From Guess2Graph: When and How Can Unreliable Experts Safely Boost Causal Discovery in Finite Samples?

Sujai Hiremath[1*], **Dominik Janzing**[2], **Philipp M. Faller**[3*],
Patrick Blöbaum[2], **Elke Kirschbaum**[2], **Shiva Prasad Kasiviswanathan**[2], **Kyra Gan**[1]
[1]Cornell Tech, [2]Amazon, [3]Karlsruhe Institute of Technology

Abstract

Causal discovery algorithms often perform poorly with limited samples. While integrating expert knowledge (including from LLMs) as constraints promises to improve performance, guarantees for existing methods require perfect predictions or uncertainty estimates, making them unreliable for practical use. We propose the *Guess2Graph* (G2G) framework, which uses expert guesses to guide the *sequence* of statistical tests rather than replacing them. This maintains statistical consistency while enabling performance improvements. We develop two instantiations of G2G: PC-Guess, which augments the PC algorithm, and gPC-Guess, a learning-augmented variant designed to better leverage high-quality expert input. Theoretically, both preserve correctness regardless of expert error, with gPC-Guess provably outperforming its non-augmented counterpart in finite samples when experts are "better than random." Empirically, both show monotonic improvement with expert accuracy, with gPC-Guess achieving significantly stronger gains.

1 INTRODUCTION

Global causal discovery provides a principled framework for inferring causal graphs from observational data, with algorithms that are asymptotically correct and statistically well-characterized (Spirtes et al., 2000). In finite samples, however, these guarantees fail to hold (Uhler et al., 2013), and performance tends to degrade substantially with insufficient data (Zuk et al., 2012). As a result, discovery in real-world applications often yields unstable (Faller et al., 2024) or inaccurate (Brouillard et al., 2025) graphs, which contradict established domain knowledge (Maasch et al., 2024).

Expert-aided discovery methods often mitigate finite-sample issues by incorporating domain knowledge, encoding such knowledge as either hard constraints that prune the search space (Ankan and Textor, 2025) or as soft priors that bias the results of statistical tests (Constantinou et al., 2023). Classical expert-aided methods rely on human experts to specify much of the constraint (Tennant et al., 2021; Petersen et al., 2021); however, as graph size grows, this dependence on human input becomes cognitively infeasible and economically unsustainable. *Large language models* (LLMs), trained on vast and diverse corpora, encode broad domain knowledge, suggesting they could serve as scalable proxies for human experts. Much like a domain expert reasoning about plausible causal mechanisms, an LLM can parse variable names and draw on its internalized knowledge to propose causal constraints, potentially reducing reliance on exhaustive statistical testing. Their promise in retrieving known direct relationships (Feng et al., 2025) and ancestral orderings (Vashishtha et al., 2025) make them a compelling alternative, spurring research into their use as expert replacements for causal discovery (Ban et al.; Cohrs et al., 2024; Darvariu et al., 2024; Xie et al., 2024; Kıcıman et al., 2024; Vashishtha et al., 2025).

Yet neither human experts nor LLMs are infallible sources of causal knowledge. Human input is prone to bias and inconsistency (Dror, 2020), while LLMs exhibit critical limitations: they often output invalid graphs (Jiralerspong et al., 2024), are brittle to prompt variations (Ban et al.), degrade with increasing complexity (Sun and Li, 2024), perform poorly on out-of-distribution domains (Feng et al., 2025), produce unreliable reasoning (Ye et al., 2024; Dong et al., 2024), as well as poor uncertainty calibration (Manggala et al., 2025). Given the potential for error, traditional methods that leverage such unreliable experts to generate

Proceedings of the 29th International Conference on Artificial Intelligence and Statistics (AISTATS) 2026, Tangier, Morocco. PMLR: Volume 300. Copyright 2026 by the author(s). *Work done while at Amazon Research Tübingen.

hard or soft constraints lack theoretical guarantees (Wu et al., 2025)—this means that misleading advice can cause error (sometimes unbounded error, see Appendix A.1) even in the large sample limit, rendering them unsuitable for safety-critical applications. To address this, we argue that *expert guidance should not replace statistical procedures, but rather, complement them to improve finite-sample efficiency without sacrificing worst-case guarantees.*

This principle directly motivates our two core research questions. First, we ask: *What formal framework ensures that integrating an unreliable expert never harms, and may improve, algorithmic performance?* Establishing this framework, however, reveals a second critical challenge. The rigid, statistically-optimized architectures of many high-performance algorithms (e.g., the PC algorithm's fixed-size conditioning set iteration) inherently resist external guidance. This leads to our second question: *How can we redesign such algorithms to become more receptive to this framework, thereby unlocking greater performance gains from accurate expertise?*

Contributions This work introduces a principled approach for leveraging fallible experts in causal discovery, addressing the dual challenges of *when* such guidance is safe and *how* to best implement it. Our contributions are fourfold:

- **A Framework for Expert-Guided Causal Discovery**: We introduce the Guess2Graph (G2G) (Sec. 3) framework, which enables causal discovery algorithms to incorporate expert predictions while guaranteeing statistical consistency (**C1**), providing a pathway to achieve monotonic improvement (**C2**) and finite-sample robustness (**C3**). G2G uses expert predictions to guide test sequences rather than outcomes, requiring no uncertainty quantification. We provide the theoretical foundation for instantiating G2G in constraint-based algorithms through subroutine modification (Sec. 4).
- **PC-Guess**: We instantiate G2G in the PC algorithm (Sec. 5.1) to create PC-Guess (Alg. 5), which maintains **C1** while partially achieving **C2** and **C3** through per-iteration performance guarantees. We prove that when starting from identical states, each iteration of PC-Guess shows provable improvement over standard PC with experts 'better than random' (Thm. 5.2), though cascading effects prevent end-to-end guarantees.
- **Augment PC for Better Expert Guidance (gPC-Guess)**: We demonstrate that the rigid, statistically optimized structure of PC limits the potential gains from expert guidance (Sec. 5.2). To overcome this, we propose a redesigned algorithm, gPC-Guess (Alg. 6), which modifies PC's core procedure to be more susceptible to expert input. This approach fully achieves all three criteria **C1**-**C3**, with provable end-to-end finite-sample performance improvements that increase monotonically with expert quality (Thms. 5.1, 5.3).
- **Empirical Validation and Insights**: Our experiments (Sec. 6) validate the theoretical distinction between algorithm augmentation and redesign for expert guidance. PC-Guess shows modest gains (up to 5%) limited by PC's inherent rigidity. In contrast, gPC-Guess fully achieves all three criteria, with up to 30% performance gains when experts are accurate, across both synthetic and real-world data. These results persist with LLM experts, confirming that full achievement of our criteria requires algorithmic redesign rather than simple augmentation.

2 RELATED WORK

Global Causal Discovery

Global causal discovery methods can be defined broadly into three classes: constraint-based, score-based, and functional causal model (FCM) based. Constraint-based methods perform conditional independence tests (Spirtes et al., 2000; Glymour et al., 2019; Spirtes, 2001), score-based methods employ search procedures to find a graph maximizing a scoring function (Chickering, 2002), while FCM methods conduct residual independence tests (Zhang and Hyvarinen, 2009). The problem of error propagation from sequential testing is widespread in the discovery literature. This manifests acutely in constraint-based and FCM methods, which generally suffer from error propagation from diminishing test power in super-exponential search spaces (Lee et al., 2025; Chickering, 2020). Score-based methods that traverse the search space through a sequence of local moves—such as hill climbing (Tsamardinos et al., 2006) and GES (Chickering, 2002)—suffer from error propagation dependent on tests used to evaluate moves (Chickering, 2020).

However, there exist other score-based approaches that avoid the problem of order-dependent sequential testing, though often at the cost of additional parametric assumptions (Deng et al., 2024), relaxing consistency guarantees (Zheng et al., 2020), or sacrificing scalability (Hyttinen et al., 2014). See Appendix A.2 for an in-depth discussion of each class of discovery methods.

While prior work has addressed the sequential testing problem by eliminating order-dependence (i.e., PC-Stable (Colombo and Maathuis, 2014)), we instead optimize test sequences using expert predictions. Our approach applies broadly since sequential testing underlies constraint-based methods, hybrid FCM methods (Peters et al., 2014; Hiremath et al., 2025) and

hybrid score-based (Tsamardinos et al., 2006; Chickering, 2020; Zhu et al., 2024) methods.

Expert-Aided Discovery Existing frameworks for integrating expert predictions into causal discovery face two primary challenges. *Direct hard or soft constraint-based approaches* (Ban et al., 2023; Susanti and Faber, 2025; Takayama et al., 2025) use expert outputs to replace or bias statistical procedures, risking unbounded error propagation with incorrect experts (Hasan and Gani, 2024) and suffering from poorly calibrated confidence scores (Campbell and Moore, 2024; Wu et al., 2025). *Guidance-based approaches* (Constantinou et al., 2023; Wu et al., 2025; Ejaz and Bareinboim, 2025) use predictions for algorithm guidance without replacing tests, but provide limited benefits—initialization helps only with near-perfect experts, while heuristic guidance lacks performance guarantees. Both approaches rely on empirical validation rather than robust statistical foundations, limiting reliability (Appendix A.3).

Algorithms with Predictions Our work builds on the *algorithms with predictions* or the *learning-augmented algorithms* paradigm (Mitzenmacher and Vassilvitskii, 2020), which integrates predictions into classical algorithms to improve performance while preserving worst-case guarantees. In online settings where data or requests arrive sequentially, this approach provides *approximation ratios* guarantees via consistency (near-optimal performance with accurate predictions) and robustness (bounded worst-case performance with poor predictions) (Lykouris and Vassilvitskii, 2018; Wei and Zhang, 2020; Jin and Ma, 2022; Liu et al., 2024). In offline settings, it reduces *computational* or *query complexity* while maintaining guarantees (Kraska et al., 2018; Lykouris and Vassilvitskii, 2021; Balcan et al., 2021; Chen et al., 2022). We adapt this framework to causal discovery by using expert predictions to optimize statistical test sequences, primarily targeting improved estimation accuracy rather than computational benefits. While learning-augmented methods have been applied to causal intervention design (Choo et al., 2023), ours is the first adaptation to purely observational causal discovery.

3 PROBLEM SETUP AND GUESS2GRAPH

Unless otherwise mentioned, we denote random variables by lowercase letters and sets of variables by uppercase letters. A *directed acyclic graph* (DAG) $\mathcal{G} = (V, E)$ consists of nodes V and edges E. We use $e_{i,j}$ to denote the *directed* edge from x_i to x_j and $n_{i,j}$ to denote the *undirected* edge between x_i and x_j, regardless of whether these edges exist in $\mathcal{G}$. The model is defined by structural equations: for $x_i \in V$, $x_i = f(\mathrm{Pa}(x_i), \varepsilon_i)$, with jointly independent noise terms $\boldsymbol{\varepsilon}$.

The skeleton $\mathcal{S}$ of $\mathcal{G}$ is its undirected version. For any partial skeleton $\mathcal{C}$, let $\mathrm{adj}(\mathcal{C}, x_i)$ be the adjacency of x_i in $\mathcal{C}$, and $\mathrm{adj}_{-j}(\mathcal{C}, x_i) = \mathrm{adj}(\mathcal{C}, x_i) \setminus \{x_j\}$ be the adjacency set excluding x_j. Let $[A]_k$ denote all size-k subsets of set A, and $[A]_{i:k} = \bigcup_{j=i}^{k} [A]_j$. A *conditional independence test* (CIT), $\mathrm{CIT}(x, y|Z)$, tests the null hypothesis that $x \perp\!\!\!\perp y|Z$. An edge ordering O is a sequence of undirected edges, while a subset list L is a sequence of variable subsets. Finally, a domain expert (or LLM) is modeled as a predictor ψ that, given a set of variables V, outputs a prediction $\widehat{\mathcal{G}}$, while a causal discovery algorithm outputs a prediction $\widetilde{\mathcal{G}}$.

3.1 Problem Statement and Design Criteria

We consider the problem of causal discovery from a finite-sample dataset $\mathcal{X}$, generated by some underlying causal system. Under the standard assumptions (Markov condition, acyclicity, faithfulness, and causal sufficiency, Def.s B.1-B.4), there exists a true completed partially DAG (CPDAG) $\mathcal{P}^*$ (Def B.8) that perfectly characterizes the conditional independence structure via d-separation (Def. B.5), and all direct common causes of variables in V are contained in V.

In practice, finite-sample conditional independence tests are error-prone, making exact recovery of $\mathcal{C}^*$ challenging. We address this by augmenting causal discovery with predictions from an expert ψ. While inspired by learning-augmented algorithms, our setting differs in two key aspects: 1) we face statistical (not adversarial) data, with potentially adversarial expert quality, and 2) we require no uncertainty quantification and treat experts as a black box (unlike typical learning-augmented approaches that require tunable confidence parameters). This yields three key criteria:

- **C1 Statistical Consistency:** As samples grow the recovery of the true graph is guaranteed regardless of expert quality: $\lim_{n\to\infty} \mathbb{P}[\tilde{\mathcal{P}} = \mathcal{P}^*] = 1$.
- **C2 Monotonic Improvement:** The algorithm's finite-sample performance improves monotonically with expert accuracy.
- **C3 Finite-Sample Robustness:** There exists an expert accuracy threshold such that, for finite samples, the algorithm's performance with expert guidance is not worse in expectation than without it when expert accuracy exceeds this threshold.

Criterion **C1** ensures the algorithm remains fundamentally sound even with poor experts, while **C3** ensures practical utility with sufficiently accurate experts. Criterion **C2** connects these guarantees, ensuring a smooth transition between regimes. We note that traditional frameworks for incorporating expert knowledge as hard or soft constraints violate **C1**, as

we illustrate in Appendix A.3.

3.2 Guess2Graph Framework

We now propose our Guess2Graph (G2G) framework, which enables algorithms to satisfy our three criteria by strategically incorporating expert guidance while maintaining statistical foundations. The core insight is that many causal discovery algorithms contain subroutines that perform sequences of statistical tests, often with orders sampled uniformly at random. In subroutines where any valid sequence maintains asymptotic consistency, we can replace random sampling with expert-guided ordering while preserving theoretical guarantees.

The G2G framework operates in three steps: (1) identify a subroutine of an asymptotically correct causal discovery algorithm that performs sequences of statistical tests; (2) request expert ψ to predict a causal structure $\widehat{\mathcal{G}}$; and (3) extract and use an ordering from $\widehat{\mathcal{G}}$ in place of random sampling. This approach automatically ensures statistical consistency (**C1**) by keeping all decisions grounded in test outcomes rather than expert judgments.[1] However, achieving monotonic improvement (Criterion **C2**) and finite-sample guarantees (Criterion **C3**) requires careful algorithmic design within this framework.

Although the framework is general, i.e., applicable to any discovery algorithm maintaining consistency across test sequences, extracting effective orderings requires careful analysis of each subroutine's role. We therefore focus on constraint-based methods in this paper, with extensions to score-based and FCM-based algorithms discussed in Appendices C.2 and C.3.

Next, in Section 4, we demonstrate how this framework can be applied to constraint-based algorithms by identifying common subroutines that can incorporate learning augmentation. In Section 5, we show how the framework can be applied to the PC algorithm to partially achieve **C2** and **C3**. By further modifying PC to create our gPC-Guess variant, we show that both **C2** and **C3** can be fully achieved.

4 G2G IN CONSTRAINT-BASED DISCOVERY

In this section, we instantiate the G2G framework for constraint-based methods (Spirtes, 2001). We focus specifically on skeleton discovery for three reasons: it bears the primary computational burden, suffices for many causal tasks, and improvements propagate to edge orientation since orientations derive from skeleton

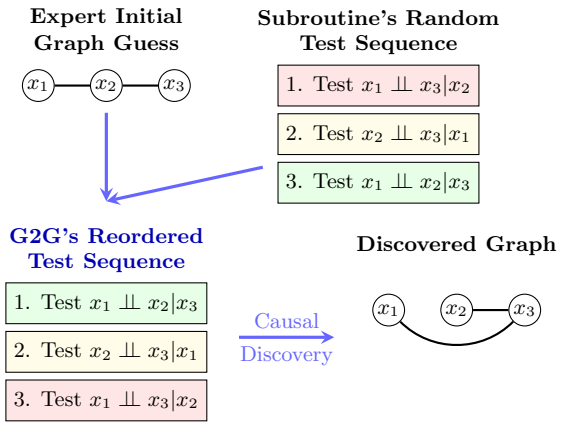


Figure 1: Guess2Graph uses expert graph predictions to reorder test sequences in causal discovery subroutines. Example above for constraint-based discovery.

tests.[2] We observe that skeleton discovery decomposes into two core subroutines that rely on uniformly sampled orderings: *Edge Prune* (EP) and *Edge Loop* (EL). The EP subroutine (Subroutine 2) tests edge $e_{i,j}$ with conditioning sets of size k following subset ordering L. The EL subroutine (Subroutine 1) sequences edge testing following edge ordering O while iterating through conditioning set sizes, from $k_{\min}$ to $k_{\max}$, calling EP for each k. For example, in the PC algorithm, $k_{\min}$ and $k_{\max}$ are both set to ℓ in iteration ℓ. In PC, an edge $e_{i,j}$ is considered valid in iteration ℓ if $n_{i,j}$ remains in the current skeleton $\mathcal{C}$ and the adjacency set of x_i (excluding x_j) is sufficiently large to test conditioning sets of size ℓ (i.e., $|\text{adj}_{-j}(\mathcal{C}, x_i)| \geq \ell$). PC calls the EL subroutine up to $|V|$ times. Different constraint-based algorithms vary in their invocation of these subroutines and validity rules R. See Appendix C.4 for further decomposition details.

Under oracle conditional independence tests, EL and EP produce identical results regardless of orderings O, L when starting from a complete graph (Lemmas D.1, D.2). However, with finite-sample errors, orderings critically impact performance (Colombo and Maathuis, 2014): edge removals update the skeleton, changing adjacency sets for subsequent tests. While PC-Stable (Colombo and Maathuis, 2014) addresses this by batching edge removals, we leverage this order-dependence for guidance. We develop modified subroutines that use expert predictions to generate O, L by prioritizing edges and subsets the expert considers relevant, rather than using uniform random orderings.

[1] While we focus on deterministic predictions, G2G can be extended with expert selection/validation (App. C.1).

[2] Accordingly, we update criterion C1 to change discovery goal to the true underlying skeleton $\mathcal{S}^*$, rather than the true CPDAG $\mathcal{P}^*$.

Subroutine 1 Edge Loop (EL)

Input: Current skeleton $\mathcal{C}$, edge ordering O, conditioning set sizes $[k_{\min}, \dots, k_{\max}]$, subset ordering L, validity rule R, EP subroutine
for $k = k_{\min}$ to $k_{\max}$ **do**
  for each undirected edge $n_{i,j}$ in order O **do**
    if $R(\mathcal{C}, e_{i,j}, k)$ **then**
      $\mathcal{C} \leftarrow \mathrm{EP}(\mathcal{C}, e_{i,j}, k, \mathsf{L})$
    end if
  end for
end for
return $\mathcal{C}$

Subroutine 2 Edge Prune (EP)

Inputs: Current skeleton $\mathcal{C}$, directed edge $e_{i,j}$, conditioning set size k, subset ordering L
Let $A \leftarrow \mathrm{adj}_{-j}(\mathcal{C}, x_i)$
for each subset $W \in [A]_k$ in order L **do**
  if $\mathrm{CIT}(x_i, x_j \mid W)$ returns independent **then**
    Remove $n_{i,j}$ from $\mathcal{C}$
    return $\mathcal{C}$
  end if
end for
return $\mathcal{C}$

4.1 A Tractable Metric for Analyzing Ordering Effects

To analyze how orderings affect algorithm performance, we require a metric that captures correctness while remaining analytically tractable. We evaluate algorithm performance by the probability of *perfect recovery* of the true skeleton $\mathcal{S}^*$. For any candidate skeleton $\mathcal{C}$ produced by the algorithm, perfect recovery occurs when $\mathcal{C}$ and $\mathcal{S}^*$ are identical on all edges. For each edge $n_{i,j}$ (undirected, or directed with $e_{i,j}$), we define the correctness indicator:

$$Y_{n_{i,j}} = \mathbb{1}\{n_{i,j} \in \mathcal{C} \wedge n_{i,j} \in \mathcal{S}^*\} + \mathbb{1}\{n_{i,j} \notin \mathcal{C} \wedge n_{i,j} \notin \mathcal{S}^*\}.$$

This indicator equals 1 when the edge $n_{i,j}$'s status in $\mathcal{C}$ matches the ground truth in $\mathcal{S}^*$, and 0 otherwise. The perfect recovery probability is then defined as:

$$\Phi = \mathbb{P}\left[\prod_{n_{i,j}: i \neq j} Y_{n_{i,j}} = 1\right],$$

representing the probability that all edges are correctly specified.

This metric has two key analytical advantages. First, the perfect recovery probability factors into a product of conditional probabilities along the edge ordering O used by the algorithm. Abusing the notation, if edges are processed in order $n_1, n_2, \dots, n_m$, then:

$$\begin{aligned}\Phi =& \mathbb{P}(Y_{n_1} = 1) \cdot \mathbb{P}(Y_{n_2} = 1 | Y_{n_1} = 1) \cdots \\ & \cdot \mathbb{P}(Y_{n_m} = 1 | Y_{n_1} = 1, \cdots, Y_{n_{m-1}} = 1). \end{aligned} \quad (1)$$

This factorization enables compositional analysis by studying the algorithm's behavior sequentially. Second, perfect recovery avoids error propagation entirely. Error propagation is difficult to analyze because false positive and false negative errors have opposing effects on adjacency sets—false positives inflate them while false negatives shrink them. This makes the overall impact of different errors dependent on the underlying graph structure, rendering the analysis of error-tolerant metrics (e.g., $\mathbb{E}[\sum Y_{n_{i,j}}]$) challenging without graph-specific assumptions (Appendix C.5).

4.2 Guiding Edge Loop

We now apply this metric to develop a principled approach for guiding EL (Subroutine 1). We start by characterizing how edge orderings affect the perfect recovery probability. This enables us to identify ordering modifications that provably improve this metric for any underlying graphical structure. We demonstrate that leveraging an expert to guide these modifications leads to monotonic improvement with expert accuracy.

Graph-independent Ordering Principles. Building on our perfect recovery metric Φ, we investigate ordering principles that improve Φ regardless of the underlying graph structure. Specifically, we analyze when correctly specifying edge $n_{i,j}$ first increases the probability of correctly specifying edge $n_{g,h}$ second—i.e., when $\mathbb{P}(Y_{n_{g,h}} = 1 \mid Y_{n_{i,j}} = 1) > \mathbb{P}(Y_{n_{g,h}} = 1)$.

The key insight is an asymmetry in how false versus true edge decisions affect subsequent tests: correctly removing false edges reduces adjacency sets (simplifying future tests), while correctly retaining true edges leaves adjacency sets unchanged. Combined with the fact that true edges become easier to retain with smaller adjacency sets (Lemma D.3), this implies that removing false edges first can only increase the probability of correctly retaining subsequent true edges, while retaining a true edge first does not affect the success probability of removing a false edge (Lemma D.4).

By placing false edges before true edges in O, we can only increase the perfect recovery probability Φ, which we formalize in Lemma D.5: for any ordering O, swapping adjacent edges to place a false edge before a true edge is never worse and sometimes strictly better.

Expert-guided Algorithm with Monotonicity Guarantees. Based on Lemma D.5, we modify the EL subroutine to incorporate expert predictions as shown in Subroutine 3. The algorithm operates as follows: given an expert graph $\widehat{\mathcal{G}}$, it extracts the skeleton

Subroutine 3 Edge Loop Guess (EL-G)

1: **Inputs**: Current skeleton $\mathcal{C}$, expert graph $\widehat{\mathcal{G}}$, conditioning set sizes $[k_{\min}, \ldots, k_{\max}]$, subset ordering L, validity rule R, EP subroutine
2: Extract skeleton $\widehat{\mathcal{S}}$ from $\widehat{\mathcal{G}}$. Randomly order $\mathcal{C}$, then set $\mathsf{O} = \mathcal{C} \setminus \widehat{\mathcal{S}} + \mathcal{C} \cap \widehat{\mathcal{S}}$
3: **return** EL($\mathcal{C}$, O, $[k_{\min}, \ldots, k_{\max}]$, L, R, EP)

Subroutine 4 Edge Prune Guess (EP-G)

1: **Inputs**: Current skeleton $\mathcal{C}$, directed edge $e_{i,j}$, conditioning set size k, expert graph $\widehat{\mathcal{G}}$
2: Let $A \leftarrow \text{adj}_{-j}(\mathcal{C}, x_i)$
3: Extract all d-sep. sets $\widehat{\mathcal{D}}_{ij}$ from $\widehat{\mathcal{G}}$. Randomly order $[A]_k$, then set $\mathsf{L} = [A]_k \cap \widehat{\mathcal{D}}_{ij} + [A]_k \setminus \widehat{\mathcal{D}}_{ij}$
4: **return** EP($\mathcal{C}$, $e_{i,j}$, k, L)

$\widehat{\mathcal{S}}$ and partitions the current skeleton $\mathcal{C}$ into edges the expert believes are false ($\mathcal{C} \setminus \widehat{\mathcal{S}}$) and true ($\mathcal{C} \cap \widehat{\mathcal{S}}$), then processes the false edges before the true edges while maintaining random order within each group.

We make the assumption that the expert ψ acts as a symmetric binary channel: for any edge $n_{i,j}$, the expert independently predicts if it exists in the true skeleton $\mathcal{S}^*$ with accuracy p_ψ. Under this assumption, we establish the following monotonicity guarantee:

Lemma 4.1 (Monotonicity of Perfect Recovery in Expert Accuracy)**.** *For a fixed partial skeleton $\mathcal{C}$ and true DAG $\mathcal{G}^*$, let $\Phi_{EL\text{-}G}(p_\psi)$ denote the perfect recovery probability when we sample an expert graph $\widehat{\mathcal{G}}$ from expert ψ with accuracy p_ψ, draw n samples from $\mathcal{G}^*$, and run EL-G. Then $\mathbb{E}[\Phi_{EL\text{-}G}(p_\psi)]$ increases monotonically with p_ψ, strictly increasing when $\mathcal{C}$ contains false edges adjacent to true edges.*

Proof sketch. The proof (App. D.6) establishes monotonicity via a coupling argument between experts with accuracies $p_{\psi_1} < p_{\psi_2}$. Both experts observe the same true skeleton $\mathcal{S}^*$ and use identical randomness for edge classification, but the higher-accuracy expert makes fewer errors. This ensures that every edge correctly classified by the weaker expert is also correctly classified by the stronger expert. Consequently, the better expert's edge ordering has fewer "inversions" (true edges incorrectly placed before false edges). These orderings are related by the weak Bruhat order, meaning the better ordering can be obtained through a sequence of adjacent swaps that move false edges leftward past true edges. By Lemma D.5, each such swap weakly improves the perfect recovery probability $\Phi_{\text{EL-G}}$, with strict improvement when swapped edges share vertices (since removing false edges first shrinks adjacency sets and reduces false negative probabilities). Since the better expert's ordering is reachable through beneficial swaps, it achieves pointwise improvement for any fixed realization of data and expert predictions. Strassen's Coupling theorem then implies that $\Phi_{\text{EL-G}}(p_{\psi_2})$ stochastically dominates $\Phi_{\text{EL-G}}(p_{\psi_1})$, yielding the monotonicity in expectation.

We note that our assumption that the expert ψ acts as a symmetric binary channel with independent errors can be weakened. In Appendix D.7 we show a monotonicity result similar to Lemma 4.1 holds even when the expert has different edge prediction accuracies for false edges and true edges, rather than a single accuracy p_ψ. This captures the discrepancy between recall and precision (with respect to identifying false edges) that typically exists for real-world experts.

4.3 Guiding Edge Prune

We now develop a principled approach to guiding EP (Subroutine 2). Unlike EL, where ordering affects accuracy, EP's ordering only impacts runtime, which can also be decreased by leveraging expert predictions. Since our focus is on accuracy improvements through expert guidance, we briefly outline how expert predictions can accelerate EP by prioritizing promising conditioning sets, and defer the complete theoretical analysis of runtime to Appendix E.

The EP subroutine requires an ordering L to sequence conditional independence tests for edge $e_{i,j}$. Given adjacency set $\text{adj}_{-j}(\mathcal{C}, x_i)$ and conditioning set size k, let $\text{CIT}^k_{\text{adj}_{-j}(\mathcal{C},x_i)} := \{\text{CIT}(x_i, x_j \mid W) : W \subseteq \text{adj}_{-j}(\mathcal{C}, x_i), |W| = k\}$ denote all possible independence tests of size k. Since EP removes an edge only if any test in $\text{CIT}^k_{\text{adj}_{-j}(\mathcal{C},x_i)}$ returns independence, and the timing of test execution does not affect test outcomes, the edge recovery accuracy $\mathbb{P}(Y_{e_{i,j}} = 1)$ remains constant across all orderings (Lemma D.7).

Although EP orderings cannot affect accuracy, they can impact computational runtime (Lemma D.8). Orderings that place d-separating sets of x_i, x_j earlier achieve lower expected runtimes (Lemma D.9). We therefore guide EP by prioritizing conditioning sets predicted to d-separate x_i, x_j according to the expert's graph $\widehat{\mathcal{G}}$, as formalized in Subroutine 4.

