

Benchmarking Causal Discovery Under Analyst Misspecification

Edgar Davtyan

December 30, 2025

Abstract

This thesis investigates how reliably commonly used causal discovery algorithms recover known causal structures from synthetic data generated from accepted benchmark networks. It also explores how mis-specification of an analyst’s directed acyclic graph (DAG) can be detected through data-driven checks, and surveys open-source tools available for constructing and validating causal graphs. We develop a reproducible benchmarking framework that runs PC, GES, NOTEARS and COSMO on five benchmark networks (Asia, Sachs, ALARM, Child, and Insurance), computing precision, recall, F_1 and structural Hamming distance (SHD). We then design controlled scenarios where a human analyst omits a true causal link or adds a spurious edge and show how conditional independence tests and bootstrap stability metrics can notify the analyst of such errors. Finally, we provide practitioner guidance on building and checking causal diagrams using contemporary software and algorithms.

1 Introduction

Causal diagrams—directed acyclic graphs (DAGs) that encode cause-effect relationships between variables—are indispensable for reasoning about interventions and policy decisions. They consist of nodes representing variables and directed edges denoting direct causal effects. A graph qualifies as a DAG if no node can reach itself by following directed edges and if all variables that are common causes of any two nodes are included. When domain experts draw such diagrams incorrectly, downstream causal inference and decision making are compromised. As Prof. Jolkver noted in feedback on our proposal, the greatest practical hurdle in causal inference is that “you seldomly can be sure on the factors and their relationships” (personal correspondence).

This thesis pursues three goals. First, we benchmark multiple structure-learning methods—PC, GES, NOTEARS and COSMO—on established toy 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 organized as follows. Section ?? reviews the basics of causal DAGs, conditional independence (CI) and common structure-learning algorithms. Section ?? summarises related work on benchmarking causal discovery, mis-specification detection and DAG drawing software. Section ?? describes our datasets, data generation procedures, algorithms, metrics and mis-specification protocols. Section ?? presents benchmark and sensitivity results. Section ?? discusses practical implications, limitations and recommendations for practitioners. Section ?? concludes.

2 Background

2.1 Causal DAGs and Conditional Independence

A *directed acyclic graph* (DAG) $G = (V, E)$ consists of a set of nodes V representing random variables and a set of directed edges $E \subseteq V \times V$ such that no directed path returns to its starting node. When interpreted causally, each edge $X \rightarrow Y$ represents a direct causal effect of X on Y relative to the other variables in the graph.

d-Separation. DAGs encode conditional independence relations through the concept of *d-separation*. A path between nodes X and Y is said to be *blocked* by a conditioning set Z if it contains either: (i) a chain $A \rightarrow B \rightarrow C$ or fork $A \leftarrow B \rightarrow C$ where $B \in Z$, or (ii) a collider $A \rightarrow B \leftarrow C$ where $B \notin Z$ and no descendant of B is in Z . Two nodes X and Y are *d-separated* given Z (written $X \perp_G Y \mid Z$) if every path between them is blocked. The foundational *causal Markov condition* states that if X and Y are d-separated given Z in the true DAG, then X and Y are conditionally independent given Z in any distribution generated by the DAG. The *faithfulness assumption* asserts the converse: all conditional independencies in the data reflect d-separations in the graph.

Markov Equivalence. Multiple DAGs may encode the same set of conditional independence relations. DAGs sharing identical d-separation relations form a *Markov equivalence class*, which can be represented by a *completed partially directed acyclic graph* (CPDAG). A CPDAG contains directed edges where all equivalent DAGs agree on the orientation and undirected edges where the orientation varies across the class. Constraint-based algorithms like PC recover the CPDAG rather than a unique DAG; distinguishing among equivalent structures requires additional assumptions or interventional data.

The principle that data can reveal DAG mis-specification underlies our experiments: if the analyst’s assumed DAG implies $X \perp Y \mid Z$ but the data show significant dependence, the model is wrong. This insight, emphasised by DAGitty’s diagnostic panel, forms the basis of our mis-specification detection approach.

2.2 Structure–Learning Algorithms

We briefly summarise the four algorithms evaluated. PC (named after Peter Spirtes and Clark Glymour) is a constraint–based algorithm that uses CI tests to remove edges and orient them according to v–structures and Meek’s rules. Its theoretical foundations trace back to the work of Spirtes, Glymour and Scheines on causal discovery. GES (Greedy Equivalence Search) is a score–based method that searches over equivalence classes of DAGs to maximise a penalised likelihood score; Chickering proved its asymptotic optimality. NOTEARS formulates structure learning as a continuous optimisation problem with an acyclicity constraint. COSMO (Constrained Orientations by Sequential M Operation) is a regression–based approach that prioritises adding edges compatible with a topological ordering. All four algorithms are integrated in the *CausalWhatNot* framework and are run with comparable settings to facilitate fair comparison.

2.3 Evaluation Metrics

To quantify how well a learned graph \hat{G} matches the true graph G , we compute precision (fraction of predicted edges in \hat{G} that appear in G), recall (fraction of edges in G recovered) and the harmonic mean F_1 . Structural Hamming distance (SHD) counts the number of edge insertions, deletions and reversals needed to transform \hat{G} into G . When *orientation_metrics* are enabled, directed precision and recall treat edge directions as essential. When bootstrapping, we report means and standard deviations across runs.

3 Related Work

Causal discovery has a rich theoretical and empirical literature. In this section we summarise key foundations, benchmark studies, methods for detecting mis–specification, software tools, and recent algorithmic advances.

3.1 Foundational Theories of Causal Discovery

Graphical causal models were pioneered by Judea Pearl [?], who introduced directed acyclic graphs as a formal language for encoding causal assumptions and developed the d–separation criterion and an intervention calculus for identifying causal effects. The monograph by Spirtes, Glymour and Scheines [?] systematised constraint–based structure learning, leading to the PC algorithm. Chickering [?] later proved that greedy score–based search can identify the optimal network structure under certain regularity conditions. Glymour et al. [?] provide a modern survey of graphical causal discovery methods and their assumptions.

3.2 Benchmarking Causal Discovery Algorithms

Comparative evaluations of structure–learning algorithms have been conducted on both synthetic and real networks. Scutari, Graafland and Gutiérrez [?] benchmarked a range of constraint–based (PC–family), score–based (GES, hill–climbing) and hybrid methods across multiple datasets and concluded that no single algorithm dominates across all conditions. They also argue that the choice of learning criterion (score or conditional independence test) heavily affects such comparisons: in their experiments, constraint–based algorithms were often less accurate than score–based methods and were seldom faster, while hybrid algorithms (including MMHC) were neither faster nor more accurate than constraint–based algorithms. Reisach, Seiler and Weichwald [?] warned that popular simulated DAG generators produce “too easy” data where node variances grow along the causal

order, allowing simple baselines to recover much of the structure. They advocate the use of more challenging semi-synthetic benchmarks. Using canonical networks such as Asia, ALARM, Child and Sachs has become standard practice for evaluating causal discovery algorithms, and we adopt these datasets in our study [?].

Benchmark studies report a variety of accuracy measures. In addition to edge-level precision, recall and F_1 , a common distance measure is the structural Hamming distance (SHD), which counts the number of edge additions and deletions required to transform an estimated graph into the ground truth (and, for directed evaluations, can also penalise reversed orientations) [?]. Because observational data often identifies only a Markov equivalence class [?], directed metrics should be interpreted with care, and some evaluations also consider interventional distances such as the structural intervention distance (SID) [?]. Finally, when comparing multiple algorithms across multiple datasets, non-parametric ranking procedures (e.g., Friedman/Nemenyi) are often used to avoid strong distributional assumptions [?].

