samples, expected degree of $\overline{d} = 2$ and maximum degree of $d_{\max} = 10$, and use a significance level of $\alpha = 0.01$ for all algorithms and CI tests.

Our results are shown in Fig. 12. Since LDECC only has to run once on the provided treatment instead of both targets, it was possible to evaluate it on binary data using G^2 CI tests.

Results for the number of CI tests time and intervention distance are similar to the case where the background knowledge is not provided. The number of CI tests and computation time increases steadily with the number of variables for all methods under all settings, except for LDECC where they decrease on binary data using G^2 tests. Since this is not the case for LDECC with d-separation or Fisher-Z on linear Gaussian data, or for any

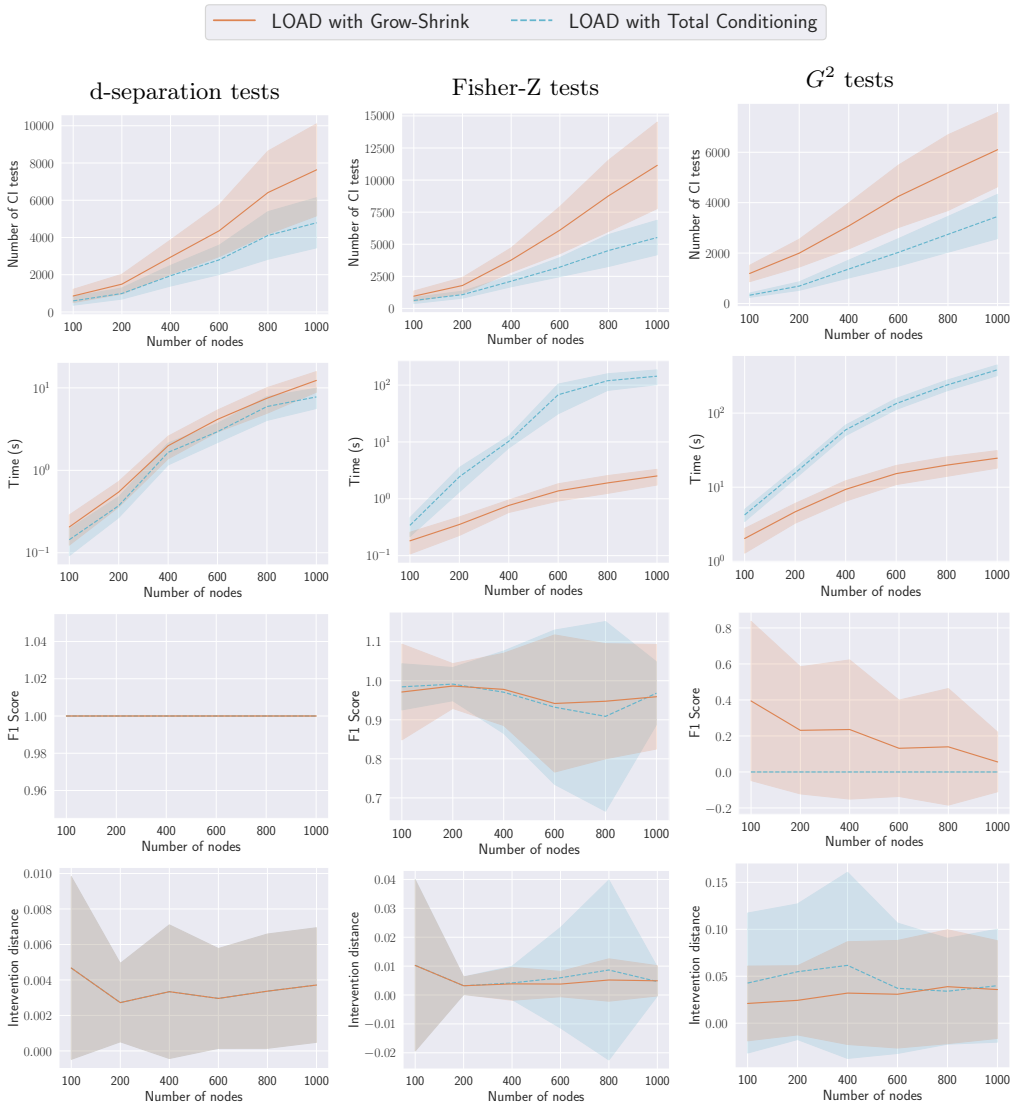


Figure 10: Results comparing LOAD using the Grow-Shrink and the Total-Conditioning algorithms for Markov blanket discovery over various number of nodes, $n_{\mathbf{D}} = 10000$, $\overline{d} = 2$ and $d_{\max} = 10$ and target pairs such that one is an explicit ancestor of the other. The shadow area denotes the range of the standard deviation.

other other method on binary data, we suspect that this arises due to errors in CI testing performed by LDECC. In general, when the treatment-outcome relationship is known, LDP is the cheapest in terms of CI tests and computation time.

On binary data with 100-400 nodes, LOAD* is able to outperform PC in terms of F1 score of the estimated Oset, and remains comparable throughout all number of nodes.

H.7 Identifiable Causal Effects

