SCALABLE POST-NONLINEAR CAUSAL DISCOVERY VIA LIKELIHOOD-BASED ORDER LEARNING

Anonymous authors
Paper under double-blind review

ABSTRACT

Post-nonlinear (PNL) structural causal models allow for flexible nonlinear mechanisms but remain difficult to learn in multivariate and high-dimensional settings. The few existing multivariate PNL methods primarily estimate a causal order through recursive sink identification and scale poorly. We present SCORPIO, a method for learning multivariate PNL DAGs under causal sufficiency. SCORPIO parameterizes each inverse outer transformation as a strictly monotone sigmoid normalizing flow, yielding a tractable node-wise likelihood that supports several noise distribution families. We estimate a topological order by greedily adding edges based on local likelihood improvements, which requires updating only the affected node model, avoiding retraining of the full system model. Candidate edges are subsequently pruned via marginal independence tests between scalar residuals. Across synthetic experiments varying the noise distribution, parental-mechanism complexity, graph size, and graph density, SCORPIO achieves the lowest order divergence and structural Hamming distance in most settings. Under a 24-hour time limit, SCORPIO is the only evaluated multivariate PNL method that completed experiments beyond 100 variables, with results reported for up to 500 variables. It also achieves the lowest order divergence and competitive DAG-recovery performance on a semi-synthetic manufacturing benchmark and the Sachs protein dataset.

1 INTRODUCTION

Causal knowledge is crucial for reasoning about interventions, which enables system optimization and informed decision-making. Inferring causal relationships from data, called causal structure learning, is hence a desired task across domains, like biology, sociology and manufacturing. Under causal sufficiency and acyclicity, this task is particularly challenging for high-dimensional data with nonlinear relationships and unknown noise distributions. While popular causal discovery methods such as PC (Spirtes & Glymour, 1991), GES (Chickering, 2002) and their extensions (Spirtes et al., 2000; Colombo et al., 2012; Colombo & Maathuis, 2014; Ramsey et al., 2016; Ejaz & Bareinboim, 2026) have shown strong performance on large problems, they generally identify only the Markov equivalence class of the system. Functional assumptions can enable identification of the ground-truth DAG. Two prominent identifiable cases are linear relationships with non-Gaussian noise (Shimizu et al., 2006) and nonlinear relationships with additive noise (Hoyer et al., 2008; Peters et al., 2014), which have been studied extensively (Hyvärinen et al., 2010; Hyvärinen & Smith, 2013; Shimizu, 2014; Maeda & Shimizu, 2020; Leyder et al., 2025; Janzing et al., 2009; Peters et al., 2011; Zhu et al., 2020; Lachapelle et al., 2020; Ng et al., 2022; Petkov et al., 2025; Wienöbst et al., 2026).

The post-nonlinear (PNL) model (Zhang & Hyvärinen, 2010; 2009) generalizes these models by applying an invertible nonlinear outer transformation to the sum of a nonlinear parental mechanism and noise. While its generality is a natural advantage compared to the previous models for representing complex data, learning the outer transformation poses an additional difficulty. Consequently, relatively few methods have been developed for the PNL framework, and even fewer address the multivariate setting.

Existing multivariate PNL algorithms have at least one of the following limitations: no architectural guarantee of invertibility, restrictive noise assumptions, estimation of the system order only, or poor scaling with system size. Furthermore, their order-estimation procedures share the iterative sink-

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

identification paradigm of RESIT (Peters et al., 2014). At each iteration, this approach fits every remaining variable against all other remaining variables, which can limit scalability and is inefficient in terms of parameter training.

We address these limitations with SCORPIO (Scalable Causal Ordering and Residual Pruning with Invertible Outer flows), a method for learning PNL DAGs from causally sufficient multivariate systems. Its central component is a greedy order-learning procedure that scores local parent additions by updating only the affected node model and measuring improvements in the PNL likelihood. Our main contributions are as follows:

Contributions.

- We introduce an empirically motivated greedy edge-ranking procedure for learning a topological order in multivariate PNL models. The procedure scores candidate edges by their local improvement in the PNL likelihood and provides an alternative to iterative sink identification.
- To obtain tractable PNL likelihoods, we parameterize the inverse outer transformation using a monotone sigmoid normalizing flow and accommodate several noise families in the likelihood. We combine the learned order with a residual-based pruning procedure that uses marginal independence tests between scalar residuals to obtain a sparse DAG.
- We evaluate SCORPIO in controlled synthetic studies of noise distribution, parental-mechanism complexity, graph size, and edge density, as well as on semi-synthetic and real-world data. Based on mean point estimates, SCORPIO achieves the lowest order divergence in 18 of 21 settings, the lowest SHD in 19 of 21 settings, and the lowest SID in 20 of 21 settings. It is also the only evaluated multivariate PNL method that completes experiments beyond 100 variables within the stated time limit and CPU allocations, with results reported for up to 500 variables.

2 Related Work

Methods explicitly targeting the PNL model comprise the bivariate approaches of Zhang & Hyvärinen (2010; 2009); Zhang et al. (2015); Uemura & Shimizu (2020); Tu et al. (2022) and the multivariate methods AbPNL (Uemura et al., 2022), RankPNL (Keropyan et al., 2023), and CAF-PoNo (Hoang et al., 2024). While Zhang & Hyvärinen (2009), Zhang et al. (2015), and Tu et al. (2022) suggest extensions to the multivariate setting, they do not provide empirical evaluations on multivariate systems or publicly available implementations. We therefore exclude these methods from the following study.

The three multivariate PNL methods AbPNL, RankPNL and CAF-PoNo differ in modeling approach and invertibility enforcement, but share the general structure of order estimation through recursive sink identification based on RESIT (Peters et al., 2014), followed by pruning of the corresponding complete DAG. An exception is RankPNL, which only estimates an order. RankPNL exploits the rank-preserving property of the outer transformation to separate estimation of the parental mechanism from independence testing. It provides RankG for Gaussian noise and RankS for general noise and establishes consistency under linear parental mechanisms. AbPNL expresses invertibility of the outer transformation through an autoencoding loss, which does not provide an architectural invertibility guarantee (Hoang et al., 2024). CAF-PoNo ensures invertibility using normalizing flows, as does SCORPIO. Whereas CAF-PoNo composes a CDF flow with an inverse sigmoid transform and assumes a Gaussian noise distribution, SCORPIO parameterizes the inverse transformation directly as a sum of sigmoid functions and accommodates several noise families in its likelihood.

In contrast to the recursive sink identification, which requires repeated refitting of each variable against all other possible variables, SCORPIO learns an order through greedy, likelihood-based edge ranking. We adapt this ranking principle from *incEdge* in CAM (Bühlmann et al.), which selects edges by their local improvement in model likelihood. CAM assumes additive parental mechanisms and independent centered Gaussian noises rather than accommodating the more general PNL setting.

For pruning, AbPNL tests the estimated target residual against an excluded observed candidate parent, while CAF-PoNo applies marginal independence tests among latent representations. This requires discarding the order model and fitting a new model for each variable. SCORPIO instead

builds upon the trained model by testing marginal independence between two learned scalar residuals. This construction is motivated by latent conditional independence testing (Duong & Nguyen, 2022; Hoang et al., 2024) and uses the kernel independence test of Zhang et al. (2011) with HSIC (Gretton et al., 2007).

3 PRELIMINARIES

3.1 POST-NONLINEAR STRUCTURAL CAUSAL MODELS

Model and assumptions. We assume a real-valued random vector $X := (X_1, X_2, \ldots, X_p)$ to be generated by an acyclic SCM (S, P_N) (Peters et al., 2017). Here $P_N \colon \mathcal{B}(\mathbb{R}^p) \to [0,1]$ is the joint distribution of the jointly independent exogenous noises $N_1, N_2, \ldots, N_p$, and S is a set of functional assignments of the form

$$X_j = f_{j,2}(f_{j,1}(\mathrm{PA}_j) + N_j) \quad \forall j \in [p]\,. \tag{1}$$

We assume that $f_{j,2} \colon \mathbb{R} \to \mathbb{R}$ is a strictly increasing, continuously differentiable bijection with a continuously differentiable inverse and that $f_{j,1} \colon \mathbb{R}^{|\mathrm{PA}_j|} \to \mathbb{R}$ is continuous. The variables $\mathrm{PA}_j \subseteq \{X_1, X_2, \ldots, X_p\} \setminus \{X_j\}$ are called the parents of X_j. The noise distribution and structural assignments induce a distribution $P_X \colon \mathcal{B}(\mathbb{R}^p) \to [0,1]$ over the endogenous variables, which we assume to have a density $p_X \colon \mathbb{R}^p \to \mathbb{R}_{\geq 0}$.

The assignments give rise to a DAG $G = (V, E)$ with nodes $V = [p]$ and edges $E = \{(i,j) \mid X_i \in \mathrm{PA}_j\}$. Our setting assumes causal sufficiency and that P_X is Markovian with respect to G, i.e.

$$p_X(x_1, \ldots, x_p) = \prod_{j=1}^{p} p_{X_j}(x_j \mid \mathrm{pa}_j). \tag{2}$$

Under their stated regularity assumptions, Zhang & Hyvärinen (2009) show that the bivariate PNL model is identifiable except in five explicitly stated classes.

Orders and fitted models. The topological orders of G are

$$\Pi(G) := \{\pi \in S(p) \mid \pi(i) < \pi(j)\ \forall (i,j) \in E\}\,, \tag{3}$$

where $S(p)$ is the symmetric group of p elements. If G is complete, meaning that exactly one of (i,j) and (j,i) belongs to E for every pair of distinct nodes $i, j \in V$, then $|\Pi(G)| = 1$. We may equivalently represent G by its adjacency matrix $A \in \{0,1\}^{p \times p}$, where $A[i,j] = 1$ if and only if $(i,j) \in E$. For any permutation π of $[p]$, we denote the complete DAG with order π by A^π.

We represent a fitted model by $\hat{M} := (\hat{A}, \hat{\mathcal{F}}_1, \hat{\mathcal{F}}_2^{-1})$, comprising an estimated adjacency matrix, the family of parental mechanisms $\hat{\mathcal{F}}_1 := (\hat{f}_{j,1})_{j \in [p]}$, and the family of inverse outer transformations $\hat{\mathcal{F}}_2^{-1} := (\hat{f}_{j,2}^{-1})_{j \in [p]}$. We write $\hat{M}_j := (\hat{A}_j, \hat{f}_{j,1}, \hat{f}_{j,2}^{-1})$ for the j-th component, where $\hat{A}_j$ is the j-th column of $\hat{A}$. Given n i.i.d. observations $\mathbf{X} = \{\mathbf{x}^{(1)}, \ldots, \mathbf{x}^{(n)}\}$ from Equation 1, we fit this model using the node-wise PNL likelihood described below.

3.2 NORMALIZING FLOWS

Normalizing flows (NFs) are neural networks that learn diffeomorphisms with tractable Jacobian determinants. This enables representing the density of a complex variable X in terms of a simple latent variable Z via the change of variables formula. Given an NF f with $X = f(Z)$, it holds

$$p_X(x) = p_Z\big(f^{-1}(x)\big)\,\big|\det J_{f^{-1}}(x)\big|\,, \tag{4}$$

where the determinant reduces to $\left|df^{-1}(x)/dx\right|$ in the scalar case. We use a scalar NF to model each inverse outer transformation $f_{j,2}^{-1}$.

4 METHOD

SCORPIO learns a DAG in two stages. First, the system's topological order is learned greedily based on node-local improvements of log-likelihood, parameterized via normalizing flows and flexible noise distributions. Second, the corresponding DAG is pruned by testing the significance of a

candidate parent's effect with residual marginal independence tests that leverage the learned model's latent representation.

4.1 Node-wise PNL Likelihood

We represent the graph by an adjacency matrix $\hat{A}$ and model each parental mechanism $\hat{f}_{j,1}$ with an MLP whose inputs are masked by the j-th column of $\hat{A}$ (Zheng et al., 2020; Lachapelle et al., 2020). We use a scalar NF to model $\hat{f}_{j,2}^{-1}$ and obtain a likelihood-based loss from its representation of $\hat{p}_{X_j}(x_j \mid \widehat{\mathrm{pa}}_j)$ in terms of $\hat{p}_{N_j}(n_j)$. We fix a noise distribution family and predict the noise via

$$\hat{n}_j^{(s)} = \hat{f}_{j,2}^{-1}\left(x_j^{(s)}\right) - \hat{f}_{j,1}\left(\widehat{\mathrm{pa}}_j^{(s)}\right), \tag{5}$$

and compute $\hat{p}_{N_j}(n_j)$ by assuming zero mean and estimating the remaining parameters from the data. The change of variables formula in Equation 4 yields the node-wise negative log-likelihood and the loss

$$\mathrm{nll}(\hat{M}) = \sum_{j=1}^{p} \mathrm{nll}_j(\hat{M}_j) = -\frac{1}{n}\sum_{s=1}^{n}\sum_{j=1}^{p}\left[\log \hat{p}_{N_j}\left(\hat{n}_j^{(s)}\right) + \log\left|\frac{d}{dx_j}\hat{f}_{j,2}^{-1}\left(x_j^{(s)}\right)\right|\right]. \tag{6}$$

Our implementation supports Gaussian, Gumbel, Laplace, Student-t, and uniform noise families. We parameterize each inverse outer transformation as a composition of monotone sigmoid sums with an additional linear and bias term (Hoang et al., 2024; Papamakarios et al., 2021):

$$\hat{f}_{j,2}^{-1} = g_j^{(L)} \circ \cdots \circ g_j^{(1)}, \quad g_j^{(\ell)}(x) = b_j^{(\ell)} + a_j^{(\ell)} x + \sum_{k=1}^{K} w_{j,k}^{(\ell)}\, \sigma\left(\alpha_{j,k}^{(\ell)} x + \beta_{j,k}^{(\ell)}\right), \tag{7}$$

Parameters $a_j^{(\ell)}, w_{j,k}^{(\ell)}, \alpha_{j,k}^{(\ell)}$ for $j \in [p], k \in [K], \ell \in [L]$ are enforced positive through a softplus parameterization, yielding a strictly positive derivative. Moreover, the positive linear term implies that the function tends to $\pm\infty$ as x tends to $\pm\infty$. Hence, $\hat{f}_{j,2}^{-1}$ is a smooth bijection from $\mathbb{R}$ to $\mathbb{R}$.

