Breaking Bad: Component-Wise Parent Deletion for Score-Based Causal Discovery

Min Woo Park[*1] **Taehui Yun**[*1] **YoungIn Jang**[1] **Yoonseok Yeom**[1] **Jonghwan Kim**[1] **Jiyeon Kang**[2]
Songseong Kim[2] **Hyemin Jung**[2] **Sangmin Lee**[2] **Jongseong Jang**[†2] **Sanghack Lee**[†1,3]

[1]Graduate School of Data Science, Seoul National University, Republic of Korea
[2] LG AI Research, Republic of Korea
[3]SNU-LG AI Research Center, Republic of Korea

Abstract

Directed acyclic graphs (DAGs) are widely used to represent complex causal relationships in real-world systems. The goal of causal discovery is to learn the underlying DAG from data generated by these systems. While *Greedy Equivalence Search* (GES) is a well-established score-based algorithm for causal discovery, the GES family often suffers from scalability and sample complexity issues due to its large search space and susceptibility to local optima. In this paper, we introduce *parent deletion*, a novel, simple, yet powerful operator for score-based causal discovery. This operator is theoretically sound and effectively alleviates these limitations. Moreover, its simplicity enables seamless compatibility with existing score-based methods, and extensive experiments demonstrate consistent improvements across a wide range of settings. The source code used in this study is available at https://github.com/LGAI-Research/breaking-bad.

1 INTRODUCTION

Causal discovery refers to the problem of inferring causal structures from data and lies at the heart of causal inference research [Pearl, 2000, Glymour et al., 2019]. *Greedy Equivalence Search* (GES) [Meek, 1997, Chickering, 2002a,b] is a cornerstone algorithm that learns a causal structure in the form of a *Markov Equivalence Class* (MEC) [Pearl, 1988, Spirtes et al., 2000] based on a proper scoring rule (e.g., BIC [Schwarz, 1978] and BDeu [Heckerman et al., 1995]). Since the true causal graph is generally not uniquely identifiable from observational data, the MEC represents the most informative structure that can be recovered. GES relies on a greedy local search over the space of MECs, applying modifications (e.g., edge insertions and deletions) to intermediate graphs guided by a scoring function. At each step, the algorithm transitions to the best-scoring neighbor and terminates at a local optimum of the score. In the large-sample limit, this strategy is guaranteed to recover the global optimum of the score, corresponding to the true MEC.

Despite its theoretical appeal, GES faces practical challenges. First, it does not readily scale to high-dimensional or dense settings, as the search space grows exponentially with structural complexity. Several variants of GES have been developed, including approaches restricting the search to a bounded parent set size [Chickering and Meek, 2015, Chickering, 2020], and parallelized implementations [Ramsey et al., 2017]. While continuous relaxations based on gradient-based optimization [Zheng et al., 2018, Ng et al., 2020, Bello et al., 2022] have been proposed, they have raised empirical and conceptual concerns and lack rigorous theoretical convergence guarantees [Reisach et al., 2021, Ng et al., 2024, Wienöbst et al., 2026].

Furthermore, GES often fails to recover the true MEC in finite-sample regimes, as sampling variability can induce numerous local optima across the search space [Nielsen et al., 2003]. In particular, when the algorithm becomes trapped in a local optimum with a dense structure, score evaluation becomes unreliable because large parent sets leave insufficient data for accurate estimation [Chickering, 2020]. To alleviate this issue, recently proposed methods employ heuristics such as prioritizing deletion operators to avoid dense intermediate structures (XGES-0) [Nazaret and Blei, 2024], or restricting edge insertions to be more conservative (LGES) [Ejaz and Bareinboim, 2025].

Motivating experiments. Fig. 1 compares the number of parents in the true causal structure with those inferred by each algorithm. Specifically, we consider GES [Chickering, 2002b] and its variants (XGES-0 [Nazaret and Blei, 2024], LGES [Ejaz and Bareinboim, 2025]) and an order-based algorithm, BOSS [Andrews et al., 2023], which traverses the

[*]Equal contribution.
[†]Corresponding authors: j.jang@lgresearch.ai, sanghack@snu.ac.kr.

space of topological orders by maximizing a score function. The results demonstrate that score-based methods tend to introduce an excessive number of parents, even when employing conservative insertion strategies (LGES) or deletion-priority heuristics (XGES-0). This observation motivates the need for a metaheuristic that enables the algorithm to escape dense local optima by incorporating a factor that explicitly promotes sparser structures.

