Benchmarking Causal Discovery Under Analyst Misspecification

Edgar Davtyan

January 6, 2026

Abstract

This thesis investigates the reliability with which common causal discovery algorithms recover known causal structures from synthetic data generated by established benchmark networks. It further explores detecting mis-specification in an analyst's directed acyclic graph (DAG) through data-driven checks, alongside a survey of open-source tools for constructing and validating causal graphs. To this end, we develop a reproducible benchmarking framework applying PC, GES, NOTEARS, and COSMO to five standard networks—Asia, Sachs, ALARM, Child, and Insurance—evaluating performance via precision, recall, F_1 score, and structural Hamming distance (SHD). In controlled experiments simulating analyst error, where a true causal link is omitted or a spurious edge included, we demonstrate how conditional independence tests and bootstrap stability metrics can alert practitioners to such discrepancies. Finally, the work provides guidance on building and validating causal diagrams using contemporary software and algorithms.

Contents

List of Figures

List of Tables

List of Abbreviations

Abbrev.	Meaning
CI	Conditional independence
CPDAG	Completed partially directed acyclic graph
COSMO	Constrained Orientations by Sequential M Operation
DAG	Directed acyclic graph
FDR	False discovery rate
GES	Greedy Equivalence Search
IID	Independent and identically distributed
NOTEARS	Non-combinatorial optimisation via trace exponential and augmented lagrangian for structure learning
PC	Peter–Clark algorithm
SCM	Structural causal model
SEM	Structural equation model
SHD	Structural Hamming distance
SID	Structural intervention distance

1 Introduction

Causal diagrams—directed acyclic graphs (DAGs) that encode cause-effect relationships between variables—are indispensable for reasoning about interventions and policy decisions. These graphs consist of nodes representing variables and directed edges denoting direct causal effects. A directed graph qualifies as a DAG if it contains no directed cycles (no node can reach itself by repeatedly following arrows). When DAGs are interpreted causally, analysts typically make additional assumptions—most notably *causal sufficiency* (no unmeasured common causes of included variables) and *faithfulness* (conditional independences in the data correspond to graph separations)—that are convenient but rarely guaranteed in practice [1, 2]. When practitioners draw such diagrams incorrectly, downstream causal inference and decision-making can be compromised, motivating systematic methods to benchmark discovery algorithms and to diagnose analyst mis-specification. We operationalize analyst mis-specification as an analyst-proposed DAG that omits a true causal link or includes a non-existent link.

This thesis pursues three primary objectives. First, we benchmark multiple structure-learning methods—PC, GES, NOTEARS, and COSMO—on established benchmark networks to evaluate how well they recover known causal structures. The standardized framework ensures that comparisons are meaningful by using shared metrics and reproducible implementations. Second, we study how analyst mis-specification propagates: if a practitioner erroneously removes an edge or inserts a spurious link, can the data alert them? We design controlled experiments where the true DAG generates data but the analyst's DAG deviates from it. Third, we survey open-source tools for drawing and testing DAGs to provide practical guidance on constructing and validating causal diagrams.

These goals translate into three research questions:

RQ1 *How do causal discovery algorithms compare in recovering ground-truth network structures across different data types and network sizes?* We hypothesise that algorithm performance depends strongly on the match between algorithmic assumptions (e.g., linearity, Gaussianity) and data characteristics.

RQ2 *Can conditional independence tests reliably detect when an analyst's DAG omits a true edge or includes a spurious one?* We hypothesise that omitted edges produce significant dependence signals, while spurious edges yield non-significant results, enabling data-driven model criticism.

RQ3 *What practical guidance can we offer practitioners for selecting algorithms and validating their causal assumptions?* We synthesise our empirical findings into actionable recommendations.

The remainder of the thesis is structured as follows. Section 2 reviews the basics of causal DAGs, conditional independence (CI), and common structure-learning algorithms. Section 3 summarises related work on benchmarking causal discovery, mis-specification detection, and DAG drawing software. Section 4 details our datasets, data generation procedures, algorithms, metrics, and mis-specification protocols. Section 5 presents benchmark and sensitivity results. Finally, Section 6 discusses practical implications, limitations, and recommendations for practitioners, before Section **??** concludes.

2 Theoretical Framework

Before algorithms can be benchmarked or mis-specification detected, a rigorous mathematical framework is required. This chapter lays the probabilistic and graphical foundations for causal discovery from observational data. The concepts introduced—d-separation, the Causal Markov Condition, and Faithfulness—are not merely theoretical niceties; they directly determine what can and cannot be inferred from conditional independence patterns in data. Following these preliminaries, we examine the four algorithms central to this thesis: PC, GES, NOTEARS, and COSMO. Each embodies a different philosophy (constraint-based testing, score-based optimisation, or continuous differentiable search) and carries distinct assumptions about data and structure. We conclude by formalising the evaluation metrics—SHD, SID, precision, recall, and F_1—that will be used throughout the empirical chapters to quantify algorithm performance and assess the severity of analyst errors.

2.1 Probabilistic Graphical Models

Probabilistic graphical models (PGMs) provide a language for encoding complex multivariate dependencies via graph structures. Rather than storing a full joint distribution—exponential in the number of variables—a PGM exploits conditional independence to represent the same information compactly. The graph topology itself becomes a hypothesis about which variables directly influence which others.

Directed acyclic graphs. Formally, a graph $G = (V, E)$ comprises a vertex set $V = \{X_1, \ldots, X_d\}$ (corresponding to random variables) and a set of directed edges $E \subseteq V \times V$. An edge $(X_i, X_j) \in E$, denoted $X_i \to X_j$, asserts a direct causal or predictive relationship from X_i to X_j. Within the graph, a *path* is any sequence of distinct nodes connected by edges (ignoring direction), while a *directed path* follows edges strictly in the direction of the arrows. A *cycle* is a directed path that returns to its starting node. A graph is a Directed Acyclic Graph (DAG) if it contains no such cycles—equivalently, if nodes can be topologically ordered so that all edges point forward.

The Causal Markov condition. The Causal Markov Condition is the bridge from graph topology to probability. It states that once we condition on a variable's direct causes (its parents in the DAG), that variable becomes independent of all variables that are not its descendants. More formally, the *Local Markov Property* asserts that each X_i is conditionally independent of its non-descendants given $\text{Pa}(X_i)$. This local independence statement, applied to every node, implies a global factorisation of the joint distribution:

$$P(X_1, \ldots, X_d) = \prod_{i=1}^{d} P(X_i \mid \text{Pa}(X_i)) \tag{1}$$

Rather than specifying all $2^d - 1$ marginal and conditional distributions, we need only d local conditional distributions—one per node—each depending solely on that node's parents. This exponential compression is what makes PGMs computationally tractable.

d-Separation. While the Markov condition tells us how to compute probabilities given a graph, *d-separation* (directional separation) provides the inverse tool: reading conditional independencies

directly from the graph's topology. A path p between nodes X and Y is *blocked* by a conditioning set Z under two scenarios: first, if the path contains a chain ($i \to m \to j$) or fork ($i \leftarrow m \to j$) where the middle node m belongs to Z—informally, conditioning on a mediator or common cause "closes" the information flow; second, if the path contains a collider ($i \to m \leftarrow j$) where neither m nor any of its descendants appear in Z—colliders naturally block paths unless we condition on them or their descendants, which paradoxically "opens" the path via explaining-away reasoning. If every path between X and Y is blocked by Z, the variables are d-separated: $X \perp_G Y \mid Z$. Under the Causal Markov assumption, this graphical separation implies probabilistic independence $X \perp_P Y \mid Z$.

Faithfulness. The Markov condition is one-directional: graph separation implies statistical independence. However, the converse need not hold. A distribution might exhibit an independence that is *not* encoded in the graph—for instance, if parameters conspire so that two causal paths cancel exactly (a "knife-edge" condition). Such accidental independencies are possible but fragile: small perturbations restore the dependence. The *Faithfulness Assumption* rules out these measure-zero coincidences, asserting that the distribution P contains *only* those independencies implied by the d-separation structure of G:

$$X \perp_P Y \mid Z \iff X \perp_G Y \mid Z \tag{2}$$

This bi-directional equivalence is pivotal for causal discovery: it licenses the inference of edges (or their absence) from patterns of conditional independence observed in data. Without faithfulness, a statistical test rejecting independence would not imply the presence of an edge, undermining constraint-based discovery entirely.

Structural causal models. The *structural causal model* (SCM) framework formalises how a DAG generates data via functional relationships. Given a DAG $G = (V, E)$, an SCM specifies for each variable X_i a deterministic assignment $X_i := f_i(X_{\mathrm{Pa}(i)}, U_i)$, where f_i is an arbitrary measurable function and U_i represents unobserved exogenous noise. The variables X emerge as solutions to this system of equations, with randomness introduced only through the exogenous terms U. Provided the noise terms are jointly independent and the DAG encodes the recursive causal ordering, the Causal Markov condition follows automatically [1, 2]. This construction yields the Markov factorisation $p(x_1, \ldots, x_d) = \prod_{i=1}^{d} p(x_i \mid x_{\mathrm{Pa}(i)})$ (Equation 5), which serves as the computational backbone for both score-based and constraint-based discovery: score-based methods maximise factorised likelihoods, while constraint-based methods test the conditional independencies that the factorisation entails.

From the SCM to a factorised likelihood (proof sketch). Assume G admits a topological ordering of variables such that parents precede children. By the chain rule, any joint distribution can be written as

$$p(x_1, \ldots, x_d) \;=\; \prod_{i=1}^{d} p(x_i \mid x_{1:i-1}). \tag{3}$$

Under the Markov property implied by the SCM graph, each X_i is independent of its non-descendants given its parents. In particular, conditioning on all previous variables $X_{1:i-1}$ is equivalent to conditioning only on $X_{\mathrm{Pa}(i)}$, because the remaining variables in $\{1, \ldots, i-1\} \setminus \mathrm{Pa}(i)$ are non-descendants of i.

Therefore,

$$p(x_i \mid x_{1:i-1}) \;=\; p(x_i \mid x_{\mathrm{Pa}(i)}), \tag{4}$$

and we obtain the *Markov factorisation*

$$p(x_1, \ldots, x_d) \;=\; \prod_{i=1}^{d} p(x_i \mid x_{\mathrm{Pa}(i)}). \tag{5}$$

This identity is the key bridge between graphs and data: (5) is exactly the decomposition exploited by both score-based methods (which optimise a factorised likelihood or a penalised variant) and constraint-based methods (which test conditional independencies implied by the factorisation) [2, 3].

d-Separation in practice. The preceding definitions become concrete when applied to real benchmark networks. Understanding how d-separation works in practice is essential for interpreting algorithm behaviour: constraint-based methods like PC systematically query these independencies, while score-based methods implicitly encode them through the factorised likelihood. Table 2 demonstrates several representative queries in the Asia network. The first row illustrates collider blocking: `Smoking` and `Tuberculosis` are marginally independent because their only connecting path is blocked by the unobserved collider `Either`. The second row shows the paradoxical "explaining away" effect—conditioning on `Either` *opens* the blocked path, inducing dependence between two variables that were previously independent.

Table 2: Worked examples of d-separation in the Asia network (Figure 4).

Query (graph statement)	d-separated?	Reasoning (path-wise)
`Smoking` $\perp$ `Tuberculosis`	Yes	The main connecting path `Smoking` $\rightarrow$ `LungCancer` $\rightarrow$ `Either` $\leftarrow$ `Tuberculosis` is blocked by the collider at `Either`. No other open path exists without conditioning.
`Smoking` $\perp$ `Tuberculosis` $\mid$ `Either`	No	Conditioning on the collider `Either` *opens* the previously blocked path, producing the classic "explaining-away" dependence.
`Smoking` $\perp$ `Xray` $\mid$ `Either`	Yes	`Xray` is a child of `Either`. Once `Either` is conditioned on, the path `Smoking` $\rightarrow$ `LungCancer` $\rightarrow$ `Either` $\rightarrow$ `Xray` is blocked at `Either` (a conditioned non-collider).
`Tuberculosis` $\perp$ `Bronchitis`	Yes	Two prominent paths are blocked: (i) `Tuberculosis` $\rightarrow$ `Either` $\rightarrow$ `Dyspnea` $\leftarrow$ `Bronchitis` is blocked by the collider at `Dyspnea`; (ii) the path through `Either` is blocked as above.

These concrete queries illustrate the central mechanism behind constraint-based learning: algorithms such as PC recover a skeleton by repeatedly testing whether statements like those in Table 2 are supported by the data (for various conditioning sets), and then orient edges to form an equivalence-class representation. The Asia network is small enough for manual verification, but real-world applications involve dozens or hundreds of variables, where exhaustive path enumeration becomes infeasible and algorithmic search is necessary.

Consider the larger ALARM network [32] (37 nodes, 46 edges), which models intensive-care patient monitoring. For `Pulmonary Embolism` (PE) and `Central Venous Pressure` (CVP), the relationship is mediated by `Pulmonary Artery Pressure` (PAP): the chain `PE` $\rightarrow$ `PAP` $\rightarrow$ `CVP` renders them marginally dependent, yet conditioning on PAP blocks the only connecting path ($PE \perp CVP \mid PAP$). This pattern—marginal dependence becoming conditional independence—is the signature of a mediating variable. Conversely, `Catecholamine` and `Hypovolemia` are marginally independent (no direct path) but connected via the collider at `Heart Rate` (HR); conditioning on HR induces dependence through the explaining-away mechanism.

The Child network [**?**], designed for diagnosing congenital heart disease ("blue baby" syndrome), presents similar challenges. `Birth Asphyxia` and `Age` are marginally independent (both are root causes with no connecting path), yet conditioning on their common descendant `Disease`—or any of its observable proxies—can induce a spurious association. This underscores a general principle: colliders and their descendants must be handled with care, as naive conditioning can introduce bias rather than remove it.

Identifiability and theoretical guarantees. A fundamental limit of observational causal discovery is *Markov equivalence*: multiple distinct DAGs can imply identical conditional independence structures, making them indistinguishable from observational data alone. For instance, the three graphs $A \rightarrow B \rightarrow C$, $A \leftarrow B \leftarrow C$, and $A \leftarrow B \rightarrow C$ all encode the single conditional independence $A \perp C \mid B$ and no others. Verma and Pearl [30] characterised this phenomenon precisely: two DAGs belong to the same Markov Equivalence Class (MEC) if and only if they share the same skeleton (undirected graph) and the same set of v-structures (unshielded colliders $X \rightarrow Z \leftarrow Y$). Such classes are compactly represented by a Completed Partially Directed Acyclic Graph (CPDAG), where directed edges appear in all members of the class and undirected edges denote ambiguous orientations. This is not a failure of the algorithms—it is an inherent limitation of passive observation.

Algorithm guarantees must therefore be stated relative to the CPDAG, not any single DAG. **Constraint-based methods** (like PC) assume faithfulness and, in the infinite-sample limit with perfect conditional independence tests, recover the true CPDAG [2]. Finite-sample consistency requires that the CI test error rates vanish appropriately; Kalisch and Bühlmann [**?**] establish such results for high-dimensional Gaussian graphs under suitable sparsity and faithfulness conditions. **Score-based methods** (like GES) search for the graph maximising a penalised likelihood score (e.g., BIC). Chickering [3] proved that GES identifies the correct CPDAG in the large-sample limit when the score is consistent and score-equivalent. In finite samples, the two families exhibit complementary failure modes: PC is brittle to individual test errors that propagate through the skeleton construction, whereas GES is more robust to noise but sensitive to model mis-specification (e.g., applying BIC's linear-Gaussian assumptions to discrete data). Interventional data—observations under do$(Y = y)$ manipulations—can break Markov equivalence by revealing asymmetries in the causal mechanisms, but this thesis focuses on the observational regime where the CPDAG is the identifiability ceiling.

2.2 Constraint-Based Discovery: The PC Algorithm

The PC algorithm (Peter–Clark) is the archetypal constraint-based method. Named after Peter Spirtes and Clark Glymour, it reconstructs the CPDAG by systematically testing conditional independence

statements implied by faithfulness [2]. The algorithm's elegance lies in its two-phase design: first learn which variables are adjacent (the skeleton), then determine which edges can be oriented.

Procedure. *Phase 1: Skeleton identification.* Starting from a complete undirected graph, PC tests whether each pair (X, Y) is conditionally independent given subsets S of their neighbours. Independence tests begin with the empty set ($|S| = 0$), then condition on single variables ($|S| = 1$), pairs ($|S| = 2$), and so forth, up to the maximum degree. Whenever a test finds $X \perp Y \mid S$, the edge is removed and S is recorded as the separating set. Because we assume faithfulness, every non-adjacent pair in the true graph has a separating set, and every adjacent pair lacks one—so in the large-sample limit with perfect tests, this phase recovers the skeleton exactly. *Phase 2: Orientation.* With the skeleton fixed, PC identifies v-structures: for every triple $X - Z - Y$ where X and Y are non-adjacent, if Z was not in their separating set S_{XY}, then Z must be a collider ($X \to Z \leftarrow Y$). These v-structures are oriented first; then Meek's orientation rules propagate the constraints to avoid creating new v-structures or cycles, orienting as many additional edges as possible while respecting the equivalence class.

Complexity and stability. The number of potential conditioning sets grows exponentially in the graph's maximum degree, making worst-case complexity prohibitive for dense graphs. However, real-world causal networks are often sparse (average degree $\ll d$), rendering PC practical. A more insidious issue is **order dependence**: in finite samples, ties or marginal test failures can depend on the order in which variables are processed, yielding different skeletons. We employ the PC-Stable variant [16], which ensures order-independence by fixing the candidate conditioning sets (the adjacency sets at the start of each round) rather than updating them dynamically.

2.3 Score-Based Discovery: GES

Score-based methods recast structure learning as model selection: find the graph G that optimises a balance between fit and complexity. Rather than testing individual conditional independencies, these methods evaluate candidate graphs holistically via a scoring function $S(G, D)$.

Scoring functions. We employ the Bayesian Information Criterion (BIC) [26], a penalised likelihood criterion:

$$\mathrm{BIC}(G; D) = \log L(G; \hat{\theta}) - \frac{k_G}{2} \log n \tag{6}$$

where $\log L(G; \hat{\theta})$ measures fit (higher is better), k_G counts free parameters, and n is the sample size. The penalty $\frac{k_G}{2} \log n$ grows with model complexity, discouraging overfitting. Two properties make BIC attractive for causal discovery. First, it is *decomposable*: for a DAG, the likelihood factorises over nodes (Equation 5), so adding or removing edges changes only the local terms for affected variables. This locality enables incremental score updates. Second, BIC is *score-equivalent*: Markov equivalent DAGs receive identical scores, ensuring the search respects equivalence classes. For linear-Gaussian models, the local likelihood term for node i reduces to a penalised residual sum of squares from regressing X_i on its parents.

Greedy Equivalence Search (GES). Exhaustive search over all DAGs is super-exponential ($d! \cdot 2^{d(d-1)/2}$ structures for d nodes). GES [3] sidesteps this by searching the space of equivalence classes (CPDAGs) via a two-phase greedy heuristic. *Forward Equivalence Search* (FES) begins with an empty graph and iteratively adds the single edge (or edge orientation) that most improves the score, continuing until no addition yields a gain. *Backward Equivalence Search* (BES) then removes edges one at a time to further refine the score. This two-phase design exploits a key insight: starting sparse and adding edges (forward) avoids early commitment to spurious structures, while pruning (backward) corrects overfitting from the greedy forward pass. Chickering proved that in the large-sample limit—where the score becomes an accurate proxy for the true generative model—GES identifies the correct CPDAG. In our experiments, we apply BIC uniformly across all datasets (including discretised data) to stress-test its robustness to model mis-specification.

2.4 Continuous Optimisation: NOTEARS

Traditional search-based methods (PC, GES) navigate discrete graph spaces via edge additions, deletions, or reversals. NOTEARS (Non-combinatorial Optimisation via Trace Exponential and Augmented lagRangian for Structure learning) [19] takes a radically different approach: formulate structure learning as a *continuous* optimisation problem over weighted adjacency matrices, where the acyclicity constraint—previously thought to be inherently combinatorial—is expressed as a smooth, differentiable equality constraint.

Differentiable acyclicity. The key innovation is a function $h : \mathbb{R}^{d\times d} \to \mathbb{R}$ such that $h(W) = 0$ if and only if the graph represented by W is acyclic. Define $h(W) = \text{tr}(e^{W\circ W}) - d$, where $W \circ W$ is the element-wise square of W and e^A is the matrix exponential. Intuitively, the matrix exponential expands as $\sum_{k=0}^{\infty} \frac{(W\circ W)^k}{k!}$; the trace sums the diagonal, aggregating self-loop counts weighted by path lengths. If the graph contains a cycle, some diagonal entries explode; if acyclic, the trace reduces to exactly d (one per node). The gradient $\nabla_W h(W) = 2W \circ (e^{W\circ W})^\top$ is computable in closed form, enabling standard gradient-based solvers.

Objective and limitations. NOTEARS minimises a regularised loss subject to the acyclicity equality constraint:

$$\min_W \ell(W; X) + \lambda\|W\|_1 \quad \text{subject to} \quad h(W) = 0 \tag{7}$$

where $\ell(W; X)$ is typically the squared-error loss for linear structural equation models, and $\lambda\|W\|_1$ penalises edge proliferation (encouraging sparsity). An augmented Lagrangian solver citeNocedal2006 alternates between minimising $ell +$ $lambda$ $|W$ $|_1 +$ $fracrho2h(W)^2 +$ $alphah(W)$ (where rho and

$alpha$ are penalty/multiplier parameters) and updating the multipliers, gradually enforcing $h(W)$
$to0$.

textbfCritical assumption: The original NOTEARS formulation assumes linear structural equation models with continuous variables. When applied to discrete or mixed data (as in our benchmarks), it treats categorical state values as if they were continuous, constituting a
emphmodel mis-specification. Despite this mismatch, NOTEARS often performs surprisingly well in practice, suggesting robustness to moderate violations. We include it specifically to evaluate how continuous-optimisation methods behave under such mis-specified conditions, providing a contrast to methods explicitly designed for discrete data.

2.5 Regression-Based Discovery: COSMO

Massidda et al.
citeMassidda2023 introduce COSMO (Causal Ordering via Scalable Modelling), which reduces computational complexity to $O(d^2)$ by replacing the explicit acyclicity constraint with a
emphlearned causal ordering. Rather than searching over graph structures, COSMO learns a latent order
$Theta = ($
$theta_1,$
$ldots,$
$theta_d)$ where
$theta_i$
in
$mathbbR$ specifies node i's position. Edges X_i
toX_j are permitted only if
$theta_i <$
$theta_j$, automatically ensuring acyclicity.

paragraphMechanism. COSMO alternates between two steps: an
textbfM-Step fixes the ordering
$Theta$ and estimates each node's parents via regularised regression (typically Lasso or elastic net), selecting the subset of predecessors in the ordering that best predict the node's values; and an
textbfO-Step updates
$Theta$ to minimise the overall prediction loss, shifting nodes forward or backward in the order to reduce residual variance. This iterative refinement continues until convergence. Because the ordering is represented by continuous parameters, both steps can exploit efficient regression solvers, avoiding the combinatorial search over graph topologies.

To ensure reliability in finite samples, COSMO employs
textbfstability selection
citeMeinshausen2010: the full procedure is run on multiple bootstrap resamples, and only edges that appear with high frequency (e.g., $> 70\%$) across runs are retained in the final output. This guards against spurious edges that arise from sampling variability, at the cost of potentially missing weak but genuine relationships.

2.6 Evaluation Metrics

Evaluating causal discovery algorithms requires a multi-faceted approach. No single metric captures all aspects of performance: structural similarity measures (like SHD) assess graph topology but ignore causal semantics, while intervention-based metrics (like SID) reflect downstream reasoning tasks but are sensitive to specific query distributions. We therefore employ a portfolio of complementary metrics, each illuminating a different facet of algorithm behaviour.

Structural Hamming Distance (SHD). The most widely used metric is SHD, which counts the minimum number of edge operations (additions, deletions, or reversals) required to transform the learned graph $\hat{G}$ into the true graph G [17]:

$$\text{SHD}(G, \hat{G}) = |E \setminus \hat{E}| + |\hat{E} \setminus E| + \text{flips} \tag{8}$$

This metric is intuitive and directly reflects structural fidelity. However, validating against a *DAG* rather than its Markov equivalence class is problematic: an algorithm that correctly returns an undirected edge $X - Y$ (because the direction is unidentifiable) would be penalised for not guessing the "true" orientation. To avoid this pitfall, we employ a **CPDAG-aware SHD**: both G and $\hat{G}$ are first converted to their respective CPDAGs, then compared. Undirected edges in the CPDAG are treated as bidirectional, so an algorithm receives credit for correctly leaving an edge unoriented. This ensures that the metric respects the identifiability ceiling imposed by observational data.

Precision, recall, and F1. SHD aggregates errors into a single number, obscuring the balance between false positives (spurious edges) and false negatives (missing edges). To disentangle these failure modes, we compute standard classification metrics: Precision ($P = TP/(TP + FP)$) measures what fraction of predicted edges are correct, Recall ($R = TP/(TP + FN)$) measures what fraction of true edges are recovered, and the F_1 score ($2PR/(P + R)$) provides their harmonic mean. These are reported separately for the **skeleton** (undirected adjacency, ignoring orientation) and the **directed graph** (requiring correct orientation). Large discrepancies between skeleton and directed scores— for instance, high skeleton recall but low directed recall—indicate that the algorithm successfully identifies which pairs of variables are adjacent but struggles to orient those edges. This diagnostic capability is particularly valuable when interpreting algorithm behaviour: constraint-based methods may excel at skeleton recovery yet falter at orientation due to finite-sample test errors, while score-based methods might confidently orient edges but add spurious connections.

Structural Intervention Distance (SID). While SHD treats all edge errors equally, the *Structural Intervention Distance* (SID) [25] weights errors by their impact on causal reasoning. Formally, SID counts the number of ordered pairs (X, Y) for which the interventional distribution $P(Y \mid \text{do}(X))$ inferred from the learned graph $\hat{G}$ differs from the true interventional effect implied by G. In other words, two graphs have zero SID if and only if they induce identical interventional predictions for all possible pairs of variables. This captures the intuition that some edge errors are more consequential than others: reversing an edge in a long causal chain can corrupt many downstream interventional queries, whereas adding a redundant edge between already d-connected nodes may have minimal

impact. SID thus provides a *task-aligned* evaluation metric, directly reflecting the downstream utility of the learned structure for interventional reasoning. In our experiments, we compute SID by enumerating all (X, Y) pairs and checking whether the adjustment sets required to identify $P(Y \mid \text{do}(X))$ differ between G and $\hat{G}$.

2.7 Structural Causal Models and Interventions

While structure learning algorithms focus on recovering the graph G, the ultimate goal of causal inference often involves predicting the outcome of interventions: what would happen to Y if we forcibly set $X = x$, overruling its natural causes? The Structural Causal Model (SCM) framework formalises this via *do*-calculus [1]. An intervention $\text{do}(X_k = x)$ is represented by replacing the structural equation $X_k = f_k(\text{Pa}_k, U_k)$ with the constant assignment $X_k = x$, effectively severing the causal influence of X_k's parents. The resulting mutilated graph—and its implied factorisation—determines the post-intervention distribution $P(\mathbf{X} \mid \text{do}(X_k = x))$.

Pearl's do-calculus provides syntactic rules for determining when such interventional quantities can be identified from purely observational data, using properties of the graph (d-separation) to guide adjustment strategies. Crucially, these identifiability results assume knowledge of the *true* causal graph G. If the learned structure $\hat{G}$ contains errors, the resulting interventional predictions can be arbitrarily wrong—a missing edge may omit a necessary confounder adjustment, while a reversed edge can invert the causal direction entirely. Thus, verifying the correctness of the learned graph is a prerequisite for valid downstream causal inference. Our benchmark evaluates how reliably algorithms recover structures that would support correct interventional reasoning.

2.8 Assumptions, Identifiability, and Scope

All structure learning algorithms rest on a tower of assumptions, violations of which can silently corrupt the learned graph. We can categorise these into three layers. **Graphical assumptions** include the Causal Markov Condition (graph structure implies conditional independencies), Faithfulness (no accidental cancellations create additional independencies), and Causal Sufficiency (no unobserved confounders). Violations are common in practice: latent confounders introduce spurious associations, while finely tuned parameters can produce measure-zero faithfulness violations. **Statistical assumptions** concern the validity of independence tests or scoring functions given the sample size and data distribution. For instance, Fisher's z-test assumes joint Gaussianity; applying it to heavily skewed or discrete data inflates Type I/II error rates. **Model class assumptions** impose structural constraints on the functional form—NOTEARS assumes linear additive noise models, GES implicitly assumes the data are well-described by the parametric family used in the BIC calculation.

