Local Causal Discovery for Estimating Causal Effects

Author names withheld

Editor: Under Review for CLeaR 2023

Abstract

Even when the causal graph underlying our data is unknown, we can use observational data to narrow down the possible values that an average treatment effect (ATE) can take by (1) identifying the graph up to a Markov equivalence class; and (2) estimating that ATE for each graph in the class. While the PC algorithm can identify this class under strong faithfulness assumptions, it can be computationally prohibitive. Fortunately, only the local graph structure around the treatment is required to identify the set of possible ATE values, a fact exploited by local discovery algorithms to improve computational efficiency. In this paper, we introduce Local Discovery using Eager Collider Checks (LDECC), a new local discovery algorithm that leverages unshielded colliders to orient the treatment's parents differently from existing methods. We show that there exist graphs where LDECC exponentially outperforms existing local discovery algorithms and vice versa. Moreover, we show that LDECC and existing algorithms rely on different faithfulness assumptions, leveraging this insight to weaken the assumptions for identifying the possible ATE values.

Keywords: Local causal discovery, local structure learning, causal inference, causal discovery

1. Introduction

Estimating an average treatment effect (ATE) from observational data typically requires structural knowledge, which can be represented in the form of a causal graph. While a rich literature offers methods for identifying and estimating causal effects given a *known* causal graph (Tian and Pearl, 2002; Pearl, 2009; Jaber et al., 2019), many applications require that we investigate the values that an ATE could possibly take when the causal graph is unknown. In such cases, we can (i) perform *causal discovery*, using observational data to identify the graph up to a Markov equivalence class (MEC); and (ii) estimate the desired ATE for every graph in the MEC, thus identifying the set of possible ATE values. We denote this (unknown) set of identified ATE values by Θ^*.

Causal discovery has been investigated under a variety of assumptions (Glymour et al., 2019; Squires and Uhler, 2022). Under causal sufficiency (i.e., no unobserved variables) and faithfulness, the PC algorithm (Spirtes et al., 2000, Sec. 5.4.2) can identify the MEC of the true graph from observational data (and thus Θ^*). However, fully characterizing the MEC can be computationally expensive. Addressing this concern, Maathuis et al. (2009) proved that the local structure around the treatment node is sufficient for identifying Θ^*. Leveraging this insight, existing local causal discovery algorithms (e.g., PCD-by-PCD (Yin et al., 2008), MB-by-MB (Wang et al., 2014)) discover just enough of the graph to identify any parents and children of the treatment that PC would have discovered. These methods sequentially discover the local structure around the treatment, its neighbors, and so on, terminating whenever all neighbors of the treatment are oriented (or no remaining neighbors can be oriented).

In this work, we introduce Local Discovery with Eager Collider Checks (LDECC), a new local discovery algorithm that provides an alternative method to orient the parents of a treatment X (Sec. 4). Initially, LDECC performs local discovery around the treatment X to discover its neighbors.

© 2023 .

Subsequently, LDECC chooses the same Conditional Independence (CI) tests as PC would choose given the state of the graph, with one crucial exception: whenever we find two nodes A and B such that $A \perp\!\!\!\perp B|\mathbf{S}$ for some set $\mathbf{S}$ with $X \notin \mathbf{S}$, LDECC immediately checks whether they become dependent when X is added to the conditioning set. If the test reveals dependence $A \not\perp\!\!\!\perp B|\mathbf{S} \cup \{X\}$, then LDECC orients the smallest subset of X's neighbors that d-separate it from $\{A, B\}$ as parents. We prove that, under faithfulness, the identified ATE set is equal to Θ^*.

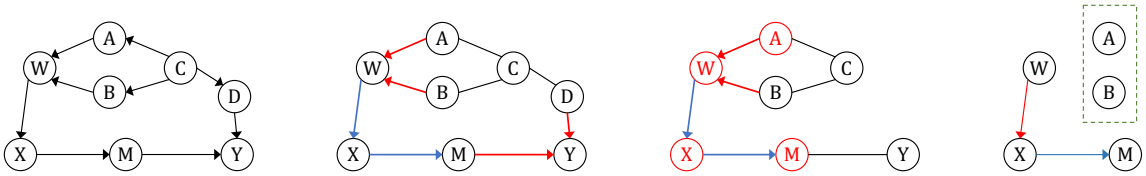
We represent the ideas underlying existing local causal discovery algorithms using a simple algorithm that we call Sequential Discovery (SD) that sequentially runs the PC algorithm locally for local structure learning. While existing algorithms differ from SD in subtle ways, they share key steps with SD allowing us to compare LDECC to this existing class of algorithms. We highlight LDECC's complementary strengths to SD in terms of computational requirements. We present classes of causal graphs where LDECC performs exponentially fewer CI tests than SD, and vice versa (Sec. 4.1). Thus, the methods can be combined profitably (by running LDECC and SD in parallel and terminating when either algorithm terminates), avoiding exponential runtimes if either algorithm's runtime is subexponential, thereby expanding the class of graphs where efficient local discovery is possible.

We also compare SD and LDECC based on their faithfulness (FF) requirements (Sec. 4.2). We show that both SD and PC require weaker assumptions to identify Θ^* than to identify the entire MEC. We also find that LDECC and SD rely on different sets of FF assumptions. There are classes of FF violations where one algorithm will correctly identify Θ^* while the other will not. Under the assumption that one of the algorithms' faithfulness assumptions is correct, we propose a procedure that recovers a conservative bound on the ATE set. Aiming to make this bound sharp, we prove that LDECC and SD can be combined to construct a procedure that can identify Θ^* under strictly weaker FF assumptions, again highlighting LDECC's complementary nature relative to existing methods. Finally, we empirically test LDECC on synthetic as well as semi-synthetic graphs (Sec. 5) and show that it performs comparably to SD (and PC) and typically runs fewer CI tests than SD.

2. Related Work

Several works develop procedures for identifying the set of possible ATE values Θ^* using local information. Maathuis et al. (2009) propose IDA, an algorithm that outputs a set of ATE values by only using the local structure around the treatment X. For any CPDAG, Θ^* can be identified by adjusting for parent sets constructed using the undirected edges from X such that no new unshielded colliders get created at X. IDA has been extended to account for multiple interventions (Nandy et al., 2017), background information (Perković et al., 2017; Fang and He, 2020), and hidden variables (Malinsky and Spirtes, 2016). However, these works do not propose local discovery methods.

A complementary line of work focuses on only discovering the local structure around a given node. Many of these works attempt to find the Markov blanket (MB) or the parent-child set of a target node (Aliferis et al., 2003; Tsamardinos et al., 2006; Yu et al., 2020; Ling et al., 2021). Unlike our work, they do not distinguish between the parent and child identities of nodes inside the MB (see Aliferis et al. (2010) and references within for a review of these methods). For determining the causal identities of the neighbors of a target node, existing works use sequential approaches by finding local structures like a MB or a parent-child set starting from the target node and using the local structure in the discovered subgraph to propagate edge orientations (Yin et al., 2008; Zhou et al., 2010; Wang et al., 2014; Gao and Ji, 2015; Ling et al., 2020). These algorithms typically differ in how they discover the local structure around each node.