Sampling target pairs where one is an explicit ancestor of the other, but its causal effect is not necessarily identifiable often leads to unidentifiable causal effects, making LOAD return early after step 2. To evaluate the scenario where LOAD should execute all of its steps, we evaluate all algorithms on target pairs where we ensure that the causal effect is identifiable.

In this setting, we only experiment with node numbers $[100, 200, 300, 400, 500]$ because we encountered time and memory issues when generating the data for larger number of nodes. In particular, without parallelization, each

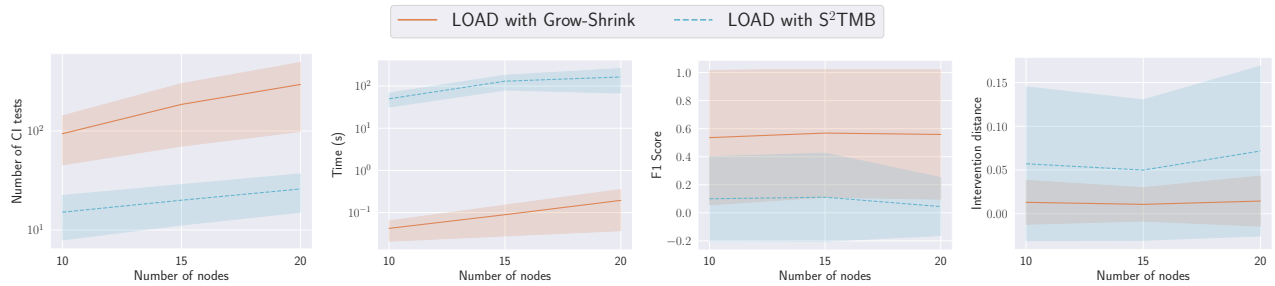


Figure 11: Results comparing LOAD using the Grow-Shrink and the S^{2}TMB algorithms for Markov blanket discovery on binary data sampled from graphs over various number of nodes, $n_{\mathbf{D}} = 10000$, $\overline{d} = 2$ and $d_{\max} = 10$ and target pairs such that one is an explicit ancestor of the other. The shadow area denotes the range of the standard deviation.

seed for 500 nodes take around one hour and 15 minutes, and we average results over a 100 seeds. With larger nodes, parallelization becomes difficult due to the size of the datasets. Otherwise, we use the same data setup as in the main paper, i.e., $n_{\mathbf{D}} = 10000$ data samples, expected degree of $\overline{d} = 2$ and maximum degree of $d_{\max} = 10$, and use a significance level of $\alpha = 0.01$ for all algorithms and CI tests.

Results for identifiable target pairs are shown in Fig. 13. Similarly to the main results report in Fig. 1, the number of CI tests performed by LOAD is similar to local methods, and its computation time is even better than LDP. The F1 score for optimal adjustment set recovery and the intervention distance of LOAD is one of the best across all settings, while all other baselines fall behind in some setting for one of these metrics.

