LoSAM: Local Search in Additive Noise Models with Mixed Mechanisms and General Noise for Global Causal Discovery

Sujai Hiremath[1] **Promit Ghosal**[2] **Kyra Gan**[1]

[1]Cornell Tech, New York, NY
[2]University of Chicago, Chicago, Illinois

Abstract

Inferring causal relationships from observational data is crucial when experiments are costly or infeasible. *Additive noise model*s (ANMs) enable unique *directed acyclic graph* (DAG) identification, but existing sample-efficient ANM methods often rely on restrictive assumptions on the data generating process, limiting their applicability to real-world settings. We propose *local search in additive noise models*, LoSAM, a topological ordering method for learning a unique DAG in ANMs with mixed causal mechanisms and general noise distributions. We introduce new causal substructures and criteria for identifying roots and leaves, enabling efficient top-down learning. We prove asymptotic consistency and polynomial runtime, ensuring scalability and sample efficiency. We test LoSAM on synthetic and real-world data, demonstrating state-of-the-art performance across all mixed mechanism settings.

1 INTRODUCTION

Inferring causal relationships from observational data is crucial for answering interventional questions [Pearl, 2009], especially when experiments are costly or infeasible [Faller et al., 2024b]. *Functional causal model* (FCM) methods address this challenge by restricting the functional form of causal relationships, ensuring a well-posed structure learning problem that enables the identification of a unique DAG [Zhang and Hyvarinen, 2009]. FCM methods usually decompose graph learning into two phases: 1) inferring a causal ordering of the variables (topological ordering), and 2) identifying edges consistent with the ordering (edge pruning).

Among FCM approaches, those based on the ANM have gained significant traction [Hoyer et al., 2008, Peters et al., 2014, Rolland et al., 2022]. The ANM assumes that each child variable in the DAG is a (potentially nonlinear) function of its parent variables plus an independent noise term. This framework is sample-efficient, and worst-case polynomial algorithms exist for structural identification (under additional assumptions) [Peters et al., 2014]. These properties make ANM-based methods theoretically appealing, and they have been applied in domains such as science, healthcare, and economics [Runge et al., 2019, Lee et al., 2022, Addo et al., 2021].

However, their practical utility is often constrained by restrictive assumptions on functional forms and noise distributions (see Table 1). For example, most existing methods require all causal mechanisms to be either fully linear or nonlinear, limiting their applicability to real-world settings where mixed mechanisms might present. In biological networks, for instance, single-gene effects are assumed to be additive, while gene-gene interactions (epistasis) introduce nonlinear interactions [Kontio et al., 2020, Manicka et al., 2023, Faure et al., 2024].

While sample-efficient polynomial-time algorithms have been proposed under these restrictive assumptions for identifying a unique DAG (Table 1), their existence remains unknown for the general mixed-mechanism setting, where some relationships are linear and others are nonlinear. Achieving stable performance across diverse ANM settings is particularly challenging, as existing methods often leverage special statistical properties of causal substructures, like leaves and roots, that hold only under specific functional or distributional constraints. Recently, Xu et al. [2024] introduce CaPS, an efficient leaf-based topological ordering method that accommodates mixed causal mechanisms. However, it assumes Gaussian noise and imposes unverifiable assumptions on noise variances (see Appendix C.5), highlighting the need for more flexible solutions.

Contributions. In this paper, we propose LoSAM, a novel *topological ordering* method for ANMs that efficiently recovers the unique DAG *without* requiring additional assumptions. Our contributions are threefold:

Proceedings of the 41st Conference on Uncertainty in Artificial Intelligence (UAI 2025), PMLR 244:1677–1709.

Algorithm	ANM Type	Ordering Approach	Uses Leaf Properties	Uses Root Properties
DirectLiNGAM [Shimizu et al., 2011]	Linear Non-Gaussian ANM	Top-Down	✗	✓
LISTEN [Ghoshal and Honorio, 2018]	Linear ANM†	Bottom-Up	✓	✗
CAM [Bühlmann et al., 2014]	Nonlinear Additive Gaussian ANM	–	–	–
SCORE [Rolland et al., 2022]	Nonlinear Gaussian ANM	Bottom-Up	✓	✗
DAS [Montagna et al., 2023c]	Nonlinear Gaussian ANM	Bottom-Up	✓	✗
DiffAN [Sanchez et al., 2023]	Nonlinear Gaussian ANM	Bottom-Up	✓	✗
RESIT [Peters et al., 2014]	ANM	Bottom-Up	✓	✗
NoGAM [Montagna et al., 2023b]	Nonlinear ANM	Bottom-Up	✓	✗
NHTS [Hiremath et al., 2024]	Nonlinear ANM	Top-Down	✗	✓
CaPS [Xu et al., 2024]	Gaussian ANM††	Bottom-Up	✓	✗
Adascore [Montagna et al., 2025]	ANM	–†††	✓	–
LoSAM (ours)	ANM	Top-Down	✓	✓

† LISTEN requires an additional condition on the inverse of the covariance matrix for identifiability.
†† CaPS requires additional conditions on marginal variances and the score function for identifiability.
††† Unlike other methods, Adascore directly returns a DAG rather than a topological ordering.

Table 1: Comparison of FCM methods based on identifiability assumptions, search strategy (top-down or bottom-up), and whether specific substructures (leaf or root nodes) are leveraged.

- **Root and Leaf Identification:** We introduce new local causal substructures: *single root descendants*, *multi-root descendants*, *v-patterns* (Section 3), *valid leaf candidates*, *nonlinear descendants*, and *linear descendants* (Section 4). We characterize their statistical properties in ANMs with mixed mechanisms, without relying on distributional constraints. Leveraging these structures, we establish novel criteria for identifying roots (Lemma 3.3) and leaves (Lemma 4.1). Building on these, we introduce LoSAM (Algorithm 3), a topological sorting algorithm that learns a DAG in a top-down manner by first identifying the roots, then using them to recursively identify leaves.
- **Theoretical Guarantees:** We prove that LoSAM is asymptotically consistent under identifiable ANMs (Theorem 4.5), ensuring correct discovery in the limit. Further, we establish the worst-case polynomial runtime of LoSAM (Theorem 4.6), enabling scaling to larger graphs. We show that LoSAM achieves theoretical gains in sample efficiency when compared to prior methods, in both the root-finding stage (Theorem 3.6) and overall procedure (Theorem 4.7).
- **Comprehensive Evaluation:** We extensively evaluate LoSAM on synthetic data, achieving state-of-the-art performance in linear and nonlinear settings, while outperforming baselines in mixed mechanism settings. We validate the real-world applicability of LoSAM on a popular biological dataset (Section 5). The source code for LoSAM is publicly available at `https://github.com/Sujai1/local-search-discovery`.

1.1 RELATED WORKS

Causal discovery methods mostly fall into three categories: constraint-based [Spirtes et al., 2000, Spirtes, 2001], scoring-based [Chickering, 2013, Lam et al., 2022], and FCM-based [Glymour et al., 2019]. The first two categories of methods return an equivalence class of models with the same conditional independence structure [Pearl, 2009], rather than a unique DAG [Montagna et al., 2023a]. Moreover, these methods can exhibit poor sample efficiency [Maasch et al., 2024], and generally suffer exponential worst-case time complexity in the number of variables unless sparsity constraints are imposed [Chickering et al., 2004, Ganian et al., 2024, Colombo et al., 2012, Claassen et al., 2013].

In contrast, existing FCM methods return a unique DAG in polynomial time, enabling point estimates for downstream causal effects (rather than bounds, e.g., Malinsky and Spirtes 2016). The topological ordering step in FCM methods mostly falls into two categories: score-matching-based methods and regression-based methods. As shown in Table 1, FCM methods typically leverage statistical properties of roots or leaves unique to different parametric models to recursively construct a topological ordering. Score-matching-based methods (LISTEN, SCORE, DAS, DiffAN, NoGAM, CaPS, Adascore[1]) typically rely on large conditioning sets or Gaussianity assumptions to estimate the score-function, leading to low sample efficiency [Hiremath et al., 2024, Montagna et al., 2023b].

Regression-based methods exploit the independence of the noise term to identify causal relationships. RESIT regresses leaves onto unsorted vertices, yielding an independent residual in both linear and nonlinear ANMs. However, it relies on high-dimensional nonparametric regression, suffering in sample complexity [Peters et al., 2014]. DirectLiNGAM

[1]Adascore leverages score-matching in mixed mechanism ANMs without explicitly constructing an ordering, but has empirical performance similar to NoGAM [Montagna et al., 2025].

takes a top-down approach but heavily relies on the linearity assumption. Closest to our approach is NHTS, which first identifies roots in nonlinear models, and then leverages them to reduce the size of conditioning sets in subsequent steps. However, NHTS fundamentally fails at handling linear mechanisms, and can face sample efficiency issues when there are many roots, as illustrated in Appendix I.1. In contrast, LoSAM extends beyond prior topological ordering algorithms by handling mixed mechanisms, without imposing additional assumptions on the noise distribution.

CAM does not fall into either category. It leverages MLE in nonlinear additive Gaussian models, rather than causal substructures, to construct a topological ordering. In our experiments, we include CAM as one of our benchmarks.

Once a topological ordering is obtained from an FCM method, it is straightforward to prune spurious edges via sparse regression (Lasso regression [Marra and Wood, 2018]) additive hypothesis testing with generalized additive models (CAM-pruning [Bühlmann et al., 2014]), or conditional independence tests (Edge Discovery [Hiremath et al., 2024]).

2 PROBLEM SETUP

An structural causal model (SCM) is represented by a DAG, denoted as $G = (V, E)$ on $|V| = d$ vertices, where E represents directed edges. An edge $x_i \to x_j \in E$ *iff* (if and only if) x_i has a *direct* causal influence on x_j. We define four pairwise relationships between vertices: 1) $\text{Ch}(x_i)$ denotes the *children* of x_i such that $x_j \in \text{Ch}(x_i)$ iff $x_i \to x_j$, 2) $\text{Pa}(x_i)$ denotes the *parents* of x_i such that $x_j \in \text{Pa}(x_i)$ iff $x_j \to x_i$, 3) $\text{An}(x_i)$ denotes the *ancestors* of x_i such that $x_j \in \text{An}(x_i)$ iff there exists a directed path $x_j \dashrightarrow x_i$, 4) $\text{De}(x_i)$ denotes the *descendants* of x_i such that $x_j \in \text{De}(x_i)$ iff there exists a directed path $x_i \dashrightarrow x_j$. We consider four vertex categories based on the totality of their pairwise relationships: 1) x_i is a *root* iff $\text{Pa}(x_i) = \emptyset$, 2) a *leaf* iff $\text{Ch}(x_i) = \emptyset$, 3) an *isolated* vertex iff x_i is both a root and a leaf, and 4) an *intermediate* vertex otherwise.

We also classify vertices in terms of their triadic relationships: x_i is a *confounder* of x_j and x_k if $x_i \in \text{An}(x_j) \cap \text{An}(x_k)$, or a *mediator* of x_j to x_k if $x_i \in \text{De}(x_j) \cap \text{An}(x_k)$.

Definition 2.1 (Topological Ordering)**.** *Consider a DAG denoted as $G = (V, E)$. We say that a mapping $\pi : V \to \{0, 1, \ldots, |V|\}$ is a linear topological sort iff $\forall x_j \in V$, whenever $x_i \in Pa(x_j)$, x_i appears before x_j in the sort π, i.e., $\pi(x_i) < \pi(x_j)$.*

Definition 2.2 (ANMs, Hoyer and Hyttinen 2009)**.** *Additive noise models are a specific class of SCMs with*

$$x_i = f_i(Pa(x_i)) + \varepsilon_i \tag{1}$$

$\forall x_i \in V$, where f_i's are arbitrary functions and ε_i's are independent arbitrary noise distributions.

Assumptions. Definition 2.2 implies the Causal Markov condition due to the joint independence of noise terms ε_i; we further assume that all variables are observed, acyclicity, faithfulness [Spirtes and Zhang, 2016], as well as the unique identifiability of the ANM. Intuitively, the identifiability condition rules out specific combinations of causal mechanisms and noise distributions, such as linear f_i and Gaussian ε_i, but allows for a broad mix of linear and nonlinear functions with general noise distributions (see Appendix C for details).

Outline. LoSAM is a topological ordering algorithm that follows a two-step procedure: 1) identifying the root vertices (Algorithm 1, Section 3), and 2) leveraging the roots to recursively identify leaf vertices of the sorted nodes, yielding the rest of the topological sort (Algorithm 2, Section 4). In Section 4.1, we build upon the above subroutines and introduce the full procedure of LoSAM (Algorithm 3).

3 ROOT FINDING

To develop a subroutine for identifying roots in ANMs with both linear and nonlinear causal mechanisms, we identify properties unique to roots. We will establish a novel set of local causal structures where roots behave distinctly from non-roots (Lemmas 3.1, 3.2, and 3.3), regardless of their functional relationships. Starting with the entire vertex set V, we iteratively prune away non-roots by leveraging these asymmetric properties, yielding only the roots.

We first note that non-roots can be partitioned into two distinct categories: those descending from a single root and those descending from multiple roots.

Definition 3.1 (Single Root Descendant)**.** *A vertex $x_i \in V$ is a* single root descendant *(SRD) iff $\exists$ only one root x_j, such that $x_j \in An(x_i)$.*

Definition 3.2 (Multi-Root Descendant)**.** *A vertex $x_i \in V$ is a* multi-root descendant *(MRD) iff $\exists$ at least two roots x_j, x_k such that $x_j, x_k \in An(x_i)$.*

Visually, we see this division of non-root vertices into SRDs and MRDs in Figure 1. For any MRD $x_i \in V$, we observe that x_i forms a 'v' with any two of its *root ancestors*. We define this ancestor-descendant local substructure as a *v-pattern* (VP),[2] and give a characterization based on marginal dependence constraints:

Definition 3.3 (VP)**.** *We say that a vertex x_i induces a* v-pattern *iff there exist two vertices $x_j, x_k \in V$ such that $x_i \not\perp\!\!\!\perp x_j, x_i \not\perp\!\!\!\perp x_k, x_j \perp\!\!\!\perp x_k$.*

Definition 3.3 is distinct from but related to the statistical constraints required by the notion of 'v-structures' [Spirtes

[2]Note that an MRD x_i will form additional VPs with *any* pair of independent ancestors.

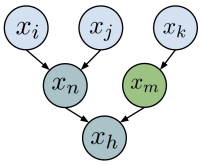

Figure 1: Example DAG where x_i, x_j, x_k are roots, x_m is an SRD, and x_n, x_h are MRDs, inducing VPs between x_i, x_j.

et al., 2000], a triplet of vertices where one vertex is a child of two parents that do not share a direct edge between them (see Appendix I.2 for details).

Prune MRDs. Our root-finding procedure starts by pruning MRDs, leveraging VPs as described in Lemma 3.1.

Lemma 3.1 (MRD Induces VP)**.** *A vertex x_i induces a VP iff x_i is an MRD.*

The proof of Lemma 3.1 (Appendix D.1) relies on the fact that all vertices in V are partitioned into roots, SRDs, and MRDs. We further observe that 1) an MRD induces a VP because roots are all independent of each other, and 2) an SRD or a root does not induce VP.

Lemma 3.1 implies that by checking whether a vertex induces a VP between any two vertices in V, we can identify and prune MRDs from V, leaving a $V' \subset V$ containing only SRDs and roots. It remains to prune SRDs from V' and identify roots.

Identify Isolated Roots. Next, in Lemma 3.2 (proof in Appendix D.2), we show that a root with no SRDs can be identified by testing for independence from the other variables in V':

Lemma 3.2 (Root with No SRDs)**.** *For any $x_i \in V'$, iff $x_i \perp\!\!\!\perp x_j \; \forall x_j \in V' \setminus \{x_i\}$, x_i is a root with no SRDs.*

Prune SRDs. Let V'' be the remaining vertices after identifying roots with no SRDs using Lemma 3.2. This set consists of roots with at least one SRD, along with the SRDs themselves. To recover the roots, we follow a two-step procedure: first, we use pairwise nonparametric regression to recover a subset of V'' containing all roots, then we prune away the remaining non-roots using bivariate nonparametric regression.

We first formally state the outcome of a nonparametric regression residual test below, considering both linear and nonlinear mechanisms.

Definition 3.4 (Regression-Identification Test)**.** *Let $x_i, x_j \in V''$, and let r_{ij} be the residual of x_j nonparametrically regressed on x_i. Then, x_i is identified as $\in An(x_j)$ iff: 1) $x_i \perp\!\!\!\perp r_{ij}$, and 2) $x_j \not\perp\!\!\!\perp r_{ji}$.*

Definition 3.4 defines the relation of "is identified as" between x_i and x_j as when the residuals from the regressions can be leveraged to identify that $x_i \in An(x_j)$. Note that nonparametric regression yields an independent residual whenever the dependent variable is an additive function of the regressor. Therefore, roots are always identified as ancestors of SRDs that are their children. However, it is possible that some SRDs will pass the Regression-Identification Test; one such subcase occurs when an SRD has a child, and is its only parent.