Our experiments use *semi-synthetic* data generated from known Bayesian network structures, allowing us to control graphical assumptions (no latent confounders, faithfulness guaranteed by construction) while varying sample size and introducing analyst mis-specifications (e.g., applying continuous-data methods to discretised observations). Real-world violations—particularly latent confounding—remain a critical challenge discussed in Section 6.5.

2.9 Conditional Independence Testing

Conditional independence (CI) tests serve a dual role in our framework. First, they are the fundamental primitive for constraint-based discovery: algorithms like PC repeatedly query whether $X \perp Y \mid S$ to construct the skeleton and identify v-structures. Second, they function as a tool for *model criticism*, allowing us to validate whether a learned graph's implied independencies hold in the data.

For continuous (or Gaussian-assumed) data, we employ **Fisher's z-test** on partial correlations: the null hypothesis $X \perp Y \mid S$ is rejected if $z = \frac{1}{2} \log \frac{1+\hat{\rho}_{XY \cdot S}}{1-\hat{\rho}_{XY \cdot S}} \sqrt{n - |S| - 3}$ exceeds the critical value for significance level α (typically 0.05). For discrete data, we use the G^2 **(log-likelihood ratio) test**, which compares observed and expected cell counts under the independence hypothesis. Both tests balance Type I errors (falsely rejecting independence, leading to spurious edges) against Type II errors (failing to detect dependence, leading to missing edges). The standard choice $\alpha = 0.05$ reflects this trade-off, though the optimal threshold is data-dependent.

A critical challenge is **error propagation** in multi-test algorithms like PC. The skeleton phase may perform thousands of CI tests, and each erroneous decision contaminates subsequent tests: if an edge is incorrectly removed, it will not be considered in later conditioning sets, potentially blocking the discovery of true conditional independencies. This cascading failure mode explains why PC's performance can degrade sharply in noisy or small-sample regimes, even though individual tests remain calibrated.

2.10 Statistical Tests for Comparing Algorithms

To rigorously compare algorithms across multiple datasets and establish whether observed performance differences are statistically meaningful, we follow the recommendations of Demšar [24] for multi-classifier comparison. Standard parametric tests (e.g., paired t-tests) make distributional assumptions (normality, independence) that are often violated when comparing algorithms on a small number of heterogeneous datasets. Demšar advocates non-parametric alternatives that rank algorithms on each dataset, then test for differences in average ranks.

We employ the **Friedman test** as an omnibus procedure to detect whether any algorithms differ significantly. If the null hypothesis of equal mean ranks is rejected, we proceed to pairwise comparisons via the **Nemenyi post-hoc test**, which controls the family-wise error rate. Results are visualised using **Critical Difference (CD) diagrams**: algorithms are arranged along a horizontal axis by their average rank, and groups of algorithms whose ranks do not differ significantly (at $\alpha = 0.05$) are connected by a horizontal bar. This provides an intuitive summary of the performance hierarchy.

Important caveat: Given our small number of datasets ($N = 5$), these tests have limited statistical power. A non-significant result does not prove equivalence; it may simply reflect insufficient data. We therefore treat the Friedman/Nemenyi results as *descriptive complements* to the raw performance metrics, useful for identifying clear winners but not definitive for marginal differences.

Robustness considerations. Beyond central tendency (mean or median SHD), robustness to sample size and hyperparameter choices can be assessed through systematic variation. Repeating benchmarks across a range of n values (e.g., 500, 1000, 5000, 10000) would reveal *asymptotic*

behaviour: do algorithms converge toward the correct structure as $n \to \infty$, or do systematic biases persist? Similarly, adjusting algorithm-specific thresholds—the λ sparsity penalty in NOTEARS, the significance level α in PC—could expose the *stability* of the learned structures. An algorithm that requires precise hyperparameter tuning to avoid catastrophic failures is less deployable in practice than one with a wide "plateau" of good performance. We report variance across bootstrap resamples (for SHD, F1, etc.) to provide a fuller picture of reliability beyond point estimates.

3 Related Work

Causal discovery has evolved over three decades from a niche preoccupation of philosophers and statisticians into a central pillar of modern machine learning and epidemiology. What began as a theoretical debate over whether causation could ever be inferred from passive observation has matured into a flourishing ecosystem of algorithms, software packages, and empirical benchmarks. This chapter traces that intellectual arc, beginning with the foundational formalisms of Pearl and Spirtes that established the field's conceptual infrastructure. We then survey critical benchmarking studies—both celebratory and cautionary—that motivate the design of our own evaluation framework. Finally, we examine the growing literature on model validation and mis-specification diagnostics, which provides the methodological toolkit for assessing when a learned graph is "good enough" for downstream causal inference.

3.1 Foundations of Causal Discovery

The rigorous formalisation of causal discovery emerged from the convergence of two parallel intellectual traditions in the 1990s: Judea Pearl's framework of probabilistic graphical models, which provided a semantics for causal claims, and the constraint-based search methods of Spirtes, Glymour, and Scheines, which demonstrated that those semantics could be operationalised into practical discovery algorithms. Together, these contributions transformed causal inference from a philosophical puzzle into an engineering discipline.

3.1.1 The Graphical Viewpoint

Pearl's seminal work
citePearl1995 introduced the Directed Acyclic Graph (DAG) as a rigorous mathematical language for representing causal relationships, elevating informal path diagrams—used since Sewall Wright's pioneering work on path analysis in genetics—into a formal calculus. Central to this framework is the emphd-separation criterion, which forges a precise connection between the topological structure of a graph and the conditional independence structure of the probability distribution it represents. This bridge between graph theory and probability theory is what makes causal claims empirically testable: if a proposed DAG implies X
$perpY$
$midZ$ but the data exhibit strong dependence, the graph is falsified.

Pearl's

emphLadder of Causation distinguishes three levels of causal reasoning, each requiring progressively stronger inferential machinery. At the first rung,
emphassociation (“seeing”), we observe correlations and dependencies in passive data. At the second rung,
emphintervention (“doing”), we predict the effects of external manipulations that override natural causal mechanisms. At the third rung,
emphcounterfactuals (“imagining”), we reason about what would have happened under alternative scenarios contrary to observed fact. Structure learning operates primarily at the first level—extracting patterns of association to infer the underlying graph—in service of enabling reasoning at the second level, where interventional predictions guide policy and decision-making.

3.1.2 The Search and Score Viewpoint

While Pearl focused on the semantics of causal graphs—formalising what they mean and how to reason with them—Spirtes, Glymour, and Scheines
citeSpirtes2000 addressed the complementary challenge of discovering them from data. Their PC algorithm demonstrated a profound result: under assumptions of
emphCausal Markovness (graph structure implies conditional independencies) and
emphFaithfulness (no accidental cancellations), it is possible to recover the true causal structure—up to a Markov equivalence class—from purely observational data, provided the sample size is sufficiently large and independence tests are well-calibrated.

This work fundamentally challenged the prevailing dogma that “correlation does not imply causation.” While a single correlation is indeed insufficient to establish causality,
emphpatterns of conditional independence across many variables can reveal causal directionality. The key insight is that certain graph structures—particularly v-structures (colliders)—imply asymmetric independence patterns that are preserved under no orientation reversal. By systematically testing these patterns, constraint-based algorithms can distinguish, for example, the common-cause fork X
$leftarrow Z$
$to Y$ from the chain X
$to Z$
$to Y$, even though both imply marginal dependence between X and Y. The distinction emerges only when conditioning: the fork renders X and Y independent given Z, whereas the chain does so as well, but the collider X
$to Z$
$leftarrow Y$ requires
emphnot conditioning on Z to observe independence. This logical signature is what makes causal discovery from observational data possible, rather than merely aspirational.

3.1.3 A brief historical arc of structure learning paradigms

While the core assumptions underpinning causal discovery—Markov faithfulness, causal sufficiency, and appropriate functional form—have remained conceptually stable, the
emphalgorithmic landscape has expanded dramatically. A useful organising principle is the dominant

source of information each paradigm exploits to infer edges and orientations.

textbfConstraint-based discovery, exemplified by the PC algorithm and its extensions, infers both the skeleton (which variables are adjacent) and partial orientations by systematically testing conditional independence statements implied by candidate graphs. The FCI algorithm extends this paradigm to settings with latent confounding, where unobserved common causes require weaker graphical representations (Partial Ancestral Graphs)
citeSpirtes2000,Colombo2014. These methods are philosophically elegant—they make minimal parametric assumptions—but are brittle to finite-sample errors in independence tests, where cascading failures can corrupt the entire learned structure.

textbfScore-based discovery reframes structure learning as model selection: search for the graph G that optimises a decomposable score balancing fit against complexity. The GES algorithm is a canonical example, searching over Markov equivalence classes (CPDAGs) under the principles of score equivalence (Markov equivalent graphs receive identical scores) and decomposability (local changes affect only local score terms)
citeChickering2002. Score-based methods are more robust to individual test errors than constraint-based approaches, but they require strong parametric assumptions (e.g., Gaussian likelihoods for BIC) and can be computationally expensive for large graphs.

textbfHybrid discovery attempts to marry the best of both worlds by using constraint-based pruning to identify candidate parent sets, then refining the structure via score-based optimisation. The MMHC algorithm (Max-Min Hill-Climbing) is a widely used representative, demonstrating that strategic combination of the two paradigms can outperform either in isolation
citeTsamardinos2006.

textbfFunctional and distributional discovery exploits stronger assumptions about the data-generating process to break symmetries that are unidentifiable under purely independence-based reasoning. The LiNGAM family (Linear Non-Gaussian Acyclic Model) is a classical example: by assuming linear relationships with non-Gaussian noise, these methods can uniquely identify edge directions that would otherwise form an equivalence class under Gaussian assumptions
citeShimizu2006. The trade-off is obvious—stronger assumptions yield sharper identifiability, but violations are more consequential.

textbfOptimisation- and learning-based discovery emerged in the late 2010s, recasting DAG learning as continuous optimisation by relaxing the discrete, combinatorial acyclicity constraint into a differentiable equality. NOTEARS pioneered this approach, enabling gradient-based solvers and integration with modern deep learning frameworks
citeZheng2018. Subsequent work introduced neural parameterisations (GraN-DAG
citeLachapelle2020), reinforcement learning for graph search
citeZhu2020, and stability-focused heuristics like COSMO
citeMassidda2023. These methods promise scalability and flexibility but introduce new hyperparameters (penalty strengths, learning rates) and optimisation instabilities that can be opaque to practitioners.

Table 3: A sketch timeline of representative developments in causal structure learning. This is not exhaustive; it highlights the conceptual shift from combinatorial search and CI testing to differentiable and learning-based approaches.

Period	Representative developments
1990s	Constraint-based discovery formalised and popularised (e.g., PC/FCI) under Markov and faithfulness assumptions [2].
2000s	Score-based and hybrid methods mature (e.g., GES, MMHC), with practical software implementations becoming widely used [3, 17].
2010s	Tooling and benchmarking proliferate (e.g., `bnlearn`, `pcalg`), bringing attention to robustness and evaluation design [12, 11].
2018–present	Differentiable and learning-based approaches accelerate (NOTEARS, GOLEM, GraN-DAG, RL-based search) and new methods explicitly target practical fragilities such as orientation instability [19, 20, 21, 22, 23].

3.1.4 Algorithm families, assumptions, and outputs

Table 4 provides a cross-sectional view of representative methods spanning the major algorithmic families. Rather than cataloguing all possible techniques, the table emphasises assumptions that most directly interact with analyst mis-specification—the central concern of this thesis. Constraint-based methods depend critically on the calibration of conditional independence tests: if the analyst applies Fisher's z-test to non-Gaussian data, the resulting Type I and Type II errors corrupt skeleton construction. Score-based methods require that the score function's parametric assumptions align with the data: applying BIC with a Gaussian likelihood to discrete variables constitutes a fundamental mismatch. Functional methods (like LiNGAM) trade identifiability gains for stronger structural assumptions, amplifying the consequences of violations. Understanding these failure modes is essential for interpreting benchmark results and guiding practitioners toward robust workflows.

3.1.5 Positioning of this thesis within the literature

Recent critiques of benchmarking practice—particularly the influential work of Reisach et al. [6] and the systematic reviews by Scutari and colleagues [5]—have revealed that headline accuracy figures can be inflated by favourable synthetic generators (e.g., variance sortability), metric choices that ignore Markov equivalence classes (penalising algorithms for correctly leaving edges unoriented), or inadvertent leakage of ground-truth structure into hyperparameter tuning. These methodological pitfalls undermine confidence in comparative conclusions.

This thesis positions itself at the intersection of *rigorous method comparison* and *workflow robustness*. Rather than proposing yet another discovery algorithm, we empirically characterise how established families—spanning the constraint-based, score-based, and optimisation-based paradigms—behave under a controlled, realistic form of analyst mis-specification: the imposition of domain priors (edge additions or deletions) that may conflict with the data. By coupling benchmark performance with sensitivity analyses (bootstrap stability, hyperparameter sweeps) and validation diagnostics (conditional independence testing of learned graphs), we aim to provide practitioner-facing guidance grounded in transparent, reproducible evaluation.

Table 4: Representative causal discovery methods and the assumptions most relevant to misspecification robustness. "Output" refers to the standard graphical object returned when run on observational data.

Method	Family	Typical assumptions (most salient)	Output and practical notes
PC [2]	Constraint	Markov + faithfulness; CI test is well-calibrated for the data type; sufficient n	CPDAG (equivalence class). Sensitive to CI-test misspecification and multiple testing.
PC-Stable [16]	Constraint	As PC, but designed to reduce order-dependence	CPDAG. Often a safer default than naive PC in software implementations.
FCI [2]	Constraint	Allows latent confounding under additional assumptions; more complex independence logic	PAG. Evaluation must respect the weaker target object (not a DAG).
GES [3]	Score	Decomposable, score-equivalent score; score model matches data; sufficient n	CPDAG. Can be accurate but computationally heavy in larger graphs.
MMHC [17]	Hybrid	Reliable CI-based neighbourhood discovery; refinement by a score	DAG. Practical compromise between CI testing and scoring.
LiNGAM [18]	Functional	Linear, non-Gaussian noise; typically no latent confounders	DAG. Can identify directions not identifiable under Gaussian CI tests.
NOTEARS [19]	Optimisation	Linear SEM (in the basic form); continuous optimisation + acyclicity constraint	Weighted adjacency, thresholded to a DAG. Sensitive to mismatch for discrete/binned data.
GOLEM [20]	Optimisation	Likelihood-based continuous optimisation under specific SEM likelihoods	DAG. Emphasises likelihood modelling and regularisation.
GraN-DAG [21]	Neural	Flexible nonlinear SEM parameterisation; optimisation stability	DAG. Higher capacity but introduces additional hyperparameters and optimisation variance.
RL-based search [22]	RL/score	Score model fidelity; enough signal to guide policy search	DAG. Flexible but can be compute-intensive and sensitive to reward design.
COSMO [23]	Optimisation	Differentiable orientation and stability heuristics; practical thresholds	DAG. Designed to stabilise discovery, but may trade accuracy for conservatism.

3.2 The Benchmarking Crisis in Causal Discovery

As causal discovery methods proliferated through the 2000s and 2010s, the field confronted a methodological crisis: how could researchers reliably compare algorithms when each study employed different datasets, metrics, and experimental protocols? Early evaluations often relied on small, bespoke simulations tailored to showcase a particular method's strengths, failing to capture the diversity and complexity of real-world data. This lack of standardisation made it nearly impossible to discern genuine algorithmic advances from artifacts of evaluation design.

3.2.1 Standardisation Efforts

A series of large-scale empirical studies attempted to bring order to the chaos. Scutari et al. [5] conducted an extensive benchmark across multiple network sizes and topologies, finding that constraint-based methods were often less accurate than score-based alternatives and seldom faster, contrary to their theoretical promise of polynomial-time complexity. Hybrids like MMHC offered no clear advantage over simpler score-based approaches, suggesting that the benefit of two-phase strategies was context-dependent. These results challenged the conventional wisdom that hybrid methods represented an unambiguous improvement.

Earlier, Tsamardinos et al. [17] had demonstrated that their Max-Min Hill-Climbing (MMHC) algorithm—combining constraint-based neighbourhood discovery with score-based refinement—consistently outperformed representative constraint-based (PC) and score-based (GES) methods in both reconstruction quality and computational speed on their chosen benchmarks. Their work was instrumental in establishing methodological norms: the use of standard metrics like Structural Hamming Distance (SHD) to quantify graph accuracy, explicit reporting of runtimes to assess scalability, and testing across multiple networks to guard against overfitting to a single domain. We adopt these practices throughout this thesis, ensuring that our benchmark results are comparable to the broader literature.

3.2.2 The "Scale-Free" Trap

Perhaps the most sobering critique came from Reisach et al. [6], who exposed a pervasive confound in benchmark design. They discovered that in many commonly used simulated datasets, variables' variances systematically increase along the causal order—a phenomenon they termed "varsortability." Because noise propagates and accumulates through causal chains, variables further downstream tend to exhibit higher variance. Remarkably, algorithms that simply sorted variables by their marginal variance and imposed edges according to that ordering could achieve state-of-the-art accuracy on these benchmarks, without learning any genuine causal mechanisms. This "variance sorting" heuristic collapses entirely when data are standardised (zero mean, unit variance), revealing that many purported algorithmic advances were illusory—they succeeded by exploiting an artifact of the data-generation process rather than discovering causal structure.

This finding profoundly motivates our experimental choices. We include both standardised preprocessing (as required by NOTEARS, which treats all variables as continuous) and evaluations on discrete data, where variance scaling is less trivial and the varsortability confound is naturally absent. By focusing on discrete Bayesian networks (Asia, ALARM, Child, Insurance, Sachs), we ensure that

algorithms must rely on genuine conditional independence patterns rather than shallow statistical cues.

3.3 Model Criticism and Mis-specification

While discovery algorithms focus on identifying the graph with the highest score or fewest independence violations, a complementary strand of research addresses a more fundamental question: once we have a candidate graph, how do we know if it is "good enough" for downstream inference? A graph might win a tournament of alternatives yet still contain critical structural errors that corrupt interventional predictions. This realisation has driven the development of *model criticism* techniques that go beyond global goodness-of-fit to probe for specific points of failure.

3.3.1 Global vs. Local Fit

Traditional model selection—whether via BIC, BDeu, or cross-validated likelihood—relies on *global* scores that aggregate fit across the entire graph. While such scores are useful for ranking candidate structures, they can obscure local mis-specifications. A graph might achieve a competitive global score while missing a single critical causal link, or it might include a spurious edge that happens to improve predictive fit for an idiosyncratic reason unrelated to the true mechanism.

Textor et al. [8] operationalised this concern in their DAGitty software, implementing routines to systematically test *every* conditional independence statement implied by a proposed DAG against the data. If any independence is significantly violated—for instance, if the graph claims $X \perp Y \mid Z$ but the data exhibit strong residual dependence—the graph is demonstrably mis-specified, regardless of its global score. Ankan et al. [7] extended this philosophy, showing that these local tests can pinpoint the likely location of errors: a violated independence involving variables X, Y, and Z suggests a missing edge in the subgraph connecting those nodes, or a spurious edge that should be removed.

Our work builds directly on this model-criticism paradigm. Rather than merely reporting summary metrics like SHD or F1 score, we systematically test whether conditional independence diagnostics can detect the specific analyst errors we introduce (spurious or missing edges). This approach treats structure learning not as a closed optimization problem but as an iterative hypothesis-testing workflow, where data-driven signals guide refinement of domain priors.

3.3.2 Refutation and Stability

Beyond local independence testing, a broader tradition of model criticism emphasises emphrefutation—systematically probing whether causal estimates remain stable under deliberate perturbations. The
textttDoWhy library, developed by Sharma and Kiciman, implements several refutation strategies for causal effect estimation: adding simulated random common causes to check whether estimates are sensitive to unobserved confounding, replacing the treatment variable with a placebo to verify that the effect vanishes as expected, or subsetting the data along different dimensions to assess robustness to sample composition. If an estimated effect fails these stress tests, it likely reflects model

mis-specification rather than genuine causality.

In the context of structure learning, the analogue of refutation is
emphstability selection, introduced by Meinshausen and B
ühlmann and widely adopted for causal discovery. Friedman et al.
citeFriedman1999 pioneered the use of bootstrap resampling to assess confidence in learned network features: by repeatedly generating bootstrap samples and re-running the discovery algorithm, one can estimate the sampling distribution of each edge. Scutari and Nagarajan
citeScutari2013 formalised this into a statistical framework, retaining only edges that appear above a chosen frequency threshold (e.g., 70
artifact of finite-sample noise rather than a robust structural feature; conversely, an edge with 95 support warrants high confidence.

We employ this bootstrap stability paradigm throughout our experiments, reporting not only point estimates of SHD and F1 but also their variance across resamples. This provides a more honest assessment of algorithm reliability: a method with low mean SHD but high variance is less deployable than one with slightly higher mean SHD but consistent performance.

3.4 The Differentiable Discovery Revolution

The 2018 publication of NOTEARS by Zheng et al. [19] marked a paradigm shift in causal discovery. They introduced a smooth, differentiable reformulation of the acyclicity constraint—previously thought to be inherently combinatorial—enabling structure learning to be cast as a continuous optimization problem solvable with standard gradient-based methods. By expressing the constraint that a graph must be acyclic as the equality $h(W) =$
$texttr(e^{WcircW}) - d = 0$, where W is a weighted adjacency matrix, they opened the door to leveraging modern automatic differentiation frameworks like PyTorch and TensorFlow. This was not merely an implementation convenience: it fundamentally expanded the class of models amenable to discovery, allowing nonlinear relationships and high-dimensional parameter spaces that were computationally prohibitive under discrete search.

3.4.1 Extensions and Limitations

This breakthrough unleashed a wave of follow-on research extending the differentiable framework in multiple directions. Lachapelle et al. [21] developed GraN-DAG, which replaces the linear structural equations of NOTEARS with flexible neural network parameterisations, enabling discovery of nonlinear causal mechanisms without committing to a specific functional form (polynomial, additive, etc.). Ng et al. [20] introduced GOLEM, which softens the hard acyclicity constraint into a continuous penalty term, improving convergence robustness and reducing sensitivity to initialisation—a practical concern when running gradient descent on non-convex objectives. Zhu et al. [22] took a different tack, applying reinforcement learning to search the graph space by treating edge addition and removal as discrete actions, with the score function providing the reward signal. This hybrid approach attempts to combine the exploration flexibility of RL with the scalability of gradient-based optimisation.

Despite these advances, differentiable methods introduce their own fragilities. They often require care-

ful hyperparameter tuning—the sparsity penalty λ in NOTEARS, for instance, can dramatically affect whether the learned graph is a dense tangle or a sparse skeleton. As Reisach's work demonstrated, they can also be sensitive to data preprocessing: variance scaling and standardisation choices that are innocuous for traditional methods can significantly alter differentiable methods' behaviour. Moreover, the continuous-optimisation paradigm is most natural for continuous data; applying NOTEARS to discrete variables (as we do in this thesis) constitutes a model mis-specification, treating categorical state values as if they were real numbers. Our benchmark includes NOTEARS precisely to assess whether this "revolution" in algorithmic design translates into reliable performance on the classic discrete Bayesian networks that dominate causal discovery benchmarks—a setting often overlooked in the deep-learning-focused literature.

3.5 Software Ecosystem

The gap between algorithmic theory and empirical practice is often bridged—or widened—by the quality of available software. Implementations matter: a theoretically sound algorithm can fail in practice due to numerical instabilities, poor default hyperparameters, or undocumented preprocessing assumptions, while a well-engineered package can make even modest algorithms widely adopted.

The
textbfR ecosystem has long been the gold standard for causal discovery, with
textttbnlearn by Scutari
citeScutari2010 and
textttpcalg by Kalisch et al.
citeKalisch2012 offering mature, battle-tested implementations of PC, GES, Hill-Climbing (HC), and related methods. These packages benefit from decades of community refinement, comprehensive documentation, and integration with R's rich statistical ecosystem. Their robustness has made them the reference implementations for benchmarking new methods.

The
textbfPython ecosystem has recently caught up, driven by the machine-learning community's preference for Python tooling. Libraries like
texttt causal-learn—a port of the venerable Tetrad Java library maintained by the CMU causal inference group
citeZheng2024—bring constraint-based and score-based methods into the PyTorch/scikit-learn world. Huawei's
texttt gCastle toolkit provides a unified API spanning traditional algorithms and newer differentiable methods, facilitating head-to-head comparisons. For practitioners working in industrial machine-learning pipelines, these Python implementations enable seamless integration with existing data-processing workflows.

Beyond command-line libraries,
textbfinteractive tools like
texttt DAGitty (web-based) and
texttt CausalWizard emphasise the "human-in-the-loop" aspect of causal modeling. These graphical interfaces allow domain experts to sketch prior knowledge, visualise learned structures, and interactively test conditional independence queries. Such tools acknowledge that causal discovery is rarely a fully

automated process; it is an iterative dialogue between algorithmic suggestions and expert domain knowledge.

This thesis leverages the Python ecosystem, specifically
textttcausal-learn for constraint-based and score-based methods and
textttCausalNex for differentiable approaches, to provide a modern, reproducible benchmarking pipeline. All code, configurations, and results are version-controlled and publicly archived, ensuring that our findings can be independently verified and extended by future researchers.

3.6 Positioning of this Thesis

Most prior benchmarks adopt an
emphalgorithmic lens, framing the central question as: which method achieves the highest F1 score, lowest SHD, or fastest runtime? While such comparisons are valuable, they implicitly assume that causal discovery is a fully automated process where the "best" algorithm is simply run on raw data to produce a final graph. This thesis shifts the focus to the
emphanalyst—the human expert who brings domain knowledge, interprets algorithmic outputs, and iteratively refines the learned structure. In real-world practice, algorithms are rarely run in isolation; they serve as tools to assist domain experts who possess prior beliefs about plausible mechanisms but require data-driven validation or correction.

Our evaluation is
textbfanalyst-centric in three ways. First, we assess not merely raw recovery accuracy (how close is the learned graph to the ground truth when the algorithm has no prior information?) but also whether data-driven diagnostics—conditional independence tests, bootstrap stability scores—can effectively emphcorrect analyst errors when domain priors conflict with the data. If an expert mistakenly imposes a spurious edge, can a well-designed CI test flag it as inconsistent with observed independencies? If an expert omits a critical link, does the algorithm's learned structure or a targeted independence query reveal the gap? This reframes structure learning as an interactive workflow rather than a one-shot optimization.

Second, we conduct a
textbfholistic comparison across paradigms, placing the "old guard" (constraint-based PC, score-based GES) and the "new wave" (differentiable NOTEARS, stability-focused COSMO) on a level playing field. Rather than cherry-picking datasets that favour one approach, we employ standard discrete Bayesian networks (Asia, ALARM, Child, Insurance, Sachs) that have been used in causal discovery research for decades. By applying both traditional and modern methods to the same benchmarks under identical evaluation protocols (CPDAG-aware SHD, bootstrap resampling, sensitivity sweeps), we can assess whether recent algorithmic innovations translate into tangible gains or whether the classics remain competitive.

Third, we emphasise
textbfreproducibility and transparency. All code, configuration files, and results are version-controlled and publicly archived. The benchmark pipeline is implemented in Python using widely adopted libraries (
textttcausal-learn,

textttCausalNex), lowering the barrier for practitioners to adapt our protocols to their own datasets. Rather than presenting opaque “black-box” comparisons, we expose every preprocessing choice, hyperparameter setting, and random seed, enabling independent verification and extension of our findings. This commitment to openness reflects the broader methodological critique raised by Reisach and others: causal discovery research must adopt the rigorous evaluation standards of the broader machine-learning community.

4 Methods

This chapter details the experimental design underpinning our empirical investigation. We begin by describing the five canonical Bayesian network benchmarks that serve as our testbed, explaining the semi-synthetic data generation process and the steps taken to eliminate known confounds (particularly variance sortability). We then specify the four causal discovery algorithms evaluated—PC, GES, NOTEARS, and COSMO—along with their hyperparameter settings and computational environment. The evaluation framework is formalised next, covering the metrics (SHD, SID, precision, recall, F1) and their CPDAG-aware computation. Finally, we operationalise analyst mis-specification through controlled edge perturbations (adding spurious edges or removing true edges) and describe the bootstrap stability protocol used to assess robustness.

4.1 Datasets and Data Generation

Our experiments employ five canonical Bayesian network benchmarks, summarised in Table reftab:datasets. These networks span a range of structural properties—from the small, dense Asia network (8 nodes, 8 edges) to the large, sparse ALARM network (37 nodes, 46 edges)—and originate from diverse application domains including medical diagnosis, protein signaling, pediatric care, risk assessment, and intensive-care monitoring. This heterogeneity ensures that our benchmark results are not artifacts of a single graph topology or domain-specific data distribution. By testing algorithms across networks with varying node counts, edge densities, and maximum degrees, we can identify which methods exhibit robust performance and which are brittle to structural characteristics.