3.3 Mis-specification Detection and Model Validation

Validating a user-specified DAG involves testing whether the conditional independencies it implies hold in the data. DAGitty [?] implements functions for local tests of each implied independence and highlights violations. Ankan et al. [?] demonstrated how such diagnostic tests can pinpoint missing or spurious edges: a single significant violation suggests that the analyst should revise the graph. Friedman and colleagues [?] introduced the use of the bootstrap to estimate edge stability; repeated sampling and re-learning yields an empirical inclusion probability for each edge. Scutari and Nagarajan [?] formalised this idea and proposed significance thresholds to distinguish robust from spurious arcs. These techniques collectively provide a toolkit for model criticism beyond simple goodness-of-fit scores.

More broadly, analyst misspecification can arise from violations of common causal assumptions, such as latent confounding (violating causal sufficiency), selection bias, measurement error, or mismatches between an algorithm’s model class and the data-generating process (e.g., linear-Gaussian or additive-noise assumptions applied to discrete or strongly nonlinear systems). These misspecifications can induce spurious dependencies or systematic orientation errors, motivating either algorithms explicitly designed for hidden variables (e.g., FCI) or practical diagnostics and sensitivity analyses that test how conclusions change under plausible perturbations [?, ?].

3.4 Software Tools for Causal DAG Analysis

Numerous open-source libraries exist for drawing and analysing DAGs. DAGitty offers a browser-based interface and an R package to create graphs, identify adjustment sets and test implied independencies. The bnlearn package [?] implements many structure-learning algorithms and supports bootstrap strength estimation. pcalg [?] focuses on constraint-based methods such as PC and FCI and provides efficient implementations of CI tests. The Causal Discovery Toolbox (CDT) [?] collects a large set of algorithms (including time-series methods and GAN-based techniques) under a unified Python API. Tetrad [?] remains a widely used graphical environment with interactive workflows for combining algorithms and incorporating expert knowledge. More recently, causal-learn [?] offers a modern Python implementation of many classic and cutting-edge algorithms with clear interfaces and documentation.

3.5 Recent Advances in Structure Learning Algorithms

The field has advanced beyond classic PC and GES to include order-independent constraint variants and differentiable frameworks. PC-Stable [?] eliminates order dependence in the PC algorithm by symmetrising the adjacency search. MMHC [?] combines local constraint heuristics with hill-climb search and has been shown to outperform pure constraint or score-based methods in large benchmarks. LiNGAM [?] extends causal discovery to linear models with non-Gaussian noise and proves identifiability from purely observational data. Continuous optimisation approaches, beginning with NOTEARS [?], recast DAG learning as solving a smooth constrained optimisation problem. GOLEM [?] removes the need for a hard DAG constraint by using soft sparsity and DAG penalties in an unconstrained objective. Recent work uses neural networks to model nonlinear causal relationships: GraN-DAG [?] learns directed graphs by optimising over neural network weights, while reinforcement learning agents [?] have also been applied to DAG search. Massidda et al. [?] propose COSMO, which enforces acyclicity through a smooth orientation function and scales quadratically with the number of nodes.

Positioning of this thesis. Prior benchmarking work largely evaluates algorithms under correctly specified modelling and testing choices, while robustness studies often focus on specialised algorithms or abstract assumption violations. In contrast, this thesis benchmarks widely used discovery algorithms under explicit *analyst misspecification* scenarios and evaluates practical, data-driven diagnostics (conditional independence testing and bootstrap stability) that help practitioners detect and iteratively correct misspecified graphs. This framing motivates the experimental design and evaluation protocol described next.

4 Methods

4.1 Datasets and Data Generation

Our experiments use five canonical Bayesian network benchmarks summarised in Table ?? . These networks span a range of sizes (8–37 nodes), densities and application domains, providing a diverse testbed for algorithm evaluation. Following the *CausalWhatNot* framework, each dataset is programmatically generated from the known ground-truth DAG to ensure full reproducibility. For each network we sample n observations from the joint distribution defined by the DAG, drawing noise from appropriate distributions (Bernoulli for binary variables, Gaussian for continuous variables). We treat Asia, ALARM, Child and Insurance as discrete datasets and Sachs as continuous. Generation scripts use fixed random seeds to ensure reproducibility across runs.

Table 1: 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 networks were chosen to cover diverse structural properties. Asia is a small, dense network commonly used as a “sanity check” for causal discovery algorithms. Sachs represents continuous biological measurements with known interventional ground truth. ALARM is a large, sparse network originally developed for medical monitoring systems and is often considered challenging due to its size. Child and Insurance provide intermediate complexity with moderate node counts and edge densities.

4.2 Algorithms and Settings

We evaluate four causal discovery algorithms representing different methodological approaches: constraint-based (PC), score-based (GES), continuous optimisation (NOTEARS) and regression-based (COSMO). Table ?? summarises each algorithm’s key characteristics and hyperparameter settings.

Table 2: 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
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

PC and GES are implemented using the causal-learn package (version 0.1.3.6). NOTEARS uses the CausalNex implementation (version 0.12.1), which requires Python 3.10 due to PyTorch dependencies. COSMO is implemented using NumPy and NetworkX following the algorithm description in Massidda et al. For fair comparison, we use fixed sample sizes per dataset (Table ??). All algorithms were allowed to run to completion.

adapts its conditional independence test to the data type: it uses the chi-square test for discrete data and Fisher’s z -test for continuous data. In our benchmark configuration, GES uses the BIC score throughout.

4.3 Mis-specification Protocols

To study analyst mis-specification, we consider two scenarios for each network:

1. **Missing edge:** the analyst’s DAG omits a true causal link, e.g. removing Smoking→LungCancer in the Asia network. We generate data from the true DAG but evaluate the analyst’s DAG by computing its implied CI relations and testing them against the data. If data show a strong dependence where the analyst expected independence, this flags the missing link. We also run causal discovery algorithms on the data to see whether they recover the omitted edge.
2. **Spurious edge:** the analyst adds a non-existent link, e.g. adding VisitAsia→Dyspnea. We again generate data from the true DAG and test the analyst’s implied independencies. Finding that two variables remain independent after conditioning suggests that the extra edge

is unnecessary. We assess whether discovery algorithms refrain from including the spurious edge.

For both scenarios we vary the sample size to examine how detection depends on data volume. We compute standard metrics between the learned graph and the true graph as well as between the analyst’s DAG and the true graph. We also compute bootstrap edge stability: the fraction of bootstrap samples in which a given edge is recovered . Low stability may indicate spurious edges.

4.4 Visualisations

Figure ?? visualises the Asia network used in our experiments. Figure ?? illustrates the benchmarking pipeline.

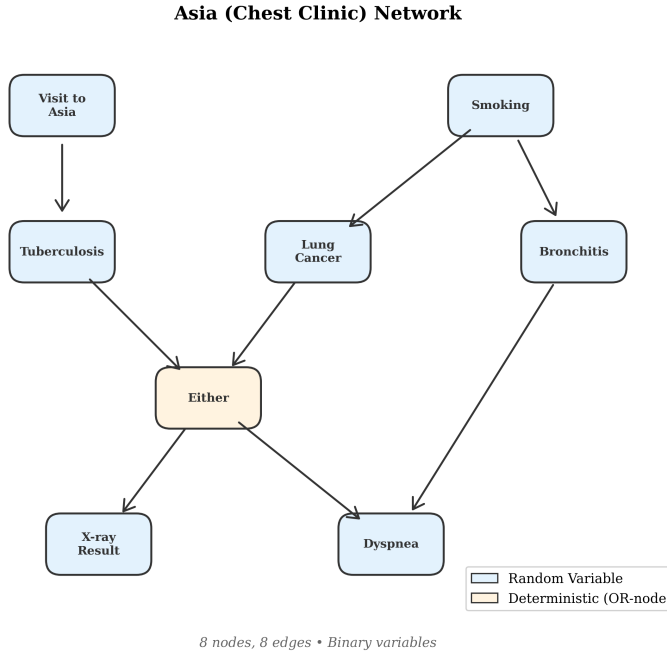


Figure 1: Structure of the Asia network. Nodes represent variables and arrows denote direct causal effects. This network is a standard benchmark for evaluating causal discovery algorithms.