The Regression-Identification Test parallels a similar idea in Hiremath et al. 2024 (Lemma 4.3), where their local-search approach leverages the result of regression and independence tests to find potential roots. However, their framework exploits local parent-child substructures (PP1, PP2, PP3, PP4) intrinsic to nonlinear models, while the Regression-Identification test captures general ancestor-descendant relationships that may occur in linear, nonlinear, and mixed mechanism models. Additionally, the Regression-Identifiation test is a univariate regression, rather than the multivariate regression employed in Lemma 4.3 of Hiremath et al. 2024.

Using the above Regression-Identification test, Lemma 3.3 establishes a subset W of V'' that contains all the roots, and further shows that the roots can be distinguished from non-roots within W:

Lemma 3.3 (Root ID)**.** *Let W be a subset of V'' such that $\forall x_i \in W$, 1) $\exists x_j \in V$ such that x_i is identified as $\in An(x_j)$, and 2) $\nexists x_k \in V$ such that x_k is identified as $\in An(x_i)$. Then W contains all roots in V''.*

For $x_i, x_j, x_k \in V$, let r^k_{ij} be the residual of x_k nonparametrically regressed onto both x_i and x_j. Then, vertex $x_i \in W$ is a root vertex if and only if for every other vertex $x_j \in W$ such that $x_i \not\perp\!\!\!\perp x_j$, $\forall x_k$ such that x_j is identified $\in An(x_k)$, we have $x_j \perp\!\!\!\perp r^k_{ij}$.

To distinguish roots from SRDs that pass the Regression-Identification test, the proof of Lemma 3.3 relies on the intuition that if $x_i \in W$ is a root (a nondescendant of any vertex in W), then adding it as a covariate should not change the independence results of any pairwise regression. However, the reverse does not hold for any non-root SRD: the inclusion of these variables in the bivariate regression would yield at least one dependent residual when the regression involves its ancestor. We formally state this intuition mathematically in Appendix D.3.

Root Finder. Algorithm 1 outlines our root-finding procedure. In Stage 1, we run marginal independence tests between all vertices, leveraging Lemmas 3.1 and 3.2 to prune MRDs and roots with no SRDs, leaving SRDs and their root ancestors. In Stage 2, we run pairwise nonparametric regressions and independence tests on regressors and residuals to obtain the root superset W. We then apply Lemma 3.3, using conditional independence tests to identify the remaining

Algorithm 1 Root Finder

Input: vertices $x_1, \ldots, x_d \in V$.
Initialize: root set RT, root superset W.
Stage 1: Prune MRDs, Obtain Some Roots
Run pairwise $\perp\!\!\!\perp$ tests between each pair of vertices $x_i, x_j \in V$ and remove all vertices that induce a VP (MRDs) from V to obtain V'.
Add any vertex $x_i \in V'$ to RT if $x_i \perp\!\!\!\perp x_j \forall x_j \in V' \setminus \{x_i\}$, and remove all x_i from V' to obtain V''.
Stage 2: Prune SRDs, Obtain Leftover Roots
Run pairwise nonparametric regression between $x_i, x_j \in V''$; if $\exists x_l$ such that $x_k \in V''$ is identified as $\in \text{An}(x_j)$, and there does not exist x_h such that x_h is identified $\in \text{An}(x_k)$, add x_k to W.
For $x_i \in W$, if $\forall x_j \in W \setminus \{x_i\}$ such that $x_j \not\perp\!\!\!\perp x_i$, $\exists x_k \in V''$ such that x_i is identified as $\in \text{An}(x_k)$ and $x_k \not\perp\!\!\!\perp x_j | x_i$, add x_i to RT.
For each $x_i, x_j \in W$ such that $x_i \not\perp\!\!\!\perp x_j$, for all x_k such that x_j is identified $\in \text{An}(x_k)$ regress x_k onto x_i, x_j and collect the residual; add any x_i that is always independent of the residual to RT.
return RT.

roots. We show the asymptotic correctness of Algorithm 1 in Proposition 3.5 (proof in Appendix E.1):

Proposition 3.5. *Given the vertices of a DAG G generated by an ANM, infinite data, a consistent nonparametric regression method, and a perfect independence test, Algorithm 1 returns the correct set of root vertices.*

We note that, under the assumption of linear mechanisms, root-based methods such as DirectLiNGAM are able to identify roots with only univariate regressions; in contrast, root-based nonlinear methods such as NHTS and leaf-based methods such as RESIT and NoGAM require multivariate nonparametric regression with potentially many covariates. Large covariate set size is a major bottleneck for sample and computational efficiency that limits many FCM-based methods from recovering the true sort from finite samples, despite asymptotic guarantees [Peters et al., 2014]. To our knowledge, our approach is the first to correctly identify root vertices in ANMs with mixed mechanisms and general noise with a bounded maximum covariate set size. In Theorem 3.6 (proof in Appendix F.1), we formally show the reduction in the size of the conditioning sets used in Alg 1:

Theorem 3.6. *Given a DAG $G = (V, E)$ under an ANM, let $M \subseteq V$ and $R \subseteq V$ be the sets of MRDs and roots in G, respectively. Let $c_{\max}^{\text{Alg1}}$ be the max covariate set size in nonparametric regressions in Algorithm 1 required to recover roots, and similarly $c_{\max}^{\text{DirectLiNGAM}}$, $c_{\max}^{\text{NHTS}}$, $c_{\max}^{\text{NoGAM}}$, $c_{\max}^{\text{RESIT}}$. If the ANM is linear, then $c_{\max}^{\text{Alg1}} = c_{\max}^{\text{DirectLiNGAM}} = 1$. If the ANM is nonlinear, Then, $c_{\max}^{\text{Alg1}} = 2$, $c_{\max}^{\text{NHTS}} = \max_{x_i \in M}(An(x_i) \cap R, 0)$, and $c_{\max}^{\text{RESIT}} = c_{\max}^{\text{NoGAM}} = d - 2$.*

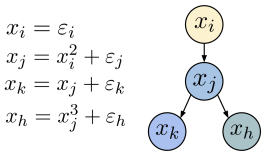

Figure 2: Exemplary DAG, where x_i has been sorted into π, x_j is a VLC, x_k is a LD, x_h is a ND.

This implies that $c_{\max}^{\text{Alg1}} < c_{\max}^{\text{RESIT}} = c_{\max}^{\text{NoGAM}}$, and when $M \neq \emptyset$, $c_{\max}^{\text{Alg1}} \leq c_{\max}^{\text{NHTS}}$.

Note, the results of Theorem 3.6 hold under the assumptions that all independence tests and regression outcomes are correct. By leveraging the existence of VPs to prune MRDs, we limit the size of covariate sets used in nonparametric regressions in Algorithm 1 to a maximum size of two. Although Algorithm 1 may run potentially more pairwise regressions, avoiding multivariate regressions likely leads to improved sample complexity in many settings, which is confirmed by our experimental results in Section 5.

4 SORT-FINDING

After obtaining the set of roots, we develop Algorithm 2, which sorts the remaining non-roots into a topological ordering π. We start by adding roots to π in any order (as they share no edges), leaving the unsorted vertices $U = V \setminus \pi$. We follow an iterative procedure by removing one vertex at a time from U and adding it to π.

To ensure the resulting π is valid (Definition 2.1), this vertex must be a leaf of the subgraph induced by the sorted vertices. Therefore, in each iteration, we identify a *valid leaf candidate* (VLC) as illustrated in Figure 2:

Definition 4.1 (VLC). *A vertex $x_i \in U$ is a* valid leaf candidate *iff $Pa(x_i) \cap U = \emptyset$.*

We proceed by first partitioning non-VLC vertices in U into different categories, relying on novel local causal substructures (Definitions 4.2 and 4.3). We then leverage asymmetric properties to prune non-VLCs from U, allowing the identification of a VLC at each iteration.

We first consider a non-VLC $x_i \in U$, such that there exists a mediator $x_j \in \text{An}(x_i) \cap U$ between vertices in π to x_i, and x_i is a nonlinear function of x_j. In Definition 4.2, we characterize such vertices as *nonlinear descendants* (ND), illustrated in Figure 2:

Definition 4.2 (ND). *A vertex $x_i \in U$ is a ND iff x_i is a nonlinear function of at least one $x_j \in U$.*

We observe that if any mediator $x_j \in U$ between vertices in π and an ND $x_i \in U$ is not included in the regression of

x_i onto vertices in π, it will introduce omitted variable bias [Pearl et al., 2016], resulting in a dependent residual e_i. In other words, if x_i is nonparametrically regressed onto the sorted vertices in π, producing a residual e_i, then e_i will not be independent of at least one vertex in π. Accordingly, we identify NDs in Lemma 4.1 (proof in Appendix D.4).

Lemma 4.1 (ND Test)**.** *Vertex x_i is an ND iff e_i is dependent on at least one sorted vertex in π.*

Let U_E be the set of residuals where $e_i \in U_E$ corresponds to the residual from regressing $x_i \in U$ onto all vertices in π. Lemma 4.1 implies that if $x_i \in U$ is a VLC, then $e_i \in U_E$ is independent of all sorted vertices in π.

When all causal mechanisms are nonlinear, all non-VLCs in U are NDs. However, when linear mechanisms are allowed, there may exist additional vertices $\in U$ that are neither VLCs nor NDs. These are the *linear descendants* (LD), as illustrated in Figure 2.

Definition 4.3 (LD)**.** *A vertex x_i is a* linear descendant *iff x_i is a linear function of all ancestors in U, and x_i is either a linear or nonlinear function of ancestors in π.*

We say that "x_i is a linear function of all ancestors in U, and x_i is either a linear or nonlinear function of ancestors in π" to mean that, if we decompose the ancestors of x_i, $An(x_i)$, into two disjoint subsets: 1) the ancestors in U ($x_u \in An(x_i) \cap U$), and 2) the ancestors in π ($x_p \in An(x_i) \cap \pi$), then for any representation of x_i as a function of x_u and other ancestors or noise ($x_i = f_i(x_u, An(x_i) \setminus x_u) + \varepsilon_i$), f_i cannot be nonlinear in x_u.

Next, in Lemma 4.2, we show that LDs and VLCs share the same independence conditions:

Lemma 4.2 (LD independence)**.** *If $x_i \in U$ is an LD, $e_i \in U_E$ is independent of all sorted vertices in π.*

Lemma 4.2 (proof in Appendix D.5) explains why prior regression-based methods that leverage roots (DirectLiNGAM, NHTS) fail when both linear and nonlinear mechanisms are present: LDs cannot be distinguished from VLCs via residual independence tests alone. When both mechanisms are present, to identify VLCs, it remains to prune LDs from U, obtaining a subset of vertices U'. Then, to improve the stability of the algorithm under a finite sample,[3] we select a vertex from U' that minimizes a test statistic, rather than checking directly for residual independence.

Pruning LDs. Lemma 4.3 establishes a subset Q of U that distinguishes LDs from VLCs, leveraging nonparametric regression:

Lemma 4.3. *For $e_i, e_j \in U_E$, let q_{ij} be the residual of e_j linearly regressed onto e_i, and q_{ji} be the residual of e_i linearly regressed onto e_j. Let Q be the set of all $x_i \in U$ such that there exists $x_j \in U$ such that the following conditions hold: 1) $e_j \perp\!\!\!\perp q_{ji}$, 2) $e_i \not\perp\!\!\!\perp q_{ij}$.*

Then $Q \subseteq U$ contains all LDs $\in U$ and no VLCs.

To distinguish LDs from NDs and VLCs, the proof of Lemma 4.3 relies on the intuition that, for any LD $x_i \in U$, there exists a VLC x_j such that x_j is an ancestor of x_i, and x_i is a linear function of x_j. This allows us to decompose the residual of x_i, e_i, into a linear function of the residual of x_j, e_j, yielding an independent residual when e_i is linearly regressed onto e_j, but a dependent residual in the reverse direction. This intuition is formalized mathematically in Appendix D.6. Lemma 4.3 enables us to prune LDs from U, leaving the subset U' containing only VLCs and NDs.

Improved Stability for ND Pruning. To determine whether a residual e_i from regression is independent of its regressors, prior work [Peters et al., 2014, Hiremath et al., 2024] requires the independence of each residual from *each* regressor. Accordingly, to prune NDs from U', we might rely on testing the independence of each residual $e_i \in U_E$ to *all* sorted vertices. While this procedure is asymptotically unbiased, in practice it requires selecting a cutoff for the independence test given the finite samples. If not done carefully, this can lead to erroneous VLC identification.[4]

To improve the numerical stability of our procedure, we propose the following test statistic, which captures the dependence between the residual e_i and all sorted vertices in π in a *continuous* manner:

$$t^*(e_i, \pi) = \sum_{x_j \in \pi} \widehat{MI}(x_j, e_i), \tag{2}$$

where $\widehat{MI}(\cdot, \cdot)$ is a nonparametric estimator of the mutual information [Kraskov et al., 2004]. Instead of selecting a single cutoff to decide independence between the residual e_i and all vertices in π, at each iteration, we choose the vertex in U' with the lowest test statistic, $x_i = \arg\min_{x_j \in U'} t^*(e_j, \pi)$, to be a VLC, making our procedure more robust under finite samples.

Lemma 4.4 (proof in Appendix D.7) shows that the test statistics $t^*(e_i, \pi)$ equals zero asymptotically only for x_i that are VLCs:

Lemma 4.4 (Consistency)**.** *Given a consistent estimator $\widehat{MI}(\cdot, \cdot)$, a fixed $x_i \in U'$ and corresponding residual e_i, $t^*(e_i, \pi)$ asymptotically approaches 0 as $n \to \infty$ iff e_i is independent of all vertices in π, i.e., x_i is a VLC.*

[3] If finite sample errors lead to even one false test result when checking the independence of a VLC's residual, the VLC would not be identified.

[4] Bootstrap procedures may be used to compute an optimal cutoff for any given set of samples, but become computationally expensive when n is large.

Algorithm 2 Sort Finder

Input: vertices $x_1, \ldots, x_d \in V$, roots RT
Initialize: add roots in RT to π, unsorted vertices $U = V \setminus \pi$.
while $U \neq \emptyset$ **do**:
    Stage 1: Prune U
    Run a nonparametric regression of each vertex in U onto all vertices in π, and collect the resulting residuals in the set U_E.
    Run pairwise linear regression between each pair of residuals $e_i, e_j \in U_E$ and collect residuals q_{ij}, q_{ji}; remove all $x_i \in U$ such that $e_j \perp\!\!\!\perp q_{ji}, e_i \not\perp\!\!\!\perp q_{ij}$ from U to obtain U'.
    Stage 2: Identify VLC
    Identify $x_* = \arg\min_{x_j \in U'} t^*(e_j, \pi)$ as a VLC, add x_* to π, remove x_* from U.
return π.

Algorithm 3 LoSAM

Input: vertices $x_1, \ldots, x_d \in V$
Run *Root Finder* (Alg 1), obtain RT.
Run *Sort Finder* (Alg 2) using RT, obtain sort π.
return π.

Sort Finder. Leveraging Lemma 4.4, we propose Algorithm 2, Sort Finder, which iteratively builds the topological sort π in a two-stage procedure: Stage 1 runs nonparametric regressions of each vertex in U onto all vertices in π to obtain the residual set U_E; we prune all LDs in U using Lemma 4.3, obtaining the subset U'. In Stage 2, we identify a VLC by finding the vertex $x_i \in U'$ that minimizes the test statistic $t^*(e_i, \pi)$, according to Lemma 4.4. We repeat this procedure until all vertices are sorted. We provide asymptotic correctness of Algorithm 2 in Proposition 4.4 (proof in Appendix E.2).

Proposition 4.4 (Correctness of Sort Finder)**.** *Given the vertices of a DAG G generated by an ANM, the roots in G, infinite data, a consistent nonparametric regression method, and a perfect independence test, Algorithm 2 returns a valid sort π.*

4.1 THEORETICAL GUARANTEES

By combining Algorithms 1 and 2, we describe our overall topological sort algorithm, *local search in additive noise models*, LoSAM, in Algorithm 3. LoSAM extends prior top-down regression-based FCM methods to ANMs with both linear and nonlinear mechanisms. It reduces the maximum size of conditioning sets in the root identification phase to two and improves algorithm stability under finite samples by selecting the vertex with the smallest test statistic at each iteration of the sorting procedure.