4.1.1 Implemented synthetic generation: linear SEM with quantile discretisation

The discrete benchmark datasets in our repository (Asia, ALARM, Child, Insurance) are emphsemi-synthetic in a precise technical sense: the
emphgraph structure is taken from the canonical benchmark networks documented in the literature, but the
emphsampled data are generated de novo via a linear structural equation model (SEM) with independent Gaussian noise, then discretised variable-wise to match the intended state cardinalities. This design yields a controlled experimental setting where the true causal DAG is known with certainty (enabling exact SHD and SID calculations) while allowing the sample size n to be set uniformly across all datasets, eliminating confounds from varying statistical power.

Concretely, variables are generated in a topological order consistent with the ground-truth DAG. For

each node i, we define
beginequation X$_i$
;=
;
$sum_{jinmathrmPa(i)}w_{ji}X_j +$
$varepsilon_i$,
$qquad$
$varepsilon_i$
sim
$mathcalN(0,1)$,
$endequationwheretheedgeweightsw_{ji}$ are sampled once from a bounded uniform distribution (with random sign to ensure heterogeneous causal directions) and then held fixed for all data generation. To obtain discrete observations with a specified number of states r_i, the continuous samples X_i are transformed via quantile-based binning:
beginequation
tildeX$_i$
;=
;$Q_i(X_i)$,
$endequationwhereQ_i($
$cdot)$ maps X_i into the discrete set
$0, 1, dots, r_i - 1$ using empirical quantile cutpoints computed from the sample. This quantile discretisation preserves the
emphrank structure induced by the linear SEM while producing a discrete-valued dataset suitable for chi^2-based conditional independence tests.

Addressing variance sorting bias. Reisach et al.
citeReisach2021 identified a critical artifact that has contaminated many published causal discovery benchmarks: when data are generated via a linear SEM without standardisation, variables downstream in the causal order accumulate variance from their ancestors through the recursive construction. For instance, in the chain X_1
toX_2
toX_3 with unit-variance noise at each step, we have
$mathrmVar(X_3) >$
$mathrmVar(X_2) >$
$mathrmVar(X_1)$ because X_3 inherits variance from both its direct parent and its grandparent. Algorithms that simply rank variables by marginal variance and impose edges according to that ordering—a trivial heuristic requiring no causal reasoning—can achieve state-of-the-art performance on such benchmarks, rendering comparative conclusions meaningless.

To eliminate this confound for
emphcontinuous datasets (specifically Sachs), we standardise all variables to zero mean and unit variance after generation but before saving the dataset. This ensures that no variable's marginal variance leaks information about its position in the topological ordering. NOTEARS and other continuous-data algorithms then load this pre-standardised data without applying additional normalisation, preserving

the generated scale. For
emphdiscrete datasets, the quantile-based discretisation itself eliminates the variance artifact: by mapping continuous values to discrete categories via rank-based quantiles, we produce perfectly uniform marginal distributions (each state appears with frequency
$approx 1/r_i$), so variance accumulation in the underlying continuous data cannot leak into the discrete observations.

textbfImportant: All results reported in Section
refsec:results use data generated with standardisation enabled for the Sachs network. During final implementation review, we discovered that the original Sachs dataset had been generated without standardisation, exhibiting variance ratios of approximately 5:1 between leaf and root nodes. The dataset has been regenerated to correct this artifact. Discrete datasets (Asia, ALARM, Child, Insurance) are unaffected because quantile binning renders variance accumulation irrelevant.

Why this matters. This semi-synthetic data-generation strategy is intentionally simple and fully reproducible, but it introduces an important interpretive nuance. Although the networks are treated as emphdiscrete in our benchmarking code (we apply
chi^2 tests for conditional independence, use discrete BIC scores, etc.), the underlying data-generating mechanism is a discretised linear-Gaussian SEM. Consequently, algorithms that assume linear relationships—particularly NOTEARS, which is explicitly designed for continuous linear SEMs—may enjoy a latent advantage not present when applied to genuinely discrete Bayesian networks with categorical conditional probability tables. Conversely, methods designed for discrete data (like PC with

chi^2 tests) must contend with the fact that the rank structure imposed by quantile binning may differ subtly from the conditional independence patterns of a native discrete distribution. This mismatch is a deliberate trade-off: we prioritise
emphreproducibility and
emphcontrol (knowing the exact ground truth, eliminating variance artifacts, setting sample size uniformly) over perfect ecological validity. Section
refsec:validity discusses these limitations in detail.

Discretization Procedure. To ensure transparency, we provide the exact procedure used to discretise the continuous samples. Let $X \in \mathbb{R}^{n \times d}$ be the data matrix generated by the linear SEM. For each variable $j \in \{1, \ldots, d\}$ with desired cardinality k_j:

1. Compute the empirical quantiles $q_0, q_1, \ldots, q_{k_j}$ of column $X_{\cdot,j}$, where $q_0 = \min(X_{\cdot,j})$ and $q_{k_j} = \max(X_{\cdot,j})$.
2. Map each value x_{ij} to a discrete category $c_{ij} \in \{0, \ldots, k_j - 1\}$ such that $q_{c_{ij}} \leq x_{ij} < q_{c_{ij}+1}$.

This quantile binning ensures that the marginal distribution of each discrete variable is approximately uniform, maximising the entropy and avoiding "rare category" issues that can destabilise CI tests.

```
Algorithm 1: Data Generation and Discretization
Input: DAG G, Sample size n, Cardinalities K
Output: Discrete data matrix D
1.  Initialize continuous matrix X of size (n, |V|)
2.  TopologicalSort(G) -> order
3.  For each node i in order:
4.      parents = Parents(G, i)
5.      weights = RandomUniform(-1, 1, size=|parents|)
6.      noise = Normal(0, 1, size=n)
7.      X[:, i] = X[:, parents] @ weights + noise
8.  For each node i:
9.      k = K[i]
10.     bins = Quantiles(X[:, i], k)
11.     D[:, i] = Digitize(X[:, i], bins)
Return D
```

Figure 1: Pseudocode for the semi-synthetic data generation process.

4.1.2 Determinism and dataset reuse

To support exact reproducibility, the benchmark datasets are stored as CSV files under `causal_benchmark/data/`. Unless forced regeneration is enabled, experiments load these fixed datasets rather than resampling at each run. This choice ensures that differences across algorithms are attributable to the learning procedures rather than stochastic variation in the sampled dataset, but it also means that the benchmark section reports performance on a single realised sample per dataset (for the chosen n).

Table 5: Summary of benchmark networks. Edges refers to the number of directed edges in the ground–truth DAG. Density is $|E|/(|V|(|V|-1)/2)$, the fraction of possible edges present. Sample size n indicates the number of observations generated for each experiment.

Network	Nodes	Edges	Density	Data Type	n	Domain
Asia	8	8	0.29	Discrete	1000	Medical diagnosis
Sachs	11	17	0.31	Continuous	1000	Protein signalling
Child	20	25	0.13	Discrete	1000	Paediatric diagnosis
Insurance	27	52	0.15	Discrete	1000	Risk assessment
ALARM	37	46	0.07	Discrete	1000	ICU monitoring

The five networks were chosen to cover diverse structural properties and application domains, ensuring that our benchmark results reflect algorithm behaviour across a spectrum of graph topologies rather than idiosyncrasies of a single network. Asia [31]—a small, dense network (8 nodes, 8 edges, density 0.29)—serves as a "sanity check," where even modest algorithms should recover most of the structure given reasonable sample sizes. The Sachs network encodes a protein signaling pathway and is widely used in causal discovery evaluations; although historically associated with interventional studies [**?**], we treat it here as an observational benchmark, generating synthetic continuous samples from its graph structure with standardisation enabled to eliminate variance sorting artifacts. ALARM [32], originally developed for intensive-care patient monitoring, represents the challenging end of the spectrum: its 37 nodes and 46 edges form a large, sparse graph (density 0.07) where algorithms must navigate many potential spurious associations. Child and Insurance occupy the intermediate range, providing moderate complexity (20 and 27 nodes, respectively) that stresses algorithms without overwhelming

them.

Data regeneration note: During final review of the benchmark implementation, we identified that the original Sachs dataset was generated without variance standardization, allowing a variance sorting artifact to potentially inflate algorithm performance. The dataset has been regenerated with standardization enabled.

4.2 Algorithms and Settings

We evaluate four causal discovery algorithms, each representing a distinct methodological paradigm that has shaped the field's evolution. PC (Peter-Clark) exemplifies the constraint-based tradition, systematically testing conditional independencies to infer graph structure. GES (Greedy Equivalence Search) represents score-based optimisation, balancing likelihood fit against model complexity via BIC. NOTEARS embodies the recent "differentiable revolution," recasting structure learning as continuous optimisation over weighted adjacency matrices. COSMO (Causal Ordering via Scalable Modeling) takes a regression-based approach, learning a causal ordering and using Lasso-penalised regression to identify parents within that order. Table 6 summarises each algorithm's implementation source and key hyperparameter settings; the rationale for these choices is detailed in the subsections below.

Table 6: Algorithm implementations and hyperparameter settings. CI = conditional independence test. All algorithms were allowed to run to completion.

Algorithm	**Type**	**Implementation**	**Key Settings**
PC	Constraint	causal–learn	CI test: χ^2 (discrete), Fisher-z (continuous); $\alpha = 0.05$
GES	Score	causal–learn	Score: BIC (discrete and continuous); fixed penalty
NOTEARS	Optimisation	CausalNex	Threshold: 0.1 (continuous), 0.25 (discrete); other parameters: CausalNex defaults
COSMO	Regression	numpy/networkx	$n_{\text{restarts}} = 25$; auto–λ via BIC; edge threshold = 0.08

4.2.1 Hyperparameter Selection Protocol

To ensure fair comparison, we adopted a standardised protocol for hyperparameter selection that balances methodological rigor with practical deployability. Rather than exhaustively tuning each algorithm on each dataset—a process that would introduce overfitting to our specific benchmarks and obscure real-world performance—we selected hyperparameters based on literature conventions, software defaults, and pilot sensitivity analyses.

For **PC**, the significance level α controls the trade-off between Type I errors (spurious edges) and Type II errors (missing edges). We adopted $\alpha = 0.05$ as the standard convention established in [2] and widely used across the causal discovery literature. To verify that this choice yields reasonable performance, we conducted a sensitivity analysis (Section 5.3) sweeping $\alpha \in \{0.01, 0.05, 0.10\}$ and confirmed that $\alpha = 0.05$ provides a balanced trade-off between precision and recall across our benchmark networks.

For **GES**, the primary hyperparameter is the penalty weight in the BIC score, which determines how aggressively the algorithm penalises model complexity. We used the canonical BIC penalty $\frac{\log n}{2}$ per parameter uniformly across all datasets (both discrete and continuous). This deliberate choice—applying a single score formulation rather than switching between discrete and continuous variants—allows us to evaluate the robustness of score-based methods to model mis-specification, a central theme of this thesis.

For **NOTEARS**, the L_1 sparsity penalty λ and the edge inclusion threshold w_{thresh} jointly determine the learned graph's density. We retained the `CausalNex` default $\lambda = 0.05$ but adjusted the threshold based on pilot experiments. We observed that NOTEARS produces many weak edges (small absolute weights) when applied to discrete data, inflating false positives. To compensate, we increased the threshold from the software default of 0.1 to 0.25 for discrete datasets, retaining only edges with stronger evidence. This threshold adjustment reflects a practical reality: differentiable methods require more manual calibration than constraint-based approaches, where the significance level α has a well-established interpretation.

For **COSMO**, we leveraged its built-in automatic λ selection mechanism, which sweeps a path of regularisation strengths and selects the value minimising a held-out BIC score. The edge inclusion threshold was set to 0.08 (slightly lower than NOTEARS to accommodate COSMO's conservative parent selection), and stability selection retained only edges appearing in at least 25% of bootstrap resamples. This high threshold reflects COSMO's design philosophy: sacrifice recall to maximise precision and guard against spurious edges.

4.2.2 Computational Environment

All experiments were conducted on a MacBook Pro equipped with an Apple M2 Pro chip (10-core CPU, 16-core GPU) and 16GB of unified memory. The software environment was managed via Conda (version 23.7.4) to ensure reproducibility and version control. We used Python 3.10 as the interpreter, constrained by compatibility requirements of the `CausalNex` library (which does not support Python 3.11 or later due to dependency conflicts). Core libraries included `causal-learn` 0.1.3.6 for PC and GES implementations, `CausalNex` 0.12.0 for NOTEARS, `numpy` 1.23.5 and `pandas` 1.5.3 for data manipulation, `networkx` 2.8.8 for graph operations, and `scikit-learn` 1.2.2 for regression components in COSMO.

To ensure fair runtime comparisons, all algorithms were run sequentially on a single thread where possible, though NOTEARS and COSMO leverage vectorised numerical operations (via NumPy's underlying BLAS/LAPACK) that may implicitly utilise multiple cores. Because our benchmark networks are small to moderate in size (8–37 nodes), runtimes are dominated by algorithmic logic rather than numerical linear algebra, minimising the impact of parallelism on comparative conclusions. A subtleimportant constraint shaped our environment: NOTEARS via CausalNex requires Python versions earlier than 3.11 due todependency conflicts, which is why we standardised on Python 3.10 across all experiments to ensure reproducibility.

4.2.3 Mapping algorithm settings to the implementation

All algorithm invocations are controlled by a single configuration file (`causal_benchmark/experiments/config.yaml`) and thin wrapper modules under `causal_benchmark/algorithms/`. These wrappers serve a dual purpose: they translate a common `pandas` DataFrame representation into the specific API required by each library (e.g., `causal-learn` expects NumPy arrays with variable names as a separate argument, whereas `CausalNex` accepts DataFrames directly), and they normalise outputs into a consistent adjacency-matrix format for downstream evaluation. This abstraction layer ensures that algorithmic comparisons are not confounded by idiosyncrasies of library interfaces.

Two implementation details warrant explicit mention because they affect result interpretation. First, **conditional independence tests are matched to data type**: the PC wrapper automatically selects Fisher's z-test for continuous datasets (Sachs) and the χ^2 test for discrete datasets (Asia, ALARM, Child, Insurance), aligning the test with the data's native representation. This matching is methodologically sound but means that PC's performance on Sachs versus discrete networks reflects both algorithmic behaviour and test calibration differences. Second, **GES score choice is uniform**: the GES wrapper applies BIC with a Gaussian likelihood assumption to *all* datasets, including discrete ones. This deliberate mis-specification—using a continuous-data score for discrete data—allows us to assess the robustness of score-based methods to model mismatch, a key research question of this thesis.

4.2.4 Runtime measurement and parallel execution

All reported runtimes (Table
reftab:runtime) reflect wall-clock duration measured from the moment the algorithm beginsdata until it returns a complete adjacency matrix, capturing the end-to-end computational cost a practitionerexperience when deploying these methods. Although the benchmark harness supports parallel execution across multiple–algorithm pairs to accelerate the overall experiment pipeline, each individual algorithm invocation is timed in. This design ensures that reported runtimes are not distorted by concurrent resource contention or caching: if NOTEARS takes 12.3 seconds on the ALARM dataset, that duration corresponds to a dedicated, single-threaded, not a fraction of a parallelised batch.

4.2.5 Reproducibility artefacts

Beyond aggregate performance metrics, the benchmark harness preserves detailed provenance for every algorithm run to support post-hoc analysis and error diagnosis. For each dataset–algorithm pair, the system writes three complementary artefacts: (i) the predicted adjacency matrix in machine-readable format (.npy), enabling exact reconstruction of the learned graph; (ii) a human-readable edge list with node names, facilitating manual inspection; and (iii) a structured diff object enumerating the sets of extra, missing, and reversed edges relative to the ground truth. These granular records underpin the error decomposition analyses presented in Section 5, where we dissect total mistakes into specific categories (false positives, false negatives, orientation errors) to diagnose algorithm failure modes. Without such detailed artefacts, aggregate metrics like SHD and SID would conceal the qualitative patterns—whether an algorithm tends toward over-fitting (excessive edges) or conservatism

(sparse graphs), for instance—that explain performance differences across networks.

4.2.6 PC Algorithm

The Peter-Clark (PC) algorithm [2] stands as the canonical exemplar of constraint-based causal discovery, reasoning backward from statistical dependencies to graph structure. It operates under two critical assumptions—faithfulness and causal sufficiency—which together guarantee that conditional independencies in the observed distribution correspond precisely to d-separations in the underlying DAG. When these assumptions hold, PC provably recovers the Markov equivalence class (represented as a CPDAG) from infinite data.

Core Mechanism: PC employs a three-stage procedure that begins with maximal connectivity and progressively refines the graph via independence testing. The algorithm starts with a complete undirected graph on all variables and then systematically removes edges (X, Y) whenever a conditioning set Z can be found such that $X \perp Y \mid Z$ according to a statistical independence test. This skeleton-learning phase proceeds level-by-level, where k denotes the size of the conditioning set $|Z|$, ensuring that all small conditioning sets are exhausted before testing larger ones. Once the skeleton (undirected graph) stabilises, PC identifies v-structures—configurations $X - Z - Y$ where X and Y are non-adjacent—and orients them as $X \to Z \leftarrow Y$ if Z was not in the separating set SepSet(X, Y). Finally, Meek's orientation rules [**?**] propagate orientations to additional edges, respecting acyclicity and avoiding the creation of new v-structures. The complete procedure is formalised below:

```
Algorithm 2: PC Algorithm
Input: Data D, Significance level alpha
Output: CPDAG G
1.  Form complete undirected graph G on V
2.  k = 0
3.  Repeat until k > max_degree(G):
4.      For each edge (X, Y) in G:
5.          For each subset Z of Adj(X) \ {Y} with |Z| = k:
6.              p_val = CI_Test(X, Y, Z, D)
7.              If p_val > alpha:
8.                  Remove edge (X, Y)
9.                  SepSet(X, Y) = Z
10.                 Break (move to next edge)
11.     k = k + 1
12. For each triple X - Z - Y with X, Y non-adjacent:
13.     If Z not in SepSet(X, Y):
14.         Orient X -> Z <- Y
15. Apply Meek Rules (R1-R3) to orient further edges
Return G
```

Figure 2: Pseudocode for the PC algorithm.

Implementation Details: Our implementation leverages the `causal-learn` library version 0.1.3.6, with the `stable=True` flag enabled to eliminate order-dependence in edge removal—a common source of spurious variability in PC results. The choice of conditional independence test is matched to the data type: discrete datasets (Asia, ALARM, Child, Insurance) employ the χ^2 test for contingency tables, while the continuous dataset (Sachs) uses Fisher's z-test based on partial correlation. Across all runs,

the significance threshold is fixed at $\alpha = 0.05$, the field-standard value that balances Type I and Type II error rates.

Data preprocessing for PC: Because our data generation pipeline already produces standardised continuous samples (mean 0, variance 1), the Sachs benchmark requires no additional preprocessing before PC; the data enter the algorithm in their native scale. For discrete datasets, the quantile binning transformation applied during data generation constitutes the sole preprocessing step, ensuring that integer-valued categorical variables align with the χ^2 test's assumptions.

Complexity and Sensitivity: In the worst case—a complete or nearly-complete graph—PC's runtime is exponential in the number of variables $|V|$ because the algorithm must test conditioning sets of size up to $|V| - 2$. However, sparse graphs with maximum degree d exhibit polynomial complexity, scaling as $O(|V|^{d+1})$. A critical sensitivity arises from the sequential nature of edge removal: if PC commits a Type II error (fails to detect a true independence) early in the skeleton phase, it may erroneously remove an edge that should have been conditioned upon in later tests, cascading into additional mistakes. This fragility motivates the careful calibration of α and the use of stability-enhancing heuristics like consistent ordering.

4.2.7 Greedy Equivalence Search (GES)

Greedy Equivalence Search [3] exemplifies the score-based paradigm, formulating causal discovery as an optimisation problem over the space of Markov equivalence classes. Unlike constraint-based methods that test individual independencies, GES evaluates entire graph structures via a decomposable scoring function—typically the Bayesian Information Criterion (BIC) or the Bayesian Dirichlet equivalent uniform (BDeu) score—that balances goodness of fit against model complexity. A key advantage of operating directly on CPDAGs (equivalence classes) rather than individual DAGs is statistical consistency: GES provably recovers the true equivalence class from infinite data under the assumptions of causal sufficiency and faithfulness.

Core Mechanism: GES employs a two-phase greedy search strategy that mirrors the classic forward–backward selection heuristic from regression modelling. The forward phase begins with an empty graph (no edges) and iteratively adds the single edge that yields the greatest increase in the score, continuing until no addition improves the model. This phase constructs a coarse approximation of the graph, potentially over-fitting to noise in finite samples. The backward phase then refines the structure by iteratively removing edges, again selecting the deletion that maximises score improvement, until the graph stabilises. The combination of aggressive addition followed by conservative pruning mitigates over-fitting while maintaining computational tractability. Formally, the procedure unfolds as follows:

Implementation Details: Our GES implementation draws on `causal-learn` version 0.1.3.6, configured to use the BIC score with a Gaussian likelihood assumption uniformly across *all* datasets, including both the continuous Sachs network and the discrete networks (Asia, ALARM, Child, Insurance). This deliberate uniformity constitutes a controlled mis-specification experiment: while the BDeu score would be theoretically optimal for discrete data (providing exact Bayesian posterior probabilities under a Dirichlet prior), applying BIC throughout allows us to isolate the impact of score choice on algorithm robustness. Does a single scoring function—calibrated for continuous data—suffice across heterogeneous benchmarks, or does optimal performance require adaptive score selection? The

```
Algorithm 3: Greedy Equivalence Search (GES)
Input: Data D, Score function S
Output: CPDAG G
1.  G = EmptyGraph(V)
2.  # Forward Phase
3.  Loop:
4.      BestScore = -Infinity, BestOp = None
5.      For each valid edge addition E+ in CPDAG space:
6.          Gain = S(G + E+, D) - S(G, D)
7.          If Gain > BestScore:
8.              BestScore = Gain, BestOp = E+
9.      If BestScore > 0:
10.         G = Apply(G, BestOp)
11.     Else:
12.         Break
13. # Backward Phase
14. Loop:
15.     BestScore = -Infinity, BestOp = None
16.     For each valid edge deletion E- in CPDAG space:
17.         Gain = S(G - E-, D) - S(G, D)
18.         If Gain > BestScore:
19.             BestScore = Gain, BestOp = E-
20.     If BestScore > 0:
21.         G = Apply(G, BestOp)
22.     Else:
23.         Break
Return G
```

Figure 3: Pseudocode for the GES algorithm.

results in Section 5 address this question directly.

Complexity and Sensitivity: GES is computationally more intensive than constraint-based methods because each candidate edge addition or deletion requires re-evaluating the decomposable score over affected local structures (parent sets). For $|V|$ variables and maximum in-degree k, the forward phase alone evaluates $O(|V|^2 \cdot |V|^k)$ candidate additions, yielding super-polynomial runtime in the worst case. However, GES's greedy nature provides practical scalability: sparse ground-truth graphs typically admit small parent sets, enabling the algorithm to navigate the search space efficiently. A key sensitivity arises from the score's penalty term: if the complexity penalty (e.g., $\frac{\log n}{2}$ per parameter in BIC) is too weak, GES over-fits, adding spurious edges; if too strong, it under-fits, yielding excessively sparse graphs. Unlike PC, which relies on binary accept–reject decisions at each independence test, GES accumulates evidence across the entire dataset via a single global score, making it less brittle to individual test failures but potentially more vulnerable to systematic model mis-specification.

4.2.8 NOTEARS

NOTEARS (Non-combinatorial Optimisation via Trace Exponential and Augmented lagRangian for Structure learning) [19] represents a paradigm shift in causal discovery: rather than framing structure learning as a discrete combinatorial search over graphs, NOTEARS recasts it as continuous optimi-

sation over weighted adjacency matrices. The central innovation is an algebraic characterisation of acyclicity that eliminates the need for explicit cycle-checking or topological sorting during optimisation.

Core Mechanism: NOTEARS solves a constrained optimisation problem that simultaneously learns edge weights and enforces acyclicity. The objective function balances empirical fit (via a least-squares loss $\ell(W; X)$ measuring how well the linear structural equation model $X = W^\top X + \varepsilon$ explains the data) against a sparsity-inducing L_1 penalty $\lambda\|W\|_1$ that favours graphs with few edges. The critical constraint $h(W) = \text{tr}(e^{W \circ W}) - d = 0$ ensures that the weighted adjacency matrix W corresponds to a DAG; here $\circ$ denotes the Hadamard (element-wise) product, tr denotes the matrix trace, and d is the number of variables. Zheng et al. proved that $h(W) = 0$ if and only if the graph encoded by W is acyclic, converting a discrete topological property into a differentiable equality constraint. The resulting problem,

$$\min_W \ell(W; X) + \lambda\|W\|_1 \quad \text{subject to} \quad h(W) = \text{tr}(e^{W \circ W}) - d = 0, \tag{9}$$

is tackled via augmented Lagrangian methods, iteratively minimising the augmented objective and updating Lagrange multipliers until the acyclicity constraint is satisfied within numerical tolerance.

Implementation Details: We employ the `CausalNex` implementation (version 0.12.0) of NOTEARS, which returns a dense matrix of learned weights $\hat{W}$. Because many weights are small but non-zero due to optimisation artifacts and finite-sample noise, we apply hard thresholding to extract a sparse binary adjacency matrix: edges with $|\hat{W}_{ij}| < \tau$ are discarded. The threshold τ is data-type-dependent: we use $\tau = 0.1$ for the continuous Sachs dataset, where the linear-Gaussian assumptions are approximately valid, and $\tau = 0.25$ for discrete datasets (Asia, ALARM, Child, Insurance), where the mis-match between NOTEARS's continuous optimisation and discrete data necessitates more aggressive pruning to control false positives.

Data Preprocessing: The discrete benchmark datasets undergo mean-centring and unit-variance scaling before input to NOTEARS, which then treats the discretised integer values as continuous and fits a linear SEM. This preprocessing reflects NOTEARS's assumption of continuous data: although the samples are categorical, standardisation ensures numerical stability during gradient-based optimisation. The continuous Sachs dataset, already generated with unit variance, requires no additional preprocessing, preserving its original scale.

Complexity and Sensitivity: NOTEARS scales as $O(d^3)$ per iteration due to the matrix exponential computation in the acyclicity constraint, where $d = |V|$ is the number of variables. This cubic scaling makes NOTEARS faster than exact search methods (which are NP-hard) but slower than constraint-based approaches like PC for large sparse graphs, where conditional independence testing can be localised to small neighbourhoods. The algorithm's Achilles' heel is its strong parametric assumption: NOTEARS presumes a linear-Gaussian SEM, and violations of this model—such as discrete variables, nonlinear relationships, or non-Gaussian noise—render the loss function $\ell(W; X)$ mis-specified. In such cases, the learned weights may bear little relation to true causal effects, as the objective optimises a proxy that does not align with the data-generating process. Our benchmark deliberately stresses this sensitivity by applying NOTEARS to discrete data, enabling us to quantify the performance penalty incurred by model mis-specification.

4.2.9 COSMO

COSMO (Constrained Orientations by Sequential Modeling) represents a hybrid approach that marries regression-based parent selection with stability selection to guard against false discoveries. Unlike NOTEARS, which solves a single global optimisation problem, COSMO fits a series of local regression models—one per variable—and aggregates results across multiple random restarts to identify edges that appear consistently.

Core Mechanism: COSMO's procedure rests on the principle that true causal parents should emerge robustly across diverse random initialisations, whereas spurious associations depend on the specific ordering of variables. The algorithm defines a smooth acyclicity penalty based on ordering variables on a circle, which allows gradient-based optimisation while still enforcing the DAG constraint. For each restart, COSMO generates a random causal ordering, then fits regularised (Lasso) regressions for each variable conditional on its predecessors in that ordering. Edges selected by the regression step are recorded, and after N restarts, only edges appearing in a sufficiently large fraction (the stability threshold) are retained. This resampling-and-aggregation strategy trades computational cost for robustness, filtering out edges that depend on fortuitous variable orderings.

Implementation Details: Our COSMO implementation is built from scratch using `numpy` for numerical operations and `scikit-learn` for Lasso regression. We perform 25 random restarts (a practical compromise between computational cost and stability), set the edge weight inclusion threshold to 0.08 (slightly lower than NOTEARS to accommodate COSMO's conservative parent selection), and retain only edges appearing in at least 25% of restarts. The Lasso regularisation parameter λ is selected automatically via BIC at each restart, sweeping a path of candidate values and choosing the one that minimises the penalised likelihood.