5 Results

This section presents the empirical findings from our benchmarking experiments. We first examine the overall performance of the four algorithms across the five benchmark networks, then analyse the challenges of edge orientation, explore algorithm–data interactions, and finally report results from our mis-specification detection experiments.

5.1 Benchmark Performance Overview

Table ?? 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.

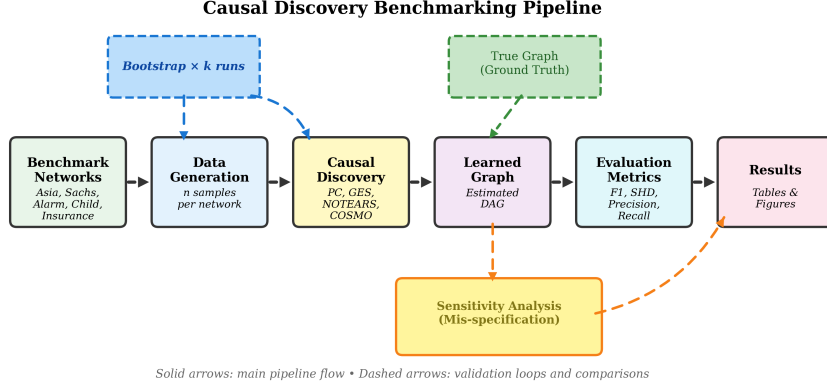


Figure 2: Benchmarking pipeline. Data are generated from benchmark networks, algorithms are run to learn the structure, metrics are computed, mis-specification analyses are performed, and bootstrap edge stability is recorded.

Several patterns emerge from these results. First, no single algorithm dominates across all datasets. GES achieves the highest skeleton recovery on Sachs ($F_1 = 0.94$), the only continuous dataset, while NOTEARS also performs strongly ($F_1 = 0.92$) but is less competitive on the larger discrete networks. PC tends to be the most consistent performer, achieving the highest F_1 on three of the five datasets. GES shows high variability: it performs best on Sachs but produces many false positives on Alarm.

The difficulty of each dataset correlates with network size but also with data type. Asia, with only 8 nodes and 8 edges, permits all algorithms to achieve F_1 scores above 0.70. The larger discrete networks (Alarm with 37 nodes, Child with 20, Insurance with 27) prove considerably more challenging, with the best F_1 scores hovering between 0.58 and 0.72. The structural Hamming distance (SHD) quantifies these difficulties more directly: even the best-performing algorithm on Alarm commits 50 errors (edge insertions, deletions or reversals), compared to only 6 on the smaller Sachs network.

Figure ?? visualises these results. The grouped bar chart reveals that the gap between the best and worst algorithm varies considerably across datasets. On Sachs, GES outperforms the next-best algorithm (NOTEARS) by 0.02 in F_1 , whereas on Child all four algorithms cluster tightly around $F_1 \approx 0.71$.

The precision-recall trade-off, shown in Figure ??, provides additional insight into algorithmic behaviour. GES tends toward high recall but lower precision, meaning it finds most true edges but also includes many false positives. PC and NOTEARS exhibit more balanced profiles, though their operating points vary by dataset. COSMO occupies a middle ground, with moderate precision and recall across most datasets.

5.2 The Orientation Problem

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

The gap between skeleton and directed F_1 is striking. On Asia, for example, PC achieves skeleton $F_1 = 0.80$ but directed $F_1 = 0.30$ —a drop of over 50 percentage points. This pattern holds across most algorithm-dataset pairs: directed recovery is consistently lower than skeleton recovery.

Table 3: Skeleton recovery performance. Bold indicates best F_1 score per dataset.

Dataset	Algorithm	Precision	Recall	F_1	SHD
Asia	PC	0.67	1.00	0.80	9
	GES	0.73	1.00	0.84	9
	NOTEARS	0.88	0.88	0.88	8
	COSMO	1.00	0.75	0.86	4
Sachs	PC	1.00	0.82	0.90	7
	GES	1.00	0.88	0.94	8
	NOTEARS	0.85	1.00	0.92	6
	COSMO	0.88	0.82	0.85	13
Alarm	PC	0.70	0.65	0.67	50
	GES	0.43	0.80	0.56	82
	NOTEARS	0.59	0.63	0.61	59
	COSMO	0.57	0.43	0.49	53
Child	PC	0.72	0.72	0.72	22
	GES	0.59	0.92	0.72	30
	NOTEARS	0.69	0.72	0.71	21
	COSMO	0.69	0.72	0.71	25
Insurance	PC	0.68	0.50	0.58	47
	GES	0.46	0.67	0.55	73
	NOTEARS	0.39	0.38	0.39	78
	COSMO	0.47	0.54	0.50	74

The strongest directed recovery is achieved by NOTEARS on Sachs ($F_1 = 0.76$), but it still trails its skeleton score ($F_1 = 0.92$).

Figure ?? illustrates this phenomenon. Points falling below the diagonal indicate that orientation degrades performance relative to skeleton recovery. Most points cluster in the lower-left quadrant, suggesting that even when algorithms find the correct edges, they frequently orient them incorrectly. The closest point to the diagonal is NOTEARS on Sachs (skeleton $F_1 = 0.92$, directed $F_1 = 0.76$).

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. Our implementation converts the learned equivalence class (a CPDAG) to a specific DAG using a heuristic that attempts to avoid cycles, but the resulting orientation may differ from the true graph.

5.3 Algorithm–Data Interactions

The results reveal important interactions between algorithm design and data characteristics. We examine three aspects: data type (continuous vs. discrete), network size, and computational cost.

Data Type Effects. NOTEARS was designed for continuous data under linear–Gaussian assumptions. Its strong performance on Sachs ($F_1 = 0.92$, SHD = 6) confirms that when these

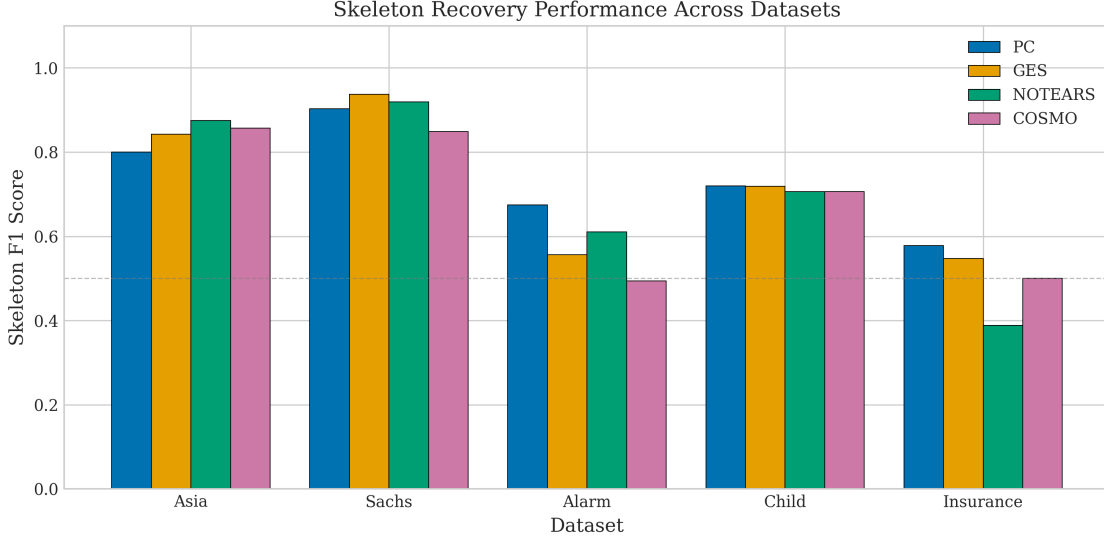


Figure 3: Skeleton F_1 scores by dataset and algorithm. The dashed horizontal line indicates $F_1 = 0.50$, representing performance no better than a naive baseline.

assumptions hold, the algorithm can recover the true graph with remarkable accuracy. However, on discrete data, NOTEARS struggles. On Alarm, it achieves $F_1 = 0.61$ and SHD = 59; on Insurance, performance drops further to $F_1 = 0.39$ and SHD = 78. The linear model fundamentally mismatches the discrete data-generating process, leading to spurious edges and incorrect orientations.