H.8 Various Number of Samples

In this section with experiment with a fixed number of nodes at $n_{\mathbf{V}} = 200$ and vary the number of synthetic data samples among $n_{\mathbf{D}} \in [500, 1000, 5000, 10000]$. We maintain every other setting at expected degree of $\overline{d} = 2$ and maximum degree of $d_{\max} = 10$, and use a significance level of $\alpha = 0.01$ for all algorithms and CI tests. We only evaluate the finite data cases of Fisher-Z CI tests on linear Gaussian data and G^2 CI tests on binary data, as oracle d-separation CI tests are not affected by the sample size.

We report our results in Fig. 14. The number of samples has no impact on the number of CI tests and computation time for Fisher-Z tests. In the case of G^2 tests it slightly increases the number of CI tests and computation time of LOAD and other local methods, and drastically increases the computation time of MARVEL.

More samples help in recovering higher quality adjustment sets in both data domains, as LOAD achieves one of the best F1 scores and intervention distance over all samples.

H.9 Various Expected Degrees

In Fig. 15 we show results for various number of expected degrees $\overline{d} \in [2.0, 2.5, 3.0]$, and a fixed number of nodes at $n_{\mathbf{V}} = 200$, maximum degree of $d_{\max} = 10$ and $n_{\mathbf{D}} = 10000$. We use a significance level of $\alpha = 0.01$ for all algorithms and CI tests.

Fig. 15 shows that increasing the expected degree makes the problem setting much more difficult for all algorithms, as computational requirements generally increase, while the quality of the recovered adjustment sets become worse. Nonetheless, the computational requirements of LOAD remain close to local methods, while still achieving one of the best F1 scores and intervention distances across data domains.