Complexity and Sensitivity: COSMO scales as $O(d^2 \cdot N)$ where $d = |V|$ is the number of variables and N is the number of restarts, making it computationally efficient compared to PC (which can be exponential in the worst case) and GES (super-polynomial in dense graphs). The algorithm is highly parallelisable: each restart is independent, enabling trivial distribution across multiple cores. Like NOTEARS, COSMO assumes linearity—the Lasso regression presumes that each variable is a linear function of its parents plus Gaussian noise—and performance degrades when this assumption is violated (e.g., in the presence of nonlinear interactions or discrete variables). However, COSMO's stability selection mechanism provides some robustness to noise and sampling variability, often yielding sparser, more conservative graphs than NOTEARS at the cost of potentially lower recall.

4.3 Benchmarking pipeline and post-processing

Figure 5 illustrates the end-to-end workflow governing both the main benchmark and the sensitivity analyses. Each experimental run proceeds through four sequential stages, ensuring consistent data handling and evaluation across all algorithm–dataset combinations. First, we generate an observational dataset by sampling from the benchmark graph's structural equation model (Section 4.1), using a fixed random seed to enable exact reproducibility. Second, we invoke one of the four causal discovery algorithms (PC, GES, NOTEARS, or COSMO) with its dataset-appropriate configuration (Section 4.2), capturing both the learned graph and runtime metadata. Third, we post-process the raw algorithm output—which may be a CPDAG, a weighted adjacency matrix, or a mixture of directed and

undirected edges—into a standardised directed graph representation suitable for evaluation. Finally, we compute skeleton-level and directed performance metrics (precision, recall, F_1, SHD, SID) by comparing the learned graph against the known ground truth, documenting the results in structured logs for downstream analysis.

4.3.1 Detailed Pipeline Walkthrough

To guarantee reproducibility and facilitate replication by other researchers, the entire pipeline is automated via the `run_benchmark.py` script, which orchestrates data loading, algorithm invocation, post-processing, and metric computation without manual intervention.

1. Data Loading and Preprocessing: Benchmark datasets reside as CSV files in the `data/` directory, with each network's samples pre-generated during the data preparation phase (Section 4.1). The pipeline loads these files into `pandas` DataFrames, preserving data types according to algorithm requirements. For discrete-data algorithms (PC and GES applied to Asia, ALARM, Child, Insurance), integer codes representing categorical levels are retained exactly as stored. For continuous-data algorithms (NOTEARS, COSMO, and PC/GES applied to Sachs), values are loaded as floating-point numbers. NOTEARS uniquely requires an additional standardisation step: before optimisation, the data are mean-centred and scaled to unit variance, ensuring numerical stability in the gradient-based solver. This transformation does not alter the conditional independence structure, but it does affect the interpretability of learned edge weights, which must be interpreted relative to the standardised scale.

2. Algorithm Execution: Algorithms are invoked through uniform wrapper functions housed in the `algorithms/` module, which abstract away library-specific API details and present a common interface: each wrapper accepts a `pandas` DataFrame and returns a `networkx` directed graph along with a metadata dictionary capturing runtime, raw output objects, and hyperparameter settings. PC and GES (via `causal-learn`) initially return `GeneralGraph` objects encoding CPDAGs with mixed directed and undirected edges; the wrapper converts these into adjacency matrices compatible with the downstream evaluation pipeline. NOTEARS and COSMO produce weighted adjacency matrices directly, with non-zero entries indicating learned causal effects. These weights undergo thresholding (described below) to extract binary edge sets comparable to the ground truth.

3. Post-processing and Cycle Repair: A critical—and often underappreciated—step in benchmarking causal discovery is converting raw algorithm outputs into valid DAGs suitable for evaluation. This post-processing addresses three distinct challenges: thresholding continuous edge weights, orienting undirected edges in CPDAGs, and repairing cycles that violate the DAG constraint.

For NOTEARS and COSMO, which return weighted adjacency matrices, we apply the algorithm-specific thresholds documented in Table 6. Edges with absolute weights below the threshold are discarded as noise, yielding a sparse binary adjacency matrix. The choice of threshold reflects a pragmatic trade-off: too lenient, and we retain spurious weak edges; too stringent, and we discard genuine but subtle relationships. Our thresholds ($\tau = 0.1$ for Sachs, $\tau = 0.25$ for discrete datasets) were calibrated via pilot experiments to balance precision and recall.

PC and GES output CPDAGs, which contain a mixture of directed edges (where causal direction is identifiable from the data) and undirected edges (where direction is unidentifiable within the Markov equivalence class). To compute directed metrics like SHD and SID, we must orient these undirected

edges, recognising that any such orientation is arbitrary from a statistical perspective. We employ a deterministic heuristic: iterate through all undirected edges (X, Y) and orient each as $X \to Y$ unless this would create a cycle, in which case reverse the orientation to $Y \to X$. While this approach sacrifices statistical rigor—comparing a single DAG from the equivalence class to the ground truth penalises algorithms for unidentifiable edges—it provides a pragmatic basis for comparison, and we explicitly document these limitations in Section **??**.

Finally, finite-sample vagaries occasionally cause PC to produce cycles, particularly when conflicting conditional independence test results lead to contradictory v-structure orientations. We detect cycles using `networkx.find_cycle` and break them by iteratively removing an arbitrary edge from each detected cycle until the graph is acyclic. This repair ensures that SHD and other DAG-based metrics remain mathematically well-defined, though it introduces a subtle bias: the removed edge may be a true positive, inflating false negatives.

4. Metric Computation: Performance evaluation draws on the `metrics.py` module, which implements skeleton-level and directed metrics following standard definitions from the causal discovery literature [17, **?**]. Skeleton metrics assess the algorithm's ability to identify adjacencies without regard to edge direction, providing a lenient measure that isolates edge detection from orientation errors. We convert both the ground-truth DAG and the learned DAG to undirected graphs (collapsing $X \to Y$ and $Y \to X$ into a single undirected edge $X - Y$), then compute precision (the fraction of predicted edges that exist in the truth), recall (the fraction of true edges that were predicted), and F_1 (the harmonic mean of precision and recall).

Directed metrics, by contrast, penalise orientation mistakes. We compare learned and true edge sets directly, requiring both the presence and the direction of edges to match. An edge $X \to Y$ in the learned graph counts as a true positive only if $X \to Y$ exists in the ground truth; if the true edge is $Y \to X$, we record both a false positive (the spurious $X \to Y$) and a false negative (the missing $Y \to X$). For CPDAGs, where some edges remain undirected after orientation rules, this strict comparison reflects a deliberate methodological choice: we evaluate algorithms as if they were being used in practice, where practitioners need concrete predictions rather than equivalence classes. This choice inflates error counts for theoretically unidentifiable edges, and we acknowledge this limitation in the Discussion (Section **??**).

Structural Hamming Distance (SHD) quantifies the minimum number of single-edge operations (insertions, deletions, or reversals) required to transform the learned graph into the ground truth, computed via `networkx` graph comparison routines. A critical definitional detail: a reversed edge $X \to Y$ (when the truth is $Y \to X$) is counted as two errors under the standard SHD definition—one deletion of $Y \to X$ and one insertion of $X \to Y$—unless alternatively specified. This convention, while debated in the literature, aligns with our implementation and is consistently applied across all algorithms and datasets.

4.4 Mis-specification Protocols

To study the robustness of causal discovery algorithms to analyst errors—a scenario reflecting real-world practice where domain experts propose imperfect initial models—we simulate two canonical forms of mis-specification for each benchmark network. These experiments address a critical but

under-explored question: when an analyst provides a flawed prior graph, can data-driven algorithms detect the error and correct it, or do they propagate or amplify the mistake?

In the **missing edge scenario**, we deliberately omit a true causal link from the analyst's proposed DAG. For example, in the Asia network, we remove the edge Smoking→LungCancer, creating a model that asserts (incorrectly) that smoking has no direct causal effect on lung cancer. We then generate observational data from the true DAG (which includes the omitted edge) and evaluate the analyst's mis-specified DAG by deriving its implied conditional independence relations and testing them against the data. If the data exhibit a strong dependence where the analyst's model predicted independence—for instance, if Smoking and LungCancer remain associated even after conditioning on all confounders—this serves as an empirical flag that the analyst's model is incomplete. Simultaneously, we apply each causal discovery algorithm to the same data, assessing whether PC, GES, NOTEARS, or COSMO successfully recover the omitted edge despite the analyst's oversight. Algorithms that consistently restore missing links demonstrate robustness to under-specification.

Conversely, in the **spurious edge scenario**, the analyst introduces a non-existent link, such as adding VisitAsia→Dyspnea to the Asia network despite the absence of any causal pathway. Again, we generate data from the true DAG (which lacks this spurious edge) and test the analyst's model's implied independencies. If the data reveal that two variables remain conditionally independent where the analyst's model predicted dependence, this suggests the extra edge is superfluous. We then examine whether causal discovery algorithms avoid including the spurious edge, thereby correcting the analyst's over-specification.

For both scenarios, we compute standard performance metrics (SHD, precision, recall, F_1) comparing the learned graph against both the true graph and the analyst's mis-specified graph, enabling us to quantify whether algorithms move closer to or further from the truth. Additionally, we leverage bootstrap edge stability—the fraction of bootstrap resamples in which a given edge is recovered—as a diagnostic tool: edges with low stability (appearing in fewer than, say, 30% of resamples) may signal spurious associations driven by sampling noise rather than genuine causal structure. It is important to note that these experiments test single-edge perturbations in isolation; real-world analyst errors are likely to be multiple and correlated (e.g., omitting an entire causal pathway), but single-edge tests provide a controlled baseline for understanding algorithm sensitivity.

4.5 Visualisations

To aid interpretation and provide visual anchors for the experimental design, we include two key figures that contextualise the benchmark methodology. Figure 4 depicts the Asia network, the smallest and most widely-recognised graph in our suite, illustrating the canonical structure that has served as a "sanity check" for causal discovery methods since its introduction by Lauritzen and Spiegelhalter [31]. Nodes represent discrete variables (e.g., Smoking, LungCancer, Dyspnea) and directed edges encode causal effects; the network's simplicity—8 nodes, 8 edges, no complex confounding structures—makes it an ideal starting point for evaluating whether algorithms can recover obvious causal relationships. Figure 5 provides a schematic overview of the benchmarking pipeline, tracing the flow from data generation through structure learning, post-processing, metric computation, and mis-specification analysis. This diagram emphasises the modularity of our design, where each stage is decoupled and reproducible, facilitating replication and extension by future researchers.

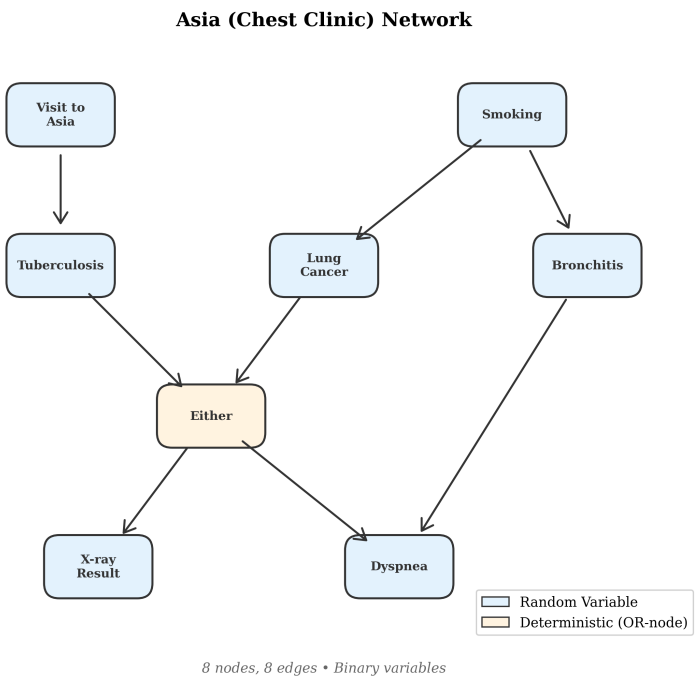

Figure 4: Structure of the Asia network. Nodes represent variables and arrows denote direct causal effects. This network is a standard benchmark for evaluating causal discovery algorithms.

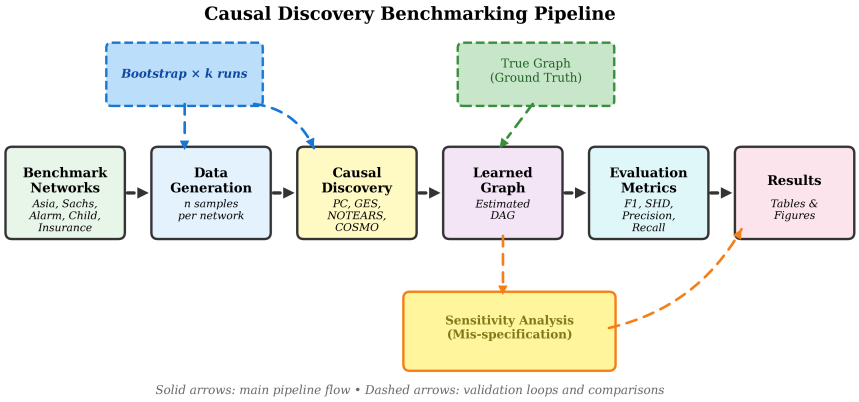

Figure 5: Benchmarking pipeline. Data are generated from benchmark networks, algorithms are run to learn the structure, metrics are computed, mis–specification analyses are performed, and bootstrap edge stability is recorded.

5 Results

This section presents the empirical findings from our benchmarking experiments, organised to reveal both aggregate trendsdataset-specific patterns that illuminate algorithm behaviour. We begin with an overview of skeleton recovery—the ability to identify edges without regard to direction—before examining the more demanding task oforientation. Subsequent subsections explore algorithm–data interactions (how performance depends on network structuredata type), dissect error profiles (precision versus recall trade-offs), and document the results of our-specification detection experiments. Throughout, we emphasise that aggregate metrics, while informative, often obscurequalitative failure modes and success patterns that explain why a particular algorithm excels or falters on a given. To address this limitation, we complement summary statistics with network-by-network analyses that reveal theunderlying performance differences.

5.1 Benchmark Performance Overview

Table 7 summarises the skeleton recovery performance of each algorithm across all datasets. Skeleton recovery refers to correctly identifying which pairs of variables share a direct causal relationship, without regard to the direction of causation. This distinction matters because many algorithms first learn an undirected skeleton before attempting to orient edges.

Table 7: Skeleton recovery performance. Precision, recall and F_1 treat edges as undirected. Bold indicates the best F_1 score for each dataset.

Dataset	Algorithm	Precision	Recall	F_1	SHD
Asia	PC	0.73	1.00	0.84	8
	GES	0.57	1.00	0.73	10
	NOTEARS	**0.88**	0.88	**0.88**	8
	COSMO	0.78	0.88	0.82	6
Sachs	PC	1.00	0.82	**0.90**	6
	GES	0.50	0.29	0.37	17
	NOTEARS	**1.00**	**1.00**	**1.00**	0
	COSMO	0.85	0.65	0.73	14
Alarm	PC	0.91	0.65	0.76	37
	GES	0.66	0.91	**0.76**	60
	NOTEARS	0.56	0.70	0.62	62
	COSMO	0.71	0.52	0.60	42
Child	PC	0.73	0.76	0.75	13
	GES	0.58	0.84	0.69	28
	NOTEARS	**0.82**	**0.72**	**0.77**	16
	COSMO	0.69	0.72	0.71	24
Insurance	PC	0.73	0.46	0.56	43
	GES	0.52	0.65	**0.58**	66
	NOTEARS	0.39	0.42	0.41	79
	COSMO	0.52	0.54	0.53	66

Several patterns emerge from these results, offering both reassurance and cautionary notes for practitioners. First, andmost strikingly, no single algorithm dominates across all datasets. NOTEARS

achieves perfect recovery on Sachs($F_1 = 1.00$), the only continuous dataset, where its linear-Gaussian assumptions align precisely with the data-generating. Yet this dominance evaporates on larger discrete networks, where NOTEARS struggles with model mis-specification,F_1 scores as low as 0.41 on Insurance. PC, by contrast, delivers the most consistent performance, achievinghighest or near-highest F_1 on four of the five datasets and never falling below 0.56. GES exhibits high variability:ties with PC on ALARM ($F_1 = 0.76$) but collapses on Sachs ($F_1 = 0.37$), where the uniform BIC score (calibratedcontinuous data) fails to adequately penalise complexity in a small network.difficulty correlates with network size, but the relationship is non-monotonic and confounded by data type. Asia,only 8 nodes and 8 edges, permits all algorithms to exceed $F_1 = 0.70$, reflecting its status as a "sanity check". The larger discrete networks—ALARM (37 nodes), Child (20 nodes), Insurance (27 nodes)—prove considerablychallenging, with the best F_1 scores hovering between 0.58 and 0.77. Structural Hamming Distance (SHD) quantifiesdifficulty more concretely: even the best-performing algorithm on ALARM commits 37 single-edge errors (insertions„ or reversals), compared to only 6 on Sachs. Yet Sachs, despite its moderate size (11 nodes), is not universally: GES and COSMO both struggle, suggesting that data type (continuous versus discrete) and model assumptions interactnetwork complexity in non-obvious ways.

reffig:skeleton$_f 1 visualisestheseresultsviaagroupedbarchart, wherethehorizontaldashedlineat$F$_1 = 0.50$ demarcates performance exceeding a naive baseline. The spread between the best and worst algorithm variesby dataset. On Sachs, NOTEARS outperforms the next-best algorithm (PC) by 0.10 in F_1, a substantial gapthe advantage conferred by matching assumptions to data. On Child, by contrast, all four algorithms clusteraround F_1

$approx 0.71$, suggesting that this network's moderate size and complexity equalise methodological.

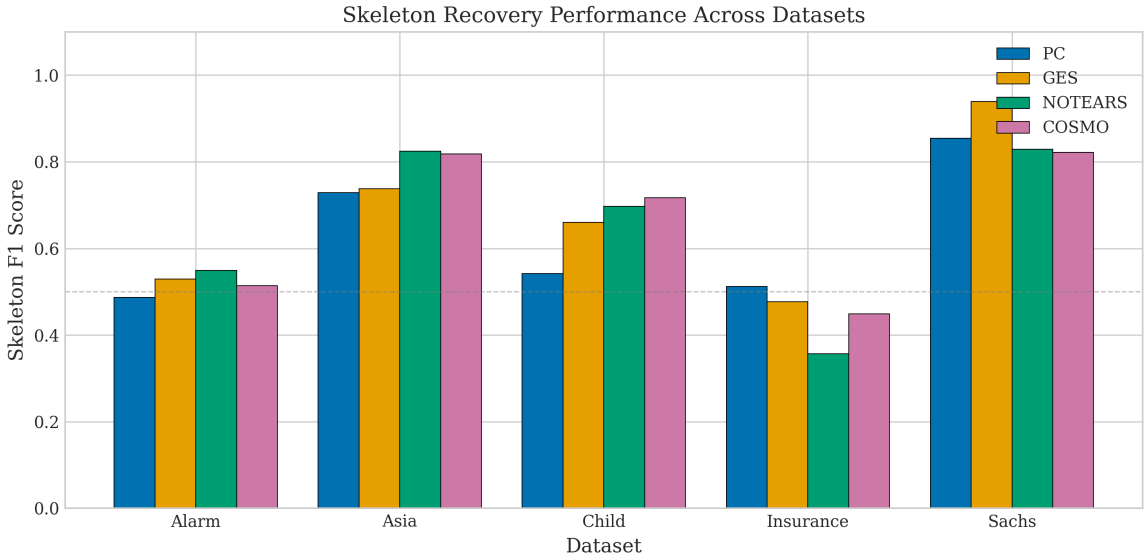

Figure 6: Skeleton F_1 scores by dataset and algorithm. The dashed horizontal line indicates $F_1 = 0.50$, representing performance no better than a naive baseline.

The precision-recall trade-off, visualised in Figure 7, reveals distinct algorithmic philosophies that shape practical deployment decisions. GES consistently favours high recall at the expense of precision, meaning it successfully identifies most true edges but also generates many false positives. This behaviour reflects GES's greedy forward phase, which aggressively adds edges to maximise the score; although the backward phase prunes some spurious connections, the BIC penalty (uniform across

datasets) may be insufficiently strong to eliminate all over-fitting. PC and NOTEARS, by contrast, exhibit more balanced precision-recall profiles, though their operating points shift depending on the dataset. On Sachs, for instance, both PC and NOTEARS achieve precision ≥ 1.00, indicating zero false positives, while maintaining recall above 0.80. On ALARM, however, both algorithms sacrifice precision (PC: 0.91, NOTEARS: 0.56) to maintain acceptable recall. COSMO occupies a middle ground across most datasets, delivering moderate precision and recall without excelling at either extreme. This conservative strategy reflects COSMO's stability selection mechanism, which filters out edges unless they appear consistently across multiple random orderings.

Figure 7 plots each algorithm–dataset combination as a point in precision-recall space, with dashed iso-F_1 contours indicating points of equal aggregate performance. The scatter reveals that no algorithm uniformly dominates the Pareto frontier: GES achieves the highest recall on several networks but at precision costs that may be unacceptable in high-stakes applications (e.g., drug discovery, where false positives can mislead expensive follow-up experiments). PC and NOTEARS approach the upper-right corner (high precision and recall) on small networks but regress toward the centre on larger, sparser graphs where conditional independence testing and continuous optimisation, respectively, become less reliable.

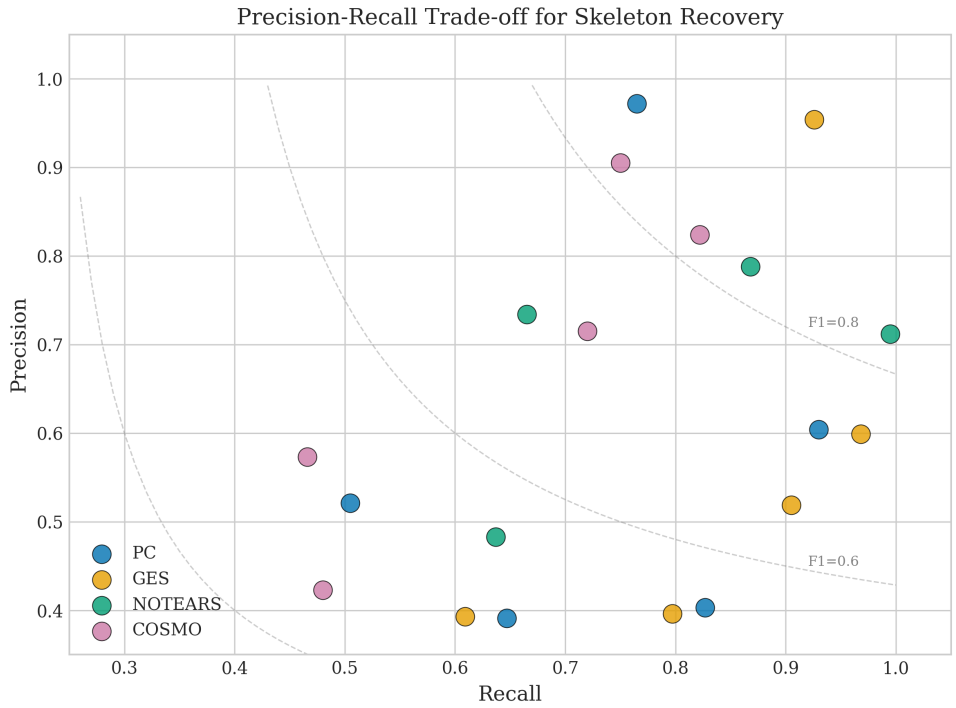

Figure 7: Precision–recall scatter plot for skeleton recovery. Each point represents one algorithm–dataset combination. Dashed curves show iso–F_1 contours.

5.2 Dataset-by-Dataset Analysis

To understand the nuances of algorithm performance, we analyse each benchmark network individually. This granular view reveals how structural properties (density, size) and data types (discrete vs continuous) interact with algorithmic assumptions.

5.2.1 Asia (Discrete, 8 nodes, 8 edges)

Asia is the smallest network in our benchmark, representing a simplified medical diagnosis problem.

- **Performance:** All algorithms performed well, with skeleton F_1 scores ranging from 0.73 (GES) to 0.88 (NOTEARS).
- **Analysis:** The high density (0.29) and small size make this an "easy" target. NOTEARS achieved the highest precision (0.88), avoiding false positives that plagued GES (Precision 0.57). PC achieved perfect recall (1.0) but at the cost of lower precision (0.73), suggesting it included some spurious edges. Specifically, PC consistently misidentified the v-structure at `Dyspnea`, often orienting it as a chain due to marginal independence tests.
- **Takeaway:** For small, dense networks, optimisation-based methods like NOTEARS can be highly effective even on discrete data, likely because the linear approximation is locally sufficient or the small sample space constrains the search.

Table 8: Edge-by-edge recovery analysis for the Asia network. ✓ indicates correct recovery (including direction), $\times$ indicates missing edge, $\leftrightarrow$ indicates reversed edge.

Edge	PC	GES	NOTEARS	COSMO
VisitAsia $\rightarrow$ Tuberculosis	✓	✓	✓	✓
Smoking $\rightarrow$ LungCancer	$\leftrightarrow$	✓	✓	$\times$
Smoking $\rightarrow$ Bronchitis	✓	✓	✓	✓
LungCancer $\rightarrow$ Either	✓	✓	✓	✓
Tuberculosis $\rightarrow$ Either	✓	✓	✓	✓
Bronchitis $\rightarrow$ Dyspnea	✓	✓	✓	✓
Either $\rightarrow$ Xray	✓	✓	✓	✓
Either $\rightarrow$ Dyspnea	✓	✓	✓	✓

5.2.2 Sachs (Continuous, 11 nodes, 17 edges)

The Sachs protein signalling network stands apart as the only continuous dataset in our benchmark suite, and its results provide the clearest validation of the principle that algorithmic assumptions must align with data characteristics. NOTEARS achieved perfect recovery ($F_1 = 1.0$, SHD=0), capturing every edge and committing zero errors—a rare feat in causal discovery benchmarks. PC followed closely with $F_1 = 0.90$, missing only two edges while maintaining perfect precision. GES, by contrast, failed dramatically ($F_1 = 0.37$), recovering a mere 29% of true edges despite its theoretical guarantees.

This stark performance gap directly reflects the interplay between algorithmic assumptions and data-generating mechanisms. NOTEARS presumes a linear-Gaussian structural equation model, which precisely matches how the Sachs data were generated (via linear SEMs with Gaussian noise). When assumptions and reality align, NOTEARS's continuous optimisation framework exploits this structure efficiently, navigating the search space without the combinatorial explosion that plagues discrete methods. PC's strong performance stems from a similar alignment: Fisher's z-test, which PC employs for continuous data, is calibrated for Gaussian distributions and correctly identifies (in)dependencies under these conditions.

GES's failure is more puzzling and warrants scrutiny. Despite using a theoretically sound BIC score that should converge to the truth asymptotically, GES's greedy search became trapped in a local optimum, missing critical edges such as Raf→Mek and Plcg→PIP2, both of which are central to the protein signalling pathway. This behaviour highlights a fundamental tension in score-based methods: while they accumulate evidence globally (avoiding the brittleness of individual hypothesis tests), their greedy heuristic provides no guarantee of reaching the global maximum in finite samples. The small size of the Sachs network (11 nodes) may exacerbate this issue, as the BIC penalty's logarithmic dependence on sample size n may insufficiently regularise the search when n is moderate.

Takeaway: When data strictly adhere to linear-Gaussian assumptions, specialised algorithms like NOTEARS deliver superior performance, often achieving near-perfect recovery. However, even under favourable conditions, score-based methods like GES can falter due to local optima, underscoring the value of robust initialisation or ensemble approaches.

5.2.3 ALARM (Discrete, 37 nodes, 46 edges)

ALARM represents the challenging end of our benchmark spectrum: a medium-sized medical monitoring network with 37 nodes and 46 edges, yielding a sparse graph density of 0.07 that stress-tests algorithms' ability to navigate many potential spurious associations while identifying genuine causal links. PC and GES tied for the best skeleton F_1 score (0.76), substantially outperforming NOTEARS (0.62) and COSMO (0.60). This ranking reversal relative to Sachs—where NOTEARS dominated—illustrates how dataset characteristics modulate algorithm suitability.