We provide the correctness of Algorithm 3 in Theorem 4.5 (proof in Appendix F.2), and a step-by-step walk-through of the method in Appendix H.1.

Theorem 4.5 (LoSAM Correctness)**.** *Given the vertices of a DAG G generated by an ANM, infinite data, a consistent nonparametric regression method, and a perfect independence test, Algorithm 3 returns a valid topological sort π.*

Next, we establish the worst-case time complexity in Theorem 4.6 (proof in Appendix F.2),

Theorem 4.6 (LoSAM Runtime)**.** *Given n samples generated from a d-dimensional DAG G under an ANM, the worst-case time complexity of Algorithm 3 is $O(d^3 n^2)$.*

The worst case runtime of LoSAM in dimensionality $O(d^3)$ is slightly higher than the $O(d^2)$ complexity of methods that learn the ordering from the leaves to the roots (bottom-up) and do not require Gaussianity, such as RESIT and NoGAM. However, it matches the complexity of NHTS ($O(d^3)$), a top-down algorithm that also learns the ordering from the roots to the leaves. Correspondingly, LoSAM enjoys improvements in sample efficiency common to root-based methods.

Efficiency. The number of multivariate nonparametric regressions run in each step of LoSAM is actually inversely related to the size of the covariate sets (similar to the root-based method NHTS), while the number of regressions in each step of leaf-based methods such as RESIT and NoGAM are directly proportionate to the covariate set size. Therefore, LoSAM preserves the reduction in covariate set size inherent to root-based methods over leaf-based methods, which leads to better sample efficiency when estimating nonlinear relationships. We provide a formal analysis of the reduction in complexity for the worst case (i.e., in a fully connected DAG) in Theorem 4.7 (proof in Appendix F.4):

Theorem 4.7 (LoSAM Efficiency)**.** *Consider a fully connected DAG $G = (V, E)$ with ANM. Let $d := |V|$. Let n_k^{LoSAM} be the number of multivariate nonparametric regressions with covariate set size $k \in [d-2]$ run by LoSAM when sorting V; we similarly define $n_k^{\text{NHTS}}, n_k^{\text{RESIT}}$ and n_k^{NoGAM} respectively. Then, $n_k^{\text{LoSAM}} = n_k^{\text{NHTS}} = d - k$, and $n_k^{\text{RESIT}} = n_k^{\text{NoGAM}} = k + 1$. This implies that for all $k > \frac{d}{2}$, $n_k^{\text{LoSAM}} = n_k^{\text{NHTS}} < n_k^{\text{RESIT}} = n_k^{\text{NoGAM}}$.*

5 EXPERIMENTAL RESULTS

We evaluate LoSAM on synthetic datasets with mixed, purely linear, and purely nonlinear mechanisms, as well as a real-world protein expression dataset.[5] LoSAM achieves state-of-the-art performance, outperforming existing algorithms in mixed mechanism settings.

[5] https://github.com/Sujai1/local-search-discovery.

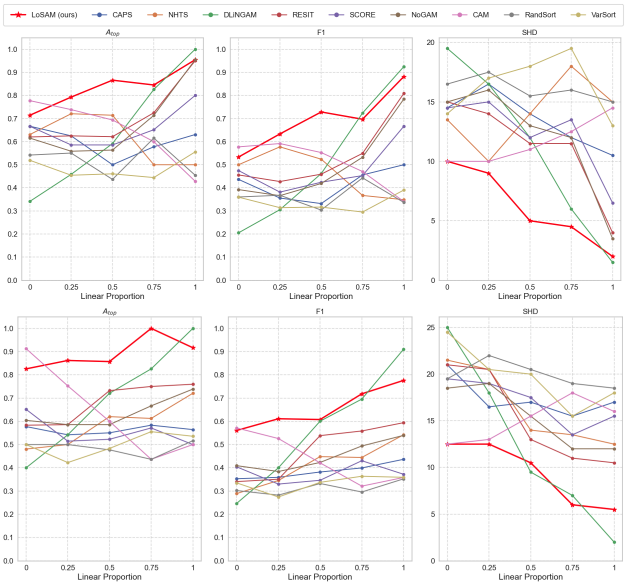

Figure 3: LoSAM performance on sparse graphs by linear mechanism proportion. Top row: uniform noise. Bottom row: Laplace noise.

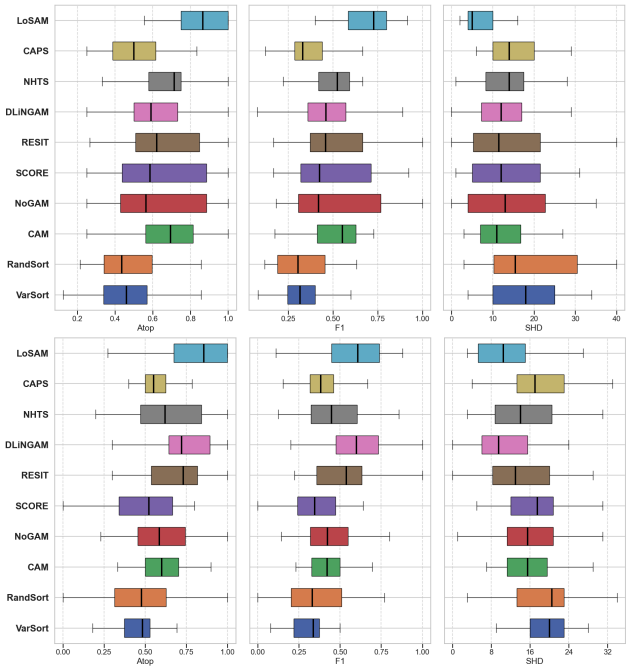

Figure 4: Performance of LoSAM on ER1 synthetic data with 50% proportion of linear mechanisms. Top row: uniform noise. Bottom row: Laplace noise.

Synthetic Dataset Generation. We produce synthetic data with varying graph sparsity, exogenous error distribution, dimensionality, and causal mechanisms. DAGs are randomly generated with the Erdos-Renyi model [Erdos and Renyi, 1960]; the average number of edges in each d-dimensional DAG is either d (ER1) for sparse DAGs, or $2d$ (ER2) for dense DAGs. Uniform, Laplace or Gaussian noise is used as the exogenous error. We randomly sample the data ($d = 10, n = 1000$) according to causal mechanisms with a different average proportion of linear mechanisms ($0\%, 25\%, 50\%, 75\%, 100\%$). We standardize and process the data to ensure that the simulated data is sufficiently challenging; methods are evaluated on 30 randomly generated seeds in each experimental setting. Further details can be found in Appendix G.

Real-World Data. To confirm the real-world applicability of our approach, we test LoSAM on the Sachs dataset [Sachs et al., 2005], a widely used real-world biological benchmark for causal discovery (see Appendix G.2 for details). The Sachs data captures the expression levels of various proteins in human cells, with a ground-truth causal network established by genetic experiments. Notably, prior work [Xu et al., 2024] has estimated that the causal relationships in the Sach's network have mixed mechanisms, making it a promising test case for assessing LoSAM.

Baselines. We benchmark LoSAM against a mix of classical and SOTA topological ordering baselines: DirectLiNGAM, CAM, RESIT, SCORE, NoGAM, NHTS, and CaPS. We include the heuristic algorithm Var-Sort [Reisach et al., 2021] and a randomly generated sort (Rand-Sort) to measure gameability, as some FCM methods are prone to exploiting artifacts common to simulated ANMs [Reisach et al., 2021].

Evaluation and Metrics. We first directly evaluate the topological orderings via the average topological divergence A_{top} (higher A_{top} is better), which is equal to the percentage of edges that can be recovered by the returned topological ordering (an edge cannot be recovered if a child is sorted before a parent) [Hiremath et al., 2024]. We note that A_{top} is a normalized version of the topological ordering divergence D_{top} defined in Rolland et al. [2022] (see Appendix G.3 for details). To produce a predicted causal graph, we apply a standard edge pruning method (CAM-pruning, Bühlmann et al. 2014) to the fully dense graph corresponding to each topological ordering. We adopt the Structural Hamming Distance (SHD) [Tsamardinos et al., 2006] and F1 score for evaluation; the SHD is the sum of false positive, false negative and reversed edges (lower SHD is better), whereas the F1 score measures the balance between precision and recall of predicted edges (higher F1 is better).

Results on Synthetic Data. Figure 3 demonstrates the robustness of LoSAM as the proportion of linear and nonlinear mechanisms changes in sparse graphs (ER1). LoSAM achieves similar performance to SOTA methods in both the linear and nonlinear settings, under both noise distributions. In the mixed mechanism setting, LoSAM significantly outperforms all baselines in the uniform noise setting, and all baselines except DirectLiNGAM (which performs poorly in nonlinear ANM) in the laplacian setting; the degradation in baseline performance when assumptions are violated

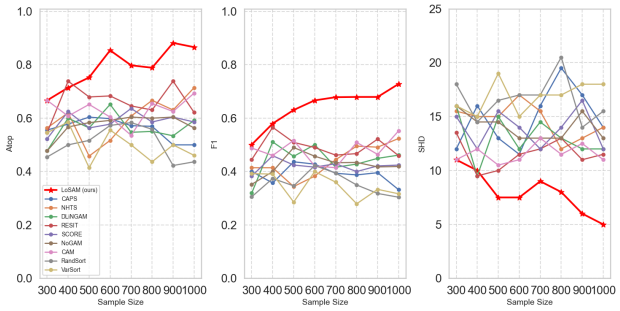

Figure 5: LoSAM performance on ER1 synthetic data with Uniform noise and 50% proportion of linear mechanisms, with increasing sample size $n = 300, 400 \ldots, 1000$.

Metrics	$A_{top}\uparrow$	F1↑	SHD↓
DirectLiNGAM	0.56±0.13	0.32±0.10	**29.90±4.13**
CAM	0.61±0.12	0.30±0.07	38.60±4.87
RESIT	0.41±0.09	0.21±0.08	35.07±4.23
SCORE	0.28±0.02	0.17±0.03	37.67±3.10
NoGAM	0.28±0.03	0.17±0.03	38.50±4.05
NHTS	0.59±0.12	0.28±0.08	35.97±5.41
CaPS	0.29±0.03	0.18±0.02	37.03±2.52
VarSort	0.46±0.03	0.25±0.03	35.33±2.47
RandSort	0.47±0.13	0.24±0.08	38.33±5.54
LoSAM	**0.62±0.09**	**0.33±0.05**	34.97±4.74

Table 2: Results on real-world Sachs protein dataset.

highlights the limited applicability of current FCM methods.

Figure 4 demonstrates the overall performance of LoSAM in sparse graphs (ER1) where the proportion of linear mechanisms is fixed to 50%. LoSAM outperformed all baselines across all three metrics, especially for uniform noise, demonstrating the enhanced sample-efficiency of our root and leaf based procedure. We note that, as expected from Theorem 3.6 and Theorem 4.7, LoSAM had higher sample computational efficiency than nonlinear methods NHTS, RESIT and NoGAM (SCORE and CAM as well), with up to 2-5× faster runtime (see Appendix G.6).

Figure 5 shows how the performance of LoSAM changes as the sample size increases from $n = 300$ to $n = 1000$ (in sparse ER1 graphs with $d = 10$, Uniform noise, 50% linear mechanisms). The results demonstrate LoSAM's superior sample efficiency, as its performance improves fastest with increasing sample size and it consistently outperforms baselines. This aligns with Theorems 3.6 and 4.7, and shows that reduced conditioning set sizes do indeed enhance statistical efficiency.

Additional Synthetic Experiments. To confirm the robustness of LoSAM, we present results in dense, high-dimensional, and Gaussian noise settings. Additionally, we test the sensitivity of LoSAM to estimation error.

In Appendix G.5.1, we examine the performance of LoSAM on denser graphs ($d = 10, n = 1000$). We find that LoSAM still maintains a performance gap over baselines, likely due to the reduced covariate set size shown in Theorem 4.7.

To understand the scalability of LoSAM we fix the proportion of linear mechanisms to 0.5, sample size $n = 2000$, noise to be uniform, and evaluate over increasing dimensionality ($d = 10, 20, \ldots, 50$) in Appendix G.5.2. We find that LoSAM maintains performance gains over baselines, demonstrating its efficacy in higher-dimensional settings.

We test the performance of LoSAM under the nonlinear Gaussian setting in Appendix G.5.3 ($d = 10, n = 1000$), a setting that is considered in prior benchmarks. We find that LoSAM achieves similar performance to CAM, which is designed to exploit Gaussian noise in its procedure, and significantly outperforms all other baselines (including other Gaussian-specific methods, e.g., SCORE and CaPS).

Finally, in Appendix G.5.4, we investigate LoSAM's sensitivity to estimation error ($d = 10, n = 300$, Uniform noise) by evaluating LoSAM variants that use different regression estimators (Random Forest, Kernel Ridge Regression with various kernels, etc.). LoSAM maintains consistent accuracy across estimators, confirming its robustness—an essential property for finite-sample settings where regression efficiency may vary.

Results on Real-World Data. The results of the Sachs dataset are presented in Table 2: LoSAM achieves the best A_{top} and F1 score, with SHD coming in second to DirectLiNGAM. We note that the SHD metric favors the prediction of sparse causal graphs [Xu et al., 2024], as it double counts errors when edges are reversed, whereas F1 and A_{top} penalize all errors equally. Therefore, taking all three metrics into account, we conclude that LoSAM outperforms baselines.

Discussion. Future work includes extending LoSAM to handle latent confounding, and applying the local search approach to a time series setting. Additionally, we aim to build upon our theoretical guarantees and develop statistical sample complexity bounds for LoSAM, extending previously derived results [Zhu et al., 2023] for FCM methods in nonlinear Gaussian ANMs.

Acknowledgments
This paper is supported by an AWS Credits Grant from The Center for Data Science for Enterprise and Society at Cornell University.

References

Peter Martey Addo, Christelle Manibialoa, and Florent McIsaac. Exploring nonlinearity on the co2 emissions, economic production and energy use nexus: a causal discovery approach. *Energy Reports*, 7:6196–6204, 2021.

Kevin Bello, Bryon Aragam, and Pradeep Ravikumar. DAGMA: Learning DAGs via M-matrices and a Log-Determinant Acyclicity Characterization. In *Advances in Neural Information Processing Systems*, 2022.

Peter Bühlmann, Jonas Peters, and Jan Ernest. CAM: Causal additive models, high-dimensional order search and penalized regression. *The Annals of Statistics*, 42(6), December 2014. ISSN 0090-5364. doi: 10.1214/14-AOS1260. arXiv:1310.1533 [cs, stat].

David Maxwell Chickering. Learning Equivalence Classes of Bayesian Network Structures. *Journal of Machine Learning Research*, 2013.

David Maxwell Chickering, Christopher Meek, and David Heckerman. Large-sample learning of bayesian networks is np-hard. *Journal of Machine Learning Research*, 5: 1287–1330, 2004.

Tom Claassen, Joris M Mooij, and Tom Heskes. Learning Sparse Causal Models is not NP-hard. *Proceedings of the Twenty-Ninth Conference on Uncertainty in Artificial Intelligence (UAI2013)*, 2013.

Diego Colombo, Marloes H. Maathuis, Markus Kalisch, and Thomas S. Richardson. Learning high-dimensional directed acyclic graphs with latent and selection variables. *The Annals of Statistics*, 40(1):294–321, 2012.

Paul Erdos and Alfred Renyi. On the evolution of random graphs. *Publication of the Mathematical Institute of the Hungarian Academy of Sciences*, 1960.

Philipp Faller, Leena Vankadara, Atalanti Mastakouri, Francesco Locatello, and Dominik Janzing. Self-compatibility: Evaluating causal discovery without ground truth. *International Conference on Artifical Intellgience and Statistics*, 2024a.

Philipp Faller, Leena Vankadara, Atalanti Mastakouri, Francesco Locatello, and Dominik Janzing. Self-compatibility: Evaluating causal discovery without ground truth. *International Conference on Artifical Intellgience and Statistics*, 2024b.

Andre J Faure, Ben Lehner, Verónica Miró Pina, Claudia Serrano Colome, and Donate Weghorn. An extension of the walsh-hadamard transform to calculate and model epistasis in genetic landscapes of arbitrary shape and complexity. *PLOS Computational Biology*, 20(5):e1012132, 2024.

Robert Ganian, Viktoriia Korchemna, and Stefan Szeider. Revisiting causal discovery from a complexity-theoretic perspective. In *Proceedings of the Thirty-Third International Joint Conference on Artificial Intelligence (IJCAI-24)*, pages 3377–3385. International Joint Conferences on Artificial Intelligence Organization, 2024.