We make similar assumptions in d-separation prediction analogously to edge prediction: the expert acts as a binary symmetric channel with accuracy $p_{\text{d-sep}}$ for identifying d-separating sets. Under a common technical condition (Definition E.1) from testing literature (Brown and Tsamardinos, 2008; Li and Wang, 2009; Strobl et al., 2019), a coupling argument similar to Lemma 4.1 shows that EP-Guess's expected runtime decreases monotonically with $p_{\text{d-sep}}$ (Lemma D.10).

Algorithm 5 PC-Guess

Inputs: Expert ψ, complete skeleton $\mathcal{C}$
$\mathsf{O}, \mathsf{L} \leftarrow$ Subroutine F.1(ψ, $\mathcal{C}$)
Define $R^{\ell}_{PC}(\mathcal{C}, e_{i,j}) = n_{i,j} \in \mathcal{C} \wedge |\text{adj}_{-j}(\mathcal{C}, x_i)| \geq \ell$
for $\ell = 0$ to $d-1$ **do**
  $\mathcal{C} \leftarrow$ EL($\mathcal{C}$, O, $[\ell, \ell]$, L, R^{ℓ}_{PC}, EP)
  if no edges satisfy R^{ℓ}_{PC} **then break**
end for
return $\mathcal{C}$

Algorithm 6 Guided PC (gPC-Guess)

Inputs: Expert ψ, complete skeleton $\mathcal{C}$
$\mathsf{O}, \mathsf{L} \leftarrow$ Subroutine F.1(ψ, $\mathcal{C}$)
Define $R_{gPC}(\mathcal{C}, e_{i,j}) = n_{i,j} \in \mathcal{C}$
$\mathcal{C} \leftarrow$ EL($\mathcal{C}$, O, $[0, |V|-1]$, L, R_{gPC}, EP)
return $\mathcal{C}$

5 EXPERT AUGMENTED ALGORITHMS

We introduce PC-Guess (Alg. 5, Section 5.1) and gPC-Guess (Alg. 6, Section 5.2), which implement the expert-guided framework from Section 4 using a unified extraction subroutine (Subroutine F.1) that generates both orderings O and L from $\widehat{\mathcal{G}}$. We provide theoretical guarantees in Section 5.3.

5.1 PC-Guess

We instantiate the G2G framework as PC-Guess (Alg. 5), which modifies PC's skeleton discovery by integrating expert guidance through orderings O and L from Subroutine F.1. The algorithm then iterates through conditioning set sizes ℓ, at each stage calling EL with validity rule R^{ℓ}_{PC} that determines whether an edge qualifies for testing by checking two conditions: (1) the edge remains in the current skeleton $\mathcal{C}$, and (2) at least one endpoint's adjacency set (excluding the other endpoint) has size at least ℓ to enable conditioning sets of size ℓ.

PC-Guess inherits PC's iterative structure for processing conditioning sets, which constrains how expert predictions are utilized. Due to the curse of dimensionality (Li et al., 2020), conditional independence tests become less reliable with larger conditioning sets. PC addresses this through a "statistical conditioning" bias that prioritizes smaller conditioning sets first—a common design pattern in causal discovery algorithms.

While this approach maximizes test reliability without expert knowledge, it limits the potential benefit of expert predictions: false edges requiring larger conditioning sets for removal cannot be eliminated early, forcing unnecessary tests at lower conditioning levels and inflating adjacency sets for neighboring edges (Appendix G). This limitation motivates removing the level-by-level constraint to enable earlier removal of false edges with non-trivial minimal d-separating sets.

5.2 gPC-Guess

To address the limitations of PC's level-by-level constraint, we propose gPC-Guess (Alg. 6), which enables immediate action on expert predictions. Like PC-Guess, gPC-Guess extracts orderings O and L from $\widehat{\mathcal{G}}$ via Subroutine F.1, but replaces PC's iterative structure with a single-pass approach that tests all edges using conditioning sets from size 0 to $|V|-1$.

This design eliminates the statistical conditioning bias in favor of expert responsiveness: gPC-Guess uses a simplified validity rule R_{gPC} that only checks edge presence, allowing false edges with non-trivial minimal d-separating sets to be removed immediately when placed early in O. While this can reduce adjacency set inflation and improve accuracy with good expert guidance, it risks testing edges with unnecessarily large conditioning sets when expert predictions are inaccurate, potentially compromising test reliability.

5.3 Theoretical Guarantees

Correctness and Statistical Consistency (C1). Both PC-Guess and gPC-Guess maintain the theoretical guarantees of their base algorithms regardless of expert quality, converging to the true skeleton with consistent conditional independence tests:

Theorem 5.1 (Asymptotic Correctness)**.** *Under a consistent conditional independence test, for both PC-Guess and gPC-Guess,* $\lim_{n\to\infty} \mathbb{P}(\widetilde{\mathcal{S}} = \mathcal{S}^*) = 1$.

The proof (Appendix D.12) follows from both algorithms eventually considering all possible edges and CITs regardless of ordering. This ensures asymptotic performance is never compromised by potentially poor expert guidance, satisfying **C1**.

Monotonic Improvement and Finite-Sample Robustness (C2, C3). We characterize how expert quality affects finite-sample performance, with per-iteration guarantees for PC-Guess and end-to-end guarantees for gPC-Guess. Consider a fixed DAG $\mathcal{G}^*$ with d variables and expert ψ with edge accuracy p_ψ and d-separation accuracy $p_{\text{d-sep}}$, drawing n samples from $\mathcal{G}^*$ and prediction $\widehat{\mathcal{G}}$ from ψ. We compare guided variants (PC-Guess, gPC-Guess) against unguided baselines (PC, gPC) denoted with overbars $\bar{(\cdot)}$.

For PC-Guess, fix iteration $\ell \in \{0, 1, \ldots, d-1\}$ and suppose both algorithms start with an identical partial skeleton $\mathcal{C}$. Let Φ_ℓ and $\bar{\Phi}_\ell$ denote perfect recovery probabilities in iteration ℓ for PC-Guess and PC, re-

spectively. Let t_ℓ denote the number of tests run in PC-Guess.

Theorem 5.2 (Performance of PC-Guess)**.** *PC-Guess satisfies **C2-C3** at per-iteration level: **(a)** $\mathbb{E}[\Phi_\ell]$ increases monotonically with p_ψ; **(b)** For fixed p_ψ, $\mathbb{E}[t_\ell]$ decreases monotonically with $p_{d\text{-}sep}$; **(c)** When $p_\psi \geq 0.5$, $\mathbb{E}[\Phi_\ell] \geq \mathbb{E}[\bar{\Phi}_\ell]$.*

Let Φ and $\bar{\Phi}$ denote perfect skeleton recovery probabilities for gPC-Guess and gPC, respectively, and let t denote the total number of tests run in gPC-Guess.

Theorem 5.3 (Performance of gPC-Guess)**.** *gPC-Guess satisfies **C2-C3**: **(a)** $\mathbb{E}[\Phi]$ increases monotonically with p_ψ; **(b)** For fixed p_ψ, $\mathbb{E}[t]$ decreases monotonically with $p_{d\text{-}sep}$; **(c)** When $p_\psi \geq 0.5$, $\mathbb{E}[\Phi] \geq \mathbb{E}[\bar{\Phi}]$.*

Proofs appear in App. D.13 and D.14, following directly from Lemmas 4.1 and D.10.

Remark 5.4. *All montonicity relations and inequalities are strict for nonempty and non-fully connected graphs. The key distinction lies in guarantee scope: while PC-Guess achieves improvements per iteration, these may not compose into end-to-end guarantees due to complex cascading effects across iterations. For example, correctly removing false edges early helps retain true edges later but may make it harder to remove persistent false edges. In contrast, gPC-Guess provides end-to-end guarantees, fully satisfying criteria **C1-C3** for the final output. This reflects the fundamental tradeoff between PC-Guess's statistical conditioning bias and gPC-Guess's expert responsiveness.*

6 EXPERIMENTS

We evaluate how our algorithms satisfy criteria **C1-C3** through experiments with synthetic and real-world data.[3] Our results validate that gPC-Guess fully achieves **C1-C3**, while PC-Guess empirically exceeds its guarantees. LLM-boosted gPC-Guess outperforms both the LLM alone and data-driven methods.

Datasets and Experts. We evaluate on synthetic data (linear Gaussian models on Erdos-Renyi graphs, Erdos and Renyi 1960) and real-world benchmarks. Experiments in the main text focus on sparse ER graphs ($d = 20$ variables, $n = 100$ samples) and subsampled Sachs protein data (Sachs et al., 2005) ($d = 11, n = 100$). We test two expert types: (1) simulated experts ψ with manually-tuned prediction accuracy (we focus on better than random edge prediction, i.e., $p_\psi \geq 0.5$ to validate **C2**, and hold constant d-separating set prediction accuracy as entirely random, i.e., $p_{\text{d-sep}} = 0.5$), and (2) a LLM expert, specifically Claude Opus 4.1 (Anthropic, 2025). Appendix H provides full simulation parameters, information about real data, and details on how predictions are generated (for both simulated and LLM experts). We explore additional experiments in App. I: varying sample size to validate **C1** (statistical consistency), dimensionality, d-separating accuracy ($p_{\text{d-sep}}$), and worst-case performance with experts below the **C3** threshold ($p_\psi \leq 0.5$).

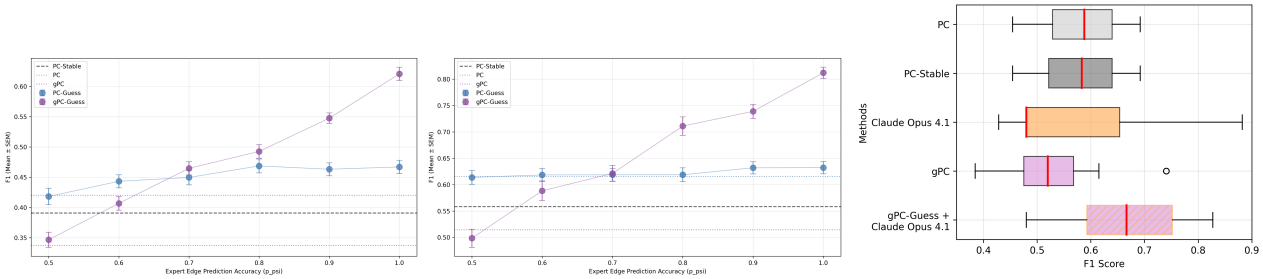
Methods and Metrics. In the main text we compare PC-Guess and gPC-Guess against the order-independent baseline PC-Stable (Ramsey et al., 2006). We include PC and gPC as baseline versions of PC-Guess and gPC-Guess where the edge predictions supplied to both methods are uniformly sampled, i.e. $p_\psi = 0.5$. In Appendix J we compare against hard constraint approaches as well as several guidance-based score algorithms (Ejaz and Bareinboim, 2025). All methods use identical CI tests (see Appendix H.6), with $\alpha = 0.05$. Performance is measured using skeleton F1 scores (see Appendix I.1 for runtime).

Simulated Expert Results. Figures 2a and 2b demonstrate how augmented algorithm performance changes as synthetic experts provide increasingly accurate edge predictions on synthetic and real-world datasets. Figure 2a shows that on synthetic data, both PC-Guess and gPC-Guess increase monotonically in F1 score with expert accuracy (verifying **C2**). Despite PC-Guess only having theoretical guarantees for per-iteration improvement, it empirically achieves monotonic improvement that is no worse than baseline for $p_\psi \geq 0.5$ (empirically satisfying **C3**). As predicted by theory, gPC-Guess benefits more from guidance, achieving the highest accuracy when expert quality is sufficient ($p_\psi \geq 0.7$). These results replicate in Figure 2b on the Sachs real-world dataset: we again observe monotonic gains in both algorithms, but PC-Guess remains relatively flat while gPC-Guess's F1 increases by over 30 percentage points, confirming that algorithmic redesign is necessary to fully realize **C2**.

LLM Expert Results. Figure 2c explores how DAG guesses from real-world expert Claude Opus 4.1 benefit our augmented algorithms on the Sachs dataset. gPC-Guess achieves a 15% performance boost when combined with Claude's predictions, outperforming the baselines by roughly 10 percentage points. This demonstrates that our framework extends beyond theory and has potential for combination with existing LLM experts in real-world applications.

Experiments in Additional Settings. Appendix I presents results across varying sparsity, sample sizes, dimensionality, and d-separation accuracy. We report the key findings: both algorithms retain monotonic

[3] github.com/amazon-science/Guess2GraphPaperCode



(a) ER3 data with simulated expert. (b) Sachs data with simulated expert. (c) Sachs data with LLM expert.

Figure 2: Performance improvement of PC-Guess and gPC-Guess with expert guidance.

improvement with expert accuracy in sparse graphs, though gains are reduced (Appendix I.2). Performance gains from expert predictions diminish with larger samples as all methods converge to the true graph, confirming **C1** (Appendix I.3). Expert guidance value increases with dimensionality, with greater improvements in high-dimensional, low-sample settings (Appendix I.4). Below the **C3** threshold ($p_\psi \leq 0.5$), performance drops $\sim 8\%$ in F1, but this penalty remains modest due to correctness guarantees (Appendix I.5).

Experiments with Additional Baselines. Appendix J presents results with additional baselines. We report the key findings: PC-Guess outperforms its hard constraint comparator for strong experts ($p_{acc} > 0.7$), while gPC-Guess outperforms its hard constraint baseline across all expert accuracies tested (Appendix J.1). When hard-constraints are stacked on top of our Guess2Graph approach (Appendix J.2), the corresponding modified versions of PC-Guess and gPC-Guess outperform the hard constraint only baselines across all expert accuracies tested. In Appendix J.3 we compare against three score-based methods (GES (Chickering, 2002), LGES Safe, and LGES Cons (Ejaz and Bareinboim, 2025)), each of which is augmented with two heuristic guidance approaches ('initialization' and 'priority'). We find that both PC-Guess and gPC-Guess outperform the score-based approaches across all expert accuracies.

Discussion. We introduced G2G and applied it to constraint-based discovery to develop PC-Guess and gPC-Guess, which provably leverage unreliable expert predictions with robust guarantees. Future work includes extensions to hybrid score/FCM-based algorithms, as well as continuous optimization approaches.

References

Bryon Agaram. Greedy equivalence search for nonparametric graphical models. *https://arxiv.org/abs/2406.17228*, 2024.

Ankur Ankan and Johannes Textor. Expert-in-the-loop causal discovery: Iterative model refinement using expert knowledge. In *Proceedings of the 41st Conference on Uncertainty in Artificial Intelligence*, UAI 2025, Rio de Janeiro, Brazil, July 2025.

Anthropic. System card: Claude opus 4 & claude sonnet 4. Technical report, Anthropic, May 2025.

Maria-Florina Balcan, Travis Dick, and Colin Manuel. Learning to link. In *International Conference on Machine Learning (ICML)*, 2021.

Taiyu Ban, Lyvzhou Chen, Xiangyu Wang, and Huanhuan Chen. From query tools to causal architects: Harnessing large language models for advanced causal discovery from data.

Taiyu Ban, Lyuzhou Chen, Derui Lyu, Xiangyu Wang, and Huanhuan Chen. Causal structure learning supervised by large language model. 2023.

Philippe Brouillard, Chandler Squires, Jonas Wahl, Konrad Körding, Karen Sachs, Alexandre Drouin, and Dhanya Sridhar. The landscape of causal discovery data: Grounding causal discovery in real-world applications. In *Proceedings of the Fourth Conference on Causal Learning and Reasoning*, volume 275 of *Proceedings of Machine Learning Research*, pages 834–873. PMLR, 2025.

Laura E. Brown and Ioannis Tsamardinos. A strategy for making predictions under manipulation. In *Proceedings of the Workshop on the Causation and Prediction Challenge at WCCI 2008*, volume 3 of *PMLR*, pages 35–52, 2008.

Sandy Campbell and Don A. Moore. Overprecision in the survey of professional forecasters. *Collabra: Psychology*, 10(1):92953, 2024.

Justin Y. Chen, Sandeep Silwal, Ali Vakilian, and Fred Zhang. Faster fundamental graph algorithms via learned predictions. In *International Conference on Machine Learning (ICML)*, 2022.

David Maxwell Chickering. Optimal structure identification with greedy search. *https://jmlr.org/papers/v3/chickering02b.html*, 2002.

Max Chickering. Statistically efficient greedy equivalence search. *https://proceedings.mlr.press/v124/chickering20a.html*, 2020.

Davin Choo, Themistoklis Gouleakis, and Arnab Bhattacharyya. Active causal structure learning with advice. In *Proceedings of the 40th International Conference on Machine Learning*, Proceedings of Machine Learning Research. PMLR, 2023.

Tom Claassen and Ioan Gabriel Bucur. Greedy Equivalence Search in the Presence of Latent Confounders. In *Proceedings of the 38th Conference on Uncertainty in Artificial Intelligence*, volume 180. PMLR, 2022.

Kai-Hendrik Cohrs, Gherardo Varando, Emiliano Diaz, Vasileios Sitokonstantinou, and Gustau Camps-Valls. Large language models for constrained-based causal discovery, 2024.

Diego Colombo and Marloes H. Maathuis. Order-independent constraint-based causal structure learning. *Journal of Machine Learning Research*, 2014.

A.C. Constantinou, Z. Guo, and N.K. Kitson. The Impact of Prior Knowledge on Causal Structure Learning. *Knowledge and Information Systems*, 65:3385–3434, 2023. doi: 10.1007/s10115-023-01858-x.

Gregory F. Cooper and Changwon Yoo. Causal discovery from a mixture of experimental and observational data. In *Proceedings of the Conference on Uncertainty in Artificial Intelligence*, pages 116–125, 1999.

Victor-Alexandru Darvariu, Stephen Hailes, and Mirco Musolesi. Large language models are effective priors for causal graph discovery, 2024.

Sanjeeb Dash, Joao Goncalves, and Tian Gao. Integer programming based methods and heuristics for causal graph learning. *https://proceedings.mlr.press/v258/dash25a.html*, 2025.

Chang Deng, Kevin Bello, Pradeep Ravikumar, and Bryon Aragam. Markov equivalence and consistency in differentiable structure learning. *https://arxiv.org/abs/2410.06163*, 2024.

Shuyu Dong, Michele Sebag, Kento Uemura, Akito Fujii, Shuang Chang, Yusuke Koyanagi, and Koji Maruhashi. Dcilp: A distributed approach for large-scale causal structure learning. *https://arxiv.org/abs/2406.10481*, 2025.

Yihong Dong, Xue Jiang, Huanyu Liu, Zhi Jin, Bin Gu, Mengfei Yang, and Ge Li. Generalization or memorization: Data contamination and trustworthy evaluation for large language models. Association for Computational Linguistics, 2024.

Itiel E. Dror. Cognitive and human factors in expert decision making: Six fallacies and the eight sources of bias. *Analytical Chemistry*, 92(12):7998–8004, 2020. doi: 10.1021/acs.analchem.0c00704.

Adiba Ejaz and Elias Bareinboim. Less greedy equivalence search. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2025.

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

Elias Eulig, Atalanti A. Mastakouri, Patrick Blöbaum, Moritz Hardt, and Dominik Janzing. Toward falsifying causal graphs using a permutation-based test. In *Proceedings of the AAAI Conference on Artificial Intelligence*, 2025.

Philipp M. Faller, Leena Chennuru Vankadara, Atalanti A. Mastakouri, Francesco Locatello, and Dominik Janzing. Self-compatibility: Evaluating causal discovery without ground truth. AISTATS, 2024.

Tao Feng, Lizhen Qu, Niket Tandon, Zhuang Li, Xiaoxi Kang, and Gholamreza Haffari. On the reliability of large language models for causal discovery. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics*, 2025.

Ronald A. Fisher. On the probable error of a correlation coefficient deduced from a small sample. *Metron*, 1:3–32, 1921.

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

Uzma Hasan and Md Osman Gani. Optimizing Data-driven Causal Discovery Using Knowledge-guided Search, 2024.

Sujai Hiremath, Jacqueline Maasch, Mengxiao Gao, Promit Ghosal, and Kyra Gan. Hybrid top-down global causal discovery with local search for linear and nonlinear additive noise models. In *Proceedings of the 37th Conference on Neural Information Processing Systems (NeurIPS)*, 2024.

Sujai Hiremath, Promit Ghosal, and Kyra Gan. Losam: Local search in additive noise models with mixed mechanisms and general noise for global causal discovery. In *Proceedings of the 38th Conference on Neural Information Processing Systems (NeurIPS)*, 2025.

Biwei Huang, Kun Zhang, Yizhu Lin, Bernhard Schölkopf, and Clark Glymour. Generalized Score Functions for Causal Discovery. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pages 1551–1560, London United Kingdom, July 2018. ACM.

Antti Hyttinen, Frederick Eberhardt, and Matti Järvisalo. Constraint-based causal discovery: Conflict resolution with answer set programming. *https://www.cs.helsinki.fi/u/mjarvisa/papers/hyttinen-eberhardt-jarvisalo.uai14.pdf*, 2014.

Billy Jin and Will Ma. Online bipartite matching with advice: Tight robustness-consistency tradeoffs for the two-stage model. *Advances in Neural Information Processing Systems*, 35:14555–14567, 2022.

Thomas Jiralerspong, Xiaoyin Chen, Yash More, Vedant Shah, and Yoshua Bengio. Efficient causal graph discovery using large language models. *arXiv preprint arXiv:2402.01207*, 2024.

Tim Kraska, Alex Beutel, Ed H. Chi, Jeffrey Dean, and Neoklis Polyzotis. The case for learned index structures. In *Proceedings of the 2018 International Conference on Management of Data (SIGMOD)*, pages 489–504, 2018.

Emre Kıcıman, Robert Ness, Amit Sharma, and Chenhao Tan. Causal reasoning and large language models: Opening a new frontier for causality. *Transactions on Machine Learning Research (TMLR)*, 2024.

Kenneth Lee, Bruno Ribeiro, and Murat Kocaoglu. Constraint-based causal discovery from a collection of conditioning sets. In *Proceedings of the 41st Conference on Uncertainty in Artificial Intelligence (UAI)*, 2025.

David A Levin and Yuval Peres. Coupling. In *Modern Discrete Probability: An Essential Toolkit*, chapter 4. 2023. Version: December 20, 2023.

Junning Li and Z. Jane Wang. Controlling the false discovery rate of the association/causality structure learned with the pc algorithm. *Journal of Machine Learning Research*, 10(17):475–514, 2009.

Runze Li, Wei Shen, and Haibo Zhou. On nonparametric conditional independence tests for continuous variables. *WIREs Computational Statistics*, 12(2): e1489, 2020.

Torgny Lindvall. On strassen's theorem on stochastic domination. *Electronic Communications in Probability*, 4:51–59, 1999. doi: 10.1214/ECP.v4-1005.

Xueqing Liu, Kyra Gan, Esmaeil Keyvanshokooh, and Susan Murphy. Online uniform sampling: Randomized learning-augmented approximation algorithms with application to digital health. *arXiv preprint arXiv:2402.01995*, 2024.

Thodoris Lykouris and Sergei Vassilvitskii. Competitive Caching with Machine Learned Advice, 2018.

Thodoris Lykouris and Sergei Vassilvitskii. Competitive caching with machine learned advice. In *Journal of the ACM*, volume 68, 2021.

Jacqueline Maasch, Weishen Pan, Shantanu Gupta, Volodymyr Kuleshov, Kyra Gan, and Fei Wang. Local discovery by partitioning: Polynomial-time causal discovery around exposure-outcome pairs. In *Proceedings of the 40th Conference on Uncertainy in Artificial Intelligence*, 2024. doi: https://doi.org/10.48550/arXiv.2310.17816.

Putra Manggala, Atalanti Mastakouri, Elke Kirschbaum, Shiva Prasad Kasiviswanathan, and Aaditya Ramdas. Qa-calibration of language model confidence scores. In *Proceedings of the International Conference on Learning Representations*, 2025.

Michael Mitzenmacher and Sergei Vassilvitskii. Algorithms with Predictions, 2020.

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

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

Achille Nazaret and David Blei. Extremely greedy equivalence search. *UAI*, 2024.

Karl Pearson. On the criterion that a given system of deviations from the probable in the case of a correlated system of variables is such that it can be reasonably supposed to have arisen from random sampling. *Philosophical Magazine Series 5*, 50(302): 157–175, 1900. doi: 10.1080/14786440009463897.

Jonas Peters, Joris M. Mooij, Dominik Janzing, and Bernhard Schoelkopf. Causal discovery with continuous additive noise models. *https://jmlr.org/papers/v15/peters14a.html*, 2014.

Anne H Petersen, Merete Osler, and Claus T Ekstrøm. Data-driven model building for life-course epidemiology. *American Journal of Epidemiology*, 2021.

Joseph Ramsey, Peter Spirtes, and Jiji Zhang. Adjacency-faithfulness and conservative causal inference. In *Proceedings of the Twenty-Second Conference on Uncertainty in Artificial Intelligence*, pages 401–408, 2006.

Joseph Ramsey, Madelyn Glymour, Ruben Sanchez-Romero, and Clark Glymour. A million variables and more: the fast greedy equivalence search algorithm for learning high-dimensional graphical causal models, with an application to functional magnetic resonance images. *Int J Data Sci Anal.*, 2017.

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

Mauro Scanagatta, Cassio P. de Campos, and Marco Zaffalon. Min-bdeu and max-bdeu scores for learning bayesian networks. In *Probabilistic Graphical Models*, volume 8754 of *Lecture Notes in Computer Science*, pages 426–441. Springer, 2014. European Workshop on Probabilistic Graphical Models (PGM 2014).

Tomi Silander and Petri Myllymaki. A simple approach for finding the globally optimal bayesian network structure. *https://arxiv.org/abs/1206.6875*, 2006.

Arjun Sondhi and Ali Shojaie. The reduced pc-algorithm: Improved causal structure learning in large random networks. *Journal of Machine Learning Research*, 20:1–31, 2019.

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

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

Eric V. Strobl, Peter L. Spirtes, and Shyam Visweswaran. Estimating and controlling the false discovery rate of the pc algorithm using edge-specific p-values. *ACM Transactions on Intelligent Systems and Technology*, 10(5):1–37, 2019.

Zhuofan Sun and Qingyi Li. Leveraging LLMs for Causal Inference and Discovery, 2024.

Yuni Susanti and Michael Faber. Can LLMs Leverage Observational Data? Towards Data-Driven Causal Discovery with LLMs, 2025.

Masayuki Takayama, Tadahisa Okuda, Thong Pham, Tatsuyoshi Ikenoue, Shingo Fukuma, Shohei Shimizu, and Akiyoshi Sannai. Integrating large language models in causal discovery: A statistical causal approach. *Transactions on Machine Learning Research*, 2025. arXiv:2402.01454.

Peter W G Tennant, Eleanor J Murray, Kellyn F Arnold, Laurie Berrie, Matthew P Fox, Sarah C Gadd, Wendy J Harrison, Claire Keeble, Lynsie R Ranker, Johannes Textor, Georgia D Tomova, Mark S Gilthorpe, and George T H Ellison. Use of directed acyclic graphs (DAGs) to identify confounders in applied health research: review and recommendations. *International Journal of Epidemiology*, 2021.

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

Caroline Uhler, Garvesh Raskutti, Peter Bühlmann, and Bin Yu. Geometry of the faithfulness assumption in causal inference. *The Annals of Statistics*, 41 (2):436–463, 2013. doi: 10.1214/12-AOS1080.

Aniket Vashishtha, Gowtham Reddy Abbavaram, Abhinav Kumar, Saketh Bachu, Vineeth N. Balasubramanian, and Amit Sharma. Causal Order: The Key to Leveraging Imperfect Experts in Causal Inference. In *The Thirteenth International Conference on Learning Representations (ICLR)*, 2025.

Xiaoqiang Wang, Yali Du, Shengyu Zhu, Liangjun Ke, Zhitang Chen, Jianye Hao, and Jun Wang. Ordering-based causal discovery with reinforcement learning. *https://www.ijcai.org/proceedings/2021/491*, 2021.

Alexander Wei and Fred Zhang. Optimal robustness-consistency trade-offs for learning-augmented online algorithms. *Advances in Neural Information Processing Systems*, 33:8042–8053, 2020.

Xingyu Wu, Kui Yu, Jibin Wu, and Kay Chen Tan. LLM Cannot Discover Causality, and Should Be Restricted to Non-Decisional Support in Causal Discovery, 2025.

Zhiqiang Xie, Yujia Zheng, Lizi Ottens, Kun Zhang, Christos Kozyrakis, and Jonathan Mace. Cloud atlas: Efficient fault localization for cloud systems using language models and causal insight. 2024.

Fanghua Ye, Mingming Yang, Jianhui Pang, Longyue Wang, Derek F. Wong, Emine Yilmaz, Shuming Shi, and Zhaopeng Tu. Benchmarking LLMs via uncertainty quantification. 2024.

Kuat Yessenov. Descents and the weak bruhat order, 2005.

Changhe Yuan, Brandon Malone, and Xiaojian Wu. Learning optimal bayesian networks using a* search. *https://www.ijcai.org/proceedings/2011/364*, 2011.

Kun Zhang and Aapo Hyvarinen. Distinguishing Causes from Effects using Nonlinear Acyclic Causal Models. 2008.