(*a*) True DAG $\mathcal{G}^*$. (*b*) CPDAG from PC. (*c*) SD's output (*d*) LDECC

Figure 1: Demonstration of PC, SD, and LDECC for the graph in (a).

```
def LocalPC(Skeleton U, Target V):
    s ← 0;
    while |Ne_U(V)| > s do
        for B ∈ Ne_U(V), S ⊆ Ne_U(V) \ {B}
         s.t. |S| = s do
            if V ⊥⊥ B|S then
                U.removeEdge(V—B);
                DSep(V, B) = S;
                break;
        end
        s ← s + 1;
    end
    return U, Ne_U(V), DSep;
def isLocallyValid(X, S):
    for every A, B ∈ S do
        if isNonCollider(A—X—B) then
            return False;
    end
    return True;
```

Figure 2: Additional subroutines.

```
Input: Treatment X.
Fully connected undirected graph U;
queue ← [X], done ← ∅;
while queue is not empty do
    V ← queue.removeFirstItem();
    U, Ne_U(V), DSep ← LocalPC(U, V);
    done.add(V);
    queue.append(Ne_U(V) \ (done ∪ queue));
    G ← GetCPDAG(U[done], DSep);
    parents, children, unoriented ← Nbrs(G, X);
    if unoriented = ∅ then break ;
end
Θ_SD ← ∅;
for S ⊆ unoriented do
    if isLocallyValid(X, S) then
        R ← S ∪ parents;
        Θ_SD.add(θ_{X→Y|R});
end
return Θ_SD;
```

Figure 3: The SD algorithm.

3. Preliminaries

In this work, we assume that the causal structure of the observational data can be encoded using a DAG $\mathcal{G}^*(\mathbf{V}, \mathbf{E})$, where $\mathbf{V}$ and $\mathbf{E}$ are the set of nodes and edges. Each edge $A \to B \in \mathbf{E}$ indicates that A is a direct cause of B. Throughout this paper, we denote the treatment node by X and the outcome node by Y. For a node V, we denote its Markov blanket, neighbors, parents, children, and descendants by $\text{MB}(X)$, $\text{Ne}(V)$, $\text{Pa}(V)$, $\text{Ch}(V)$, and $\text{Desc}(V)$. Let $\text{Ne}^+(V) = \text{Ne}(V) \cup \{V\}$ and $\text{MB}^+(V) = \text{MB}(V) \cup \{V\}$. An unshielded collider (UC) is a triple $P \to R \leftarrow Q$ such that P—$Q \notin \mathbf{E}$. For a UC $\alpha = (P \to R \leftarrow Q)$, let $\text{sep}(\alpha) = \min\{|\mathbf{S}| : \mathbf{S} \subseteq \mathbf{V} \setminus \{P, Q\}$ and $P \perp\!\!\!\perp Q|\mathbf{S}\}$.

A DAG entails a set of CIs via *d-separation* (Pearl, 2009, Sec. 1.2.3). DAGs that entail the same set of CIs form an MEC, which can be characterized by a *completed partially directed acyclic graph* (CPDAG). For the true DAG $\mathcal{G}^*$, we denote by Θ^* the set of ATE values (of X on Y) in each DAG in the MEC corresponding to $\mathcal{G}^*$. The causal faithfulness assumption (CFA) holds iff all CIs satisfied

by the observational joint distribution $\mathbb{P}(\mathbf{V})$ are entailed by $\mathcal{G}^*$. Throughout this work, we focus on causally sufficient graphs. Under the CFA, the PC algorithm recovers this MEC roughly as follows (demonstrated for the DAG in Fig. 1(*a*)): (i) estimate the skeleton by running CI tests; (ii) find UCs in this skeleton (Fig. 1(*b*) red edges); and (iii) orient additional edges using Meek's rules (Meek, 2013) (Fig. 1(*b*) blue edges) to get a CPDAG (full details on the PC algorithm are in Appendix A).

We instantiate the key ideas of existing local discovery algorithms using SD (Fig. 3). SD sequentially finds the local structure around nodes (*LocalPC* in Fig. 2) starting from X, then its neighbors, and so on (Lines 3–7). After each such local discovery step, SD orient nodes in the subgraph discovered until that point using UCs and Meek's rules like PC (Line 8; see Fig. 12 for *GetCPDAG* and Fig. 15 for *Nbrs*). If at any point all neighbors of X get oriented, SD terminates. The ATE set is estimated by applying the backdoor adjustment (Pearl, 2009, Thm. 3.3.2) with every *locally valid* parent set of X, i.e., one that does not create a new UC at X (see Lines 12–17 and *isLocallyValid* in Fig. 2). The ATE estimated using each such parent set $\mathbf{R}$ is denoted as $\theta_{X \to Y|\mathbf{R}}$ (Line 16). Consider the DAG in Fig. 1(*a*). Here, SD will discover the neighbors of X, W, M, and A. Then the UC $A \to W \leftarrow B$ will be detected and, after propagating orientations, both W and M will be oriented and SD will terminate (Fig. 1(*c*)) and output $\Theta_{\text{SD}} = \{\theta_{X \to Y|\{W\}}\} = \Theta^*$.

4. Local Discovery using Eager Collider Checks (LDECC)

In this section, we propose LDECC, a local causal discovery algorithm that orients the parents of X by leveraging UCs differently from existing methods (Prop. 4). We prove its correctness under the CFA (Thm. 5) and show its complementary nature relative to SD in terms of computational (Sec. 4.1) and faithfulness (Sec. 4.2) requirements. We defer the proofs to Appendix B.1. We first define a *Minimal Neighbor Separator* (MNS) which plays a key role in LDECC.

Definition 1 (Minimal Neighbor Separator (MNS)) *For a DAG $\mathcal{G}(\mathbf{V}, \mathbf{E})$ and nodes X and $A \notin Ne^+(X)$; $mns_X(A) \subseteq Ne(X)$ is the unique set (see Prop. 3) of nodes such that (i) (d-separation) $A \perp\!\!\!\perp X | mns_X(A)$, and (ii) (minimality) for any $\mathbf{S} \subset mns_X(A)$, $A \not\perp\!\!\!\perp X|\mathbf{S}$.*

For the graph in Fig. 1(*a*), we have $\text{mns}_X(A) = \text{mns}_X(B) = \text{mns}_X(C) = \{W\}$ and $\text{mns}_X(Y) = \{W, M\}$. While an MNS need not exist for every node (see Example 6 in Appendix B), $\text{mns}_X(V)$ always exists $\forall V \notin \text{Desc}(X)$ (which is sufficient for correctness of LDECC):

Proposition 2 *For any node $V \notin (Desc(X) \cup Ne^+(X))$, $mns_X(V)$ exists and $mns_X(V) \subseteq Pa(X)$.*

Proposition 3 (Uniqueness of MNS) *For every node V such that $mns_X(V)$ exists, it is unique.*

Proposition 4 (Eager Collider Check) *For nodes $A, B \in \mathbf{V} \setminus Ne^+(X)$ and $\mathbf{S} \subseteq \mathbf{V} \setminus \{A, B, X\}$, if (i) $A \perp\!\!\!\perp B|\mathbf{S}$; and (ii) $A \not\perp\!\!\!\perp B|\mathbf{S} \cup \{X\}$; then $A, B \notin Desc(X)$ and $mns_X(A), mns_X(B) \subseteq Pa(X)$.*

Prop. 4 suggests a different strategy for orienting parents of X: if two nodes that are d-separated by $\mathbf{S}$ become d-connected by $\mathbf{S} \cup \{X\}$, the MNS of such nodes contains the parents of X. Thus, unlike PC and SD, the skeleton is not needed to propagate orientations via Meek's rules.