Asish Ghoshal and Jean Honorio. Learning linear structural equation models in polynomial time and sample complexity. In *International Conference on Artificial Intelligence and Statistics*, pages 1466–1475. PMLR, 2018.

Clark Glymour, Kun Zhang, and Peter Spirtes. Review of Causal Discovery Methods Based on Graphical Models. *Frontiers in Genetics*, 10:524, June 2019. ISSN 1664-8021. doi: 10.3389/fgene.2019.00524.

Sujai Hiremath, Jacqueline Maasch, Mengxiao Gao, Promit Ghosal, and Kyra Gan. Hybrid top-down global causal discovery with local search for linear and nonlinear additive noise models. *Proceedings of the 38th Conference on Neural Information Processing Systems (NeurIPS)*, 2024. https://arxiv.org/abs/2405.14496.

Patrik O Hoyer and Antti Hyttinen. Bayesian discovery of linear acyclic causal models. In *Proceedings of the 25th Conference on Uncertainty in Artificial Intelligence*, 2009.

Patrik O Hoyer, Dominik Janzing, Joris M Mooij, Jonas Peters, and Bernhard Schölkopf. Nonlinear causal discovery with additive noise models. In *Advances in Neural Information Processing Systems 21 (NIPS 2008)*, 2008.

Nan Ke, Silvia Chiappa, Jane Wang, Jorg Bornschein, Anirudh Goyal, Melanie Rey, Theophane Weber, Matthew Botvinick, Michael Mozer, and Danilo Rezende. Learning to induce causal structure. *International Conference on Learning Representations*, 2023.

Juho A. J. Kontio, Marko J. Rinta-aho, and Mikko J. Sillanpää. Estimating linear and nonlinear gene coexpression networks by semiparametric neighborhood selection. *Genetics*, 215(3):597–607, July 2020. doi: 10.1534/genetics.120.303186.

Alexander Kraskov, Harald Stögbauer, and Peter Grassberger. Estimating mutual information. *Physical Review E*, 69(6):066138, June 2004. ISSN 1539-3755, 1550-2376. doi: 10.1103/PhysRevE.69.066138.

Wai-Yin Lam, Bryan Andrews, and Joseph Ramsey. Greedy Relaxations of the Sparsest Permutation Algorithm, 2022.

Jaron J.R. Lee, Ranjani Srinivasan, Chin Siang Ong, Diane Alejo, Stefano Schena, Ilya Shpitser, Marc Sussman, Glenn J.R. Whitman, and Daniel Malinsky. Causal determinants of postoperative length of stay in cardiac surgery using causal graphical learning. *The Journal of Thoracic*

and Cardiovascular Surgery, page S002252232200900X, August 2022. ISSN 00225223. doi: 10.1016/j.jtcvs.2022.08.012.

Phillip Lippe, Taco Cohen, and Efstratios Gavves. Efficient neural causal discovery without acyclicity constraints. *International Conference on Learning Representations*, 2022.

Jacqueline 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 *Proceedings of the 40th Conference on Uncertainy in Artificial Intelligence*, 2024. doi: https://doi.org/10.48550/arXiv.2310.17816.

Daniel Malinsky and Peter Spirtes. Estimating causal effects with ancestral graph markov models. In *Proceedings of the Conference on Probabilistic Graphical Models (PGM)*, pages 319–330. PMLR, 2016.

Santosh Manicka, Karl Johnson, Michael Levin, and Guy Karlebach. The nonlinearity of regulation in biological networks. *npj Systems Biology and Applications*, 9(10), April 2023. doi: 10.1038/s41540-023-00273-w.

Giampiero Marra and Simon Wood. Regression Shrinkage and Selection via the Lasso., 2018.

Francesco Montagna, Atalanti A. Mastakouri, Elias Eulig, Nicoletta Noceti, Lorenzo Rosasco, Dominik Janzing, Bryon Aragam, and Francesco Locatello. Assumption violations in causal discovery and the robustness of score matching. In *37th Conference on Neural Information Processing Systems*, October 2023a.

Francesco Montagna, Nicoletta Noceti, Lorenzo Rosasco, Kun Zhang, and Francesco Locatello. Causal Discovery with Score Matching on Additive Models with Arbitrary Noise. In *Proceedings of the 2nd Conference on Causal Learning and Reasoning*. arXiv, April 2023b. arXiv:2304.03265 [cs, stat].

Francesco Montagna, Nicoletta Noceti, Lorenzo Rosasco, Kun Zhang, and Francesco Locatello. Scalable Causal Discovery with Score Matching. In *Proceedings of the 2nd Conference on Causal Learning and Reasoning*. arXiv, April 2023c. arXiv:2304.03382 [cs, stat].

Francesco Montagna, Philipp M Faller, Patrick Bloebaum, Elke Kirschbaum, and Francesco Locatello. Score matching through the roof: linear, nonlinear, and latent variables causal discovery. *Causal Learning and Reasoning (CLeaR), PMLR.*, 2025.

Gunwoong Park. Identifiability of additive noise models using conditional variances. *Journal of Machine Learning Research*, 21(75):1–34, 2020.

Gunwoong Park and Youngwhan Kim. Identifiability of gaussian linear structural equation models with homogeneous and heterogeneous error variances. *Journal of the Korean Statistical Society*, 49:276–292, 2020. doi: 10.1007/s42952-020-00018-7.

Judea Pearl. Causal inference in statistics: An overview. *Statistics Surveys*, 3, January 2009. ISSN 1935-7516. doi: 10.1214/09-SS057.

Judea Pearl, Madelyn Glymour, and Nicholas P. Jewell. *Causal inference in statistics: a primer*. Wiley, Chichester, West Sussex, 2016. ISBN 978-1-119-18684-7.

J. Peters and P. Bühlmann. Identifiability of gaussian structural equation models with equal error variances. *Biometrika*, 101(1):219–228, March 2014. doi: 10.1093/biomet/ast043.

Jonas Peters, Joris Mooij, Dominik Janzing, and Bernhard Schölkopf. Causal Discovery with Continuous Additive Noise Models, April 2014. arXiv:1309.6779 [stat].

Alexander Reisach, Christof Seiler, and Sebastian Weichwald. Beware of the simulated dag! causal discovery benchmarks may be easy to game. *Advances in Neural Information Processing Systems*, 34:27772–27784, 2021.

Alexander G. Reisach, Myriam Tami, Christof Seiler, Antoine Chambaz, and Sebastian Weichwald. A Scale-Invariant Sorting Criterion to Find a Causal Order in Additive Noise Models. In *37th Conference on Neural Information Processing Systems*. arXiv, October 2023. arXiv:2303.18211 [cs, stat].

Paul Rolland, Volkan Cevher, Matthaus Kleindessner, Chris Russel, Bernhard Scholkopf, Dominik Janzing, and Francesco Locatello. Score Matching Enables Causal Discovery of Nonlinear Additive Noise Models. In *Proceedings of the 39 th International Conference on Machine Learning*, 2022.

Jakob Runge, Sebastian Bathiany, Erik Bollt, Gustau Camps-Valls, Dim Coumou, Ethan Deyle, Clark Glymour, Marlene Kretschmer, Miguel D. Mahecha, Jordi Muñoz-Marí, Egbert H. van Nes, Jonas Peters, Rick Quax, Markus Reichstein, Marten Scheffer, Bernhard Schölkopf, Peter Spirtes, George Sugihara, Jie Sun, Kun Zhang, and Jakob Zscheischler. Inferring causation from time series in Earth system sciences. *Nature Communications*, 10(1):2553, December 2019. ISSN 2041-1723. doi: 10.1038/s41467-019-10105-3.

Karen Sachs, Omar Perez, Dana Pe'er, Douglas A Lauffenburger, and Garry P Nolan. Causal protein-signaling networks derived from multiparameter single-cell data. *Science*, 308(5721):523–529, 2005.

Pedro Sanchez, Xiao Liu, Alison Q O'Neil, and Sotirios A Tsaftaris. Diffusion models for causal discovery via topological ordering. *The Eleventh International Conference on Learning Representations (ICLR 2023)*, 2023. URL https://arxiv.org/abs/2210.06201.

Shohei Shimizu, Takanori Inazumi, Yasuhiro Sogawa, Aapo Hyvarinen, Yoshinobu Kawahara, Takashi Washio, Patrik O Hoyer, Kenneth Bollen, and Patrik Hoyer. Directlingam: A direct method for learning a linear non-gaussian structural equation model. *Journal of Machine Learning Research-JMLR*, 12(Apr):1225–1248, 2011.

Peter Spirtes. An Anytime Algorithm for Causal Inference. In *Proceedings of the Eighth International Workshop on Artificial Intelligence and Statistics*, volume R3, pages 278–285. PMLR, 2001.

Peter Spirtes and Richard Scheines. Causal inference of ambiguous manipulations. *Philosophy of Science*, 2004.

Peter Spirtes and Kun Zhang. Causal discovery and inference: concepts and recent methodological advances. *Applied Informatics*, 3(1):3, December 2016. ISSN 2196-0089. doi: 10.1186/s40535-016-0018-x.

Peter Spirtes, Clark Glymour, and Richard Scheines. *Causation, Prediction, and Search*, volume 81 of *Lecture Notes in Statistics*. Springer New York, New York, NY, 2000. ISBN 978-1-4612-7650-0 978-1-4612-2748-9. doi: 10.1007/978-1-4612-2748-9.

Ioannis Tsamardinos, Laura E. Brown, and Constantin F. Aliferis. The max-min hill-climbing Bayesian network structure learning algorithm. *Machine Learning*, 65(1): 31–78, October 2006. ISSN 0885-6125, 1573-0565. doi: 10.1007/s10994-006-6889-7.

Zhuopeng Xu, Yujie Li, Cheng Liu, and Ning Gui. Ordering-based causal discovery for linear and nonlinear relations. In *Proceedings of the 37th Conference on Neural Information Processing Systems (NeurIPS)*, 2024.

Kun Zhang and Aapo Hyvarinen. On the Identifiability of the Post-Nonlinear Causal Model. *Uncertainty in Artificial Intelligence*, 2009.

Kun Zhang, Jonas Peters, Dominik Janzing, and Bernhard Scholkopf. Kernel-based Conditional Independence Test and Application in Causal Discovery. *Proceedings of the Twenty-Seventh Conference on Uncertainty in Artificial Intelligence*, 2011.

Zhenyu Zhu, Francesco Locatello, and Volkan Cevher. Sample Complexity Bounds for Score-Matching: Causal Discovery and Generative Modeling, 2023. URL https://arxiv.org/abs/2310.18123.

APPENDIX

A NOTATION

$G = (V, E)$	A DAG G with $\lvert V\rvert = d$ vertices, where E represents the set of directed edges between vertices.
$\mathrm{Ch}(x_i)$	The set of child vertices of x_i.
$\mathrm{Pa}(x_i)$	The set of parent vertices of x_i.
$\mathrm{De}(x_i)$	The set of vertices that are descendants of x_i.
$\mathrm{An}(x_i)$	The set of vertices that are ancestors of x_i.
π	A topological ordering of vertices, i.e. a mapping $\pi : V \to \{0, 1, \ldots, d\}$.
f_i	An arbitrary function, used to generate vertex x_i.
ε_i	An independent noise term sampled from an arbitrary distribution, used to generate vertex x_i.
SRD	A single root descendant, a vertex with only one root ancestor.
MRD	A multi root descendant, a vertex with at least two root ancestors.
VP	A v-pattern, a causal substructure between three vertices.
V'	A subset of V that contains only and all roots and SRDs.
V''	A subset of V' that contains only and all SRDs and roots with at least one SRD.
r_{ij}	The residual of the vertex x_j nonparametrically regressed on the vertex x_i.
W	A subset of V'' that contains only and all vertices identified by the Regression-Identification Test.
M	The set of MRDs in V.
R	The set of roots in V.
$c_{\max}^{\mathrm{Alg1}}$	The maximum size of conditioning sets used in Algorithm 1.
$c_{\max}^{\mathrm{DirectLiNGAM}}$	The maximum size of conditioning sets used in the root identification for DirectLiNGAM.
$c_{\max}^{\mathrm{NHTS}}$	The maximum size of conditioning sets used in the root identification step of NHTS.
$c_{\max}^{\mathrm{RESIT}}$	The maximum size of conditioning sets used to identify roots in RESIT.
$c_{\max}^{\mathrm{NoGAM}}$	The maximum size of conditioning sets used to identify roots in NoGAM.
U	The set of vertices left unsorted, i.e., not yet added to π.
VLC	A valid leaf candidate; a vertex in U with no parents in U.
U_E	The set of residuals produced from regressing each $x_i \in U$ onto vertices in π.
e_i	A residual in U_E corresponding to $x_i \in U$.
ND	A nonlinear descendant; a vertex in U that is a nonlinear function of at least one vertex in U.
LD	A linear descendant; a vertex in U that is a linear function of all vertices in U.
q_{ij}	The residual produced by linearly regressing e_j onto e_i.
Q	A subset of U that contains all LDs, but no VLCs.
U'	A subset of U that contains all VLCs and some NDs.
$\widehat{MI}(x, y)$	A nonparametric estimator of mutual information between x, y.
$t^*(e_i, \pi)$	A test statistic corresponding to $x_i \in U'$.
r_{ij}^k	The residual produced by nonparametrically regressing x_k onto x_i, x_j.

B GRAPH TERMINOLOGY

In this section we clarify the term 'd-separation', a foundational concept for analyzing causal graphical models [Spirtes et al., 2000]. First, we classify the types of paths that exist between vertices, then define d-separation in terms of those paths.

We first note that a path between x_i, x_k can either start and end with an edge out of x_j, x_k ($x_k \dashrightarrow \cdots \dashleftarrow x_k$), start with an edge of out of x_j and end with an edge into x_k ($x_j \dashrightarrow \cdots \dashrightarrow x_k$) or start with an edge into x_j and edge with an edge into x_k ($x_j \dashleftarrow \cdots \dashrightarrow x_k$). Paths such as x_k ($x_k \dashrightarrow \cdots \dashleftarrow x_k$) do not transmit causal information between x_j, x_k. Undirected paths that transmit causal information between two vertices x_j, x_k can be differentiated into *frontdoor* and *backdoor paths* [Spirtes et al., 2000]. A frontdoor path is a directed path $x_j \dashrightarrow \cdots \dashrightarrow x_k$, while a backdoor path is a path $x_j \dashleftarrow \cdots \dashrightarrow x_k$.

Paths between two vertices are further classified, relative to a vertex set $\mathbf{Z}$, as either *active* or *inactive* [Spirtes et al., 2000]. A path between vertices x_j, x_k is active relative to $\mathbf{Z}$ if every node on the path is active relative to $\mathbf{Z}$. Vertex $x_i \in V$ is active on path relative to $\mathbf{Z}$ if one of the following holds: 1) $x_i \in \mathbf{Z}$ and x_i is a collider 2) $x_i \notin \mathbf{Z}$ and x_i is not a collider 3) $x_i \notin \mathbf{Z}$, x_i is a collider, but $\text{De}(x_i) \cap \mathbf{Z} \neq \emptyset$. An inactive path is simply a path that is not active. Causally paths are typically described active or inactive with respect to $\mathbf{Z} = \emptyset$ unless otherwise specified.

Vertices x_i, x_j are said to be d-separated by a set $\mathbf{Z}$ iff there is no active path between x_i, x_j relative to Z.

C ASSUMPTIONS

C.1 CAUSAL MARKOV

The Causal Markov condition implies that a variable x_i is independent of all non-descendants x_j, given its parents $\text{Pa}(x_i)$ [Spirtes, 2001]. Equivalently, we say that an ANM $G = (V, E)$ satisfies the Causal Markov Condition if the joint distribution $p_V(V)$ over all $x_i \in V$ admits the following factorization:

$$p_V(V) = \prod_i^d p_i(x_i|\text{Pa}(x_i)). \tag{3}$$

C.2 ACYCLICITY

We say that a causal graph G is acyclic if there does not exist any directed cycles in G [Spirtes, 2001].

C.3 FAITHFULNESS

Note that by assuming that a causal graph G satisfies the Causal Markov assumption, we assume that data produced by the DGP of G satisfies all independence relations implied by G [Spirtes, 2001]. However, this does not necessarily imply that all independence relations observed in the data are implied by G. Under the additional assumption of faithfulness, the independence relations implied by G are the *only* independence relations found in the data generated by G's DGP [Spirtes, 2001].