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

Xun Zheng, Chen Dan, Bryon Aragam, Pradeep Ravikumar, and Eric P. Xing. Learning sparse nonparametric dags. *https://arxiv.org/abs/1909.13189*, 2020.

Shengyu Zhu, Ignavier Ng, and Zhitang Chen. Causal discovery with reinforcement learning. *https://openreview.net/forum?id=S1g2skStPB*, 2020.

Yuehua Zhu, Panayiotis V. Benos, and Maria Chikina. A hybrid constrained continuous optimization approach for optimal causal discovery from biological data. *Bioinformatics*, 40(Suppl 2):ii87–ii97, 2024. doi: 10.1093/bioinformatics/btae411.

Or Zuk, Shiri Margel, and Eytan Domany. On the number of samples needed to learn the correct structure of a bayesian network. In *Proceedings of the 28th Conference on Uncertainty in Artificial Intelligence (UAI)*, pages 560–569, 2012.

Checklist

1. For all models and algorithms presented, check if you include:
   (a) A clear description of the mathematical setting, assumptions, algorithm, and/or model. Yes, see Section 3.
   (b) An analysis of the properties and complexity (time, space, sample size) of any algorithm. Yes, see Appendix.
   (c) (Optional) Anonymized source code, with specification of all dependencies, including external libraries. Yes.

2. For any theoretical claim, check if you include:
   (a) Statements of the full set of assumptions of all theoretical results. Yes.
   (b) Complete proofs of all theoretical results. Yes.
   (c) Clear explanations of any assumptions. Yes.

3. For all figures and tables that present empirical results, check if you include:
   (a) The code, data, and instructions needed to reproduce the main experimental results (either in the supplemental material or as a URL). We include all details needed to reproduce our experimental results in our appendix. Code is released.
   (b) All the training details (e.g., data splits, hyperparameters, how they were chosen). Yes, see appendix.
   (c) A clear definition of the specific measure or statistics and error bars (e.g., with respect to the random seed after running experiments multiple times). Yes, see Appendix.
   (d) A description of the computing infrastructure used. (e.g., type of GPUs, internal cluster, or cloud provider). Yes, see Appendix.

4. If you are using existing assets (e.g., code, data, models) or curating/releasing new assets, check if you include:
   (a) Citations of the creator If your work uses existing assets. Yes, see Appendix.
   (b) The license information of the assets, if applicable. Yes, see Appendix.
   (c) New assets either in the supplemental material or as a URL, if applicable. Not Applicable.
   (d) Information about consent from data providers/curators. Not Applicable.
   (e) Discussion of sensible content if applicable, e.g., personally identifiable information or offensive content. Not Applicable.

5. If you used crowdsourcing or conducted research with human subjects, check if you include:
   (a) The full text of instructions given to participants and screenshots. Not Applicable.
   (b) Descriptions of potential participant risks, with links to Institutional Review Board (IRB) approvals if applicable. Not Applicable.
   (c) The estimated hourly wage paid to participants and the total amount spent on participant compensation. Not Applicable.

Appendix

A Introduction

A.1 Unbounded Error Example

To illustrate why hard constraints from unreliable experts can cause unbounded error, we provide a concrete example using the PC algorithm. Recall that PC discovers the skeleton of the underlying DAG by running conditional independence tests, removing edges between pairs of variables when there exists at least one conditioning set that renders them conditionally independent.

Example construction. Consider the true DAG on four variables $\{x_1, x_2, x_3, x_4\}$ where x_1 is a common cause: $x_1 \to x_2$, $x_1 \to x_3$, $x_1 \to x_4$. In this structure, x_1 confounds the relationships between x_2, x_3, and x_4. Now suppose an expert provides a hard constraint specifying that x_1 is an isolated node with no edges, implying $x_1 \perp\!\!\!\perp x_2$, $x_1 \perp\!\!\!\perp x_3$, $x_1 \perp\!\!\!\perp x_4$.

Cascading failure. Under this incorrect hard constraint, PC with oracle independence tests will incorrectly infer edges between all pairs in $\{x_2, x_3, x_4\}$: specifically, edges (x_2, x_3), (x_2, x_4), and (x_3, x_4). This occurs because the only conditioning set that renders these variables pairwise independent is $\{x_1\}$. However, PC's adjacency-based testing procedure means that when testing whether to remove edge (x_i, x_j), the algorithm only considers conditioning sets composed of variables adjacent to at least one of x_i or x_j in the current skeleton. Since the expert's hard constraint excludes x_1 from all adjacency sets by decree, the critical conditioning set $\{x_1\}$ is never tested, and all three spurious edges are incorrectly retained.

Unbounded error. The resulting skeleton exhibits maximum possible error: every edge in the recovered skeleton is absent from the true graph (three false positives), and every edge in the true graph is absent from the recovered skeleton (three false negatives). This demonstrates unbounded error in the sense that expert misinformation leads to arbitrarily poor performance relative to running PC without any expert guidance—even with oracle conditional independence tests and infinite samples.

General mechanism of error propagation. This example illustrates a fundamental vulnerability of hard-constraint methods: causal discovery algorithms typically leverage results from early tests to determine which tests are run (and how their results are interpreted) at later stages. This sequential dependence is often necessary to efficiently navigate the super-exponential search space of DAGs. However, it creates a pathway for error propagation, where mistakes in discovering one part of the causal structure (such as incorrectly excluding edges based on expert constraints) cause: (i) crucial tests to never be performed, and (ii) results of later tests to be misinterpreted or leveraged incorrectly. When expert advice in the form of a hard constraint incorrectly determines parts of the causal graph, it initiates a cascade of errors that propagates throughout the algorithm, potentially leading to catastrophic failure even in the large-sample limit. This motivates our approach of using expert guidance to optimize test sequences rather than to replace statistical testing with hard constraints.

A.2 Sequential Testing in Causal Discovery

Here we discuss how sequential statistical testing underlies the three major paradigms of causal discovery: constraint-based, score-based, and functional causal model (FCM) based methods. In each paradigm, the tests performed at later stages fundamentally depend on the outcomes of earlier tests, creating a sequential dependence structure susceptible to error propagation.

Constraint-based methods. Constraint-based algorithms operate by performing conditional independence tests (CITs) to iteratively refine a candidate graph structure. Critically, which CITs are performed at any given stage depends on the results of CITs performed in earlier stages. This sequential dependence manifests primarily through *adjacency sets*: when testing whether to remove an edge between variables x_i and x_j, algorithms such as

PC (Spirtes, 2001) only condition on subsets of variables currently adjacent to x_i or x_j in the working skeleton. When earlier tests incorrectly remove (or retain) edges, the adjacency sets used in subsequent tests are corrupted. For instance, if a true edge $x_i \to x_k$ is incorrectly removed early in the algorithm, then x_k will be excluded from the adjacency set when later testing edges involving x_i, potentially causing the algorithm to miss the conditioning set needed to correctly identify other edges. This cascading effect means that errors in early tests propagate through the entire discovery process via their influence on which variables are considered in later conditioning sets.

Score-based methods. Score-based algorithms aim to find a DAG that maximizes a scoring function (e.g. Bayesian Information Criterion). The field encompasses diverse search strategies including: continuous optimization methods that relax the DAG constraint (Zheng et al., 2020), reinforcement learning approaches that learn search policies (Darvariu et al., 2024; Zhu et al., 2020; Wang et al., 2021), exact methods that guarantee global optimality through dynamic programming (Silander and Myllymaki, 2006), A* search (Yuan et al., 2011), integer programming (Dash et al., 2025; Dong et al., 2025), or Boolean satisfiability solvers (Hyttinen et al., 2014), and local search methods such as hill climbing (Tsamardinos et al., 2006) and greedy equivalence search (Chickering, 2002). Different score-based approaches differ fundamentally in their assumptions and guarantees: exact methods typically require additional structural assumptions for tractability (e.g., linear Gaussian assumptions in integer programming (Dash et al., 2025; Dong et al., 2025), decomposable scores in dynamic programming (Silander and Myllymaki, 2006) and A* (Yuan et al., 2011), continuous optimization methods either make parametric assumptions (Deng et al., 2024) or relax consistency guarantees (Zheng et al., 2020), and RL approaches often assume additive noise models (Darvariu et al., 2024; Zhu et al., 2020; Wang et al., 2021).

We focus specifically on applying Guess2Graph to local search methods such as GES that traverse the search space through sequential edge modifications—adding, deleting, or reversing edges one at a time. This is because the local search approach has the advantage of tractability, as well as established consistency results that do not rely on parametric assumptions or restrictions on functional forms (Agaram, 2024). Therefore, this subclass of methods aligns well with our general setting, as defined by Assumptions B.1-B.4.

Among local search methods, the GES algorithm has a few notable advantages. GES has been modified and parallelized to allow scaling to extremely large graphs (1 million variables considered by FGES (Ramsey et al., 2017)), while retaining reasonable performance. If the user prioritizes accuracy and is willing to supply more computational resources, GES can be enhanced with restart heuristics that forcefully perturb the search to escape local optima (as explored in XGES (Nazaret and Blei, 2024)). This can boost performance (somewhat) proportionally to the amount of additional resources supplied (i.e. the # of restart attempts), which allows GES to scale in finite samples by "searching for longer". Indeed, GES, unlike many newer search approaches, has also been extended to more general settings with even looser assumptions than the ones we study in our paper. These include unobserved confounders (Claassen and Bucur, 2022) and mixed observational-interventional datasets (Ejaz and Bareinboim, 2025). This underscores how widespread the local search approach underlying GES is in the causal discovery literature.

At each step, local search algorithms such as GES evaluate the score change associated with each possible modification and select the move that most improves the score. Crucially, the score assigned to any proposed edge modification depends on the *current parent sets* of the variables involved, which are themselves determined by all prior edge additions and deletions. For example, when considering whether to add edge $x_i \to x_j$, the score change is computed based on x_j's current parent set $\mathrm{Pa}(x_j)$; if previous iterations incorrectly modified $\mathrm{Pa}(x_j)$, the score for this new edge will be miscalculated. Recent work has formalized score-based methods as performing implicit conditional independence tests (Chickering, 2020), making the connection to sequential testing even more explicit. Like constraint-based methods, score-based approaches exhibit cascading errors where early mistakes in edge selection corrupt the parent sets used for scoring future edge modifications.

FCM-based methods. Functional causal model (FCM) based methods, specifically those based on additive noise models (ANM) (Zhang and Hyvarinen, 2008), typically operate by constructing a topological ordering of variables through recursive identification of roots (variables with no parents) or leaves (variables with no children (Montagna et al., 2023b,a; Hiremath et al., 2024, 2025)). The standard approach maintains a set of unsorted variables and, at each iteration, tests which of these variables satisfy the root or leaf condition by checking whether they are independent of other unsorted variables conditional on already-sorted variables. For instance, when identifying leaves, a variable x_i is classified as a leaf if its residuals (after regressing on each other unsorted variable individually) are independent of those regressors. The key sequential dependence arises because: (i) the

set of candidate variables tested for root/leaf status at each iteration depends on which variables were identified in previous iterations, and (ii) the conditioning sets used in these independence tests (the already-sorted variables) are built up incrementally based on prior test results. If an earlier iteration incorrectly identifies a non-leaf as a leaf, this error propagates by corrupting both the candidate set and the conditioning sets used in all subsequent iterations. This can be viewed as a series of tests for root/leaf status, where the outcome of each test determines which variables and conditioning sets are used in future tests.

Implications for expert guidance. This universal reliance on sequential testing across all three paradigms creates both a challenge and an opportunity. The challenge is that error propagation can cause early mistakes to cascade throughout the algorithm. The opportunity is that by optimizing the *sequence* in which tests are performed—without changing the tests themselves—we can improve finite-sample performance while preserving asymptotic correctness. Our G2G framework exploits this insight by using expert predictions to guide test sequences rather than to replace statistical testing, making it broadly applicable across constraint-based, score-based, and FCM-based methods.

A.3 Expert Error Violating Guarantees for Expert-Aided Discovery with Soft Constraints

Traditional expert-aided discovery methods integrate expert predictions through either hard constraints (which enforce expert beliefs directly) or soft constraints (which bias statistical procedures toward expert beliefs). In Appendix A.1, we demonstrated how hard constraints can lead to unbounded error through error propagation. Here, we discuss how soft constraint approaches fundamentally compromise the theoretical guarantees of causal discovery algorithms in both score-based and constraint-based methods. In what follows we summarize the analysis of Wu et al. (2025), who demonstrate that soft constraint methods break the mathematical foundations underlying both paradigms.

Soft constraints in score-based methods. Score-based methods incorporating soft constraints typically modify the scoring function by adding a prior term based on expert predictions:

$$\sigma(G; D, \lambda) = \sigma(G; D) + \sigma(G; \lambda)$$

where $\sigma(G; D)$ evaluates how well graph G fits the data D, and $\sigma(G; \lambda)$ rewards or penalizes structures based on expert constraints λ. This direct summation creates three fundamental problems that undermine theoretical guarantees:

First, *the terms operate in incompatible probability spaces.* The data term $\sigma(G; D)$ reflects log-likelihood under sample data, while the prior term $\sigma(G; \lambda)$ encodes expert beliefs independent of data. These quantities have different statistical foundations and cannot be meaningfully combined through simple addition. The scale mismatch is severe: for example, in the BIC score $\sigma_{BIC}(G; D) = \log P(D|G) - \frac{k}{2} \log N$, the sample size N dramatically affects the magnitude of the data term, while $\sigma(G; \lambda)$ is independent of N. Without principled normalization, one term typically dominates, either drowning out expert guidance or allowing poor expert predictions to overwhelm statistical evidence.

Second, *existing scoring functions already contain implicit priors that conflict with expert priors.* For instance, the BDeu score (Scanagatta et al., 2014) assumes Dirichlet uniform priors and prior independence of all variables—assumptions that directly contradict typical expert predictions about causal relationships. Introducing $\sigma(G; \lambda)$ creates competing prior specifications with no principled mechanism for reconciliation, requiring ad-hoc hyperparameters to balance multiple conflicting priors and data fit.

Third, *adding expert priors breaks critical structural properties.* Traditional scoring functions satisfy *decomposability*, allowing the score to be written as $\sigma(G; D) = \sum_{i=1}^{n} \sigma(x_i, \mathrm{Pa}(x_i); D)$, which enables efficient local optimization. Expert priors $\sigma(G; \lambda)$ often impose global constraints across multiple variables that cannot be decomposed into local contributions, rendering many optimization algorithms inapplicable. Additionally, soft constraints violate *score local consistency*, which guarantees that improving local fit to data improves the overall score. When expert priors incorrectly emphasize certain relationships, local changes that better reflect data may receive lower scores, breaking the connection between score optimization and causal structure recovery.

Soft constraints in constraint-based methods. While hard constraints in constraint-based methods create error propagation pathways (Appendix A.1), soft constraint approaches that incorporate edge priors from experts into conditional independence testing face a different but equally fundamental issue: they invalidate the distributional theory underlying hypothesis tests.

Some approaches modify conditional independence test statistics by incorporating expert edge priors. For example, methods that integrate soft constraints into the G^2 test adjust the statistic as:

$$G^2(X, Y|Z) - p > \chi^2_{\alpha,f}$$

where p encodes prior beliefs about the strength or likelihood of the causal relationship between X and Y. While this appears to simply modulate the rejection region based on expert confidence, it fundamentally invalidates the statistical theory underlying the test.

The core issue is that *subtracting p distorts the asymptotic distribution.* The G^2 statistic has well-established asymptotic properties—specifically, it follows a chi-squared distribution under the null hypothesis of conditional independence. Subtracting an arbitrary prior term p based on expert beliefs destroys these properties: the modified statistic $G^2(X, Y|Z) - p$ no longer follows a chi-squared distribution, invalidating hypothesis tests based on comparing to $\chi^2_{\alpha,f}$ critical values. Even if the modified statistic could be shown to asymptotically follow some chi-squared distribution, the degrees of freedom f and critical values would need to be rederived from first principles—a non-trivial task that existing methods do not address. Without valid asymptotic theory, there are no guarantees about Type I error rates, power, or consistency.

This problem extends beyond the G^2 test to any approach that incorporates expert edge priors by modifying test statistics. Whether using Fisher's Z-test, permutation tests, or other conditional independence tests, directly adjusting the test statistic or critical values based on soft expert constraints breaks the calibration that ensures valid statistical inference. The result is that even with infinite data, these methods cannot guarantee asymptotic correctness when expert predictions are inaccurate.

Implications. Both score-based and constraint-based soft constraint approaches sacrifice the rigorous statistical foundations that provide guarantees for purely data-driven methods. In score-based methods, soft constraints introduce mathematically inconsistent score combinations and break structural properties needed for optimization. In constraint-based methods, soft constraints invalidate the distributional theory underlying hypothesis tests. These are not merely technical concerns—they represent fundamental incompatibilities between integrating unreliable expert predictions through soft constraints and maintaining statistical guarantees. Our G2G framework addresses these issues by using expert predictions to guide test *sequences* rather than test *outcomes*, preserving the statistical validity of each individual test while leveraging expert knowledge to improve finite-sample efficiency.

B Definitions

This section details the core assumptions underlying our causal structure learning framework.

Definition B.1 (Causal Markov Condition, Spirtes 2001)**.** *A causal graph $\mathcal{G} = (V, E)$ satisfies the Causal Markov Condition if and only if every variable $x_i \in V$ is independent of its non-descendants given its parents $\mathrm{Pa}(x_i)$. This implies that the joint distribution $p(V)$ factorizes as:*

$$p(V) = \prod_{i=1}^{d} p(x_i \mid \mathrm{Pa}(x_i)). \tag{2}$$

Definition B.2 (Acyclicity, Spirtes 2001)**.** *A causal graph $\mathcal{G}$ is acyclic if it contains no directed paths starting and ending at the same node.*

Definition B.3 (Faithfulness, Spirtes 2001)**.** *A distribution p is faithful to a graph $\mathcal{G}$ if every conditional independence relation present in p is entailed by the Causal Markov Condition applied to $\mathcal{G}$. That is, for any disjoint sets of variables x, y, Z:*

$$x \perp\!\!\!\perp y \mid Z \text{ in } p \quad \rightarrow \quad Z \text{ d-separates } x \text{ and } y \text{ in } \mathcal{G}.$$

Definition B.4 (Causal Sufficiency, Spirtes 2001)**.** *A set of variables V is causally sufficient if there exist no unobserved confounders for any pair of variables in V. Formally, for any $x_i, x_j \in V$, there is no unmeasured variable $U \notin V$ that is a direct cause of both x_i and x_j in the true causal graph.*

Definition B.5 (d-Separation, Spirtes 2001)**.** *A set of variables $\mathbf{Z}$ d-separates variables x and y in a graph $\mathcal{G}$ if and only if $\mathbf{Z}$ blocks all paths between x and y. This graphical condition implies the conditional independence $x \perp\!\!\!\perp y \mid \mathbf{Z}$ in every distribution that is Markov with respect to $\mathcal{G}$.*

Definition B.6 (Markov Equivalence Class, Spirtes 2001)**.** *The Markov equivalence class (MEC) of a DAG $\mathcal{G}$ is the set of all DAGs that imply the same set of conditional independence statements via d-separation. Under the assumptions of causal Markov, faithfulness, and causal sufficiency, the MEC can be identified from perfect conditional independence tests and is typically represented by a Completed Partially Directed Acyclic Graph (CPDAG).*

Definition B.7 (Sufficient Power and Mutual Independence of CITs)**.** *We assume that conditional independence tests satisfy two properties:*

1. ***Sufficient Specificity**: CITs are adequately powered such that $1 - \alpha > \beta$, where α is the Type I error rate (false positive rate) and β is the Type II error rate (false negative rate) of each CIT.*
2. ***Mutual Independence**: The outcomes of different CITs are mutually independent. Formally, for CITs indexed by $i \in \{1, \ldots, m\}$:*

$$\mathbb{P}(CIT_1, \ldots, CIT_m) = \prod_{i=1}^{m} \mathbb{P}(CIT_i)$$

The 'sufficient specificity' assumption claims that each CIT has a false positive and false negative rate of α, β respectively, and ensures the true negative rate exceeds the false negative rate, i.e., that there is enough data collected such that CITs that condition on d-separating sets return independence with higher probability than those that do not.

The 'mutual independence' assumption is a simplifying assumption for theoretical analysis commonly made in causal discovery (Brown and Tsamardinos, 2008; Li and Wang, 2009; Strobl et al., 2019; Cooper and Yoo, 1999). The statement holds exactly with infinite data or when using data-splitting (where each CIT uses an independent sample), but is an approximation when CITs reuse the same finite dataset, as is common in causal discovery.

We invoke these assumptions for our runtime analysis (Subroutine 4, Lemma D.9), but note that they are both not used in our accuracy analysis (Subroutine 3, Lemma 4.1).

Definition B.8 (Completed Partially Directed Acyclic Graph)**.** *Given a DAG $\mathcal{G}$, its* completed partially directed acyclic graph *(CPDAG) is the unique partially directed graph representing the Markov equivalence class of $\mathcal{G}$. A directed edge in the CPDAG is* compelled*, meaning it has the same orientation in every DAG in the class, while an undirected edge is* reversible*, meaning it admits both orientations across DAGs in the class.*

C CD-GUESS Framework and Application to Constraint-Based Discovery

C.1 Extensions of CD-GUESS Framework

Here we outline a few possible extensions of the CD-GUESS Framework.

C.1.1 Heuristic Selection of Experts and Pruning of Guesses

When integrating LLM predictions with data-driven causal discovery, we identify three distinct regimes: World 1 where data-driven methods alone perform best (LLM guidance degrades performance), World 2 where hybrid approaches excel (combining data and LLM guidance), and World 3 where the LLM guess alone suffices (outperforming empirical methods). Which world applies depends on the interplay between LLM guess quality, sample size, and the underlying graph structure—with higher-quality guesses and smaller samples favoring Worlds 2 and 3. Our primary concern is preventing World 1 scenarios by detecting when LLM predictions are harmful, as this represents the most critical failure mode where expert guidance actively degrades baseline performance. There are two main approaches to trying to prevent World 1 from occurring. The first is validate whether the specific guess given by an expert is better than random (Checking Guess Quality). If this is infeasible, we can attempt to identify whether the expert is any good in this domain (Checking Expert Competency).

Guess Quality. This approach evaluates specific causal structure predictions by assigning fitness scores that correlate with graph accuracy—higher scores indicate more accurate graphs. To assess whether a given score is meaningful, we construct a null distribution by sampling random graphs and testing whether the expert's guess scores significantly above this baseline. Two primary methodologies exist: likelihood-based and nonparametric approaches, which differ in computational complexity and modeling flexibility.

Likelihood-based methods (Huang et al., 2018) assume parametric models for the underlying functional relationships (e.g., residing in specific kernel spaces or exponential families) and derive scores that measure graph-data compatibility. Random graphs of similar density provide the comparison baseline. Notably, likelihood scores do not monotonically increase with graph accuracy as measured by skeleton or edge metrics, serving instead as relative comparison measures between candidate structures.

Nonparametric approaches, exemplified by permutation-based methods (Eulig et al., 2025), evaluate graphs by counting how many conditional independence tests in the data align with the d-separation statements implied by each graph. When sampling comparison graphs, we can either generate them uniformly at random or preserve specific properties of the expert guess, such as sparsity constraints or causal ordering structure. While these methods provide monotonic relationships between scores and accuracy (measured by satisfied CI statements), they incur higher computational costs and may exhibit greater sensitivity to finite-sample effects.

Formally, we suggest to compute a p-value as $p = \frac{1}{m}\sum_{i=1}^{m} \mathbb{1}[\text{score}(\widehat{\mathcal{G}}_{\mathcal{R}_i}) \geq \text{score}(\widehat{\mathcal{G}})]$, where $\{\widehat{\mathcal{G}}_{\mathcal{R}_i}\}_{i=1}^{m}$ are randomly sampled graphs preserving relevant properties (e.g., edge density), and $\widehat{\mathcal{G}}$ is the expert prediction. If $p < \alpha_{\text{val}}$ for significance level α_{val} (e.g., 0.05), indicating the expert guess scores significantly better than random, we incorporate the guidance into test prioritization; otherwise, we proceed without expert guidance to maintain baseline performance guarantees.

We note that there are roughly 3 possible scenarios where it is difficult to directly assess the quality of a guess. The first is that data-driven methods may require strong assumptions on the DGP, such as parametric functional assumptions, that may not be satisfied. Additionally, some types of graph structures, such as dense graphs, provide relatively few CIs for nonparametric scores to validate, leading to low power in those regimes. Additionally, while there are many methods for assigning a score to causal graphs, there are fewer methods for assessing the fittingness of causal orderings, a fundamentally more difficult problem. Therefore, in these cases where it is hard to leverage the data itself, we propose to rely on a data-independent measure: the self-consistency of the expert.

Expert Self-Consistency. As suggested by Faller et al. (2024), we assess expert reliability by evaluating self-consistency across predictions on overlapping variable subsets. This data-free approach requires the expert to produce causal graphs for multiple variable subsets and measures agreement where these subsets overlap—paralleling recent findings that LLM response consistency under prompt variations correlates with reliability. We suggest to compute a consistency score $\mathcal{C}(\psi) = \frac{1}{|\mathcal{P}|}\sum_{(S_1,S_2)\in\mathcal{P}} \text{sim}(\widehat{\mathcal{G}}_\psi(S_1 \cup S_2)|_{S_1\cap S_2}, \widehat{\mathcal{G}}_\psi(S_1 \cap S_2))$, where ψ is the expert, $\mathcal{P}$ is a collection of overlapping variable subset pairs, and $\text{sim}(\cdot,\cdot)$ is some sort of measure of structural similarity

on the shared variables. This metric provides theoretical guarantees: a perfectly accurate expert must exhibit perfect self-consistency. If $\mathcal{C}(\psi) > \tau_c$ for a pre-specified threshold τ_c (e.g., 0.7), we proceed with expert-guided discovery; otherwise, we default to vanilla PC to preserve baseline performance. While this approach effectively screens for domain competence, it faces two limitations: the computational cost of querying numerous subsets may be prohibitive, and it evaluates general expert reliability rather than the quality of specific structural predictions.

C.1.2 Integrating Uncertainty Quantification

We note that PC-Guess can be extended to leverage confidence estimates, for use in situations where an expert can do uncertainty quantification for their predictions. For example, the ordering algorithms (Subroutines 3, 4) can be modified such that the orderings are not drawn uniformly from two buckets, but rather according to confidence score. More formally, when the expert provides edge confidence scores $w_\psi(x_i, x_j) \in [0, 1]$ and ordering confidence $\pi_\psi(x_i \prec x_j) \in [0, 1]$, we suggest to modify test prioritization to sample probabilistically rather than deterministically. For edge testing order, we sample the next pair (x_i, x_j) with probability proportional to $1 - w_\psi(x_i, x_j)$, favoring likely non-edges. For conditioning set selection, we sample set S with probability proportional to $\prod_{z \in S} \pi_\psi(z \prec \{x_i, x_j\})$, favoring sets containing likely ancestors. This probabilistic approach naturally interpolates between expert-guided and random search based on prediction confidence, maintains theoretical guarantees (all tests have non-zero probability), and becomes exploratory under high uncertainty. With uniform uncertainties, it reduces to standard PC.

C.1.3 Leveraging Causal Reasoning

Additionally, when economically or computationally feasible, after a number of structural decisions (e.g., edge confirmation/removal) are made, we could in theory reprompt the expert to provide a new guess using the updated partial structure, repeating the process until the algorithm terminates. This iterative refinement would then leverage the compositional reasoning abilities of experts to produce progressively better predictions as structural information is confirmed, allowing them to correct earlier mistakes and incorporate validated constraints. To fully exploit expert capabilities, we reprompt the expert for updated predictions after each edge decision during discovery. This iterative approach offers two key advantages: first, as the structure is progressively verified, the problem dimensionality decreases, where LLMs demonstrably achieve higher accuracy on smaller graphs; second, the confirmed structural information provides additional context that allows the expert to correct earlier mistakes and make more informed predictions about remaining edges. Thus, rather than using a static initial guess, we dynamically update expert guidance throughout the discovery process.

C.2 Extension to Score-Based Methods

Method description. Score-based algorithms like GES (Chickering, 2002) discover causal structure by greedily optimizing a scoring function, iteratively adding or removing edges that maximize the score improvement. A variant called SE-GES (Chickering, 2020) restricts the search to statistically efficient operators—those conditioning on the fewest variables—to improve finite-sample performance:

SE-GES (Algorithm 2) operates as follows. First (Line 1), it applies FINDIMAP to identify an initial graph structure which contains the conditional independencies in the data. It then progressively iterates through increasing values of bound k on the conditioning set size, starting from $k = 0$. At each iteration, two key steps occur:

Line 5 - UPDATESEPARATORS: This subroutine identifies all order-k separators M^k by testing conditional independence statements of size k. These separators capture which variable pairs are conditionally independent given k conditioning variables, forming the statistical basis for edge removal decisions. Specifically, UPDATESEPARATORS tests CI statements of the form $\text{CIT}(x_i, x_j \mid W)$ where $|W| = k$, storing those that return independent as order-k separators. This collection M^k grows as k increases, accumulating evidence about which edges can be safely removed.

Line 6 - SE-BES: Using the separators M^k, SE-BES performs backward elimination by evaluating deletion operators (edge removals) while restricting to those of order at most k, optimizing a scoring function that balances fit and complexity.