4.2 Order Learning through Greedy Edge Ranking

To learn the order, we adapt the ideas behind CAM's *incEdge* procedure (Bühlmann et al.), which scores potential edges by their local improvement in model likelihood and greedily adds them to the graph. We instantiate this largely model-agnostic principle using the node-wise PNL likelihoods defined above. Algorithm 1 provides compressed pseudocode, and Algorithm 2 provides a detailed version. We note that the identifiability results for the bivariate PNL model do not imply that finite-sample local likelihood improvements consistently recover a multivariate topological order, and we do not establish such a recovery guarantee here.

Initialization and edge scores. We denote by E_{cand} the set of candidate edges that can be added to the graph at each iteration. Initially, the graph $\hat{A}$ is empty and $E_{\mathrm{cand}} = \{(i,j) \in [p]^2 \mid i \neq j\}$. The matrix $S \in \mathbb{R}^{p\times p}$ assigns each edge (i,j) a score quantifying the value of adding (i,j) to $\hat{A}$. All training is performed until the negative log-likelihood ceases to improve or a maximum number of steps is reached. First, we pretrain the $2p$ networks $\hat{f}_{j,1}$ and $\hat{f}_{j,2}^{-1}$. Since $\hat{A}$ is empty, the models learn the marginal distribution of each variable. To compute $S[i,j]$, we copy the current model $\hat{M}_j$ to a candidate model $\hat{M}_j^{+i}$, add (i,j) to its parent mask, and train $\hat{f}_{j,1}^{+i}$. We observed that keeping $\hat{f}_{j,2}^{-1,+i}$ frozen improves the accuracy of the estimated order, likely because candidates for the same target are evaluated under a common outer transformation. Because the likelihood decomposes node-wise, adding (i,j) changes only the j-th term. The score is therefore

$$S[i,j] = \mathrm{nll}_{\mathrm{val},j} - \mathrm{nll}_{\mathrm{val},j}^{+i}, \tag{8}$$

where $\mathrm{nll}_{\mathrm{val},j}$ denotes the validation negative log-likelihood of the j-th model.

Algorithm 1: Order Learning by Greedy Edge Likelihood Ranking

Input: Initial candidate edge set E_{cand}
Output: Fitted model $\hat{M} = (\hat{A}, \hat{\mathcal{F}}_1, \hat{\mathcal{F}}_2^{-1})$, topological order $\hat{\pi}$

$\hat{A} \leftarrow \mathbf{0}^{p \times p}, S \leftarrow -\infty^{p \times p}$;
Train $\hat{M}$ until nll_{val} stagnates; `// pretraining`
For all $(i, j) \in E_{\text{cand}}$: train $\hat{M}_j^{+i}$ and set $S[i, j] = \text{nll}_{\text{val},j} - \text{nll}_{\text{val},j}^{+i}$;
while $E_{\text{cand}} \neq \emptyset$ **do**
  Pick edge $(i^\star, j^\star)$ with maximum score;
  Refine $\hat{M}_{j^\star}^{+i^\star}$ with additional training;
  Update model $\hat{M}_{j^\star} \leftarrow \hat{M}_{j^\star}^{+i^\star}$; `// includes updating` $\hat{A}$
  Remove $(i^\star, j^\star)$ and any edges that would render $\hat{A}$ cyclic from E_{cand};
  Update $S[i, j^\star]$ for all $(i, j^\star) \in E_{\text{cand}}$;
end
$\hat{\pi} \leftarrow$ GETTOPOLOGICALORDER$(\hat{A})$;
return $\hat{M}, \hat{\pi}$

Greedy order construction. After S is initialized for all $(i, j) \in E_{\text{cand}}$, we identify the highest-scoring edge

$$(i^\star, j^\star) = \underset{(i,j) \in E_{\text{cand}}}{\arg\max}\ S[i, j] \tag{9}$$

and remove it from E_{cand}. If $(i^\star, j^\star)$ creates a cycle when added to $\hat{A}$, it is discarded. Otherwise, we add $(i^\star, j^\star)$ to the ranking DAG, replace the previous model $\hat{M}_{j^\star}$ with the candidate model $\hat{M}_{j^\star}^{+i^\star}$, and jointly refine $\hat{f}_{j^\star,1}$ and $\hat{f}_{j^\star,2}^{-1}$ for additional steps, bounded by an upper limit of commit steps $T_{\text{commit}}^{\max}$. Then, scores $S[i, j^\star]$ for remaining candidate parents of $X_{j^\star}$ are updated. This local update, constrained to the j-th model, avoids fitting every remaining variable against all other remaining variables as in recursive sink identification. Without PNS, this process continues until all $\binom{p}{2}$ edges have been added to form a complete ranking DAG $\hat{A}$, from which the unique topological order $\hat{\pi}$ is inferred.

Preliminary neighborhood selection. For larger systems, we perform CAM's preliminary neighborhood selection (PNS) (Bühlmann et al.), which uses generalized additive model (GAM) boosting to restrict the initial E_{cand} to a subset of the $p(p-1)$ directed edges. PNS is enabled for all experiments with $p \geq 50$ and is therefore part of the reported large-system pipeline. Results without PNS are provided in Appendix E.4. Although its additive model assumption does not match the PNL model, we find PNS effective as a screening tool using the default settings of the CAM implementation. The resulting ranking DAG may not be complete, and $\hat{\pi}$ need not be unique; the algorithm returns one such order.

4.3 PRUNING

Conditional independence testing is challenging in high dimensions, while testing estimated noise against an observed candidate parent may retain inherited correlations. To avoid these limitations, we formulate a heuristic based on the inherent latent representation of our model (Duong & Nguyen, 2022; Hoang et al., 2024).

Let $i = \hat{\pi}^{-1}(k)$ and $j = \hat{\pi}^{-1}(\ell)$ with $k < \ell$, i.e. $X_i \in \widehat{\text{PA}}_j$. Denote by $f_{j,1}^{-i}$ the population reduced mechanism obtained after excluding X_i, and define its residual as

$$Z_j^{-i} := f_{j,2}^{-1}(X_j) - f_{j,1}^{-i}(\text{PA}_j \setminus \{X_i\})\,. \tag{10}$$

Under a correct specification, $X_i \notin \text{PA}_j$ implies $f_{j,1}^{-i} = f_{j,1}$ and hence $Z_j^{-i} = N_j$. Independence of the exogenous noises therefore gives $N_i \perp\!\!\!\perp Z_j^{-i}$. Taking the contrapositive yields

$$N_i \not\perp\!\!\!\perp Z_j^{-i} \implies X_i \in \text{PA}_j\,. \tag{11}$$

In practice, we approximate N_i with $\hat{N}_i$ and train a copy $\hat{M}_j^{-i}$ of the target model with edge (i, j) excluded to estimate $\hat{Z}_j^{-i}$. We then test $H_0 \colon \hat{N}_i \perp\!\!\!\perp \hat{Z}_j^{-i}$ using a marginal independence test between the two scalar residuals. If H_0 is rejected, we retain X_i in $\widehat{\mathrm{PA}}_j$, and remove it otherwise. This removal rule is heuristic because failure to reject H_0 does not establish independence. Moreover, a descendant X_k of X_i contained in $\widehat{\mathrm{PA}}_j \setminus \{X_i\}$ may provide information that the retrained $\hat{f}_{j,1}^{-i}$ uses to compensate for the omitted input X_i. We decrease the probability of this case by considering candidate parents in reverse topological order. Our results in Section 5 indicate that this heuristic performs well in the causally sufficient settings considered. In particular, it avoids high-dimensional conditional independence tests and does not impose a distributional assumption in the pruning test. Pseudocode is given in Algorithm 3.

4.4 Complexity Analysis

Overall, SCORPIO has cubic worst-case complexity in p, matching the complexity of the other multivariate PNL methods considered. Appendix B.3 gives a derivation. However, wall-clock time can be substantially lower than the worst-case work bound suggests, see Section 5.1.4, as PNS reduces the number of candidate edges and hence the number of fitted models, while our implementation parallelizes candidate-model training.

5 Experiments

We compare SCORPIO against the multivariate PNL methods AbPNL, RankPNL, and CAF-PoNo, and against CAM, NOTEARS-linear (Zheng et al., 2018), DAGMA (Bello et al., 2022), PC (Spirtes & Glymour, 1991), and LGES (Ejaz & Bareinboim, 2026). We use RankG for Gaussian noise and RankS otherwise. CPDAG results for PC and LGES are given in Appendix D. For methods returning an order, we report order divergence $D(\hat{\pi}, A) := \#\{(i, j) \in [p]^2 \mid A_{i,j} = 1, \hat{\pi}(i) > \hat{\pi}(j)\}$ (Rolland et al., 2022). We evaluate DAG estimates using Structural Hamming Distance (SHD) (Tsamardinos et al., 2006) and Structural Intervention Distance (SID) (Peters & Bühlmann, 2015). F_1 scores are given in Appendix D. Initialization and hyperparameters are given in Appendix C.2.

5.1 Synthetic Data

While the objective of our work is causal discovery for complex data, evaluation on controlled synthetic data allows ablation studies that uncover the behavior of the algorithms with respect to isolated data characteristics. Data are generated with the causal-profiler (Panayiotou et al., 2025) Python package, which we extended with PNL mechanisms and additional noise families. For each benchmark, we independently sample m Erdős–Rényi DAGs of size p, with expected node in-degree q (ERq), and node-wise mechanisms. We then generate n observations in topological order. We consider four complexity classes for the parental mechanisms $f_{j,1}$; see Appendix C.1.1. The outer transformation $f_{j,2}$ is assigned from five possible strictly increasing functions (Appendix C.1.2). Unless stated otherwise, non-deterministic methods are evaluated five times, and results are averaged across trials before aggregation across graphs.

5.1.1 Noise Distribution

We consider Gaussian, Gumbel, Laplace, Student-t, and uniform noise to cover symmetric, asymmetric, light-tailed, heavy-tailed, and bounded distributions. Specifically, we instantiate families as $\mathcal{N}(0, 2)$, $\mathrm{Gumbel}(0.5, 1.5)$, $\mathrm{Laplace}(-2, 4)$, $t_5(0, 2)$, and $\mathcal{U}(-1, 1)$. For each family, we independently generate 100 ER2 graphs with $p = 10$, $n = 1000$, and linear parental mechanisms. We set the noise distribution family in SCORPIO to the generating family, assuming a centered distribution and estimating other parameters from data. Appendix E.3 evaluates a protocol that automatically chooses the distribution family.

SCORPIO attains the lowest mean order divergence for four of five noise families (Figure 1(a)) and the lowest mean SHD and SID for all five, with the largest margin under uniform noise (Figure 2).

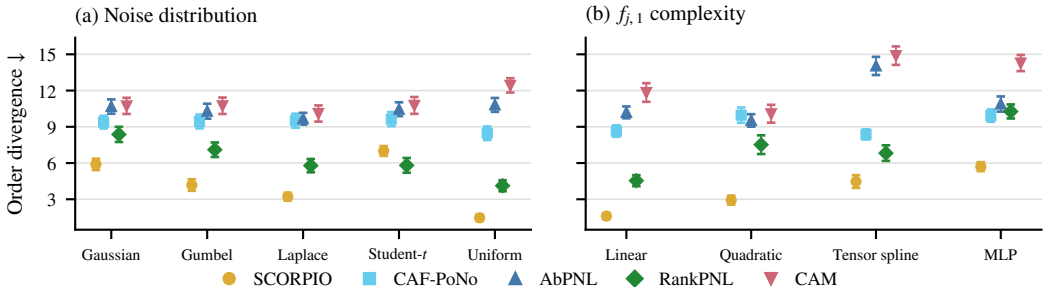

Figure 1: Mean order divergence (lower is better) across (a) noise families and (b) parental-mechanism classes, with 95% Student-t confidence intervals across graphs.