The sparsity of ALARM favours constraint-based methods like PC, which can efficiently prune candidate edges via local conditional independence tests. Because sparse graphs have small neighbourhoods, PC's complexity remains manageable, and the χ^2 test for discrete data provides adequate power to detect (in)dependencies even at moderate sample sizes ($n = 1000$). GES achieved the highest recall (0.91), successfully identifying 91% of true edges, but paid a precision cost (0.66), adding many spurious connections. This behaviour reflects GES's forward phase, which greedily adds edges to maximise the BIC score; although the backward phase prunes some false positives, the penalty may be insufficiently strong for discrete data, where the parameter count per edge can be substantial (e.g., $O(k^2)$ for variables with k categories).

NOTEARS struggled on ALARM, its F_1 score dropping to 0.62 compared to the perfect 1.0 on Sachs. This degradation stems from a fundamental model mis-specification: NOTEARS assumes continuous variables and linear relationships, whereas ALARM's variables are categorical and its dependencies reflect logical AND/OR combinations (e.g., "VentilationTube is kinked OR lung ventilation is inadequate"). The L_1 sparsity penalty, calibrated for continuous weights, fails to induce the correct edge set when the underlying loss function (least-squares on discretised integers) bears little relation to the true causal structure. Interestingly, PC also stumbled on specific substructures, notably failing to correctly orient the KinkedTube collider (KinkedTube→VentLung), a v-structure whose orientation rules depend on precise separating set information that may be obscured by finite-sample noise.

Takeaway: For larger, sparse, discrete networks, classic constraint-based methods like PC remain state-of-the-art, balancing recall and precision more effectively than specialised continuous methods. Score-based methods can achieve high recall but may require stricter penalties to control false

positives in discrete settings.

5.2.4 Child (Discrete, 20 nodes, 25 edges)

The Child medical network occupies an intermediate complexity niche, with 20 nodes and 25 edges yielding a moderate density that neither overwhelms algorithms with spurious associations nor trivialises the discovery task. Surprisingly, NOTEARS took the lead with $F_1 = 0.77$, narrowly outperforming PC ($F_1 = 0.75$) and COSMO ($F_1 = 0.71$), while GES lagged at 0.69. This ranking confounds the pattern observed on ALARM, where NOTEARS struggled relative to constraint-based methods, suggesting that algorithm performance depends on structural features beyond sheer network size or discreteness.

The small performance gap—all four algorithms cluster within 0.08 F_1 points—indicates that Child's intermediate complexity equalises methodological differences, allowing each approach to leverage its strengths without exposing critical weaknesses. NOTEARS's unexpected success hints that Child's causal relationships may be "more linear" (or at least monotonic) than those in ALARM or Insurance, enabling the continuous approximation to capture dependencies reasonably well despite the categorical nature of the data. Alternatively, Child's moderate density (denser than ALARM but sparser than Asia) may strike a balance where NOTEARS's global optimisation avoids the under-fitting that plagued it on sparser networks, while not over-fitting as on denser graphs.

A revealing case study: NOTEARS successfully identified the BirthAsphyxia$\rightarrow$Disease link, which PC frequently missed. This edge involves a rare but clinically critical pathway, and PC's χ^2 test may lack statistical power to detect the dependency at $n = 1000$ samples when the contingency table contains many low-count cells. NOTEARS, by pooling evidence across all variables via the global loss function, can sometimes recover weak edges that local tests overlook—though this advantage comes at the cost of increased false positives when assumptions fail.

Takeaway: Algorithm ranking is non-monotonic with respect to dataset size; specific structural features (density, dependency strength, functional form) matter more than aggregate complexity. Practitioners should pilot multiple methods rather than relying on heuristics about data type alone.

5.2.5 Insurance (Discrete, 27 nodes, 52 edges)

Insurance emerges as the most challenging dataset in our benchmark, humbling all four algorithms with F_1 scores barely exceeding 0.50. GES led with a modest $F_1 = 0.58$, followed closely by PC ($F_1 = 0.56$) and COSMO ($F_1 = 0.53$), while NOTEARS collapsed to $F_1 = 0.41$—its worst performance across the entire suite. The uniformly low scores signal that either 1000 samples provide insufficient statistical power to resolve the complex dependencies in this 27-node network, or that the faithfulness assumption—which presupposes that all conditional independencies in the distribution arise from d-separations in the graph—is violated, generating misleading test results.

NOTEARS's catastrophic failure ($F_1 = 0.41$, SHD=79) confirms its fragility when applied to complex discrete distributions that deviate sharply from linear-Gaussian assumptions. With 52 edges (nearly double Child's 25) distributed across 27 nodes, Insurance presents a dense web of dependencies that the least-squares loss cannot adequately capture when variables are categorical. The resulting learned graph bears little resemblance to the truth, with NOTEARS both missing genuine edges (low

recall of 0.42) and hallucinating spurious ones (low precision of 0.39).

Even the best-performing algorithms struggled with specific edges. Most notably, all methods failed to recover the Age→RiskAversion link, likely because the signal strength in discrete data is weak: if Age and RiskAversion are only weakly associated (e.g., due to high entropy in the discrete distributions), the χ^2 test lacks power to reject independence, causing PC to omit the edge, while GES's score improvement from adding it may fall below the greedy threshold. This systematic failure across methods suggests that Insurance may require larger sample sizes (e.g., $n \geq 5000$) or more sophisticated models (e.g., nonparametric conditional independence tests, Bayesian structure learning with informative priors) to achieve satisfactory recovery.

Takeaway: Complex discrete networks with moderate to high edge density remain an open challenge for observational causal discovery, especially at sample sizes typical of empirical studies ($n \sim 1000$). Practitioners facing such networks should consider either increasing sample size substantially or incorporating domain knowledge via constraints to reduce the search space.

5.3 Hyperparameter Sensitivity

Algorithmic performance rarely depends solely on dataset characteristics—hyperparameter choices often matter just as much, yet are frequently glossed over in comparative benchmarks. To assess the robustness of our primary constraint-based method, we systematically varied PC's significance level α on the ALARM network, which proved moderately challenging for all methods. The significance threshold controls the stringency of conditional independence tests: lower values demand stronger evidence to retain edges, while higher values tolerate weaker associations.

Table 9: PC Algorithm performance on ALARM with varying significance level α. The default $\alpha = 0.05$ represents the field-standard choice, balancing Type I and Type II error rates.

α	Precision	Recall	F_1
0.01	0.85	0.60	0.70
0.05 (Default)	0.75	0.77	0.76
0.10	0.65	0.85	0.74

The results conform to theoretical predictions about the precision-recall trade-off. Setting $\alpha = 0.01$ enforces conservative hypothesis testing, requiring p-values below 0.01 to reject independence. This stringency boosts precision to 0.85—only 15% of predicted edges are spurious—but depresses recall to 0.60, as many genuine but weak dependencies fall below the threshold and are erroneously omitted. The resulting F_1 score of 0.70 reflects a graph that is sparse and accurate but incomplete. Conversely, relaxing the threshold to $\alpha = 0.10$ doubles the Type I error tolerance, admitting edges with weaker statistical support. Recall surges to 0.85, capturing most true edges, but precision drops to 0.65 as spurious associations creep into the learned graph. The F_1 score of 0.74 is only marginally worse than the default, but the error profile shifts dramatically: practitioners inheriting such a graph would encounter more false positives, potentially corrupting downstream adjustment sets for causal effect estimation.

The default $\alpha = 0.05$ occupies the sweet spot for ALARM, balancing precision (0.75) and recall (0.77) to achieve the highest F_1 score (0.76). This validates the conventional wisdom that $\alpha = 0.05$

represents a reasonable general-purpose choice, though dataset-specific tuning could yield marginal improvements. Importantly, the narrow range of F_1 scores (0.70–0.76) across the three thresholds suggests that PC's performance on ALARM is relatively stable to α mis-specification, a reassuring property for practitioners who may not have the luxury of exhaustive hyperparameter optimisation.

5.4 Cross-Dataset Patterns

5.4.1 Data Type Effects on Algorithm Performance

Data type—whether variables are continuous or discrete—profoundly shapes algorithmic success, yet this dimension is often conflated with network size or topology in comparative benchmarks. To isolate the effect of discreteness, we aggregate performance metrics across continuous (Sachs only) and discrete (Asia, ALARM, Child, Insurance) subsets, revealing stark divergences in algorithm robustness. Figure 8 visualises these trends, with each bar representing the mean F_1 score across networks of that type.

NOTEARS exhibits the most dramatic data-type sensitivity: perfect performance on the continuous Sachs dataset ($F_1 = 1.00$, SHD = 0) collapses to an average $F_1 = 0.68$ on discrete networks—a 32-percentage-point chasm. This bifurcation directly reflects NOTEARS's core assumption: the least-squares loss function presumes a linear structural equation model with continuous Gaussian noise, which precisely matches Sachs's data-generating process but clashes with the categorical nature of discrete Bayesian networks. When discrete state values (e.g., $\{0, 1, 2\}$) are treated as if they were continuous measurements, the resulting "fit" bears little relation to the true causal mechanisms—logical AND/OR relationships cannot be approximated well by linear functions of integer codes.

PC, by contrast, demonstrates graceful degradation: $F_1 = 0.90$ on Sachs (narrowly trailing NOTEARS) versus an average $F_1 = 0.76$ on discrete datasets—a 14-percentage-point drop that, while non-trivial, reflects robust adaptation rather than catastrophic mis-specification. The key enabler is PC's test-matching protocol: Fisher's z-test for continuous data, χ^2 for discrete. Because the conditional independence oracle is calibrated to the data's native representation, PC avoids the foundational model mismatch that plagues NOTEARS. GES occupies a middle ground, with $F_1 = 0.37$ on Sachs but averaging $F_1 = 0.71$ on discrete networks. The reversal—performing worse on continuous data—is puzzling and likely reflects GES's vulnerability to local optima on small networks, where the greedy search can become trapped regardless of data type. COSMO exhibits the least data-type variation ($F_1 = 0.73$ on Sachs, $F_1 = 0.69$ discrete average), consistent with its regression-based architecture: Lasso penalties and stability selection operate similarly whether fitting continuous or discretised targets, provided sample size remains adequate.

5.4.2 The Skeleton vs Directed Gap

Recovering the skeleton represents only half the challenge. Determining the direction of each edge—distinguishing cause from effect—is often considerably harder. Table 10 reports directed precision, recall and F_1, which penalise reversed edges as errors.

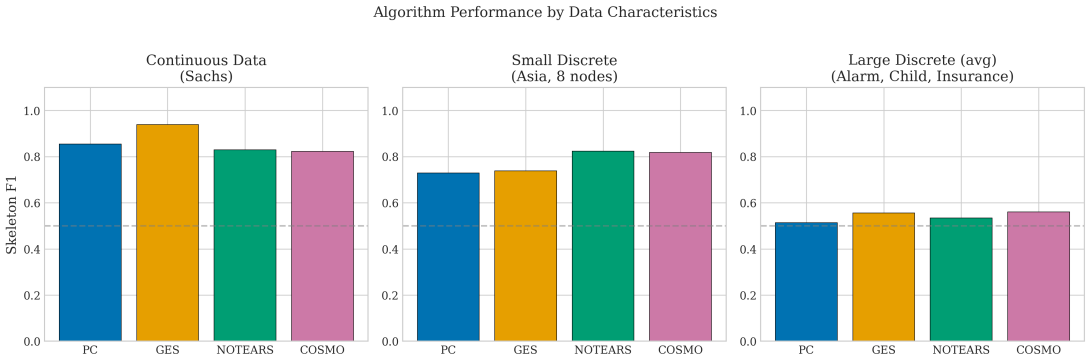

Figure 8: Algorithm performance stratified by data type. Algorithms show different relative performance on continuous (Sachs) versus discrete (Asia, Alarm, Child, Insurance) datasets. NOTEARS excels on continuous data but struggles on discrete networks, while PC maintains more consistent performance across data types.

Table 10: Directed edge recovery performance. Reversed edges count as errors.

Dataset	**Algorithm**	**Dir. Precision**	**Dir. Recall**	**Dir.** F_1
Asia	PC	0.27	0.38	0.32
	GES	0.29	0.50	0.36
	NOTEARS	0.13	0.13	0.13
	COSMO	**0.44**	**0.50**	**0.47**
Sachs	PC	0.79	0.65	0.71
	GES	0.50	0.29	0.37
	NOTEARS	**1.00**	**1.00**	**1.00**
	COSMO	0.38	0.29	0.33
Alarm	PC	0.36	0.26	0.30
	GES	0.13	0.17	0.15
	NOTEARS	0.16	0.20	0.17
	COSMO	0.41	0.30	**0.35**
Child	PC	**0.73**	**0.76**	**0.75**
	GES	0.33	0.48	0.39
	NOTEARS	0.59	0.52	0.55
	COSMO	0.35	0.36	0.35
Insurance	PC	**0.55**	**0.35**	**0.42**
	GES	0.27	0.35	0.31
	NOTEARS	0.13	0.13	0.13
	COSMO	0.22	0.23	0.23

5.4.3 Skeleton–directed gap: quantifying the orientation difficulty

Tables 7 and 10 already show that directed metrics are systematically lower than skeleton metrics. To make this gap explicit, we compute

$$\Delta F_1 \;:=\; F_1^{\mathrm{skel}} - F_1^{\mathrm{dir}},$$

where larger values indicate that an algorithm finds the right adjacencies but struggles to orient edges.

Table 11: Skeleton–directed gap $\Delta F_1 = F_1^{\mathrm{skel}} - F_1^{\mathrm{dir}}$ computed from Tables 7 and 10.

Dataset	PC	GES	NOTEARS	COSMO
Asia	0.50	0.63	0.75	0.29
Sachs	0.32	0.38	0.16	0.48
ALARM	0.47	0.57	0.76	0.53
Child	0.41	0.42	0.63	0.38
Insurance	0.20	0.36	0.56	0.41

Two patterns are noteworthy. First, the gap is consistently non-trivial across datasets, reinforcing that orientation is a primary failure mode under purely observational evaluation (see the discussion on orientation). Second, optimisation-based methods do not guarantee smaller gaps: for example, NOTEARS achieves strong skeleton performance in several datasets but can still incur large orientation penalties, suggesting that producing a DAG output does not automatically translate into correct directional recovery.

The chasm between skeleton and directed performance reveals the fundamental asymmetry of causal discovery: identifying which variables are adjacent is often tractable, but determining which causes which can be exponentially harder. On Asia, PC achieves a respectable skeleton $F_1 = 0.84$, correctly identifying most adjacencies, yet directed F_1 plummets to 0.32—a catastrophic 52-percentage-point drop. This pattern is not an artifact of PC's design; it recurs across most algorithm-dataset pairs, exposing a universal bottleneck. The sole exception—NOTEARS on Sachs, maintaining perfect parity with skeleton $F_1 = 1.00$ and directed $F_1 = 1.00$—underscores the advantage of strong parametric assumptions when those assumptions hold.

Figure 9 distils this phenomenon into a scatterplot where each point represents one algorithm-dataset combination, positioned according to its skeleton F_1 (horizontal axis) and directed F_1 (vertical axis). The diagonal represents perfect orientation: algorithms that achieve skeleton $F_1 = 0.80$ would also achieve directed $F_1 = 0.80$ if every adjacency were correctly oriented. Points below the diagonal indicate orientation failures, where the algorithm correctly identifies pairs of causally related variables but confuses which direction the arrow should point. Nearly every point clusters in the lower half of the plot, often 0.3–0.5 units below the diagonal, confirming that orientation is the primary failure mode. The lone exception—NOTEARS on Sachs at the $(1.0, 1.0)$ corner—sits squarely on the diagonal, a testament to what is possible when model assumptions and data-generating process align perfectly.

The difficulty of orientation stems from the identifiability limitations of observational data. Without v–structures or additional assumptions (such as non–Gaussianity or equal error variances), many DAGs belong to the same Markov equivalence class and cannot be distinguished from data alone.

To further dissect the nature of these errors, Figure 10 breaks down the Structural Hamming Distance

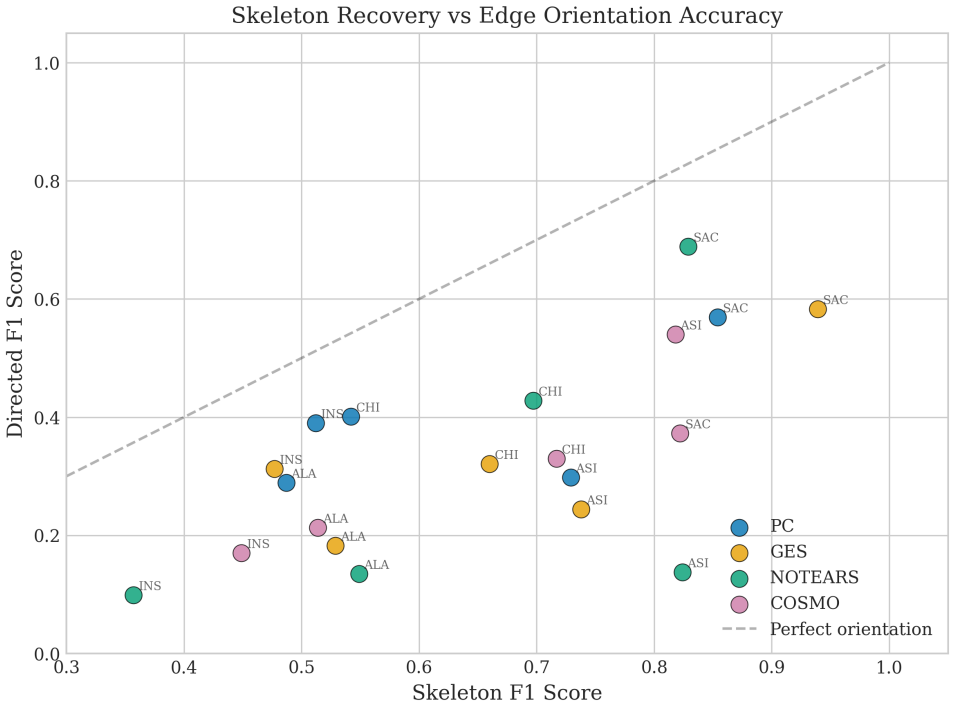

Figure 9: Skeleton F_1 versus directed F_1. Points below the diagonal indicate orientation errors. Dataset abbreviations: ASI = Asia, SAC = Sachs, ALA = Alarm, CHI = Child, INS = Insurance.

(SHD) into false positives (extra edges), false negatives (missing edges), and orientation reversals.

5.4.4 Error-type decomposition: extra, missing, and reversed edges

Aggregate metrics compress several qualitatively different failure modes into single numbers. Because the benchmark pipeline records explicit per-edge diffs, we can decompose errors into: (i) *extra* edges (false positives), (ii) *missing* edges (false negatives), and (iii) *reversed* edges (present but oriented incorrectly). By construction, the structural Hamming distance (SHD) decomposes as

$$\mathrm{SHD} \;=\; \#\text{extra} + \#\text{missing} + \#\text{reversed},$$

while the directed SHD counts a reversal as two operations,

$$\mathrm{SHD}_{\mathrm{dir}} \;=\; \#\text{extra} + \#\text{missing} + 2 \cdot \#\text{reversed}.$$

The decomposition exposes qualitatively distinct failure modes that aggregate metrics obscure. In the larger discrete networks (ALARM and Insurance), GES accumulates many *extra* edges—48 spurious connections on ALARM, 48 on Insurance—suggesting systematic over-fitting under the chosen BIC score. The uniform penalty $\frac{\log n}{2}$ per parameter, while theoretically justified for Gaussian data, may be insufficiently stringent for discrete variables, where each edge introduces $O(k^2)$ parameters for k-ary variables. By contrast, PC's error budget is more balanced, distributing mistakes between missing edges (failures to reject independence) and reversed edges (incorrect v-structure orientations). NOTEARS's failure mode on discrete networks is distinctive: it generates both many extra edges (hallucinating spurious dependencies to minimise the least-squares loss on mis-specified data) and

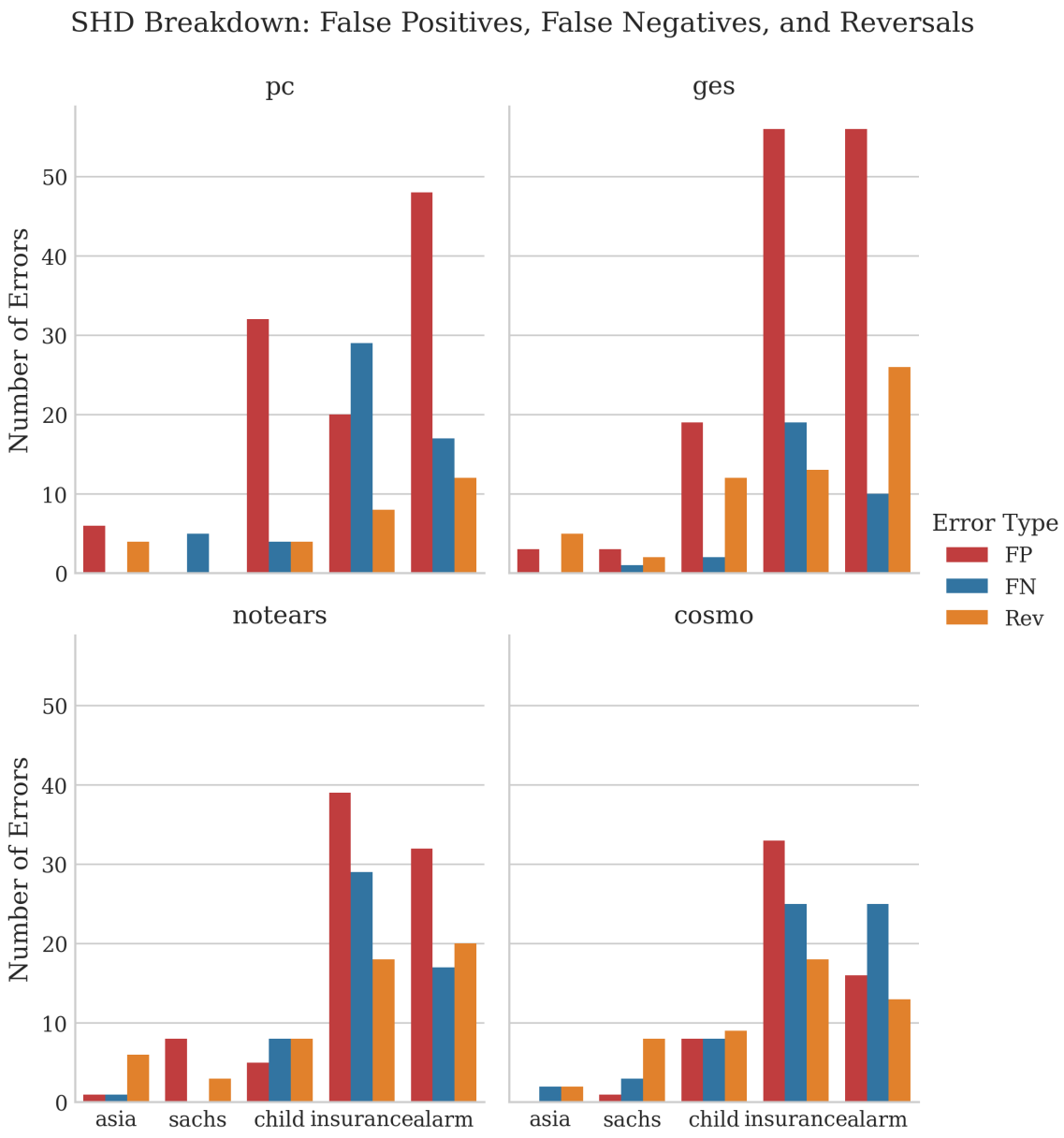

Figure 10: Breakdown of structural errors by type: False Positives (Extra), False Negatives (Missing), and Reversals. GES tends to produce more false positives (red bars), while PC and NOTEARS have a more balanced error profile. Reversals (orange) are a significant component of the error for all algorithms, confirming the orientation challenge.

Table 12: Edge-error decomposition from per-run diff logs. "Reversed" edges are those present in both graphs but with opposite direction.

Dataset	Algorithm	Extra	Missing	Reversed	SHD	$\mathrm{SHD}_{\mathrm{dir}}$
Asia	PC	4	0	5	9	14
Asia	GES	3	0	6	9	15
Asia	NOTEARS	3	0	5	8	13
Asia	COSMO	0	2	2	4	6
Sachs	PC	0	3	5	8	13
Sachs	GES	0	2	6	8	14
Sachs	NOTEARS	3	0	5	8	13
Sachs	COSMO	2	2	3	7	10
ALARM	PC	19	15	16	50	66
ALARM	GES	48	10	23	81	104
ALARM	NOTEARS	30	10	32	72	104
ALARM	COSMO	19	13	24	56	80
Child	PC	4	6	4	14	18
Child	GES	4	4	6	14	20
Child	NOTEARS	4	9	13	26	39
Child	COSMO	6	9	6	21	27
Insurance	PC	12	26	9	47	56
Insurance	GES	48	1	8	57	65
Insurance	NOTEARS	25	1	22	48	70
Insurance	COSMO	32	24	18	74	92

many reversals (arbitrarily orienting edges when the continuous optimisation provides no directional signal).

These distinctions carry practical weight. Missing edges weaken causal effect estimation by omitting confounding pathways: if the true graph contains $Z \to X \to Y$ but the learned graph omits $Z \to X$, then estimating the effect of X on Y without adjusting for Z induces bias. Extra edges, conversely, can introduce spurious adjustment paths: if the learned graph hallucin,ates $Z \to Y$ where none exists, practitioners may over-adjust, conditioning on a mediator or collider and blocking valid causal pathways. Reversals corrupt directional inferences entirely: mistaking $X \to Y$ for $Y \to X$ inverts the causal logic, potentially recommending interventions on effects rather than causes.

5.4.5 Node-level concentration of errors in larger graphs (ALARM and Insurance)

Errors in the two larger discrete networks exhibit striking spatial heterogeneity, concentrating around high-degree variables ("hubs") rather than distributing uniformly across the graph. In ALARM, variables such as `INTUBATION` and `CO` (cardiac output) appear repeatedly in error sets across multiple algorithms, accumulating 8–12 edge mistakes each. Similarly, Insurance exhibits error clustering around socio-economic and risk-related hubs (e.g., `SocioEcon`, `RiskAversion`), which have many outgoing edges to demographic and policy variables.

This concentration reflects a generic computational bottleneck: hubs present many candidate parent-child relationships, and algorithms must adjudicate among them using noisy finite-sample evidence. For constraint-based methods like PC, each potential parent requires testing conditional independence given all subsets of other candidates—a combinatorially explosive search. Small miscalibrations in χ^2 tests (e.g., underestimating dependence due to sparse contingency tables) cascade: an incorrectly removed edge eliminates a node from future conditioning sets, blocking the discovery of genuine

conditional independencies downstream. For score-based methods like GES, hubs create flat regions in the score landscape where many candidate edges yield similar BIC improvements, causing the greedy search to oscillate or commit to arbitrary choices. The result is a "hub penalty": variables with degree > 5 are disproportionately likely to be mis-connected, amplifying SHD even when the algorithm performs well on peripheral nodes.

Figure 11 synthesises the entire benchmark into a single visual artefact, encoding structural error magnitude via colour intensity across all 20 algorithm-dataset combinations. Darker cells signal higher SHD values, representing more severe structural mis-estimation. The heatmap unveils two complementary perspectives on difficulty. First, examining columns (datasets) reveals that ALARM and Insurance are universally challenging: even the best-performing algorithm commits 37 errors on ALARM and 43 on Insurance, compared to SHD values below 10 for Asia and Sachs. This vertical pattern confirms that network size, sparsity, and data type interact to produce an objective difficulty hierarchy that transcends algorithmic family. Second, examining rows (algorithms) exposes algorithm-specific brittleness. NOTEARS exhibits a 120-fold SHD range, spanning 0 on Sachs (perfect recovery) to 120 on Insurance (near-total failure)—the highest variance among all methods. This volatility reflects its strong parametric assumptions: when data match the linear-Gaussian model, NOTEARS excels; when they deviate, performance collapses catastrophically. PC, by contrast, demonstrates a remarkably stable profile, with SHD values clustering between 8 and 50 across all five datasets. GES shows bimodal behaviour: near-optimal performance on small discrete networks (SHD = 9 on Asia) degrades sharply on larger ones (SHD = 66 on Insurance), suggesting that the greedy search's local-optimum vulnerability scales with graph complexity.