Algorithm A Statistically Efficient Greedy Equivalence Search (SE-GES)

1: **Input**: Data D
2: **Output**: CPDAG $\mathcal{C}$
3: $\mathcal{C} \leftarrow$ FINDIMAP(D)
4: $M^{-1} \leftarrow$ UNDEFINED for all node pairs
5: $k \leftarrow 0$
6: **repeat**
7: $\quad M^k \leftarrow$ UPDATESEPARATORS(M^{k-1}, k)
8: $\quad \mathcal{C} \leftarrow$ SE-BES($\mathcal{C}$, k)
9: $\quad$ **if** every node in $\mathcal{C}$ has $\leq k$ parents **then**
10: $\quad\quad$ **return** $\mathcal{C}$
11: $\quad$ **else**
12: $\quad\quad k \leftarrow k + 1$
13: $\quad$ **end if**
14: **until** convergence

After SE-BES completes, Line 7 checks whether the resulting CPDAG is consistent with bound k—whether every node has at most k parents. If consistent, the algorithm terminates; otherwise, it increments k and repeats, testing higher-order conditional independencies.

Computational challenge and proposed modification. As graph size grows, exhaustively testing all k-order CI statements becomes computationally prohibitive. This necessitates imposing a budget Q on the number of CI tests per edge per round, inducing a preference over which tests to prioritize and in what order to execute them.

We propose using G2G-guided test ordering as a subroutine within SE-GES. Specifically, for each conditioning set size k, rather than testing CI statements uniformly at random, we apply the ordering principles from PC-Guess (Section 4) to prioritize tests: (1) extract edge predictions from expert graph $\widehat{\mathcal{G}}$, (2) prioritize testing edges the expert believes are false (likely to yield independence), and (3) within each edge's tests, prioritize conditioning sets the expert predicts are d-separating.

Potential benefits. This modification offers several advantages while preserving SE-GES's theoretical guarantees:

- *Budget efficiency*: Under limited testing budgets, prioritizing tests more likely to reveal true independencies increases the probability of correctly identifying removable edges within the allocated Q tests per edge
- *Search space reduction*: Earlier identification of true non-edges prunes the graph faster, reducing downstream computational costs
- *Preserved guarantees*: Since SE-GES's correctness depends only on testing sufficient CI statements (not their order), expert-guided prioritization cannot compromise asymptotic correctness

Conjectured theoretical properties. Analogous to our constraint-based results (Theorems 5.1, 5.3), we conjecture that expert-guided SE-GES satisfies similar guarantees. Specifically, we expect that as sample size $n \to \infty$ and testing budget $Q \to \infty$, the algorithm maintains statistical consistency and recovers the true skeleton regardless of expert quality, since all edges eventually receive their full budget of tests. For fixed Q and n, we conjecture that the expected probability of correct skeleton recovery increases monotonically with expert edge prediction accuracy p_ψ—better experts identify true non-edges earlier, increasing the probability that budget-limited tests discover removable edges. Finally, we expect finite-sample robustness: when $p_\psi \geq 0.5$, expert-guided test ordering should achieve performance at least as good as random ordering, since experts performing at chance produce orderings distributionally equivalent to random, with strictly better performance when $p_\psi > 0.5$. While we have not completed formal proofs of these properties, the intuition directly mirrors our constraint-based analysis, and we believe the results could be established using similar coupling arguments we use in our proofs.

Prior work has proposed expert-augmented score-based methods through heuristics like initialization with expert guesses or preferring operations agreeing with expert predictions (Constantinou et al., 2023; Ejaz and Bareinboim,

2025), but without formal guarantees. Our G2G framework suggests a principled approach with conjectured theoretical properties worth investigating in score-based methods.

C.3 Extension to ANM-Based Methods

Method description. ANM-based algorithms discover causal structure by exploiting asymmetries in functional relationships under additive noise models. A prominent example is RESIT (Peters et al., 2014), which identifies the causal DAG by iteratively finding leaf nodes (variables with no children):

Algorithm B RESIT (Simplified)

1: **Input**: Data D on variables $V = \{x_1, \ldots, x_d\}$
2: **Output**: Parent sets $\{\text{Pa}(x_i)\}_{i=1}^d$
3: $S \leftarrow V$, $\pi \leftarrow []$ ▷ S tracks remaining variables, π builds topological order
4: **Phase 1: Determine topological order**
5: **repeat**
6: **for** each $x_k \in S$ **do**
7: Regress x_k on $S \setminus \{x_k\}$
8: Test independence between residuals and $S \setminus \{x_k\}$
9: Record dependence strength
10: **end for**
11: Let x_{k^*} be variable with weakest dependence
12: $\text{Pa}(x_{k^*}) \leftarrow S \setminus \{x_{k^*}\}$
13: $S \leftarrow S \setminus \{x_{k^*}\}$
14: $\pi \leftarrow [x_{k^*}, \pi]$ ▷ Prepend to topological order
15: **until** $S = \emptyset$
16: **Phase 2: Prune superfluous edges** (analogous to constraint-based edge removal)
17: **return** $\{\text{Pa}(x_i)\}_{i=1}^d$

RESIT (Algorithm B) operates in two phases. Phase 1 (Lines 4-13) iteratively identifies leaf nodes by testing all remaining variables to find which has residuals most independent of the others—this variable is a leaf and removed from consideration. The algorithm builds a topological ordering π by prepending each identified leaf. Phase 2 (Line 14) then prunes unnecessary edges from the identified parent sets.

The critical computational bottleneck is Line 5: at each iteration with $|S|$ remaining variables, RESIT must test all $|S|$ candidates, requiring $O(d^2)$ total tests across iterations. Moreover, the order in which variables are tested affects accuracy—if a non-leaf is incorrectly identified as a leaf due to test errors (false positive), this error propagates through subsequent iterations.

Computational challenge and proposed modification. The key insight is that the order in which variables are tested in Line 5 critically impacts both efficiency and accuracy. Testing true leaves first reduces the number of tests (since the leaf is found immediately) and prevents false positive errors from incorrectly identifying non-leaves.

We propose modifying RESIT to use G2G-guided variable ordering. Specifically: (1) query expert ψ for a predicted topological ordering $\hat{\pi}$ over variables V, (2) in Line 5, test variables in the order specified by $\hat{\pi}$ rather than arbitrary order, prioritizing variables the expert predicts appear later in topological order (more likely to be leaves). This modification directly parallels our edge ordering principle from Section 4—prioritizing tests more likely to succeed.

Potential benefits. This modification offers several advantages:

- *Computational efficiency*: Testing true leaves first reduces the expected number of independence tests per iteration from $O(|S|)$ to $O(1)$ when expert predictions are accurate
- *Error reduction*: Identifying true leaves early prevents false positives on non-leaves, reducing error propagation through the topological ordering

- *Preserved guarantees*: Since RESIT's correctness depends on testing all variables (not their order), expert-guided prioritization cannot compromise asymptotic correctness

Conjectured theoretical properties. Analogous to our constraint-based results (Theorems 5.1, 5.3), we conjecture that expert-guided RESIT satisfies similar guarantees. Let p_π denote the expert's accuracy in predicting topological orderings (probability that for a random pair (x_i, x_j), if x_i is an ancestor of x_j, then $\hat{\pi}(x_i) < \hat{\pi}(x_j)$). We expect that as sample size $n \to \infty$, the algorithm maintains statistical consistency and recovers the true DAG structure regardless of p_π, since all variables are eventually tested. For fixed n, we conjecture that the expected probability of correct DAG recovery increases monotonically with p_π—better topological ordering predictions place true leaves earlier in the test sequence, increasing the probability of correct identification before errors accumulate. We also expect finite-sample robustness: when $p_\pi \geq 0.5$, expert-guided ordering should achieve performance at least as good as random variable ordering, since random orderings provide the baseline ($p_\pi = 0.5$) with improvement when $p_\pi > 0.5$. Finally, we conjecture that the expected number of tests decreases monotonically with p_π because testing true leaves first terminates the inner loop earlier, providing computational efficiency gains. While we have not completed formal proofs of these properties, the intuition follows directly from our core insights, and we believe the results are straightforward to establish using similar analytical techniques.

This extension demonstrates how G2G principles—using expert predictions to guide test sequences—apply beyond constraint-based methods to functional causal model approaches, suggesting potential broad applicability of the framework.

C.4 Decomposition of Constraint-Based Discovery into Edge Prune and Edge Loop

We demonstrate how two constraint-based algorithms—PC and rPC-approx—decompose into the Edge Loop (EL) and Edge Prune (EP) subroutines defined in Section 4.

PC Algorithm. The PC algorithm iteratively tests edges with increasing conditioning set sizes ℓ, removing edges when independence is found.

Algorithm C PC (Skeleton Discovery)

Input: Complete skeleton $\mathcal{C}$ over variables V
Sample O uniformly, sample L uniformly
Define $R^{\ell}_{PC}(\mathcal{C}, e_{i,j}) = \mathbb{1}[n_{i,j} \in \mathcal{C}] \wedge \mathbb{1}[|\text{adj}_{-j}(\mathcal{C}, x_i)| \geq \ell]$
for $\ell = 0$ to $|V| - 1$ **do**
    $\mathcal{C} \leftarrow \text{EL}(\mathcal{C}, \mathsf{O}, [\ell, \ell], \mathsf{L}, R^{\ell}_{PC}, \text{EP})$
    if no edges satisfy R^{ℓ}_{PC} **then break**
end for
return $\mathcal{C}$

rPC-approx Algorithm. The rPC-approx algorithm (Sondhi and Shojaie, 2019) bounds conditioning set sizes to $\eta < |V| - 1$ and modifies the conditioning set search space to use local neighborhoods.

Algorithm D rPC-approx (Skeleton Discovery)

Input: Complete skeleton $\mathcal{C}$ over variables V, maximum size η
Sample O uniformly, sample L uniformly
Define $R_{\text{rPC}}(\mathcal{C}, e_{i,j}) = \mathbb{1}[n_{i,j} \in \mathcal{C}]$
for $\ell = 0$ to η **do**
    $\mathcal{C} \leftarrow \text{EL}(\mathcal{C}, \mathsf{O}, [\ell, \ell], \mathsf{L}, R_{\text{rPC}}, \text{EP})$
end for
return $\mathcal{C}$

Key differences in decomposition:

- *Conditioning set size bound*: PC iterates up to $|V| - 1$; rPC-approx stops at η.
- *Validity rule*: PC's R^{ℓ}_{PC} requires $|\text{adj}_{-j}(\mathcal{C}, x_i)| \geq \ell$ (sufficient adjacency); rPC-approx's R_{rPC} only checks edge presence.

- *Conditioning set source*: In EP, PC draws subsets from $\text{adj}_{-j}(\mathcal{C}, x_i)$; rPC-approx draws from $\text{adj}(\mathcal{C}, x_i) \cup \text{adj}(\mathcal{C}, x_j) \setminus \{x_i, x_j\}$ (union of both endpoints' neighborhoods).

Both algorithms follow the same template—iteratively calling EL with increasing ℓ—differing only in their validity rules, conditioning set size bounds, and the search space for conditioning sets within EP.

C.5 Complexities of Error Propagation in Edge Loop

We explain why error-tolerant metrics—such as expected number of correct edges $\mathbb{E}[\sum_{n_{i,j}} Y_{n_{i,j}}]$—are analytically intractable for ordering optimization, motivating our focus on perfect recovery probability Φ in Section 4.1.

Notation. Recall that true edges $n_{i,j} \in \mathcal{S}^*$ are edges present in the true skeleton, while false edges $n_{i,j} \notin \mathcal{S}^*$ are edges absent from the true skeleton but present in the current candidate skeleton $\mathcal{C}$ being tested.

Requirements for analyzing error-tolerant metrics. To optimize orderings under error-tolerant metrics, we must:

(a) *Identify qualitatively different error types*: Constraint-based algorithms make two types of errors—false positives (incorrectly retaining false edges) and false negatives (incorrectly removing true edges)

(b) *Determine the magnitude of each error type's impact*: Quantify how each error affects the probability of correctly classifying subsequent edges

(c) *Characterize how sequence changes affect errors*: Predict which sequence modifications reduce overall error rates

Opposing downstream effects (Challenge for (a)). False positive and false negative errors have conflicting impacts on future tests. Consider an edge ordering where edge n_k is incorrectly decided at position k. For any subsequent edge n_j where $j > k$:

False Positive (incorrectly retaining false edge $n_k \notin \mathcal{S}^*$):

- The retained false edge artificially inflates $|\text{adj}(\mathcal{C}, x_m)|$ for vertices x_m adjacent to n_k
- By Lemma D.3, larger adjacency sets increase the number of CI tests, which:
  - *Helps* remove subsequent false edges (more tests → higher chance of finding independence)
  - *Hurts* retention of subsequent true edges (more tests → higher chance of spurious independence due to finite-sample errors)

False Negative (incorrectly removing true edge $n_k \in \mathcal{S}^*$):

- The removed true edge artificially deflates $|\text{adj}(\mathcal{C}, x_m)|$ for vertices x_m adjacent to n_k
- By Lemma D.3, smaller adjacency sets decrease the number of CI tests, which:
  - *Helps* retention of subsequent true edges (fewer tests → lower chance of spurious independence)
  - *Hurts* removal of subsequent false edges (fewer tests → lower chance of finding true independence; may also exclude crucial d-separating variables from adjacency sets)

Since the two error types have opposing effects—false positives benefit false edge detection but harm true edge detection, while false negatives do the reverse— its not clear whether a single ordering strategy dominates without knowing how the effect of the true graph structure, i.e. whether there are many true edges or false edges.

Unknown Error Magnitudes (Challenge for (b)). The magnitude of error propagation depends critically on graph topology. Consider testing edges $n_{i,j}$ and $n_{g,h}$:

- If $n_{i,j}$ and $n_{g,h}$ share vertices, errors on $n_{i,j}$ directly modify the adjacency sets used when testing $n_{g,h}$

- The severity depends on whether the removed/retained edge contains d-separating variables for $n_{g,h}$
- The true/false edge ratio in the graph determines whether false positive or false negative errors dominate overall performance
- The error rate values (α, β play a role in the likliehood of different types of errors; if the β is high or low, this might affect whether we optimize the sequence towards preventing false positives or false negatives.

Analytical Intractability (Challenge for (c)). The challenges for (a) and (b) make it difficult to analyze theoretically whether a one sequence dominates another, as (a) presents a challenge to the notion that, even with perfect information of the ground truth, its not clear how to determine the effect of swapping edges in a sequence, and (b) shows that there is important information such as graph structure and error rates missing that may be crucial to the analysis.

Conclusion. This graph-dependent complexity motivates our restriction to perfect recovery probability $\Phi = \mathbb{P}[\prod Y_{n_{i,j}} = 1]$ in Section 4.1. By conditioning on no prior errors (perfect recovery up to position $i-1$), we eliminate error propagation from the analysis, enabling the clean characterization in Lemma D.5: placing false edges before true edges is universally beneficial regardless of graph structure.

D Lemmas, Theorems, and Proofs

D.1 Proof of Lemma D.1

Lemma D.1. *Under oracle CITs, when $\mathcal{C}$ is the complete graph over V, EL($\mathcal{C}$, O, $[0, |V|-1]$, L, R, EP) returns the same skeleton for any edge ordering O (and any L by Lemma D.2).*

Proof. We prove that when given oracle CITs, if $\mathcal{C}$ is complete then the set of edges removed by EL is independent of O. We show that each edge's fate is determined solely by the existence of d-separating sets, not by the order of processing.

All true edges are retained: Consider any edge $n_{i,j} \in \mathcal{C}$ that is a true edge in the skeleton of $\mathcal{G}$. By faithfulness, $x_i \not\perp\!\!\!\perp x_j \mid S$ for all $S \subseteq V \setminus \{x_i, x_j\}$. When EL processes this edge (at any position in O), all CI tests return dependent under perfect tests, so $n_{i,j}$ is never removed. Thus, all true edges remain in $\mathcal{C}$ throughout execution regardless of O.

All false edges are removed: Consider any edge $n_{i,j} \in \mathcal{C}$ that is not in the true skeleton. By faithfulness and causal sufficiency, there exists $S \subseteq V \setminus \{x_i, x_j\}$ such that $x_i \perp\!\!\!\perp x_j \mid S$. By the Markov Condition (Definition B.1), one such set must exist, for example at least one of $\mathrm{Pa}(x_i)$ or $\mathrm{Pa}(x_j)$. Let S be such a d-separating set.

Since $\mathcal{C}$ is initially complete and all parent edges are true edges in the skeleton, these parent edges remain in $\mathcal{C}$ throughout execution (as all true edges are retained). Therefore, regardless of when EL processes edge $n_{i,j}$ in the ordering O, we have $S \subseteq \mathrm{adj}_{-j}(\mathcal{C}, x_i)$ when this edge is tested. EP will test with conditioning set S, the test returns independent under perfect CI tests, and $n_{i,j}$ is removed.

Since both the retention of true edges and removal of false edges are independent of O when $\mathcal{C}$ is complete, EL returns the same skeleton for any edge ordering. □

D.2 Proof of Lemma D.2

Lemma D.2. *Under oracle CITs, for any edge $e_{i,j}$ and conditioning set size k, EP($\mathcal{C}$, $e_{i,j}$, k, L) returns the same result for any subset ordering L.*

Proof. EP tests all subsets $s \subseteq [\mathrm{adj}_{-j}(\mathcal{C}, x_i)]_k$ and removes edge $n_{i,j}$ if any $\mathrm{CIT}(x_i, x_j \mid s)$ returns independent. With perfect CITs, the outcome of each test is deterministic and depends only on the conditioning set s, not on when it is tested.

If there exists $s \subseteq [\mathrm{adj}_{-j}(\mathcal{C}, x_i)]_k$ such that $x_i \perp\!\!\!\perp x_j \mid s$, then EP will remove $n_{i,j}$ when it tests this conditioning set, regardless of which ordering L is used to sequence the tests.

If no such s exists, then all tests return dependent and $n_{i,j}$ remains, again regardless of L.

Since the decision to remove or retain the edge depends only on the existence of a d-separating set among the tested conditioning sets, not the order in which they are tested, EP returns the same result for any L under perfect CI tests. □

D.3 Proof Lemma D.3

Lemma D.3. *For a true edge $n_{i,j}$ (where $n_{i,j} \in \mathcal{S}^*$), adding vertices to the adjacency set $\mathrm{adj}(\mathcal{C}, x_i)$ strictly decreases $\mathbb{P}(Y_{n_{i,j}} = 1)$, while for a false edge $n_{i,j}$ (where $n_{i,j} \notin \mathcal{S}^*$) adding vertices strictly increases $\mathbb{P}(Y_{n_{i,j}} = 1)$.*

Proof. We say an edge $n_{i,j}$ is a *true edge* if $n_{i,j} \in \mathcal{S}^*$ and a *false edge* if $n_{i,j} \notin \mathcal{S}^*$.

Note in the k^{th} step of Subroutine 2, a true edge $n_{i,j}$ is kept if no CIT conditioning on a subset of size k from adjacency set $\mathrm{adj}_{-j}(\mathcal{C}, x_i)$ returns independent. Given that $n_{i,j}$ is a true edge, all CITs of size k return independent with the same false negative rate β. Then, increasing the size of $\mathrm{adj}(\mathcal{C}, x_i)$ can only increase the number of CITs run, which can only increase the probability that $n_{i,j}$ is incorrectly removed, which therefore decreases $\mathbb{P}(Y_{n_{i,j}} = 1)$.

For a false edge $n_{i,j}$, correct specification (i.e., $Y_{n_{i,j}} = 1$) occurs when the edge is removed, which happens if and only if at least one CIT in the k^{th} step returns independent. Let $A_1 \subset A_2$ where $A_1 = \text{adj}_{-j}(\mathcal{C}_1, x_i)$ and $A_2 = \text{adj}_{-j}(\mathcal{C}_2, x_i)$ denote two adjacency sets differing by the inclusion of additional vertices in A_2. Denote by $\mathcal{T}_1 = [A_1]_k$ and $\mathcal{T}_2 = [A_2]_k$ the corresponding sets of size-k conditioning sets. Since $A_1 \subset A_2$, we have $\mathcal{T}_1 \subset \mathcal{T}_2$. The probability of correct removal is:

$$\mathbb{P}(Y_{n_{i,j}} = 1 \mid A) = \mathbb{P}\left(\bigcup_{W \in [A]_k} \{\text{CIT}(x_i, x_j \mid W) = \text{independent}\} \right)$$

For any fixed false edge $n_{i,j} \notin \mathcal{S}^*$, each CIT has a positive probability of returning independent. Since $\mathcal{T}_1 \subset \mathcal{T}_2$, the union over $\mathcal{T}_2$ includes all events in the union over $\mathcal{T}_1$ plus additional events corresponding to tests on conditioning sets in $\mathcal{T}_2 \setminus \mathcal{T}_1$. Therefore:

$$\mathbb{P}(Y_{n_{i,j}} = 1 \mid A_1) < \mathbb{P}(Y_{n_{i,j}} = 1 \mid A_2)$$

. Thus, increasing $|\text{adj}(\mathcal{C}, x_i)|$ strictly increases $\mathbb{P}(Y_{n_{i,j}} = 1)$ for false edges. □

D.4 Proof of Lemma D.4

Lemma D.4. *For an ordering* O *containing true edge* $n_{g,h}$ *(where* $n_{g,h} \in \mathcal{S}^*$*) and false edge* $n_{i,j}$ *(where* $n_{i,j} \notin \mathcal{S}^*$*), we have that* $\mathbb{P}(Y_{n_{i,j}} = 1 | Y_{n_{g,h}} = 1) = \mathbb{P}(Y_{n_{i,j}} = 1)$ *while* $\mathbb{P}(Y_{n_{g,h}} = 1 | Y_{n_{i,j}} = 1) \geq \mathbb{P}(Y_{n_{g,h}} = 1)$*. Further, the inequality is strict when* $n_{i,j}$ *and* $n_{g,h}$ *share a vertex.*

Proof. We say an edge is a *true edge* if it is in $\mathcal{S}^*$ and a *false edge* if it is not in $\mathcal{S}^*$.

We define $\mathbb{P}(Y_{n_{i,j}} = 1 | Y_{n_{g,h}} = 1)$ as the probability of removing a false edge $n_{i,j}$ after correctly retaining a true edge $n_{g,h}$, and $\mathbb{P}(Y_{n_{i,j}} = 1)$ as the probability of removing the false edge $n_{i,j}$ without running a test on $n_{g,h}$.

If $n_{g,h}$ does not share a vertex with $n_{i,j}$ in $\mathcal{C}$, then the adjacency sets of $n_{g,h}$ and $n_{i,j}$ do not overlap. This implies that the event of correctly removing a false edge $n_{i,j}$ is independent of the event of correctly retaining a true edge $n_{g,h}$, as these two events depend only on the data on the node itself and its neighbors. However, if at least one vertex in $n_{g,h}$ is already in the adjacency set of one of the vertices $n_{i,j}$ in $\mathcal{C}$, then correctly retaining $n_{g,h}$ does not change the adjacency set of either vertex in $n_{i,j}$, since retaining an edge means keeping it in the skeleton without modification, which preserves all existing adjacency relationships and therefore does not alter which vertices are available for conditioning. This means which tests are run for $n_{i,j}$ don't change after correctly testing $n_{g,h}$, which means the probability of removing $n_{i,j}$ doesn't change.

We now define $\mathbb{P}(Y_{n_{g,h}} = 1 | Y_{n_{i,j}} = 1)$ as the probability of retaining true edge $n_{g,h}$ after correctly removing false edge $n_{i,j}$, while $\mathbb{P}(Y_{n_{g,h}} = 1)$ as the probability of correctly retaining true edge $n_{g,h}$ without testing $n_{i,j}$. Note that if $n_{i,j}$ does not share any vertex with $n_{g,h}$ in $\mathcal{C}$, then correctly removing $n_{i,j}$ does not affect the probability of retaining $n_{g,h}$. However, if they do share at least one vertex, then correctly removing $n_{i,j}$ reduces the adjacency set of at least one vertex of $n_{g,h}$. Then by Lemma D.3 the probability of correctly retaining $n_{g,h}$ strictly increases. □

D.5 Proof of Lemma D.5

Lemma D.5. *Given a sequence of edges* O *that are either true edges (in* $\mathcal{S}^*$*) or false edges (not in* $\mathcal{S}^*$*), for any pair of adjacent edges consisting of a true edge followed by a false edge, the sequence generated by swapping the pair is no worse in terms of* $\mathbb{P}\left[\bigcap_{n \in sequence} Y_n = 1\right]$*, and strictly better if the false edge and true edge share a node.*

Proof. To establish the lemma, it suffices to show that for any adjacent pair where a true edge n_t precedes a false edge n_f, swapping their order weakly improves the joint probability $\mathbb{P}\left[\bigcap_{n \in \text{sequence}} Y_n = 1\right]$, with strict improvement when n_t and n_f share a vertex.

We say an edge is a *true edge* if it is in $\mathcal{S}^*$ and a *false edge* if it is not in $\mathcal{S}^*$. Consider a sequence $\mathsf{O} = \ldots, n_i, n_t, n_f, n_j, \ldots$ and the swapped sequence $\mathsf{O}' = \ldots, n_i, n_f, n_t, n_j, \ldots$, where n_t is a true edge and n_f is a false edge.

The key observation is that we need only compare how the swap affects $\mathbb{P}(Y_{n_t} = 1, Y_{n_f} = 1 \mid$ all prior edges in O correctly classified). This is because: (1) edges processed before n_t and n_f are unaffected by their relative ordering, and (2) conditioned on both n_t and n_f being correctly classified, the resulting skeleton state is identical regardless of their processing order—correctly retaining n_t preserves adjacency sets while correctly removing n_f removes it from adjacency sets, and these operations commute. Therefore, the conditional probabilities for all subsequent edges n_k remain unchanged given correct classification of both n_t and n_f.

By Lemma D.4, we have:

$$\mathbb{P}(Y_{n_f} = 1 | Y_{n_t} = 1, \text{prior edges correct}) = \mathbb{P}(Y_{n_f} = 1 | \text{prior edges correct})$$

since correctly retaining true edge n_t does not modify adjacency sets and therefore does not affect the probability of removing false edge n_f. However, also by Lemma D.4:

$$\mathbb{P}(Y_{n_t} = 1 | Y_{n_f} = 1, \text{prior edges correct}) \geq \mathbb{P}(Y_{n_t} = 1 | \text{prior edges correct})$$

with strict inequality when n_t and n_f share a vertex, since correctly removing n_f reduces the adjacency set of n_t, which by Lemma D.3 strictly increases the probability of correctly retaining n_t.

Therefore, placing n_f before n_t (sequence O') achieves weakly better joint probability than the original ordering (sequence O), with strict improvement when n_t and n_f share a vertex. □

D.6 Proof of Lemma 4.1

Lemma 4.1 (Monotonicity of Perfect Recovery in Expert Accuracy)**.** *For a fixed partial skeleton $\mathcal{C}$ and true DAG $\mathcal{G}^*$, let $\Phi_{EL\text{-}G}(p_\psi)$ denote the perfect recovery probability when we sample an expert graph $\widehat{\mathcal{G}}$ from expert ψ with accuracy p_ψ, draw n samples from $\mathcal{G}^*$, and run EL-G. Then $\mathbb{E}[\Phi_{EL\text{-}G}(p_\psi)]$ increases monotonically with p_ψ, strictly increasing when $\mathcal{C}$ contains false edges adjacent to true edges.*

Proof. We establish that the expected perfect recovery probability $\mathbb{E}[\Phi_{\text{EL-G}}(p_\psi)]$ increases monotonically with expert accuracy p_ψ.

Setup and notation. For a fixed partial skeleton $\mathcal{C}$ and true DAG $\mathcal{G}^*$, consider running EL-G with expert accuracy p_ψ. The randomness in this process comes from three sources: (1) the expert's prediction $\widehat{\mathcal{G}}$ sampled according to accuracy p_ψ, (2) the finite-sample data sampled for use in conditional independence tests, and (3) the random shuffling used when EL-G generates the initial permutation of edges in $\mathcal{C}$ within each partition. Let Φ_{p_ψ} denote the probability that EL-G produces the true skeleton $\mathcal{S}^*$ after sampling from each of the three sources of randomness. The expectation $\mathbb{E}[\Phi_{p_\psi}]$ is taken over all three sources of randomness.

Goal. To show monotonicity, we must prove that for $p_{\psi_2} > p_{\psi_1}$, we have $\mathbb{E}[\Phi_{p_{\psi_2}}] \geq \mathbb{E}[\Phi_{p_{\psi_1}}]$. It suffices to show that $\Phi_{p_{\psi_2}}$ stochastically dominates $\Phi_{p_{\psi_1}}$.

Coupling and stochastic dominance. We employ a coupling argument. The following classical result provides our main tool (see (Lindvall, 1999) for a more abstract discussion of the result, and see Theorem 4.2.3. in (Levin and Peres, 2023) for a more direct formulation):

Theorem D.1 (Strassen's Coupling Theorem)**.** *The real random variable X stochastically dominates Y if and only if there exists a coupling $(\widehat{X}, \widehat{Y})$ of X and Y such that $\mathbb{P}[\widehat{X} \geq \widehat{Y}] = 1$. We refer to $(\widehat{X}, \widehat{Y})$ as a monotone coupling of X and Y.*

A coupling of random variables X and Y is a joint distribution $(\widehat{X}, \widehat{Y})$ where $\widehat{X}$ and $\widehat{Y}$ are two entirely different random variables whose marginal distributions coincide with the distributions of X and Y, respectively. More formally, a coupling is a probability measure on the product space whose projections onto each coordinate recover the original distributions. In simpler terms, $(\widehat{X}, \widehat{Y})$ is constructed such that $\widehat{X}$ has the same distribution as X, $\widehat{Y}$ has the same distribution as Y, but $\widehat{X}$ and $\widehat{Y}$ may be dependent.