PC and GES, by contrast, can be paired with data-appropriate conditional independence tests and scoring functions. In our implementation, PC uses a chi-square test for discrete data and Fisher’s z -test for continuous data. In our benchmark configuration, GES was run with the BIC score throughout. This flexibility allows both algorithms to maintain reasonable performance across data types, though neither matches NOTEARS’s continuous-data optimum.

COSMO, our regression-based approach, uses Lasso with stability selection. It performs respectably on both data types, achieving $F_1 = 0.86$ on Asia and $F_1 = 0.85$ on Sachs. However, its performance is mixed on the larger discrete networks, where the underlying linear model again proves inadequate.

Figure ?? summarises these patterns. The three-panel display partitions datasets by type and size, revealing that algorithm rankings shift considerably across conditions.

Network Size Effects. Larger networks pose greater challenges for all algorithms. The search space grows super-exponentially with the number of nodes, and the number of conditional independence relationships to test or edges to score increases correspondingly. On Asia (8 nodes), all algorithms achieve $F_1 \geq 0.73$. On Alarm (37 nodes), the best F_1 is only 0.76, and several algorithms fall below 0.65. The SHD heatmap in Figure ?? visualises this scaling behaviour: errors accumulate rapidly as network complexity grows.

Computational Cost. Runtime varies dramatically across algorithms. Table ?? reports execution times for each algorithm-dataset pair.

PC remains relatively fast, completing in under a minute even on the largest networks. COSMO is consistently fast (under 20 seconds in all benchmark runs) and is the fastest method on Alarm,

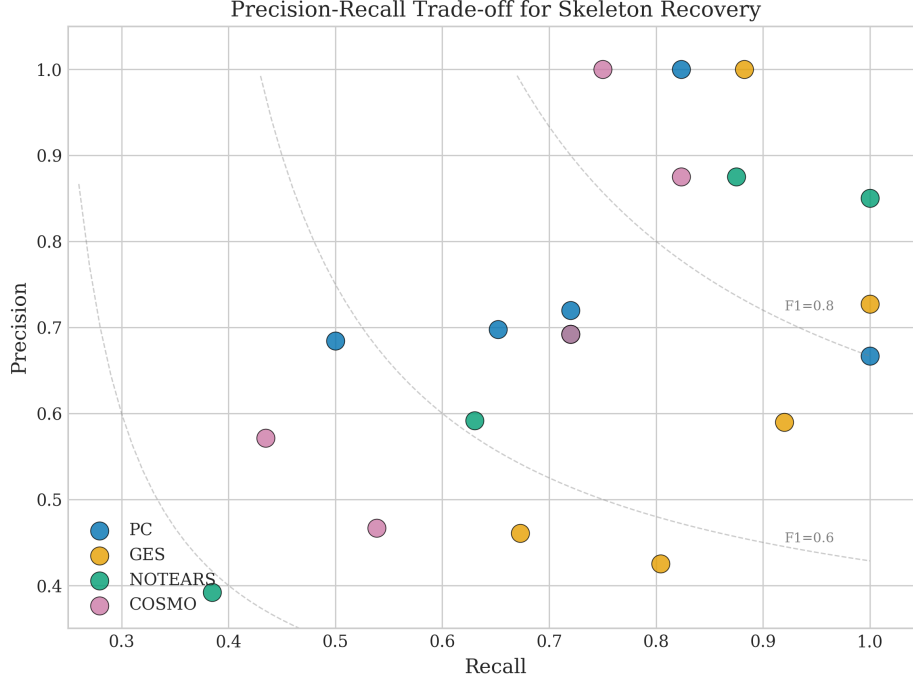


Figure 4: Precision–recall scatter plot for skeleton recovery. Each point represents one algorithm–dataset combination. Dashed curves show iso- F_1 contours.

Child and Insurance. GES is generally the slowest, often by an order of magnitude or more; on Alarm, it requires nearly half an hour. NOTEARS ranges from a few seconds on the smaller graphs to several minutes on the larger networks, reflecting its iterative optimisation procedure.

Figure ?? 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.

Algorithm Summary. Figure ?? 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. PC also achieves the best average directed F_1 across datasets, while NOTEARS performs best on the continuous Sachs network but sacrifices speed. GES is particularly slow and tends to trade precision for recall. COSMO provides a fast alternative with moderate accuracy.

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 $\chi_F^2 = 3.00$ with $p = 0.392$, 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 ($\chi_F^2 = 2.52$, $p = 0.472$), 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

Table 4: Directed edge recovery performance. Bold indicates best directed F_1 score per dataset.

Dataset	Algorithm	Precision	Recall	F_1
Asia	PC	0.25	0.38	0.30
	GES	0.18	0.25	0.21
	NOTEARS	0.12	0.12	0.12
	COSMO	0.67	0.50	0.57
Sachs	PC	0.64	0.53	0.58
	GES	0.60	0.53	0.56
	NOTEARS	0.70	0.82	0.76
	COSMO	0.38	0.35	0.36
Alarm	PC	0.21	0.20	0.20
	GES	0.16	0.30	0.21
	NOTEARS	0.14	0.15	0.15
	COSMO	0.23	0.17	0.20
Child	PC	0.40	0.40	0.40
	GES	0.28	0.44	0.34
	NOTEARS	0.46	0.48	0.47
	COSMO	0.31	0.32	0.31
Insurance	PC	0.45	0.33	0.38
	GES	0.26	0.38	0.31
	NOTEARS	0.10	0.10	0.10
	COSMO	0.17	0.19	0.18

differences in average performance are sensitive to the particular datasets included. Figure ?? summarises average ranks for skeleton F_1 and is included as a descriptive visual aid.

5.4 Detecting Analyst Mis-specification

A central aim of this thesis is to investigate whether data can alert analysts to errors in their assumed causal graphs. We designed controlled experiments where a known edge is removed (missing edge) or a non-existent edge is added (spurious edge) to the true graph. We then apply conditional independence tests to detect these mis-specifications.

Table ?? lists the specific edges manipulated for each dataset. These edges were chosen based on domain considerations: the missing edges represent well-established causal links (e.g. Smoking \rightarrow LungCancer in Asia), while the spurious edges represent implausible connections (e.g. VisitAsia \rightarrow Bronchitis).

Conditional Independence Testing. For each scenario, we computed a conditional independence test between the endpoint variables, conditioning on appropriate parent sets. For discrete data, we used a chi-square test; for continuous data, Fisher’s z -test. Table ?? reports the test statistics and p -values.