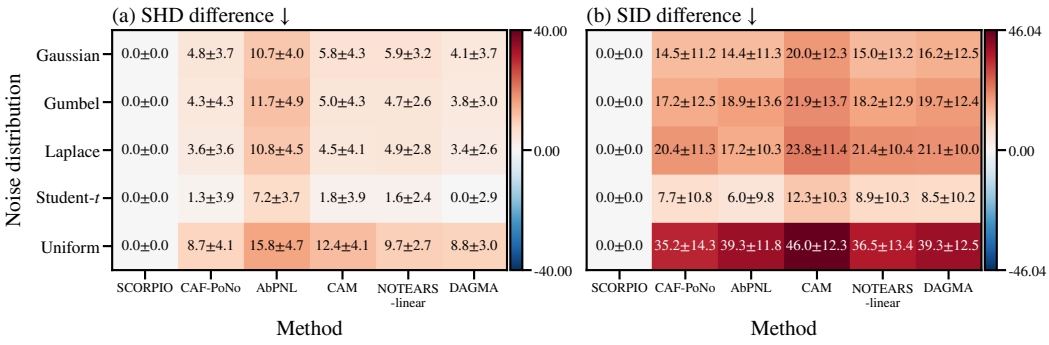

Figure 2: Paired (a) SHD and (b) SID differences Δ = method − SCORPIO across noise families (mean ± standard deviation across graphs). Positive values favor SCORPIO.

5.1.2 Complexity of Nonlinear Parental Mechanisms

We compare linear, quadratic, tensor-product spline, and two-layer MLP parental mechanisms. A detailed description is found in Appendix C.1.1. Each condition consists of $m = 100$ ER2 graphs with $p = 10$ variables, standard uniform noise, and $n = 1000$ samples. SCORPIO attains the lowest mean order divergence, SHD, and SID across all four classes (Figures 1(b) and 3). The margins are smallest for MLP mechanisms, likely because the architecture of the parental model was fixed across the complexity classes. Increasing the MLP complexity could reduce errors further.

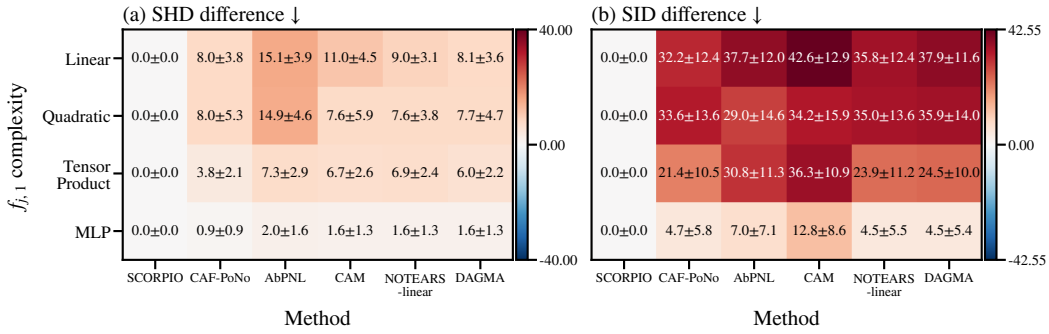

Figure 3: Paired (a) SHD and (b) SID differences Δ = method − SCORPIO across parental-mechanism classes (mean ± standard deviation across graphs). Positive values favor SCORPIO.

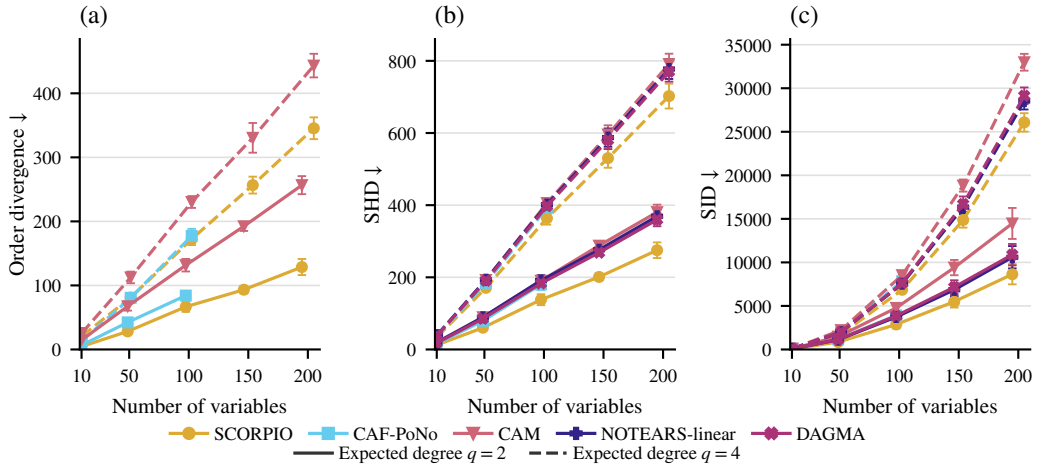

Figure 4: Mean (a) order divergence, (b) SHD, and (c) SID (lower is better) by graph size and expected in-degree, with 95% Student-t confidence intervals across 10 graphs. Missing curves indicate a 24-hour timeout.

Table 1: Mean wall-clock runtime in minutes across 10 Erdős–Rényi graphs with expected in-degree $q = 2$. Dashes indicate a 24-hour timeout. CPU allocations are given in Appendix C.3.

p	**SCORPIO**	**CAF-PoNo**	**AbPNL**	**RankPNL**
10	1.48	3.23	25.77	167.82
50	8.99	100.79	–	–
100	21.62	822.57	–	–
150	34.88	–	–	–
200	48.20	–	–	–

5.1.3 Scalability with Graph Size and Density

To assess scaling with size and density (Lopez et al., 2022; Colombo et al., 2012), we use 10 ER2 and ER4 graphs for each $p \in \{10, 50, 100, 150, 200\}$, with $n = 1000$, tensor-product spline mechanisms, and uniform noise. SCORPIO achieves the lowest mean SHD and SID throughout and the lowest mean order divergence except at $(p, q) = (10, 4)$ and $(50, 4)$, where RankPNL and CAF-PoNo, respectively, perform best (Figure 4). Appendix E.4 isolates the effect of PNS. Table 6 reports results for $p = 500$.

5.1.4 Runtime

We compare the wall-clock runtime of the multivariate PNL algorithms SCORPIO, CAF-PoNo, AbPNL, and RankPNL on the ER2 graphs of Section 5.1.3. AbPNL and RankPNL use one CPU core because their implementations do not support parallel execution, whereas CAF-PoNo and SCORPIO use eight cores for $p \leq 50$ and 16 cores otherwise. Table 1 shows that SCORPIO has the lowest observed wall-clock time under these allocations and is the only method that returns results for $p = 150$ and $p = 200$ within 24 hours. At $p = 100$, SCORPIO averages 21.62 minutes per DAG prediction compared with 13.71 hours for CAF-PoNo.

5.2 Semi-Synthetic Data

We sampled 20 datasets with $p = 98$ variables and $n = 1000$ observations from the semi-synthetic manufacturing-data generator causal_assembly (Göbler et al., 2024). Details are given in Appendix D.4. AbPNL and RankPNL did not complete a single trial within 24 hours and were therefore

Table 2: Order and DAG recovery on the Sachs dataset ($p = 11$, $n = 853$), evaluated against the proposed ground-truth graph. For non-deterministic methods, entries are mean $\pm$ standard deviation across 100 trials. Dashes indicate unavailable order or DAG estimates.

Method	**OD** $\downarrow$	**SHD** $\downarrow$	**SID** $\downarrow$	F_1 **score** $\uparrow$
SCORPIO - Gaussian	$\underline{7.00} \pm 1.81$	15.60 ± 1.50	78.57 ± 7.63	0.3393 ± 0.0716
SCORPIO - Uniform	$\mathbf{5.67} \pm 2.12$	$\underline{14.83} \pm 1.45$	$\underline{71.06} \pm 11.83$	$\underline{0.3957} \pm 0.0927$
CAF-PoNo	10.58 ± 2.30	16.66 ± 1.08	85.62 ± 11.00	0.2443 ± 0.0972
AbPNL	7.61	20.93	86.12	0.2578
RankPNL	11.00	–	–	–
CAM	$\underline{7.00}$	**13.00**	**61.00**	**0.5000**
NOTEARS-linear	–	17.00	76.00	0.1818
DAGMA	–	16.91 ± 0.74	94.42 ± 5.17	0.1787 ± 0.0622

excluded from this benchmark. SCORPIO yields the lowest mean order divergence and SID and the second-lowest mean SHD. CAM and CAF-PoNo achieve the lowest mean SHD and highest mean F_1 score, respectively.

5.3 Real-World Data

Real-world data entail substantial uncertainty about the data-generating process. We evaluate on the protein-signaling dataset commonly referred to as "Sachs" (Sachs et al., 2005), using its proposed ground-truth graph as the reference, and report SCORPIO's performance with Gaussian and uniform noise models. Table 2 shows that SCORPIO's uniform model achieves the lowest order divergence and second-lowest values in all other metrics. CAM achieves the lowest SHD and SID and the highest F_1 score. It should be noted that the results are not directly comparable to those of Hoang et al. (2024) and Lachapelle et al. (2020); those studies evaluated against the model-inference graph of Sachs et al. (2005), whereas we evaluate against the proposed ground-truth graph.

6 Conclusion

We proposed SCORPIO, a causal discovery framework for learning multivariate PNL models. Based on mean point estimates across the 21 evaluated benchmark settings, SCORPIO achieved the lowest order divergence in 18 cases and the lowest SHD in 19 cases. Its greedy edge-ranking procedure achieved lower mean order divergence than CAM's version in the tested settings. SCORPIO is the only multivariate PNL method among CAF-PoNo, AbPNL, and RankPNL that completed experiments with 150 and 200 variables within the 24-hour limit and stated CPU allocations. With prior PNS screening, SCORPIO was able to scale to systems with 500 variables. It performed competitively on both the semi-synthetic manufacturing benchmark and the real-world Sachs benchmark, recovering the most accurate mean order in both settings and the second-lowest mean SHD among estimated DAGs. Overall, the experiments support likelihood-based edge ranking as an empirical alternative to recursive sink identification for multivariate PNL discovery, including in systems containing several hundred variables.

Limitations and Future Work. Several limitations remain. We do not establish a recovery guarantee for multivariate topological orders learned by the greedy procedure. Pruning is heuristic as failure to reject independence does not establish it, and descendants of a true parent may compensate for its omission when the reduced mechanism is retrained. While our synthetic experiments isolate four characteristics often prominent in complex data, namely noise distribution, parental-mechanism complexity, graph size, and edge density, our evaluation includes only one semi-synthetic benchmark from a manufacturing domain and one real-world dataset. These benchmarks may reflect specific causal mechanisms or noise properties and are not representative of real-world data in general. Further semi-synthetic and real-world evaluations are needed to assess SCORPIO's applicability to complex data more broadly. The accuracy of SCORPIO's and CAF-PoNo's predictions may also improve with more samples to support neural network training, particularly in higher-dimensional systems.

AI USE STATEMENT

In this work, we used generative AI to generate synthetic data sets, provide feedback on research methodology or experiments, aid in implementation of specific parts of the method, assist with translation, clean and reformat datasets, and support qualitative and thematic data analysis.

We have not used generative AI tools to help develop theoretical models or conceptual frameworks, propose or refine hypotheses or interpret results.

Formulating mathematical claims, providing critical ingredients for proving mathematical claims, and assisting in the writing of proofs is not applicable to this work.

Additionally, we used generative AI to extend the causal_profiler Python package, create figures and LaTeX tables from data, create or edit software code and artifacts, search for information, edit the paper for readability, and propose a title.

We have reviewed all AI-assisted work. LLM-generated code was verified and tested for correctness. We take responsibility for the final content of this work, including text, claims or artifacts produced with the aid of generative AI.

REPRODUCIBILITY STATEMENT

A zip file with the full repository used to produce the results of this paper is found in the supplementary documents. It includes the implementation of SCORPIO as well as the synthetic data generation pipeline, configuration files specifying seeds and parameters for data creation, experiments and methods and wrappers and code for the benchmarked methods. Thereby, it allows for generation of the data and reproducing of results. Most of the already generated data is also included, although larger systems had to be removed due to size constraints. Furthermore, we list required Python and R packages and provide YAML files for the environments used. Dockerfiles reveal the exact containers that were used to run the experiment on a cluster. The notebooks used to automatically generate plots from results are also provided. Minor numerical differences may occur across hardware and library versions.

REFERENCES

Kevin Bello, Bryon Aragam, and Pradeep Ravikumar. Dagma: Learning dags via m-matrices and a log-determinant acyclicity characterization. In S. Koyejo, S. Mohamed, A. Agarwal, D. Belgrave, K. Cho, and A. Oh (eds.), *Advances in Neural Information Processing Systems*, volume 35, pp. 8226–8239. Curran Associates, Inc., 2022.

Peter Bühlmann, Jonas Peters, and Jan Ernest. CAM: Causal additive models, high-dimensional order search and penalized regression. 42(6). ISSN 0090-5364.

David Maxwell Chickering. Optimal structure identification with greedy search. *J. Mach. Learn. Res.*, 3:507–554, 2002.

Diego Colombo and Marloes H. Maathuis. Order-independent constraint-based causal structure learning. *Journal of Machine Learning Research*, 15(116):3921–3962, 2014.

Diego Colombo, Marloes Maathuis, Markus Kalisch, and Thomas Richardson. Learning high-dimensional directed acyclic graphs with latent and selection variables. *The Annals of Statistics*, 40, 02 2012.

Bao Duong and Thin Nguyen. Conditional independence testing via latent representation learning. In *2022 IEEE International Conference on Data Mining (ICDM)*, pp. 121–130, 2022.

Adiba Ejaz and Elias Bareinboim. Less greedy equivalence search. *Advances in Neural Information Processing Systems*, 38:123068–123109, 2026.