Contributions. In this paper, we investigate the expansion of the neighborhood of local search to reach better solutions. To this end, we apply heuristic perturbations to the current local optimum, namely, *parent deletion*. Surprisingly, this simple perturbation operator enables faster escape from local optima with low computational overhead, leading to notable speedups and substantial performance gains, without sacrificing theoretical guarantees.

The remainder of this paper is organized as follows. Secs. 2 and 3 provide the necessary background and discuss related work. In Sec. 4, we formally define the parent deletion operator, starting with a single-node deletion and extending it to a component-wise formulation with complete validity conditions. Finally, Sec. 5 presents extensive experimental results, which corroborate our theoretical findings.

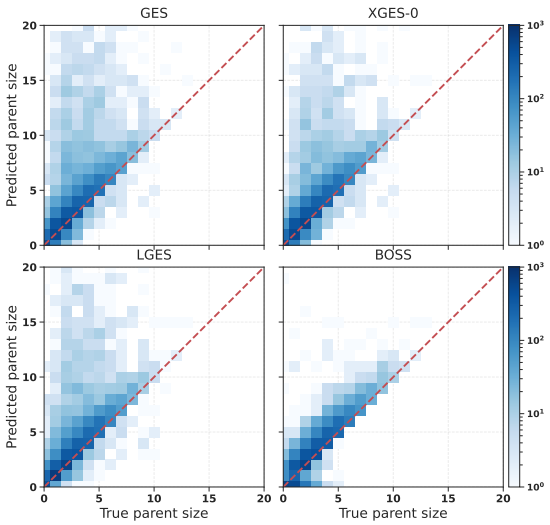


Figure 1: True parent set size in the ground-truth DAG (x-axis) versus parent set size in the learned CPDAGs (y-axis). The data are generated from linear-Gaussian models based on underlying Erdős–Rényi DAGs with 200 nodes, average degree 5, and 1,000 samples (30 DAGs in total). Across all methods, the predicted parent set sizes are biased toward larger values, indicating a general tendency of score-based approaches to overestimate parent set sizes.

2 PRELIMINARIES

We introduce necessary notation and background concepts. Following conventions, we use a capital letter, such as X, to represent a variable. Boldface is employed to represent a set of variables, denoted by $\mathbf{X}$.

A causal graph [Pearl, 2000, Bareinboim et al., 2022] is a *directed acyclic graph* (DAG) $\mathcal{G} = \langle \mathbf{V}, \mathbf{E} \rangle$ where an edge $V_i \to V_j \in \mathbf{E}$ denotes a direct causal link from V_i to V_j. If there is an edge between two vertices X and Y in $\mathcal{G}$, we say that the two nodes are *adjacent* in $\mathcal{G}$, denoted by $X \in \mathtt{Adj}_Y^{\mathcal{G}}$ or $Y \in \mathtt{Adj}_X^{\mathcal{G}}$. A variable X is a *parent* of Y in $\mathcal{G}$ if they are adjacent and the edge is directed into Y (and Y is a child of X). The sets of parents and children of X in $\mathcal{G}$ are denoted by $\mathtt{Pa}_X^{\mathcal{G}}$ and $\mathtt{Ch}_X^{\mathcal{G}}$, respectively. For a set of variables, we define $\mathtt{Pa}_{\mathbf{X}}^{\mathcal{G}} = \bigcup_{X \in \mathbf{X}} \mathtt{Pa}_X^{\mathcal{G}}$ and similarly for other relationships. A given distribution $p(\mathbf{V})$ is said to be *Markov* with respect to a DAG $\mathcal{G}$ if for all disjoint sets $\mathbf{X}, \mathbf{Y}, \mathbf{Z} \subseteq \mathbf{V}$, $\mathbf{X} \perp\!\!\!\perp_{\mathcal{G}} \mathbf{Y} \mid \mathbf{Z}$ implies $\mathbf{X} \perp\!\!\!\perp_p \mathbf{Y} \mid \mathbf{Z}$. If the converse is also true, $p(\mathbf{V})$ is said to be *faithful* to $\mathcal{G}$ [Spirtes et al., 2000].