C.4 IDENTIFIABILITY OF ANM

Following the style of [Montagna et al., 2023a], we first observe that the following condition guarantees that the observed distribution of a pair of variables x_i, x_j can only be generated by a unique ANM:

Condition C.1 (Hoyer and Hyttinen 2009)**.** *Given a bivariate model $x_i = \varepsilon_i, x_j = f_j(x_i) + \varepsilon_j$ generated according to (1), we call the SEM an identifiable bivariate ANM if the triple $(f_i, p_{\varepsilon_i}, p_{\varepsilon_j})$ does not solve the differential equation $k''' = k''(-\frac{g'''f'}{g''} + \frac{f''}{f'}) - 2g''f''f' + g'f''' + \frac{g'g'''f''f'}{g''} - \frac{g'(f'')^2}{f'}$ for all x_i, x_j such that $f'(x_i)g''(x_j - f_j(x_i)) \neq 0$, where $p_{\varepsilon_i}, p_{\varepsilon_j}$ are the density of $\varepsilon_i, \varepsilon_j$, $f = f_j, k = \log p_{\varepsilon_i}, g = p_{\varepsilon_j}$. The arguments $x_j - f_j(x_i), x_i$ and x_i of g, k and f respectively, are removed for readability.*

There is a generalization of this condition to the multivariate ANM proved by [Peters et al., 2014]:

Theorem C.2. *(Peters et al. 2014). An ANM corresponding to DAG G is identifiable if $\forall x_j \in V, x_i \in Pa(x_j)$ and all sets $S \subseteq V$ with $Pa(x_j) \setminus \{i\} \subseteq S \subseteq \overline{De(j)} \setminus \{x_i, x_j\}$, $\exists\ X_S$ with positive joint density such that the triple $\left(f_j(Pa(j) \setminus \{x_i\}, x_i), p_{x_i|X_s}, p_{\varepsilon_j}\right)$ satisfies Condition C.1, and f_j are non-constant in all arguments.*

In this paper, we assume that all DAGs are generated by identifiable ANMs, as defined in Theorem C.2.

Our work builds on the classical ANM identifiability assumptions from [Peters et al., 2014] (Theorem C.2). Intuitively, these ensure each ANM generates a unique joint distribution, enabling unique identification. This identifiability condition rules out specific mechanisms-noise pairs such as linear f_i and Gaussian ϵ_i. Formally, for any $x_i = f_i(Pa(x_i)) + \varepsilon_i$ where f_i is linear in at least one of the parents in Pa_i, then ε_i must be non-gaussian. Thus, LoSAM does not cover linear Gaussian ANMs (LiGAM) but applies to any linear, nonlinear, or mixed-mechanism setting satisfying Theorem C.2.

While LiGAMs are generally non-identifiable, additional assumptions can enable identification: equal/known error variance [Peters and Bühlmann, 2014], heteroscedastic errors via variance/edge-weight conditions [Ghoshal and Honorio, 2018, Park and Kim, 2020], conditional variance constraints [Park, 2020], Varsortability/R^2-sortability [Reisach et al., 2021, 2023], and score function restrictions [Xu et al., 2024]. While extending LoSAM to LiGAM is an interesting research direction, the focus of this work is on providing a polynomial-time discovery algorithm for general ANMs with mixed mechanisms, without relying on strong functional, variance, or distributional assumptions. We leave extensions to LiGAM for future work.

C.5 ASSUMPTIONS OF CAPS ON NOISE DISTRIBUTION

The causal discovery method CaPS [Xu et al., 2024] requires that the noise terms be Gaussian, and that at least one of the following conditions holds (see section 3.1 and 4.1 of their paper):

CaPS Assumptions *(Sufficient conditions for identifiability).*

1. *Non-decreasing variance of noises.* For any two noises ϵ_i and ϵ_j, $\sigma_j \geq \sigma_i$ if $\pi(i) < \pi(j)$.
2. *Non-weak causal effect.* For any non-leaf nodes x_j,

$$\sum_{i \in Ch(j)} \frac{1}{\sigma_i^2} \mathbb{E}\left[\left(\frac{\partial f_i}{\partial x_j}(pa_i(x))\right)^2\right] \geq \frac{1}{\sigma_{\min}^2} - \frac{1}{\sigma_j^2},$$

where $\sigma_{\min}$ is the minimum variance for all noises. They comment that condition 1 is an extension of the equal variance assumption Peters and Bühlmann [2014], while condition 2 is a new sufficient condition that quantifies a lower bound of identifiable causal effects.

We note that, in contrast, LoSAM does not require or leverage either of the above conditions to recover a correct topological ordering, and allows for general noise distributions (beyond Gaussian noise).

D LEMMA PROOFS

D.1 PROOF OF LEMMA 3.1

Lemma 3.1 (MRD Induces VP)**.** *A vertex x_i induces a VP iff x_i is an MRD.*

Proof. Suppose x_i is an MRD. Then, $\exists x_j, x_k$ such that $x_j, x_k \in \text{An}(x_i)$, and x_j, x_k are roots. Note that this implies that $x_j \perp\!\!\!\perp x_k, x_i \not\perp\!\!\!\perp x_j, x_i \not\perp\!\!\!\perp x_k$, which is a VP between x_j, x_k induced by x_i.

Suppose x_i is not an MRD. We prove by contradiction that x_i cannot induces a VP.

Suppose x_i is an SRD. Let x_p be any vertex such that $x_p \not\perp\!\!\!\perp x_i$, which implies that there must exist an active causal path between x_i, x_p. Note, as x_i is an SRD, there exists only one root vertex x_j such that $x_j \in \text{An}(x_i)$. Suppose for contradiction that $x_p \notin \text{De}(x_j)$. Suppose there is a frontdoor path from x_i to x_p: then $x_p \in \text{De}(x_i) \implies x_p \in \text{De}(x_j)$, which contradicts our assumption that $x_p \notin \text{De}(x_j)$. Suppose there is a frontdoor path from x_p to x_i: as $x_p \notin \text{De}(x_j)$, WLOG $\exists$ a root

$x_m \neq x_j$ such that $x_p \in \text{De}(x_m)$. This implies that $x_i \in \text{De}(x_m)$ which contradicts the assumption that x_i is an SRD. Suppose for contradiction that there exists a backdoor path between x_p, x_i: this implies that $\exists\, x_c$ that is a confounder between x_p, x_i. Again, this implies that WLOG $\exists$ a root $x_m \neq x_j$ such that $x_c \in \text{De}(x_m) \implies x_i \in \text{De}(x_m)$, which contradicts the assumption that x_i is an SRD. Therefore, it must be that, for any vertex $x_p \not\perp\!\!\!\perp x_i$, $x_i, x_p \in \text{De}(x_j)$, where x_j is a root. Note, the above implies that for any vertices x_j, x_k such that $x_j \not\perp\!\!\!\perp x_i, x_k \not\perp\!\!\!\perp x_i$, we have $x_j, x_k \in \text{De}(x_j) \implies x_j \not\perp\!\!\!\perp x_k$. This implies that x_i cannot induce a VP between any two vertices x_j, x_k.

Suppose x_i is a root. Let x_j, x_k be any two vertices such that $x_j \not\perp\!\!\!\perp x_i, x_k \not\perp\!\!\!\perp x_i$. Note that as x_i is a root, there can only exist frontdoor paths between x_i and other vertices, so the dependence relations imply that $x_j, x_k \in \text{De}(x_i)$. This means that x_i is a confounder of $x_j, x_k \implies x_j \not\perp\!\!\!\perp x_k$. Therefore, x_i cannot induce a VP between x_j, x_k. □

D.2 PROOF OF LEMMA 3.2

Lemma 3.2 (Root with No SRDs). *For any $x_i \in V'$, iff $x_i \perp\!\!\!\perp x_j \; \forall x_j \in V' \setminus \{x_i\}$, x_i is a root with no SRDs.*

Proof. Note that V' is the union of vertices that are either SRDs or root vertices. We prove this by contradiction. Suppose that x_i is an SRD: then $\exists x_k \in V'$ such that x_k is a root and $x_i \in \text{De}(x_k)$ which implies $x_i \not\perp\!\!\!\perp x_k$. However, this contradicts our assumption that $x_i \perp\!\!\!\perp x_j, \forall x_j \in V'$. Now we show that x_i cannot a root vertex with SRDs. Suppose that x_i is a root vertex with SRDs: then $\exists x_k \in \text{De}(x_i) \implies x_i \not\perp\!\!\!\perp x_k$. However, this contradicts our assumption that $x_i \perp\!\!\!\perp x_j, \forall x_j \in V'$. Additionally, note that a root vertex x_i with no SRDs satisfies $x_i \perp\!\!\!\perp x_j, \forall x_j \in V'$. Therefore, x_i must be a root vertex with no SRDs.

Suppose x_i is a root with no SRDs. Note that V' contains only SRDs and roots. Then, as x_i has no SRDs, it has no descendants. As x_i is a root, it has no ancestors; therefore, $x_i \perp\!\!\!\perp x_j \; \forall x_j \in V' \setminus \{x_i\}$. □

D.3 PROOF OF LEMMA 3.3

Lemma 3.3 (Root ID). *Let W be a subset of V'' such that $\forall x_i \in W$, 1) $\exists x_j \in V$ such that x_i is identified as $\in An(x_j)$, and 2) $\nexists x_k \in V$ such that x_k is identified as $\in An(x_i)$. Then W contains all roots in V''.*

For $x_i, x_j, x_k \in V$, let r_{ij}^k be the residual of x_k nonparametrically regressed onto both x_i and x_j. Then, vertex $x_i \in W$ is a root vertex if and only if for every other vertex $x_j \in W$ such that $x_i \not\perp\!\!\!\perp x_j$, $\forall x_k$ such that x_j is identified $\in An(x_k)$, we have $x_j \perp\!\!\!\perp r_{ij}^k$.

Proof. In this proof, we say that x_i is in 'AD relation' to x_k iff x_i is identified as $\in An(x_k)$ by the Regression-Identification Test (Definition 3.4)

By definition, any root vertex $x_i \in V''$ has at least one SRD x_j, such that $x_j \in \text{Ch}(x_i)$. Note that as $x_j = f_{ij}(x_i) + \varepsilon_j$, x_i is in AD relation to x_j. Note that as x_i is a root vertex, all $x_j \in V''$ are not ancestors of x_i. If $x_j \perp\!\!\!\perp x_i$, then x_j will not be in AD relation to x_i as $x_i \perp\!\!\!\perp r_{ij}$, violating condition 2) in the AD definition. If $x_j \not\perp\!\!\!\perp x_i$, then $x_j \notin \text{An}(x_i)$, and therefore by assumption of the restricted ANM C.2 we have that $x_j \not\perp\!\!\!\perp r_{ji}$, implying that x_j is not in AD relation to x_i. This implies that all root vertices in V'' satisfy condition 1) and 2) of the root superset W, implying that all root vertices in V'' are contained in W.

Let $x_i \in W$ be a root. Consider x_j such that $x_j \not\perp\!\!\!\perp x_i$. Consider any x_k such that x_j is in AD relation to x_k. Note that $x_j \not\perp\!\!\!\perp x_i$ and x_i being a root implies that x_i is not a descendant of either x_j or x_k. Additionally, as x_j is in AD relation to x_k, x_i cannot be a confounder of x_j, x_k. This implies that $x_i \perp\!\!\!\perp x_k | x_j$, which implies that $r_{ij}^k \perp\!\!\!\perp x_i$.

Let $x_i \in W$ be a nonroot. Then, $\exists x_j \not\perp\!\!\!\perp x_i$ such that x_j is the root ancestor of x_i. Note that x_i is in AD relation to all of its children, $\text{Ch}(x_i)$. Note that $\exists x_k \in \text{Ch}(x_i)$ such that x_k is an ancestor of x_i. As x_i is a descendant of x_k, this implies that $x_i \not\perp\!\!\!\perp r_{ij}^k$.

More intuitively: if x_j is identified as $\in \text{An}(x_k)$, implying that x_j is independent of residual of x_k regressed onto x_j, x_j should remain independent of residual produced by the bivariate regression of x_k onto x_i, x_j. In contrast, for any non-root SRD x_p included in W and identified as in $\text{An}(x_h)$, there exists a root ancestor $x_l \in W$; therefore, if x_h is regressed onto x_l, x_p the resulting residual will be dependent on x_p.

□

D.4 PROOF OF LEMMA 4.1

Proposition D.1. *Given the vertices of a DAG G generated by an ANM, infinite data, a consistent nonparametric regression method, and a perfect independence test, Algorithm 1 returns the correct set of root vertices.*

Proof. Suppose $x_i \in U$ is a ND. Then, by definition $\exists x_j \in U$ such that x_j is a nonlinear function of its parents in U ($\text{Pa}(x_j) \cap U$), and x_j is an ancestor of ND ($x_j \in \text{An}(x_i)$). Note that this implies that the regression of x_i onto sorted vertices π leads to omitted variable bias [Pearl et al., 2016], which leads to e_i being dependent on any sorted vertex $x_k \in \pi$ such that x_k is an ancestor of x_j ($x_k \in \text{An}(x_j)$). Therefore, e_i is dependent on at least one sorted vertex in π.

Suppose e_i is dependent on at least on sorted vertex in π. Note that by assumption there are only nonlinear functions, x_i is not an LD. Suppose for contradiction that x_i is a VLC. Then, $x_i = f_i(\text{Pa}(x_i)) + \varepsilon_i$. Note that as π is a valid topological sort, it follows that $\text{Pa}(x_i) \subseteq \pi$ and $\text{De}(x_i) \cap \pi = \emptyset$. Therefore, $e_i = \varepsilon_i$, which implies $e_i \perp\!\!\!\perp x_j, \forall x_j \in \pi$. This implies that e_i is independent of all vertices in π, contradicting our above assumption.

□

D.5 PROOF OF LEMMA 4.2

Lemma 4.2 (LD independence)**.** *If $x_i \in U$ is an LD, $e_i \in U_E$ is independent of all sorted vertices in π.*

Proof. Let $x_i \in U$ be a LD. This implies that we can decompose x_i into a potentially nonlinear function of its parents in π and its the error terms of its ancestors in U:

$$x_i = f_i(\text{Pa}(x_i) \cap \pi) + \sum_{x_j \in \text{An}(x_i) \cap U} \alpha_{ij}\varepsilon_j + \varepsilon_i. \tag{4}$$

This implies that the residual of x_i nonparametrically regressed onto π equals

$$e_i = \sum_{x_j \in \text{An}(x_i) \cap U} \alpha_{ij}\varepsilon_j + \varepsilon_i. \tag{5}$$

As ε are mutually marginally independent, we have that $e_i \perp\!\!\!\perp x_j, \forall x_j \in \pi$.

□

D.6 PROOF OF LEMMA 4.3

Lemma 4.3. *For $e_i, e_j \in U_E$, let q_{ij} be the residual of e_j linearly regressed onto e_i, and q_{ji} be the residual of e_i linearly regressed onto e_j. Let Q be the set of all $x_i \in U$ such that there exists $x_j \in U$ such that the following conditions hold: 1) $e_j \perp\!\!\!\perp q_{ji}$, 2) $e_i \not\perp\!\!\!\perp q_{ij}$.*

Then $Q \subseteq U$ contains all LDs $\in U$ and no VLCs.

Proof. Suppose $x_i \in U$ is an LD. Let $x_j \in U$ be an ancestor of x_i such that x_j has no parents in U, i.e. $\text{Pa}(x_j) \cap U = \emptyset$. Such a vertex must always exist, as other x_i would be a VLC. We can decompose x_j into a function of parents in π, and an independent error term:

$$x_j = f_j(\text{Pa}(x_j) \cap \pi) + \varepsilon_j. \tag{6}$$

We can decompose x_i into a function of parents in π, a sum of linear functions of parents in U, and the independent error term ε_i:

$$x_i = f_i(\text{Pa}(x_i) \cap \pi) + \sum_{x_j \in \text{Pa}(x_i) \cap U} \alpha_{ij}x_j + \varepsilon_i. \tag{7}$$

Note that as x_i is a LD, all of its ancestors in U are linear functions of their parents in U. Therefore, we can further decompose the sum of ancestors in U as a linear sum of error terms:

$$x_i = f_i(\text{An}(x_i) \cap S) + \sum_{x_j \in \text{An}(x_i) \cap U} \beta_{ij}\varepsilon_j + \varepsilon_i. \tag{8}$$

Now, we consider the residuals $e_i, e_j \in U_E$, which we can write as:

$$e_j = \varepsilon_j \tag{9}$$

and

$$e_i = \sum_{x_j \in \text{An}(x_i) \cap U} \beta_{ij}\varepsilon_j + \varepsilon_i. \tag{10}$$

As e_i is a linear function of e_j, we have that $e_j \perp\!\!\!\perp q_{ji}, e_i \not\perp\!\!\!\perp q_{ij}$, implying that $x_i \in Q$. Therefore, Q contains all LDs.