Konstantin Göbler, Tobias Windisch, Mathias Drton, Tim Pychynski, Martin Roth, and Steffen Sontag. `causalAssembly`: Generating realistic production data for benchmarking causal discovery. In Francesco Locatello and Vanessa Didelez (eds.), *Proceedings of the Third Conference on*

Causal Learning and Reasoning, volume 236 of *Proceedings of Machine Learning Research*, pp. 609–642. PMLR, 01–03 Apr 2024.

Arthur Gretton, Kenji Fukumizu, Choon Teo, Le Song, Bernhard Schölkopf, and Alex Smola. A kernel statistical test of independence. *Advances in neural information processing systems*, 20, 2007.

Nu Hoang, Bao Duong, and Thin Nguyen. Enabling causal discovery in post-nonlinear models with normalizing flows. In *ECAI 2024: 27th European Conference on Artificial Intelligence*, volume 392 of *Frontiers in Artificial Intelligence and Applications*, pp. 2661–2668. IOS Press, 2024.

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

Aapo Hyvärinen and Stephen M. Smith. Pairwise likelihood ratios for estimation of non-gaussian structural equation models. *Journal of Machine Learning Research*, 14(4):111–152, 2013.

Aapo Hyvärinen, Kun Zhang, Shohei Shimizu, and Patrik O. Hoyer. Estimation of a structural vector autoregression model using non-gaussianity. *Journal of Machine Learning Research*, 11 (56):1709–1731, 2010.

D. Janzing, J. Peters, JM. Mooij, and B. Schölkopf. Identifying confounders using additive noise models. In *Proceedings of the 25th Conference on Uncertainty in Artificial Intelligence*, pp. 249–257, Corvallis, OR, USA, June 2009. Max-Planck-Gesellschaft, AUAI Press.

Grigor Keropyan, David Strieder, and Mathias Drton. Rank-based causal discovery for post-nonlinear models. In Francisco Ruiz, Jennifer Dy, and Jan-Willem van de Meent (eds.), *Proceedings of The 26th International Conference on Artificial Intelligence and Statistics*, volume 206 of *Proceedings of Machine Learning Research*, pp. 7849–7870. PMLR, 25–27 Apr 2023.

Sébastien Lachapelle, Philippe Brouillard, Tristan Deleu, and Simon Lacoste-Julien. Gradient-based neural dag learning. In *International Conference on Learning Representations*, 2020.

Richard A. Levine and George Casella. Implementations of the monte carlo em algorithm. *Journal of Computational and Graphical Statistics*, 10(3):422–439, 2001. ISSN 10618600.

Sarah Leyder, Jakob Raymaekers, and Tim Verdonck. Tslingam: Directlingam under heavy tails. *Journal of Computational and Graphical Statistics*, 34(2):437–447, 2025.

Romain Lopez, Jan-Christian Huetter, Jonathan Pritchard, and Aviv Regev. Large-scale differentiable causal discovery of factor graphs. In S. Koyejo, S. Mohamed, A. Agarwal, D. Belgrave, K. Cho, and A. Oh (eds.), *Advances in Neural Information Processing Systems*, volume 35, pp. 19290–19303. Curran Associates, Inc., 2022.

Takashi Nicholas Maeda and Shohei Shimizu. Rcd: Repetitive causal discovery of linear non-gaussian acyclic models with latent confounders. In Silvia Chiappa and Roberto Calandra (eds.), *Proceedings of the Twenty Third International Conference on Artificial Intelligence and Statistics*, volume 108 of *Proceedings of Machine Learning Research*, pp. 735–745. PMLR, 26–28 Aug 2020.

Ignavier Ng, Shengyu Zhu, Zhuangyan Fang, Haoyang Li, Zhitang Chen, and Jun Wang. Masked gradient-based causal structure learning. In *Proceedings of the 2022 SIAM International Conference on Data Mining (SDM)*, pp. 424–432. Society for Industrial and Applied Mathematics, 2022. ISBN 978-1-611977-17-2.

Panayiotis Panayiotou, Audrey Poinsot, Alessandro Leite, Nicolas Chesneau, Marc Schoenauer, and Özgür Şimşek. Causalprofiler: Generating synthetic benchmarks for rigorous and transparent evaluation of causal machine learning. *arXiv preprint arXiv:2511.22842*, 2025.

George Papamakarios, Eric Nalisnick, Danilo Jimenez Rezende, Shakir Mohamed, and Balaji Lakshminarayanan. Normalizing flows for probabilistic modeling and inference. *Journal of Machine Learning Research*, 22(57):1–64, 2021.

J. Peters and P. Bühlmann. Structural intervention distance (sid) for evaluating causal graphs. *Neural Computation*, 27(3):771–799, 2015.

J. Peters, D. Janzing, and B. Schölkopf. *Elements of Causal Inference: Foundations and Learning Algorithms*. MIT Press, Cambridge, MA, USA, 2017.

Jonas Peters, Dominik Janzing, and Bernhard Scholkopf. Causal inference on discrete data using additive noise models. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 33(12): 2436–2450, 2011.

Jonas Peters, Joris M. Mooij, Dominik Janzing, and Bernhard Schölkopf. Causal discovery with continuous additive noise models. *Journal of Machine Learning Research*, 15(58):2009–2053, 2014.

Hristo Petkov, Calum MacLellan, and Feng Dong. DAGAF: A directed acyclic generative adversarial framework for joint structure learning and tabular data synthesis. *Applied Intelligence*, 55(7): 602, 2025.

Joseph Ramsey, Madelyn Glymour, Ruben Sanchez-Romero, and Clark Glymour. A million variables and more: the fast greedy equivalence search algorithm for learning high-dimensional graphical causal models, with an application to functional magnetic resonance images. *International journal of data science and analytics*, 3:121 – 129, 2016.

Paul Rolland, Volkan Cevher, Matthäus Kleindessner, Chris Russell, Dominik Janzing, Bernhard Schölkopf, and Francesco Locatello. Score matching enables causal discovery of nonlinear additive noise models. In Kamalika Chaudhuri, Stefanie Jegelka, Le Song, Csaba Szepesvari, Gang Niu, and Sivan Sabato (eds.), *Proceedings of the 39th International Conference on Machine Learning*, volume 162 of *Proceedings of Machine Learning Research*, pp. 18741–18753. PMLR, 17–23 Jul 2022.

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.

Shohei Shimizu. Lingam: Non-gaussian methods for estimating causal structures. *Behaviormetrika*, 41:65–98, 01 2014.

Shohei Shimizu, Patrik O. Hoyer, Aapo Hyvärinen, and Antti Kerminen. A linear non-gaussian acyclic model for causal discovery. *Journal of Machine Learning Research*, 7(72):2003–2030, 2006.

Peter Spirtes and Clark Glymour. An algorithm for fast recovery of sparse causal graphs. *Social Science Computer Review*, 9:62–72, 04 1991.

Peter Spirtes, Clark N. Glymour, and Richard Scheines. *Causation, Prediction, and Search*. The MIT Press, Cambridge, Massachusetts, 2nd edition, 2000. ISBN 978-0-262-19440-2.

Ioannis Tsamardinos, Laura E. Brown, Constantin F. Aliferis, and Andrew W. Moore. The max-min hill-climbing bayesian network structure learning algorithm. 65(1):31–78, 2006.

Ruibo Tu, Kun Zhang, Hedvig Kjellstrom, and Cheng Zhang. Optimal transport for causal discovery. In *International Conference on Learning Representations*, 2022.

Kento Uemura and Shohei Shimizu. Estimation of post-nonlinear causal models using autoencoding structure. In *ICASSP 2020 - 2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 3312–3316, 2020.

Kento Uemura, Takuya Takagi, Kambayashi Takayuki, Hiroyuki Yoshida, and Shohei Shimizu. A multivariate causal discovery based on post-nonlinear model. In Bernhard Schölkopf, Caroline Uhler, and Kun Zhang (eds.), *Proceedings of the First Conference on Causal Learning and Reasoning*, volume 177 of *Proceedings of Machine Learning Research*, pp. 826–839. PMLR, 11–13 Apr 2022.

Marcel Wienöbst, Leonard Henckel, and Sebastian Weichwald. Embracing discrete search: A reasonable approach to causal structure learning. In C. Vondrick, B. Hariharan, C. Raffel, L. Pinto, D. Yang, and A. Faust (eds.), *International Conference on Learning Representations*, volume 2026, pp. 54630–54652, 2026.

K Zhang and Aapo Hyvärinen. On the identifiability of the post-nonlinear causal model. In *Proc. 25th Conference on Uncertainty in Artificial Intelligence (UAI2009)*, 2009.

Kun Zhang and Aapo Hyvärinen. Distinguishing causes from effects using nonlinear acyclic causal models. In Isabelle Guyon, Dominik Janzing, and Bernhard Schölkopf (eds.), *Proceedings of Workshop on Causality: Objectives and Assessment at NIPS 2008*, volume 6 of *Proceedings of Machine Learning Research*, pp. 157–164, Whistler, Canada, 12 Dec 2010. PMLR.

Kun Zhang, Jonas Peters, Dominik Janzing, and Bernhard Schölkopf. Kernel-based conditional independence test and application in causal discovery. In *Proceedings of the Twenty-Seventh Conference on Uncertainty in Artificial Intelligence*, UAI'11, pp. 804–813, Arlington, Virginia, USA, 2011. AUAI Press. ISBN 9780974903972.

Kun Zhang, Zhikun Wang, Jiji Zhang, and Bernhard Schölkopf. On estimation of functional causal models: General results and application to the post-nonlinear causal model. *ACM Trans. Intell. Syst. Technol.*, 7(2), December 2015. ISSN 2157-6904.

Xun Zheng, Bryon Aragam, Pradeep K Ravikumar, and Eric P Xing. Dags with no tears: Continuous optimization for structure learning. *Advances in neural information processing systems*, 31, 2018.

Xun Zheng, Chen Dan, Bryon Aragam, Pradeep Ravikumar, and Eric Xing. Learning sparse nonparametric dags. In *International conference on artificial intelligence and statistics*, pp. 3414–3425. Pmlr, 2020.

Shengyu Zhu, Ignavier Ng, and Zhitang Chen. Causal discovery with reinforcement learning. In *International Conference on Learning Representations*, 2020.

A Extended Related Work

The PNL model was first introduced by Zhang & Hyvärinen (2010) for bivariate relationships. They proposed MLPs for function approximation and independent component analysis for determining the causal direction. In the follow-up work Zhang & Hyvärinen (2009), they proved identifiability of the PNL model except for five explicitly listed cases and outlined a multivariate extension: use a constraint-based method to learn the Markov equivalence class structure and subsequently orient candidate edges by estimating effect-variable noises and testing their independence from candidate parents. Zhang et al. (2015) improved on drawbacks of the previous method like initialization dependence and function identifiability issues with a warped Gaussian process framework with Gaussian-mixture noise models. They showed that for any functional model, mutual information minimization between causes and noises is equivalent to maximizing the likelihood of the system variables and leveraged that finding by using Monte Carlo expectation maximization (Levine & Casella, 2001) to avoid computationally heavy independence testing. Uemura & Shimizu (2020) instead introduced an autoencoding reconstruction objective for the invertibility constraint, while Tu et al. (2022) formulated bivariate discovery through optimal transport and proposed velocity-field divergence as an alternative to HSIC (Gretton et al., 2007).

Scaling the PNL framework to the multivariate case has largely followed the same structure: determine a topological order through iterative sink identification based on residual-independence scores, following the principle of RESIT (Peters et al., 2014), and subsequently prune superfluous edges. AbPNL (Uemura et al., 2022) extends the autoencoding loss formulation to the multivariate setting, but does not impose architectural invertibility. To avoid constructing an invertible outer transformation, RankPNL (Keropyan et al., 2023) exploits the rank-preserving property of the outer transformation and estimates a topological order rather than a pruned DAG. CAF-PoNo (Hoang et al., 2024) was the first to use NFs to learn the outer transformation. The CDF flow provides a tractable Jacobian determinant and is concatenated with an inverse sigmoid transformation to obtain a full real line image. They avoid the restriction imposed by the standard uniform output of an individually trained CDF flow by fitting the flow jointly with the parental mechanisms through maximum

likelihood. While SCORPIO also uses NFs, the sum of monotone sigmoids architecture directly yields an image equal to the real line and could potentially be trained separately from the parental mechanisms.

B METHOD DETAILS

We complement the method description with detailed pseudocode for order learning (Algorithm 2) and pruning (Algorithm 3) and elaborate on the method's computational complexity.