Example: Bernoulli couplings. Consider Bernoulli random variables X and Y with $\mathbb{P}[X = 1] = q$ and $\mathbb{P}[Y = 1] = r$ where $0 \leq q < r \leq 1$.

- *Independent coupling*: We can construct $(\widehat{X}, \widehat{Y})$ where $\widehat{X}$ has the same distribution as X and $\widehat{Y}$ has the same distribution as Y and they are independent. This gives joint probabilities $\mathbb{P}[(\widehat{X}, \widehat{Y}) = (i, j)] = \mathbb{P}[\widehat{X} = i]\mathbb{P}[\widehat{Y} = j]$ for $i, j \in \{0, 1\}$.

- *Monotone coupling*: Alternatively, sample U uniformly from $[0, 1]$, and set $\widehat{X} = \mathbb{1}\{U \leq q\}$ and $\widehat{Y} = \mathbb{1}\{U \leq r\}$. Then $(\widehat{X}, \widehat{Y})$ is a coupling with $\mathbb{P}[\widehat{X} \leq \widehat{Y}] = 1$, where $\widehat{X}$ and $\widehat{Y}$ still follow the same Bernoulli distributions as X and Y respectively. This demonstrates that a single source of randomness can induce dependence while preserving marginals.

Proof strategy. Our proof proceeds in four steps:

1. Describe a hypothetical process for generating a monotone coupling $(\widehat{\Phi}_{p_{\psi_2}}, \widehat{\Phi}_{p_{\psi_1}})$ using shared randomness (analogous to the monotone Bernoulli coupling above). This construction assumes access to the ground truth skeleton $\mathcal{S}^*$ and is purely for theoretical analysis. Importantly, Strassen's theorem only requires showing that such a monotone coupling is *possible* to construct, not that we can construct it in practice with knowledge of only finite samples.

2. Verify that the marginal distributions coincide with the original distributions: $\widehat{\Phi}_{p_{\psi_2}} \overset{d}{=} \Phi_{p_{\psi_2}}$ and $\widehat{\Phi}_{p_{\psi_1}} \overset{d}{=} \Phi_{p_{\psi_1}}$.

3. Show the coupling is monotone: $\mathbb{P}[\widehat{\Phi}_{p_{\psi_2}} \geq \widehat{\Phi}_{p_{\psi_1}}] = 1$.

4. Conclude from Strassen's theorem that $\mathbb{E}[\Phi_{p_{\psi_2}}] \geq \mathbb{E}[\Phi_{p_{\psi_1}}]$.

D.6.1 Step 1: Describing the Process for Generating the Montone Coupling $(\widehat{\Phi}_{p_{\psi_2}}, \widehat{\Phi}_{p_{\psi_1}})$

We construct a coupling between two random variables $\widehat{\Phi}_{p_{\psi_1}}$ and $\widehat{\Phi}_{p_{\psi_2}}$ by describing a hypothetical generative process that uses shared randomness. In our case, both $\widehat{\Phi}_{p_{\psi_1}}$ and $\widehat{\Phi}_{p_{\psi_2}}$ are random variables taking values in $[0, 1]$ representing the probability of perfect skeleton recovery, and will be designed to have the same marginal distributions as $\Phi_{p_{\psi_1}}$ and $\Phi_{p_{\psi_2}}$ respectively.

Sources of shared randomness. We fix the expert accuracies p_{ψ_1} and p_{ψ_2}, but allow the expert predictions $\widehat{\mathcal{G}}$ to vary. Our coupling uses three sources of shared randomness:

1. Let c be the number of edges in $\mathcal{C}$. Suppose there are k true edges in $\mathcal{C}$ (edges in $\mathcal{S}^*$ and $\mathcal{C}$) and $c - k$ false edges (edges in $\mathcal{C}$ but not in $\mathcal{S}^*$). Let $L = [l_1, l_2, \ldots, l_c]$ be a random variable corresponding to a uniformly sampled permutation of k ones and $c - k$ zeros. That is, L contains exactly k entries equal to 1 and $c - k$ entries equal to 0, where the ordering is uniformly random among all such permutations.

2. A collection of independent uniform random variables $\mathcal{R} = \{r_1, r_2, \ldots, r_c\}$ where each $r_i \sim \text{Uniform}[0, 1]$. These will determine, for each position in L independently, whether the expert correctly classifies the corresponding edge.

3. Two uniform random permutations: π_T over the k true edges in $\mathcal{S}^*$, and π_F over the $c - k$ false edges not in $\mathcal{S}^*$. These determine the relative ordering within each partition.

Generating the expert prediction at accuracy p_{ψ_1}. We now describe how to sample an expert graph $\widehat{\mathcal{S}}^{(p_{\psi_1})}$ with accuracy p_{ψ_1} using the shared randomness $(L, \mathcal{R}, \pi_T, \pi_F)$. For each position i in L, the expert independently classifies the corresponding edge correctly with probability p_{ψ_1}:

- If $l_i = 1$ (corresponds to a true edge in $\mathcal{S}^*$): the expert correctly predicts this edge exists if $r_i \leq p_{\psi_1}$, otherwise incorrectly predicts it does not exist.

- If $l_i = 0$ (corresponds to a false edge not in $\mathcal{S}^*$): the expert correctly predicts this edge does not exist if $r_i \leq p_{\psi_1}$, otherwise incorrectly predicts it exists.

Constructing the edge ordering $\widehat{\mathsf{O}}^{p_{\psi_1}}$. Given the expert prediction $\widehat{\mathcal{S}}^{(p_{\psi_1})}$, we construct the edge ordering as specified by EL-G (Subroutine 3). Initialize two empty lists $B_1 = [], B_2 = []$. For each position i in L (going in the order specified by L)

- If $l_i = 1$ (true edge) and $r_i \leq p_{\psi_1}$ (correctly classified): add position i to the end of B_2.
- If $l_i = 1$ (true edge) and $r_i > p_{\psi_1}$ (incorrectly classified): add position i to the end of B_1.
- If $l_i = 0$ (false edge) and $r_i \leq p_{\psi_1}$ (correctly classified): add position i to the end of B_1.
- If $l_i = 0$ (false edge) and $r_i > p_{\psi_1}$ (incorrectly classified): add position i to the end of B_2.

Concatenate the buckets: $L_F^{p_{\psi_1}} = B_1 + B_2$. Within $L_F^{p_{\psi_1}}$, assign relative ordering among true edges using π_T and among false edges using π_F to obtain the final edge ordering $\widehat{\mathsf{O}}^{p_{\psi_1}}$.

Computing $\widehat{\Phi}_{p_{\psi_1}}$. Given the edge ordering $\widehat{\mathsf{O}}^{p_{\psi_1}}$, let $\widehat{\Phi}_{p_{\psi_1}}$ denote the probability of perfect skeleton recovery by EL (Subroutine 1) if were to randomly draw n finite-samples of the variables V from the DGP $\mathcal{G}^*$.

Generating $\widehat{\Phi}_{p_{\psi_2}}$ using the same randomness. We follow the exact same procedure as above, crucially reusing the same shared randomness $(L, \mathcal{R}, \pi_T, \pi_F)$. The only difference is that we use accuracy p_{ψ_2} instead of p_{ψ_1} when determining expert classifications, and redraw a new batch of finite-sample data. This yields a potentially different expert graph $\widehat{\mathcal{S}}^{(p_{\psi_2})}$, a potentially different edge ordering $\widehat{\mathsf{O}}^{p_{\psi_2}}$, and a potentially different recovery probability $\widehat{\Phi}_{p_{\psi_2}}$.

By this coupling procedure, we generate the joint random variable $(\widehat{\Phi}_{p_{\psi_1}}, \widehat{\Phi}_{p_{\psi_2}})$.

D.6.2 Step 2: Showing the Marginals of the Two Variables Coincide with Original Distributions

Our goal is to verify that $\widehat{\Phi}_{p_{\psi_1}} \overset{d}{=} \Phi_{p_{\psi_1}}$ and $\widehat{\Phi}_{p_{\psi_2}} \overset{d}{=} \Phi_{p_{\psi_2}}$. We focus on showing $\widehat{\Phi}_{p_{\psi_1}} \overset{d}{=} \Phi_{p_{\psi_1}}$; the argument for p_{ψ_2} follows identically by symmetry.

Reduction to orderings. Let $\mathsf{O}^{p_{\psi_1}}$ denote the random edge ordering generated by Subroutine 3 using expert accuracy p_{ψ_1}, and let $\widehat{\mathsf{O}}^{p_{\psi_1}}$ denote the random edge ordering generated in our coupling procedure (Step 1) using accuracy p_{ψ_1}. Given any fixed edge ordering, the probability of perfect skeleton recovery when randomly drawing n finite samples from $\mathcal{G}^*$ is deterministically fixed. Therefore, the distributions of $\Phi_{p_{\psi_1}}$ and $\widehat{\Phi}_{p_{\psi_1}}$ are determined entirely by the distributions of $\mathsf{O}^{p_{\psi_1}}$ and $\widehat{\mathsf{O}}^{p_{\psi_1}}$ respectively. To show $\widehat{\Phi}_{p_{\psi_1}} \overset{d}{=} \Phi_{p_{\psi_1}}$, it suffices to show that $\widehat{\mathsf{O}}^{p_{\psi_1}} \overset{d}{=} \mathsf{O}^{p_{\psi_1}}$.

Decomposition of orderings. Any edge ordering O over all c edges can be uniquely decomposed into three components:

1. π_T: the relative ordering (permutation) among the k true edges
2. π_F: the relative ordering (permutation) among the $c - k$ false edges
3. π_{TF}: the relative placement of true edges versus false edges (which false/true edges come before which other true/false edges)

Therefore, the distribution over edge orderings is uniquely determined by the joint distribution over (π_T, π_F, π_{TF}). We will show that this joint distribution is identical for both $\mathsf{O}^{p_{\psi_1}}$ and $\widehat{\mathsf{O}}^{p_{\psi_1}}$.

Analysis of the coupling procedure. In our coupling construction (Step 1), we explicitly sampled π_T and π_F uniformly and independently. The relative placement π_{TF} is then determined by: for each position i in L, whether l_i is correctly classified (with probability p_{ψ_1}) determines whether that edge goes to B_1 or B_2, and then π_{TF} corresponds to the partition structure (B_1, B_2). Since each edge's classification is independent and depends only on its true label and p_{ψ_1}, we have:

- $\widehat{\pi}_T$ is uniform over all permutations of k true edges

- $\widehat{\pi}_F$ is uniform over all permutations of $\binom{d}{2} - k$ false edges
- $\widehat{\pi}_{TF}$ has distribution determined by p_{ψ_1} (probability of correct classification)
- $\widehat{\pi}_T, \widehat{\pi}_F, \widehat{\pi}_{TF}$ are mutually independent

Analysis of Subroutine 3. We now show that $\mathsf{O}^{p_{\psi_1}}$ has the same distributional structure. By construction of Subroutine 3:

- Edges in $\mathcal{C}$ are initially shuffled uniformly at random
- Each edge is independently classified as being in $\widehat{\mathcal{S}}$ or not with probability p_{ψ_1} of correct classification
- Edges are partitioned into $B_1 = \mathcal{C} \setminus \widehat{\mathcal{S}}$ and $B_2 = \mathcal{C} \cap \widehat{\mathcal{S}}$, preserving their random ordering within each bucket

To show π_T is uniform: Start with an initial uniform random permutation π_T^{init} of all k true edges. Some subset of size $m \sim \text{Binomial}(k, 1 - p_{\psi_1})$ are misclassified and placed in B_1, while the remaining $k - m$ are placed in B_2. For any fixed m and any fixed choice of which m edges are misclassified, this operation defines a bijection from π_T^{init} to the resulting permutation: given the final permutation and knowing which edges went to which bucket, we can uniquely recover π_T^{init}, and vice versa. We note that as bijections preserve uniformity, the marginal distribution of π_T (after marginalizing over m and the choice of which edges) remains uniform. By the same argument, π_F is uniform.

For independence: The distribution of π_{TF} is determined solely by which edges are correctly classified (controlled by p_{ψ_1}). For any fixed realization of π_{TF} (i.e., fixed partition (B_1, B_2)), the above uniformity argument shows that π_T and π_F remain uniform. Therefore (π_T, π_F) are independent of π_{TF}.

Conclusion. Both $\widehat{\mathsf{O}}^{p_{\psi_1}}$ and $\mathsf{O}^{p_{\psi_1}}$ decompose into (π_T, π_F, π_{TF}) where each component has identical marginal distributions and the same independence structure. Therefore $\widehat{\mathsf{O}}^{p_{\psi_1}} \overset{d}{=} \mathsf{O}^{p_{\psi_1}}$, which implies $\widehat{\Phi}_{p_{\psi_1}} \overset{d}{=} \Phi_{p_{\psi_1}}$. By the same argument, $\widehat{\Phi}_{p_{\psi_2}} \overset{d}{=} \Phi_{p_{\psi_2}}$.

D.6.3 Step 3: Showing that the Coupling is Monotone.

Our goal is to show that $\mathbb{P}[\widehat{\Phi}_{p_{\psi_2}} \geq \widehat{\Phi}_{p_{\psi_1}}] = 1$. We establish this by showing that the edge ordering $\widehat{\mathsf{O}}^{p_{\psi_2}}$ can be obtained from $\widehat{\mathsf{O}}^{p_{\psi_1}}$ through a sequence of beneficial swaps.

Structure of the two orderings. Recall from Step 1 that both $\widehat{\mathsf{O}}^{p_{\psi_1}}$ and $\widehat{\mathsf{O}}^{p_{\psi_2}}$ are generated using the same shared randomness $(L, \mathcal{R}, \pi_T, \pi_F)$. Crucially, the permutations π_T (ordering among true edges) and π_F (ordering among false edges) are identical in both orderings. The only difference lies in the relative placement π_{TF} of true edges versus false edges, which is determined by which edges are correctly classified.

More edges correctly classified at higher accuracy. Since $p_{\psi_1} < p_{\psi_2}$, for each position i in L:

- If edge i is correctly classified under accuracy p_{ψ_1} (i.e., $r_i \leq p_{\psi_1}$), then it is also correctly classified under accuracy p_{ψ_2} (since $r_i \leq p_{\psi_1} < p_{\psi_2}$)
- Additional edges may be correctly classified under p_{ψ_2} that were misclassified under p_{ψ_1} (those with $p_{\psi_1} < r_i \leq p_{\psi_2}$)

This means:

- True edges that were correctly placed in B_2 in $L_F^{p_{\psi_1}}$ remain in B_2 in $L_F^{p_{\psi_2}}$
- Some true edges that were incorrectly placed in B_1 in $L_F^{p_{\psi_1}}$ may move to B_2 in $L_F^{p_{\psi_2}}$ (moving rightward)
- False edges that were correctly placed in B_1 in $L_F^{p_{\psi_1}}$ remain in B_1 in $L_F^{p_{\psi_2}}$
- Some false edges that were incorrectly placed in B_2 in $L_F^{p_{\psi_1}}$ may move to B_1 in $L_F^{p_{\psi_2}}$ (moving leftward)

Characterizing the transformation via inversions. To formalize how $\widehat{\mathsf{O}}^{p_{\psi_2}}$ relates to $\widehat{\mathsf{O}}^{p_{\psi_1}}$, we introduce a numerical ordering. Assign integers to edges as follows:

- Assign $\{1, 2, \ldots, c-k\}$ to the false edges according to their fixed relative order π_F
- Assign $\{c-k+1, \ldots, \binom{d}{2}\}$ to the true edges according to their fixed relative order π_T

Under this assignment, every false edge has a smaller numerical label than every true edge. An **inversion** is any pair of edges that appears out of numerical order—specifically, a true edge appearing before a false edge in the ordering. Since π_T and π_F are fixed, edges of the same type (both true or both false) never form inversions relative to each other.

The transformation from $\widehat{\mathsf{O}}^{p_{\psi_1}}$ to $\widehat{\mathsf{O}}^{p_{\psi_2}}$ only moves true edges rightward and false edges leftward. This process cannot create new inversions: if a true edge precedes a false edge in $\widehat{\mathsf{O}}^{p_{\psi_2}}$, that pair must have been in the same order in $\widehat{\mathsf{O}}^{p_{\psi_1}}$. Therefore, the set of inversions in $\widehat{\mathsf{O}}^{p_{\psi_2}}$ is a subset of the inversions in $\widehat{\mathsf{O}}^{p_{\psi_1}}$.

Weak Bruhat order and reachability. The weak Bruhat order on permutations provides a formal framework for this relationship. A standard result ((Yessenov, 2005), Proposition 2.2) states:

Proposition D.2. *For permutations v and w, $v \leq w$ in weak Bruhat order if and only if* $\mathrm{Inv}(v) \subseteq \mathrm{Inv}(w)$, *where* $\mathrm{Inv}(\cdot)$ *denotes the set of inversions.*

Equivalently, v can be obtained from w by a sequence of adjacent transpositions (w_i, w_{i+1}) where $w_i > w_{i+1}$ in the numerical ordering. In our setting, moving a false edge leftward past an adjacent true edge (moving w_{i+1} left of w_i) is precisely such an inversion-reducing swap, since false edges have smaller numerical labels than true edges.

Since $\mathrm{Inv}(\widehat{\mathsf{O}}^{p_{\psi_2}}) \subseteq \mathrm{Inv}(\widehat{\mathsf{O}}^{p_{\psi_1}})$, it follows that $\widehat{\mathsf{O}}^{p_{\psi_2}}$ can be obtained from $\widehat{\mathsf{O}}^{p_{\psi_1}}$ through a sequence of adjacent swaps that move false edges leftward past true edges.

Monotonicity via beneficial swaps. By Lemma D.5, each such swap (placing a false edge before a true edge) weakly improves the probability of perfect skeleton recovery, with strict improvement when the swapped edges share a vertex. Since $\widehat{\mathsf{O}}^{p_{\psi_2}}$ is reachable from $\widehat{\mathsf{O}}^{p_{\psi_1}}$ through a sequence of such beneficial swaps, we have $\widehat{\Phi}_{p_{\psi_1}} \leq \widehat{\Phi}_{p_{\psi_2}}$, establishing that $\mathbb{P}[\widehat{\Phi}_{p_{\psi_2}} \geq \widehat{\Phi}_{p_{\psi_1}}] = 1$.

D.6.4 Step 4: Conclusion.

It follows from Strassen's Coupling Theorem that $\mathbb{E}[\Phi_{p_{\psi_2}}] \geq \mathbb{E}[\Phi_{p_{\psi_1}}]$, with strict inequality when $\mathcal{C}$ contains at least one true edge adjacent to a false edge. We note that when $\mathcal{C}$ contains false edges adjacent to true edges, there are at least one pair of orderings $\widehat{\mathsf{O}}^{p_{\psi_1}}, \widehat{\mathsf{O}}^{p_{\psi_2}}$ that can be created through the process outlined in Step 1 such that a false edge adjacent to a true edge is swapped to be earlier in the sequence. In that case we have $\widehat{\Phi}_{p_{\psi_1}} < \widehat{\Phi}_{p_{\psi_2}}$, which implies by the Coupling Theorem that $\mathbb{E}[\Phi_{p_{\psi_2}}] > \mathbb{E}[\Phi_{p_{\psi_1}}]$.

□

D.7 Proof of Lemma D.6

Lemma D.6 (Monotonicity of Perfect Recovery in Expert Accuracy Under Asymmetric Accuracy)**.** *Suppose the expert ψ acts as a binary but potentially asymmetric channel: for any edge $n_{i,j}$, if $n_{i,j}$ does not exist in the true skeleton S^* (a false edge) the expert independently predicts whether it exists in S^* with accuracy p^f_ψ. Similarly, if $n_{i,j}$ does exist in S^* (true edge), then the expert independently predicts whether it exists in S^* with accuracy p^t_ψ. For a fixed partial skeleton $\mathcal{C}$ and true DAG $\mathcal{G}^*$, let $\Phi_{\text{EL-G}}(p^t_\psi, p^f_\psi)$ denote the perfect recovery probability when we sample an expert graph $\widehat{\mathcal{G}}$ from expert ψ with accuracies p^t_ψ, p^f_ψ, draw n samples from $\mathcal{G}^*$, and run EL-G. Then $\mathbb{E}[\Phi_{\text{EL-G}}(p^t_\psi, p^f_\psi)]$ increases monotonically in both p^t_ψ and p^f_ψ, strictly increasing when $\mathcal{C}$ contains false edges adjacent to true edges.*

Proof. Suppose we consider two different experts ψ_1, ψ_2 with a fixed accuracy for true edges $p^t_{\psi_1} = p^t_{\psi_2}$, but differing accuracy for false edges $p^f_{\psi_1} < p^f_{\psi_2}$. Similar to the proof of Lemma 4.1, we again use a coupling argument

where both experts observe the same true skeleton S^* and use identical randomness for edge classification. As $p^t_{\psi_1} = p^t_{\psi_1}$, ψ_1, ψ_2 classify every true edge identically. However, as $p^f_{\psi_1} < p^f_{\psi_2}$, every error made by ψ_2 in classifying false edges is made by ψ_1, while some errors made by ψ_1 in classifying false edges are not made by ψ_2. This leads to ψ_2's edge ordering having equal or fewer inversions (true edges incorrectly placed before false edges). One can then follow the the rest of the argument made in Lemma 4.1 to get that $\Phi_{\text{EL-G}}(p^t_{\psi_2}, p^f_{\psi_2})$ stochastically dominates $\Phi_{\text{EL-G}}(p^t_{\psi_1}, p^f_{\psi_1})$, yielding monotonicity in p^f_ψ in expectation. WLOG, one can follow the same argument to find monotonicity in p^t_ψ in expectation when p^f_ψ is fixed.

We validate this theoretical result experimentally (see Appendix I.7). □

D.8 Proof of Lemma D.7

Lemma D.7. *Accuracy* $\mathbb{P}(Y_{n_{i,j}} = 1)$ *is constant for all possible orderings* $\mathsf{L}_1, \mathsf{L}_2, \ldots$ *of* $CIT^k_{\text{adj}_{-j}(\mathcal{C}, x_i)}$.

Proof. Note that in EP (Subroutine 2), edge $n_{i,j}$ is removed when a subset s is found to render x_i, x_j independent, i.e. $x_i \perp\!\!\!\perp x_j | s$. This implies that for $n_{i,j}$ to be removed at least one CI test must return independent, and if no CI tests return independent $n_{i,j}$ will not be removed. Whether at least one CI test returns independent depends only on which CI tests are run (i.e., what the adjacent set $\text{adj}_{-j}(\mathcal{C}, x_i)$ is), implying that the order L in which the CI tests are run is irrelevant to whether the edge is removed or not. This implies that the probability of accuracy $\mathbb{P}(Y_{n_{i,j}} = 1)$ is also independent of ordering. □

D.9 Proof of Lemma D.8

Lemma D.8. *Under the assumptions of Definition E.1, if* $\nexists$ *a size* k *subset of* $\text{adj}_{-j}(\mathcal{C}, x_i)$ s_i *such that* $x_i \perp\!\!\!\perp x_j | s_i$, *then any pair of orderings* L, L' *achieves the same* $\mathbb{E}[t_{e_{i,j}}]$, *where* $t_{e_{i,j}}$ *is the number of tests conducted by EP using either* L *or* L'.

Proof. Suppose $\nexists$ a size k subset s_i of $\text{adj}_{-j}(\mathcal{C}, x_i)$ such that $x_i \perp\!\!\!\perp x_j | s_i$. Then for all $\text{CIT}(x_i, x_j \mid s_u) \in \text{CIT}^k_{\text{adj}_{-j}(\mathcal{C}, x_i)}$, the subset s_u is not a d-separating set of x_i, x_j, meaning all tests should return dependent under an oracle.

The runtime is determined by when the first test returns independent (or when all tests have been run). By Definition E.1, since no d-separating sets exist, each CIT falsely returns independent with the same false negative rate β, and the outcomes of these tests are mutually independent. Therefore, the timing of when a test will return independent is the same no matter the order of the tests, implying that $\mathbb{E}[t_{e_{i,j}}]$ remains constant for all orderings of tests. □

D.10 Proof of Lemma D.9

Lemma D.9. *Under the assumptions of Definition E.1, given a sequence of CITs* L *for edge* $n_{i,j}$, *for any pair of adjacent CITs consisting of a test on a non-d-separating set* s_1 *followed by a test on a d-separating set* s_2, *the sequence* L' *generated by swapping the pair to place the d-separating test* s_2 *first achieves strictly better runtime, i.e.,* $\mathbb{E}[t^{\mathsf{L}'}_{e_{i,j}}] < \mathbb{E}[t^{\mathsf{L}}_{e_{i,j}}]$, *where* $t^{\mathsf{L}}_{e_{i,j}}$ *is the number of tests conducted by EP under ordering* L.

Proof. We say a CIT$= I$ if it returns independent, and CIT$= N$ if it returns dependent (not independent). For edge $n_{i,j}$, let CIT^{d-sep}_i denote a CIT at position i in the sequence that tests a d-separating set of x_i, x_j, and let CIT^{non}_i denote a CIT at position i that tests a non-d-separating set of x_i, x_j.

Consider a pair of orderings L, L' that differ in only two positions, where the tests are swapped: ordering $\mathsf{L} = \ldots, \text{CIT}^{non}_i, \text{CIT}^{d-sep}_{i+1}, \ldots$, while ordering $\mathsf{L}' = \ldots, \text{CIT}^{d-sep}_i, \text{CIT}^{non}_{i+1}, \ldots$. Under the mutual independence

assumption (Definition E.1), we can simplify the difference between the expected runtimes of the orderings as:

$$\begin{aligned}\mathbb{E}[t^{\mathsf{L}'}_{e_{i,j}}] - \mathbb{E}[t^{\mathsf{L}}_{e_{i,j}}] = &\left(\prod_{j=1}^{i-1} \mathbb{P}(\text{CIT}_j = N)\right)\Big[\mathbb{P}(\text{CIT}_i^{d-sep} = I)\cdot i \\ &+ \mathbb{P}(\text{CIT}_i^{d-sep} = N)\mathbb{P}(\text{CIT}_{i+1}^{non} = I)\cdot(i+1)\Big] \\ &- \left(\prod_{j=1}^{i-1} \mathbb{P}(\text{CIT}_j = N)\right)\Big[\mathbb{P}(\text{CIT}_i^{non} = I)\cdot i \\ &+ \mathbb{P}(\text{CIT}_i^{non} = N)\mathbb{P}(\text{CIT}_{i+1}^{d-sep} = I)\cdot(i+1)\Big] \end{aligned} \tag{1}$$

Dividing both sides by the positive value $\left(\prod_{j=1}^{i-1} \mathbb{P}(\text{CIT}_j = N)\right)$ yields the following on the RHS:

$$\begin{aligned}= &\Big[\mathbb{P}(\text{CIT}_i^{d-sep} = I)\cdot i + \mathbb{P}(\text{CIT}_i^{d-sep} = N)\mathbb{P}(\text{CIT}_{i+1}^{non} = I)\cdot(i+1)\Big] \\ &- \Big[\mathbb{P}(\text{CIT}_i^{non} = I)\cdot i + \mathbb{P}(\text{CIT}_i^{non} = N)\mathbb{P}(\text{CIT}_{i+1}^{d-sep} = I)\cdot(i+1)\Big]. \end{aligned} \tag{1}$$

Filling in the probabilities with the correct false positive and false negative rates (where α is the false positive rate and β is the false negative rate) yields:

$$\begin{aligned}&= [(1-\alpha)\cdot i + \alpha\beta(i+1)] - [\beta\cdot i + (1-\beta)(1-\alpha)\cdot(i+1)] \\ &= [(1-\alpha)\cdot i + \alpha\beta(i+1)] - [\beta\cdot i + (1-\beta-\alpha+\alpha\beta)\cdot(i+1)] \end{aligned} \tag{1}$$

We note that both terms in the above equation can be rewritten as weighted sums of $i, i+1$, with the weights adding up to $1-\alpha+\alpha\beta$. Under the assumption that $1-\alpha > \beta$, this implies that the weight on i is higher in the first term, which implies the weight on $i+1$ is higher in the second term. As $i < i+1$, under the rearrangement inequality this implies that the sum is negative, and therefore $\mathbb{E}[t^{\mathsf{L}'}_{e_{i,j}}] - \mathbb{E}[t^{\mathsf{L}}_{e_{i,j}}] < 0$. □

D.11 Proof of Lemma D.10

Lemma D.10 (Monotonicity of EP Runtime in Expert Accuracy)**.** *Under Assumption E.1, let $T_{EP\text{-}G}(p_{d\text{-}sep})$ denote the number of tests executed by EP-Guess (Subroutine 4) when testing edge $n_{i,j}$ at conditioning set size k with expert ψ having d-separation accuracy $p_{d\text{-}sep}$. Then:*

(a) $\mathbb{E}[T_{EP\text{-}G}(p_{d\text{-}sep})]$ decreases monotonically with $p_{d\text{-}sep}$, strictly decreasing when $[A]_k$ contains both d-separating and non-d-separating sets (where $A = \text{adj}_{-j}(\mathcal{C}, x_i)$)

(b) When $p_{d\text{-}sep} \geq 0.5$, $\mathbb{E}[T_{EP\text{-}G}(p_{d\text{-}sep})] \leq \mathbb{E}[T_{random}]$ where T_{random} denotes runtime under random ordering

Proof. We establish that the expected runtime $\mathbb{E}[T_{\text{EP-G}}(p_{\text{d-sep}})]$ decreases monotonically with expert d-separation accuracy $p_{\text{d-sep}}$.