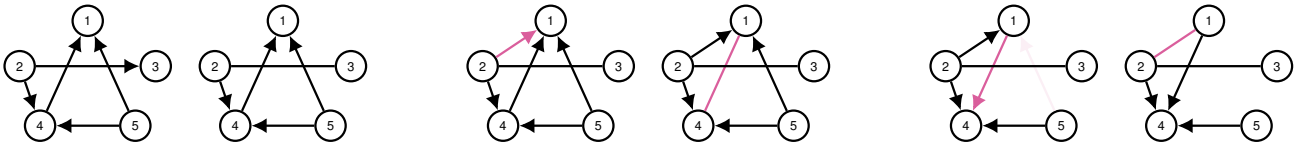
In this work, we adopt the standard assumptions of (i) *causal sufficiency*, i.e., the true DAG $\mathcal{G}^*$ generating $p(\mathbf{V})$ contains no unobserved confounders, and (ii) $p(\mathbf{V})$ is both Markov and faithful with respect to $\mathcal{G}^*$.

Markov equivalence. DAGs that induce the same conditional independencies are called *Markov equivalent* and they form a Markov Equivalence Class (MEC). We use $\mathcal{E}$ to denote a Markov equivalence class. To denote a particular equivalence class to which a graph $\mathcal{G}$ belongs, we write $\mathcal{E}(\mathcal{G})$. The *skeleton* is the undirected graph obtained by ignoring the directionality of all edges. A *v-structure* is an ordered triple of nodes $\langle X, Y, Z \rangle$ such that (i) $X \to Y$ and $Z \to Y$ are present, and (ii) X and Z are not adjacent. The skeleton and v-structures play a crucial role in characterizing Markov equivalence.

Theorem 1 (Theorem 1 in Verma and Pearl [1991])**.** *Two DAGs are Markov equivalent if and only if they have the same skeleton and the same v-structures.*

A *partially directed acyclic graph* (PDAG) $\mathcal{P}$ is a graph that contains both directed and undirected edges, and can be used to represent an equivalence class of DAGs. If a DAG $\mathcal{G}$ has the same skeleton and the same set of v-structures as a PDAG $\mathcal{P}$ (i.e., $\mathcal{E}(\mathcal{G}) = \mathcal{E}(\mathcal{P})$) and if every directed edge in $\mathcal{P}$ has the same orientation in $\mathcal{G}$, we say that $\mathcal{G}$ is a *consistent extension* of $\mathcal{P}$, and denote this by $\mathcal{G} \in [\mathcal{P}]$.

An edge $X \to Y$ in a PDAG $\mathcal{P}$ is *compelled* if that edge exists in every DAG that is equivalent to $\mathcal{P}$. If an edge $X \to Y$ in $\mathcal{P}$ is not compelled, we say that it is *reversible*. If there is at least one consistent extension of a PDAG $\mathcal{P}$, we say that $\mathcal{P}$ *admits* a consistent extension. We use *completed PDAGs* (CPDAGs) $\mathcal{P}^c$ to represent equivalence classes of DAGs [Andersson et al., 1997]. The CPDAG is the PDAG that represents an MEC, with directed edges corresponding to compelled edges and undirected edges corresponding to



(a) $\mathcal{G}$ (b) $\mathcal{P}^c$ (c) INSERT$(2, 1, \mathbf{T} = \emptyset)$ (d) DELETE$(5, 1, \mathbf{H} = \{4\})$

Figure 2: All changes are highlighted in pink. (a, b) A DAG and its corresponding CPDAG representing an MEC $\mathcal{E}(\mathcal{G})$. (c) The left PDAG illustrates the resulting graph after applying INSERT$(2, 1, \mathbf{T} = \emptyset)$. The operator inserts $2 \to 1$. This modification induces $4 \to 1$ to become reversible; thus, $4 - 1$ appears in the corresponding CPDAG (right). (d) The resulting graph after applying DELETE$(5, 1, \mathbf{H} = \{4\})$. The operator deletes $5 \to 1$ and generates a new v-structure $5 \to 4 \leftarrow 1$. Again, this change induces $2 - 1$ in the corresponding CPDAG.

reversible edges. For instance, Figs. 2a and 2b show a DAG and its corresponding CPDAG, respectively. Thus, $2 \to 3$ in Fig. 2a is reversible, whereas all other edges are compelled.