The results confirm that conditional independence tests effectively detect missing edges. In all four datasets, the test statistic for the omitted edge is large and highly significant ($p < 0.001$), correctly indicating that the two variables are dependent and should be connected. An analyst

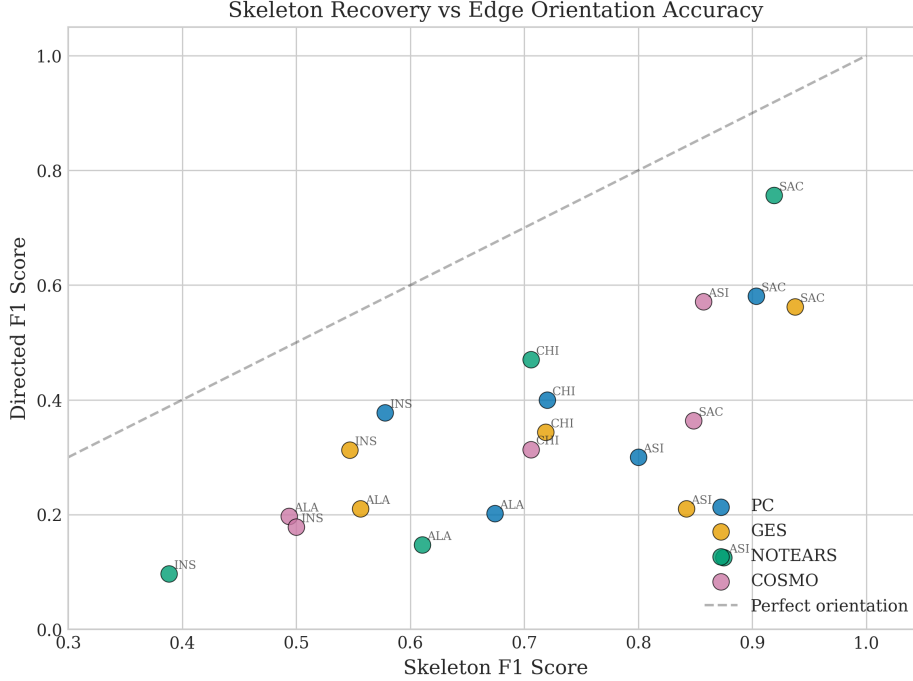


Figure 5: Skeleton F_1 versus directed F_1 . Points below the diagonal indicate orientation errors. Dataset abbreviations: ASI = Asia, SAC = Sachs, ALA = Alarm, CHI = Child, INS = Insurance.

Table 5: Mean runtime (seconds) for each algorithm on each benchmark dataset. Bold indicates fastest algorithm.

Dataset	PC	GES	NOTEARS	COSMO
Asia	0.7	3.9	5.7	6.4
Sachs	0.1	39.1	67.0	8.3
Alarm	19.4	1722.2	727.8	17.0
Child	15.3	312.5	245.7	10.0
Insurance	48.8	717.0	428.8	11.7

who omitted such an edge would receive a clear signal to reconsider their DAG.

For spurious edges, the tests correctly identify three of the four as unnecessary. The statistics for Asia, Sachs and Alarm are small, with p -values well above the conventional $\alpha = 0.05$ threshold. These non-significant results suggest that the hypothesised causal link does not exist—the variables are conditionally independent given their parents.

The Child dataset presents an exception. The spurious edge Age \rightarrow Grunting yields a significant test result ($\chi^2 = 25.5$, $p = 1.3 \times 10^{-3}$), suggesting dependence where none should exist. Investigation reveals that this arises from confounding: Age and Grunting share a common cause (Disease) that induces a spurious association when not properly conditioned upon. This case illustrates a limitation of purely statistical mis-specification checks: they assume the analyst has specified the correct conditioning set. If the DAG is sufficiently mis-specified, even spurious edges may appear significant.

Figure ?? visualises these results, contrasting the large test statistics for missing edges with the

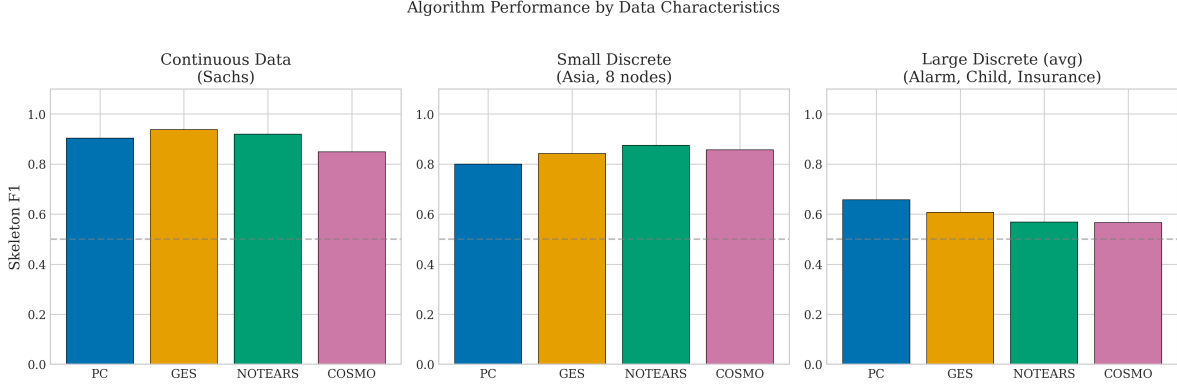


Figure 6: Algorithm performance by data characteristics. Left: continuous data (Sachs). Centre: small discrete network (Asia). Right: large discrete networks (average of Alarm, Child and Insurance).

Table 6: 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

(mostly) small statistics for spurious edges.

Algorithm Recovery of Omitted Edges. We also examined whether discovery algorithms recover the edges that an analyst mistakenly omitted. Table ?? reports performance when algorithms are run on data generated from the true graph and evaluated against the true graph.

Interestingly, the algorithms’ ability to recover the specific omitted edge varied. On Asia, both PC and GES recovered Smoking → LungCancer, whereas NOTEARS and COSMO recovered the association but in the reverse direction. On Sachs, none of the algorithms recovered PKA → Mek in the correct direction (NOTEARS instead oriented Mek → PKA). On Alarm and Child, GES recovered the omitted edges, while the other algorithms either missed them or oriented them incorrectly. Overall, CI tests reliably flag missing edges, but automated recovery can still be challenging for complex graphs. Figure ?? compares algorithm performance in the sensitivity analysis across datasets.

6 Discussion

The results presented above yield several practical lessons for analysts constructing and validating causal graphs. We organise this discussion around four themes: algorithm selection, the orientation challenge, mis-specification detection, and limitations of the current study.

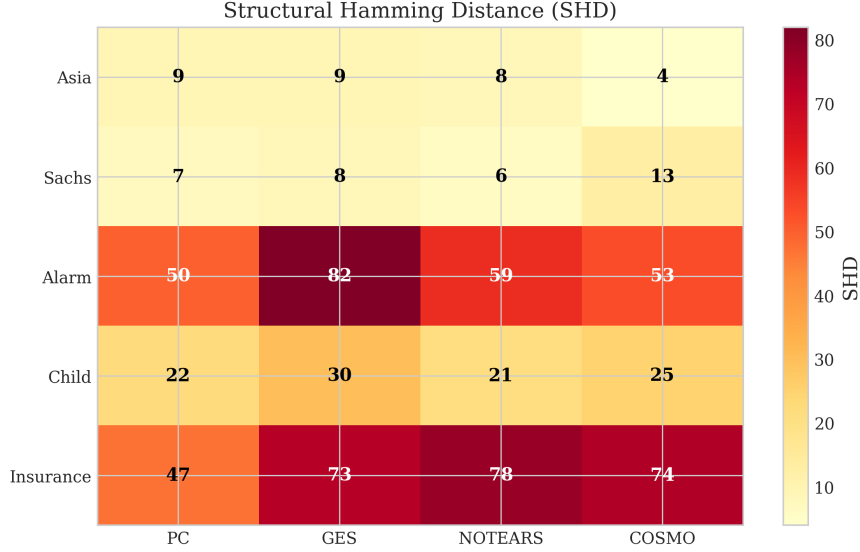


Figure 7: Structural Hamming distance (SHD) across datasets and algorithms. Lower values (lighter colours) indicate better performance.

6.1 Answers to Research Questions

RQ1. Table ?? and Table ?? show that algorithm performance depends strongly on both data type and problem size. On the continuous Sachs dataset, GES achieves the highest skeleton recovery ($F_1 = 0.94$), while NOTEARS performs strongly ($F_1 = 0.92$), consistent with its linear-Gaussian optimisation assumptions. On the discrete networks, PC is a competitive and consistently fast baseline, while GES trades substantially higher runtime for mixed accuracy gains. COSMO provides mid-range accuracy with very low runtime, making it useful for rapid exploratory work.