Let $x_i \in U$ be a VLC. Then, we can write x_i as the sum of a function of parents (which are all contained in π) and independent error term:

$$x_i = f_i(\text{Pa}(x_i)) + \varepsilon_i. \tag{11}$$

Note that $e_i \in U_E$ is then just the independent error term:

$$e_i = \varepsilon_i. \tag{12}$$

Note that e_i has no parents in U_E: therefore, for any vertex x_j with associated residual $e_j \in U_E$, either $e_i \perp\!\!\!\perp e_j \implies e_j \perp\!\!\!\perp q_{ji}, e_i \perp\!\!\!\perp q_{ij}$, or $e_i \not\perp\!\!\!\perp e_j \implies e_j \not\perp\!\!\!\perp q_{ji}, e_i \not\perp\!\!\!\perp q_{ij}$. The last statement follows from the fact that if $e_i \not\perp\!\!\!\perp e_j$, as e_i is an independent error term, e_j must be a function of e_i; then, the conclusion follows by assumption of ANM C.2. Therefore, $x_i \notin Q$, and therefore no VLCs are included in Q. □

D.7 PROOF OF LEMMA 4.4

Lemma 4.4 (Consistency)**.** *Given a consistent estimator $\widehat{MI}(\cdot,\cdot)$, a fixed $x_i \in U'$ and corresponding residual e_i, $t^*(e_i, \pi)$ asymptotically approaches 0 as $n \to \infty$ iff e_i is independent of all vertices in π, i.e., x_i is a VLC.*

Proof. Note, U' contains only ND and VLC vertices.

Suppose test stastistic $t^*(e_i, S)$ asymptotically aproaches 0 as $n \to \infty$. Suppose for contradiction that e_i is not independent of $x_j \in \pi$. Note that this implies that the mutual information $\hat{M}I(x_j, e_i) \neq 0$, which contradicts our assumption that $t^*(e_i, S) \to 0$.

Suppose e_i is independent of all vertices in S. This implies that, for each $x_j \in \pi$ the mutual information approaches 0: $\hat{M}I(x_j, e_i) \to 0$. This implies that the sum also approaches 0: $t^*(e_i, S) = \sum_{x_j \in \pi} \hat{M}I(x_j, e_i) \to 0$.

□

E PROPOSITION PROOFS

E.1 PROOF OF PROPOSITION 3.5

Proposition E.1. *Given the vertices of a DAG G generated by an ANM, infinite data, a consistent nonparametric regression method, and a perfect independence test, Algorithm 1 returns the correct set of root vertices.*

Proof. By Definition 3.1, Definition 3.2, Definition 3.3, and Lemma 3.1, Lemma 3.2, we have Stage 1 of Algorithm 1, correctly identifies all VPs in graph G, eliminates all MRDs to obtain V', then identifies all roots with no SRDs to obtain V'', a set that contains only SRDs and roots. By the Regression-Identification test (Definition 3.4), a superset of roots $W \subseteq V''$ is identified, and by Lemma 3.3 Stage 2 identifies all roots $\in V''$ by pruning nonroots from W. Therefore, Root ID (Algorithm 1) correctly returns all roots in G. □

E.2 PROOF OF PROPOSITION 4.4

Proposition E.2 (Correctness of Sort Finder)**.** *Given the vertices of a DAG G generated by an ANM, the roots in G, infinite data, a consistent nonparametric regression method, and a perfect independence test, Algorithm 2 returns a valid sort π.*

Proof. Note that the roots of graph G are provided as input to Sort ID. Therefore, π is initialized with the roots.

We now induct on the length of π to show that Sort ID recovers a correct topological sort of vertices in G.

Base Iterations $(1, 2)$

1. Suppose there are m roots identified by Root ID. Then, m out of d vertices have been correctly sorted into π.
2. Note that in Stage 1, Sort Finder uses Lemma to prune the unsorted vertices U to obtain U', which contains only NDs and VLCs. Then, by Lemma 4.4, a VLC x_* is selected using the test statistic t^*. Therefore, when x_* is added to π, π is still a correct topological sort.

Iteration $k-1$, Inductive Assumption We have correctly sorted $k-1$ vertices of G into π.

1. Note that in Stage 1, Sort Finder uses Lemma to prune the unsorted vertices U to obtain U', which contains only NDs and VLCs. Then, by Lemma 4.4, a VLC x_* is selected using the test statistic t^*. Therefore, when x_* is added to π, π is still a correct topological sort.

Iteration $k-1$ Inductive Assumption is satisfied for iteration k, therefore we recover a valid topological sort π from $|\pi| = m$ to k. Thus, for a DAG with d vertices, Sort ID correctly recovers the full topological sort when provided the roots. □

F THEOREM PROOFS

F.1 PROOF OF THEOREM 3.6

Theorem 3.6. *Given a DAG $G = (V, E)$ under an ANM, let $M \subseteq V$ and $R \subseteq V$ be the sets of MRDs and roots in G, respectively. Let $c_{\max}^{\text{Alg1}}$ be the max covariate set size in nonparametric regressions in Algorithm 1 required to recover roots, and similarly $c_{\max}^{\text{DirectLiNGAM}}$, $c_{\max}^{\text{NHTS}}$, $c_{\max}^{\text{NoGAM}}$, $c_{\max}^{\text{RESIT}}$. If the ANM is linear, then $c_{\max}^{\text{Alg1}} = c_{\max}^{\text{DirectLiNGAM}} = 1$. If the ANM is nonlinear, Then, $c_{\max}^{\text{Alg1}} = 2$, $c_{\max}^{\text{NHTS}} = \max_{x_i \in M}(An(x_i) \cap R, 0)$, and $c_{\max}^{\text{RESIT}} = c_{\max}^{\text{NoGAM}} = d - 2$.*

This implies that $c_{\max}^{\text{Alg1}} < c_{\max}^{\text{RESIT}} = c_{\max}^{\text{NoGAM}}$, and when $M \neq \emptyset$, $c_{\max}^{\text{Alg1}} \leq c_{\max}^{\text{NHTS}}$.

Proof. Suppose the ANM is linear. Then, in Stage 2 of Root ID, after running pairwise univariate regression, roots are found to be in $An()$ relation to all vertices they are dependent on, and are immediately identified. No bivariate regressions take place. For DirectLiNGAM, only univariate regressions are used at any stage in the method, so the maximum conditioning set size is 1.

In Root ID, nonparametric regression is used in Stage 2 to identify the root superset W. These regressions are either pairwise or bivariate; thus, the upper bound on the covariate set size is 2.

Suppose the ANM is nonlinear. Then in Stage 2 of Root ID, bivariate regressions may be used, rendering the max covariate set size 2. In Stage 2 of the root-finding procedure of NHTS, regressions are run where x_j is regressed on x_i and all $x_k \in P_{ij} = \{x_k \in V : x_k \perp\!\!\!\perp x_i, x_k \not\perp\!\!\!\perp x_j\}$. We note that for x_j that is an MRD, P_{ij} is nonempty when x_i is a root ancestor of x_j: in particular, P_{ij} contains all other root ancestors of x_j. Therefore, NHTS requires multivariate regression whenever the graph contains MRDs; additionally, the size the largest multivariate regression is P_{ij}, which is bounded below by $\max_{x_i \in M}(\text{An}(x_i) \cap R, 0)$.

The results for RESIT and NoGAM follow from Theorem 4.7.

□

F.2 PROOF OF THEOREM 4.5

Theorem 4.5 (LoSAM Correctness)**.** *Given the vertices of a DAG G generated by an ANM, infinite data, a consistent nonparametric regression method, and a perfect independence test, Algorithm 3 returns a valid topological sort π.*

Proof. The correctness of Algorithm 1 follows from Proposition 3.5, so the correct set of roots is returned. Then, it follows from Proposition 4.4 that when Algorithm 2 is given the roots it correctly returns a valid topological ordering π. □

F.3 PROOF OF THEOREM 4.6

Theorem 4.6 (LoSAM Runtime)**.** *Given n samples generated from a d-dimensional DAG G under an ANM, the worst-case time complexity of Algorithm 3 is $O(d^3n^2)$.*

Proof. We first find the runtime of Algorithm 1, then find the runtime of Algorithm 2.
Part 1: Root Finder
In Stage 1, Root ID first runs at most $O(d^2)$ marginal independence tests that each have $O(n^2)$ complexity. Then, to identify all VPs and prune MRDs, every triplet of vertices is checked, which has worst case $O(d^3)$ complexity. In Stage 2, at most, $O(d^2)$ pairwise nonparametric regressions are run, each with $O(dn\log(n))$ complexity. Therefore, Root ID has $O(d^3n\log(n) + d^2n^2)$ time complexity.
Part 2: Sort Finder
In the worst case of a fully connected graph, there are $O(d)$ iterations of the Sort ID algorithm. In each iteration LoSAM runs at most $O(d^2)$ nonparametric mutual information tests, each of which has $O(n^2)$ time complexity. Therefore, Sort ID has worst case $O(d^3n^2)$ time complexity.
Overall Time Complexity
LoSAM (Algorithm 3) has an overall time complexity of $O(d^3n^2)$, due mainly to the time complexity of Sort ID. □

F.4 PROOF OF THEOREM 4.7

Theorem 4.7 (LoSAM Efficiency)**.** *Consider a fully connected DAG $G = (V, E)$ with ANM. Let $d := |V|$. Let n_k^{LoSAM} be the number of multivariate nonparametric regressions with covariate set size $k \in [d-2]$ run by LoSAM when sorting V; we similarly define n_k^{NHTS}, n_k^{RESIT} and n_k^{NoGAM} respectively. Then, $n_k^{\text{LoSAM}} = n_k^{\text{NHTS}} = d - k$, and $n_k^{\text{RESIT}} = n_k^{\text{NoGAM}} = k + 1$. This implies that for all $k > \frac{d}{2}$, $n_k^{\text{LoSAM}} = n_k^{\text{NHTS}} < n_k^{\text{RESIT}} = n_k^{\text{NoGAM}}$.*

Proof. Results regarding NHTS, RESIT and NoGAM follow from Theorem 4.7 in Hiremath et al. [2024]. In the case of a fully directed graph, the Root Finder stage in LoSAM runs no multivariate regressions, as only the true root vertex is identified as a potential root. In the Sort Finder stage, LoSAM regresses each unsorted vertex onto all sorted vertices, finding vertices with independent residuals. Therefore, the number of regressions run is equal to d minus the size of the covruate set. THerefore, when the covariate set is $k > \frac{d}{2}$, there are $d - k$ regressions run. □

G EXPERIMENTS

G.1 SYNTHETIC DATA GENERATION

Details on the synthetic DGP:

- We first generate a d-dimensional DAG according to a Erdos-Renyi model, with the average number of edges is either d or $2d$.
- Each variable x_i in the DAG is generated according to an ANM, i.e. $x_i = f_i(Pa(x_i)) + \varepsilon_i$. Each mechanism f_i is picked independently. Function f_i is linear with probability p and nonlinear with probability $1 - p$ ($p = 0, 0.25, 0.5, 0.75, 1$).
- If f_i is linear, then the coefficients of each variable in $Pa(x_i)$ are drawn from Uniform$[-1.5, -0.5]$ with probability $1/2$ or drawn from Uniform$[0.5, 1.5]$ with probability $1/2$.
- If f_i is nonlinear, we follow recent related literature [Lippe et al., 2022, Ke et al., 2023] and parameterize f_i as a single-hidden-layer feedforward neural network with $\tanh$ activation. Specifically, for a variable x_i with d parents, the neural network takes the parent data $x_{\text{Pa}(i)} \in \mathbb{R}^d$ as input, applies a linear transformation with weights sampled from Uniform$[-5, 5]$ followed by a $\tanh$ nonlinearity, and outputs a scalar value using a second linear transformation with weights also sampled from Uniform$[-5, 5]$. The number of hidden units is fixed to 10. The neural network weights are initialized independently for each variable.
- Each ε_i is independently drawn from either a Uniform, Laplacian, or Gaussian distribution with mean zero and variance $1/12$.

- We then process the data to ensure it is challenging enough (see Appendix G.2 for more details). We first standardize the data to mean 0 and variance 1. We then check to see if the R^2-sortability is not too high - if the R^2-sortability is above 0.75, we resample the data.

G.2 DATA PROCESSING

Synthetic Data Recent work have pointed out that simulated ANM often have statistical artifacts missing from real-world data [Reisach et al., 2021, 2023], leaving their real-world applicability open to question; for example, Reisach et al. [2021] develop Var-Sort, which sorts variables according to increasing variance and is performant on synthetic data that is not standardized. Additionally, Reisach et al. [2023] develop a scale-invariant heuristic method R^2-Sort by sorting variables according to increasing coefficient of determination R^2; they find that when synthetic data is naively generated it tends to have high R^2-sortability, while the prevalence of high R^2-sortability in real data is unknown.

To alleviate concern that our data is gameable by Var-Sort, we standardize all data to zero mean and unit variance. However, processing data to be challenging for R^2-sort is more difficult: Reisach et al. [2023] explain how the R^2 coefficients depend on the graph structure, noise variances, and edge weights of the underlying DAG in a complex manner, concluding that "one cannot isolate the effect of individual parameter choices on R^2-sortability, nor easily obtain the expected R^2-sortability when sampling ANMs given some parameter distributions, because the R^2 values are determined by a complex interplay between the different parameters." Therefore, we are unable to directly generate data with low R^2; instead, to generate each instance of data we randomly sample DAGs up to 100 times until a DAG with R^2-sortability less than 0.75 is generated. We were able to achieve lowered R^2-sortability in all experiments except linear ANMs on dense graphs. These processing steps prevent the heuristic algorithms Var-Sort [Reisach et al., 2021] and R^2-Sort [Reisach et al., 2023] from leveraging arbitrary features of simulated data to accurately recover a topological sort, and ensure that the data is sufficiently challenging for our discovery algorithm and the baselines.

Sachs Data To generate the results in Table 2, we ran the methods on 30 different random samples from the Sachs Dataset of size $n = 300$.

G.3 ATOP METRIC

Montagna et al. [2023c] introduce topological order divergence as a measure of the discrepancy between the estimated topological ordering π and the adjacency matrix of the true causal graph A, expressed as:

$$D_{top}(\pi, A) = \sum_{i=1}^{d} \sum_{j:\pi_i > \pi_j} A_{i,j}. \tag{13}$$

They note that "if π is a correct topological order for A, then $D_{top}(\pi, A) = 0$. Otherwise, $D_{top}(\pi, A)$ counts the number of edges that cannot be recovered due to the choice of topological order."

Hiremath et al. [2024] introduce a normalized version of D_{top}, A_{top}, expressed as:

$$A_{top} = \frac{D_{top}}{|A|}, \tag{14}$$

where $|A|$ is the number of edges in graph A. If π is a correct topological order for A, then $A_{top}(\pi, A) = 1$. Otherwise, $A_{top}(\pi, A)$ equals the percentage of edges that cannot be recovered due to the choice of topological order.

G.4 DETAILS

Implementation details of LoSAM and all considered baselines.

- For regression tasks, LoSAM uses a RandomForestRegressor from the `sklearn.ensemble` package with the following settings: n_estimators = 100, max_depth = 10, min_sample_split = 10, min_sample_leaf = 5, max_features = "sqrt." LoSAM uses the kernel independence test (KCI) developed in [Zhang et al., 2011], with the estimator KCI from the `causal-learn` package for independence tests, and the mutual information test developed in [Kraskov et al.,

2004], with the estimator from the `npeet` package (both with default settings). The cutoff for all independence tests is 0.01.

- All baselines were used with default hyperparameters encoded by the packages they were imported from. CaPS and NHTS were imported from the github repositories associated with those papers (https://github.com/E2real/CaPS/tree/main, https://github.com/Sujai1/hybrid-discovery). DirectLiNGAM and RESIT were imported from the `lingam` package. SCORE, NoGAM, and CAM were imported from the `DoDiscover` package. VarSort and Randsort were implemented by hand. The R^2-sortability metric was imported from the `CausalDisco` package. Please see the supplemental code to find the exact code used to run each baseline.

All experiments were conducted in python; experiments involving CaPS were conducted on an Apple M2 Pro Chip, 16 Gb of RAM, with no parallelization, while experiments with all other methods were conducted on a r6a.metal EC2 instance with 192vCPUs, 1536 GiB memory.

G.5 ADDITIONAL EXPERIMENTAL RESULTS

G.5.1 DENSE GRAPHS

Experimental results for dense graphs (ER2).