Score-based causal discovery. Score-based methods assign a score $S(\mathcal{G}; \mathbf{D})$ to a DAG $\mathcal{G}$ given i.i.d. data $\mathbf{D}$ sampled from $p(\mathbf{V})$. The score function is designed to be maximized for the true DAG $\mathcal{G}^*$. We consider the most commonly used scoring criterion, the Bayesian Information Criterion (BIC) [Schwarz, 1978], and assume that the data $\mathbf{D}$ are generated from a linear-Gaussian model. Under this assumption, the BIC score can be written as follows:

$$S(\mathcal{G}; \mathbf{D}) = \log p_{\hat{\boldsymbol{\theta}}}(\mathbf{D}; \mathcal{G}) - \frac{\lambda}{2}|\boldsymbol{\theta}| \log n, \tag{1}$$

where λ is a hyperparameter, n denotes the number of samples, $\hat{\boldsymbol{\theta}}$ is the maximum likelihood estimate of the model parameters, and $|\boldsymbol{\theta}|$ denotes the number of free parameters.

The BIC is consistent and, moreover, satisfies *score-equivalence* and *decomposability* for curved exponential families [Haughton, 1988, Geiger et al., 2001].

Definition 1 (Score-equivalence [Chickering, 2002b])**.** A scoring criterion is *score-equivalent* if the score function assigns the same score to all Markov equivalent DAGs.

By score-equivalence, we overload notation and write $S(\mathcal{P}^c; \mathbf{D})$ to denote the score of any DAG in $[\mathcal{P}^c]$.

Definition 2 (Decomposability [Chickering, 2002b])**.** A scoring criterion S is *decomposable* if it can be written as:

$$S(\mathcal{G}; \mathbf{D}) = \sum_{V \in \mathbf{V}} s(V, \mathtt{Pa}_V^{\mathcal{G}}; \mathbf{D}) \tag{2}$$

where each *local score* s is a function of V and $\mathtt{Pa}_V^{\mathcal{G}}$.

In particular, the consistency and decomposability of the BIC imply an attractive property known as *local consistency*, which plays a key role in the correctness of GES.

Definition 3 (Local consistency [Chickering, 2002b])**.** Let $\mathcal{G}$ be a DAG without an edge between X and Y, and let $\mathcal{G}'$ be the DAG resulting from adding the directed edge $X \to Y$. A scoring criterion S is *locally consistent* if, as n goes to infinity, the following conditions hold:

(i) If $X \not\perp\!\!\!\perp_p Y \mid \mathtt{Pa}_Y^{\mathcal{G}}$, then $S(\mathcal{G}; \mathbf{D}) < S(\mathcal{G}'; \mathbf{D})$.

(ii) If $X \perp\!\!\!\perp_p Y \mid \mathtt{Pa}_Y^{\mathcal{G}}$, then $S(\mathcal{G}; \mathbf{D}) > S(\mathcal{G}'; \mathbf{D})$.

Intuitively, adding an edge improves the score exactly when it eliminates a conditional independence that is false in p; otherwise, the additional edge is penalized.

Greedy Equivalence Search. GES [Chickering, 2002b] is a two-phase greedy search over the space of MECs (i.e., CPDAGs). The algorithm represents search states as CPDAGs and traverses the space by applying operators to these graphs, maximizing a scoring criterion given samples.

In the first (forward) phase, GES iteratively applies INSERT operators that lead to the greatest improvement in the score, continuing until no insertion yields a higher score. The operator INSERT$(X, Y, \mathbf{T})$ inserts the edge $X \to Y$ and directs previously undirected edges $T - Y$ as $T \to Y$ for every $T \in \mathbf{T}$, such that the nodes in $\mathbf{T}$ become the *tails* of v-structures $T \to Y \leftarrow X$.