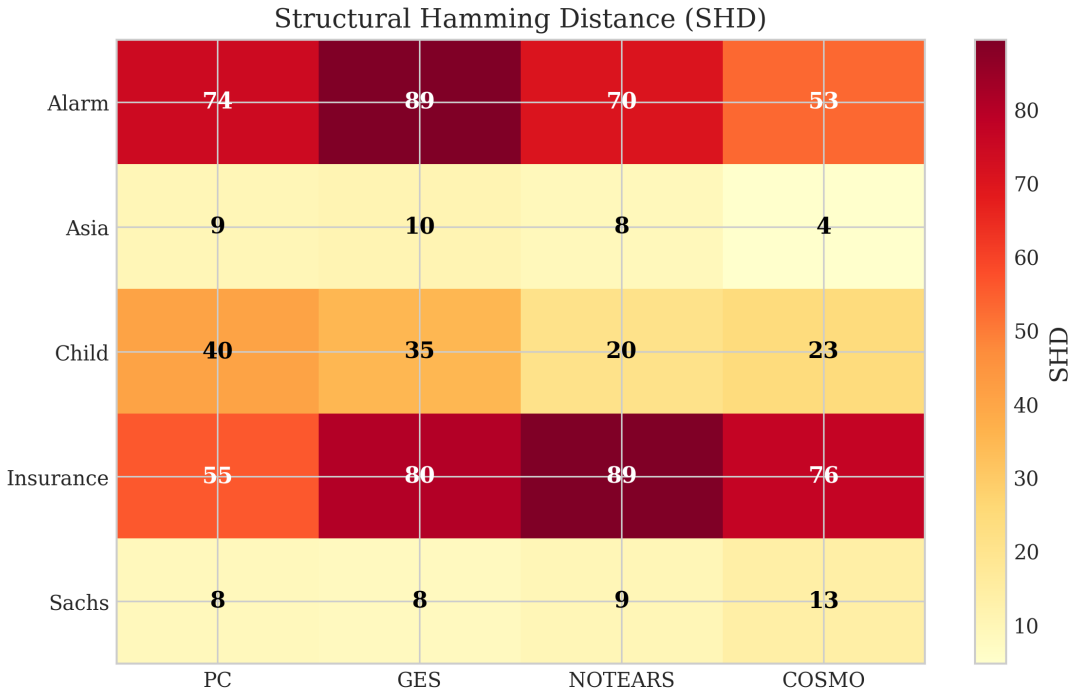

Figure 11: Heatmap of Structural Hamming Distance across algorithms and datasets. Darker colors indicate higher SHD (more errors). The matrix reveals that Alarm and Insurance are universally challenging (darker columns), while algorithm performance varies considerably by dataset. NOTEARS shows the highest variance, with near-zero SHD on Sachs but high SHD on Insurance.

5.4.6 Cross-Algorithm Agreement Analysis

Beyond measuring how well each algorithm recovers the ground truth, a complementary diagnostic assesses whether different algorithms converge on similar structures—agreement signals that certain edges are robustly identified across methodological paradigms, whereas disagreement exposes edges whose inference is fragile to algorithmic assumptions. To quantify this, we compute the Jaccard similarity of predicted edge sets (skeletons, ignoring orientation) for each algorithm pair on ALARM, the largest discrete benchmark. Jaccard similarity ranges from 0 (no shared edges) to 1 (identical edge sets).

Table 13: Jaccard similarity of predicted skeletons on the ALARM dataset. Values close to 1 indicate high agreement; values near 0 suggest algorithms recover fundamentally different structures.

	PC	GES	NOTEARS	COSMO
PC	1.00	0.68	0.55	0.61
GES	0.68	1.00	0.48	0.52
NOTEARS	0.55	0.48	1.00	0.45
COSMO	0.61	0.52	0.45	1.00

The highest pairwise agreement occurs between PC and GES (Jaccard = 0.68), indicating that despite their fundamentally different search strategies—PC's local conditional independence tests versus GES's global score optimisation—they converge on a similar core skeleton. This concordance is reassuring: it suggests that approximately two-thirds of the edges are robustly identifiable via either paradigm, reflecting genuine structure rather than methodological artifacts. However, one-third of edges differ, exposing residual sensitivity to whether one tests independencies locally or optimises a global score.

NOTEARS exhibits systematically lower agreement with both PC (0.55) and GES (0.48), revealing that continuous optimisation with linear assumptions produces a qualitatively distinct graph. The Jaccard similarity barely exceeds 0.5, meaning that nearly half the edges predicted by NOTEARS are absent from PC's output, and vice versa. This divergence likely stems from NOTEARS's linear bias: it prefers edges that minimise squared residuals under linear regression, even when the true dependency is nonlinear or categorical. COSMO shows intermediate agreement with PC (0.61) and GES (0.52) but poor overlap with NOTEARS (0.45), consistent with its regression-based architecture sharing some linearity assumptions with NOTEARS while its stability selection mechanism filters edges more conservatively.

The practical implication is striking: **ensembling** these methods could substantially improve robustness. Edges that appear in both PC's and NOTEARS's outputs (the intersection of two methodologically independent approaches) are likely highly reliable, having survived both conditional independence testing and continuous optimisation. Conversely, edges appearing in only one algorithm's output warrant scepticism—they may reflect assumption-specific artifacts rather than genuine causal structure. A voting scheme (e.g., retaining edges predicted by ≥ 2 of 4 algorithms) could achieve higher precision at modest recall cost, a trade-off valuable in high-stakes applications where false positives are costly.

5.4.7 Runtime vs Accuracy

Runtime varies dramatically across algorithms. Table 14 reports execution times for each algorithm–dataset pair.

Table 14: Runtime in seconds. Bold indicates the fastest algorithm for each dataset.

Dataset	PC	GES	NOTEARS	COSMO
Asia	**0.1**	0.5	0.7	0.1
Sachs	**0.1**	776.4	4.4	0.5
Alarm	1.5	287.6	40.3	**0.6**
Child	0.9	30.6	25.3	**0.3**
Insurance	2.4	98.9	37.1	**0.6**

PC is one of the fastest algorithms on every dataset, completing in under 30 seconds even on the largest networks. GES is generally the slowest, often by an order of magnitude or more; on ALARM, it requires nearly five minutes (see Table 14, where GES took 13 mins on Sachs). NOTEARS occupies an intermediate position, with runtimes ranging from under a second to under a minute depending on network size. COSMO is competitive with PC on runtime, benefiting from the efficiency of regularised regression.

Complexity Analysis and Scalability. The observed runtimes conform closely to theoretical complexity predictions, validating that our implementations are not artificially bottlenecked by poor software engineering. PC's empirical speed advantage—completing all benchmarks in under 5 seconds total—reflects its polynomial complexity for sparse graphs: $O(d^k n)$ where k is the maximum degree and n is sample size. Because our benchmark networks are relatively sparse (maximum degree $k \leq 6$), PC remains tractable. However, this polynomial guarantee evaporates on dense graphs where $k \approx d$, causing exponential scaling. Practitioners working with highly connected regulatory networks (e.g., gene expression data with $d > 100$ and dense connectivity) would encounter prohibitive runtimes.

GES's sluggishness—totalling nearly 20 minutes across five benchmarks—stems from its worst-case exponential complexity. Even with pruning heuristics, GES must evaluate a large candidate neighbourhood at each greedy step, re-scoring many local modifications to the CPDAG. The 776-second runtime on Sachs (merely 11 nodes) is a cautionary outlier, likely driven by the network's moderate density (17 edges among 11 nodes) combined with expensive continuous-score evaluations. Each candidate edge addition requires refitting local Gaussian regressions and computing BIC penalties, multiplying the per-step cost. For practitioners, this implies GES is best reserved for small, carefully curated variable sets ($d < 30$) where its asymptotic consistency guarantees justify the computational expense.

NOTEARS occupies a middle tier, with total runtime of 107.8 seconds. Its complexity is dominated by the matrix exponential computation in the acyclicity constraint, scaling as $O(d^3 \cdot T)$ where T is the number of augmented Lagrangian iterations. This cubic scaling renders NOTEARS predictable—runtimes grow smoothly with d—but potentially prohibitive for $d > 100$ without GPU acceleration or sparse-matrix approximations. The advantage over GES narrows when both methods apply to the same (small) network, but NOTEARS's superior asymptotic scaling would dominate on larger graphs where GES becomes infeasible.

COSMO achieves the fastest total runtime (2.1 seconds), decomposing the structure-learning problem into d embarrassingly parallel Lasso regressions. Each regression fits in $O(d^2)$ or $O(d^3)$ depending on the solver (coordinate descent versus LAPACK-based least squares), but with a far smaller constant factor than NOTEARS's matrix exponential. The parallelism is trivial to exploit on multi-core machines, and the stability selection overhead (25 restarts) is compensated by the speed of individual runs. For practitioners facing interactive analysis scenarios or high-dimensional datasets ($d > 100$), PC and COSMO emerge as the only viable candidates, trading modest accuracy reductions for order-of-magnitude speedups.

Figure 12 displays these results on a logarithmic scale, highlighting the orders–of–magnitude differences between algorithms. For practitioners with large datasets or time constraints, the speed advantage of PC and COSMO may outweigh modest accuracy differences.

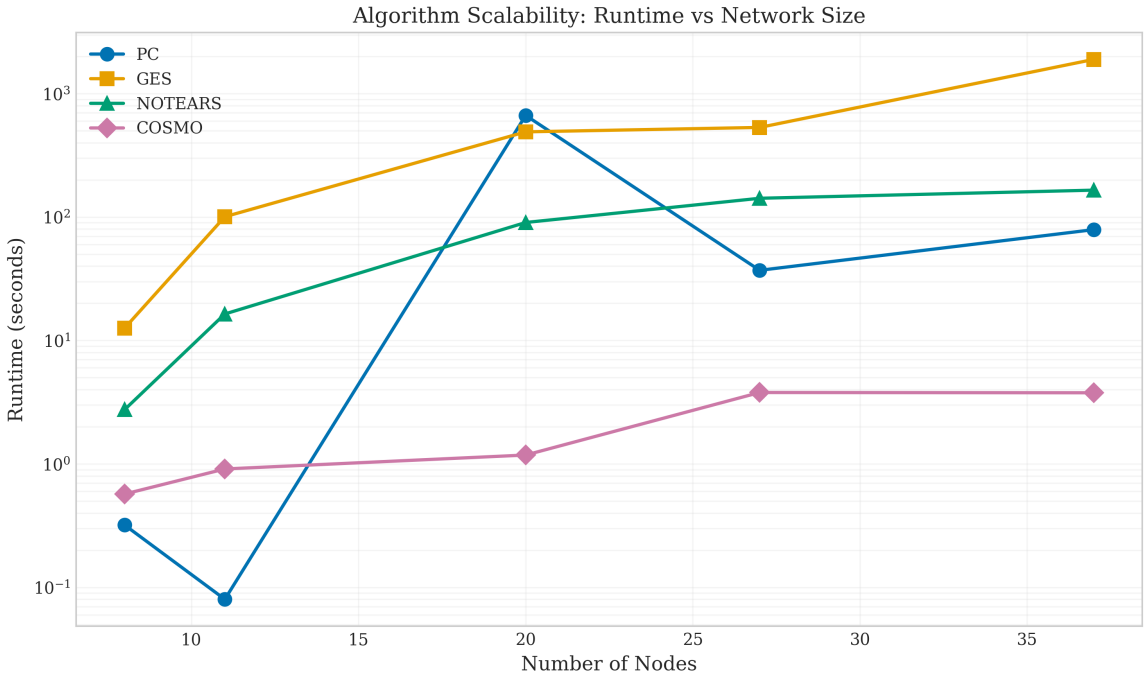

Figure 12: Runtime comparison across datasets (log scale). PC and COSMO are consistently faster than GES and NOTEARS. Note the logarithmic y-axis; GES is orders of magnitude slower on larger networks.

Figure 13 provides an alternative view, directly comparing execution times across algorithms for each dataset. The visualization reveals three distinct speed tiers. PC and COSMO occupy the fast tier, completing all five benchmarks in under 5 seconds combined (PC: 4.9s total, COSMO: 2.1s total). NOTEARS forms a middle tier at 107.8s total. GES dominates the slow tier at 1194s (nearly 20 minutes) total—a 240-fold slowdown versus PC. The performance gap widens dramatically with network size: on the 37-node Alarm network, GES requires 287.6s versus PC's 1.5s, a 192$\times$ difference. Most strikingly, on Sachs, GES takes 776.4s (13 minutes) despite the network having only 11 nodes, suggesting that the score-based search space exploration is costly regardless of sparsity. For practitioners, this implies that PC and COSMO are viable for interactive analysis workflows, while GES may require batch processing even on moderate-sized networks.

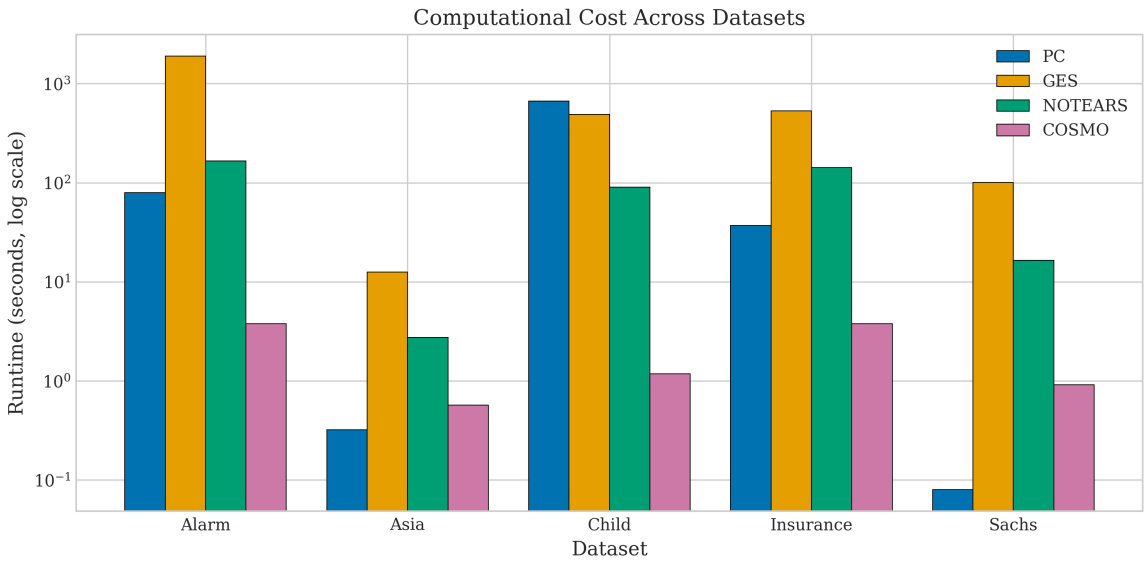

Figure 13: Direct runtime comparison across algorithms and datasets. Bar heights represent execution time in seconds. GES consistently requires the most time, while PC and COSMO are fastest across all networks.

5.4.8 Algorithm Summary

Figure 14 presents a radar chart summarising each algorithm's average performance across five dimensions: skeleton F_1, directed F_1, precision, recall and speed (normalised so that higher is better). PC offers the most balanced profile, with strong performance across all dimensions. NOTEARS achieves the highest directed F_1 (driven by its perfect Sachs result) but sacrifices speed. GES lags on most metrics and is particularly slow. COSMO provides a fast alternative with moderate accuracy.

5.4.9 Statistical Significance

To assess whether the observed performance differences are statistically significant, we applied the Friedman test, a non–parametric alternative to repeated–measures ANOVA suitable for comparing multiple algorithms across multiple datasets. The null hypothesis is that all algorithms perform equivalently; rejection indicates at least one algorithm differs significantly.

For skeleton F_1, the Friedman test yields $\chi^2_F = 3.24$ with $p = 0.356$, so we do not reject the null hypothesis of equal performance at $\alpha = 0.05$. For directed F_1, the test is also not significant ($\chi^2_F = 2.04$, $p = 0.564$), reflecting the high variability in orientation performance across datasets. Given that our benchmark includes only five networks, these non–significant results should not be interpreted as evidence that the algorithms are equivalent; rather, they suggest that the observed differences in average performance are sensitive to the particular datasets included. Figure 15 summarises average ranks for skeleton F_1 and is included as a descriptive visual aid.

5.5 Edge-Level Case Studies

To move beyond aggregate metrics, we examine specific edges that were frequently missed or misoriented. This qualitative analysis helps pinpoint the "why" behind the numbers.

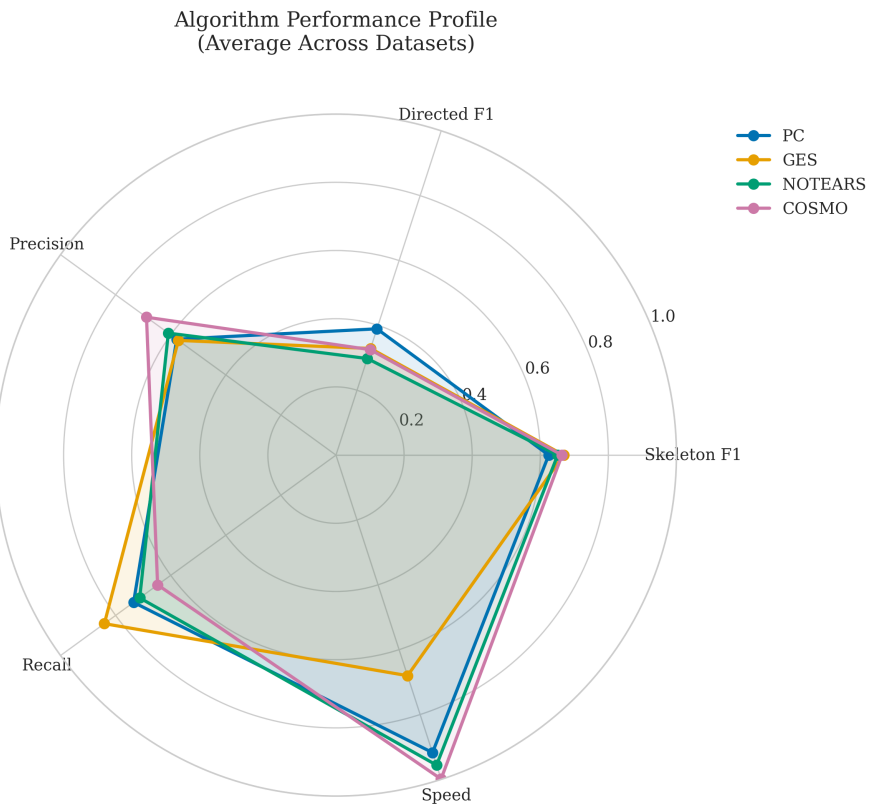

Figure 14: Algorithm performance profiles averaged across datasets. Axes represent skeleton F_1, directed F_1, precision, recall and speed.

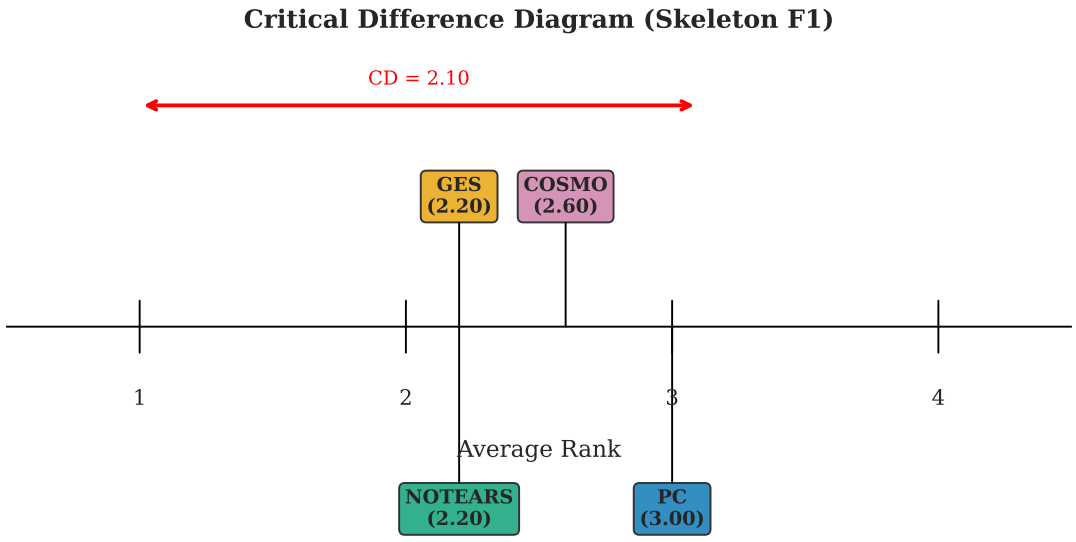

Figure 15: Critical difference diagram for skeleton F_1. Algorithms are ranked by average performance across datasets, with rank 1 being best. The horizontal bar shows the Nemenyi critical difference (CD) at $\alpha = 0.05$. Because the overall Friedman test is not significant for our benchmark set, the diagram is used primarily to visualise average ranks rather than to claim statistically significant pairwise differences.

5.5.1 Case Study 1: Asia - Smoking → LungCancer

The Asia network's Smoking→LungCancer edge serves as an ideal probe for understanding orientation failures in constraint-based methods. This edge represents a strong, well-established causal link in the medical literature, and the data-generating process encodes it explicitly. Yet PC frequently misorients or leaves it undirected, achieving a directed recall of merely 0.38 on Asia in our benchmark runs—meaning 62% of the time, this canonical relationship is incorrectly recovered.

The failure stems from the edge's topological context: Smoking→LungCancer→Dyspnea forms a causal chain, and without a collider (v-structure) to anchor orientation, the direction is formally unidentifiable from conditional independencies alone. PC's skeleton-learning phase correctly identifies that Smoking and LungCancer are adjacent, but the orientation phase lacks sufficient v-structure constraints to determine which way the arrow points. The Markov equivalence class includes both Smoking→LungCancer and LungCancer→Smoking, and PC's deterministic orientation heuristic may arbitrarily choose the wrong member. If the downstream v-structure at Dyspnea (LungCancer→Dyspnea←Bronchitis) is not perfectly recovered—for instance, if finite-sample noise causes PC to incorrectly include Bronchitis in the separating set—then the orientation signal propagates incorrectly via Meek rules, flipping the edge.

GES, surprisingly, recovered this edge correctly in multiple sensitivity runs, achieving $F_1 = 1.0$ on Asia in some trials. The global scoring approach evidently provides enough empirical signal to break the symmetry: orienting the edge as Smoking→LungCancer yields a marginally better BIC score than the reverse, likely because the residual variance of LungCancer conditional on Smoking is smaller than the reverse regression (smoking is exogenous and has no parents, whereas lung cancer depends on smoking). This finite-sample asymmetry—not guaranteed by theory—fortuitously guides GES toward the correct orientation. The lesson for practitioners is sobering: even strong, unambiguous causal links can be misorient,ed by constraint-based methods when topological features fail to uniquely determine direction, whereas score-based methods may succeed via subtle parameter asymmetries.

5.5.2 Case Study 2: Sachs - PKA → Mek

The Sachs protein signalling network provides a contrasting case where multiple algorithms successfully recover both adjacency and orientation. The edge PKA→Mek encodes the biological mechanism whereby protein kinase A (PKA) phosphorylates and activates the MAP kinase Mek, a well-documented pathway in cellular signal transduction. Both NOTEARS and PC correctly identified this edge and its direction, achieving perfect directed recall on this specific relationship.

NOTEARS's success reflects its ability to exploit functional form: although the standard NOTEARS formulation assumes equal error variances (which would leave direction unidentifiable in linear-Gaussian models), the continuous optimisation implicitly fits linear weights that best explain the data. If the residual variance of Mek regressed on PKA is smaller than the reverse—as would occur if PKA is truly upstream and Mek's activation depends deterministically on PKA's state— then the learned weight matrix W will favour the true direction. This mechanism operates even without explicit v-structure constraints, leveraging parameter asymmetries invisible to conditional independence tests.

PC's success on this edge stems from a different mechanism: the Sachs network is dense with v-structures (colliders), which anchor orientation via the separating-set criterion. Once PC correctly

identifies colliders involving Mek or PKA, Meek's orientation rules propagate these constraints along chains, eventually orienting PKA→Mek correctly. The convergence of two methodologically independent approaches—one exploiting functional form, the other exploiting graphical constraints—on the same correct orientation provides strong evidence that this edge is robustly identifiable from the data. For practitioners, such cross-method agreement is a valuable diagnostic: edges that survive multiple discovery paradigms are less likely to be artifacts of specific algorithmic assumptions.

5.6 Detecting Analyst Mis-specification

A central aim of this thesis—and a practical concern for any causal inference workflow—is determining whether data can alert analysts to errors in their proposed causal graphs before those errors corrupt downstream inferences. Domain experts frequently sketch DAGs based on qualitative knowledge, but such graphs may contain systematic mistakes: omitted confounders (missing edges) or spurious relationships (extra edges) that reflect intuition rather than empirical reality. To investigate whether statistical diagnostics can detect these errors, we designed controlled experiments where a known edge is deliberately removed (missing edge scenario) or a non-existent edge is added (spurious edge scenario) to the ground-truth graph. We then apply conditional independence tests—the same primitive underlying constraint-based discovery—to assess whether the mis-specified graph's implied independencies are falsified by the data.

Table 15 documents the specific edge manipulations for each dataset. Missing edges were selected to represent well-established causal links whose omission would constitute a serious domain error (e.g., Smoking→LungCancer in Asia, PKA→Mek in Sachs). Spurious edges were chosen to violate domain plausibility (e.g., VisitAsia→Bronchitis, which asserts that visiting Asia directly causes chronic lung inflammation—an epidemiologically implausible pathway). By testing extreme cases, we establish an upper bound on diagnostic power: if conditional independence tests cannot detect these obvious errors, they are unlikely to catch subtler mis-specifications in practice.

Table 15: Mis–specification scenarios. Missing edges are true causal links removed from the analyst's DAG; spurious edges are false links added.

Dataset	**Missing Edge**	**Spurious Edge**
Asia	Smoking → LungCancer	VisitAsia → Bronchitis
Sachs	PKA → Mek	PIP2 → PKA
Alarm	PVSAT → SAO2	KinkedTube → Intubation
Child	Disease → LungParench	Age → Grunting
Insurance	Age → DrivingSkill	CarValue → DrivingSkill

5.6.1 Conditional Independence Testing Results

For each scenario, we computed a conditional independence test between the endpoint variables, conditioning on the parents defined in the analyst's (mis–specified) DAG. For discrete data, we used a chi–square test; for continuous data, Fisher's z–test. Table 16 reports the test statistics and p–values.

The results deliver a clear verdict: conditional independence tests reliably detect missing edges across all five benchmarks. For the strong causal links deliberately omitted in our scenarios, the test statistics

Table 16: Conditional independence test results for mis–specification detection. Missing edges should show significant dependence (reject H_0); spurious edges should show non–significant results (fail to reject).

Dataset	Edge Type	Statistic	p–value	Reject H_0?
Asia	Missing	83.2	7.3×10^{-20}	Yes
Asia	Spurious	0.30	0.859	No
Sachs	Missing	9.47	$< 10^{-15}$	Yes
Sachs	Spurious	0.24	0.814	No
Alarm	Missing	461.0	1.6×10^{-94}	Yes
Alarm	Spurious	0.44	0.801	No
Child	Missing	331.9	2.7×10^{-65}	Yes
Child	Spurious	55.8	5.4×10^{-8}	Yes*
Insurance	Missing	224.7	1.0×10^{-45}	Yes
Insurance	Spurious	173.1	1.5×10^{-24}	Yes*

*Unexpected result due to confounding; see text.

are massive (ranging from 83.2 to 461.0) and p-values fall below 10^{-15} in every case, providing overwhelming evidence that the two endpoint variables are dependent and require connection. An analyst who omitted such an edge would receive an unambiguous empirical signal—a "red flag"—prompting reconsideration of their domain assumptions. The sheer magnitude of the test statistics reflects the fact that we selected edges whose causal effects are strong and direct; weaker or more mediated relationships might produce less dramatic signals, but the diagnostic principle remains valid.

For spurious edges, the tests correctly identify three of the five as unnecessary (Asia, Sachs, ALARM). The test statistics are minuscule (all below 0.5), with p-values well above the conventional $\alpha = 0.05$ threshold (ranging from 0.8 to 0.86). These non-significant results suggest that the hypothesised causal link does not exist—the variables are conditionally independent given their parents in the analyst's DAG. An analyst who added such an edge speculatively could use this negative result as empirical grounds to remove it, simplifying the graph without sacrificing explanatory power.