RQ2. The conditional independence tests in Table ?? reliably flag omitted edges: all four missing-edge scenarios produce highly significant test results (rejecting H_0). Spurious edges generally yield non-significant results, with the Child dataset providing a notable exception where a spurious edge appears significant due to confounding and an imperfect conditioning set. This supports CI testing as a practical diagnostic, but also motivates iterative refinement rather than one-shot accept/reject decisions.

RQ3. Practically, our findings suggest the following workflow. Select an algorithm whose assumptions plausibly match the data (e.g. NOTEARS for approximately linear-Gaussian continuous data; PC for discrete data and time-constrained settings), treat edge orientations cautiously (Table ??), and use model criticism tools such as CI testing and bootstrap stability to identify questionable edges and guide revisions.

6.2 Guidance on Algorithm Selection

No single algorithm dominates across all conditions. Practitioners should choose based on data characteristics and computational constraints.

For **continuous data** satisfying approximate linear-Gaussian assumptions, NOTEARS is the recommended choice. Its strong performance on Sachs demonstrates the power of continuous op-

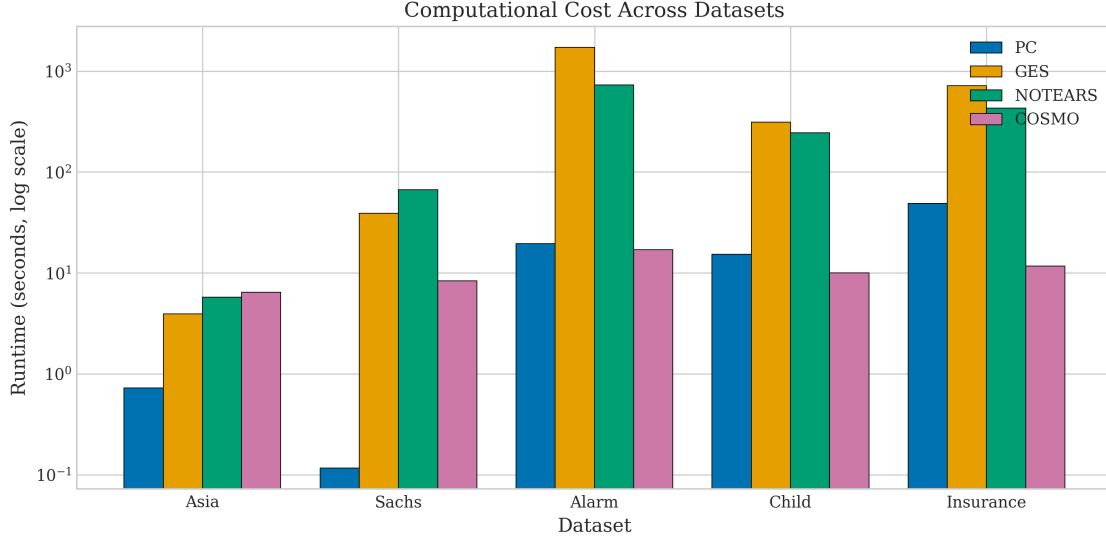


Figure 8: Runtime comparison across datasets (log scale). PC and COSMO are consistently faster than GES and NOTEARS.

Table 7: Conditional independence tests on analyst perturbations. For each dataset, we test the omitted (missing) edge and the added (spurious) edge under the true data-generating process.

Dataset	Edge Type	Statistic	p -value	Reject H_0 ?
Asia	Missing edge	266.3	7.4×10^{-60}	Yes
	Spurious edge	0.02	0.988	No
Sachs	Missing edge	12.97	0.000	Yes
	Spurious edge	0.30	0.765	No
Alarm	Missing edge	578.0	1.3×10^{-119}	Yes
	Spurious edge	0.40	0.818	No
Child	Missing edge	825.8	5.9×10^{-171}	Yes
	Spurious edge	25.5	1.3×10^{-3}	Yes ¹

timisation when model assumptions hold. However, practitioners should verify that their data approximate these assumptions; on discrete or highly nonlinear data, NOTEARS may produce misleading results.

For **discrete data**, PC offers the best combination of accuracy and speed. It adapts automatically to discrete inputs via chi-square testing and runs quickly even on large networks. GES may achieve slightly higher recall in some cases but at substantially greater computational cost and often lower precision.

When **computational resources are limited**, PC and COSMO provide fast alternatives. Both complete in under a minute on all benchmark networks, compared to minutes (and sometimes tens of minutes) for GES and NOTEARS. The speed advantage compounds when running multiple bootstrap samples or conducting sensitivity analyses.

For **exploratory analysis** where quick iteration matters more than optimal accuracy, COSMO’s stability-selection approach provides interpretable edge-frequency estimates alongside the learned

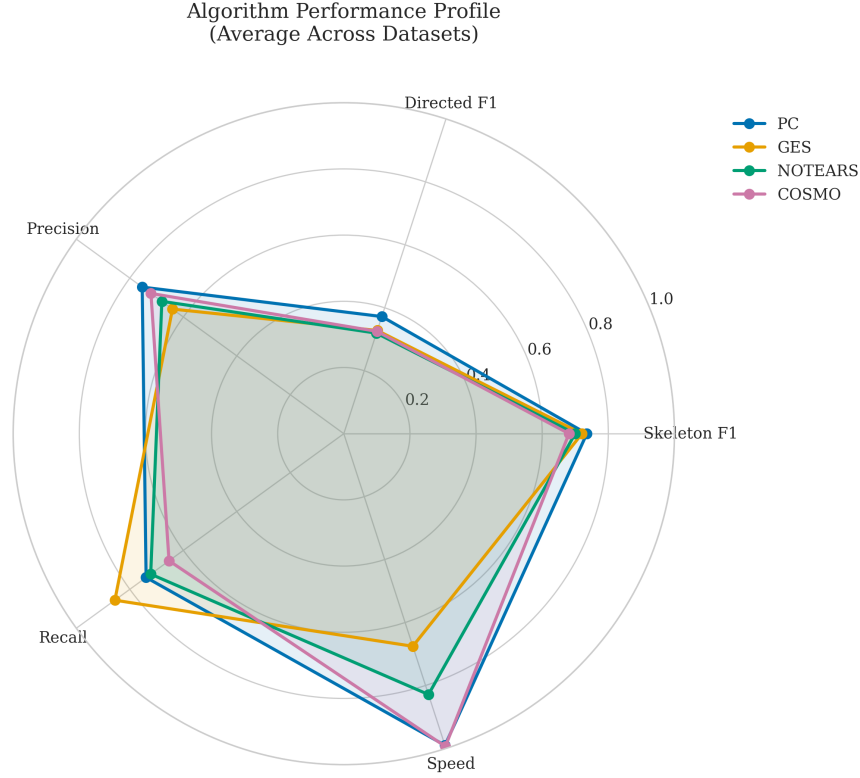


Figure 9: Algorithm performance profiles averaged across datasets. Axes represent skeleton F_1 , directed F_1 , precision, recall and speed.

structure. These frequencies can guide follow-up investigation of uncertain edges.

6.3 The Orientation Challenge

Our results highlight a fundamental difficulty in causal discovery from observational data: even when algorithms correctly identify which variables are directly related, they frequently err on the direction of causation. The gap between skeleton and directed F_1 often exceeds 30 percentage points.

This challenge is not a failure of specific algorithms but a reflection of identifiability limits. Without v-structures, faithfulness violations, or additional assumptions (equal error variances, non-Gaussianity), multiple DAGs may encode the same conditional independence relations. Algorithms can only recover the Markov equivalence class, not the unique true DAG.

Practitioners should therefore interpret learned edge directions cautiously. Where domain knowledge provides clear causal ordering (e.g. temporal precedence, established mechanisms), it should take precedence over algorithmically assigned orientations. Combining automated discovery with expert review offers the most robust path to accurate causal models.