Setup and notation. For a fixed partial skeleton $\mathcal{C}$, true DAG $\mathcal{G}^*$, edge $n_{i,j} \in \mathcal{C}$, and conditioning set size k, consider running EP-Guess with expert accuracy $p_{\text{d-sep}}$. EP-Guess tests conditional independence $\text{CIT}(x_i, x_j \mid W)$ for all $W \in [A]_k$ where $A = \text{adj}_{-j}(\mathcal{C}, x_i)$, terminating when the first test returns independence. The randomness in this process comes from three sources: (1) the expert's prediction $\widehat{\mathcal{G}}$ sampled according to accuracy $p_{\text{d-sep}}$, (2) the finite-sample data D sampled for use in conditional independence tests, and (3) the random shuffling used when EP-Guess generates the initial permutation of conditioning sets within each partition. Let $T_{p_{\text{d-sep}}}$ denote the number of tests executed by EP before termination after sampling from each of the three sources of randomness. The expectation $\mathbb{E}[T_{p_{\text{d-sep}}}]$ is taken over all three sources of randomness.

Goal. To show monotonicity, we must prove that for $p_{\text{d-sep}_2} > p_{\text{d-sep}_1}$, we have $\mathbb{E}[T_{p_{\text{d-sep}_1}}] \geq \mathbb{E}[T_{p_{\text{d-sep}_2}}]$. By Strassen's theorem, it suffices to show that $T_{p_{\text{d-sep}_1}}$ stochastically dominates $T_{p_{\text{d-sep}_2}}$.

Coupling and stochastic dominance. We employ a coupling argument. The following classical result provides our main tool (see (Lindvall, 1999) for a more abstract discussion of the result, and see Theorem 4.2.3. in (Levin and Peres, 2023) for a more direct formulation):

Theorem D.3 (Strassen's Coupling Theorem)**.** *The real random variable X stochastically dominates Y if and only if there exists a coupling $(\widehat{X}, \widehat{Y})$ of X and Y such that $\mathbb{P}[\widehat{X} \geq \widehat{Y}] = 1$. We refer to $(\widehat{X}, \widehat{Y})$ as a monotone coupling of X and Y.*

A coupling of random variables X and Y is a joint distribution $(\widehat{X}, \widehat{Y})$ where $\widehat{X}$ and $\widehat{Y}$ are two entirely different random variables whose marginal distributions coincide with the distributions of X and Y, respectively. More formally, a coupling is a probability measure on the product space whose projections onto each coordinate recover the original distributions. In simpler terms, $(\widehat{X}, \widehat{Y})$ is constructed such that $\widehat{X}$ has the same distribution as X, $\widehat{Y}$ has the same distribution as Y, but $\widehat{X}$ and $\widehat{Y}$ may be dependent.

Example: Bernoulli couplings. Consider Bernoulli random variables X and Y with $\mathbb{P}[X = 1] = q$ and $\mathbb{P}[Y = 1] = r$ where $0 \leq q < r \leq 1$.

- *Independent coupling*: We can construct $(\widehat{X}, \widehat{Y})$ where $\widehat{X}$ has the same distribution as X and $\widehat{Y}$ has the same distribution as Y and they are independent. This gives joint probabilities $\mathbb{P}[(\widehat{X}, \widehat{Y}) = (i, j)] = \mathbb{P}[\widehat{X} = i]\mathbb{P}[\widehat{Y} = j]$ for $i, j \in \{0, 1\}$.
- *Monotone coupling*: Alternatively, sample U uniformly from $[0, 1]$, and set $\widehat{X} = \mathbb{1}\{U \leq q\}$ and $\widehat{Y} = \mathbb{1}\{U \leq r\}$. Then $(\widehat{X}, \widehat{Y})$ is a coupling with $\mathbb{P}[\widehat{X} \leq \widehat{Y}] = 1$, where $\widehat{X}$ and $\widehat{Y}$ still follow the same Bernoulli distributions as X and Y respectively. This demonstrates that a single source of randomness can induce dependence while preserving marginals.

Proof strategy. Our proof proceeds in four steps:

1. Describe a hypothetical process for generating a monotone coupling $(\widehat{T}_{p_{\text{d-sep}_1}}, \widehat{T}_{p_{\text{d-sep}_2}})$ using shared randomness (analogous to the monotone Bernoulli coupling above). This construction assumes access to the ground truth DAG $\mathcal{G}^*$ and is purely for theoretical analysis. Importantly, Strassen's theorem only requires showing that such a monotone coupling is *possible* to construct, not that we can construct it in practice with knowledge of only finite samples.
2. Verify that the marginal distributions coincide with the original distributions: $\widehat{T}_{p_{\text{d-sep}_1}} \stackrel{d}{=} T_{p_{\text{d-sep}_1}}$ and $\widehat{T}_{p_{\text{d-sep}_2}} \stackrel{d}{=} T_{p_{\text{d-sep}_2}}$.
3. Show the coupling is monotone: $\mathbb{P}[\widehat{T}_{p_{\text{d-sep}_1}} \geq \widehat{T}_{p_{\text{d-sep}_2}}] = 1$.
4. Conclude from Strassen's theorem that $\mathbb{E}[T_{p_{\text{d-sep}_1}}] \geq \mathbb{E}[T_{p_{\text{d-sep}_2}}]$.

D.11.1 Step 1: Describing the Process for Generating the Monotone Coupling $(\widehat{T}_{p_{\text{d-sep}_1}}, \widehat{T}_{p_{\text{d-sep}_2}})$

We construct a coupling between two random variables $\widehat{T}_{p_{\text{d-sep}_1}}$ and $\widehat{T}_{p_{\text{d-sep}_2}}$ by describing a hypothetical generative process that uses shared randomness. Both random variables take values in positive integers representing the number of tests executed before EP terminates, and will be designed to have the same marginal distributions as $T_{p_{\text{d-sep}_1}}$ and $T_{p_{\text{d-sep}_2}}$ respectively.

Sources of shared randomness. We fix the expert accuracies $p_{\text{d-sep}_1}$ and $p_{\text{d-sep}_2}$, the conditioning set size k, and allow the expert predictions $\widehat{\mathcal{G}}$ to vary. For edge $n_{i,j}$ with adjacency set $A = \text{adj}_{-j}(\mathcal{C}, x_i)$, EP-Guess considers all size-k subsets of A, i.e., all $W \in [A]_k$. Let $c = |[A]_k|$ denote the total number of such conditioning sets. Our coupling uses three sources of shared randomness:

1. We partition the size-k conditioning sets in $[A]_k$ based on true d-separation in $\mathcal{G}^*$. Suppose there are m sets in $[A]_k$ that d-separate x_i, x_j in $\mathcal{G}^*$ (d-separating sets) and $c - m$ sets that do not (non-d-separating sets). Let $L = [l_1, l_2, \ldots, l_c]$ be a random variable corresponding to a uniformly sampled permutation of m ones and $c - m$ zeros. That is, L contains exactly m entries equal to 1 (indicating d-separating sets) and $c - m$ entries equal to 0 (indicating non-d-separating sets), where the ordering is uniformly random among all such permutations.

2. A collection of independent uniform random variables $\mathcal{R} = \{r_1, r_2, \ldots, r_c\}$ where each $r_i \sim \text{Uniform}[0,1]$. These will determine, for each position in L independently, whether the expert correctly classifies the corresponding conditioning set.

3. Two uniform random permutations: $\pi_{\text{d-sep}}$ over the m d-separating sets in $[A]_k$, and $\pi_{\text{non-d-sep}}$ over the $c-m$ non-d-separating sets in $[A]_k$. These determine the relative ordering within each partition.

Generating the expert prediction at accuracy $p_{\text{d-sep}_1}$. We now describe how to sample an expert graph $\widehat{\mathcal{G}}^{(p_{\text{d-sep}_1})}$ with d-separation accuracy $p_{\text{d-sep}_1}$ using the shared randomness $(L, \mathcal{R}, \pi_{\text{d-sep}}, \pi_{\text{non-d-sep}})$. For each position i in L, the expert independently classifies the corresponding conditioning set correctly with probability $p_{\text{d-sep}_1}$:

- If $l_i = 1$ (corresponds to a true d-separating set in $\mathcal{G}^*$): the expert correctly predicts this set d-separates x_i, x_j if $r_i \le p_{\text{d-sep}_1}$, otherwise incorrectly predicts it does not d-separate.
- If $l_i = 0$ (corresponds to a non-d-separating set): the expert correctly predicts this set does not d-separate x_i, x_j if $r_i \le p_{\text{d-sep}_1}$, otherwise incorrectly predicts it d-separates.

Constructing the conditioning set ordering $\widehat{\mathsf{L}}^{p_{\text{d-sep}_1}}$. Given the expert prediction $\widehat{\mathcal{G}}^{(p_{\text{d-sep}_1})}$, we construct the conditioning set ordering as specified by EP-Guess (Subroutine 4). Initialize two empty lists $B_1 = [], B_2 = []$. For each position i in L (going in the order specified by L):

- If $l_i = 1$ (d-separating set) and $r_i \le p_{\text{d-sep}_1}$ (correctly classified): add position i to the end of B_1 (predicted d-separating sets tested first).
- If $l_i = 1$ (d-separating set) and $r_i > p_{\text{d-sep}_1}$ (incorrectly classified): add position i to the end of B_2.
- If $l_i = 0$ (non-d-separating set) and $r_i \le p_{\text{d-sep}_1}$ (correctly classified): add position i to the end of B_2.
- If $l_i = 0$ (non-d-separating set) and $r_i > p_{\text{d-sep}_1}$ (incorrectly classified): add position i to the end of B_1.

Concatenate the buckets: $L_F^{p_{\text{d-sep}_1}} = B_1 + B_2$. Within $L_F^{p_{\text{d-sep}_1}}$, assign relative ordering among d-separating sets using $\pi_{\text{d-sep}}$ and among non-d-separating sets using $\pi_{\text{non-d-sep}}$ to obtain the final conditioning set ordering $\widehat{\mathsf{L}}^{p_{\text{d-sep}_1}}$.

Computing $\widehat{T}_{p_{\text{d-sep}_1}}$. Given the conditioning set ordering $\widehat{\mathsf{L}}^{p_{\text{d-sep}_1}}$, let $\widehat{T}_{p_{\text{d-sep}_1}}$ denote the expected number of tests executed by EP (Subroutine 2) before termination if we were to randomly draw n finite samples from the DGP $\mathcal{G}^*$ and run tests according to $\widehat{\mathsf{L}}^{p_{\text{d-sep}_1}}$.

Generating $\widehat{T}_{p_{\text{d-sep}_2}}$ using the same randomness. We follow the exact same procedure as above, crucially reusing the same shared randomness $(L, \mathcal{R}, \pi_{\text{d-sep}}, \pi_{\text{non-d-sep}})$. The only difference is that we use accuracy $p_{\text{d-sep}_2}$ instead of $p_{\text{d-sep}_1}$ when determining expert classifications. This yields a potentially different expert graph $\widehat{\mathcal{G}}^{(p_{\text{d-sep}_2})}$, a potentially different conditioning set ordering $\widehat{\mathsf{L}}^{p_{\text{d-sep}_2}}$, and a potentially different expected runtime $\widehat{T}_{p_{\text{d-sep}_2}}$.

By this coupling procedure, we generate the joint random variable $(\widehat{T}_{p_{\text{d-sep}_1}}, \widehat{T}_{p_{\text{d-sep}_2}})$.

D.11.2 Step 2: Showing the Marginals of the Two Variables Coincide with Original Distributions

Our goal is to verify that $\widehat{T}_{p_{\text{d-sep}_1}} \overset{d}{=} T_{p_{\text{d-sep}_1}}$ and $\widehat{T}_{p_{\text{d-sep}_2}} \overset{d}{=} T_{p_{\text{d-sep}_2}}$. We focus on showing $\widehat{T}_{p_{\text{d-sep}_1}} \overset{d}{=} T_{p_{\text{d-sep}_1}}$; the argument for $p_{\text{d-sep}_2}$ follows identically by symmetry.

Reduction to orderings. Let $\mathsf{L}^{p_{\text{d-sep}_1}}$ denote the random conditioning set ordering generated by Subroutine 4 using expert accuracy $p_{\text{d-sep}_1}$, and let $\widehat{\mathsf{L}}^{p_{\text{d-sep}_1}}$ denote the random conditioning set ordering generated in our coupling procedure (Step 1) using accuracy $p_{\text{d-sep}_1}$. Given any fixed conditioning set ordering, the expected number of tests executed when randomly drawing n finite samples from $\mathcal{G}^*$ is deterministically fixed. Therefore, the distributions of $T_{p_{\text{d-sep}_1}}$ and $\widehat{T}_{p_{\text{d-sep}_1}}$ are determined entirely by the distributions of $\mathsf{L}^{p_{\text{d-sep}_1}}$ and $\widehat{\mathsf{L}}^{p_{\text{d-sep}_1}}$ respectively. To show $\widehat{T}_{p_{\text{d-sep}_1}} \overset{d}{=} T_{p_{\text{d-sep}_1}}$, it suffices to show that $\widehat{\mathsf{L}}^{p_{\text{d-sep}_1}} \overset{d}{=} \mathsf{L}^{p_{\text{d-sep}_1}}$.

Decomposition of orderings. Any conditioning set ordering L over all c conditioning sets can be uniquely decomposed into three components:

1. $\pi_{\text{d-sep}}$: the relative ordering (permutation) among the m d-separating sets
2. $\pi_{\text{non-d-sep}}$: the relative ordering (permutation) among the $c - m$ non-d-separating sets
3. π_{rel}: the relative placement of d-separating sets versus non-d-separating sets (which sets come before which other sets)

Therefore, the distribution over conditioning set orderings is uniquely determined by the joint distribution over $(\pi_{\text{d-sep}}, \pi_{\text{non-d-sep}}, \pi_{\text{rel}})$. We will show that this joint distribution is identical for both $\mathsf{L}^{p_{\text{d-sep}_1}}$ and $\widehat{\mathsf{L}}^{p_{\text{d-sep}_1}}$.

Analysis of the coupling procedure. In our coupling construction (Step 1), we explicitly sampled $\pi_{\text{d-sep}}$ and $\pi_{\text{non-d-sep}}$ uniformly and independently. The relative placement π_{rel} is then determined by: for each position i in L, whether l_i is correctly classified (with probability $p_{\text{d-sep}_1}$) determines whether that conditioning set goes to B_1 or B_2, and then π_{rel} corresponds to the partition structure (B_1, B_2). Since each conditioning set's classification is independent and depends only on its true d-separation status and $p_{\text{d-sep}_1}$, we have:

- $\widehat{\pi}_{\text{d-sep}}$ is uniform over all permutations of m d-separating sets
- $\widehat{\pi}_{\text{non-d-sep}}$ is uniform over all permutations of $c - m$ non-d-separating sets
- $\widehat{\pi}_{\text{rel}}$ has distribution determined by $p_{\text{d-sep}_1}$ (probability of correct classification)
- $\widehat{\pi}_{\text{d-sep}}, \widehat{\pi}_{\text{non-d-sep}}, \widehat{\pi}_{\text{rel}}$ are mutually independent

Analysis of Subroutine 4. We now show that $\mathsf{L}^{p_{\text{d-sep}_1}}$ has the same distributional structure. By construction of Subroutine 4:

- Conditioning sets in $[A]_k$ are initially shuffled uniformly at random
- Each conditioning set is independently classified based on whether it d-separates in $\widehat{\mathcal{G}}$ with probability $p_{\text{d-sep}_1}$ of correct classification
- Sets are partitioned into B_1 (predicted d-separating) and B_2 (predicted non-d-separating), preserving their random ordering within each bucket

To show $\pi_{\text{d-sep}}$ is uniform: Start with an initial uniform random permutation $\pi^{\text{init}}_{\text{d-sep}}$ of all m d-separating sets. Some subset of size $\ell \sim \text{Binomial}(m, 1 - p_{\text{d-sep}_1})$ are misclassified and placed in B_2, while the remaining $m - \ell$ are placed in B_1. For any fixed ℓ and any fixed choice of which ℓ sets are misclassified, this operation defines a bijection from $\pi^{\text{init}}_{\text{d-sep}}$ to the resulting permutation: given the final permutation and knowing which sets went to which bucket, we can uniquely recover $\pi^{\text{init}}_{\text{d-sep}}$, and vice versa. Since bijections preserve uniformity, the marginal distribution of $\pi_{\text{d-sep}}$ (after marginalizing over ℓ and the choice of which sets) remains uniform. By the same argument, $\pi_{\text{non-d-sep}}$ is uniform.

For independence: The distribution of π_{rel} is determined solely by which conditioning sets are correctly classified (controlled by $p_{\text{d-sep}_1}$). For any fixed realization of π_{rel} (i.e., fixed partition (B_1, B_2)), the above uniformity argument shows that $\pi_{\text{d-sep}}$ and $\pi_{\text{non-d-sep}}$ remain uniform. Therefore $(\pi_{\text{d-sep}}, \pi_{\text{non-d-sep}})$ are independent of π_{rel}.

Conclusion. Both $\widehat{\mathsf{L}}^{p_{\text{d-sep}_1}}$ and $\mathsf{L}^{p_{\text{d-sep}_1}}$ decompose into $(\pi_{\text{d-sep}}, \pi_{\text{non-d-sep}}, \pi_{\text{rel}})$ where each component has identical marginal distributions and the same independence structure. Therefore $\widehat{\mathsf{L}}^{p_{\text{d-sep}_1}} \overset{d}{=} \mathsf{L}^{p_{\text{d-sep}_1}}$, which implies $\widehat{T}_{p_{\text{d-sep}_1}} \overset{d}{=} T_{p_{\text{d-sep}_1}}$. By the same argument, $\widehat{T}_{p_{\text{d-sep}_2}} \overset{d}{=} T_{p_{\text{d-sep}_2}}$.

D.11.3 Step 3: Showing that the Coupling is Monotone

Our goal is to show that $\mathbb{P}[\widehat{T}_{p_{\text{d-sep}_1}} \geq \widehat{T}_{p_{\text{d-sep}_2}}] = 1$. We establish this by showing that the conditioning set ordering $\widehat{\mathsf{L}}^{p_{\text{d-sep}_2}}$ can be obtained from $\widehat{\mathsf{L}}^{p_{\text{d-sep}_1}}$ through a sequence of runtime-reducing swaps.

Structure of the two orderings. Recall from Step 1 that both $\widehat{\mathsf{L}}^{p_{\text{d-sep}_1}}$ and $\widehat{\mathsf{L}}^{p_{\text{d-sep}_2}}$ are generated using the same shared randomness $(L, \mathcal{R}, \pi_{\text{d-sep}}, \pi_{\text{non-d-sep}})$. Crucially, the permutations $\pi_{\text{d-sep}}$ (ordering among d-separating sets) and $\pi_{\text{non-d-sep}}$ (ordering among non-d-separating sets) are identical in both orderings. The only difference lies in the relative placement π_{rel} of d-separating sets versus non-d-separating sets, which is determined by which conditioning sets are correctly classified.

More sets correctly classified at higher accuracy. Since $p_{\text{d-sep}_1} < p_{\text{d-sep}_2}$, for each position i in L:

- If conditioning set i is correctly classified under accuracy $p_{\text{d-sep}_1}$ (i.e., $r_i \leq p_{\text{d-sep}_1}$), then it is also correctly classified under accuracy $p_{\text{d-sep}_2}$ (since $r_i \leq p_{\text{d-sep}_1} < p_{\text{d-sep}_2}$)
- Additional sets may be correctly classified under $p_{\text{d-sep}_2}$ that were misclassified under $p_{\text{d-sep}_1}$ (those with $p_{\text{d-sep}_1} < r_i \leq p_{\text{d-sep}_2}$)

This means:

- D-separating sets that were correctly placed in B_1 in $L_F^{p_{\text{d-sep}_1}}$ remain in B_1 in $L_F^{p_{\text{d-sep}_2}}$
- Some d-separating sets that were incorrectly placed in B_2 in $L_F^{p_{\text{d-sep}_1}}$ may move to B_1 in $L_F^{p_{\text{d-sep}_2}}$ (moving leftward)
- Non-d-separating sets that were correctly placed in B_2 in $L_F^{p_{\text{d-sep}_1}}$ remain in B_2 in $L_F^{p_{\text{d-sep}_2}}$
- Some non-d-separating sets that were incorrectly placed in B_1 in $L_F^{p_{\text{d-sep}_1}}$ may move to B_2 in $L_F^{p_{\text{d-sep}_2}}$ (moving rightward)

Characterizing the transformation via inversions. To formalize how $\widehat{\mathsf{L}}^{p_{\text{d-sep}_2}}$ relates to $\widehat{\mathsf{L}}^{p_{\text{d-sep}_1}}$, we introduce a numerical ordering. Assign integers to conditioning sets as follows:

- Assign $\{1, 2, \ldots, m\}$ to the d-separating sets according to their fixed relative order $\pi_{\text{d-sep}}$
- Assign $\{m+1, \ldots, c\}$ to the non-d-separating sets according to their fixed relative order $\pi_{\text{non-d-sep}}$

Under this assignment, every d-separating set has a smaller numerical label than every non-d-separating set. An **inversion** is any pair of conditioning sets that appears out of numerical order—specifically, a non-d-separating set appearing before a d-separating set in the ordering. Since $\pi_{\text{d-sep}}$ and $\pi_{\text{non-d-sep}}$ are fixed, sets of the same type (both d-separating or both non-d-separating) never form inversions relative to each other.

The transformation from $\widehat{\mathsf{L}}^{p_{\text{d-sep}_1}}$ to $\widehat{\mathsf{L}}^{p_{\text{d-sep}_2}}$ only moves d-separating sets leftward and non-d-separating sets rightward. This process cannot create new inversions: if a non-d-separating set precedes a d-separating set in $\widehat{\mathsf{L}}^{p_{\text{d-sep}_2}}$, that pair must have been in the same order in $\widehat{\mathsf{L}}^{p_{\text{d-sep}_1}}$. Therefore, the set of inversions in $\widehat{\mathsf{L}}^{p_{\text{d-sep}_2}}$ is a subset of the inversions in $\widehat{\mathsf{L}}^{p_{\text{d-sep}_1}}$.

Weak Bruhat order and reachability. The weak Bruhat order on permutations provides a formal framework for this relationship. A standard result ((Yessenov, 2005), Proposition 2.2) states:

Proposition D.4. *For permutations v and w, $v \leq w$ in weak Bruhat order if and only if* $\text{Inv}(v) \subseteq \text{Inv}(w)$, *where* $\text{Inv}(\cdot)$ *denotes the set of inversions.*

Equivalently, v can be obtained from w by a sequence of adjacent transpositions (w_i, w_{i+1}) where $w_i > w_{i+1}$ in the numerical ordering. In our setting, moving a d-separating set leftward past an adjacent non-d-separating set (moving w_i left of w_{i+1}) is precisely such an inversion-reducing swap, since d-separating sets have smaller numerical labels than non-d-separating sets.

Since $\text{Inv}(\widehat{\mathsf{L}}^{p_{\text{d-sep}_2}}) \subseteq \text{Inv}(\widehat{\mathsf{L}}^{p_{\text{d-sep}_1}})$, it follows that $\widehat{\mathsf{L}}^{p_{\text{d-sep}_2}}$ can be obtained from $\widehat{\mathsf{L}}^{p_{\text{d-sep}_1}}$ through a sequence of adjacent swaps that move d-separating sets leftward past non-d-separating sets.

Monotonicity via runtime-reducing swaps. By Lemma D.9, each such swap (placing a d-separating set before a non-d-separating set) strictly decreases the expected number of tests executed. Since $\widehat{\mathsf{L}}^{p_{\text{d-sep}_2}}$ is reachable from $\widehat{\mathsf{L}}^{p_{\text{d-sep}_1}}$ through a sequence of such runtime-reducing swaps, we have $\widehat{T}_{p_{\text{d-sep}_1}} \geq \widehat{T}_{p_{\text{d-sep}_2}}$, establishing that $\mathbb{P}[\widehat{T}_{p_{\text{d-sep}_1}} \geq \widehat{T}_{p_{\text{d-sep}_2}}] = 1$.

D.11.4 Step 4: Conclusion

It follows from Strassen's Coupling Theorem that $\mathbb{E}[T_{p_{\text{d-sep}_1}}] \geq \mathbb{E}[T_{p_{\text{d-sep}_2}}]$, with strict inequality when $\text{CIT}^k_{\text{adj}_{-j}(\mathcal{C},x_i)}$ contains at least one d-separating set and one non-d-separating set. When such mixed sets exist, there is at least one pair of orderings $\widehat{\mathsf{L}}^{p_{\text{d-sep}_1}}, \widehat{\mathsf{L}}^{p_{\text{d-sep}_2}}$ that can be created through the process outlined in Step 1 such that a d-separating set is swapped to be earlier in the sequence. In that case we have $\widehat{T}_{p_{\text{d-sep}_1}} > \widehat{T}_{p_{\text{d-sep}_2}}$, which implies by the Coupling Theorem that $\mathbb{E}[T_{p_{\text{d-sep}_1}}] > \mathbb{E}[T_{p_{\text{d-sep}_2}}]$.

For part (b), observe that when $p_{\text{d-sep}} = 0.5$, the expert's predictions are independent of the true d-separation structure, producing orderings distributionally equivalent to random orderings. Therefore $\mathbb{E}[T_{\text{EP-G}}(0.5)] = \mathbb{E}[T_{\text{random}}]$. Part (a) then implies $\mathbb{E}[T_{\text{EP-G}}(p_{\text{d-sep}})] \leq \mathbb{E}[T_{\text{random}}]$ for all $p_{\text{d-sep}} \geq 0.5$.

□

D.12 Proof of Theorem 5.1

Theorem 5.1 (Asymptotic Correctness)**.** *Under a consistent conditional independence test, for both PC-Guess and gPC-Guess,* $\lim_{n\to\infty} \mathbb{P}(\widetilde{\mathcal{S}} = \mathcal{S}^*) = 1$.

Proof. We will first show that, under the assumption of oracle CITs (CITs that always return independence when conditioning on a d-separating set and dependence if not), both PC-Guess and gPC-Guess return the correct graph. We will then note that the probability that either method returns the incorrect graph is upper bounded by the probability that at least one CIT run returns an incorrect result. We will conclude by noting that, under a consistent CIT, the probability that any CIT test returns independence goes to 0, yielding $\lim_{n\to\infty} \mathbb{P}(\widetilde{\mathcal{G}} = \mathcal{G}^*) = 1$.

We note that, with access to oracle CIT, PC has previously been shown to always return the correct graph when using any ordering of the vertices any ordering O to guide EL, and by Lemma (Spirtes, 2001), and suborderings L given to EP do not affect whether EP will remove an edge or retain it (Lemma D.7). Therefore, PC-Guess returns the true graph with correct (in)dependence results from tests.

We note that gPC-Guess performs a single pass of the EL, running EP with CITs conditioning on subsets of size 0 to $|V| - 1$. Then, the correctness of gPC-Guess with access to oracle CITs follows directly from Lemmas D.1, D.2).

Now we establish the asymptotic result with consistent CITs. Let K denote the maximum number of CIT tests that could possibly be run by either PC-Guess or gPC-Guess. Since the number of vertices $|V|$ is fixed, K is finite—specifically, $K \leq |V|^2 \cdot 2^{|V|-2}$, as each test involves choosing two variables and a conditioning set from the remaining vertices.

For each possible test τ involving variables X, Y and conditioning set S, let $E_\tau^{(n)}$ denote the event that test τ returns an incorrect result when run on sample size n. By the oracle correctness established above, the algorithm returns the incorrect graph only if at least one test returns an incorrect result. Therefore:

$$\mathbb{P}(\tilde{\mathcal{G}} \neq \mathcal{G}^*) \leq \mathbb{P}\left(\bigcup_{\tau \in \text{tests run}} E_\tau^{(n)}\right)$$

Since the set of tests actually run is a subset of all K possible tests, we have:

$$\mathbb{P}(\tilde{\mathcal{G}} \neq \mathcal{G}^*) \leq \mathbb{P}\left(\bigcup_{\tau=1}^{K} E_\tau^{(n)}\right) \leq \sum_{\tau=1}^{K} \mathbb{P}(E_\tau^{(n)})$$

where the final inequality follows from the union bound.

By the consistency assumption, for each test τ, we have $\lim_{n\to\infty} \mathbb{P}(E_\tau^{(n)}) = 0$. Since K is finite:

$$\lim_{n\to\infty} \mathbb{P}(\tilde{\mathcal{G}} \neq \mathcal{G}^*) \leq \lim_{n\to\infty} \sum_{\tau=1}^{K} \mathbb{P}(E_\tau^{(n)}) = \sum_{\tau=1}^{K} \lim_{n\to\infty} \mathbb{P}(E_\tau^{(n)}) = 0$$

Therefore, $\lim_{n\to\infty} \mathbb{P}(\tilde{\mathcal{G}} = \mathcal{G}^*) = 1$.