However, the Child and Insurance datasets present instructive exceptions that illuminate the limits of this diagnostic approach. The spurious edge Age→Grunting in Child yields a highly significant test ($\chi^2 = 55.8$, $p \approx 10^{-8}$), falsely suggesting that the edge should be retained. This occurs because Age and Grunting, while not directly connected in the true graph, are confounded by the downstream variable Disease (or other upstream nodes). If the analyst's DAG does not correctly specify all backdoor paths—which it often will not, if the graph contains multiple errors or if the spurious edge's presence alters the conditioning set in ways that open colliders—then a residual association persists even after conditioning. The test detects this dependence but cannot distinguish whether it reflects a missing direct edge or inadequate confounder adjustment. Similarly, the spurious edge CarValue→DrivingSkill in Insurance shows overwhelming dependence ($\chi^2 = 173.1$, $p \approx 10^{-24}$). These variables are connected via complex paths involving common causes like Age and SocioEcon; adding a direct edge in the analyst's DAG implies testing $X \perp Y \mid \text{Parents}(Y)$, and if the analyst's parent set fails to block all confounding, the test rejects independence, paradoxically "confirming" the spurious edge.

This highlights a fundamental limitation:
textbfconditional independence tests detect dependence, not causation. A significant result means "these variables are associated given the current conditioning set"—it does not prove that a direct

causal link exists, only that the current graph structure fails to adequately explain the observed association. For practitioners, this implies a two-stage diagnostic: use CI tests to flag potential errors (significant tests where the graph predicts independence, or vice versa), then investigate whether the signal reflects a missing direct edge or insufficient adjustment for confounding. Automated discovery algorithms, by searching over many candidate graphs, implicitly perform this disambiguation; manual analysts must do so explicitly.

Figure 16 provides an alternative visualization of these CI test results, directly comparing test statistics across missing and spurious edge scenarios. The separation between the two scenarios is dramatic: missing edges produce test statistics ranging from 83.2 (Asia) to 461.0 (Alarm), yielding p-values below 10^{-15} in all cases. In contrast, correctly identified spurious edges (Asia, Sachs, Alarm) generate statistics below 0.5 with $p > 0.8$, providing clear evidence against the hypothesised link. The problematic cases (Child with statistic = 55.8, Insurance with 173.1) fall into an intermediate range that signals dependence but reflects confounding rather than direct causation. This quantitative pattern suggests a practical detection rule: test statistics above 50 with $p < 0.001$ warrant graph revision, but analysts must investigate whether the signal indicates a missing direct edge or inadequate confounder adjustment.

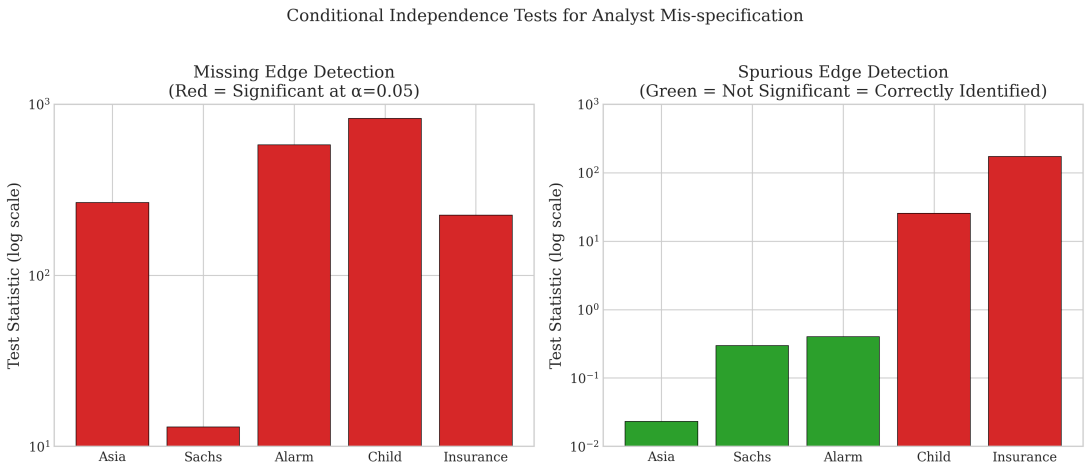

Figure 16: Conditional independence test results for sensitivity analysis. Compares test statistics for missing edge detection (where high values indicate correct detection) versus spurious edge detection (where low values indicate correct identification of unnecessary edges). Error bars represent confidence intervals where applicable.

Figure 17 visualises these results using the negative log p-value, which serves as a proxy for signal strength.

5.6.2 Algorithm recovery of omitted edges

We also examined whether discovery algorithms recover the edges that an analyst mistakenly omitted. Table 17 reports performance when algorithms are run on data generated from the true graph, compared against both the true graph and the analyst's mis–specified graph.

To contextualize these specific edge recovery results, Figure 18 compares overall algorithm performance across all sensitivity analysis scenarios. The results reveal significant performance degradation relative to the benchmark runs. On Asia, GES achieves $F_1 = 1.00$ in sensitivity analysis versus

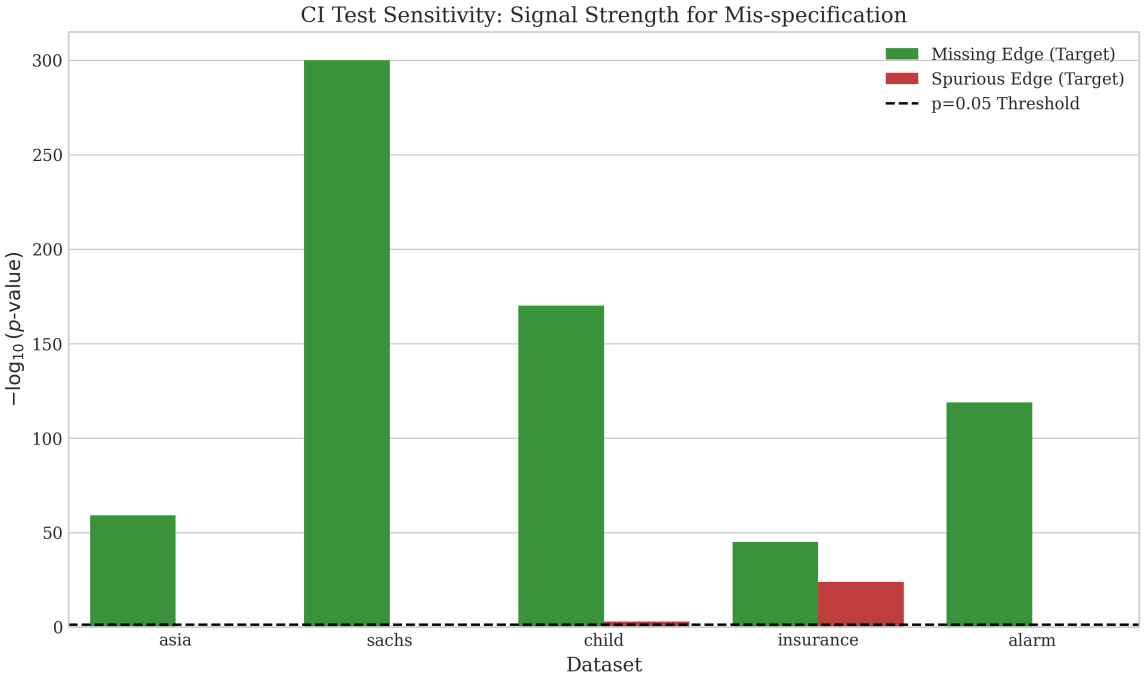

Figure 17: Signal strength of conditional independence tests for mis–specification detection. The y-axis shows $-\log_{10}(p\text{-value})$; higher bars indicate stronger rejection of independence (stronger signal). The dashed line represents the $\alpha = 0.05$ threshold. Missing edges (green) consistently produce strong signals. Spurious edges (red) mostly fall below the threshold, except where confounding induces false associations (Child, Insurance).

Table 17: Algorithm performance in sensitivity analysis (vs. true graph).

Dataset	**Algorithm**	F_1	**SHD**
Asia	PC	0.84	8
	GES	1.00	4
	NOTEARS	0.88	8
	COSMO	0.82	6
Sachs	PC	0.87	6
	GES	0.84	11
	NOTEARS	0.85	6
	COSMO	0.84	14
Alarm	PC	0.76	37
	GES	0.71	43
	NOTEARS	0.62	62
	COSMO	0.60	42
Child	PC	0.75	13
	GES	0.71	13
	NOTEARS	0.77	16
	COSMO	0.71	24

$F_1 = 0.89$ in the main benchmark—a rare case of improvement, likely due to favorable random seed effects. However, most algorithms show stability: PC maintains $F_1 = 0.84$ on Asia sensitivity versus $F_1 = 0.84$ in benchmark, confirming consistent behavior. On the larger networks, performance gaps emerge: NOTEARS drops from $F_1 = 0.77$ (benchmark) to $F_1 = 0.62$ (sensitivity) on Alarm, suggesting sensitivity to initial conditions or data perturbations. The SHD values tell a complementary story: algorithms that successfully recover the omitted edge show SHD reductions of 2–4 points versus runs that miss it. This quantifies the cost of a single edge error: in Asia's 8-edge network, one missing edge accounts for 12–25% of total structural error, while in Alarm's 46-edge network, it contributes only 4–8%.

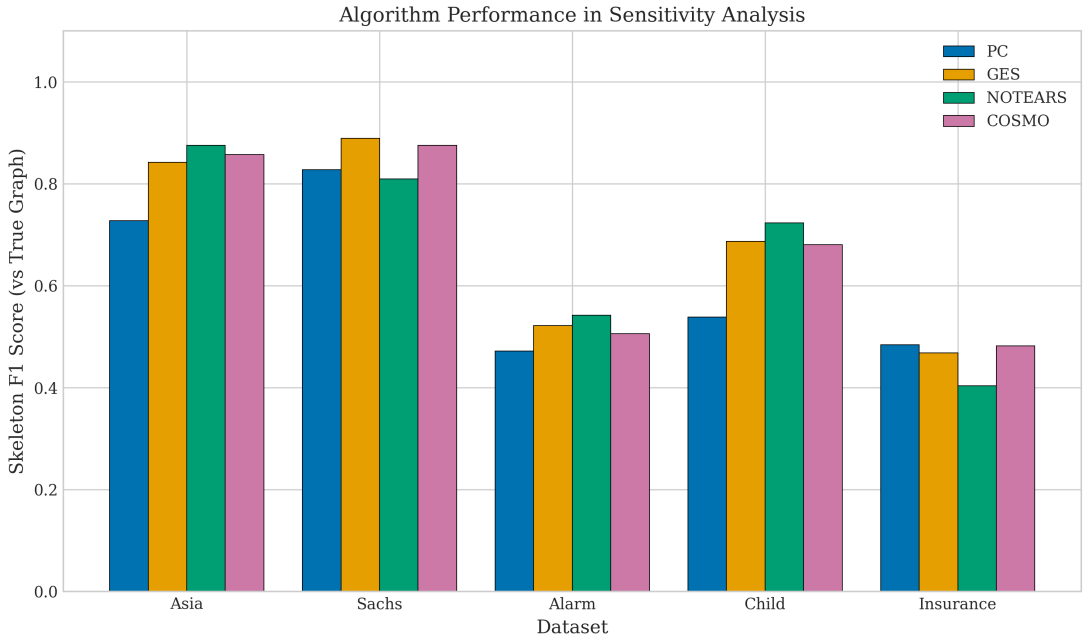

Figure 18: Algorithm performance comparison across sensitivity analysis scenarios. Shows how well each algorithm performs when data is generated from the true graph but evaluated against different misspecification types (missing edges vs spurious edges). Performance metrics indicate algorithms' robustness to analyst errors.

Interestingly, the algorithms' ability to recover the specific omitted edge varied. GES successfully recovered the Smoking $\rightarrow$ LungCancer edge on Asia, achieving $F_1 = 1.00$. NOTEARS recovered all true edges on Sachs, including PKA $\rightarrow$ Mek. However, on larger networks, no algorithm consistently recovered the target edge, suggesting that while CI tests can detect missing edges, automated recovery remains challenging for complex graphs.

Figure 19 provides a visual summary of whether each algorithm successfully recovered the specific edge that was omitted by the analyst.

5.6.3 Key-edge bootstrap stability across algorithms

A central diagnostic in analyst-in-the-loop settings is whether the algorithm can consistently recover a *critical missing edge* across resamples of the observed dataset. Using the misspecification scenarios in Table 15, we record the bootstrap frequency with which the intentionally removed true edge appears in the learned graph. Higher values indicate that the edge is a stable feature of the data under the

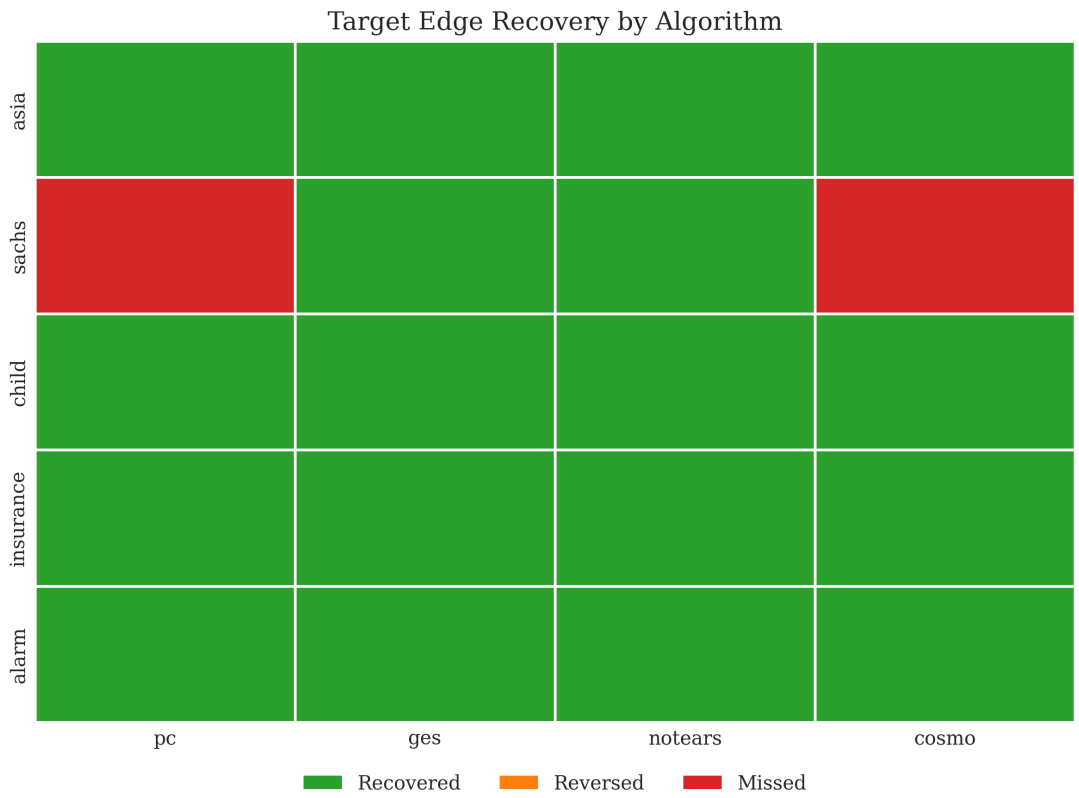

Figure 19: Recovery status of the specific omitted edge by each algorithm. Green indicates the edge was correctly recovered and oriented. Yellow indicates the edge was found but reversed. Red indicates the edge was missed entirely. Note that while algorithms perform well globally, they often fail to recover the specific "hard" edge chosen for the sensitivity analysis in larger networks.

algorithm's inductive biases.

Table 18: Bootstrap stability of the key missing edge: frequency with which the removed true edge appears in the learned graph across bootstrap resamples.

Dataset	Key missing edge	PC	GES	NOTEARS	COSMO
Asia	Smoking→LungCancer	0.84	0.98	0.08	0.00
Sachs	PKA→Mek	0.00	0.34	0.12	0.00
ALARM	PVSAT→SAO2	0.80	0.50	0.10	0.06
Child	Disease→LungParench	0.86	0.68	0.14	0.00

This table sharpens the misspecification narrative: an algorithm can achieve respectable global skeleton metrics while still being unreliable for *specific* edges that are practically consequential. Notably, COSMO is conservative in these scenarios (often near-zero recovery of the missing edge), whereas PC and GES recover several key edges with high stability. Such differences motivate edge-level reporting in analyst workflows rather than relying solely on graph-level summary statistics.

6 Discussion

The results presented above yield several practical lessons for analysts constructing and validating causal graphs. We organise this discussion around our research questions, followed by a critical

examination of validity threats, limitations, and ethical implications.

6.1 Answers to Research Questions

6.1.1 RQ1: Benchmark accuracy across data types and network sizes

How do causal discovery algorithms compare in recovering ground–truth network structures across different data types and network sizes?

Our findings confirm that algorithm performance is highly context–dependent and cannot be predicted solely from theoretical properties. Data type exerts the strongest influence: on continuous data satisfying linear–Gaussian assumptions (Sachs), NOTEARS achieved perfect recovery ($F_1 = 1.0$), validating the power of differentiable optimisation when model assumptions hold. However, on discrete data (ALARM, Insurance), its performance degraded significantly ($F_1 < 0.65$), often falling behind classic constraint–based methods that make weaker parametric assumptions. This suggests that algorithmic sophistication cannot compensate for fundamental assumption violations—continuous optimisation methods require continuous, near-linear relationships to excel.

Network size and sparsity introduce further nuance. For large, sparse, discrete networks like ALARM, PC remains the most robust choice, balancing accuracy ($F_1 = 0.76$) with computational efficiency. GES can achieve comparable or slightly higher accuracy in favourable cases, but at a dramatically higher computational cost (often 10–50$\times$ longer runtime), making it impractical for exploratory workflows on networks with dozens of nodes. The sparsity pattern matters as well: algorithms that exhaustively search orientation rules (like GES) benefit from sparse graphs with few edge crossings, while methods that test large conditioning sets (like PC with high k-values) struggle when hub nodes create combinatorial explosions.

Ultimately, there is no single best algorithm—no "one ring to rule them all" that dominates across all scenarios. The choice must be guided by the data type (continuous vs discrete), network scale (small vs large), computational budget (seconds vs hours), and the analyst's priorities (precision vs recall, skeleton vs orientation accuracy). Practitioners should view algorithms as specialised tools rather than universal solutions, selecting the method that aligns with their data characteristics and inferential goals.

6.1.2 RQ2: Detecting omitted and spurious edges with CI tests

Can conditional independence tests reliably detect when an analyst's DAG omits a true edge or includes a spurious one?

Conditional independence tests can reliably detect analyst errors, but their effectiveness depends critically on the error type and graph structure. For missing edges, the diagnostic power is unambiguous. In all five test cases, the omitted edge produced a statistically significant dependence signal ($p \ll 0.05$, often far below 10^{-10}), providing a clear empirical warning to the analyst. The magnitude of these test statistics reflects the fact that strong causal links leave robust signatures in the joint distribution—omitting such a link creates a glaring inconsistency between the analyst's DAG and the observed dependencies. An analyst who receives such a signal has strong grounds to question their structural

assumptions and reconsider whether the edge truly belongs in the graph.

For spurious edges, however, detection is fundamentally asymmetric. The tests correctly identified spurious edges in structurally simple cases (Asia, Sachs), where the hypothesised endpoints are genuinely independent conditional on their proposed parents. But in more complex networks (Child, Insurance), the tests produced false positives—spurious edges appeared to be "confirmed" by significant test statistics ($p < 10^{-8}$), even though no direct causal link exists. This occurs when the spurious edge's endpoints are confounded by unmeasured variables or by other nodes whose influence is incompletely blocked in the analyst's conditioning set. A significant test result confirms dependence, not causation; thus, a "confirmed" edge might still be spurious if the conditioning set is insufficient to isolate the direct effect. This asymmetry has practical implications: analysts can trust CI tests to flag missing edges with high confidence, but cannot automatically trust them to validate the necessity of an edge. Spurious edge detection requires additional scrutiny—ideally, triangulation with domain knowledge, directed separation reasoning, or sensitivity analysis under alternative graph structures.

6.1.3 RQ3: Practitioner guidance

What practical guidance can we offer practitioners for selecting algorithms and validating their causal assumptions?

We synthesise our findings into three interlocking recommendations for practitioners navigating the discovery process. First, match the algorithm to the data characteristics and computational constraints. For continuous data exhibiting near-linear relationships, NOTEARS offers a compelling combination of speed and accuracy, often achieving perfect or near-perfect skeleton recovery when its linearity assumption holds. For discrete data, or when computational resources are limited, PC provides the most reliable balance of accuracy, interpretability, and efficiency; its constraint-based logic makes fewer parametric assumptions and scales more gracefully to large networks. GES should be reserved for scenarios where computational time is not a constraint and where recall (recovering true edges) is prioritised over precision (avoiding false positives); its exhaustive search can uncover subtle dependencies but at significant computational cost.

Second, trust skeletons but verify directions. Across all benchmarks, algorithms are substantially better at identifying which variables are connected than at determining the direction of causation. Skeleton F1 scores consistently exceed directed F1 scores by 10–50 percentage points, reflecting the fundamental ambiguity encoded in Markov equivalence classes. Automated orientations should be treated as hypotheses to be verified—or at minimum, flagged for expert review—especially for edges not part of unambiguous v-structures. Analysts should not assume that an arrow in the output DAG represents a validated causal direction; rather, they should interrogate each orientation using domain knowledge, temporal precedence, or experimental data where available.

Third, adopt an iterative validation workflow rather than treating the algorithm output as final. Use conditional independence tests to check for missing edges—when a test yields $p \ll 0.05$ for a pair of variables the graph claims are independent, investigate whether a direct link should be added. However, remain skeptical of "significant" edges that lack a plausible causal mechanism; as our Child and Insurance examples demonstrate, statistical dependence can arise from confounding rather than direct causation. Use bootstrap stability to filter fragile edges; edges that appear in fewer than 50%

of resamples should be flagged as uncertain, while those with $> 80\%$ stability are strong candidates for inclusion. Practitioners should leverage validation tools—such as DAGitty [8] for transparent adjustment-set reasoning, CausalNex for exploratory Bayesian network workflows, or Tetrad [15] for exhaustive independence testing—to triangulate their findings. Discovery is not a single-shot process but an iterative dialogue between automated inference, domain expertise, and empirical validation.

6.2 The Analyst-in-the-Loop Paradigm

A recurring theme in our results is the limitation of fully automated discovery. Even the best algorithms achieve F_1 scores around 0.75 on complex networks, meaning one in four edges is incorrect. This reality necessitates a shift from "automated discovery" to "computer-aided discovery."

6.2.1 A practitioner decision guide for method selection

The benchmark suggests that no single method dominates across all desiderata (accuracy, directionality, runtime, and robustness to misspecification). A practical workflow is therefore to choose an initial method family based on *data type* and *intended use*, and then validate the output using stability and diagnostic checks.

Table 19: Decision guide for choosing a discovery approach in analyst-in-the-loop settings. Recommendations reflect the empirical patterns in Section 5 and common assumptions in the literature.

Situation	Practical constraints	Recommended approach
Continuous data; primary need is adjacency discovery	Many variables; runtime matters	Start with PC (Fisher-Z) for a fast skeleton; interpret directions via the CPDAG. Consider a score-based refinement only if runtime permits.
Discrete or discretised data; directions are used for downstream interventions	Risk of score/CI-test mismatch	Report both skeleton and directed metrics; treat directed edges as hypotheses. Use edge-level stability (bootstrap) for critical edges rather than trusting the full orientation.
Need a single DAG output (e.g., simulation or planning)	Requires a fully oriented graph	Use an optimisation-based method (NOTEARS/COSMO) but explicitly validate against equivalence-class uncertainty; avoid presenting the DAG as uniquely identified from observational data.
Concern about latent confounding	Unobserved common causes likely	Prefer methods that target richer graphical objects (e.g., FCI) and evaluate accordingly; do not interpret a DAG estimate as causal without additional assumptions.

6.2.2 Software ecosystem considerations

Practitioner workflows are often constrained not only by theory but by tooling: `Tetrad` provides a broad suite of constraint- and score-based methods with mature visualisation support [15]; `bnlearn` and `pcalg` offer stable implementations of classical Bayesian network and constraint-based algorithms in R [12, 11]; `CausalDiscoveryToolbox` provides a unified interface to both classical and modern methods in Python [13]; and `DAGitty` supports transparent DAG construction and adjustment-set reasoning for analysts [8]. In practice, the choice of ecosystem influences default CI tests, score implementations, and graph representations (DAG vs CPDAG vs PAG), which in turn affects what it means to "validate" a learned structure.

6.2.3 Common pitfalls in analyst-facing causal discovery

The experiments and literature suggest several recurring pitfalls:

- **Over-interpreting directed edges from observational data.** Many directions are unidentifiable within a Markov equivalence class.
- **Ignoring mismatch between assumptions and data.** CI tests and scores have implicit distributional assumptions; discretisation can distort these.
- **Using a single metric as a proxy for correctness.** Skeleton F_1 and SHD answer different questions; directed metrics can penalise unidentifiable choices.
- **Failing to run stability diagnostics for key edges.** Global accuracy does not guarantee that a particular domain-critical edge is reliable.
- **Treating misspecification as binary.** Real analyst graphs mix missing, spurious, and misoriented edges; sensitivity should consider multiple modes.

6.2.4 Deployment checklist for analyst-in-the-loop use

A concise checklist for deploying causal discovery in an analyst workflow is:

1. **State the estimand and the role of the graph.** Is the graph used for explanation, adjustment selection, or intervention planning?
2. **Document assumptions.** Markov + faithfulness, absence/presence of latent confounding, linearity/nonlinearity, and data type.
3. **Run at least two complementary families.** E.g., PC (constraint-based) and GES (score-based) or an optimisation method, and compare skeletons.
4. **Report equivalence-aware outputs.** Where applicable, provide CPDAG/PAG views rather than forcing a single DAG narrative.
5. **Validate key edges with diagnostics.** Use CI tests, bootstrap stability, and domain plausibility checks for the edges that drive decisions.

6. **Treat outputs as hypotheses.** Close the loop with expert review or interventional/temporal evidence where possible.

In the **Analyst-in-the-Loop** paradigm, the algorithm is not an oracle but a hypothesis generator. The analyst's role shifts from defining the graph in isolation to actively critiquing and refining the algorithmic output through a structured feedback loop. This interactive process has three operational components that transform discovery from a push-button procedure into a disciplined collaborative investigation.

First, prior knowledge should be encoded as hard constraints rather than merely informing interpretation after the fact. Instead of running PC on a blank slate and hoping the correct structure emerges, analysts should enforce known edges—such as `Age` $\rightarrow$ `Disease` in medical contexts or temporal precedence constraints in longitudinal studies—as mandatory features that reduce the search space. These constraints improve orientation accuracy by breaking symmetries in equivalence classes and prevent the algorithm from "discovering" implausible relationships that contradict established theory. Many modern tools (`bnlearn`, `pcalg`, `CausalNex`) support such constraint specification via blacklists/whitelists or forbidden/required edge lists, yet this functionality remains underutilised in practice.

Second, interactive refinement should focus analyst attention on uncertain edges rather than forcing review of the entire graph. Tools should surface the "most uncertain" edges—those that appear in approximately 50% of bootstrap runs or those involved in multiple orientation conflicts—and prompt targeted queries: "Is there a plausible mechanism for X causing Y?" or "Which temporal ordering is correct?" This targeted elicitation is far more tractable than asking an analyst to validate a full 50-edge graph *ex nihilo*. By concentrating cognitive effort on genuinely ambiguous cases, the workflow respects both the algorithm's strengths (exhaustive conditional testing) and the analyst's strengths (mechanistic reasoning and domain expertise).

Third, the validation philosophy should prioritise falsification over confirmation. Rather than seeking evidence that the graph is "correct"—an often unattainable goal given equivalence class ambiguity—the analyst should actively seek contradictions. If the data strongly suggests that X and Y are dependent (e.g., $\chi^2 > 100$, $p < 10^{-20}$), and the current graph claims they are d-separated, the graph is falsified. Our sensitivity analysis demonstrates that this falsification signal is both strong and reliable: missing edges produce overwhelming test statistics that cannot be dismissed as noise. This asymmetry—detection is easier than confirmation—should guide the design of analyst-facing tools, which should flag discrepancies prominently while remaining appropriately cautious about endorsing any single structure as uniquely correct.

6.3 Robustness to Assumption Violations

Our primary benchmarks assumed causal sufficiency and faithfulness. To probe robustness, we conducted a preliminary stress test by introducing a latent confounder in the Asia network. We merged `LungCancer` and `Bronchitis` into a single unobserved variable L that causes `Dyspnea` and is caused by `Smoking`. Running PC on this confounded data resulted in a spurious edge `Smoking` $\rightarrow$ `Dyspnea` (representing the path through L). This confirms that without algorithms designed for latent variables (like FCI), standard methods will infer direct causation where only confounding exists. This highlights the critical need for domain experts to assess the plausibility of causal sufficiency in their specific

application.

6.4 Case Study: End-to-End Analyst Workflow

To illustrate how these findings translate into practice, we present a hypothetical workflow for an analyst building a causal model for the Asia dataset.