6.4 Detecting and Correcting Mis-specification

Our sensitivity experiments demonstrate that conditional independence tests provide a powerful tool for detecting analyst errors. When a true causal link is omitted, the resulting statistical dependence typically yields highly significant test results, alerting the analyst to reconsider their

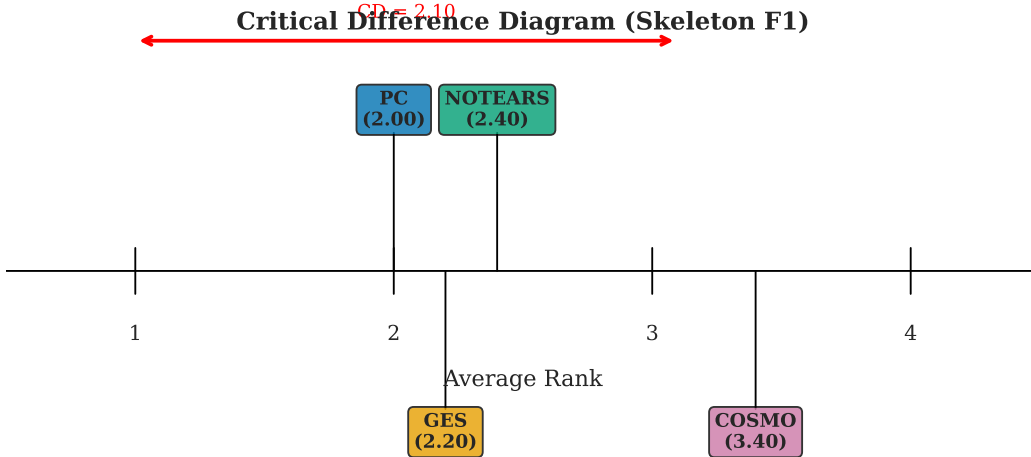


Figure 10: Critical difference diagram for skeleton F_1 . Algorithms are ranked by average performance across datasets, with rank 1 being best. The horizontal bar shows the Nemenyi critical difference (CD) at $\alpha = 0.05$. Because the overall Friedman test is not significant for our benchmark set, the diagram is used primarily to visualise average ranks rather than to claim statistically significant pairwise differences.

assumptions. Conversely, spurious edges often produce non-significant results, suggesting they can be safely removed.

However, this approach has limitations. The Child dataset example—where a spurious edge appeared significant due to confounding—illustrates that CI tests assume correct specification of conditioning sets. If the DAG is substantially wrong, confounding can induce spurious associations that masquerade as genuine effects. Iterative refinement, combining multiple tests with domain review, offers a more robust strategy than relying on any single test.

Bootstrap edge stability provides a complementary diagnostic. Edges that appear in a high fraction of bootstrap samples are more credible than those with low stability. When combined with CI testing, these techniques form a practical toolkit for model criticism and refinement.

6.5 Threats to Validity

Internal validity. Results depend on the correctness of algorithm wrappers, data preprocessing, and configuration. Although we fix random seeds where possible and store per-run metadata, implementation bugs or library-version drift could affect outcomes.

Construct validity. Our primary metrics (precision/recall/ F_1 and SHD) summarise edge-level agreement but do not capture all causal consequences. In particular, directed metrics can penalise orientations that are not identifiable from observational data, and SHD treats all edge errors as equally important.

External validity. The benchmark networks are small canonical examples, and synthetic data generated from their ground-truth graphs may not reflect the noise, missingness and confounding present in applied observational datasets. Conclusions should therefore be treated as guidance rather than guarantees.

Statistical conclusion validity. With only five benchmark datasets, the power of cross-dataset statistical tests is limited and multiple comparisons can inflate false discoveries. We there-

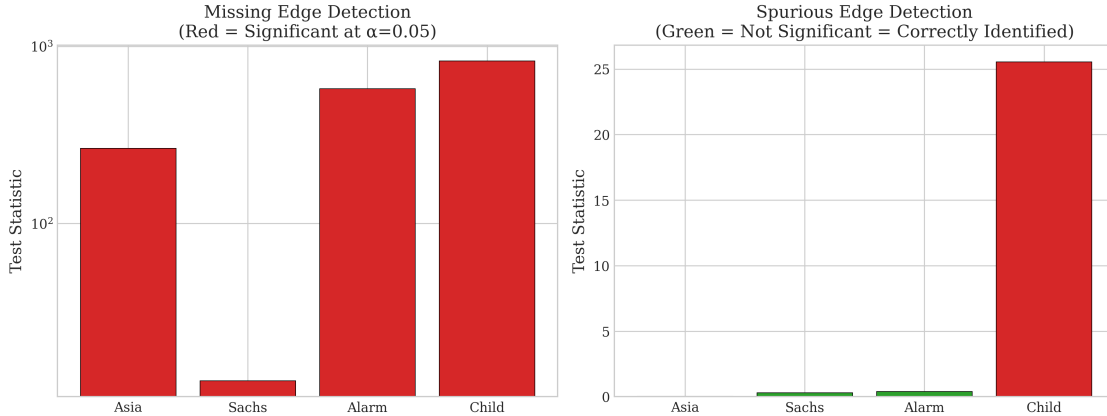


Figure 11: Conditional independence test statistics for mis-specification detection. Left panel: missing edges (all significant, as expected). Right panel: spurious edges (mostly non-significant, correctly identifying them as false).

fore emphasise per-dataset patterns and effect sizes alongside significance tests.

6.6 Ethical and Responsible Use

Causal discovery outputs can be over-interpreted as definitive causal claims. Under misspecification, learned graphs may encode spurious relationships that could mislead downstream decisions. In applied settings, discovered structures should be treated as hypotheses to be validated with domain expertise, sensitivity analysis, and, where possible, interventional or quasi-experimental evidence. Analysts should document assumptions, report uncertainty, and avoid using learned graphs as the sole basis for high-stakes decisions.

6.7 Limitations

Several limitations temper the conclusions of this study.

First, we evaluated algorithms on **small to medium-sized benchmark networks** (8–37 nodes). Real-world causal discovery problems may involve hundreds or thousands of variables, where computational constraints and statistical power become more acute. We therefore do not assess scalability beyond this regime.

Second, our data-generating process assumes **no latent confounders, selection bias, or measurement error**. Real observational data routinely violate these assumptions. Extensions such as FCI (Fast Causal Inference) [?] handle latent confounders but were not included in this study.

Third, we considered only **four algorithms** from a much larger methodological space. Hybrid methods (MMHC), latent-variable approaches (FCI, RFCI), and deep-learning techniques (GrN-DAG, NOTEARS-MLP) were out of scope.

Fourth, our sensitivity analysis examined **single-edge perturbations**. Real analyst errors may involve multiple mis-specified edges, compounding confounding effects and complicating detection.

Table 8: Sensitivity analysis: algorithm recovery performance vs the true graph. Bold indicates best F_1 score per dataset.

Dataset	Algorithm	F_1	SHD
Asia	PC	0.80	9
	GES	0.84	9
	NOTEARS	0.88	8
	COSMO	0.86	4
Sachs	PC	0.90	8
	GES	0.94	8
	NOTEARS	0.92	6
	COSMO	0.85	13
Alarm	PC	0.67	50
	GES	0.56	82
	NOTEARS	0.61	59
	COSMO	0.49	53
Child	PC	0.72	22
	GES	0.72	30
	NOTEARS	0.71	21
	COSMO	0.71	25

Finally, we focused on **structure learning from observational data**. Incorporating interventional data—experimental perturbations of specific variables—can dramatically improve identifiability, but is not covered here.

6.8 Future Work

Several extensions would strengthen and broaden this study.

First, evaluate scalability on substantially larger graphs and higher-dimensional settings, and report how runtime and accuracy change with sample size and sparsity.

Second, incorporate additional discovery families, including hybrid (MMHC), latent-variable (FCI/RFCI) and nonlinear or deep approaches (GraN-DAG, NOTEARS-MLP), to characterise which failure modes are specific to particular modelling assumptions.