□

D.13 Proof of Theorem 5.2

Theorem 5.2 (Performance of PC-Guess). *PC-Guess satisfies C2-C3 at per-iteration level:* ***(a)*** $\mathbb{E}[\Phi_\ell]$ *increases monotonically with* p_ψ*;* ***(b)*** *For fixed* p_ψ*,* $\mathbb{E}[t_\ell]$ *decreases monotonically with* $p_{d\text{-}sep}$*;* ***(c)*** *When* $p_\psi \geq 0.5$*,* $\mathbb{E}[\Phi_\ell] \geq \mathbb{E}[\bar{\Phi}_\ell]$.

Proof. We prove each part separately.

Part (a): The monotonic increase in $\mathbb{E}[\Phi_\ell]$ with p_ψ follows directly from Lemma 4.1. For any iteration ℓ with partial skeleton $\mathcal{C}$, EL-G (Subroutine 3) partitions edges based on expert predictions and processes false edges before true edges. By Lemma 4.1, as expert accuracy p_ψ increases, the expected perfect recovery probability $\mathbb{E}[\Phi_\ell]$ increases monotonically. By Lemma 4.1 this inequality is strict when $\mathcal{C}$ contains both true edges (edges in $\mathcal{S}^*$) and false edges (edges not in $\mathcal{S}^*$) and $\mathcal{G}$ is a nonempty and non-fully connected graph.

Part (b): Under the assumption of Definition E.1, the monotonic decrease in $\mathbb{E}[t_\ell]$ with $p_{\text{d-sep}}$ follows directly from Lemma D.10. For fixed expert edge accuracy p_ψ, as d-separation prediction accuracy $p_{\text{d-sep}}$ increases, when testing any edge using its adjacency set, EP-G (Subroutine 4) places d-separating sets earlier in the test sequence, reducing the expected number of tests $\mathbb{E}[t_\ell]$ conducted before finding a d-separating set or exhausting all tests. Note that the distribution over how likely an edge will be tested with a particular adjacency set remains fixed due to p_ψ remaining fixed. The strict inequality occurs when at least one edge in $\mathcal{C}$ is tested with a sequence of CITs where at least one CIT conditions on a subset which d-separates that edge, and one other CIT does not condition on a subset which d-separates that edge. This occurs exactly when $\mathcal{C}$ contains at least one false edge where at least one vertex in the edge has an adjacency set that contains at least one d-separating subset and one non-d-separating subset.

Part (c): When $p_\psi = 0.5$, the expert classifies each edge as true or false with equal probability. The procedure in EL-G (Subroutine 3) then partitions the edges into two sets and orders them as $\mathsf{O} = \mathcal{C} \setminus \widehat{\mathcal{S}} + \mathcal{C} \cap \widehat{\mathcal{S}}$, with random ordering within each partition. Since each edge is assigned to each partition with probability 0.5 independently, and edges within each partition are randomly ordered, this is equivalent to sampling a uniformly random ordering of all edges in $\mathcal{C}$. Therefore, when $p_\psi = 0.5$, PC-Guess has the same distribution over orderings as the baseline PC (which uses uniformly random orderings), implying $\mathbb{E}[\Phi_\ell] = \mathbb{E}[\bar{\Phi}_\ell]$ at $p_\psi = 0.5$. Combined with part (a), which establishes that $\mathbb{E}[\Phi_\ell]$ increases monotonically with p_ψ, we have $\mathbb{E}[\Phi_\ell] \geq \mathbb{E}[\bar{\Phi}_\ell]$ for all $p_\psi \geq 0.5$. □

D.14 Proof of Theorem 5.3

Theorem 5.3 (Performance of gPC-Guess). *gPC-Guess satisfies C2-C3:* ***(a)*** $\mathbb{E}[\Phi]$ *increases monotonically with* p_ψ*;* ***(b)*** *For fixed* p_ψ*,* $\mathbb{E}[t]$ *decreases monotonically with* $p_{d\text{-}sep}$*;* ***(c)*** *When* $p_\psi \geq 0.5$*,* $\mathbb{E}[\Phi] \geq \mathbb{E}[\bar{\Phi}]$.

Proof. We prove each part separately.

Part (a): The monotonic increase in $\mathbb{E}[\Phi]$ with p_ψ follows directly from Lemma 4.1. gPC-Guess runs EL once on the complete skeleton $\mathcal{C}$, where EL-G (Subroutine 3) partitions edges based on expert predictions and processes false edges before true edges. By Lemma 4.1, as expert accuracy p_ψ increases, the expected perfect recovery probability $\mathbb{E}[\Phi]$ increases monotonically. By Lemma 4.1 this inequality is strict when $\mathcal{C}$ contains both true edges (edges in $\mathcal{S}^*$) and false edges (edges not in $\mathcal{S}^*$). Since gPC-Guess starts with the complete skeleton, $\mathcal{C}$ contains both true and false edges whenever $\mathcal{G}$ is a nonempty and non-fully connected graph.

Part (b): Under the assumption of Definition E.1, the monotonic decrease in $\mathbb{E}[t]$ with $p_{\text{d-sep}}$ follows directly from Lemma D.10. For fixed expert edge accuracy p_ψ, as d-separation prediction accuracy $p_{\text{d-sep}}$ increases, when testing any edge using its adjacency set, EP-G (Subroutine 4) places d-separating sets earlier in the test sequence, reducing the expected number of tests $\mathbb{E}[t]$ conducted before finding a d-separating set or exhausting all tests. Note that the distribution over how likely an edge will be tested with a particular adjacency set remains fixed due to p_ψ remaining fixed. The strict inequality occurs when at least one edge in $\mathcal{C}$ is tested with a sequence of CITs where at least one CIT conditions on a subset which d-separates that edge, and one other CIT does

not condition on a subset which d-separates that edge. This occurs exactly when G^* is not entirely empty (all subsets are d-separating), or entirely connected (no subsets are d-separating).

Part (c): When $p_\psi = 0.5$, the expert classifies each edge as true or false with equal probability. The procedure in EL-G (Subroutine 3) then partitions the edges into two sets and orders them as $\mathsf{O} = \mathcal{C} \setminus \widehat{\mathcal{S}} + \mathcal{C} \cap \widehat{\mathcal{S}}$, with random ordering within each partition. Since each edge is assigned to each partition with probability 0.5 independently, and edges within each partition are randomly ordered, this is equivalent to sampling a uniformly random ordering of all edges in $\mathcal{C}$. Therefore, when $p_\psi = 0.5$, gPC-Guess has the same distribution over orderings as the baseline gPC (which uses uniformly random orderings), implying $\mathbb{E}[\Phi] = \mathbb{E}[\bar{\Phi}]$ at $p_\psi = 0.5$. Combined with part (a), which establishes that $\mathbb{E}[\Phi]$ increases monotonically with p_ψ, we have $\mathbb{E}[\Phi] \geq \mathbb{E}[\bar{\Phi}]$ for all $p_\psi \geq 0.5$. □

E Theoretical Details Concerning EP-G and its Guarantees

In this appendix, we provide the complete theoretical analysis for guiding the Edge Prune (EP) subroutine with expert predictions. While Section 4.3 established that EP orderings cannot affect accuracy (only runtime), we formalize here the full analysis: when and how different orderings impact computational efficiency, what ordering principles are optimal, and how expert accuracy translates to monotonic runtime improvements.

E.1 The Edge Prune Subroutine and Ordering Choices

Recall that the EP subroutine (Subroutine 2) is called by constraint-based algorithms to test individual edges $e_{i,j}$ at a fixed conditioning set size k. Given the current skeleton $\mathcal{C}$, EP computes the adjacency set $A = \text{adj}_{-j}(\mathcal{C}, x_i)$ and tests conditional independence for all size-k subsets $W \subseteq A$. The subroutine terminates as soon as any test $\text{CIT}(x_i, x_j \mid W)$ returns independence, at which point edge $n_{i,j}$ is removed from the skeleton.

Let $\text{CIT}_A^k := \{\text{CIT}(x_i, x_j \mid W) : W \subseteq A, |W| = k\}$ denote the collection of all possible size-k conditional independence tests for this edge. The EP subroutine requires an ordering L that specifies the sequence in which these tests are executed. Our goal is to understand how the choice of L affects algorithm performance.

E.2 Accuracy is Invariant to Ordering

We begin by establishing that EP orderings cannot affect the probability of correctly deciding edge $n_{i,j}$:

Lemma D.7. *Accuracy $\mathbb{P}(Y_{n_{i,j}} = 1)$ is constant for all possible orderings $\mathsf{L}_1, \mathsf{L}_2, \ldots$ of $CIT^k_{\text{adj}_{-j}(\mathcal{C},x_i)}$.*

Proof sketch. Edge $n_{i,j}$ is removed if and only if at least one test in CIT_A^k returns independence. Since each test's outcome depends only on the data, the variables tested, and the conditioning set—not the execution order—the probability of correct removal or retainment of the edge depends only on which tests are run, not their sequence. See Appendix D.8 for full proof.

This result implies that unlike EL (where edge ordering affects accuracy via adjacency set modifications), EP orderings leave the correctness probability unchanged. Our focus therefore shifts to computational efficiency.

E.3 When Orderings Affect Runtime

While accuracy is invariant, the computational cost—measured by the number of tests executed before EP terminates—varies across orderings. To understand when and why, we partition CIT_A^k based on the true d-separation structure of $\mathcal{G}^*$:

- $\text{CIT}_A^{\text{d-sep}} := \{\text{CIT}(x_i, x_j \mid W) : W \text{ d-separates } x_i, x_j \text{ in } \mathcal{G}^*\}$
- $\text{CIT}_A^{\text{non-d-sep}} := \text{CIT}_A^k \setminus \text{CIT}_A^{\text{d-sep}}$

Tests in $\text{CIT}_A^{\text{d-sep}}$ have high probability of returning independence (specifically $1-\alpha$, where α is the Type I error rate), while tests in $\text{CIT}_A^{\text{non-d-sep}}$ have low probability of returning independence (specifically β, the Type II error rate or power deficit).

Lemma D.8. *Under the assumptions of Definition E.1, if $\nexists$ a size k subset of $\text{adj}_{-j}(\mathcal{C}, x_i)$ s_i such that $x_i \perp\!\!\!\perp x_j | s_i$, then any pair of orderings L, L' achieves the same $\mathbb{E}[t_{e_{i,j}}]$, where $t_{e_{i,j}}$ is the number of tests conducted by EP using either L or L'.*

Proof sketch. When no d-separating sets exist, all tests return independence with identical probability β (the false negative rate). Under mutual independence of test outcomes, the expected stopping time follows a geometric distribution with parameter β, which is invariant to the ordering of tests. See Appendix D.9 for full proof.

This lemma reveals the key insight: runtime optimization is only possible when CIT_A^k contains both d-separating and non-d-separating sets. In such cases, strategic ordering can significantly reduce computational cost.

E.4 Optimal Ordering Principles

We now characterize orderings that minimize expected runtime. Our analysis relies on standard technical assumptions from the conditional independence testing literature:

Assumption E.1 (CIT Specificity and Independence)**.** *We assume:*

(i) ***Adequate Specificity:*** $1-\alpha > \beta$*, i.e., the true negative rate (probability of correctly detecting independence when it exists) exceeds the false negative rate (probability of failing to detect dependence when it exists). This holds asymptotically as sample size $n \to \infty$ under standard regularity conditions.*

(ii) ***Conditional independence of tests:*** *For any two distinct conditioning sets $W_1, W_2 \subseteq A$, the outcomes of CIT($x_i, x_j \mid W_1$) and CIT($x_i, x_j \mid W_2$) are (conditionally) independent given the data. This holds asymptotically under faithfulness and sufficient sample size.*

Remark on assumptions. Assumption E.1(ii) is a technical simplification that ensures tractability. In practice, test outcomes are not strictly independent due to shared data and overlapping conditioning sets. However, our main result—that d-separating sets should be placed first—likely holds under weaker conditions. Specifically, we conjecture that the monotonicity guarantee (Lemma D.10) extends to any setting where: (a) tests on d-separating sets have strictly higher independence probability than tests on non-d-separating sets, and (b) test outcomes exhibit limited positive dependence. A rigorous proof under these relaxed conditions remains an open problem, but the intuition is clear: placing high-probability tests first reduces expected runtime regardless of the specific dependence structure among tests.

Under Assumption E.1, we can precisely characterize optimal orderings:

Lemma D.9. *Under the assumptions of Definition E.1, given a sequence of CITs L for edge $n_{i,j}$, for any pair of adjacent CITs consisting of a test on a non-d-separating set s_1 followed by a test on a d-separating set s_2, the sequence L' generated by swapping the pair to place the d-separating test s_2 first achieves strictly better runtime, i.e., $\mathbb{E}[t^{\mathsf{L}'}_{e_{i,j}}] < \mathbb{E}[t^{\mathsf{L}}_{e_{i,j}}]$, where $t^{\mathsf{L}}_{e_{i,j}}$ is the number of tests conducted by EP under ordering L.*

Proof sketch. Consider orderings L and L' differing only in positions i and $i+1$, where L places a non-d-separating test before a d-separating test, and L' swaps them. The expected runtime difference equals $\mathbb{P}(\text{all prior tests fail}) \cdot [(1-\alpha)\cdot i + \alpha\beta(i+1)] - [\beta \cdot i + (1-\beta)(1-\alpha)(i+1)]$. Since $1-\alpha > \beta$, the first term places more weight on i while the second places more weight on $i+1$. By the rearrangement inequality, this difference is negative, establishing $\mathbb{E}[t^{\mathsf{L}'}_{e_{i,j}}] < \mathbb{E}[t^{\mathsf{L}}_{e_{i,j}}]$. See Appendix D.10 for full proof.

This lemma establishes a clear ordering principle: placing d-separating sets before non-d-separating sets minimizes expected runtime. Any ordering that violates this principle can be improved by swapping adjacent "inversions."

E.5 Expert-Guided Algorithm with Monotonicity Guarantees

Building on Lemma D.9, we design EP-Guess (Subroutine 4) to leverage expert predictions of d-separating sets. The algorithm extracts all d-separating sets $\widehat{\mathcal{D}}_{ij}$ from the expert graph $\widehat{\mathcal{G}}$ and constructs ordering L by placing predicted d-separating sets ($[A]_k \cap \widehat{\mathcal{D}}_{ij}$) before predicted non-d-separating sets ($[A]_k \setminus \widehat{\mathcal{D}}_{ij}$), with random order within each partition.

We model expert ψ's d-separation predictions using the same framework as edge predictions: for any pair (x_i, x_j) and conditioning set W, the expert independently predicts whether W d-separates x_i, x_j in $\mathcal{G}^*$ with accuracy $p_{\text{d-sep}}$. This can be formalized as a binary symmetric channel where the expert observes the true d-separation status and reports it correctly with probability $p_{\text{d-sep}}$.

Lemma D.10 (Monotonicity of EP Runtime in Expert Accuracy)**.** *Under Assumption E.1, let $T_{EP\text{-}G}(p_{d\text{-}sep})$ denote the number of tests executed by EP-Guess (Subroutine 4) when testing edge $n_{i,j}$ at conditioning set size k with expert ψ having d-separation accuracy $p_{d\text{-}sep}$. Then:*

(a) $\mathbb{E}[T_{EP\text{-}G}(p_{d\text{-}sep})]$ decreases monotonically with $p_{d\text{-}sep}$, strictly decreasing when $[A]_k$ contains both d-separating and non-d-separating sets (where $A = \text{adj}_{-j}(\mathcal{C}, x_i)$)

(b) When $p_{d\text{-}sep} \geq 0.5$, $\mathbb{E}[T_{EP\text{-}G}(p_{d\text{-}sep})] \leq \mathbb{E}[T_{random}]$ where T_{random} denotes runtime under random ordering

Proof sketch. The proof (App. D.11) establishes monotonicity via a coupling argument between experts with accuracies $p_{\text{d-sep}_1} < p_{\text{d-sep}_2}$. Both experts observe the same true d-separating sets and use identical randomness for classification, but the higher-accuracy expert makes fewer errors. This ensures that every conditioning set correctly classified by the weaker expert is also correctly classified by the stronger expert.

Consequently, the better expert's ordering has fewer "inversions" (non-d-separating sets incorrectly placed before d-separating sets). The better ordering can be obtained from the weaker ordering through a sequence of adjacent swaps that move d-separating sets leftward past non-d-separating sets. By Lemma D.9, each such swap strictly decreases expected runtime.

Since the better expert's ordering is reachable through runtime-reducing swaps, it achieves pointwise improvement for any fixed realization of data and expert predictions. Strassen's Coupling Theorem then implies that $T_{\text{EP-G}}(p_{\text{d-sep}_2})$ stochastically dominates $T_{\text{EP-G}}(p_{\text{d-sep}_1})$ in the first-order sense, yielding monotonicity in expectation. Part (b) follows from observing that $p_{\text{d-sep}} = 0.5$ produces orderings distributionally equivalent to random orderings.

E.6 Discussion and Comparison to Edge Loop Guidance

These results establish that EP-Guess provides complementary benefits to EL-Guess:

- **EL-Guess** (Section 4.2): Improves *accuracy* by prioritizing false edges, thereby reducing adjacency set inflation and improving the probability of correct edge decisions. The benefit is measured by increased perfect recovery probability Φ.
- **EP-Guess** (this section): Improves *computational efficiency* by prioritizing d-separating sets, thereby increasing early termination probability. The benefit is measured by decreased expected runtime $\mathbb{E}[T]$.

Both forms of guidance share the same underlying structure:

1. Identify an ordering principle that universally improves performance (false edges first for EL; d-separating sets first for EP)
2. Model the expert as a binary symmetric channel predicting the relevant classification
3. Use coupling arguments to prove monotonic improvement with expert accuracy
4. Establish robustness: performance is never worse than random when expert accuracy ≥ 0.5

The main technical difference is that EL guidance affects accuracy (via adjacency set modifications), while EP guidance affects only runtime (leaving accuracy invariant). This difference arises because EP operates at a fixed conditioning set size, testing all relevant subsets regardless of order, whereas EL modifies the graph structure dynamically, affecting which tests are even possible for subsequent edges.

In practice, both forms of guidance can be applied simultaneously: EL-Guess sequences edges optimally (prioritizing false edges), and within each edge's testing, EP-Guess sequences conditioning sets optimally (prioritizing predicted d-separating sets). The combined algorithm achieves both accuracy improvements and computational speedups when expert predictions are accurate.

F Extract Ordering Subroutine

Subroutine F.1 Extract Orderings from Expert (EOE)

1: **Inputs**: Expert ψ, complete skeleton $\mathcal{C}$
2: Obtain $\hat{\mathcal{G}}$ from ψ. Extract skeleton $\hat{\mathcal{S}}$ and d-separating sets $\hat{\mathcal{D}}$ from $\hat{\mathcal{G}}$
3: Randomly order $\mathcal{C}$ and $[V]_{1:|V|-1}$. Set $\mathsf{O} = \mathcal{C} \setminus \hat{\mathcal{S}} + \mathcal{C} \cap \hat{\mathcal{S}}$ and $\mathsf{L} = \left([V]_{1:|V|-1} \cap \hat{\mathcal{D}}\right) + \left([V]_{1:|V|-1} \setminus \hat{\mathcal{D}}\right)$
4: **return** O, L

Subroutine F.1 extracts orderings O, L from an expert's guess of the causal DAG $\widehat{\mathcal{G}}$. It combines the first few steps of both Subroutine 3 and 4.

G Suboptimality of PC-Guess under Perfect Guidance

We demonstrate that PC-Guess's level-by-level structure can prevent it from fully exploiting expert guidance, even when the expert provides perfect predictions. This inefficiency stems from PC's statistical conditioning bias, which prioritizes testing smaller conditioning sets before larger ones regardless of expert predictions.

The core limitation. When an expert correctly identifies a false edge $n_{i,j}$ and places it early in the edge ordering O, PC-Guess can only remove this edge once it reaches conditioning set size $\ell = k$, where k is the size of the minimal d-separating set for x_i and x_j. At all prior levels $\ell \in \{0, 1, \ldots, k-1\}$, PC-Guess must test the edge and find dependence, leaving the false edge in the skeleton. During these early levels, this retained false edge inflates the adjacency sets of x_i and x_j, forcing unnecessary conditioning set tests for all other edges incident to these vertices.

Concrete example: 4-node chain. Consider the true causal graph $\mathcal{G}^* : x_1 \to x_2 \to x_3 \to x_4$. The complete initial skeleton $\mathcal{C}$ contains 6 edges: three true edges $\{n_{1,2}, n_{2,3}, n_{3,4}\}$ and three false edges $\{n_{1,3}, n_{1,4}, n_{2,4}\}$. The false edge $n_{1,4}$ has minimal d-separating set $\{x_2\}$ of size $k = 1$. Suppose the expert provides perfect guidance by placing $n_{1,4}$ first in the ordering O. At level $\ell = 0$, PC-Guess tests $n_{1,4}$ with conditioning set $\emptyset$ and finds dependence (correct, due to the chain path), leaving the false edge in the skeleton. Because $n_{1,4}$ remains, all subsequent level-0 tests of edges involving x_1 or x_4 use inflated adjacency sets: when testing $n_{1,2}$, we have $\text{adj}_{-2}(\mathcal{C}, x_1) = \{x_3, x_4\}$ instead of $\{x_3\}$; when testing $n_{3,4}$, we have $\text{adj}_{-4}(\mathcal{C}, x_3) = \{x_1, x_2\}$ instead of $\{x_2\}$. Only at level $\ell = 1$ does PC-Guess test $n_{1,4}$ with $\{x_2\}$ and successfully remove it. The inefficiency: PC-Guess performed one guaranteed-to-fail test of $n_{1,4}$ at level 0, plus all level-0 tests of edges incident to x_1 or x_4 used unnecessarily large adjacency sets—waste that scales as $O(d^2)$ for chains of length d, all avoidable if the algorithm could immediately test $n_{1,4}$ with its minimal d-separator.

Conclusion. This example demonstrates that PC-Guess's adherence to the level-by-level structure prevents it from fully exploiting perfect expert guidance. The algorithm must exhaust all smaller conditioning set sizes before testing the conditioning set size that would actually remove the false edge, wasting both direct tests on the false edge and indirect tests on neighboring edges with inflated adjacency sets. This motivates gPC-Guess (Section 5.2), which eliminates the level-by-level constraint and allows immediate testing with conditioning sets of any size, enabling the algorithm to act on expert predictions without delay.

H Experimental Details

H.1 Synthetic Data Generation Parameters

We generate synthetic data using linear Gaussian structural equation models on Erdős-Rényi (ER) random graphs. The data generation process follows these steps:

- **Graph Structure**: We generate d-dimensional DAGs using the Erdős-Rényi model where edges are added independently with probability p_{edge}. The DAG property is ensured by making the adjacency matrix lower triangular, preventing cycles and self-loops.
- **Edge Weights**: For each edge in the binary adjacency matrix, we assign weights sampled uniformly from $[-2.5, -1.5] \cup [1.5, 2.5]$ to avoid faithfulness violations that occur when weights are near zero.
- **Data Generation**: Each variable x_i follows the linear structural equation model:

$$x_i = \sum_{j \in \mathrm{Pa}(x_i)} w_{ji} x_j + \varepsilon_i$$

  where w_{ji} are the edge weights and $\varepsilon_i \sim \mathcal{N}(0, 1)$ are independent Gaussian noise terms with mean 0 and variance 1.0. All noise terms are generated independently across variables and samples.
- **Standardization**: All generated data is standardized to zero mean and unit variance to ensure fair comparison across different graph structures.
- **Variable Randomization**: Before feeding data to any causal discovery method, we randomly permute the order of variables to prevent information leakage from variable ordering.

We consider two sparsity levels based on edge probability:

- **Sparse graphs (ER1)**: Edge probability $p_{edge} = 1/(d-1)/2$, yielding approximately d edges in expectation
- **Dense graphs (ER3)**: Edge probability $p_{edge} = 3/(d-1)/2$, yielding approximately $3d$ edges in expectation

In the main text results, we focus on sparse ER3 graphs with $d = 20$ variables and $n = 100$ samples.

H.2 Real-World Data

We use the discrete version of the Sachs protein signaling dataset from the BNLearn repository, which contains measurements of 11 phosphoproteins and phospholipids in human immune system cells. The dataset details include:

- **Variables**: 11 proteins/phospholipids: Raf, Mek, Plcg, PIP2, PIP3, Erk, Akt, PKA, PKC, P38, Jnk.
- **Ground Truth**: Known causal DAG structure based on established biological pathways.
- **Sample Sizes**: We subsample the original dataset to create experiments with $n = 100$ samples.
- **Preprocessing**: Data is standardized to zero mean and unit variance.

The Sachs dataset provides a realistic benchmark for evaluating causal discovery methods on real biological networks with known ground truth structure.

H.3 Algorithm Input and Baseline Methods

Algorithm input and ordering choices. For all experiments reported in Section 6 and Appendix I (except Appendix I.6), algorithms receive only a predicted *skeleton*—an undirected graph over the variables—not a full DAG. This skeleton guides the Edge Loop (EL) subroutine in determining which edges to test and in what order (see Section 4). The ordering used to guide Edge Prune (EP)—the sequence of conditioning sets tested for each individual edge—is *always generated uniformly at random* for all methods. This design choice reflects our focus on accuracy improvements rather than runtime gains: by Lemma D.7, EP ordering does not affect edge decision accuracy, only computational cost.

Baseline methods. PC and gPC serve as baseline versions of PC-Guess and gPC-Guess respectively. These baselines always receive skeletons generated with expert edge prediction accuracy $p_\psi = 0.5$ (equivalent to random guessing) and use random EP ordering (equivalent to $p_{\text{d-sep}} = 0.5$). By comparing PC-Guess and gPC-Guess against PC and gPC, we isolate the benefit of expert guidance over random ordering.

Number of trials. All results reported in Section 6 and Appendix I reflect averages over 30 independent trials, each with a different random seed controlling data generation, expert prediction sampling, and algorithmic randomness.

H.4 Simulated Expert Implementation

We implement simulated experts that generate skeleton predictions for evaluating algorithm performance across controlled accuracy levels.

Skeleton generation for simulated experts. For experiments with simulated experts (Figures 2a, 2b, and most experiments in Appendix I), we generate predicted skeletons as follows:

1. For each potential edge pair (x_i, x_j) where $i < j$, we check whether the edge exists in the true skeleton $\mathcal{S}^*$
2. With probability p_ψ (the expert edge prediction accuracy), we correctly classify the edge (add it to $\widehat{\mathcal{S}}$ if it exists in $\mathcal{S}^*$, exclude it otherwise)
3. With probability $1 - p_\psi$, we misclassify the edge (add it to $\widehat{\mathcal{S}}$ if it does *not* exist in $\mathcal{S}^*$, exclude it otherwise)

This process simulates a binary symmetric channel and allows systematic evaluation across controlled accuracy levels $p_\psi \in [0.3, 1.0]$. Each of the 30 trials in each experiment with simulated experts uses an independently sampled skeleton prediction.

D-separation prediction (Appendix I.6 only). The experiments in Appendix I.6 differ from all other experiments because they focus on evaluating how d-separation prediction accuracy $p_{\text{d-sep}}$ affects runtime (not accuracy). For these experiments:

1. The skeleton $\widehat{\mathcal{S}}$ is generated using a simulated expert with accuracy $p_\psi = 0.5$ (random edge prediction), ensuring the skeleton provides no informative signal about true edge structure
2. We use a simulated expert to predict d-separating sets: for each edge $e_{i,j}$ being tested, and for each candidate conditioning set W, the expert correctly identifies whether W d-separates x_i, x_j in $\mathcal{G}^*$ with probability $p_{\text{d-sep}}$
3. This d-separation prediction guides EP ordering according to Subroutine 4: predicted d-separating sets are tested before predicted non-d-separating sets
4. We vary $p_{\text{d-sep}} \in [0.5, 1.0]$ to measure how d-separation accuracy affects expected runtime (Lemma D.10)

H.5 LLM Expert Implementation

We use Claude Opus 4.1 as our LLM expert, accessed through Amazon Bedrock. For experiments with the LLM expert (Figure 2c), the skeleton generation process is:

- **Prompting Strategy**: For each of the 30 trials, we prompt Claude Opus 4.1 once using the following prompt template, with variable names randomly shuffled for that trial (to prevent prompt bias):

"Analyze this protein signaling network step by step: {var_names_str}
Step 1: Consider each protein's known biological functions
Step 2: Identify which proteins can directly interact with each other
Step 3: Look for signaling pathways and cascades
Step 4: Include regulatory relationships (activation/inhibition)
For each pair of proteins, ask: Can protein A directly influence protein B's activity or state?
List ALL direct causal relationships as pairs. Be comprehensive - missing edges is worse than including uncertain ones. Return your answer as a list of variable name pairs that have direct edges between them. Format your response as pairs of variable names in parentheses, separated by commas. Start your final answer with the tag EDGES: followed by your list."

This prompt uses step-by-step reasoning to systematically guide the LLM through the causal discovery process. It employs chain-of-thought prompting by breaking down the analysis into discrete steps, and uses recall-oriented instructions ("Be comprehensive", "missing edges is worse") to encourage high recall of potential causal relationships.