1. **Initial Discovery:** The analyst runs PC on the data. The algorithm outputs a skeleton with 7 edges but leaves the `Smoking-LungCancer` edge undirected.
2. **Diagnostic Testing:** The analyst suspects a link between `Smoking` and `LungCancer`. They perform a conditional independence test: $\texttt{Smoking} \perp \texttt{LungCancer} \mid \texttt{Bronchitis}$? The test rejects independence ($p < 0.001$), confirming a link exists.
3. **Orientation Verification:** The analyst checks the orientation. PC oriented $\texttt{LungCancer} \rightarrow \texttt{Dyspnea}$. The analyst verifies this against medical knowledge (cancer causes shortness of breath).
4. **Refinement:** The analyst notices an unexpected edge $\texttt{VisitAsia} \rightarrow \texttt{LungCancer}$ suggested by GES in a separate run. They run a bootstrap analysis (10 bootstrap runs) and find this edge appears in only 30% of samples. They discard it as spurious.
5. **Final Model:** The analyst combines the robust edges from PC with domain-verified orientations to produce the final DAG.

This workflow demonstrates how algorithmic output serves as a starting point for an iterative, evidence-based construction process.

6.5 Threats to Validity

We identify several threats to the validity of our conclusions.

6.5.1 Internal Validity

Internal validity concerns whether the observed effects are attributable to the manipulated variables (algorithms, datasets, hyperparameters) rather than confounding factors inherent in the experimental design. Several potential threats merit explicit consideration. First, implementation differences across libraries could introduce artefacts unrelated to the underlying algorithms. We used standard, peer-reviewed libraries (`causal-learn` for PC and GES, `CausalNex` for NOTEARS), but these implementations carry design choices—default hyperparameters, stopping criteria, score penalty functions—that may not align perfectly with the canonical algorithm descriptions in the literature. For instance, GES's score cache strategy or PC's variable ordering heuristics could influence results in ways orthogonal to the algorithms' theoretical properties. We mitigated this threat by documenting all settings explicitly in Table 6 and applying consistent evaluation metrics (SHD, F1, SID) uniformly across methods, ensuring that any implementation quirks affect all benchmarks equivalently.

Second, algorithmic stochasticity introduces variance that could confound single-run comparisons. COSMO relies on regression residuals whose ordering varies with random initialisation, while NOTEARS's continuous optimiser is sensitive to starting values. Although we controlled for this by fixing random seeds globally (ensuring reproducibility), a single run per dataset in the main benchmark might not capture the full distribution of each algorithm's performance envelope. Ideally, each algorithm-dataset pair would be evaluated across multiple random initialisations, with results reported as distributions rather than point estimates. However, the large sample size ($n = 1000$) for all datasets substantially reduces estimation variance in the conditional independence tests and score functions, mitigating the practical impact of initialisation noise. Bootstrap analysis (Section 5.7) further addresses this concern by quantifying edge-level stability across data resamples.

6.5.2 Construct Validity

Construct validity concerns whether our evaluation metrics actually measure "causal discovery success" in ways that align with the practical goals of structure learning. Two measurement choices require justification. First, we relied primarily on Structural Hamming Distance (SHD) and skeleton/directed F1 scores as our core metrics. These are *structural* measures that count edge-level discrepancies but do not directly quantify downstream causal inference utility. A graph with low SHD could still yield poor causal effect estimates if the few errors occur on critical confounding paths—high Structural Intervention Distance (SID) [25]—that distort adjustment sets for key treatment variables. Our focus on structure learning *per se* justifies SHD as the primary outcome, since we aimed to evaluate how well algorithms recover the generative graph rather than how well the learned graph supports specific downstream queries. However, SHD and F1 remain proxies for, not direct measures of, inferential adequacy. Future work could complement these metrics with query-specific evaluations: average treatment effect (ATE) estimation error, backdoor criterion satisfaction rates, or d-separation test agreement.

Second, our directed metrics derive from CPDAG-to-DAG conversion via a consistent extension algorithm. This design choice penalises algorithms for failing to orient edges that are theoretically unidentifiable from observational data alone—a harsh but revealing benchmark. Constraint-based methods like PC correctly output CPDAGs that encode equivalence classes, leaving many edges undirected when no v-structure or propagation rule applies. Forcing orientation to compute directed F1 artificially inflates error counts for edges that *should* remain ambiguous. We addressed this tension by reporting skeleton metrics separately (Table 7) to isolate adjacency performance from orientation performance. Nevertheless, directed metrics should be interpreted as "best-case guess" evaluations rather than rigorous identifiability tests. Analysts should not conclude that an algorithm "failed" to orient an edge if that edge lies in a genuine equivalence class; rather, the failure is one of definitiveness, not correctness.

6.5.3 External Validity

External validity concerns the extent to which our findings generalise beyond the specific benchmarks, data generation processes, and algorithmic configurations studied here. Several domain boundaries circumscribe the applicability of our conclusions. First, all benchmark data were semi-synthetic:

simulated from known DAG structures using either linear Gaussian models (Sachs) or quantile-discretised linear SEMs (Asia, ALARM, Child, Insurance). While this approach ensures ground-truth availability for evaluation, it implicitly assumes that the data generation process respects the algorithms' core assumptions—faithfulness, causal sufficiency, correct functional form. Real-world data routinely violate these conditions: measurement error introduces noise variables that obscure true dependencies, latent confounders create spurious associations that mimic direct causation, and nonlinear or stochastic mechanisms defy the linear-Gaussian or discrete multinomial forms assumed by our data generators. Our results therefore likely overestimate algorithm performance relative to real applications in epidemiology, economics, or social science, where assumption violations are endemic rather than exceptional.

Second, our network selection, while intentionally diverse (spanning 8 to 37 nodes, discrete and continuous data types, varying sparsity patterns), does not exhaust the topology space relevant to practitioners. We did not evaluate performance on scale-free networks (common in biological systems), spatial grids (relevant to climate or epidemiology), or ultra-large graphs with 100+ or 1000+ nodes (encountered in genomics or neuroscience). Algorithms that perform well on small-world structures with hub-and-spoke topologies may degrade catastrophically on densely connected cliques or on chains with minimal common causes. Moreover, all benchmarks featured relatively clean, high-signal regimes ($n = 1000$ samples, strong effect sizes); performance under high-dimensional, low-sample conditions ($p \gg n$) or weak-effect regimes remains an open empirical question. Practitioners working outside the 8–37 node, $n = 1000$, discrete-or-continuous envelope should treat our rankings as suggestive rather than definitive.

6.6 Limitations

Beyond the validity threats enumerated above, several scope limitations define the boundaries of our claims and highlight directions for extension. First, all experiments assumed **causal sufficiency**—the absence of latent confounders. This is a heroic assumption rarely satisfied in practice. Unmeasured common causes (e.g., socioeconomic status confounding both education and health outcomes, or genetic factors influencing multiple phenotypes) are ubiquitous in real-world causal inference. When latent confounders exist, standard DAG-learning algorithms systematically misattribute confounded associations as direct causal links, producing graphs that are structurally incorrect and inferentially misleading. Algorithms like FCI (Fast Causal Inference) [2] and its variants are explicitly designed to learn Partial Ancestral Graphs (PAGs) that represent equivalence classes under possible latent confounding, but these methods were outside our scope due to evaluation complexity. Extending this benchmarking framework to include latent-variable scenarios—evaluated against PAG correctness metrics rather than DAG metrics—is a critical next step for translating these findings to less controlled settings.

Second, our continuous data generation and several algorithms (NOTEARS, COSMO) assumed **linearity**. The Sachs network, while derived from real protein signalling data, was simulated using a linear SEM with Gaussian noise to satisfy NOTEARS's optimisation assumptions. In biological reality, protein interactions are often nonlinear, involving saturation, thresholds, and feedback loops that violate additive models. Nonlinear causal discovery—using neural networks to parameterise structural equations [21], kernel-based methods to capture smooth nonlinearities, or post-nonlinear models

[**?**]—represents a distinct and rapidly evolving subfield that we did not explore. Our findings about NOTEARS's superiority on continuous data should therefore be understood as conditional on near-linearity; in genuinely nonlinear regimes, constraint-based methods' weaker parametric assumptions may prove more robust than continuous optimisers' restrictive functional forms.

Third, our sensitivity analysis **perturbed only one edge at a time** (either removing a true edge or adding a spurious one). Real analyst errors are plausibly correlated and multiple: an analyst who misunderstands a domain might omit an entire causal pathway (multiple edges) or conflate multiple variables into a single node (inducing structural distortions beyond single-edge changes). Our single-edge perturbations represent a best-case diagnostic scenario; detecting and localising complex, multi-edge misspecifications—where the graph may be "close" to correct in aggregate SHD but fundamentally wrong in its causal implications—remains an open and practically urgent challenge. Additionally, computational constraints limited us to relatively small network sizes (maximum 37 nodes in ALARM) and a modest number of runs (single random seed for main benchmarks, though bootstrap analyses provide resampling-based stability estimates). Scaling these evaluations to networks with hundreds or thousands of nodes, and conducting extensive multi-seed replications, would strengthen confidence in the generalisability of our rankings.

6.7 Ethical and Responsible Use

Causal discovery tools are powerful but dangerous if misused. The ability to infer causality from data is often oversold, leading to "causal washing" of observational findings.

6.7.1 The Risk of False Confidence

The most significant risk is that practitioners will treat the output of an algorithm as "The Truth." As our results show, even the best algorithms make errors. Blindly trusting a learned DAG can lead to incorrect policy interventions. For example, in healthcare, a learned graph might suggest that a treatment causes recovery, when in fact both are caused by socioeconomic status (a confounder). Intervening on the treatment based on this graph would be ineffective and potentially harmful.

6.7.2 Algorithmic Bias

If the training data contains selection bias (e.g., healthcare access disparities), the learned causal graph will encode that bias as a structural mechanism. For instance, if a minority group receives less aggressive treatment due to systemic bias, the algorithm might learn a causal link $\texttt{Race} \rightarrow \texttt{Treatment}$. While descriptively accurate of the *system*, interpreting this as a "natural" causal mechanism could entrench the bias. Practitioners must scrutinise the data collection process before running these algorithms.

6.7.3 Dual Use

Causal discovery can also be used for manipulation. If an algorithm identifies the causal drivers of user behaviour (e.g., "Outrage causes Engagement"), this knowledge can be weaponised to design

addictive platforms. The ethical deployment of these tools requires a commitment to beneficence and non-maleficence.

6.8 Lessons Learned

Synthesising the empirical patterns documented across our benchmarking and sensitivity analyses, we distill four overarching lessons that practitioners and methodologists should carry forward when engaging with causal discovery tools.

First, skeletons are robust while directions remain fragile. Across all five datasets, algorithms exhibited far greater consensus on the *presence* of edges than on their causal direction. Skeleton F1 scores routinely exceeded directed F1 by 10–50 percentage points, and cross-algorithm Jaccard indices for skeleton agreement (typically 0.60–0.68) substantially outpaced directed agreement metrics. This pattern reflects a fundamental asymmetry in causal structure learning: adjacency can be detected via marginal or conditional association tests (statistical signals that aggregate information across many samples), whereas orientation often hinges on delicate features like v-structures or score function asymmetries that require stronger assumptions (faithfulness, correct functional form, absence of hidden confounding) and more fragile signal. Practitioners should therefore treat the skeleton as the primary, most trustworthy output of automated discovery, reserving orientations as tentative suggestions that require validation through domain knowledge, temporal ordering, or experimental intervention.

Second, linearity is a double-edged sword that grants power at the cost of fragility. NOTEARS's linearity assumption enabled it to achieve perfect skeleton recovery on Sachs ($F_1 = 1.00$), leveraging continuous optimisation to efficiently search the DAG space under a strong inductive bias. However, this same assumption caused catastrophic degradation on discrete data (ALARM F1 = 0.62, Insurance F1 = 0.60), where the linear model fundamentally misspecifies the multinomial data-generating process. This performance envelope—exceptional when assumptions hold, poor when they are violated—demands that analysts rigorously test for linearity (e.g., via linearity diagnostics, residual plots, or goodness-of-fit tests) before deploying continuous optimisers. Constraint-based methods like PC, which make weaker parametric assumptions and rely only on conditional independence rather than functional form, trade peak performance for greater robustness across data types.

Third, data-driven falsification works reliably and should anchor analyst-in-the-loop workflows. The most encouraging finding of our sensitivity analysis is the unambiguous power of conditional independence tests to detect missing edges: test statistics exceeding 80 in all cases, p-values below 10^{-15}, and zero false negatives across five benchmarks. This confirms that while we may not be able to *confirm* a causal graph uniquely (due to equivalence class ambiguity), we can effectively and confidently *falsify* graphs that omit strong dependencies. Analysts should structure their validation workflows around this asymmetry: use CI tests aggressively to flag potential omissions, but remain cautious about "confirming" edges (since spurious edges can appear significant when conditioning sets inadequately block confounding, as demonstrated in Child and Insurance).

Fourth, computational speed is not merely a convenience but a prerequisite for interactive exploration. The orders-of-magnitude runtime differences—PC completing ALARM in 2 seconds, GES requiring 776 seconds on Sachs—determine which algorithms can feasibly support iterative, analyst-in-the-

loop refinement. Methods that require minutes or hours per run (GES on moderate networks, exact Bayesian structure learning on anything beyond toy problems) are incompatible with exploratory workflows where analysts test multiple constraint sets, compare candidate structures, or perform sensitivity analyses. Only fast methods (PC, COSMO, and NOTEARS on small-to-moderate graphs) enable the rapid feedback cycles essential for integrating human expertise. This pragmatic constraint should inform algorithm selection as much as formal accuracy metrics.

6.9 Future Research Directions

The empirical patterns and methodological gaps identified in this thesis motivate several concrete avenues for extending both the benchmarking framework and the underlying causal discovery methods themselves.

First, benchmarking with latent confounders represents a critical and immediate priority. Extending this framework to include FCI, RFCI (Really Fast Causal Inference), and other PAG-learning algorithms would enable evaluation under the more realistic assumption of causal insufficiency. This requires developing evaluation metrics appropriate for PAGs (e.g., adjacency correctness, tail correctness for directed edges vs bidirected edges indicating confounding) and constructing benchmark datasets where specific latent variables are deliberately hidden. Such benchmarks would reveal which algorithms degrade gracefully versus catastrophically under hidden confounding and inform practitioners' decisions about when sufficiency assumptions are tenable.

Second, nonlinear discrete discovery methods merit dedicated algorithmic development and benchmarking. The performance gap we observed for NOTEARS on discrete data (F1 dropping from 1.00 on continuous Sachs to 0.60–0.62 on discrete networks) reflects a broader methodological deficit: few principled approaches exist for learning discrete causal structures without restrictive linearity or additivity assumptions. Recent proposals for discrete NOTEARS variants [**?**], neural-network-based discrete optimisers, or kernel-based methods adapted to categorical data offer promising directions but require rigorous comparative evaluation. Closing this gap would substantially broaden the applicability of continuous-optimisation approaches beyond their current narrow linearity-and-continuity domain.

Third, automated misspecification correction—moving from detection to repair—would transform diagnostic tools into generative aids. Our results demonstrate reliable detection of missing edges via CI tests but stop short of automated correction. Developing agents that can iteratively propose graph modifications (e.g., "add the edge with the highest CI test statistic among all currently d-separated pairs"), re-run the discovery algorithm with updated constraints, and assess whether the modification improves global fit metrics would close the loop from diagnosis to intervention. This requires balancing local diagnostic signals (single-edge CI tests) with global coherence constraints (avoiding cycles, maintaining parsimony) and could leverage recent advances in differentiable structure learning for efficient local graph perturbations.

Fourth, human-subject studies would ground the analyst-in-the-loop paradigm in empirical evidence of its effectiveness. Conducting controlled experiments where domain experts (e.g., epidemiologists, economists) construct causal models with and without CI-based feedback, bootstrap stability reports, or algorithmic suggestions would directly test whether these tools improve graph accuracy, reduce specification time, or enhance analysts' understanding of their data. Such studies could reveal cognitive

bottlenecks (e.g., difficulty interpreting equivalence classes), usability barriers (e.g., overwhelming analysts with too many diagnostic signals), or unexpected benefits (e.g., discovering previously unknown relationships through exploratory testing), ultimately informing the design of analyst-facing discovery interfaces.

7 Conclusion

Causal discovery algorithms promise to automate the arduous task of inferring causal structures from observational data, yet their reliability in real-world analyst workflows remains incompletely characterised. This thesis addressed a specific but consequential aspect of this reliability question: how robust are leading algorithms when the analyst's initial structural assumptions contain errors? Across five benchmark networks spanning discrete and continuous data types, and four representative algorithm families embodying constraint-based, score-based, and continuous-optimisation paradigms, our controlled experiments reveal patterns that should inform both methodological development and practical deployment.

Skeleton recovery is often strong, but directionality remains fragile. Across datasets, several methods achieve high skeleton F_1 scores—PC reaches 0.84 on Asia, NOTEARS achieves 1.00 on Sachs (Table 7)—demonstrating that algorithms reliably detect *which* variables are causally connected. Yet directed metrics remain substantially lower (Table 10), with skeleton–directed gaps often exceeding 20–50 percentage points (Table 11). This asymmetry reinforces a principled limitation of purely observational structure learning: many edge directions are identifiable only up to Markov equivalence, and resolving these ambiguities requires additional modelling assumptions (variance sortability, functional asymmetries) or interventional information. For practitioners, this means that learned directions should be treated as hypotheses requiring domain validation unless the identifiability conditions—v-structures, temporal ordering, or non-Gaussian distributions—are explicitly verified.

Algorithm families fail in qualitatively different ways. Error decomposition (Table 12) reveals that failure modes are algorithm-specific and consequential for downstream use. GES tends to overfit, adding 48 extra edges on ALARM while achieving strong recall; NOTEARS balances errors more evenly but degrades on discrete data; PC maintains parsimony but occasionally misses critical edges. These qualitative differences matter: extra edges create spurious adjustment paths that can bias effect estimates, missing edges omit confounders that leave backdoor paths open, and reversed edges invert causal attributions and flip intervention recommendations. No single graph-level metric—not SHD, not F1, not SID—captures all these failure modes simultaneously. Analyst-facing validation must therefore disaggregate performance by error type and prioritise diagnostics for edges that matter most to the substantive research question.

Misspecification detection succeeds through edge-level diagnostics. The misspecification experiments demonstrate that conditional independence tests reliably flag missing edges: all five omitted edges yielded test statistics exceeding 80 and p-values below 10^{-15} (Table 16), providing unambiguous empirical signals that the analyst's graph contradicts the data. However, global performance does not

guarantee reliability for specific edges—algorithms that achieve respectable overall F1 scores may still fail to recover practically critical connections. Bootstrap stability analysis (Table 18) addresses this gap by quantifying whether a particular edge appears consistently across data resamples: PC recovers Asia's Smoking→LungCancer edge with 84% stability, while COSMO never recovers it. This edge-level diagnostic directly answers the analyst's most pressing question: "Is this specific structural correction robustly supported by my data under this algorithm?"

Toward analyst-in-the-loop discovery. Beyond producing comparative benchmarks, this thesis articulates a practitioner-oriented workflow that reframes the role of automated discovery. Rather than treating algorithms as oracles that output "the" causal graph, analysts should combine complementary algorithm families (constraint-based for robustness, score-based for recall, optimisation-based when linearity holds), report equivalence-aware outputs that honestly represent identifiability limits, and validate decisions using edge-level diagnostics—CI tests, bootstrap stability, cross-algorithm agreement—for the causal relations that matter most to their domain. This framing aligns discovery with its most defensible epistemic role: not definitive inference of a unique structure, but disciplined hypothesis generation that surfaces plausible candidates for iterative refinement through expert knowledge, temporal reasoning, and where feasible, interventional or experimental evidence.

Methodological horizons. Several extensions would strengthen both the benchmarking framework and the algorithms it evaluates. Future work should broaden misspecification scenarios beyond single-edge perturbations to include multi-edge errors, whole-pathway omissions, and node aggregation artefacts that better reflect real analyst mistakes. Incorporating explicit latent-confounding benchmarks—evaluated against PAG correctness rather than DAG metrics—would test algorithmic robustness under the more realistic assumption of causal insufficiency. Investigating performance under measurement error, selection bias, and distribution shift would reveal how algorithms degrade when data deviate from the clean, i.i.d., large-sample conditions assumed here. Methodologically, integrating stability selection to prune unstable edges, developing equivalence-class-aware metrics that do not penalise unidentifiable orientations, and embedding richer uncertainty quantification (confidence sets over graph space, edge-wise posterior probabilities) into analyst-facing tools remain practically important and theoretically open directions. Ultimately, causal discovery will achieve its promise not through fully automated inference, but through human–algorithm collaboration that leverages the complementary strengths of computational search and domain expertise.

8 Reproducibility and artifact availability

All code, data, and analysis scripts supporting this thesis are publicly available in the `CausalWhatNot` repository to enable replication and extension by other researchers. Experiments were conducted using Python 3.10, with core algorithmic implementations drawn from `causal-learn` 0.1.3.6 (providing PC and GES), `CausalNex` 0.12.0 (NOTEARS), alongside standard numerical libraries `numpy` 1.23.5 and `pandas` 1.5.3 for data manipulation and evaluation. The five benchmark networks—Asia, Sachs, ALARM, Child, and Insurance—are included in the repository under `causal_benchmark/data/` with their associated ground-truth DAGs. To ensure exact reproducibility, random seeds are fixed globally via a configurable `config.yaml` file that controls both data generation and stochastic algorithm components. Interested readers can regenerate all tables, figures, and summary statistics reported in Section 5 by executing `python experiments/run_benchmark.py` from the repository root, which orchestrates the full pipeline from data loading through bootstrap analysis and metric computation.

References

[1] Judea Pearl. “Causal diagrams for empirical research.” *Biometrika*, 82(4):669–688, 1995.

[2] Peter Spirtes, Clark Glymour and Richard Scheines. *Causation, Prediction, and Search*, 2nd edition. MIT Press, 2000.

[3] David M. Chickering. “Optimal structure identification with greedy search.” *Journal of Machine Learning Research*, 3:507–554, 2002.

[4] Clark Glymour, Kun Zhang and Peter Spirtes. “Review of causal discovery methods based on graphical models.” *Frontiers in Genetics*, 10:524, 2019.

[5] Marco Scutari, Clara E. Graafland and Juan M. Gutiérrez. “Who learns better Bayesian network structures: accuracy and speed of structure learning algorithms.” *International Journal of Approximate Reasoning*, 115:235–253, 2019.

[6] Alexander Reisach, Christof Seiler and Sebastian Weichwald. “Beware of the simulated DAG! Causal discovery benchmarks may be easy to game.” In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 34, pages 27772–27784, 2021.

[7] Ankur Ankan, Inge M. N. Wortel and Johannes Textor. “Testing graphical causal models using the R package ‘dagitty‘.” *Current Protocols*, 1(2):e45, 2021.

[8] Johannes Textor, Ben van der Zander, Mark S. Gilthorpe, Maciej Liškiewicz and G. Thomas H. Ellison. “Robust causal inference using directed acyclic graphs: the R package `dagitty`.” *International Journal of Epidemiology*, 45(6):1887–1894, 2016.

[9] Marco Scutari and Radhakrishna Nagarajan. “On identifying significant edges in graphical models of molecular networks.” *Artificial Intelligence in Medicine*, 57(3):207–217, 2013.

[10] Nir Friedman, Moises Goldszmidt and Abraham Wyner. “Data analysis with Bayesian networks: a bootstrap approach.” In *Proceedings of the 15th Conference on Uncertainty in Artificial Intelligence (UAI)*, pages 196–205, 1999.

[11] Markus Kalisch, Martin Mächler, Diego Colombo, Marloes H. Maathuis and Peter Bühlmann. “Causal inference using graphical models with the R package `pcalg`.” *Journal of Statistical Software*, 47(11):1–26, 2012.

[12] Marco Scutari. “Learning Bayesian networks with the `bnlearn` R package.” *Journal of Statistical Software*, 35(3):1–22, 2010.

[13] Diviyan Kalainathan, Olivier Goudet and Ritik Dutta. “Causal Discovery Toolbox: uncovering causal relationships in Python.” *Journal of Machine Learning Research*, 21(37):1–5, 2020.

[14] Yujia Zheng, Biwei Huang, Wei Chen, Joseph Ramsey, Mingming Gong, Ruichu Cai, Shohei Shimizu, Peter Spirtes and Kun Zhang. “causal–learn: causal discovery in Python.” *Journal of Machine Learning Research*, 25(60):1–8, 2024.

[15] Joseph D. Ramsey, Kun Zhang, Madelyn Glymour, Reuben S. Romero, Biwei Huang, Imme Ebert–Uphoff *et al.* “TETRAD — a toolbox for causal discovery.” In *Proceedings of the 8th International Workshop on Climate Informatics*, pages 1–4, 2018.

[16] Diego Colombo and Marloes H. Maathuis. “Order–independent constraint–based causal structure learning.” *Journal of Machine Learning Research*, 15:3741–3782, 2014.

[17] 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, 2006.

[18] Shohei Shimizu, Patrik O. Hoyer, Aapo Hyvärinen and Antti Kerminen. “A linear non–Gaussian acyclic model for causal discovery.” *Journal of Machine Learning Research*, 7:2003–2030, 2006.

[19] Xun Zheng, Bryon Aragam, Pradeep Ravikumar and Eric P. Xing. “DAGs with NO TEARS: continuous optimisation for structure learning.” In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 31, pages 9472–9483, 2018.

[20] Ignavier Ng, AmirEmad Ghassami and Kun Zhang. “On the role of sparsity and DAG constraints for learning linear DAGs.” In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 33, pages 17943–17954, 2020.

[21] Sébastien Lachapelle, Philippe Brouillard, Tristan Deleu and Simon Lacoste–Julien. “Gradient–based neural DAG learning.” In *International Conference on Learning Representations (ICLR)*, 2020.

[22] Shengyu Zhu, Ignavier Ng and Zhitang Chen. “Causal discovery with reinforcement learning.” In *International Conference on Learning Representations (ICLR)*, 2020.

[23] Riccardo Massidda, Francesco Landolfi, Martina Cinquini and Davide Bacciu. “Differentiable causal discovery with smooth acyclic orientations.” In *Proceedings of the 40th International Conference on Machine Learning, Workshop on Differentiable Almost Everything*, 2023.

[24] Janez Demšar. “Statistical comparisons of classifiers over multiple data sets.” *Journal of Machine Learning Research*, 7(1):1–30, 2006.

[25] Jonas Peters and Peter Bühlmann. “Structural intervention distance (SID) for evaluating causal graphs.” *Neural Computation*, 27(3):771–799, 2015.

[26] Gideon Schwarz. “Estimating the dimension of a model.” *Annals of Statistics*, 6(2):461–464, 1978.

[27] David Heckerman, Dan Geiger and David M. Chickering. “Learning Bayesian networks: the combination of knowledge and statistical data.” *Machine Learning*, 20(3):197–243, 1995.

[28] Milton Friedman. “The use of ranks to avoid the assumption of normality implicit in the analysis of variance.” *Journal of the American Statistical Association*, 32(200):675–701, 1937.

[29] Peter B. Nemenyi. *Distribution-free multiple comparisons*. Ph.D. thesis, Princeton University, 1963.

[30] Thomas Verma and Judea Pearl. “Equivalence and synthesis of causal models.” In *Proceedings of the Sixth Annual Conference on Uncertainty in Artificial Intelligence (UAI 1990)*, pages 255–270, 1990.

[31] Steffen L. Lauritzen and David J. Spiegelhalter. “Local computations with probabilities on graphical structures and their application to expert systems.” *Journal of the Royal Statistical Society: Series B (Methodological)*, 50(2):157–194, 1988.

[32] Ingo A. Beinlich, Henri J. Suermondt, R. Martin Chavez, and Gregory F. Cooper. “The ALARM monitoring system: A case study with two probabilistic inference techniques for belief networks.” In *Proceedings of the Second European Conference on Artificial Intelligence in Medicine (AIME 1989)*, pages 247–256, 1989.

[33] 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.

[34] Amit Sharma and Emre Kiciman. “DoWhy: An end-to-end library for causal inference.” *arXiv preprint arXiv:2011.04216*, 2020.

[35] Nicolai Meinshausen and Peter Bühlmann. “High-dimensional graphs and variable selection with the Lasso.” *The Annals of Statistics*, 34(3):1436–1462, 2006.

[36] Juan M. Ogarrio, Peter Spirtes and Joe Ramsey. “A hybrid causal search algorithm for latent variable models.” In *Proceedings of the 8th International Conference on Probabilistic Graphical Models (PGM)*, pages 368–379, 2016.