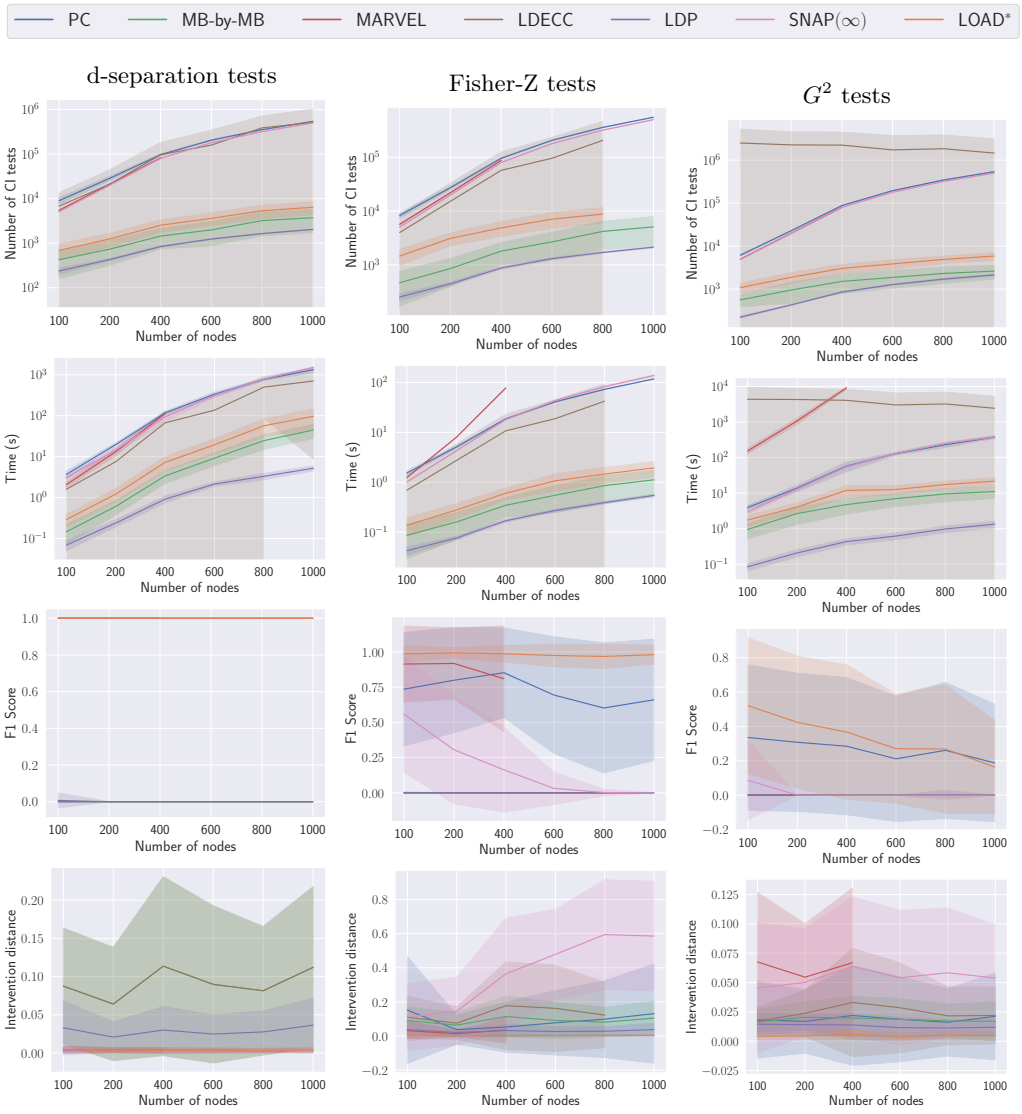


Figure 12: Results over number of nodes with $n_{\mathbf{D}} = 10000$, $\overline{d} = 2$ and $d_{\max} = 10$ for the scenario where the treatment-outcome relation is provided as background knowledge for MB-by-MB, LDECC, LDP and LOAD*. The shadow area denotes the range of the standard deviation.