In the second (backward) phase, the algorithm greedily applies DELETE operators to further improve the score, again terminating when no deletion increases the score. The operator DELETE$(X, Y, \mathbf{H})$ deletes an edge $X - Y$ or $X \to Y$ and directs previously undirected edges $X - H$ as $X \to H$ and $Y - H$ as $Y \to H$ for every $H \in \mathbf{H}$, such that the nodes in $\mathbf{H}$ become the *heads* of new v-structures $X \to H \leftarrow Y$. After applying an operator to a CPDAG, the resulting PDAG need not be a CPDAG; converting it takes $\mathcal{O}(|\mathbf{V}|^3)$ [Wienöbst et al., 2021a]. For instance, in Figs. 2c and 2d (left), the results of the operators are not CPDAGs.

The INSERT/DELETE operators correspond precisely to all single-edge insertions/deletions across *all* DAG members of the current CPDAG. If the PDAG that results from an operator admits a consistent extension, we say that the operator is *valid*. The formal definitions of the operators and the complete validity conditions are deferred to Appendix A.

3 RELATED WORK

Learning the causal structure underlying a data-generating process from observational data is a central problem in

Algorithm 1: Score-based method with ILS

Input: Score-based method(s) SEARCH; Perturbation strategy PERTURB; Data $\mathbf{D}$; score function S

1 $\mathcal{P}_0 \leftarrow \text{SEARCH}_1(\mathbf{D}, S, \mathcal{P}_0 = \emptyset)$ ▷ initial local optimum
2 $\mathcal{P}^* \leftarrow \mathcal{P}_0$
3 **repeat**
4 $\quad \mathbb{P} \leftarrow \text{PERTURB}(\mathcal{P}^*)$ ▷ enumerate perturbations
5 $\quad$ **foreach** $\mathcal{P}' \in \mathbb{P}$ **do**
6 $\quad\quad \mathcal{P} \leftarrow \text{SEARCH}_2(\mathbf{D}, S, \mathcal{P}')$ ▷ re-optimize
7 $\quad\quad$ **if** $S(\mathcal{P}; \mathbf{D}) > S(\mathcal{P}^*; \mathbf{D})$ **then**
8 $\quad\quad\quad \mathcal{P}^* \leftarrow \mathcal{P}$
9 $\quad\quad\quad$ **break** ▷ restart from updated $\mathcal{P}^*$
10 **until** terminated
11 **return** A CPDAG $\mathcal{P}^*$

causal discovery [Glymour et al., 2019]. In this work, we focus on score-based causal discovery approaches. Although exact methods [De Campos and Ji, 2011, Yuan and Malone, 2013, Parviainen and Koivisto, 2013] can be used to carry out this task, they are limited to a small number of variables due to the NP-hardness of identifying the highest-scoring DAG [Chickering, 1996, Chickering et al., 2004]. For this reason, local search methods are widely used.

DAG space local search. The most intuitive local search algorithm is greedy hill climbing over DAG space [Heckerman et al., 1995, Friedman et al., 1999, Tsamardinos et al., 2006]. Starting from an initial DAG, the method greedily modifies edges until no further improvement in the score is possible. In general, algorithms operating directly over DAG space are less effective than those over MEC space, due to the strict acyclicity constraint and the large size of the search space [Chickering, 2002a].

CPDAG space local search. GES [Chickering, 2002b] maximizes a scoring criterion via greedy search over the space of CPDAGs and is guaranteed to recover a CPDAG with an optimal score in the large-sample limit. Numerous extensions and variants of GES have been proposed [Alonso-Barba et al., 2013, Chickering and Meek, 2015, Ramsey et al., 2017, Nandy et al., 2018, Chickering, 2020]. In practice, however, from a purely optimization perspective, the GES family is inevitably susceptible to local optima when applied to real data. Empirical benchmarks [Rios et al., 2025] indicate that the best-performing heuristics either facilitate escaping from local optima—for example, through metaheuristic strategies [Alonso et al., 2018, Kuipers et al., 2022, Liu et al., 2023]—or expand the effective neighborhood explored during search [Hauser and Bühlmann, 2012, Nazaret and Blei, 2024, Ejaz and Bareinboim, 2025].