- **Response Parsing**: We parse the LLM response (which returns edge pairs in the format "EDGES: (var1, var2), (var3, var4), ...") to extract the predicted skeleton $\widehat{\mathcal{S}}$ for that trial.
- **Trial-specific pairing**: For each trial, we sample $n = 100$ observations from the Sachs dataset and provide both the data and the corresponding LLM-predicted skeleton to gPC-Guess. This approach generates 30 independent LLM predictions (one per trial), each paired with an independent data subsample.

The LLM expert leverages pre-trained biological knowledge to make predictions about protein signaling networks, providing a realistic test of how modern AI systems can augment causal discovery.

H.6 Conditional Independence Testing

For synthetic data experiments with linear Gaussian models, all algorithms use Fisher's Z-test (Fisher, 1921) for conditional independence testing with significance level $\alpha = 0.05$. The test statistic is:

$$Z = \frac{1}{2}\sqrt{n - |S| - 3}\log\left(\frac{1 + \hat{\rho}_{XY|S}}{1 - \hat{\rho}_{XY|S}}\right)$$

where $\hat{\rho}_{XY|S}$ is the sample partial correlation between variables X and Y given conditioning set S, and n is the sample size. Under the null hypothesis of conditional independence, Z follows a standard normal distribution. Fisher's Z-test is appropriate for continuous data generated from linear Gaussian structural equation models.

For experiments with the Sachs dataset, we use the chi-square test (Pearson, 1900) of independence, which is appropriate for discrete data. The test statistic is:

$$\chi^2 = \sum_{i,j} \frac{(O_{ij} - E_{ij})^2}{E_{ij}}$$

where O_{ij} are observed frequencies and E_{ij} are expected frequencies under the null hypothesis of independence in the contingency table. This test evaluates conditional independence by comparing observed and expected frequencies across all combinations of variable values and conditioning set states.

H.7 Packages and Dependencies

The experimental code uses the following Python packages:

- **numpy** (1.21.0+): Array operations and linear algebra
- **scipy** (1.7.0+): Statistical functions and hypothesis testing
- **causal-learn** (0.1.3.8+): PC and PC-Stable algorithm implementations
- **networkx** (2.6.0+): Graph operations and d-separation queries

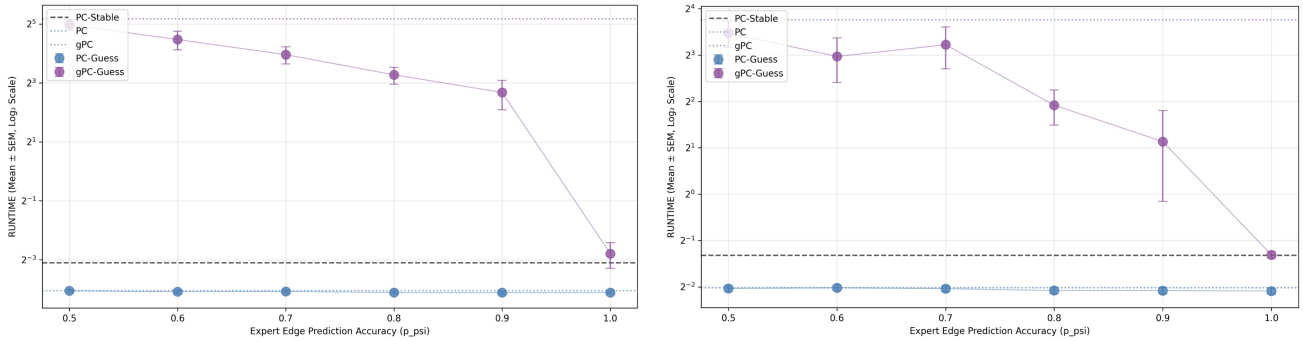
- **boto3** (1.26.0+): Amazon Bedrock API access for LLM experiments
- **matplotlib** (3.5.0+): Plotting and visualization
- **tqdm** (4.62.0+): Progress bars for long-running experiments
- **json**: Configuration and results serialization (Python standard library)
- **itertools**: Combinatorial operations for conditioning sets (Python standard library)
- **concurrent.futures**: Parallel experiment execution (Python standard library)
- **datetime**: Experiment timestamping and logging (Python standard library)

H.8 Compute Details

All experiments were conducted using Python 3.8+, and run on a EC2 instance with AMD EPYC 7R13 processors, 192 vCPUs (96 cores with 2 threads per core), and 740 GiB of memory running Amazon Linux 2. Parallel experiments used up to 8 concurrent workers via Python's ProcessPoolExecutor to balance computational efficiency with resource constraints.

I Experimental Results in Different Settings

I.1 Runtime Results for Figure 2a and 2b in Main Text



(a) Runtime (s) results for Figure I.2 (ER1, $d = 20, n = 100$).

(b) Runtime (s) results for Figure 2a (ER3, $d = 20, n = 100$).

Figure I.1: Results for runtime when varying p_ψ.

Runtime of gPC-Guess and PC-Guess both decline as expert prediction p_ψ increases, although the reduction is much larger in the dense rather than sparse setting, and in both settings the runtime reduction for gPC-Guess is far larger than for PC-Guess.

I.2 Sparse Graphs

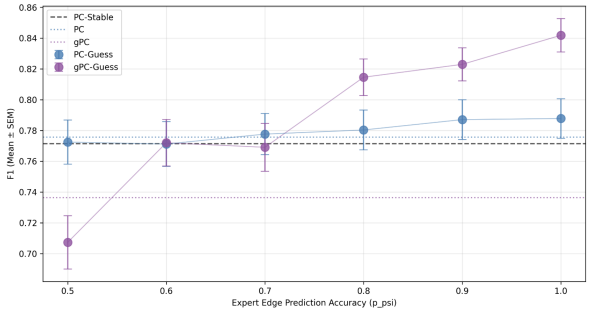


Figure I.2: Method performance as p_ψ increases in sparse graphs (ER1, $d = 10$, $n = 100$).

PC-Guess continues to outperform baselines as p_ψ grows, but with a smaller increase in F1 than observed in the dense setting. Similar to the dense setting, gPC-Guess performance increases the most with p_ψ, again surpassing all other methods when $p_\psi = 0.7$.

I.3 Varying Sample Size

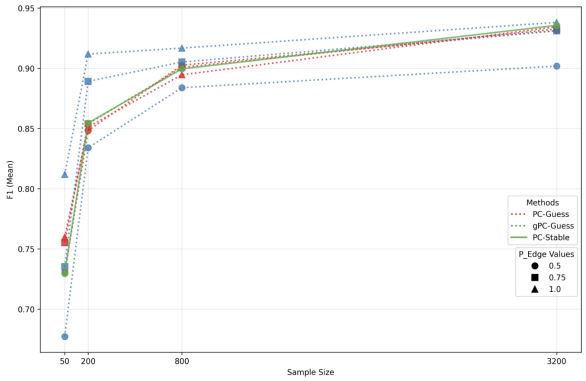


Figure I.3: Method performance across different values of p_ψ, in sparse graphs (ER1, $d = 10$), as sample size is rapidly increased.

As expected from the correctness result provided for PC-Guess and gPC-Guess (Theorem 5.2) that holds independent of expert quality, all methods (no matter what the expert edge prediction accuracy p_ψ is) are converging to perfect accuracy with increasing sample size.

I.4 Varying Dimensionality

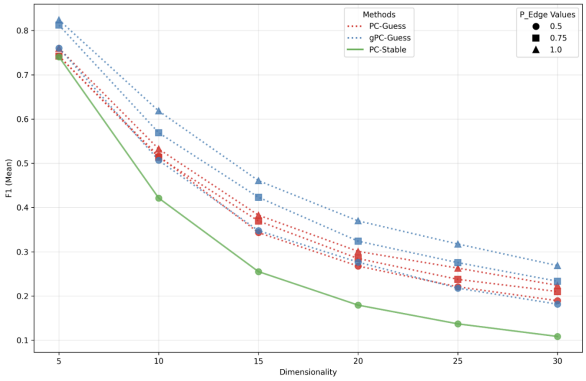
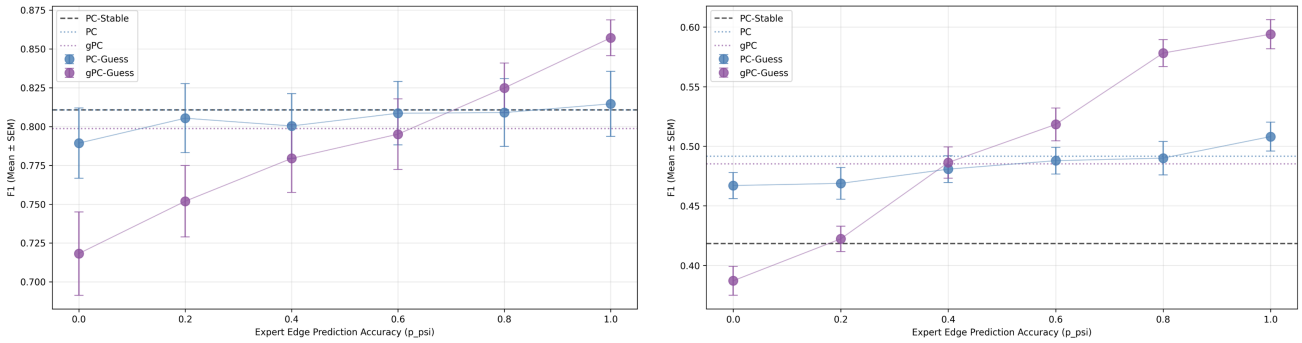


Figure I.4: Method performance across different values of p_ψ, in dense graphs (ER3, $n = 100$), as graph dimensionality d is increased.

We note that gPC-Guess and PC-Guess continue to outperform the baseline PC-Stable as dimensionality is increased, even with sample size fixed. The gap between gPC-Guess and the baseline PC-Stable widens as dimensionality increases—for $p_\psi = 1.0$, the gap between gPC-Guess and PC-Stable at $d = 5$ is only $\sim$7 percentage points, whereas at $d = 30$ the gap between them is $\sim$18 percentage points. This suggests that the value of expert guidance to performance increases in high-dimensional settings that are challenging for purely data-driven methods.

I.5 Results for Worst-Case Expert Performance



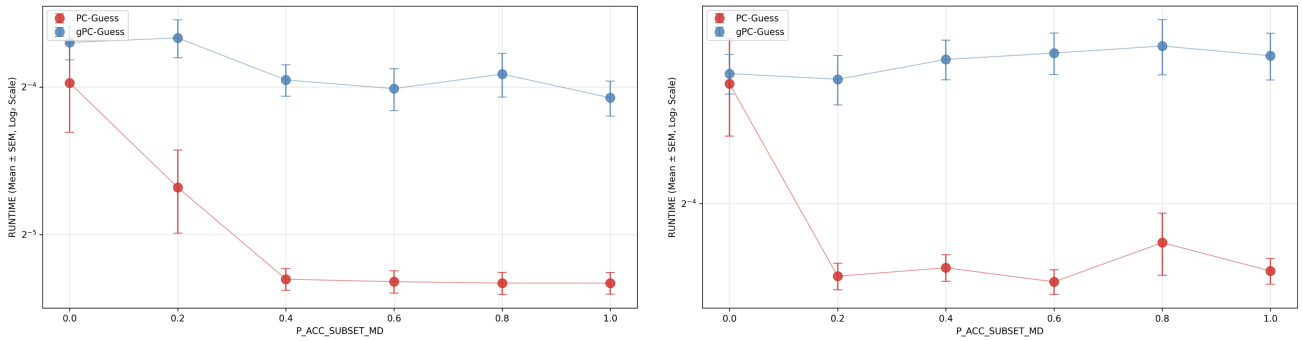
(a) Method performance in sparse graphs (ER1, $d = 10, n = 100$) with varying p_ψ.

(b) Method performance in dense graphs (ER3, $d = 10, n = 100$) with varying p_ψ.

Figure I.5: Results for varying p_ψ in the worst case, i.e., the expert is worse than random ($p_\psi \leq 0.5$).

As expected from our theoretical monotonicity results (Lemma 4.1, Theorem 5.2, Theorem 5.3), we see that both PC-Guess's and gPC-Guess's performances are worse than their counterparts PC and gPC when the expert prediction is worse than random, i.e., $p_\psi \leq 0.5$. We note that PC-Guess's performance is impacted less than gPC-Guess's performance, with a smaller reduction when the expert is poor, but gPC-Guess has a larger gain in performance when the expert is good. However, due to our robust correctness guarantees (Theorem 5.2), in both the dense and sparse setting the worst case performance (i.e., when the expert is entirely inaccurate, every single edge prediction is wrong, $p_\psi = 0$) the performance drop from baseline is only up to roughly 8 percentage points. Unlike expert-aided soft/hard constraint methods, even when expert guidance is poor the drop in performance is bounded because the expert never replaces tests, only guides sequences.

I.6 Varying D-Separation Prediction



(a) Method runtime in sparse graphs (ER1, $d = 10$, $n = 100$).

(b) Method runtime in dense graphs (ER3, $d = 10$, $n = 100$).

Figure I.6: Method runtime as d-separating prediction accuracy $p_{\text{d-sep}}$ is varied.

We note that increasing $p_{\text{d-sep}}$ appears to decrease runtime of both PC-Guess and gPC-Guess in sparse settings as expected by Lemma D.10, but the reduction in dense graphs is not observable for gPC-Guess, and only observable for low $p_{\text{d-sep}}$ values for PC-Guess.

I.7 Verifying Lemma D.6

To validate our theoretical extension (Lemma D.6) allowing for asymmetric expert accuracies, we conduct an additional experiment demonstrating that the monotonicity guarantees hold empirically when p^t_ψ and p^f_ψ vary independently.

Details of our experiment:

-We use synthetic data on $d = 20$ nodes with $n = 100$ samples and edge probability 0.316, conducting 30 trials where all algorithms use FisherZ independence tests with significance level $\alpha = 0.05$.

- We simulate experts with asymmetric accuracies: p^t_ψ (accuracy on true edges) and p^f_ψ (accuracy on false edges), where $p_{MD} = p_{NMD} = 0.5$ are held constant.

- We run two sets of experiments: (1) fixing $p^f_\psi = 0.5$ while varying $p^t_\psi \in \{0.5, 0.6, 0.7, 0.8, 0.9, 1.0\}$, and (2) fixing $p^t_\psi = 0.5$ while varying $p^f_\psi \in \{0.5, 0.6, 0.7, 0.8, 0.9, 1.0\}$.

- We compare PC-GUESS and SGS-GUESS to observe how F1 scores change as each accuracy parameter increases.

Table 1: F1 Scores with $p^f_\psi = 0.5$ (varying p^t_ψ)

Method	$p = 0.5$	$p = 0.6$	$p = 0.7$	$p = 0.8$	$p = 0.9$	$p = 1.0$
PC-GUESS	0.438 ± 0.075	0.442 ± 0.068	0.448 ± 0.071	0.452 ± 0.070	0.446 ± 0.075	0.455 ± 0.074
SGS-GUESS	0.357 ± 0.074	0.384 ± 0.072	0.404 ± 0.062	0.433 ± 0.075	0.433 ± 0.078	0.465 ± 0.074

Table 2: F1 Scores with $p^t_\psi = 0.5$ (varying p^f_ψ)

Method	$p = 0.5$	$p = 0.6$	$p = 0.7$	$p = 0.8$	$p = 0.9$	$p = 1.0$
PC-GUESS	0.438 ± 0.075	0.441 ± 0.068	0.449 ± 0.082	0.448 ± 0.078	0.456 ± 0.066	0.465 ± 0.072
SGS-GUESS	0.357 ± 0.074	0.394 ± 0.078	0.423 ± 0.073	0.445 ± 0.058	0.471 ± 0.058	0.504 ± 0.070

The results confirm our theoretical predictions: F1 scores generally increase monotonically as either p^t_ψ or p^f_ψ increases, with SGS-GUESS showing particularly strong gains at higher accuracy values. These experiments validate that our framework's guarantees extend to more realistic expert error models where precision and recall differ.

J Experiments with Additional Baselines

J.1 Comparing Guess2Graph Directly to Hard Constraints

We include a direct comparison between the impact of hard-constraints (-HC) vs the impact of the G2G framework (-Guess) on the performance of both PC and gPC.

p_{acc}	PC		gPC	
	PC-HC	PC-Guess	gPC-HC	gPC-Guess
0.5	0.387 ± 0.060	$\mathbf{0.425} \pm 0.063$	0.362 ± 0.046	$\mathbf{0.375} \pm 0.054$
0.6	0.417 ± 0.074	$\mathbf{0.445} \pm 0.055$	0.410 ± 0.057	$\mathbf{0.431} \pm 0.062$
0.7	0.472 ± 0.050	$\mathbf{0.483} \pm 0.062$	0.426 ± 0.052	$\mathbf{0.465} \pm 0.063$
0.8	$\mathbf{0.503} \pm 0.054$	0.486 ± 0.071	0.476 ± 0.061	$\mathbf{0.493} \pm 0.048$
0.9	$\mathbf{0.534} \pm 0.063$	0.491 ± 0.059	0.489 ± 0.050	$\mathbf{0.549} \pm 0.067$
1.0	$\mathbf{0.571} \pm 0.054$	0.494 ± 0.053	0.531 ± 0.053	$\mathbf{0.635} \pm 0.051$

Table 3: Performance across different expert accuracy levels (p_{acc}) for PC and gPC variants. Bold indicates the better-performing method within each group.

Details of our experiment:

- We replicate the setting of Figure 2a, i.e. we use the ER3 synthetic data on 20 nodes, vary the simulated expert accuracy, and conduct 30 trials where all algorithms use FisherZ independence tests.
- We introduce a new baseline, PC with hard constraints (PC-HC), which emulates a standard hard constraint approach by randomly sampling % of the expert's predictions of the edges and enforcing them as fixed background knowledge (hard constraints) within the PC algorithm. We similarly define gPC-HC.

We note that PC-Guess outperforms PC-HC in the low accuracy expert range ($p_{acc} < 0.8$), demonstrating the advantage of our G2G framework when handling imperfect experts compared to hard constraint approaches. Notably, gPC-Guess outperforms gPC-HC across all expert accuracy levels, including high accuracy regimes. This superior performance aligns with gPC-Guess's theoretical design: by eliminating PC's level-by-level constraint, it can act immediately on expert predictions, removing false edges earlier regardless of their minimal d-separating set size. This expert responsiveness enables gPC-Guess to better leverage high-quality experts, sometimes to even greater extent than hard-constraint approaches.

J.2 Stacking Guess2Graph on top of Hard Constraints

We run an additional set of experiments that demonstrate the utility of stacking Guess2Graph on top of hard constraints.

p_{acc}	PC		gPC	
	PC-HC	PC-HC-Guess	gPC-HC	gPC-HC-Guess
0.5	0.387 ± 0.060	$\mathbf{0.398} \pm 0.077$	0.362 ± 0.046	$\mathbf{0.364} \pm 0.062$
0.6	0.417 ± 0.074	$\mathbf{0.435} \pm 0.062$	0.410 ± 0.057	$\mathbf{0.433} \pm 0.051$
0.7	$\mathbf{0.472} \pm 0.050$	0.458 ± 0.058	0.426 ± 0.052	$\mathbf{0.491} \pm 0.069$
0.8	0.503 ± 0.054	$\mathbf{0.515} \pm 0.059$	0.476 ± 0.061	$\mathbf{0.557} \pm 0.053$
0.9	0.534 ± 0.063	$\mathbf{0.552} \pm 0.065$	0.489 ± 0.050	$\mathbf{0.608} \pm 0.053$
1.0	0.571 ± 0.054	$\mathbf{0.585} \pm 0.064$	0.531 ± 0.053	$\mathbf{0.685} \pm 0.047$

Table 4: F1 scores for different expert quality levels (p_{acc}) with hard constraint only (-HC) vs. hard constraint and Guess2Graph guidance (-HC-Guess). Constant fraction of edges constrained (0.2).

Details of our experiment:

- We replicate the setting of Figure 2a, i.e. we use the ER3 synthetic data on 20 nodes, vary the simulated expert accuracy, and conduct 30 trials where all algorithms use FisherZ independence tests.

- We adapt PC-HC to PC-HC-Guess, which again randomly samples % of the expert's predicted edges and enforces them as fixed background knowledge. However, it also uses the expert predictions on the remaining % of edges to guide the sequence of tests as described in PC-Guess. We similarly define gPC-HC-Guess.

Note that the combination of constraint and G2G (-HC-Guess) (almost) always has higher F1 score than hard constraint alone (-HC). We find that the trend remains the same for ablations on the % of edges constrained by expert (e.g., 10% or 30% of edges constrained). These results demonstrate the added utility of the G2G framework when used in conjunction with existing hard constraint approaches.

J.3 Comparing Guess2Graph directly to Guidance-based Score Algorithms

We provide a direct comparison between our PC-Guess and gPC-Guess algorithms when compared to guidance-based approaches.

We compare gPC-Guess and PC-Guess against three score-based algorithms—GES (Chickering, 2002), LGES Safe (Ejaz and Bareinboim, 2025), and LGES Cons (Ejaz and Bareinboim, 2025)—each augmented with two guidance approaches from prior work (Ejaz and Bareinboim, 2025): (1) 'init', which initializes the search with expert-suggested edges, and (2) 'priority', which uses expert predictions to prioritize candidate edge insertions. Details of our experiment and method descriptions:

- We replicate the setting of Figure 2b, i.e. discrete Sachs data, simulated expert, randomly subsamples of $n = 100$ for each of 20 trials.
- The base algorithm GES is a traditional score-based causal discovery method that greedily selects the highest-scoring INSERT operator at each step, where INSERT(X, Y, T) adds a directed edge X -¿ Y between non-adjacent nodes and orients certain neighboring undirected edges as pointing into Y. GES performs greedy forward and backward search over Markov equivalence classes, iteratively adding and deleting edges to maximize a decomposable score (e.g., BIC), and is guaranteed in the large-sample limit to recover the true MEC under standard assumptions.
- The algorithm LGES Safe is a relaxation of GES that avoids inserting edges between variables (X,Y) when a score comparison on a single DAG $G \in E$ implies conditional independence (i.e., when G scores higher than $G \cup X \rightarrow Y$). This SafeInsert strategy is provably guaranteed to find a score-increasing insertion whenever one exists, thereby improving finite-sample accuracy and runtime while preserving GES's asymptotic correctness.
- The algorithm LGES Cons is a relaxation of GES that discards all INSERT(X,Y,*) operators when any valid INSERT(X,Y,T) results in a score decrease, indicating conditional independence under some conditioning set. This more aggressive ConservativeInsert strategy avoids edges suggested to be conditionally independent, but has only partial theoretical guarantees compared to SafeInsert.
- The label (init) indicates that the algorithm is initialized in a Markov equivalence class consistent with the expert's predicted edges (i.e., all expert-suggested edges are included in the starting graph). The algorithm then proceeds with its usual greedy or less-greedy search, effectively hard-wiring the prior into the starting point of the search.
- The label (priority) indicates that expert predictions are used to prioritize the order in which INSERT operators are considered via a priority ranking, with expert-suggested edges evaluated first. The algorithm starts from a default MEC and inserts edges only when they improve the score, rather than hard-wiring them into the initialization.
- The label (no prior) indicates that the algorithm was run without any assistance/guidance from an expert.
- We use code from (Ejaz and Bareinboim, 2025) for the implementation of GES, LGES Safe, LGES Cons (all using discrete loss functions) as well as the guidance augmentations.

Our gPC-Guess method outperforms all guidance-based approaches across the entire range of expert accuracy, and is the only method exhibiting monotonic improvement as expert quality increases. This behavior reflects gPC-Guess's unique satisfaction of our C2 (monotonicity) and C3 (robust finite-sample performance) criteria, whereas

p_{acc}	Guess2Graph		Init			Priority		No prior		
	gPC-Guess	PC-Guess	GES	LGES Safe	LGES Cons	LGES Safe	LGES Cons	GES	LGES Safe	LGES Cons
0.5	<u>0.501</u> $\pm$ 0.057	**0.615** $\pm$ 0.067	0.429 $\pm$ 0.065	0.431 $\pm$ 0.033	0.390 $\pm$ 0.061	0.476 $\pm$ 0.064	0.485 $\pm$ 0.053	0.485 $\pm$ 0.053	0.485 $\pm$ 0.053	0.485 $\pm$ 0.053
0.6	<u>0.582</u> $\pm$ 0.058	**0.618** $\pm$ 0.052	0.421 $\pm$ 0.064	0.431 $\pm$ 0.055	0.389 $\pm$ 0.057	0.476 $\pm$ 0.064	0.485 $\pm$ 0.053	0.485 $\pm$ 0.053	0.485 $\pm$ 0.053	0.485 $\pm$ 0.053
0.7	**0.615** $\pm$ 0.065	<u>0.613</u> $\pm$ 0.067	0.453 $\pm$ 0.055	0.425 $\pm$ 0.048	0.438 $\pm$ 0.064	0.485 $\pm$ 0.053	0.485 $\pm$ 0.053	0.485 $\pm$ 0.053	0.485 $\pm$ 0.053	0.485 $\pm$ 0.053
0.8	**0.713** $\pm$ 0.045	<u>0.619</u> $\pm$ 0.069	0.434 $\pm$ 0.080	0.439 $\pm$ 0.053	0.408 $\pm$ 0.034	0.485 $\pm$ 0.053	0.476 $\pm$ 0.043	0.485 $\pm$ 0.053	0.485 $\pm$ 0.053	0.485 $\pm$ 0.053
0.9	**0.741** $\pm$ 0.071	<u>0.627</u> $\pm$ 0.062	0.431 $\pm$ 0.062	0.459 $\pm$ 0.065	0.437 $\pm$ 0.076	0.485 $\pm$ 0.053	0.485 $\pm$ 0.053	0.485 $\pm$ 0.053	0.485 $\pm$ 0.053	0.485 $\pm$ 0.053
1.0	**0.822** $\pm$ 0.049	<u>0.627</u> $\pm$ 0.049	0.441 $\pm$ 0.078	0.436 $\pm$ 0.050	0.416 $\pm$ 0.060	0.485 $\pm$ 0.053	0.485 $\pm$ 0.053	0.485 $\pm$ 0.053	0.485 $\pm$ 0.053	0.485 $\pm$ 0.053

Table 5: F1 scores across different expert accuracy levels (p_{acc}) for Guess2Graph, GES, and LGES variants. Bold indicates the best result in each row, and underlining indicates the second-best result.

the other guidance-based approaches only guarantee asymptotic consistency (C1). These results underscore the practical value of our theoretical guarantees: practitioners can be more confident in improved performance when leveraging imperfect experts with G2G, unlike with heuristic guidance-based methods.

We note that the init and priority guidance augmentation for the score-based methods do not outperform no prior in our results. We contextualize this somewhat counterintuitive finding below:

Search efficiency vs. score reliability Guidance-based augmentations help local search algorithms find high-scoring graphs more efficiently, by initializing the search from a graph closer to the true graph, or nudging the method to consider edge additions/removals suggested by the expert. They do not make the score estimates themselves more reliable. In limited-sample settings, the bottleneck is often score reliability - statistical noise can cause the highest-scoring graph (when scores are computed naively) to differ from the true graph. When this is the primary source of error, guiding the search toward the "right" region of the space provides little benefit: the algorithm will still favor incorrect graphs that happen to score higher due to noise. This is why even a perfect expert (accuracy=1.0) does not improve performance of GES/LGES Safe/LGES Cons in our above experiments - algorithms may reject the expert's correct suggestions because noisy score estimates make incorrect structures appear favorable.

Consistency of our results with results from prior work This interpretation of our empirical results aligns with the setup of experimental results presented in (Ejaz and Bareinboim, 2025). For the majority of experiments (Figures 4a, 4b, 5b in (Ejaz and Bareinboim, 2025)) the authors use n=10,000 samples (2 orders of magnitude more data than the n=100 samples we draw for our experiments). To demonstrate the advantage of their method, their setup focuses primarily on high-dimensional settings with these large sample sizes, where search space navigation now becomes the relevant bottleneck, rather than error from finite samples. Additionally, note that in Figure 4a of (Ejaz and Bareinboim, 2025), the confidence intervals around the point estimates for each method are highly overlapping for moderate graph sizes, only strongly diverging around d=75 variables, indicating that the guidance from experts can have limited impact on score-based methods in some regimes. This aligns with our experimental results, as the standard deviations in our table (5% for different methods) indicate substantially overlapping confidence intervals across the score-based methods. The apparent ranking of no prior as better than priority or init likely reflects statistical noise rather than meaningful performance differences.

Verifying our implementation The authors of (Ejaz and Bareinboim, 2025) report results for GES, LGES Safe, and LGES Cons with no guidance (i.e., no prior) on the full discrete Sachs dataset (see Appendix D.4 in (Ejaz and Bareinboim, 2025)). To help verify our software implementation, we cross-checked whether running the score-based methods on the full Sachs dataset (rather than subsampling) using our code gave the same results as reported in (Ejaz and Bareinboim, 2025). Our results exactly match the findings of (Ejaz and Bareinboim, 2025); in both (Ejaz and Bareinboim, 2025) and our results, all algorithms find the same skeleton, which is missing 9 edges and contains no false edges (yielding an F1 score of 0.64 for skeleton prediction).