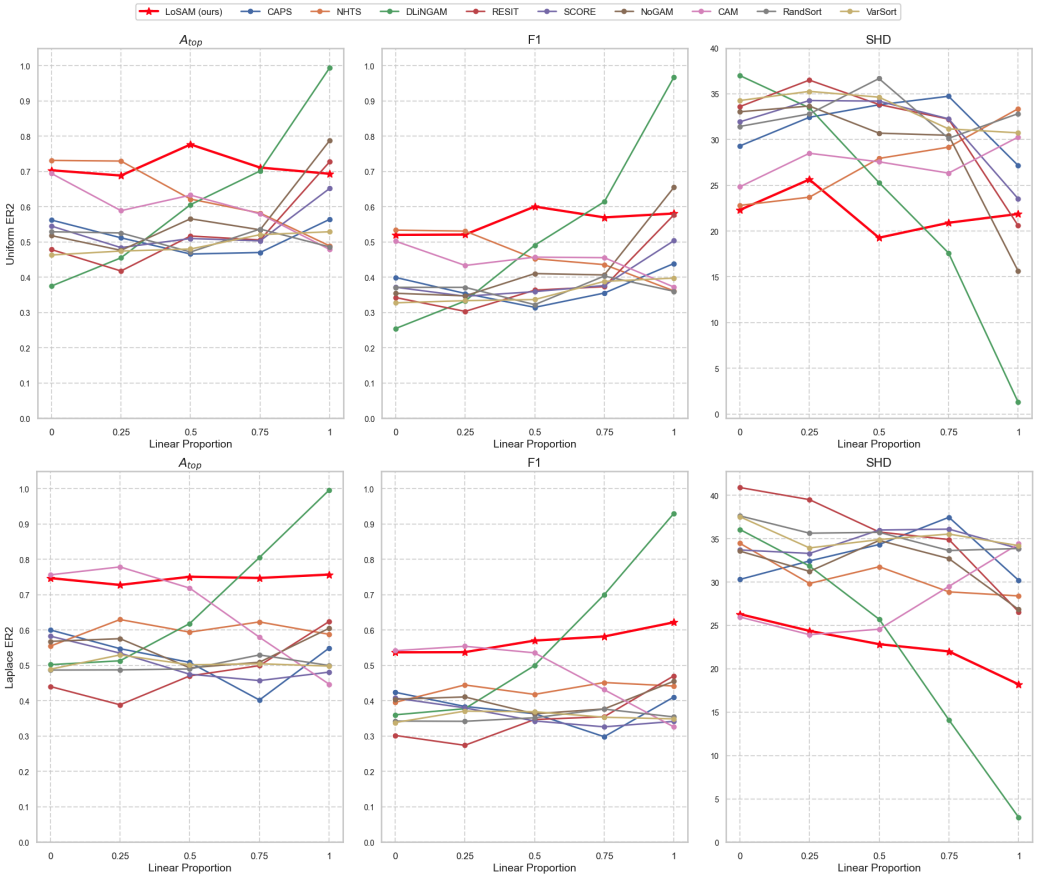

Figure 6: Performance of LoSAM across diff. proportion of linear mechanism on dense ER2 data; uniform (top) and laplacian noise (bottom).

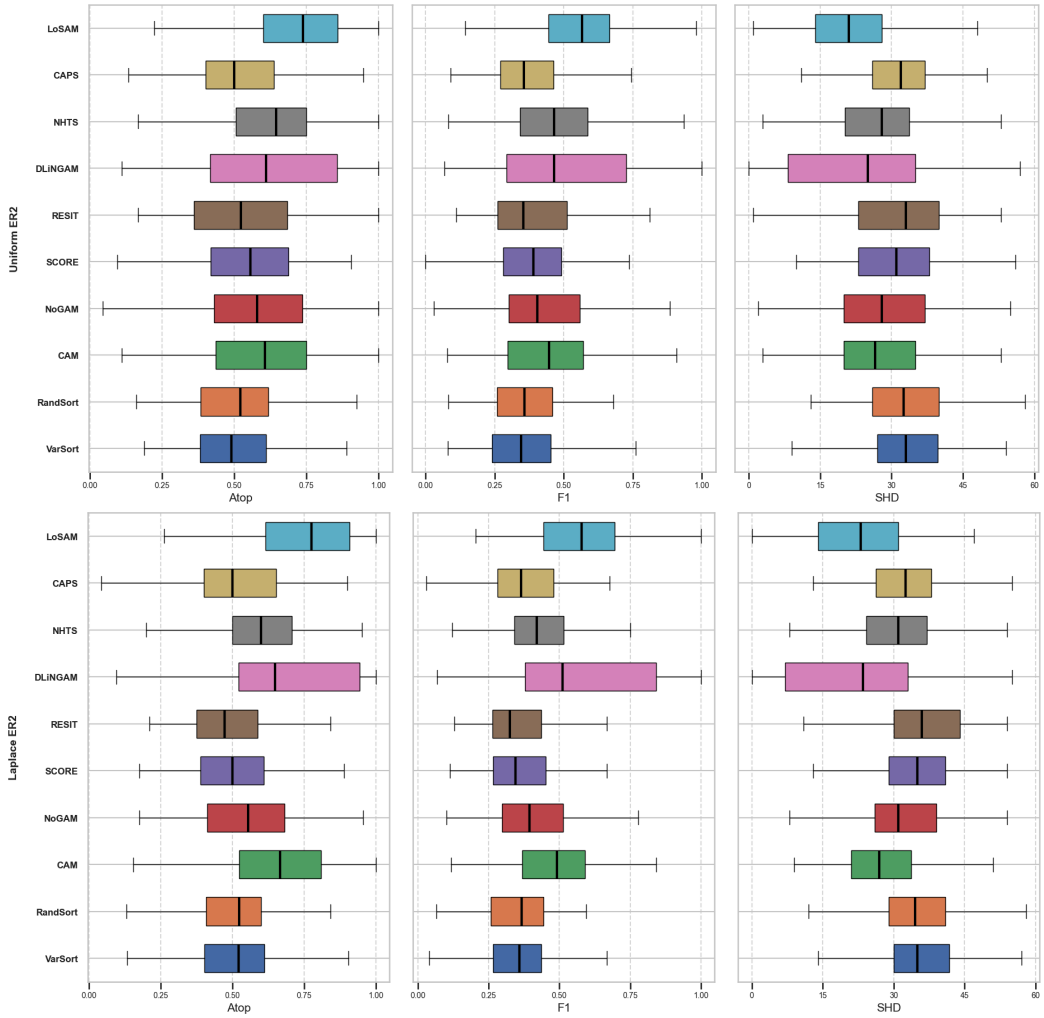

Figure 7: Performance of LoSAM on dense ER2 synthetic data with 50% proportion of linear mechanisms. Top row: uniform noise. Bottom row: Laplace noise.

G.5.2 HIGH-DIMENSIONAL GRAPHS

We evaluate how LoSAM's performance metrics evolve in high-dimensional settings (we omit other baselines due to computational cost). Results show LoSAM remains effective as the complexity of the data increases, though performance declines as d increases with fixed n – an expected consequence of fixed n in high dimensions. This degradation occurs because the same $n = 2000$ samples become increasingly sparse as d grows, which will reduce estimation accuracy for any method.

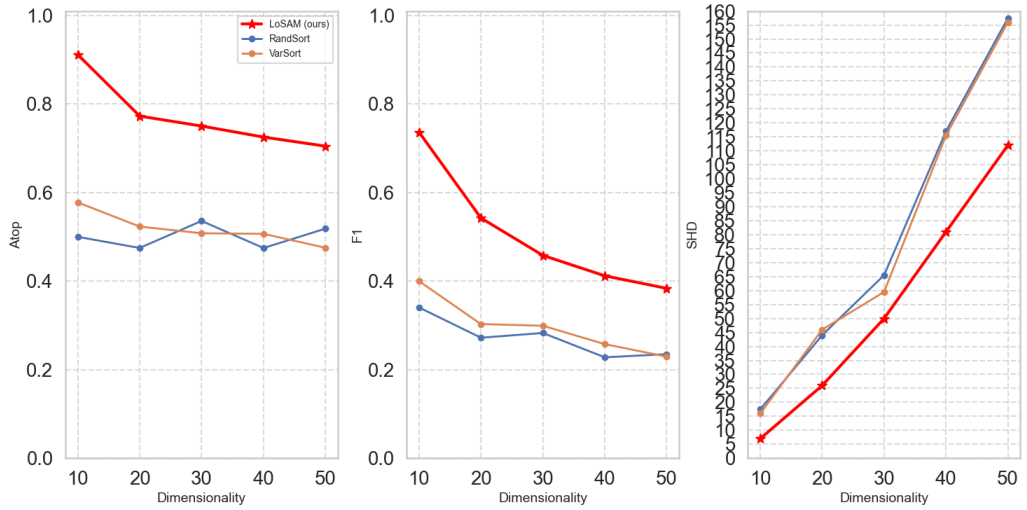

Figure 8: Performance of LoSAM on ER1 graph with uniform noise and linear proportion = 0.5, across diff. dimensionalities.

G.5.3 NONLINEAR GAUSSIAN

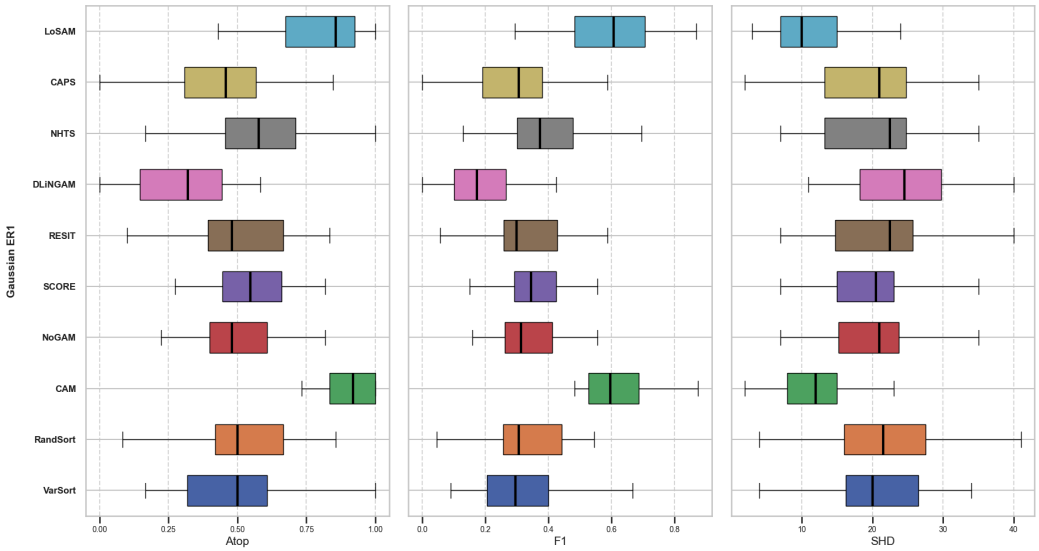

Figure 9: Performance of LoSAM across all DGMs; uniform noise (left) and laplacian noise (right).

Experimental results for nonlinear gaussian ANMs.

G.5.4 SENSITIVITY TO ESTIMATION ERROR

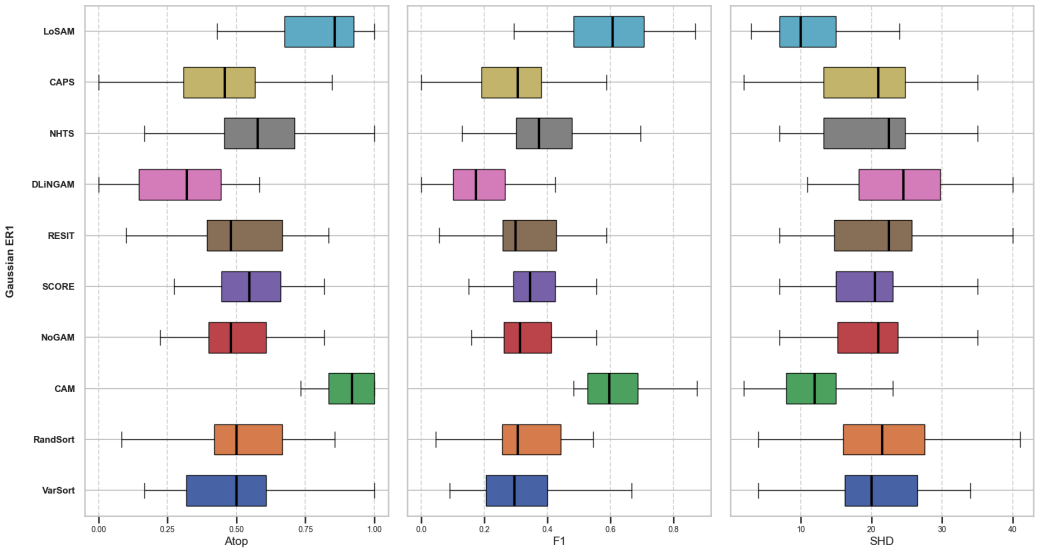

Figure 10: Performance of LoSAM on ER1 graph with uniform noise, $d = 10, n = 300$ across all DGMs, with different regression estimators.

We evaluate LoSAM variants with different regression estimators (all taken from the `Sklearn` package):

1. LoSAM: RandomForestRegressor.
2. LoSAM_KRR_Poly4: Kernel Ridge Regression (polynomial kernel, degree 4).
3. LoSAM_KRR_Poly8: Kernel Ridge Regression (polynomial kernel, degree 8).
4. LoSAM_KRR_RBF: Kernel Ridge Regression (RBF kernel).

Hyperparameters for each estimator:

1. LoSAM: n_estimators = 100, max_depth = 10, min_sample_split = 10, min_sample_leaf = 5, max_features = "sqrt".
2. LoSAM_KRR_Poly4: kernel = "poly", degree = 4, alpha =0.1, coef0=1.
3. LoSAM_KRR_Poly8: kernel = "poly", degree = 8, alpha =0.1, coef0=1.
4. LoSAM_KRR_RBF: kernel = "rbf", alpha =0.1, gamma = 0.01.

G.5.5 COMAPARISON TO OPTIMIZATION-BASED ANM METHOD

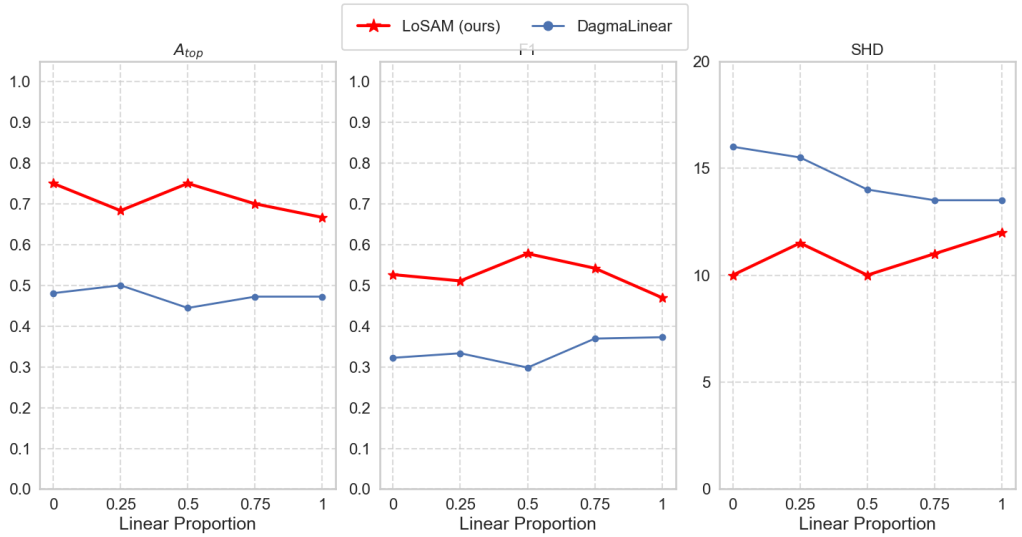

Figure 11: Performance of LoSAM and DagmaLinear on ER1 graph with uniform noise, $d = 10, n = 300$ across all DGMs.

We evaluate LoSAM against a linear optimization-based method (implementation from `DAGMA` package, hyperparameters taken from Appendix C.1.1. of Bello et al. 2022). Results show that LoSAM's performance is superior to this optimization-based ANM method.