The LDECC algorithm (Fig. 5) first finds the Markov blanket $\text{MB}(X)$ (Line 1): we use the IAMB algorithm (Tsamardinos et al., 2003, Fig. 2) in our experiments (see Fig. 14 in Appendix B.1). Next, we run *LocalPC* within the subgraph containing $\text{MB}(X)$ to prune this set to obtain $\text{Ne}(X)$

```
def UCChildren(MB(X), Ne(X), DSep):
    spousesX ← MB(X) \ Ne(X);
    children ← ∅;
    for D ∈ spousesX do
        for C ∈ Ne(X) \ DSep(D, X) do
            if C ⫫̸ D|DSep(D, X) then
                children.add(C) ;
        end
    end
    return children;
def GetMNS(V ∉ Ne⁺(X)):
    s ← 0;
    while |Ne(X)| > s do
        for S ⊆ Ne(X) s.t. |S| = s do
            if V ⫫ X|S then return S ;
        end
        s ← s + 1;
    end
    return MNS not found;
def ECCParents(A, B, check=False):
    if check and {A, B} ∩ Ne(X) = ∅ then
        m_A ← GetMNS(A) ;
        m_B ← GetMNS(B);
        if m_A = m_B then return m_A;
        else return ∅;
    parents ← ∅;
    for V ∈ {A, B} do
        if V ∈ Ne(X) then parents.add(V);
        else parents.add(GetMNS(V));
    end
    return parents;
```

Figure 4: Subroutines used by LDECC.

```
Input: Treatment X.
1  MB(X) ← FindMarkovBlanket(X);
2  _, Ne(X), DSep ← LocalPC(MB⁺(X), X);
3  children ← UCChildren(MB(X), Ne(X), DSep);
4  parents ← ∅, unoriented ← Ne(X) \ children;
5  for (A ⫫ B|S) ∈ PC test (A, B ≠ X) do
6      if A, B ∈ Ne(X) and X ∉ S then
7          parents.add({A, B});
8          parents.add(S ∩ Ne(X));
9      else if A, B ∈ Ne(X) and X ∈ S then
10         MarkNonCollider(A—X—B);
11         for V ∈ Ne(X) \ (S ∪ {A, B}) do
12             if A ⫫̸ B|{S, V} then
                   children.add(V) ;
13         end
14     else if X ∉ S and A ⫫̸ B|{S, X} then
15         parents.add(ECCParents(A, B, S));
16     end
17     for P ∈ parents, C ∈ unoriented do
18         if isNonCollider(P—X—C) then
               children.add(C) ;
19     end
20     unoriented.remove(parents ∪ children);
21     if unoriented = ∅ then break ;
22 end
23 Θ_LDECC ← ∅;
24 for S ⊆ unoriented do
25     if isLocallyValid(S) then
26         R ← S∪ parents;
27         Θ_LDECC.add(θ_{X→Y|R});
28 end
Output: Θ_LDECC
```

Figure 5: The LDECC algorithm.

(Line 2). We then use the nodes in $\text{MB}(X) \setminus \text{Ne}(X)$ to detect children C of X that are part of UCs of the form $X \to C \leftarrow D$ (Line 3 and *GetUCChildren* in Fig. 4). LDECC then starts running CI tests in the same way as PC would (excluding tests for X since we already know $\text{Ne}(X)$) (the for-loop in Line 5). Every time we detect a CI $A \perp\!\!\!\perp B|\mathbf{S}$, we check the following cases: (i) if $A, B \in \text{Ne}(X)$ and $X \notin \mathbf{S}$, then there must be a UC $A \to X \leftarrow B$ and so we mark A and B as parents (Lines 6, 7); (ii) if $A, B \in \text{Ne}(X)$ and $X \in \mathbf{S}$, then we mark A—X—B as a non-collider (Lines 9, 10); and (iii) **Eager Collider Check (ECC):** if $X \notin \mathbf{S}$ and $A \not\perp\!\!\!\perp B|\mathbf{S} \cup \{X\}$, then, by leveraging Prop. 4, we mark $\text{GetMNS}(A)$ and $\text{GetMNS}(B)$ as parents (Lines 14, 15 and *ECCParents* in Fig. 4). Next, for each oriented parent, we use the non-colliders detected in Case (ii) to mark children (Lines 17, 18).

LDECC terminates if there are no unoriented neighbors. The ATE set is computed similarly to SD by applying the backdoor adjustment with every *locally valid* parent set. Lines 11–13 and Line 8 are needed to account for Meek's rules 3 and 4. The key departure from PC is Case (iii) where an ECC (and not the skeleton) is used to orient parents. The *check=True* argument in *ECCParents* aims to make it more robust to faithfulness violations (see Remark 21 in Sec. 4.2).

Consider the running example of the DAG in Fig. 1(*a*). We first find the local structure around X. Then we start running CI tests in the same way as PC. After we run $A \perp\!\!\!\perp B|C$, we do an ECC and find $A \not\perp\!\!\!\perp B|\{C, X\}$. Since $\text{mns}_X(A) = \text{mns}_X(B) = \{W\}$, we mark W as a parent of X. Next, we find that $W \perp\!\!\!\perp M|X$ and mark W—X—M as a non-collider which lets us mark M as a child (because W is a parent). LDECC terminates as all neighbors of X are oriented (Fig. 1(*d*)) and the ATE set is computed by using $\{W\}$ as the backdoor adjustment set: $\Theta_{\text{LDECC}} = \{\theta_{X \to Y|\{W\}}\} = \Theta^*$.

Theorem 5 (Correctness) *Under the CFA and with access to a CI oracle, we have* $\Theta_{LDECC} \overset{set}{=} \Theta^*$.

Remark *A DAG can have multiple valid adjustment sets. We show that local information suffices for identifying the optimal adjustment set, i.e., the set with the lowest asymptotic variance (Henckel et al., 2019, Sec. 3.4) and present a procedure to discover it (see Fig. 19, Prop. 44 in Appendix D.2).*

4.1. Comparison of computation requirements

In this section, we compare PC, SD, and LDECC based on the number of CI tests they perform. We show that there are classes of DAGs where LDECC is subexponential while SD is not, and vice versa (Props. 7,8). We use this insight to construct a hybrid algorithm that runs in polynomial time on a larger class of DAGs (Remark 13). We defer some proofs to Appendix B.2. We make the following assumption throughout this section.