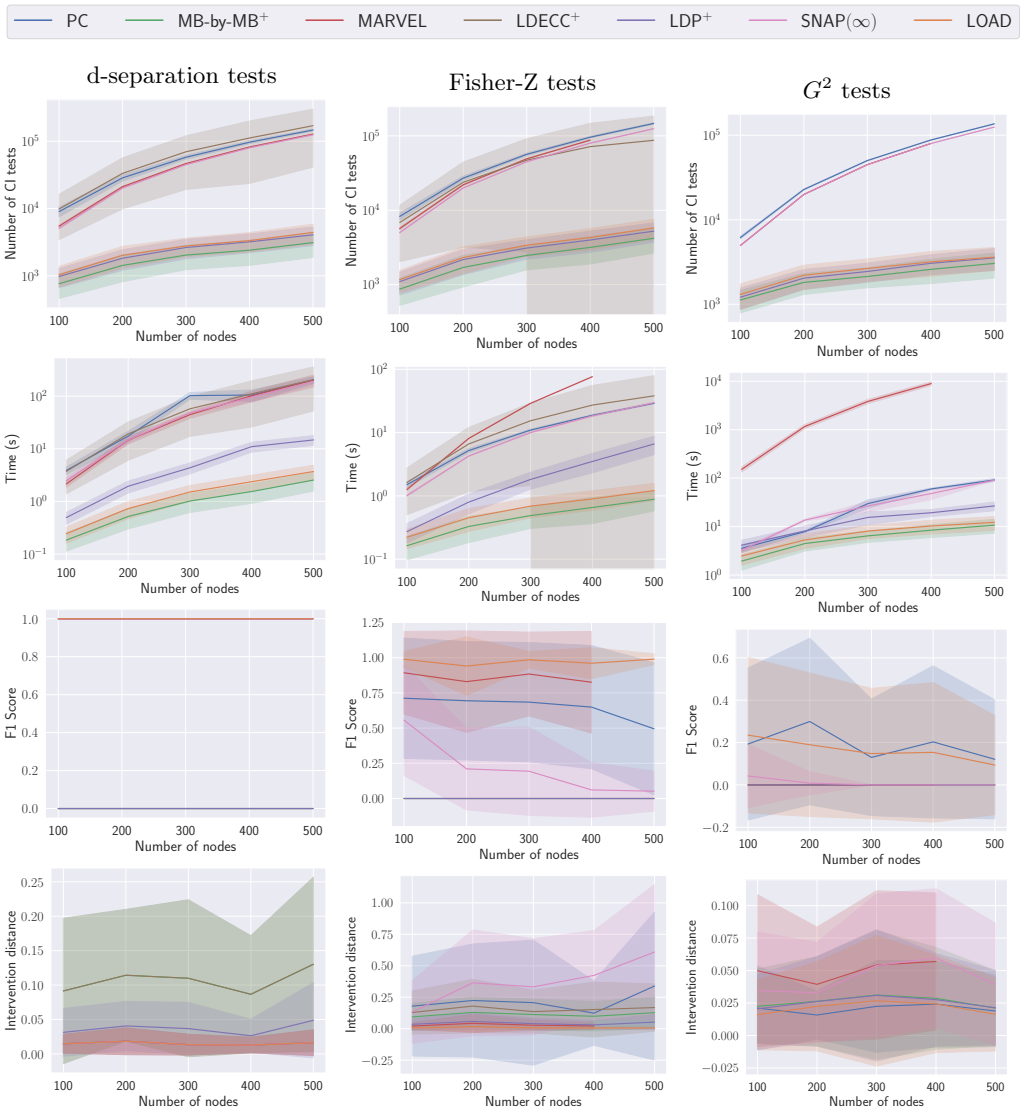


Figure 13: Results over number of nodes with $n_{\mathbf{D}} = 10000$, $\overline{d} = 2$ and $d_{\max} = 10$ with identifiable target pairs. The shadow area denotes the range of the standard deviation.

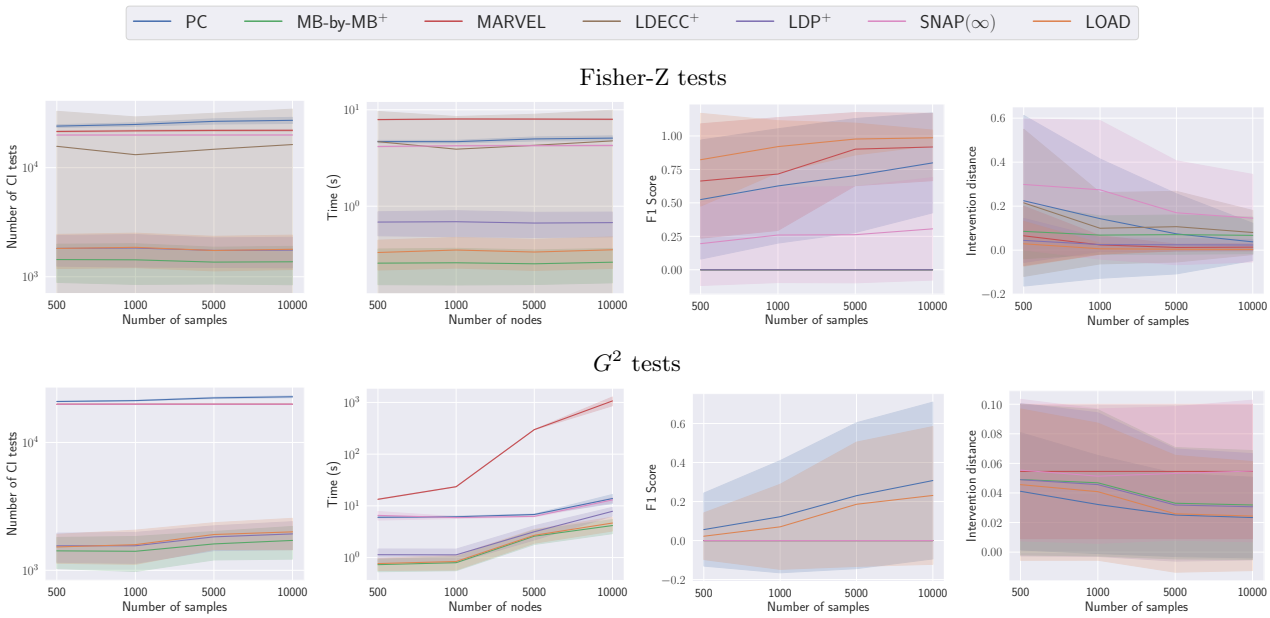


Figure 14: Results over number of samples with $n_{\mathbf{V}} = 200$, $\overline{d} = 2$ and $d_{\max} = 10$ and target pairs such that one is an explicit ancestor of the other. The shadow area denotes the range of the standard deviation.