Iterated local search. Various metaheuristic schemes operating directly on the DAG search space have been proposed to escape local optima, including tabu search [Acid and De Campos, 2003], simulated annealing [Chickering et al., 1995], the greedy randomized adaptive search procedure [De Campos et al., 2002], and iterated local search [De Campos et al., 2003].

Table 1: Comparison of key components across methods.

Algorithm	Key strategy
ILGES	Random operators + noisy score function
GESSS	Local search over a sampled DAG
XGES	DELETE + deletion-priority
LGES+	DELETE + conservative insertion
Ours	DELETEPA over components

Iterated local search (ILS) [Lourenço et al., 2003, 2018] repeatedly applies local search starting from perturbed versions of locally optimal solutions. By escaping the current basin of attraction through controlled perturbations and subsequently restarting local optimization, ILS enables exploration of diverse regions of the search space. The pseudocode for ILS is presented in Alg. 1. Its effectiveness largely depends on the design of the perturbation strategy (Line 4): perturbations that are too weak leave the search confined to the same basin of attraction, whereas overly strong perturbations reduce the search to random restarts[1].

Table 1 summarizes the most closely related methods. ILGES [Alonso et al., 2018] introduces perturbations through randomly sampled operations (insertions and deletions) guided by a noisy scoring function, while GESSS [Liu et al., 2023] perturbs the search by iteratively switching between CPDAG and DAG search spaces. Motivated by the observation that score-based methods tend to produce graphs with large parent sets, XGES [Nazaret and Blei, 2024] employs a perturbation strategy based solely on DELETE operations, and LGES+ [Ejaz and Bareinboim, 2025] adopts a similar approach. Building on this line of work, we propose a perturbation strategy that explicitly leverages the empirical observation that score-based methods are prone to enlarging parent sets.

4 ITERATED LOCAL SEARCH FOR SCORE-BASED METHODS

For a node $X \in \mathbf{V}$ in a PDAG $\mathcal{P}$, let $\Pi_X^{\mathcal{P}} = |\mathtt{Pa}_X^{\mathcal{P}}|$ denote the *parent size*, i.e., the number of parents of X in $\mathcal{P}$. Under a decomposable scoring criterion (Def. 2), local scores $s(V, \mathtt{Pa}_V^{\mathcal{P}^c}; \mathbf{D})$ depend only on the node and its parent set, with their finite-sample variability governed by the parent size $\Pi_V^{\mathcal{P}^c}$. In particular, the parent sizes in a CPDAG provide tight lower bounds on the parent sizes across all DAGs in an equivalence class, i.e., $\Pi_V^{\mathcal{G}} \geq \Pi_V^{\mathcal{P}^c}$ for every $V \in \mathbf{V}$ and

[1]The SEARCH_1 and SEARCH_2 procedures need not be identical; for instance, one may use XGES-0 for initial search and GES for subsequent local search.

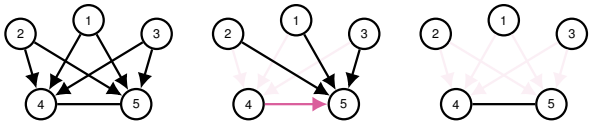


Figure 3: (a) A local structure of a CPDAG containing five nodes. (b, c) Parent deletion on a single node 4, and on the undirected component $\{4, 5\}$.

$\mathcal{G} \in [\mathcal{P}^c]$. Consequently, any strictly larger parent size than that in the true DAG implies that the corresponding Markov equivalence class *cannot* contain the true underlying DAG.

4.1 PARENT DELETION OPERATOR

We first formally define the operator $\textsc{DeletePa}(X, \mathbf{H})$, which removes *all* edges between a node X and its parents $\texttt{Pa}_X^{\mathcal{P}^c}$ in a given CPDAG $\mathcal{P}^c$. We say that Y is a *neighbor* of X in a PDAG $\mathcal{P}$, denoted $Y \in \texttt{Ne}_X^{\mathcal{P}}$, if X and Y are connected by an undirected edge in $\mathcal{P}$.