Assumption 1 *The CFA holds and we have access to a CI oracle.*

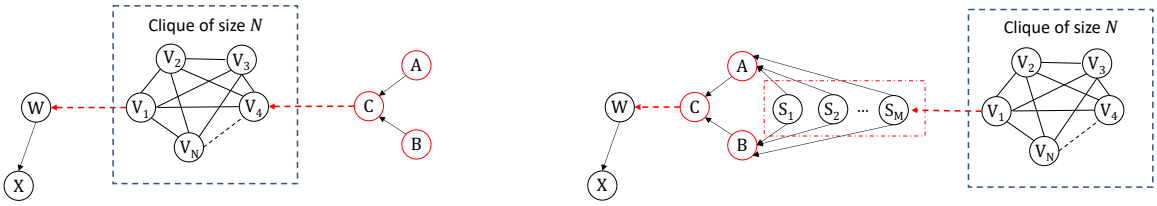
Let the number of CI tests performed by PC, SD, and LDECC be T_{PC}, T_{SD}, and T_{LDECC}. We first show that, in the worst case, LDECC only runs a polynomial (in $|\mathbf{V}|$) number of extra CI tests compared to PC when $|\text{Ne}(X)|$ is bounded. In practice, we observe that the upper bound is quite loose: Even when $|\text{Ne}(X)|$ is large, PC also has to run a large number of tests for X.

Proposition 6 (PC vs LDECC) *We have* $T_{LDECC} \leq T_{PC} + \mathcal{O}(|\mathbf{V}|^2) + \mathcal{O}\left(|\mathbf{V}| \cdot 2^{|Ne(X)|}\right)$.

Proposition 7 (LDECC exponentially better) *There exist classes of graphs such that* $T_{LDECC} + \Theta(2^{|\mathbf{V}|}) \leq T_{PC}$; *and* $T_{LDECC} + \Theta(2^{|\mathbf{V}|}) \leq T_{SD}$.

Proof Consider the class of graphs shown in Fig. 6(*a*) where there is a clique of size N on the path from the UC $A \to C \leftarrow B$ to W. Both PC and SD will perform $\Theta(2^N)$ CI tests for nodes inside this clique. Thus, when $N \asymp |\mathbf{V}|$, SD will have exponential complexity. By contrast, LDECC unshields the UC and orients W as a parent via an ECC in $\mathcal{O}(|\mathbf{V}|^2)$ CI tests. ■

Qualitatively, the result shows that if there is a dense region between a UC and the parent it orients, SD might perform poorly because it has to wade through this dense region to get to the UC. By contrast, LDECC can avoid this problem as it does not rely on the skeleton to orient the parents. However, LDECC does *not* uniformly dominate SD in terms of the number of tests. There are classes of causal graphs where LDECC performs an exponentially greater number of CI tests than SD:



(*a*) LDECC outperforms SD. (*b*) SD outperforms LDECC.

Figure 6: Classes of graphs where LDECC and SD have different runtimes.

Proposition 8 (SD exponentially better) *Let* $\mathbf{U}$ *be the set of UCs in* $\mathcal{G}^*$ *and* $M = \max_{U \in \mathbf{U}} sep(U)$. *There exist classes of graphs such that* $T_{SD} + \Theta(2^{M-1}) \leq T_{PC}$; *and* $T_{SD} + \Theta(2^{M-1}) \leq T_{LDECC}$.

Proof Consider the class of graphs shown in Fig. 6(*b*) with a UC with a separating set of size M and a clique of size N upstream of that UC. Since LDECC runs CI tests in the same order as PC, if $N \geq M$, LDECC performs $\Theta(2^{M-1})$ CI tests for nodes inside the clique (all tests of the form $V_i \perp\!\!\!\perp V_j | \mathbf{S}$ s.t. $\mathbf{S} \subseteq \{V_1, \ldots, V_N\} \setminus \{V_i, V_j\}$ and $|\mathbf{S}| < M$ will be run) before the UC is unshielded via the test $A \perp\!\!\!\perp B | \{S_1, \ldots, S_M\}$. By contrast, SD terminates before even getting to the clique, thus avoiding these tests. So if $M \asymp |\mathbf{V}|$, LDECC will perform exponentially more tests than SD. ■

Qualitatively, this shows that if UCs have large separating sets and there are dense regions upstream of such UCs, LDECC might perform poorly whereas SD can avoid tests in these regions since it would not reach these dense regions.

Next, we present upper bounds for the computational requirements of LDECC and SD for a restricted class of graphs that we call *Locally Orientable Graphs*.

Definition 9 (Locally Orientable Graph.) *A DAG* $\mathcal{G}$ *is said to be* locally orientable *with respect to a node* X *if,* $\forall V \in Ne(X)$, *the edge* X—V *is oriented in the CPDAG for the MEC of* $\mathcal{G}$. *Since all nodes in* $Ne(X)$ *are oriented, the ATE is point identified, i.e.,* $|\Theta^*| = 1$.

Definition 10 (Parent Orienting Collider (POC)) *Consider a DAG* $\mathcal{G}^*$. *For each* $P \in Pa(X)$, *let* $POC(P)$ *be the set of UCs that can orient* P, *i.e., either (i)* P *is a part of that UC; or (ii) Applying Meek's rules from that UC in the (undirected) skeleton will orient* $P \to X$.

In Props. 11,12, we provide sufficient graphical conditions for when both the algorithms run in polynomial time. For LDECC, we show that the complexity of orienting the parents depends only on the size of the separating sets for the UCs, i.e., *sep* (Prop. 11). Importantly, the bound is agnostic to how far away the UCs are from the parents.

Proposition 11 (LDECC upper bound) *For a locally orientable DAG* $\mathcal{G}^*$, *let* $D = \max_{V \in MB^+(X)} |Ne(V)|$ *and* $S = \max_{P \in Pa(X)} \min_{\alpha \in POC(P)} sep(\alpha)$. *Then* $T_{LDECC} \leq \mathcal{O}(|\mathbf{V}|^{\max\{S,D\}})$.

For SD, we show that the complexity of orienting a parent is upper bounded by (i) the *sep* of the closest UC that orients that parent and (ii) the complexity of discovering the neighbors of nodes on the path to that collider (Prop. 12). Importantly, this bound is agnostic to the structure of the nodes that are upstream of the closest colliders.

Proposition 12 (SD upper bound) *For a locally orientable DAG $\mathcal{G}^*$, let $\pi : \mathbf{V} \to \mathbb{N}$ be the order in which nodes are processed by SD (Line 4 of SD). For $P \in Pa(X)$, let $CUC(P) = \operatorname{argmin}_{\alpha \in POC(P)} \pi(\alpha)$ denote the closest UC to P. Let $C = \max_{P \in Pa(X)} sep(CUC(P))$, $D = \max_{V \in MB^+(X)} |Ne(V)|$, and $E = \max_{\{V : \pi(V) < \pi(CUC(P))\}} |Ne(V)|$. Then $T_{SD} \leq \mathcal{O}(|\mathbf{V}|^{\max\{C,D,E\}})$.*

Remark 13 (Hybrid algorithm) *Consider the following procedure: Run both SD and LDECC in parallel and terminate when either algorithm terminates. The number of CI tests performed will be $T_{hybrid} \leq 2 \min\{T_{SD}, T_{LDECC}\}$. Thus this procedure broadens the class of graphs where local discovery can be performed efficiently because it will be subexponential if at least one of the algorithms is subexponential (whereas PC might still be exponential).*

4.2. Comparison of faithfulness requirements

In this section, we compare the FF requirements of SD and LDECC and show that they rely on different assumptions. The CFA is controversial (Uhler et al., 2013) and some works attempt to weaken or test the CFA (Zhang and Spirtes, 2008; Ramsey et al., 2012; Zhang and Spirtes, 2016). Sharing a similar motivation, we show that LDECC and SD can be fruitfully combined to strictly weaken the FF assumptions needed to identify Θ^*. We defer the proofs to Appendix B.3.