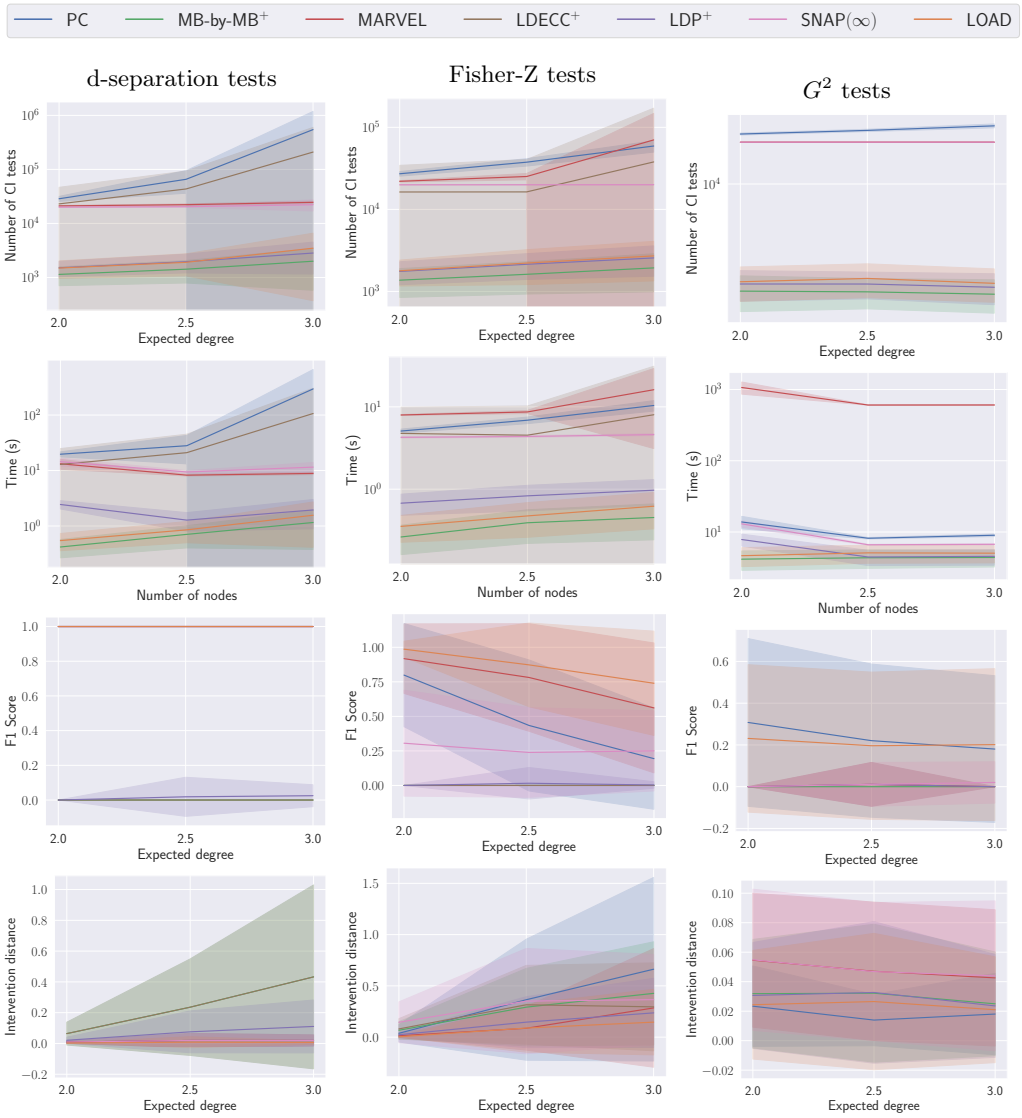


Figure 15: Results over number of expected degrees, with $n_{\mathbf{V}} = 200$, $n_{\mathbf{D}} = 10000$, and $d_{\max} = 10$ and target pairs such that one is an explicit ancestor of the other, for different settings in each column. The shadow area denotes the range of the standard deviation.