B.1 ORDER LEARNING - DETAILED PSEUDOCODE

Algorithm 2: Order Learning by Greedy Edge Likelihood Ranking - Detailed

Input: Initial candidate edge set E_{cand}, training data $\mathbf{X}_t \in \mathbb{R}^{n_1 \times p}$, validation data $\mathbf{X}_v \in \mathbb{R}^{n_2 \times p}$, upper bounds for training steps $T^{\max}_{\text{pretrain}}, T^{\max}_{\text{score}}, T^{\max}_{\text{commit}}$

Output: Fitted model $\hat{M} = (\hat{A}, \hat{\mathcal{F}}_1, \hat{\mathcal{F}}_2^{-1})$, topological order $\hat{\pi}$

Function SCOREEDGE($\hat{M}_j, i$)
  $\hat{M}_j^{+i} \leftarrow$ COPY($\hat{M}_j$);
  Add edge (i, j) to parent mask of $\hat{M}_j^{+i}$;
  Train $\hat{f}_{j,1}^{+i}$ until $\text{nll}_{\text{val},j}^{+i}$ stagnates or $T^{\max}_{\text{score}}$ steps are reached, keeping $\hat{f}_{j,2}^{-1,+i}$ fixed;
  return $\text{nll}_{\text{val},j}^{+i}, \hat{M}_j^{+i}$
end
$\hat{A} \leftarrow \mathbf{0}^{p \times p}, S \leftarrow -\infty^{p \times p}$;
foreach $j \in [p]$ **do**
  `// pretraining`
  Train $\hat{M}_j$ until $\text{nll}_{\text{val},j}$ stagnates or $T^{\max}_{\text{pretrain}}$ steps are reached;
  foreach $i \in [p]$ *such that* $(i, j) \in E_{\text{cand}}$ **do**
    `// initialize score`
    $\text{nll}_{\text{val},j}^{+i}, \hat{M}_j^{+i} \leftarrow$ SCOREEDGE($\hat{M}_j, i$);
    $S[i, j] \leftarrow \text{nll}_{\text{val},j} - \text{nll}_{\text{val},j}^{+i}$;
  end
end
`// greedy edge selection`
while $E_{\text{cand}} \neq \emptyset$ **do**
  $(i^\star, j^\star) \leftarrow \arg\max_{(i,j) \in E_{\text{cand}}} S[i, j]$;
  `// update model to include new edge`
  Jointly train $\hat{f}_{j^\star,1}^{+i^\star}$ and $\hat{f}_{j^\star,2}^{-1,+i^\star}$ until $\text{nll}_{\text{val},j^\star}$ stagnates or $T^{\max}_{\text{commit}}$ steps are reached;
  $\hat{M}_{j^\star} \leftarrow \hat{M}_{j^\star}^{+i^\star}$; `// includes updating` $\hat{A}$
  `// update score matrix`
  Remove $(i^\star, j^\star)$ and any edges that would render $\hat{A}$ cyclic from E_{cand};
  foreach $i \in [p]$ *such that* $(i, j^\star) \in E_{\text{cand}}$ **do**
    $\text{nll}_{\text{val},j^\star}^{+i}, \hat{M}_{j^\star}^{+i} \leftarrow$ SCOREEDGE($\hat{M}_{j^\star}, i$);
    $S[i, j^\star] \leftarrow \text{nll}_{\text{val},j^\star} - \text{nll}_{\text{val},j^\star}^{+i}$;
  end
end
$\hat{\pi} \leftarrow$ GETTOPOLOGICALORDER($\hat{A}$);
return $\hat{M}, \hat{\pi}$

B.2 PRUNING - PSEUDOCODE

Algorithm 3: Pruning

Input: Fitted order estimation model $\hat{M}$ entailing $\hat{A}$, estimated order $\hat{\pi}$, training data $\mathbf{X}_t \in \mathbb{R}^{n_1 \times p}$, validation data $\mathbf{X}_v \in \mathbb{R}^{n_2 \times p}$, significance level α, maximum number of training steps $T_{\text{prune}}^{\max}$

Output: Pruned and refitted $\hat{M}$

for $\ell \leftarrow 2$ **to** p **do**
  $j \leftarrow \hat{\pi}^{-1}(\ell)$;
  $\widehat{\mathrm{PA}}_j \leftarrow \left\{ i \in [p] \mid \hat{A}[i,j] = 1 \right\}$;
  foreach $i \in \widehat{\mathrm{PA}}_j$ *in decreasing order of* $\hat{\pi}(i)$ **do**
    $\hat{M}_j^{-i} \leftarrow \textsc{Copy}(\hat{M}_j)$;
    Remove X_i from parent mask of $\hat{f}_{j,1}^{-i}$;
    Train $\hat{f}_{j,1}^{-i}$ on $\mathbf{X}_t$ until $\mathrm{nll}_{\mathrm{val},j}^{-i}$ stagnates or $T_{\text{prune}}^{\max}$ steps are reached, keep $\hat{f}_{j,2}^{-1,-i}$ fixed;
    Sample batch $\mathbf{x}$ from $\mathbf{X}_v$;
    Predict $\hat{n}_i$ with $\hat{M}_i$ `// unmodified`;
    Predict $\hat{z}_j^{-i}$ with $\hat{M}_j^{-i}$ `// candidate parent removed`;
    $\mathrm{pval} \leftarrow \textsc{KernelIndependenceTest}(\hat{n}_i, \hat{z}_j^{-i})$;
    if $\mathrm{pval} \geq \alpha$ **then**
      $\hat{A}[i,j] \leftarrow 0$;
      $\hat{M}_j \leftarrow \hat{M}_j^{-i}$;
    end
  end
end

B.3 COMPUTATIONAL COMPLEXITY ANALYSIS

Let C be an upper bound on the cost of training any candidate model $\hat{M}_j^{+i}$ or reduced model $\hat{M}_j^{-i}$; it contains the dependence on the sample size, network architecture, and optimizer iterations. To initialize the score matrix, $p(p-1)$ candidate models are trained. After initialization, each of the $\binom{p}{2}$ edge additions requires updating the j-th column of S through training of at most p candidate models. The fixed-budget commit refinements and the pruning phase add only quadratically many node-model updates. Together, these terms give cubic worst-case work in p.

C EXPERIMENTAL SETUP

C.1 SYNTHETIC DATA GENERATION

C.1.1 PARENTAL MECHANISMS

We consider four parameterizations of the parental mechanisms $f_{j,1}(\mathrm{pa}_j)$: linear, quadratic, tensor-product spline, and MLP. The parental-mechanism benchmark in Section 5.1.2 includes all four classes, using $K = 6$ univariate B-spline basis functions per coordinate for the tensor-product spline and an MLP hidden-layer width of $h = 10$. The noise-distribution benchmark in Section 5.1.1 uses linear mechanisms, whereas the graph-size-and-density benchmark in Section 5.1.3 uses tensor-product splines with $K = 6$. Figure 5 shows one realization of each class for two parents. We abbreviate $f := f_{j,1}$, $d := |\mathrm{PA}_j|$, and $\mathbf{x} := \mathrm{pa}_j \in \mathbb{R}^d$ for simplicity.

- **Linear:** The simplest option for the parental mechanism is an affine-linear function
$$f(\mathbf{x}) = \mathbf{w}^T \mathbf{x} + b. \tag{12}$$
Parameters are sampled as $\mathbf{w} \sim \mathcal{N}(\mathbf{0}, d^{-1} I_d)$ and $b \sim \mathcal{N}(0,1)$. The fan-in scaling keeps the variance of the linear component approximately stable with respect to d.

- **Quadratic:** The quadratic mechanism contains coordinate-wise squares, pairwise interactions, linear terms and a bias

$$f(\mathbf{x}) = \sum_{i=1}^{d} a_{ii} x_i^2 + \sum_{\substack{i,k \in [d] \\ i<k}} a_{ik} x_i x_k + \mathbf{w}^\top \mathbf{x} + b. \tag{13}$$

  Let $Q_d := d + \binom{d}{2}$ be the number of quadratic terms. The weights are sampled independently as $a_{ii}, a_{ik}, w_i \sim \mathcal{N}\big(0, (d+Q_d)^{-1}\big)$ for $i \in [d]$ and $1 \le i < k \le d$, and $b \sim \mathcal{N}(0,1)$.

- **Tensor-product of cubic B-splines:** Let $K \ge 4$ be the number of univariate, cubic B-spline basis functions per coordinate. For the tensor-product spline mechanism, we transform the input $\mathbf{x} \in \mathbb{R}^d$ through a logistic map to $\tilde{\mathbf{x}} \in (0,1)^d$ and use the open, clamped knot vector

$$t = (t_0, \dots, t_{K+3}) = \left(0,0,0,0, \frac{1}{K-3}, \frac{2}{K-3}, \dots, \frac{K-4}{K-3}, 1,1,1,1\right) \tag{14}$$

  for $K > 4$ and $t = (0,0,0,0,1,1,1,1)$ otherwise. The ℓ-th B-spline basis function $B_{\ell,3}$ is defined through the Cox–de Boor recursion with support on the interval $[t_\ell, t_{\ell+4}]$. The tensor-product basis functions are then constructed as a product of one B-spline basis function per coordinate:

$$\Phi_{\boldsymbol{\alpha}}(\mathbf{x}) = \prod_{\ell=1}^{d} B_{\alpha_\ell,3}(x_\ell), \tag{15}$$

  where $\boldsymbol{\alpha} = (\alpha_1, \dots, \alpha_d) \in \{0, \dots, K-1\}^d$. Finally, the parent mechanism is a linear combination of these basis functions

$$f(\mathbf{x}) = \sum_{\boldsymbol{\alpha} \in \{0,\dots,K-1\}^d} w_{\boldsymbol{\alpha}} \Phi_{\boldsymbol{\alpha}}(\mathbf{x}) + b, \tag{16}$$

  where the sum iterates over all K^d possible basis functions if $K^d \le 1024$. If $K^d > 1024$, we sample 1024 distinct multi-indices uniformly from $\{0, \dots, K-1\}^d$ to reduce computational cost. Weights are sampled independently as $w_{\boldsymbol{\alpha}}, b \sim \mathcal{N}(0,1)$.

- **MLP**: The neural-network parental mechanism is a randomly initialized feed-forward network with two hidden layers of width h, ReLU activations and a linear output:

$$f(\mathbf{x}) = \sum_{\ell=1}^{h} w_\ell^{(2)} a\left(\sum_{k=1}^{h} w_{\ell k}^{(1)} a\left(\sum_{i=1}^{d} w_{ki}^{(0)} x_i + b_k^{(0)}\right) + b_\ell^{(1)}\right) + b^{(2)}, \tag{17}$$

  where $a(x) = \max\{0, x\}$. Parameters are sampled using PyTorch's default initialization for linear layers and remain fixed during data generation.

C.1.2 Outer Transformations

The strictly increasing outer transformation $f_{j,2}$ is determined from the following five possibilities:

- **Signed logarithm:**

$$f(z) = \operatorname{sgn}(z) \log(1 + |z|). \tag{18}$$

- **Inverse hyperbolic sine:**

$$f(z) = \operatorname{asinh}(z). \tag{19}$$

- **Signed power:**

$$f(z) = \operatorname{sgn}(z)\ |z|^{3/2}. \tag{20}$$

- **Leaky hyperbolic tangent:**

$$f(z) = \tanh(z) + 0.1\, z. \tag{21}$$

- **Wavy linear:**

$$f(z) = z + \frac{1-\varepsilon}{k}\sin(kz). \tag{22}$$

We used $\varepsilon = 10^{-4}$ and $k = 3$ in all of our experiments.

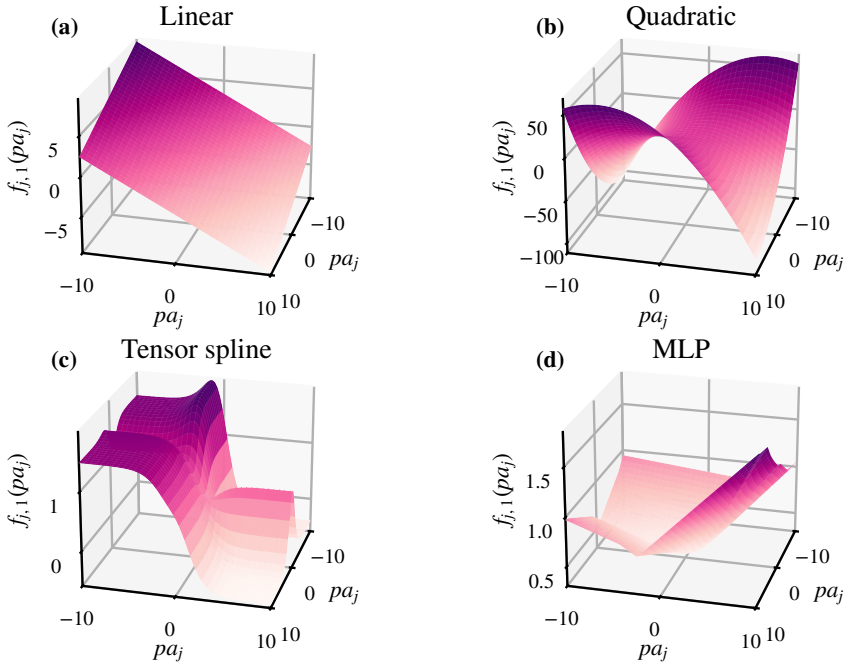

Figure 5: Examples of the four parental-mechanism classes used in the synthetic benchmarks: (a) linear, (b) quadratic, (c) tensor-product spline with cubic B-spline basis, and (d) two-layer MLP. Each panel shows one realization of $f_{j,1}$ with two parents. Vertical scales differ across panels.