Definition 4 (Parent deletion)**.** Given a CPDAG $\mathcal{P}^c$, let X be a node in $\mathcal{P}^c$ with at least one parent, and let $\mathbf{H}$ be a subset of $\texttt{Ne}_X^{\mathcal{P}^c}$. The operator $\textsc{DeletePa}(X, \mathbf{H})$ modifies $\mathcal{P}^c$ by (i) deleting *all* edges from $\texttt{Pa}_X^{\mathcal{P}^c}$ to X and (ii) orienting each undirected edge between X and H as $X \to H$ for every $H \in \mathbf{H}$.

For concreteness, consider the subgraph of a CPDAG shown in Fig. 3a. The operator $\textsc{DeletePa}(4, \mathbf{H} = \{5\})$ first removes all directed edges from the parents of 4, and then directs the undirected edge between 4 and $5 \in \mathbf{H}$. This operation creates new v-structures mediated by $\mathbf{H}$, as depicted in Fig. 3b. Note that, similar to INSERT and DELETE, the graph resulting immediately after parent deletion is not necessarily a CPDAG.

Proposition 1 (Validity of parent deletion)**.** *The operator* $\textsc{DeletePa}(X, \mathbf{H})$ *is valid for* $\mathcal{P}^c$ *if and only if* $\texttt{Ne}_X^{\mathcal{P}^c} \setminus \mathbf{H}$ *is a clique*[2].

Corollary 1 (Increase in local score)**.** *For any score-equivalent decomposable scoring criterion* S*, the increase in score resulting from applying a valid operator* $\textsc{DeletePa}(X, \mathbf{H})$ *to a CPDAG* $\mathcal{P}^c$ *is:*

$$\delta \triangleq s(X, \texttt{Ne}_X^{\mathcal{P}^c} \setminus \mathbf{H}; \mathbf{D}) - s(X, \texttt{Pa}_X^{\mathcal{P}^c} \cup (\texttt{Ne}_X^{\mathcal{P}^c} \setminus \mathbf{H}); \mathbf{D}). \tag{3}$$

All omitted proofs are provided in Appendix D. At first glance, $\textsc{DeletePa}(X, \mathbf{H})$ appears computationally attractive, since its validity condition depends solely on the local structure around the target node X, rather than on the individual parent nodes in $\texttt{Pa}_X^{\mathcal{P}^c}$ whose incident edges are removed. As a consequence, checking its validity incurs the same computational cost $\mathcal{O}(|\texttt{Ne}_X^{\mathcal{P}^c} \setminus \mathbf{H}|^2)$ as that of verifying a single-edge deletion[3], despite removing multiple parent edges simultaneously. However, this seemingly efficient approach masks a fundamental limitation, namely, the effects of $\textsc{DeletePa}(X, \mathbf{H})$ are not confined to the target node.

Collateral parental growth. While $\textsc{DeletePa}(X, \mathbf{H})$ efficiently reduces the parent size of a target variable, it may increase the parent sizes of other nodes as a side effect. In particular, nodes in $\mathbf{H}$ acquire additional parents by participating in newly induced v-structures. To formalize this, we introduce the following entanglement property for a pair of nodes connected by an undirected edge in a CPDAG.

Lemma 1 (Lemma 21 in Chickering [2002a])**.** *If* $X - Y$ *is an undirected edge in a CPDAG* $\mathcal{P}^c$*, then* $\texttt{Pa}_X^{\mathcal{P}^c} = \texttt{Pa}_Y^{\mathcal{P}^c}$.

To see this, consider the subgraph of a CPDAG $\mathcal{P}^c$ in Fig. 3a, and suppose $\texttt{Pa}_4^{\mathcal{P}^c} = \{1, 2, 3\}$. Since 5 is connected by an undirected edge with 4, it follows that $\texttt{Pa}_5^{\mathcal{P}^c} = \{1, 2, 3\}$.