Definition 14 (Adjacency Faithfulness (AF)) *Given a DAG $\mathcal{G}(\mathbf{V}, \mathbf{E})$, AF holds for node $A \in \mathbf{V}$ iff $\forall A—B \in \mathbf{E}$, we have $\forall \mathbf{S} \subseteq \mathbf{V} \setminus \{A, B\},\ A \not\perp\!\!\!\perp B|\mathbf{S}$.*

Definition 15 (Orientation Faithfulness (OF)) *Given a DAG $\mathcal{G}(\mathbf{V}, \mathbf{E})$, OF holds for an unshielded triple $A—C—B$ (where $A—B \notin E$) iff (i) when it is a UC, $A \not\perp\!\!\!\perp B|\mathbf{S}$ for any $\mathbf{S} \subseteq \mathbf{V} \setminus \{A, B\}$ s.t. $C \in \mathbf{S}$; and (ii) when it is a non-collider, $A \not\perp\!\!\!\perp B|\mathbf{S}$ for any $\mathbf{S} \subseteq V \setminus \{A, B\}$ s.t $C \notin \mathbf{S}$.*

Definition 16 (Local Faithfulness (LF)) *Given a DAG $\mathcal{G}(\mathbf{V}, \mathbf{E})$, LF holds for node X iff (i) AF holds for X; and (ii) OF holds for every unshielded triple $A—X—B \in \mathcal{G}$.*

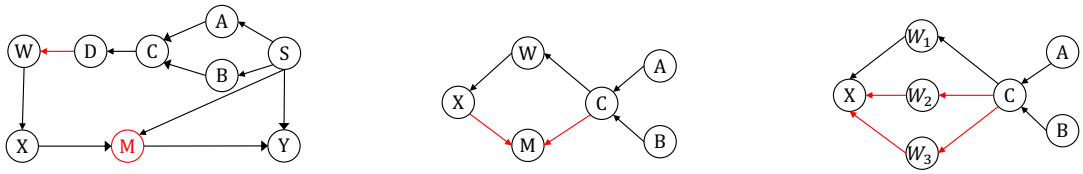
Proposition 17 (Ramsey et al. (2012)) *PC can identify the MEC of $\mathcal{G}^*$ if LF holds for all nodes.*

When the goal is to identify Θ^* (and not the entire MEC), weaker FF conditions than those of Prop. 17 are sufficient for SD and PC (Prop. 18). Let $\mathcal{U}$ be the undirected graph on the nodes $\mathbf{V}$ such that an edge $A—B$ exists iff $\forall \mathbf{S} \subseteq \mathbf{V} \setminus \{A, B\},\ A \not\perp\!\!\!\perp B|\mathbf{S}$. Informally, $\mathcal{U}$ is the skeleton of the observational distribution. Let $J^* = \{(A \to C \leftarrow B) \in \mathcal{G}^*\}$ be the set of UCs in the true DAG $\mathcal{G}^*$.

Proposition 18 (FF for PC and SD) *PC and SD will identify Θ^* if (i) LF holds $\forall V \in MB^+(X)$; (ii) $\forall (A \to C \leftarrow B) \in J^*$, (a) LF holds for A, B, and C, and (b) LF holds for each node on all paths $C \to \ldots \to V \in \mathcal{G}^*$ s.t. $V \in Ne(X)$; (iii) For every edge $A—B \notin \mathcal{G}^*$, $\exists \mathbf{S} \subseteq Ne_{\mathcal{U}}(A) \cup Ne_{\mathcal{U}}(B)$ s.t. $A \perp\!\!\!\perp B|\mathbf{S}$; and (iv) OF holds for all unshielded triples in $\mathcal{G}^*$.*

Conditions (i,ii) ensure that UCs in $\mathcal{G}^*$ are discovered and orientations are propagated correctly to nodes in $\mathrm{Ne}(X)$. Conditions (iii,iv) ensure that spurious UCs are not detected by PC and SD. Prop. 18 is weaker because AF violations are allowed if they are not on paths from UCs to $\mathrm{Ne}(X)$.

We now present sufficient FF conditions for LDECC. Let $\mathbf{L} = \{(A, B, \mathbf{S}) : |\{A, B\} \cap \mathrm{Ne}(X)| = 1,\ \mathbf{S} \subseteq \mathbf{V} \setminus \{A, B, X\}\}$ and $H^* = \{(A, B, \mathbf{S}) \in \mathbf{L} : A \perp\!\!\!\perp_{\mathcal{G}^*} B|\mathbf{S},\ A \not\perp\!\!\!\perp_{\mathcal{G}^*} B|\mathbf{S} \cup \{X\}\}$, where $\perp\!\!\!\perp_{\mathcal{G}^*}$ ($\not\perp\!\!\!\perp_{\mathcal{G}^*}$) denotes d-separation (d-connection). Informally, H^* contains the set of ECCs entailed by the true DAG $\mathcal{G}^*$. Analogously, let $H = \{(A, B, \mathbf{S}) \in \mathbf{L} : A \perp\!\!\!\perp B|\mathbf{S},\ A \not\perp\!\!\!\perp B|\mathbf{S} \cup \{X\}\}$. Informally, H contains the set of ECCs entailed by the observational joint distribution $\mathbb{P}(\mathbf{V})$.



(a) DAG with FF violations (b) Testable FF violation for SD (c) MFF fails for node A, B

Figure 7: Different faithfulness violations.

Definition 19 (MNS Faithfulness (MFF)) *Given a DAG $\mathcal{G}(\mathbf{V}, \mathbf{E})$, MFF holds for node $V \notin Ne^{+}(X)$ iff (a) When $mns_X(V)$ does* not *exist, $\forall \mathbf{S} \subseteq Ne(X)$, we have $V \not\perp\!\!\!\perp X|\mathbf{S}$; and (b) When $mns_X(V)$ exists, $\forall \mathbf{S} \subseteq Ne(X)$ s.t. $\mathbf{S} \neq mns_X(V)$ and $V \perp\!\!\!\perp X|\mathbf{S}$, we have $|\mathbf{S}| > |mns_X(V)|$.*

Proposition 20 (FF for LDECC) *LDECC will identify Θ^* if (i) LF holds $\forall V \in MB^{+}(X)$; (ii) $H \subseteq H^*$; (iii) $\forall (A, B, \mathbf{S}) \in H$, MFF holds for $\{A, B\} \setminus Ne(X)$; and (iv) $\forall (A, B, \mathbf{S}) \in H^*$ s.t. there is a UC $(A \rightarrow C \leftarrow B) \in \mathcal{G}^*$, we have (a) AF holds for A and B; and (b) $(A, B, \mathbf{S}) \in H$.*

Proof Sketch By Condition (i), the structure within $\text{MB}^{+}(X)$ is correctly discovered. By Condition (ii), every ECC run by LDECC will be valid (since it is also part of H^*), and together with Condition (iii), every ECC will mark the correct parents. By Condition (iv), an ECC will be run for every UC that can orient parents of X, ensuring that all parents get oriented: (iv)(a) ensures that the UC gets unshielded, and (iv)(b) ensures that an ECC is run in Line 15 after the UC is unshielded. ■

Remark 21 *In LDECC, if we run* ECCParents *with* check=True *(Fig. 4) where we mark parents with an ECC only if GetMNS(A) = GetMNS(B) (Lines 20–24), then LDECC is correct under weaker FF conditions than Prop. 20 (see Prop. 29 in Appendix B.3) and does better empirically (Sec. 5).*

We demonstrate that SD (and PC) and LDECC rely on different FF conditions. There are FF violations that affect one algorithm but not the other. This motivates a procedure that outputs a (conservative) ATE set containing the values in Θ^* under weaker FF assumptions (Prop. 22).