G.6 RUNTIME RESULTS

G.6.1 SPARSE

The runtime results for Figure 3

	Linear Proportion				
Methods	**0**	**0.25**	**0.5**	**0.75**	**1**
LoSAM	8.495 ± 1.998	8.853 ± 2.022	9.606 ± 2.034	9.818 ± 2.214	8.634 ± 1.529
CAPS	8.753 ± 0.228	8.589 ± 0.295	8.849 ± 0.281	8.877 ± 0.486	9.273 ± 0.414
NHTS	38.962 ± 12.904	42.804 ± 15.097	45.760 ± 19.363	57.838 ± 12.157	39.070 ± 11.562
DLiNGAM	3.301 ± 0.983	3.251 ± 1.095	3.737 ± 1.160	4.004 ± 1.592	4.152 ± 1.432
RESIT	36.666 ± 1.848	35.099 ± 1.676	37.113 ± 2.322	35.920 ± 2.623	33.576 ± 2.350
SCORE	9.717 ± 8.122	7.766 ± 26.275	11.759 ± 16.300	13.039 ± 27.730	17.037 ± 33.070
NoGAM	14.205 ± 12.660	14.176 ± 9.061	21.131 ± 14.918	34.412 ± 35.567	42.827 ± 49.857
CAM	37.760 ± 5.892	39.825 ± 5.133	39.192 ± 6.005	43.425 ± 6.491	38.608 ± 9.230
RandSort	2.503 ± 0.812	2.556 ± 1.103	2.499 ± 0.896	3.150 ± 1.360	3.102 ± 1.373
VarSort	2.923 ± 0.806	2.528 ± 0.961	2.254 ± 0.695	2.455 ± 1.077	2.841 ± 1.116

Table 3: Runtime results for ER1 graphs and Uniform noise.

	Linear Proportion				
Methods	**0**	**0.25**	**0.5**	**0.75**	**1**
LoSAM	9.049 ± 1.802	9.564 ± 2.945	9.963 ± 1.965	9.708 ± 2.601	10.650 ± 2.108
CAPS	8.851 ± 0.275	8.915 ± 0.534	9.434 ± 0.363	9.522 ± 0.682	9.290 ± 0.408
NHTS	31.039 ± 6.179	29.860 ± 9.266	39.188 ± 12.381	38.780 ± 13.329	44.217 ± 16.933
DLiNGAM	2.910 ± 0.880	3.263 ± 1.177	3.320 ± 1.106	4.450 ± 1.212	4.673 ± 1.320
RESIT	36.100 ± 1.195	36.212 ± 2.022	34.985 ± 1.477	36.036 ± 2.705	36.697 ± 2.106
SCORE	9.117 ± 25.930	10.183 ± 19.615	8.483 ± 30.603	12.368 ± 18.493	34.701 ± 24.086
NoGAM	16.238 ± 24.006	9.448 ± 14.069	12.992 ± 27.836	13.347 ± 13.847	26.202 ± 19.485
CAM	41.502 ± 13.920	40.390 ± 9.508	38.531 ± 11.565	37.154 ± 4.340	39.573 ± 13.534
RandSort	2.632 ± 2.123	2.535 ± 1.557	2.306 ± 1.526	2.649 ± 0.796	2.757 ± 1.124
VarSort	2.777 ± 1.965	2.859 ± 1.317	2.225 ± 0.643	2.593 ± 0.689	2.532 ± 1.167

Table 4: Runtime results for ER1 graphs and Laplace noise.

G.6.2 DENSE

The runtime results for Figure 6.

	Linear Proportion				
Methods	**0**	**0.25**	**0.5**	**0.75**	**1**
LoSAM	12.203 ± 2.274	13.083 ± 8.218	12.951 ± 5.856	14.761 ± 4.445	14.646 ± 6.186
CAPS	8.345 ± 0.195	8.551 ± 0.256	8.714 ± 0.244	8.177 ± 0.145	8.196 ± 0.260
NHTS	68.842 ± 10.248	72.542 ± 8.703	88.527 ± 10.292	84.519 ± 10.007	67.167 ± 6.453
DLiNGAM	4.461 ± 1.220	4.080 ± 1.903	5.185 ± 1.172	5.599 ± 1.518	4.602 ± 1.371
RESIT	35.534 ± 3.293	35.021 ± 7.270	36.113 ± 4.412	36.214 ± 5.124	35.061 ± 5.673
SCORE	103.090 ± 66.855	138.517 ± 45.652	113.100 ± 64.391	115.136 ± 61.357	151.977 ± 15.391
NoGAM	83.755 ± 25.638	93.860 ± 18.106	57.771 ± 24.034	60.016 ± 35.062	108.342 ± 14.460
CAM	38.067 ± 37.486	36.290 ± 26.036	36.541 ± 32.798	37.179 ± 39.518	40.331 ± 6.242
RandSort	3.012 ± 1.442	2.831 ± 0.785	3.410 ± 3.181	2.618 ± 3.355	3.579 ± 0.945
VarSort	2.524 ± 1.168	2.644 ± 0.904	2.996 ± 3.431	2.495 ± 2.434	3.163 ± 1.066

Table 5: Results for ER1 graphs and Uniform noise.

	Linear Proportion				
Methods	**0**	**0.25**	**0.5**	**0.75**	**1**
LoSAM	19.513 ± 10.874	15.102 ± 7.256	17.479 ± 11.748	15.625 ± 10.218	17.261 ± 3.968
CAPS	8.554 ± 0.212	8.281 ± 0.263	8.137 ± 0.254	8.392 ± 0.224	8.296 ± 0.187
NHTS	41.759 ± 4.190	54.444 ± 11.174	67.633 ± 14.289	75.535 ± 11.840	94.058 ± 12.138
DLiNGAM	4.238 ± 1.499	4.866 ± 1.486	5.100 ± 1.208	4.828 ± 1.667	5.094 ± 1.194
RESIT	36.790 ± 2.213	36.638 ± 6.152	39.732 ± 2.838	36.152 ± 4.840	39.040 ± 4.453
SCORE	136.591 ± 66.444	142.725 ± 63.160	23.886 ± 42.393	41.719 ± 56.895	82.003 ± 67.079
NoGAM	85.983 ± 30.039	70.836 ± 28.574	31.275 ± 17.726	62.495 ± 32.673	68.000 ± 45.448
CAM	34.128 ± 38.388	37.716 ± 24.377	50.529 ± 30.624	48.106 ± 32.871	35.543 ± 22.036
RandSort	3.241 ± 2.999	3.687 ± 1.932	3.160 ± 3.058	2.868 ± 3.216	3.082 ± 1.783
VarSort	2.849 ± 2.216	3.393 ± 1.501	2.393 ± 3.510	2.910 ± 1.249	2.836 ± 1.105

Table 6: Runtime results for ER1 graphs and Laplace noise.

G.6.3 HIGH DIMENSIONAL

The runtime results for Figure 8.

	d **Value**			
Methods	**10**	**15**	**20**	**25**
LoSAM	9.606 ± 2.034	34.217 ± 12.832	70.256 ± 28.749	94.054 ± 25.865
DLiNGAM	3.737 ± 1.160	19.336 ± 4.231	48.557 ± 24.781	55.616 ± 7.221
RandSort	2.499 ± 0.896	18.547 ± 2.984	39.321 ± 11.026	53.576 ± 3.136
VarSort	2.254 ± 0.695	17.907 ± 4.758	32.088 ± 12.803	52.063 ± 7.968

Table 7: Runtime results for different d values.

G.6.4 NONLINEAR GAUSSIAN

The runtime results for Figure 9.

Methods	**0**
LoSAM	8.228 ± 2.200
CAPS	9.149 ± 0.337
NHTS	44.242 ± 13.680
DLiNGAM	3.526 ± 0.923
RESIT	35.377 ± 1.334
SCORE	16.902 ± 30.607
NoGAM	12.432 ± 12.605
CAM	39.349 ± 10.694
RandSort	3.330 ± 2.406
VarSort	3.085 ± 2.322

Table 8: Results for ER1 graphs and Gaussian noise with linear proportion 0.

H ALGORITHM WALKTHROUGH

In this section we walk-through LoSAM on two different examplary DAGs.

H.1 LOSAM

Example 1:

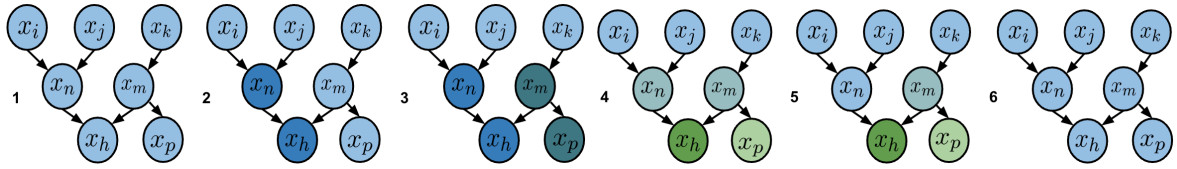

Figure 12: LoSAM walkthrough on example DAG.

Subfigure 1 of Figure 12 illustrates the example causal graph from which data is generated. Now, we walk through how LoSAM would obtain a topological sort of this DAG. In Stage 1 Root Finder, x_n, x_h are identified as MRDs and are pruned (subfigure 2). Then, x_i, x_j are recovered as roots, as they are independent of x_k, x_m, x_p. Then, in Stage 2, x_m, x_p are pruned as they are not identified as ancestors of any vertices, while x_k is (subfigure 3). We therefore recover x_i, x_j, x_k as roots. Subfigure 4 illustrates the decomposition of the graph into roots (x_i, x_j, x_k), VLCs (x_n, x_m), a ND x_h and an LD (x_p). In Stage 1 of Sort Finder, x_p is pruned from $U = \{x_n, x_m, x_p, x_h\}$ as it as an LD. Then, x_n is determined as a VLC, as a minimizer of the test statistic t^* (subfigure 5), and is sorted. This is again repeated 3 more times to sort all of U (subfigure 6); therefore, LoSAM correctly obtains a valid sort π.

Example 2:
Consider a DAG G with five vertices x_1, x_2, x_3, x_4, x_5, where $x_1 \to x_3, x_2 \to x_3, x_1 \to x_4, x_4 \to x_5$. Suppose the functional relationships are given by $x_1 = \varepsilon_1, x_2 = \varepsilon_2, x_3 = f_3(x_1, x_2) + \varepsilon_3, x_4 = f_4(x_1) + \varepsilon_4, x_5 = f_5(x_4) + \varepsilon_5$, where all f_i are nonlinear.

In stage 1 of the root-finding subroutine of LoSAM (Algorithm 1), we run pairwise independence tests between all vertices. We find that x_3 induces a VP between x_1, x_2; this means that x_3 is an MRD and so we prune it. We then note that x_2 is independent of all non-pruned vertices, and so we classify it as a root. In stage 2, we run pairwise nonparametric regression between the remaining vertices x_1, x_4, x_5. We note that regressing x_4 onto x_1 yields an independent residual, and regressing x_5 onto x_4 yields an independent residual. Therefore, by Definition 3.4, x_1 is identified $\in An(x_4)$, and x_4 is identified $\in An(x_5)$. As x_5 is not identified as the ancestor of any variable, it cannot be a root and is pruned. Note that, although x_4 is identified $\in An(x_5)$, it is identified as a descendant of x_1 – therefore, x_4 cannot be a root and is pruned. We are left with x_1 being the other root in this DAG.

I CLARIFYING EXPLANATIONS

I.1 NHTS FAILURE MODE

LoSAM shares some similiarities with NHTS, as they both utilize a local search approach and regression-based tests in their topological ordering procedure. However, LoSAM differs fundamentally from NHTS on a conceptual basis.

NHTS defines a set of local substructures (PP1, PP2, PP3, PP4 relations) that characterize the space of possible parent-child relationships. In particular it finds that, under nonlinear relationships, only the roots (+ a small class of nonroots) satisfy PP2 relations and can be identified through regression.To find the rest of the ordering, NHTS exploits the fact that, under nonlinear relationships, only vertices in the next unknown topological layer (which are only a children of the sorted vertices) will yield an independent residual after regression onto the sorted vertices.

In contrast, LoSAM defines a set of local causal substructures (SRDs, MRDs, VP, VLC, ND, LD) that are defined in terms

of ancestor-descendant relationships. This difference results in LoSAM being able to handle mixed mechanism ANM with fewer high dimensional regressions.

Identifiability Issues NHTS's reliance on local structures that are defined in terms of parent-child relations limits the method in terms of identifiability: as shown by Shimizu et al. 2011, linear mechanisms can mean that even ancestor-descendant relationships can yield independent residuals. We provide Example 1 below to demonstrate that the residual independence relations differ between the same ancestor-descendant pairs when the underlying functions are linear or nonlinear, and thus NHTS fails to accurately recover the correct topological sort. In contrast, it follows from Theorem 4.7 that LoSAM accurately recovers the topological sort in Example 1.

Sample Efficiency Issues NHTS's reliance on the PP2 framework limits the method in terms of sample efficiency - by leveraging parent-child relations, this requires that all parents are included as covariates in a regression to recover an independent residual. This can lead to high-dimensional regressions in the root finding stage. We provide Example 2 below to demonstrate that, even when the underlying mechanism is nonlinear, the PP2-framework approach of NHTS requires high-dimensional regressions to recover root vertics. In contrast, it follows from Theorem 3.6 that LoSAM recovers the roots with no high-dimensional regressions.

Example 1 Consider a DAG G with three vertices x_1, x_2, x_3, where $x_1 \to x_2, x_2 \to x_3$. Suppose NHTS has correctly identified the root vertex x_1.

Suppose the functional causal relationships are nonlinear, given by $x_1 = \varepsilon_1, x_2 = x_1^2 + \varepsilon_2, x_3 = x_1 x_2 + \varepsilon_3$. Then, when NHTS tries to determine the next variable in the sort, it will regress both x_3 and x_2 onto x_1, and find that x_1 is independent of the residual only in one regression; thus, NHTS can distinguish between the two and will accurately sort x_2 before x_3.

Suppose the functional causal relationships are linear, given by $x_1 = \varepsilon_1, x_2 = x_1 + \varepsilon_2, x_3 = x_2 + \varepsilon_3$, where the ε_is are mutually independent noise variables. Then, when NHTS tries to determine the next variable in the sort, it will regress both x_3 and x_2 onto x_1, and find that x_1 is independent of the residual in both regressions. Thus, NHTS cannot distinguish between the two, and will fail to accurately sort them. Residual independence from naively running regressions is insufficient to handle both linear and nonlinear mechanisms.

Example 2 Consider a DAG G with 100 vertices $x_1, x_2, \ldots, x_{99}, x_{100}$, where $x_1, \ldots, x_{99}$ are all roots, and are all parents of x_{100} ($x_1 \to x_{100}, x_2 \to x_{100}, \ldots, x_{99} \to x_{100}$).

Suppose the functional causal relationships are nonlinear, given by $x_1 = \varepsilon_1, x_2 = \varepsilon_2, \ldots, x_{99} = \varepsilon_{99}, x_{100} = \tanh(x_1 \times x_2 \ldots \times x_{99}) + \varepsilon_{100}$. Then, when NHTS tries to determine that x_{100} is in PP2 relation any of other vertices, to obtain an independent residual it will need to regress x_{100} onto *all* 99 other vertices! Therefore, the PP2 framework approach is incredibly sample inefficient, tending towards using high-dimensional regressions as the number of roots grows large.

I.2 COMPARISON OF V-PATTERNS AND V-STRUCTURES

We note that v-structures do not always correspond to the statistical constraints required to be a VP, in the sense that x_i, x_j may be conditionally independent, rather than marginally independent as required in our definition of VP. For example, in a DAG where $x_i \to x_j, x_i \to x_k, x_j \to x_p, x_k \to x_p, x_p \to x_m$, x_j, x_k, x_p form a v-structure but not a VP. However, v-structures are also not always more general than VPs, as in a DAG where $x_j \to x_p, x_k \to x_p, x_p \to x_m$, x_j, x_k, x_m form a VP but not a v-structure. Therefore, it is accurate to characterize the statistical constraints of v-structures and VPs as distinct but related concepts that overlap when either a v-structure has marginally independent parent vertices or when a triplet pair is a VP and has only parent-child relations.

J LIMITATIONS

We note that the standard approach in the causal discovery literature to assess a method's finite sample performance is to evaluate the method on synthetically generated data [Reisach et al., 2021]. This is because causal ground truth is incredibly rare, as it requires real-world experiments [Faller et al., 2024a]; these experiments can expensive, potentially unethical, but most importantly are often infeasible or ill defined [Spirtes and Scheines, 2004]. This lack of substantial real-world benchmark datasets is important because key assumptions that are used to generate synthetic data, which discovery methods rely on (such as additive noise, faithfulness, and causal sufficiency), may be violated in practice [Faller et al., 2024a]. Therefore we caution that experimental results on synthetic data should be interpreted as a demonstration of a method's theoretical performance on somewhat idealized data, which may not reflect measurements from real-world settings where

assumptions critical to the method are not met.

K ASSET INFORMATION

DirectLiNGAM and RESIT were imported from the `lingam` package. R^2-sort was imported from the `CausalDisco` package. NHTS and LoSAM were implemented using the kernel ridge regression function from the `Sklearn` package, used kernel-based independence tests from the `causal-learn` package, and a mutual information estimator from the `npeet` package. All assets used have a CC-BY 4.0 license.