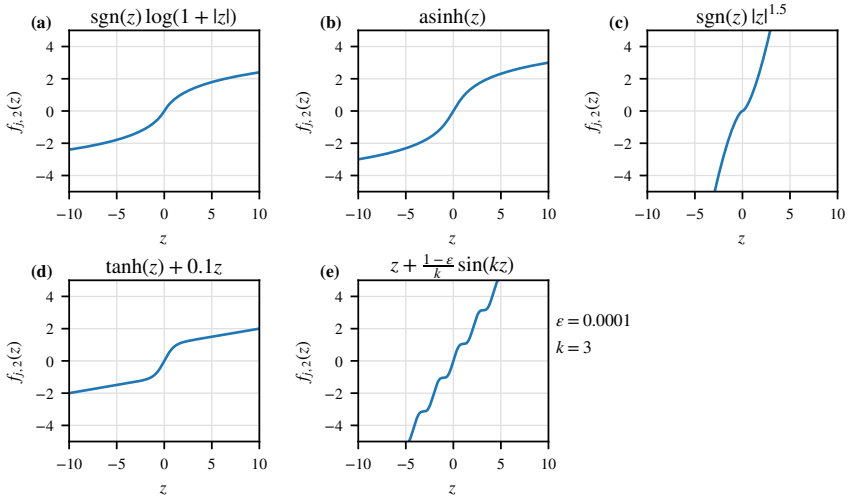

Figure 6: The five strictly increasing outer transformations $f_{j,2}$ used during synthetic data generation: (a) signed logarithm, (b) inverse hyperbolic sine, (c) signed power, (d) leaky hyperbolic tangent, and (e) wavy linear transformation with $\varepsilon = 10^{-4}$ and $k = 3$.

Table 3: Default SCORPIO hyperparameters used in all experiments. The noise distribution family is varied as specified in the respective experimental sections.

Parameter	Value
Data Preprocessing	
Z-Score normalization	True
Validation fraction	0.3
Batch size	512
Model Architecture	
$\hat{f}_{j,1}$ hidden layers	1
$\hat{f}_{j,1}$ hidden dimension	11
$\hat{f}_{j,1}$ activation	LeakyReLU
$\hat{f}_{j,2}$ NF type	Monotone sigmoid
$\hat{f}_{j,2}$ components	10
$\hat{f}_{j,2}$ layers	2
Noise distribution	Gaussian
PNS	
Execute	True for $p \geq 50$
Settings	CAM implementation defaults
Order Estimation	
Optimizer	RMSprop
Optimizer Learning Rate	10^{-3}
Max pretraining iterations	1000
Max edge-added model training iterations	300
Max edge-added model commit iterations	800
Validation NLL std window	20
Validation NLL std threshold	10^{-2}
Pruning	
Max edge-removed model training iterations	1000
Validation NLL std window	20
Validation NLL std threshold	10^{-4}
Kernel type	Polynomial
KCI samples	4000
Significance level	10^{-3}

All five functions are continuously differentiable, strictly increasing bijections from $\mathbb{R}$ to $\mathbb{R}$. The inverse of the signed-power transformation is not differentiable at zero, a measure-zero exception to the stated model regularity assumption. Figure 6 shows their shapes over a common input range.

C.2 Hyperparameters

In all experiments, $\hat{f}_{j,1}$ is Xavier-initialized with LeakyReLU gain, and $\hat{f}_{j,2}$ is initialized to approximate the identity. The default hyperparameters used throughout the experiments are listed in Table 3. In the main experiments, the only varied model hyperparameter is the noise distribution family, whose realizations are specified in the respective experimental sections.

C.3 Runtime Evaluation

Runtime experiments use one CPU core for AbPNL and RankPNL because their implementations do not support parallel execution. CAF-PoNo and SCORPIO use eight CPU cores for $p \leq 50$ and 16 cores otherwise. We report wall-clock time, and omit a result if a single execution exceeded 24

hours. AbPNL timed out at $p = 50$ and was therefore not evaluated on larger systems. RankS was infeasible within the time limit for $p = 10$, so we report RankG as an alternative.

D Additional Results

This section supplements the DAG results with F_1 scores and evaluations in Markov-equivalence-class space. PC and LGES are two popular causal discovery methods that estimate the CPDAG of a system. For a joint comparison, we convert DAG estimates of SCORPIO, CAF-PoNo, AbPNL, CAM, NOTEARS-linear, and DAGMA to their CPDAG and compute SHD-CPDAG and SID-CPDAG. We emphasize that only PC and LGES are designed for CPDAG prediction. SHD-CPDAG is averaged over trials and reported as mean $\pm$ standard deviation across datasets, or across trials for Sachs. SID-CPDAG is the interval of SID values for the DAGs contained in that Markov equivalence class. Interval bounds are averaged across datasets and trials, and computed with the R package `SID`.

D.1 Noise Distribution

Across the five noise families, SCORPIO has the highest mean F_1 score in every setting (Figure 7). In CPDAG space, it attains the lowest mean SHD-CPDAG for Gaussian, Gumbel, and uniform noise, while LGES performs best for Laplace and Student-t noise (Table 4).

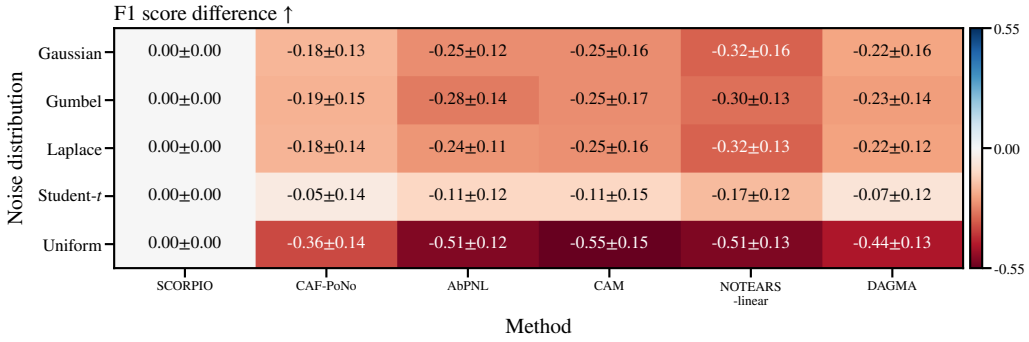

Figure 7: Paired F_1 score differences $\Delta F_1 = F_1(\text{method}) - F_1(\text{SCORPIO})$ across five noise families. For each of the 100 graphs per family, five trials are first averaged for nondeterministic methods. Values report the mean $\pm$ standard deviation of the paired differences across graphs. Positive values favor the baseline and negative values favor SCORPIO.

D.2 Parental-Mechanism Complexity

SCORPIO has the highest mean F_1 score in all four mechanism classes, with the smallest advantage for MLP mechanisms (Figure 8). It also obtains the lowest mean SHD-CPDAG in every class (Table 5), although the gap to the next-best method narrows for the more complex tensor-product spline and MLP mechanisms.

D.3 Scalability with Graph Size and Density

Figure 9 shows that the F_1 score stays relatively stable across graph sizes and methods, with a lower baseline for denser graphs with expected node in-degree $q = 4$. SCORPIO achieves the highest F_1 scores. Figure 10 reports SHD-CPDAG for all methods, including PC and LGES, highlighting the strong performance of SCORPIO with PNS execution even in Markov equivalence class space.

Table 4: CPDAG recovery across five noise families on 100 graphs per family ($p = 10$, $q = 2$, $n = 1000$). SHD-CPDAG entries are mean $\pm$ standard deviation across graphs. SID-CPDAG entries are the mean lower and upper bounds across graphs. Lower values are better. Bold and underlined SHD-CPDAG values mark the best and second-best methods within each noise family.

Noise distribution	Metric	SCORPIO	CAF-PoNo	AbPNL	CAM	NOTEARS-linear	DAGMA	LGES	PC
Gaussian	SHD-C	**13.2 ± 4.1**	18.3 ± 4.3	24.1 ± 5.0	18.5 ± 4.9	19.1 ± 3.4	17.8 ± 3.4	13.6 ± 5.7	14.8 ± 4.3
	SID-C	[42.0, 60.0]	[51.3, 71.7]	[57.3, 68.2]	[55.6, 71.9]	[52.4, 72.3]	[50.9, 74.3]	[45.4, 56.2]	[53.8, 65.7]
Gumbel	SHD-C	**13.6 ± 3.9**	18.5 ± 4.2	25.0 ± 5.0	17.8 ± 4.7	18.7 ± 3.6	17.9 ± 3.7	13.9 ± 5.1	15.2 ± 4.1
	SID-C	[38.0, 58.5]	[48.5, 69.9]	[54.8, 67.0]	[50.5, 67.9]	[50.3, 70.6]	[47.2, 71.5]	[41.7, 53.9]	[50.3, 64.1]
Laplace	SHD-C	14.7 ± 4.2	18.2 ± 4.4	24.8 ± 5.5	17.3 ± 4.8	19.2 ± 3.5	18.5 ± 3.8	**14.2 ± 5.4**	15.9 ± 4.1
	SID-C	[38.0, 59.5]	[47.8, 71.0]	[49.7, 65.0]	[48.7, 67.5]	[50.1, 69.5]	[47.5, 72.4]	[41.8, 54.5]	[52.7, 64.5]
Student-t	SHD-C	17.4 ± 4.0	18.8 ± 4.3	24.3 ± 4.6	17.8 ± 4.7	18.8 ± 3.8	18.0 ± 3.8	**14.2 ± 5.7**	15.8 ± 4.1
	SID-C	[46.6, 66.1]	[50.9, 70.3]	[54.8, 66.5]	[52.2, 70.4]	[51.7, 71.9]	[49.1, 73.0]	[44.1, 55.7]	[53.6, 66.1]
Uniform	SHD-C	**7.6 ± 3.7**	16.8 ± 4.5	23.8 ± 4.9	19.1 ± 4.2	17.8 ± 3.4	17.1 ± 3.6	12.1 ± 5.7	13.4 ± 4.2
	SID-C	[23.3, 45.2]	[47.3, 67.0]	[57.2, 68.5]	[57.2, 72.3]	[49.2, 71.4]	[49.5, 71.7]	[40.8, 53.8]	[48.7, 61.1]

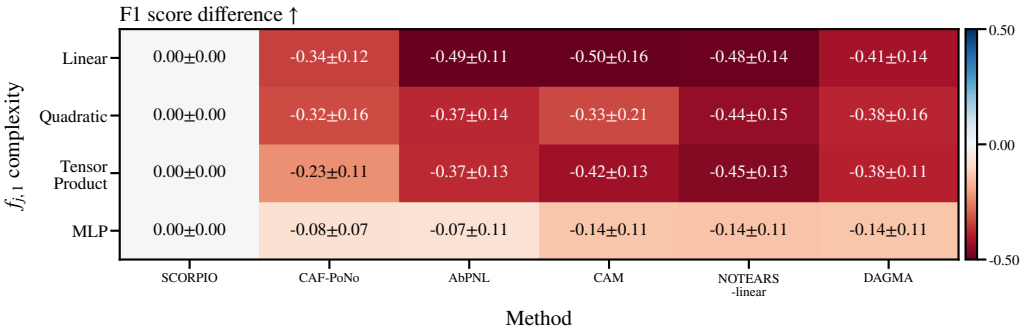

Figure 8: Paired F_1 score differences $\Delta F_1 = F_1(\text{method}) - F_1(\text{SCORPIO})$ across four parental-mechanism classes. For each of the 100 graphs per class, five trials are first averaged for nondeterministic methods. Cells then report the mean $\pm$ standard deviation of the paired differences across graphs. Positive values favor the baseline and negative values favor SCORPIO.

Table 5: CPDAG metrics on the parental-mechanism complexity dataset, consisting of 100 Erdős–Rényi graphs with $p = 10$ variables, expected node in-degree $q = 2$ and $n = 1000$ samples per complexity class. SHD-CPDAG entries are mean $\pm$ standard deviation across graphs. SID-CPDAG entries are the mean lower and upper bounds across graphs. Lower values are better. Bold and underlined SHD-CPDAG values mark the best and second-best methods within each class.

Algorithm	Linear		Quadratic		Tensor Product		MLP	
	SHD	SID	SHD	SID	SHD	SID	SHD	SID
SCORPIO	**8.1 ± 3.5**	[24.0, 45.1]	**10.9 ± 4.9**	[25.8, 47.5]	**14.7 ± 4.3**	[38.8, 60.9]	**19.4 ± 3.4**	[57.7, 66.6]
CAF-PoNo	16.8 ± 4.2	[45.3, 66.2]	19.6 ± 5.2	[48.6, 71.0]	17.8 ± 3.2	[51.6, 71.3]	19.8 ± 3.3	[59.8, 66.6]
AbPNL	23.8 ± 4.3	[57.3, 67.4]	26.6 ± 6.1	[48.9, 65.1]	20.9 ± 3.7	[65.2, 74.6]	20.9 ± 3.2	[60.7, 68.6]
CAM	18.6 ± 4.5	[55.7, 70.1]	18.3 ± 5.5	[46.2, 68.0]	19.8 ± 3.5	[66.2, 77.8]	19.8 ± 3.3	[61.6, 71.2]
NOTEARS-linear	17.8 ± 3.3	[49.7, 71.0]	18.8 ± 3.4	[51.8, 74.2]	19.6 ± 3.4	[58.4, 70.4]	20.2 ± 3.3	[62.8, 62.8]
DAGMA	17.2 ± 3.9	[49.3, 71.4]	19.3 ± 4.4	[53.1, 72.4]	19.0 ± 3.6	[56.9, 70.7]	20.1 ± 3.3	[62.6, 62.9]
LGES	12.1 ± 5.6	[40.9, 53.4]	16.2 ± 7.0	[43.0, 53.0]	16.4 ± 4.4	[53.1, 65.1]	19.5 ± 3.3	[59.9, 69.9]
PC	13.4 ± 4.1	[47.8, 60.8]	15.7 ± 4.2	[53.4, 65.9]	17.3 ± 4.0	[57.6, 68.7]	19.8 ± 3.5	[61.5, 67.9]

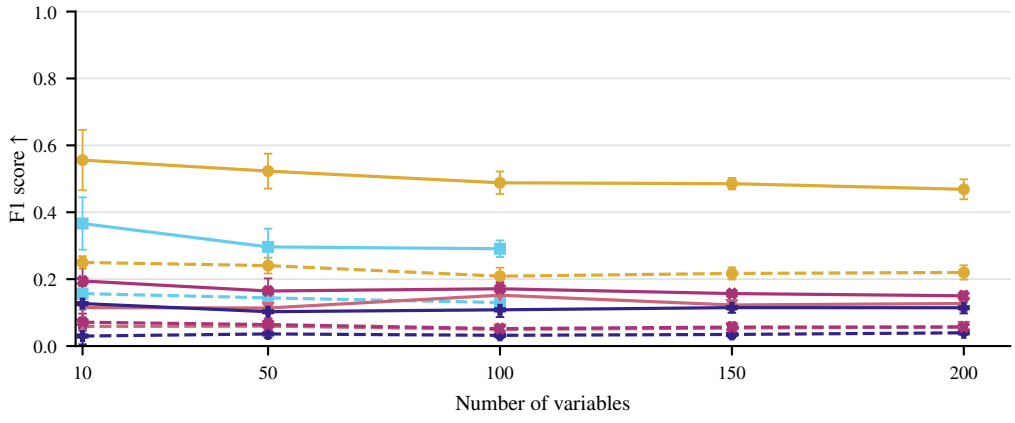

Figure 9: Mean F_1 score (higher is better) on Erdős–Rényi graphs of size $p \in \{10, 50, 100, 150, 200\}$ and expected in-degree $q \in \{2, 4\}$. Solid and dashed curves denote $q = 2$ and $q = 4$, respectively. Error bars show 95% Student-t confidence intervals across 10 graphs. Values are omitted when a method exceeds the 24-hour limit.