Example 1 (SD incorrect.) *Consider the FF violation $W \perp\!\!\!\perp D|M$ for the DAG in Fig. 7(a) (conditioning on the collider M cancels out the $W \leftarrow D$ edge). Thus, the edge $W \leftarrow D$ might be incorrectly removed and both SD and PC will not mark W as a parent since the orientations will not be propagated to W. However, LDECC can still correctly mark W as a parent using an ECC.*

Example 2 (LDECC incorrect.) *Consider the FF violations $A \perp\!\!\!\perp X|M$ and $B \perp\!\!\!\perp X|M$ for the DAG in Fig. 7(a). We might incorrectly find $mns_X(A) = mns_X(B) = \{M\}$ causing the ECC to orient M as a parent. However, these violations do not impact SD and PC.*

Proposition 22 (Conservative ATE set) *Consider the following procedure: (i) run both SD and LDECC to get Θ_{SD} and Θ_{LDECC}; (ii) output the set $\Theta_{union} = \Theta_{LDECC} \cup \Theta_{SD}$. If the FF assumptions for either SD or LDECC hold, then $\Theta^* \subseteq \Theta_{union}$.*

Next, we show that some FF violations for SD and LDECC are detectable (however, our procedures cannot detect all FF violations of SD and LDECC). Leveraging this insight, we construct a procedure that identifies Θ^* under strictly weaker FF assumptions than those of Props. 18,20.

Input: Node $A \notin \text{Ne}^+(X), \text{Ne}(X)$

1 $\mathbf{Q} \leftarrow \{\mathbf{S} \subseteq \text{Ne}(X) : A \perp\!\!\!\perp X|\mathbf{S}\}$;

2 **if** $|\mathbf{Q}| = 0$ **then return** noValidMNS ;

3 $\mathbf{Q}_{\min} \leftarrow \{\mathbf{S} \in \mathbf{Q} : \forall \mathbf{S}' \subset \mathbf{S}, \mathbf{S}' \notin \mathbf{Q}\}$;

4 **if** $|\mathbf{Q}_{\min}| > 1$ **then return** Fail ;

5 $\mathbf{S} \leftarrow$ GetElement($\mathbf{Q}_{\min}$) ; // Here $|\mathbf{Q}_{\min}| = 1$.

6 **for** $\mathbf{S}' \in \mathbf{Q}$ *such that* $\mathbf{S} \subset \mathbf{S}'$ **do**

7 | **if** $\exists \mathbf{S}'' \notin \mathbf{Q}$ *s.t.* $\mathbf{S} \subset \mathbf{S}'' \subset \mathbf{S}'$ **then return** Fail ;

8 **end**

9 **return** Unknown;

Figure 8: Testing MFF.

Input: UC $\alpha = (A \rightarrow C \leftarrow B)$

1 $\mathbf{Q}(A) \leftarrow \{\mathbf{S} \subseteq \text{Ne}(X) : A \perp\!\!\!\perp X|\mathbf{S}\}$;

2 $\mathbf{Q}(B) \leftarrow \{\mathbf{S} \subseteq \text{Ne}(X) : B \perp\!\!\!\perp X|\mathbf{S}\}$;

3 $\mathbf{M} \leftarrow \{P \in \text{Pa}(X) : \alpha \in \text{POC}(P)\}$;

4 $\widetilde{M}(A) \leftarrow \{\mathbf{S} \in \mathbf{Q}(A) : \mathbf{M} \subseteq \mathbf{S}\}$;

5 $\widetilde{M}(B) \leftarrow \{\mathbf{S} \in \mathbf{Q}(B) : \mathbf{M} \subseteq \mathbf{S}\}$;

6 **if** $|\widetilde{M}(A)| = 0$ *or* $|\widetilde{M}(B)| = 0$ **then**
return Fail ;

7 **return** Unknown;

Figure 9: Testing FF for SD and PC.

Proposition 23 (Testing FF for LDECC) *Consider running the algorithm in Fig. 8 before invoking* GetMNS *for LDECC. If the algorithm returns* Fail*, MFF is violated for node A. If the algorithm returns* Unknown*, we could not ascertain if MFF holds for node A.*

Example 3 (Testing MFF) *Consider the FF violations $A \perp\!\!\!\perp X|M$ and $B \perp\!\!\!\perp X|M$ for the DAG in Fig. 7(a). Line 4 in Fig. 8 will return* Fail *because for A, B, we have $\mathbf{Q}_{\min} = \{\{W\}, \{M\}\}$.*

Example 4 (Testing MFF) *Consider the DAG in Fig. 7(c). If a FF violation causes the paths $X \leftarrow W_2 \leftarrow C$ and $X \leftarrow W_3 \leftarrow C$ to cancel each other out, we will observe $A \perp\!\!\!\perp X|W_1$ and thus wrongly detect $mns_X(A) = \{W_1\}$. But Line 7 in Fig. 8 will detect this failure since $A \perp\!\!\!\perp X|\{W_1, W_2, W_3\}$ but $A \not\perp\!\!\!\perp X|\{W_1, W_2\}$ and $A \not\perp\!\!\!\perp X|\{W_1, W_3\}$.*

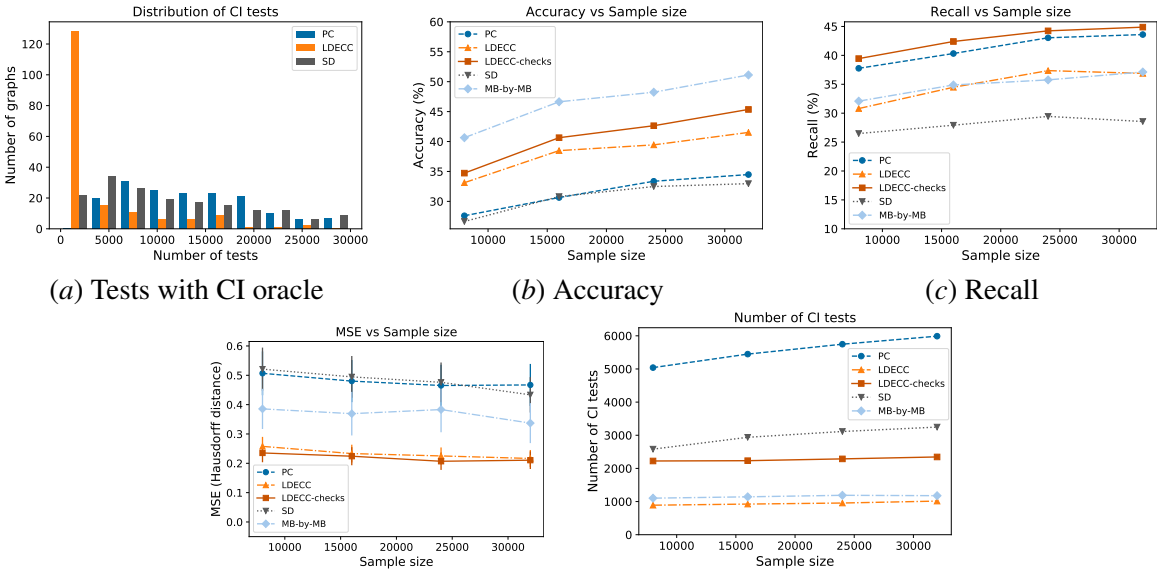
Proposition 24 (Testing FF for SD) *Consider running the algorithm in Fig. 9 with each UC detected by SD. If the algorithm returns* Fail*, then FF is violated for SD. If the algorithm returns* Unknown*, we could not ascertain if the FF assumptions for SD hold.*