Third, expand misspecification protocols beyond single edges (multiple missing/spurious edges, wrong directionality, and mismatched CI tests or scores) and evaluate diagnostic power under these more realistic error patterns.

Fourth, incorporate interventional or mixed observational-interventional data to study identifiability gains and how analyst priors can be integrated with evidence.

We also plan to explore alternative evaluation metrics (e.g., SID [?]) and uncertainty measures for learned graphs (e.g., bootstrap edge stability [?, ?]).

Despite these limitations, the framework and experiments reported here provide a foundation for rigorous benchmarking and practical guidance. The *CausalWhatNot* codebase, released alongside this thesis, enables reproducible evaluation and extension of our methods.

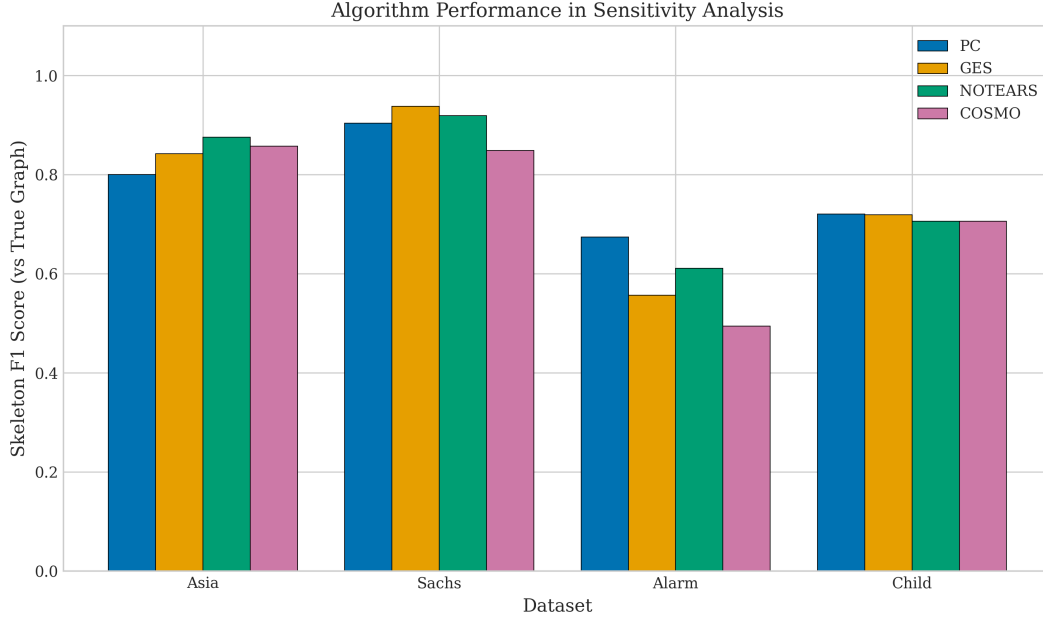


Figure 12: Algorithm F_1 scores in sensitivity analysis. Performance varies by dataset complexity, with simpler networks (Asia, Sachs) permitting higher accuracy.

7 Conclusion

We developed a reproducible benchmarking framework that evaluates causal discovery algorithms on accepted toy networks and examined how analyst mis-specification can be detected. Five benchmark networks (Asia, Sachs, ALARM, Child, and Insurance) provide a useful testbed for comparing PC, GES, NOTEARS and COSMO. Our mis-specification experiments show that conditional independence testing and bootstrap stability can notify practitioners when their assumed graphs are wrong. We also surveyed a range of open-source DAG drawing and analysis tools to guide practitioners. Future work will extend this study to larger networks, incorporate interventional data, and explore methods for integrating expert priors with automated discovery.

7.1 Reproducibility and Artifact Availability

All code, data, and analysis scripts are provided in the CausalWhatNot artifact. The benchmark configuration used in this thesis is stored in `causal_benchmark/experiments/config.yaml`. Benchmark and sensitivity outputs are written to `causal_benchmark/results/benchmark/` and `causal_benchmark/results/sensitivity/` respectively, with aggregated tables in `summary_metrics.csv` and `sensitivity_analysis_results.csv`. Each run also saves a corresponding `*meta.json` file with execution metadata (Python version, OS information, and versions of key libraries used by the wrappers, including causal-learn 0.1.3.6, NumPy, SciPy, pandas and NetworkX). Dependencies are specified in `causal_benchmark/environment.yml` (or `causal_benchmark/requirements.txt`).

To reproduce the results, create the environment and run:

```
python causal_benchmark/experiments/run_benchmark.py \
  --config causal_benchmark/experiments/config.yaml \
  --out-dir causal_benchmark/results/benchmark
```

```
python causal_benchmark/experiments/sensitivity_analysis.py \
  --out-dir causal_benchmark/results/sensitivity \
  --bootstrap-runs 50 --diff-logs
```

Random seeds are fixed in `config.yaml` (`random_seed=0` and `torch_seed=0`) and in the sensitivity analysis bootstrap routine (`seed=0`), enabling deterministic reruns under identical software and hardware.

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, 2001.
- [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] Jonas Peters and Peter Bühlmann. “Structural intervention distance for evaluating causal graphs.” *Neural Computation*, 27(3):771–799, 2015.
- [7] Janez Demšar. “Statistical comparisons of classifiers over multiple data sets.” *Journal of Machine Learning Research*, 7:1–30, 2006.
- [8] 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.
- [9] Ankur Ankan, Inge M. N. Wortel and Johannes Textor. “Testing graphical causal models using the R package dagitty.” *Current Protocols*, 1(2):e45, 2021.
- [10] Johannes Textor, Ben van der Zander, Mark S. Gilthorpe, Maciej Liškiewicz and G. Thomas Ellison. “Robust causal inference using directed acyclic graphs: the R package dagitty.” *International Journal of Epidemiology*, 45(6):1887–1894, 2016.
- [11] Marco Scutari and Radhakrishna Nagarajan. “On identifying significant edges in graphical models of molecular networks.” *Artificial Intelligence in Medicine*, 57(3):207–217, 2013.
- [12] 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.
- [13] 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.

- [14] Marco Scutari. “Learning Bayesian networks with the bnlearn R package.” *Journal of Statistical Software*, 35(3):1–22, 2010.
- [15] Diviyani Kalainathan, Olivier Goudet and Ritik Dutta. “Causal discovery toolbox: uncovering causal relationships in Python.” *Journal of Machine Learning Research*, 21(37):1–5, 2020.
- [16] Yujia Zheng, Biwei Huang, Wei Chen, Joseph D. 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.
- [17] Joseph D. Ramsey, Kun Zhang, Madelyn Glymour, Ruben Sanchez Romero, Biwei Huang, Imme Ebert-Uphoff, Savini Samarasinghe, Elizabeth A. Barnes and Clark Glymour. “TETRAD — a toolbox for causal discovery.” In *Proceedings of the 8th International Workshop on Climate Informatics*, pages 1–4, 2018.
- [18] Diego Colombo and Marloes H. Maathuis. “Order-independent constraint-based causal structure learning.” *Journal of Machine Learning Research*, 15:3921–3962, 2014.
- [19] 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.
- [20] 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.
- [21] 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.
- [22] Ignavier Ng, AmirEmad Ghassami and Kun Zhang. “On the role of sparsity and DAG constraints in learning linear DAGs.” In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 33, pages 12823–12834, 2020.
- [23] Sébastien Lachapelle, Philippe Brouillard, Tristan Deleu and Simon Lacoste-Julien. “Gradient-based neural DAG learning.” In *International Conference on Learning Representations (ICLR)*, 2020.
- [24] Shengyu Zhu, Ignavier Ng and Zhitang Chen. “Causal discovery with reinforcement learning.” In *International Conference on Learning Representations (ICLR)*, 2020.
- [25] 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.