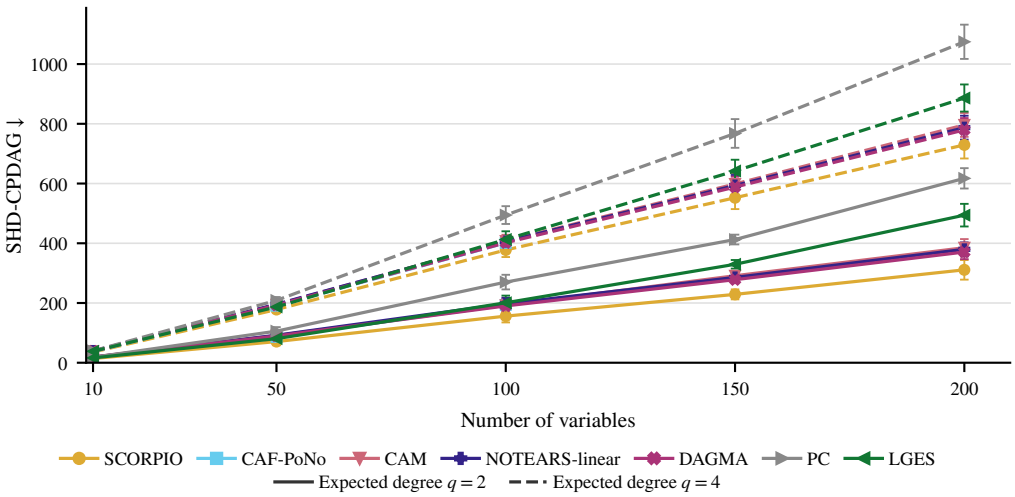

Figure 10: Mean SHD-CPDAG (lower is better) on Erdős–Rényi graphs of size $p \in \{10, 50, 100, 150, 200\}$ and expected in-degree $q \in \{2, 4\}$. Solid and dashed curves denote $q = 2$ and $q = 4$, respectively. Error bars show 95% Student-t confidence intervals across 10 graphs. Values are omitted when a method exceeds the 24-hour limit.

D.3.1 500-NODE GRAPHS

We extended the benchmark of Section 5.1.3 by 10 ER2 and ER4 graphs of size $p = 500$. The causal mechanisms are generated in accordance with the other data of the benchmark: post-nonlinear mechanisms with tensor-product spline parental functions, outer transformations assigned as described in Appendix C.1.2, and $\mathcal{U}(0, 1)$ noise. Table 6 reports mean order divergence, SHD, and F_1 score for the DAG-predicting methods that remained feasible on graphs of this size within a 24-hour time limit. SCORPIO achieves the best result across all reported accuracy metrics and graph densities, with the drawback of being slower than CAM and NOTEARS-linear. Table 7 lists SHD-CPDAG, highlighting SCORPIO as yielding the CPDAG prediction closest to the ground-truth CPDAG in terms of edge manipulations.

Table 6: DAG recovery and wall-clock runtime on 10 Erdős–Rényi graphs of size $p = 500$, expected node in-degree $q \in \{2, 4\}$ and $n = 1000$ samples. OD, SHD, and F_1 score are mean $\pm$ standard deviation across graphs. Runtime is mean in minutes. Dashes denote unavailable estimates. Bold and underlined values mark the best and second-best method for each q and metric.

q	Method	OD ↓	SHD ↓	F_1 score ↑	Runtime (min.) ↓
2	SCORPIO	**352.62 ± 27.01**	**727.93 ± 44.20**	**0.45 ± 0.02**	228.77
2	CAM	<u>674.10 ± 33.32</u>	1040.40 ± 47.20	<u>0.11 ± 0.02</u>	<u>52.08</u>
2	NOTEARS-linear	–	960.70 ± 43.90	0.09 ± 0.02	**5.06**
2	DAGMA	–	<u>955.92 ± 44.34</u>	0.10 ± 0.02	395.31
4	SCORPIO	**875.62 ± 48.20**	**1773.04 ± 75.45**	**0.21 ± 0.02**	224.70
4	CAM	<u>1111.30 ± 51.49</u>	2025.00 ± 60.03	<u>0.05 ± 0.01</u>	<u>55.77</u>
4	NOTEARS-linear	–	1935.10 ± 56.08	0.03 ± 0.01	**4.79**
4	DAGMA	–	<u>1931.46 ± 56.29</u>	0.04 ± 0.00	339.07

Table 7: CPDAG recovery on 10 Erdős–Rényi graphs for each expected in-degree $q \in \{2, 4\}$ at $p = 500$ and $n = 1000$. Entries are mean $\pm$ standard deviation of SHD-CPDAG across graphs. Lower is better. Dashes denote unavailable estimates. Bold and underlined values mark the best and second-best methods within each q.

q	Method	SHD-CPDAG ↓
2	SCORPIO	**817.89 ± 47.42**
2	CAF-PoNo	–
2	CAM	1055.80 ± 43.67
2	NOTEARS-linear	986.70 ± 40.61
2	DAGMA	991.77 ± 41.97
2	PC	2266.30 ± 79.84
2	LGES	<u>956.20 ± 121.59</u>
4	SCORPIO	**1841.37 ± 72.69**
4	CAF-PoNo	–
4	CAM	2034.10 ± 61.37
4	NOTEARS-linear	<u>1956.80 ± 57.52</u>
4	DAGMA	–
4	PC	3401.60 ± 106.69
4	LGES	2013.20 ± 156.97

D.4 SEMI-SYNTHETIC DATA

Semi-synthetic data provide an alternative to real-world data when the ground-truth graph is unavailable or only partially known, or when privacy constraints prevent the release of observations. The causal_assembly data generator (Göbler et al., 2024) is based on a real-world press-in and staking assembly line. It fits distributional random forests to an established partial ground-truth graph of $p = 98$ variables, thereby generating synthetic observations that reflect real-world relationships.

Table 8: DAG recovery on 20 datasets sampled from the semi-synthetic causal_assembly generator, each with $p = 98$ variables and $n = 1000$ observations. Entries are mean $\pm$ standard deviation across datasets. Lower OD, SHD, and SID and higher F_1 score are better. A dash indicates that the method does not return an order. Bold and underlined values mark the best and second-best method for each metric.

Method	OD ↓	SHD ↓	SID ↓	F_1 score ↑
SCORPIO	**206.99** $\pm$ 14.37	<u>454.28</u> $\pm$ 3.53	**4904.78** $\pm$ 77.50	0.1570 $\pm$ 0.0092
CAF-PoNo	<u>244.71</u> $\pm$ 15.55	521.72 $\pm$ 8.61	5285.39 $\pm$ 160.74	**0.1822** $\pm$ 0.0133
CAM	253.45 $\pm$ 34.18	**452.50** $\pm$ 5.05	5210.55 $\pm$ 186.98	<u>0.1711</u> $\pm$ 0.0168
NOTEARS-linear	–	464.15 $\pm$ 2.81	<u>5021.20</u> $\pm$ 105.12	0.0854 $\pm$ 0.0087
DAGMA	–	465.91 $\pm$ 3.38	5224.50 $\pm$ 73.17	0.1180 $\pm$ 0.0075

Table 9: CPDAG recovery on 20 datasets with $n = 1000$ samples each, generated from the semi-synthetic causal_assembly benchmark, which is based on a manufacturing pipeline with $p = 98$ variables. SHD-CPDAG is mean $\pm$ standard deviation across datasets. SID-CPDAG gives the mean lower and upper bounds across datasets. Lower values are better.

Method	SHD-CPDAG ↓	SID-CPDAG ↓
SCORPIO	<u>471.42 $\pm$ 3.58</u>	[4267.42, 5327.30]
CAF-PoNo	540.52 $\pm$ 7.18	[4681.76, 5688.34]
CAM	**465.45 $\pm$ 5.92**	[4642.80, 5476.60]
NOTEARS-linear	482.20 $\pm$ 1.47	[4122.40, 5342.85]
DAGMA	487.16 $\pm$ 3.66	[4192.98, 5686.04]
PC	485.35 $\pm$ 7.98	[4609.20, 5560.05]
LGES	477.15 $\pm$ 12.47	[4681.10, 5134.85]

We sample 20 datasets with $n = 1000$ samples each and evaluate non-deterministic methods five times per dataset. Tables 8 and 9 report DAG and CPDAG recovery, respectively. SCORPIO and CAM yield the most accurate DAG and CPDAG predictions. In particular, CAM achieves the lowest SHD, with an average of 1.78 fewer edge errors than SCORPIO, and the lowest SHD-CPDAG, while SCORPIO attains the lowest SID. SCORPIO also achieves the most accurate predicted topological order.

D.5 Real-World Data

Table 10 complements the DAG results of Table 2 with corresponding CPDAG performance. CAM, PC, and LGES tie for the lowest SHD-CPDAG, followed by the uniform-noise variant of SCORPIO.

Table 10: CPDAG recovery performance on the real-world dataset Sachs with $p = 11$ variables and $n = 853$ samples. Non-deterministic methods are evaluated 100 times. SHD-CPDAG is reported as mean $\pm$ standard deviation across trials. SID-CPDAG entries are the mean lower and upper bounds across trials. Lower values are better.

Method	SHD-CPDAG ↓	SID-CPDAG ↓
SCORPIO - Gaussian	16.52 $\pm$ 1.50	[55.62, 102.82]
SCORPIO - Uniform	<u>16.42 $\pm$ 1.05</u>	[52.27, 102.83]
CAF-PoNo	17.49 $\pm$ 0.73	[68.99, 103.49]
AbPNL	22.37	[66.69, 96.63]
CAM	**16.00**	[51.00, 104.00]
NOTEARS-linear	18.00	[75.00, 97.00]
DAGMA	17.17 $\pm$ 0.43	[71.63, 100.87]
PC	**16.00**	[73.00, 103.00]
LGES	**16.00**	[73.00, 103.00]

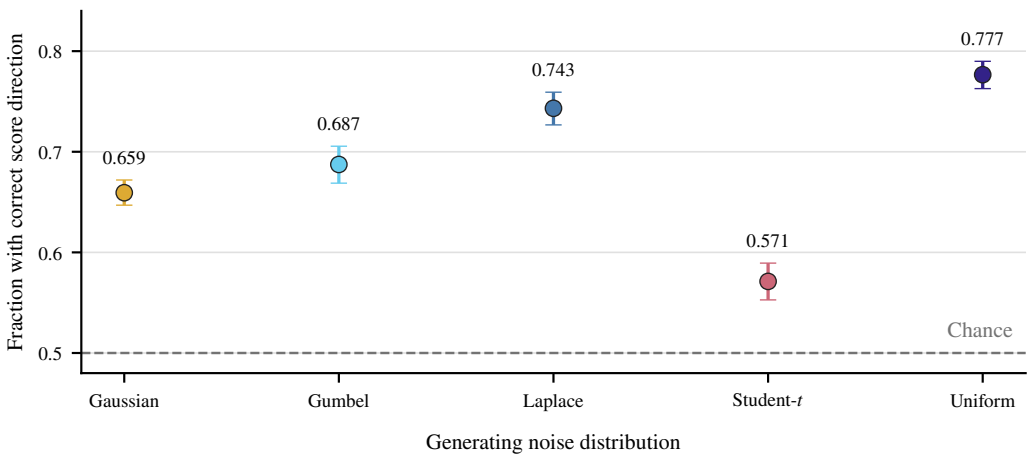

Figure 11: Fraction of ground-truth edges (i, j) for which the initial PNL likelihood scores satisfy $S[i, j] > S[j, i]$, i.e. the PNL score indicates the true edge direction. Fractions pool edge comparisons over 100 ER2 graphs and five trials per graph within each noise family. Error bars show dataset-clustered nonparametric bootstrap 95% confidence intervals based on 10,000 bootstrap samples.