Example 5 (Testing FF for SD) *Consider an OF violation for the DAG in Fig. 7(b) that causes $X \rightarrow M \leftarrow C$ to be detected as a non-collider. SD will mark both W and M as parents via the UC $A \rightarrow C \leftarrow B$. But Line 6 will return* Fail *because $\mathbf{M} = \{W, M\}$ and $\mathbf{Q}(A) = \mathbf{Q}(B) = \{\{W\}\}$.*

Proposition 25 (Hybrid procedure) *Consider the following procedure: (i) run SD to get Θ_{SD}; (ii) test for FF violations using Prop. 24; (iii) if a FF violation is found, run LDECC and output Θ_{LDECC}, else output Θ_{SD}. This procedure identifies Θ^* under strictly weaker assumptions than Prop. 18.*

Proof If the conditions of Prop. 18 hold, this procedure will output Θ_{SD} because we will not detect a FF violation. However, when a FF violation is detected, this procedure will identify Θ^* if the FF conditions for LDECC hold. Thus, this strictly weakens the assumptions needed to identify Θ^*. ■

We can construct a similar hybrid procedure for LDECC. This again shows the complementary nature of the two algorithms: If there are detectable FF violations for one algorithm, we have an alternative ATE set that we can output thereby allowing us to identify Θ^* under weaker FF assumptions than would be possible with access to only one of the algorithms.



(*a*) Tests with CI oracle (*b*) Accuracy (*c*) Recall

(*d*) MSE vs sample size (*e*) Number of CI tests

Figure 10: Results on synthetic linear Gaussian graphs.

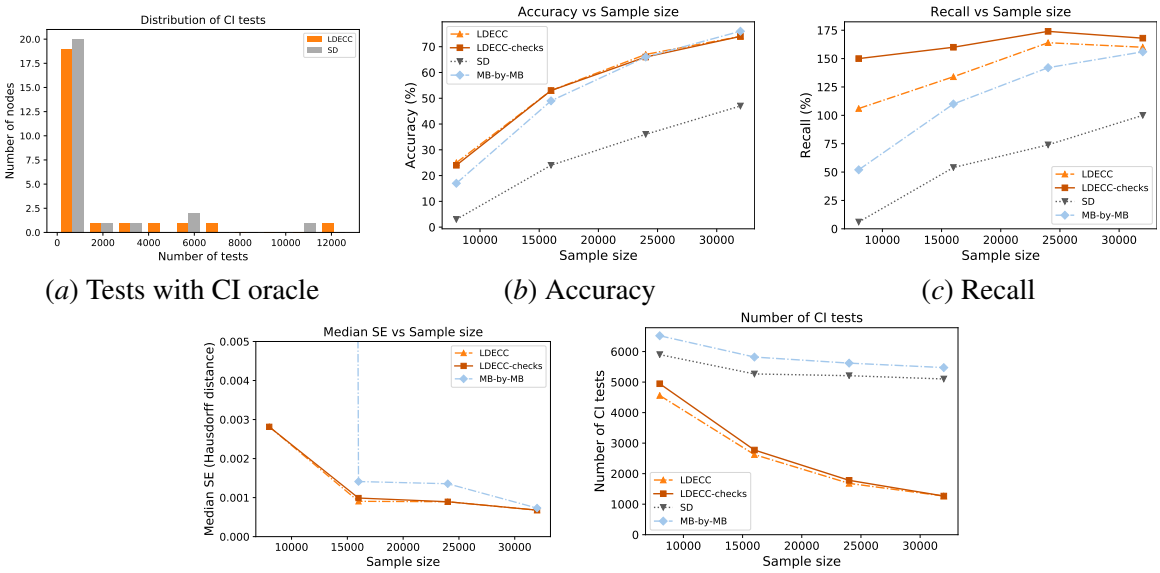
5. Experiments

We present results on both synthetic linear graphs and the semi-synthetic *MAGIC-NIAB* graph from *bnlearn* (Scutari, 2009). LDECC typically outperforms SD while running a similar number of CI tests. Results on synthetic binomial graphs and additional graphs from *bnlearn* are in Appendix C.

Results on synthetic data. We generated synthetic linear graphs with Gaussian errors, 20 covariates—non-descendants of X and Y with paths to both X and Y—and 3 mediators—nodes on some causal path from X to Y. We sampled edges between the nodes with varying probabilities and sampled the edge weights uniformly from $[-1, -0.25] \cup [0.25, 1]$ (see Appendix C.1 for the precise procedure). We compared LDECC, SD, and PC based on the number of CI tests performed with a CI oracle on 250 synthetic graphs (of which $\sim 77\%$ were *locally orientable*; see Defn. 9). The distribution of CI tests (Fig. 10(*a*)) shows that LDECC performs substantially fewer tests than both SD and PC for ~ 120 graphs and has similar performance as SD on the other graphs.

Next, we evaluated our methods on the 250 synthetic graphs at four sample sizes. From each graph, we generated data 5 times (so we have $N = 1250$ runs). We used the Fisher-z CI test and ordinary least squares regression for ATE estimation. We also ran LDECC with *check=True* as described in Remark 21 (denoted by *LDECC-checks*) and MB-by-MB (Wang et al., 2014), a local causal discovery algorithm similar to SD that uses IAMB (Fig. 14) and *LocalPC* for local structure learning. We define *accuracy* as the fraction of times we get the ATE set that we would estimate for the true DAG $\mathcal{G}^*$, and *recall* as the fraction of times the ATE estimated from the true DAG $\mathcal{G}^*$ is contained in the estimated ATE set, i.e., whether $\widehat{\theta}_{\text{ATE}}(\mathcal{G}^*) \in \widehat{\Theta}$. We used Hausdorff distance to compute mean squared error (MSE) between the estimated $\widehat{\Theta}$ and the ground-truth set Θ^*:

$$\text{MSE}_{\text{Hausdorff}}(\{\widehat{\Theta}\}_{i=1}^N, \{\Theta^*\}_{i=1}^N) = \frac{1}{N}\sum_{i=1}^N \max\left\{\sup_{u\in\widehat{\Theta}_i}\inf_{v\in\Theta_i^*}(u-v)^2, \sup_{v\in\Theta_i^*}\inf_{u\in\widehat{\Theta}_i}(u-v)^2\right\}.$$



(*a*) Tests with CI oracle (*b*) Accuracy (*c*) Recall

(*d*) Median SE vs sample size (*e*) Number of CI tests

Figure 11: Results on the semi-synthetic *MAGIC-NIAB* linear Gaussian graph.