E Diagnostic and Ablation Analyses

E.1 Initial PNL Score Directionality

Our PNL score in Equation 8 is outside the additive-model setting analyzed for CAM, and we do not claim an inheritance of the consistency guarantees. We therefore examine whether the initial local likelihood scores nevertheless contain directional signal. For each noise family in the benchmark of Section 5.1.1, we evaluate 100 ER2 graphs in five trials. Starting from the empty graph, we pretrain each marginal node model for at most 1000 steps. We then compute both initial scores $S[i, j]$ and $S[j, i]$ for every node pair by training each candidate parental mechanism for at most 300 steps while keeping its inverse outer transformation fixed. These scores are recorded before any edge is committed.

For every ground-truth edge, we count a correct comparison when the score in the true direction exceeds the score in the reverse direction. The reported fraction pools these edge comparisons over all graphs and trials within a noise family. The 95% confidence intervals use 10,000 nonparametric bootstrap samples at the dataset level, keeping all five trials and all edges of a sampled graph together. Figure 11 shows agreement above 65% for all noise families, except for Student-t noise, which ranks lower but above chance with 57.1%. The strongest agreement occurs on uniform noise with 77.7%. This diagnostic provides evidence that the initial score contains useful directional information. Later scores after model and in particular adjacency updates may yield different results.

E.2 Pruning Module Ablation

We isolate the effect of the pruning phase by providing SCORPIO with the true topological order and fitting our PNL parameterization for 2000 training steps on the corresponding complete DAG. We then sweep over significance levels $\alpha \in \{10^{-4}, 10^{-3}, \frac{0.05}{45}, 10^{-2}\}$, where $\frac{0.05}{45}$ is the Bonferroni threshold for a system with $p = 10$ variables. Figure 12 reports results on the parental-mechanism benchmark of Section 5.1.2. Across most complexity classes, increasing α generally reduces SHD and SID, at the cost of a modest decrease in precision. MLP mechanisms constitute the exception, where precision increases with α. The default $\alpha = 10^{-3}$ used in all experiments in the main text and the Bonferroni threshold $\frac{0.05}{45} \approx 0.0011$ yield nearly identical results.

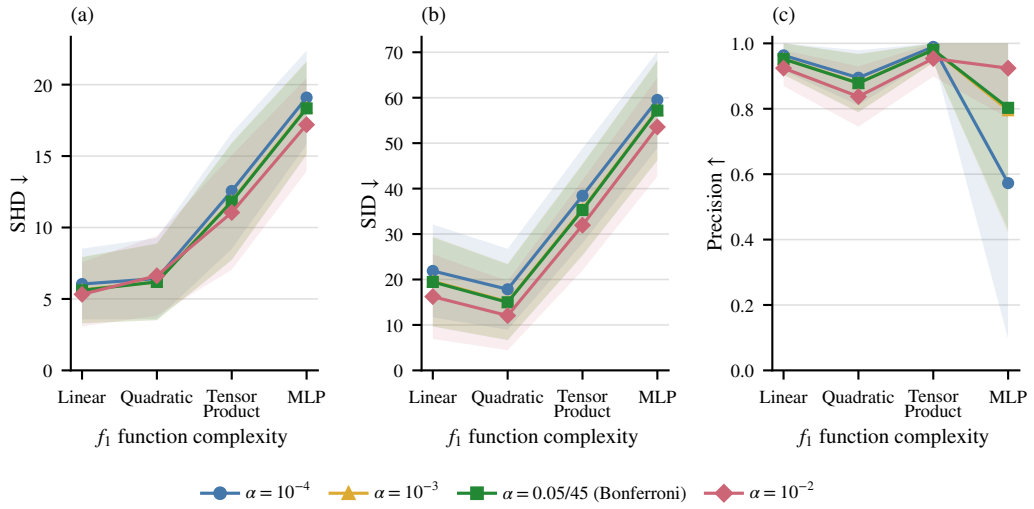

Figure 12: (a) SHD, (b) SID, and (c) precision of SCORPIO when provided with the systems' topological orders, isolating the error induced by pruning. Evaluation is performed on 100 ER2 graphs of size $p = 10$ for each parental-mechanism class from Section 5.1.2, using five trials per graph. Lines show significance levels α for the kernel independence test in pruning. Bands show mean $\pm$ one standard deviation across graph-level means after averaging trials within each graph. Lines for the default used in all experiments, $\alpha = 10^{-3}$, and the Bonferroni threshold $\alpha = \frac{0.05}{45} \approx 0.0011$ nearly overlap.

E.3 Noise Distribution Choice

While SCORPIO supports several noise families, the family must be selected before training. We therefore evaluate an end-to-end selection protocol that fits all five candidate likelihoods through order learning for each graph and trial, selects the candidate with the lowest validation negative log-likelihood (NLL), and runs pruning for the selected model. Its runtime therefore includes all five order-learning phases and the selected pruning run. We compare this protocol with a matched-family oracle that is given the generating noise family and therefore only computes one order-learning and one pruning phase.

Figure 13 reports how frequently each distribution family is selected across five trials on each of the 100 graphs per generating distribution from Section 5.1.1. The protocol selects the true generating family in 88.2%, 67.0%, 95.8%, 29.0%, and 89.8% of Gaussian, Gumbel, Laplace, Student-t, and uniform runs, respectively. Figure 14 compares the resulting structural performance with the matched-family oracle. We first average trials within each graph and then aggregate across graphs, so each graph receives equal weight.

Selection yields performance close to the matched-family oracle for Gaussian and Laplace noise, with modest degradation for Gumbel and uniform noise. The exception is Student-t noise, where the protocol selection instead improves all four reported metrics despite selecting Student-t in only 29.0% of runs, showing that minimum validation NLL is more useful here as a model-selection criterion than as a reliable identifier of the generating family. For Gaussian, Gumbel, Laplace, and uniform data, selection costs 2.45–3.13$\times$ the matched-family runtime. For Student-t data, it costs 0.94$\times$ as much (5.05 versus 5.40 minutes on average), because pruning the matched Student-t model is comparatively expensive whereas the selection protocol often chooses another family.

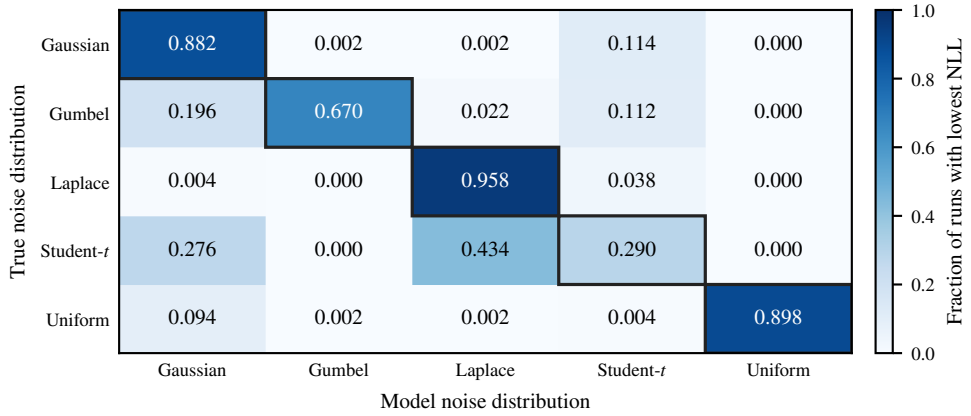

Figure 13: Noise-family selections based on minimal validation NLL after order learning. Each row corresponds to a generating noise family and reports the fraction of 500 runs (100 graphs and five trials) in which each candidate family attained the lowest validation NLL. Diagonal cells therefore show the frequency with which the selection rule recovers the true generating family.

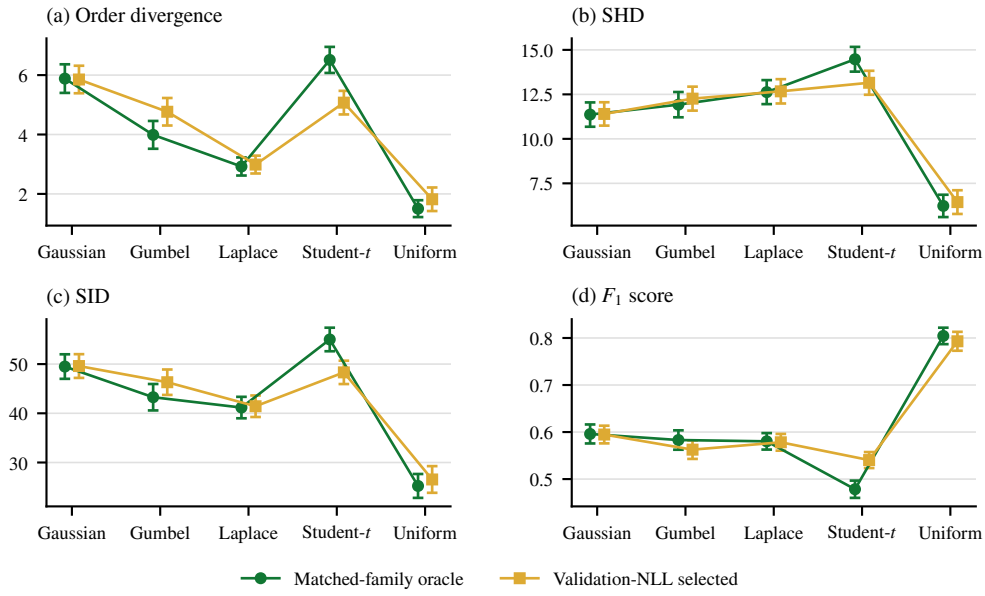

Figure 14: End-to-end performance of validation-NLL noise-family selection versus a matched-family oracle on the synthetic noise benchmark with 100 ER2 graphs: (a) order divergence, (b) SHD, (c) SID, and (d) F_1 score. Lower is better except for F_1. For each graph and trial, the selection protocol fits all five candidate likelihoods through order learning, selects the family with the lowest validation NLL, and prunes only the selected model. The matched-family oracle is given the true generating family. Points show means across graphs after averaging five trials within each graph, and error bars show 95% Student-t confidence intervals across graphs.

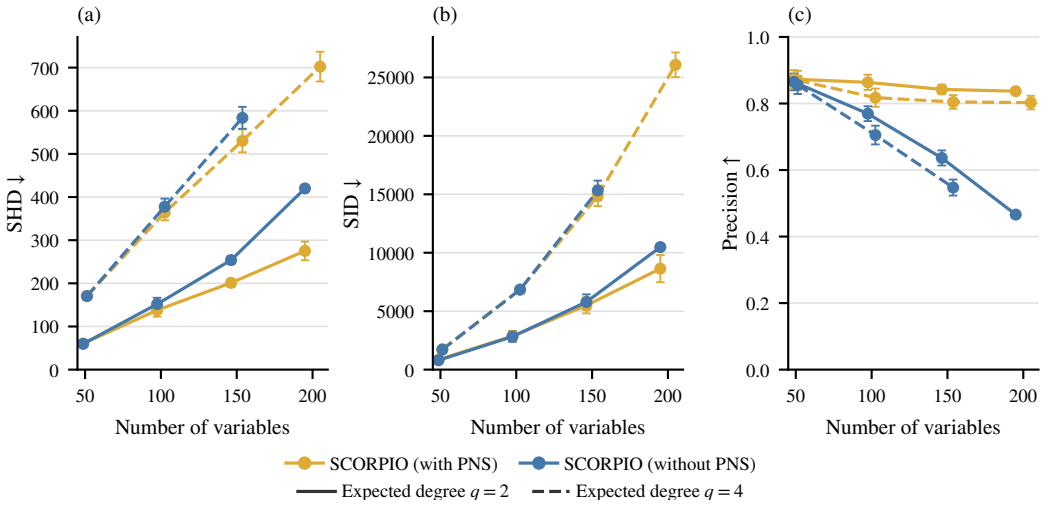

Figure 15: SCORPIO's performance with respect to (a) SHD, (b) SID, and (c) precision on the graph size and density benchmark of Section 5.1.3 ($m = 10$ ER2 and ER4 graphs of increasing size), with and without preliminary neighborhood selection (PNS) applied before the order-learning phase.

Table 11: Average runtime in minutes of SCORPIO with and without preliminary neighborhood selection (PNS) across 10 graphs with expected in-degree $q = 2$ and varying number of variables p.

p	**SCORPIO with PNS**	**SCORPIO without PNS**
50	8.99	51.73
100	21.62	269.66
150	34.88	724.66
200	48.20	1366.42

E.4 Impact of PNS

We study the effect of applying PNS as a prior screening phase in large systems in order to eliminate irrelevant edges from the set of candidate edges for the order estimation phase and hence reduce runtime. In our experiments, we enabled PNS for systems with $p \geq 50$ variables, i.e. in the graph size and density benchmark, Section 5.1.3, and in the causal_assembly manufacturing benchmark, Appendix D.4. We reevaluated the graph size and density benchmark without prior PNS application and compare DAG performance in Figure 15, as well as runtime in Table 11, to the PNS case. In these experiments, PNS improves both DAG performance and runtime, with margins increasing with problem size. In particular, PNS increases precision by reducing the number of false-positive edges.