LDECC has higher accuracy than both SD and PC (Fig. 10(*b*)) but lower than MB-by-MB. In terms of recall, LDECC does worse than PC but LDECC-checks outperforms PC, SD, and MB-by-MB suggesting that the check helps in practice (Fig. 10(*c*)). Both variants of LDECC have substantially lower MSE than MB-by-MB, PC, and SD (Fig. 10(*d*)) while still doing a similar number of CI tests to SD with LDECC-checks doing more tests than MB-by-MB (Fig. 10(*e*)).

Results on *MAGIC-NIAB* from bnlearn. *MAGIC-NIAB* is a linear Gaussian DAG with 44 nodes. With a CI oracle, PC performed $\sim 1.472 \times 10^6$ tests. For SD and LDECC, the distribution of CI tests on a subset of nodes (restricting to nodes where both algorithms perform < 20000 tests) shows that both algorithms perform comparably with SD doing slightly better (Fig. 11(*a*)). Next, we designated the nodes *G266* and *HT* as the treatment and outcome, respectively. We compared the methods by sampling the data 100 times at four sample sizes. LDECC has similar accuracy (Fig. 11(*b*)) and recall (Fig. 11(*c*)) as MB-by-MB but outperforms SD on both metrics. We plot *median* squared error (Fig. 11(*d*)) for MB-by-MB and LDECC (MB-by-MB does poorly on the mean due to outliers). Both are similar at large sample sizes (we omit SD because it does poorly). Additionally, LDECC performs fewer CI tests than MB-by-MB and SD (Fig. 11(*e*)).

6. Conclusion

Broadening the landscape of local causal discovery, we propose a new algorithm that uses ECCs to orient parents. We show that LDECC has different computational and faithfulness requirements compared to the existing class of sequential algorithms. This allows us to profitably combine the two methods to get polynomial runtimes on a larger class of graphs as well as identify the set of possible ATE values under weaker assumptions. In future work, we hope to further weaken the faithfulness requirements and extend our methods to handle causal graphs with latent variables.

References

Constantin F Aliferis, Ioannis Tsamardinos, and Alexander Statnikov. Hiton: a novel markov blanket algorithm for optimal variable selection. In *AMIA annual symposium proceedings*. American Medical Informatics Association, 2003.

Constantin F Aliferis, Alexander Statnikov, Ioannis Tsamardinos, Subramani Mani, and Xenofon D Koutsoukos. Local causal and markov blanket induction for causal discovery and feature selection for classification part i: algorithms and empirical evaluation. *Journal of Machine Learning Research*, 11(1), 2010.

Zhuangyan Fang and Yangbo He. Ida with background knowledge. In *Conference on Uncertainty in Artificial Intelligence*. PMLR, 2020.

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

Clark Glymour, Kun Zhang, and Peter Spirtes. Review of causal discovery methods based on graphical models. *Frontiers in genetics*, 2019.

Leonard Henckel, Emilija Perković, and Marloes H Maathuis. Graphical criteria for efficient total effect estimation via adjustment in causal linear models. *arXiv preprint arXiv:1907.02435*, 2019.

Amin Jaber, Jiji Zhang, and Elias Bareinboim. Causal identification under markov equivalence: Completeness results. In *International Conference on Machine Learning*. PMLR, 2019.

Zhaolong Ling, Kui Yu, Hao Wang, Lei Li, and Xindong Wu. Using feature selection for local causal structure learning. *IEEE Transactions on Emerging Topics in Computational Intelligence*, 2020.

Zhaolong Ling, Kui Yu, Yiwen Zhang, Lin Liu, and Jiuyong Li. Causal learner: A toolbox for causal structure and markov blanket learning. *arXiv preprint arXiv:2103.06544*, 2021.

Marloes H Maathuis and Diego Colombo. A generalized back-door criterion. *The Annals of Statistics*, 2015.

Marloes H Maathuis, Markus Kalisch, and Peter Bühlmann. Estimating high-dimensional intervention effects from observational data. *The Annals of Statistics*, 2009.

Daniel Malinsky and Peter Spirtes. Estimating causal effects with ancestral graph markov models. In *Conference on Probabilistic Graphical Models*. PMLR, 2016.

Christopher Meek. Causal inference and causal explanation with background knowledge. *arXiv preprint arXiv:1302.4972*, 2013.

Preetam Nandy, Marloes H Maathuis, and Thomas S Richardson. Estimating the effect of joint interventions from observational data in sparse high-dimensional settings. *The Annals of Statistics*, 2017.

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

Emilija Perković, Markus Kalisch, and Maloes H Maathuis. Interpreting and using cpdags with background knowledge. *arXiv preprint arXiv:1707.02171*, 2017.

Joseph Ramsey, Jiji Zhang, and Peter L Spirtes. Adjacency-faithfulness and conservative causal inference. *arXiv preprint arXiv:1206.6843*, 2012.

Andrea Rotnitzky and Ezequiel Smucler. Efficient adjustment sets for population average treatment effect estimation in non-parametric causal graphical models. *arXiv preprint arXiv:1912.00306*, 2019.

Marco Scutari. Learning bayesian networks with the bnlearn r package. *arXiv preprint arXiv:0908.3817*, 2009.

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

Chandler Squires and Caroline Uhler. Causal structure learning: a combinatorial perspective. *arXiv preprint arXiv:2206.01152*, 2022.

Jin Tian and Judea Pearl. *A general identification condition for causal effects*. eScholarship, University of California, 2002.

Ioannis Tsamardinos, Constantin F Aliferis, Alexander R Statnikov, and Er Statnikov. Algorithms for large scale markov blanket discovery. In *FLAIRS conference*. St. Augustine, FL, 2003.

Ioannis Tsamardinos, Laura E Brown, and Constantin F Aliferis. The max-min hill-climbing bayesian network structure learning algorithm. *Machine learning*, 2006.

Caroline Uhler, Garvesh Raskutti, Peter Bühlmann, and Bin Yu. Geometry of the faithfulness assumption in causal inference. *The Annals of Statistics*, 2013.

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*, 2014.

Jianxin Yin, You Zhou, Changzhang Wang, Ping He, Cheng Zheng, and Zhi Geng. Partial orientation and local structural learning of causal networks for prediction. In *Causation and Prediction Challenge*. PMLR, 2008.

Kui Yu, Xianjie Guo, Lin Liu, Jiuyong Li, Hao Wang, Zhaolong Ling, and Xindong Wu. Causality-based feature selection: Methods and evaluations. *ACM Computing Surveys (CSUR)*, 2020.

Jiji Zhang and Peter Spirtes. Detection of unfaithfulness and robust causal inference. *Minds and Machines*, 2008.

Jiji Zhang and Peter Spirtes. The three faces of faithfulness. *Synthese*, 2016.

You Zhou, Changzhang Wang, Jianxin Yin, and Zhi Geng. Discover local causal network around a target to a given depth. In *Causality: Objectives and Assessment*. PMLR, 2010.