We now recall the operation $\textsc{DeletePa}(4, \mathbf{H} = \{5\})$, under which all parental edges of node 4 are removed. Meanwhile, node 5 acquires node 4 as an additional parent, expanding its original parent set $\texttt{Pa}_5^{\mathcal{P}^c} = \{1, 2, 3\}$ to $\texttt{Pa}_5^{\mathcal{P}^{c\prime}} = \{1, 2, 3, 4\}$. Therefore, an operation applied to a single node can induce collateral parental growth in other nodes. As a result, contrary to the intended effect of reducing parent sizes, the local score evaluation for node 5 becomes more unstable under finite samples.

Corollary 2. *Let* $\mathcal{P}^c$ *be a CPDAG and* $\mathcal{P}^{c\prime}$ *be the PDAG obtained by applying* $\textsc{DeletePa}(X, \mathbf{H})$ *to* $\mathcal{P}^c$*. Then,* $\Pi_X^{\mathcal{P}^c} < \Pi_H^{\mathcal{P}^{c\prime}}$ *for every* $H \in \mathbf{H}$.

Proof. Since $\mathbf{H} \subseteq \texttt{Ne}_X^{\mathcal{P}^c}$, it follows from Lem. 1 that $\texttt{Pa}_X^{\mathcal{P}^c} = \texttt{Pa}_H^{\mathcal{P}^c}$ for any $H \in \mathbf{H}$. Furthermore, by the definition of $\textsc{DeletePa}(X, \mathbf{H})$, the operator preserves all directed edges from $\texttt{Pa}_H^{\mathcal{P}^c}$ to H and additionally introduces $X \in \texttt{Pa}_H^{\mathcal{P}^{c\prime}}$ whenever $\mathbf{H} \neq \emptyset$. Therefore, it follows that $\Pi_X^{\mathcal{P}^c} < \Pi_H^{\mathcal{P}^{c\prime}}$. □

This phenomenon is not unique to $\textsc{DeletePa}(X, \mathbf{H})$; it also arises in the ordinary deletion operation $\textsc{Delete}(P, X, \mathbf{H})$ for $P \in \texttt{Pa}_X^{\mathcal{P}^c}$ since X becomes a parent of nodes in $\mathbf{H}$ by construction. For example, revisit the CPDAG $\mathcal{P}^c$ shown on the right in Fig. 2c, where $\Pi_4^{\mathcal{P}^c} = 2$. After applying $\textsc{Delete}(5, 1, \mathbf{H} = \{4\})$, the resulting graph $\mathcal{P}^{c\prime}$ (on the left in Fig. 2d) has $\Pi_4^{\mathcal{P}^{c\prime}} = 3$. Thus, to break entangled relations induced by shared parents, it is natural to consider deleting parent edges for all nodes connected by undirected edges, *all at once*, rather than one at a time.

[2] In a graph, a clique is a set of pairwise adjacent vertices.

[3] For $P \in \texttt{Pa}_X^{\mathcal{P}^c}$, the operator $\textsc{Delete}(P, X, \mathbf{H})$ is valid *if and only if* $\texttt{Ne}_X^{\mathcal{P}^c} \setminus \mathbf{H}$ is a clique [Chickering, 2002b]. See Thm. 4 in Appendix A.

Algorithm 2: Enumerate perturbations using the parent deletion operator over undirected components

Input: A CPDAG $\mathcal{P}^c$

function PERTURB($\mathcal{P}^c$)
  Initialize $\mathbb{P} = \emptyset$.
  for each undirected component $\mathbf{U}_i \in \mathbb{U}^{\mathcal{P}^c}$ **do**
    $\mathcal{P}^{c\prime} \leftarrow$ a copy of $\mathcal{P}^c$ with DELETEPA($\mathbf{U}_i$)
    $\mathcal{Q} \leftarrow$ PDAG-TO-CPDAG($\mathcal{P}^{c\prime}$)
    Add $\mathcal{Q}$ to $\mathbb{P}$.
  return a set of CPDAGs $\mathbb{P}$

4.2 PARENT DELETION OVER UNDIRECTED COMPONENTS

To overcome the collateral parental growth induced by single-node parent deletions, we extend parent deletion to operate at the level of nodes connected by undirected edges. An *undirected component* is defined as a maximal connected component of the graph obtained by removing all directed edges, denoted by $\mathbf{U}^{\mathcal{P}}$ (or simply $\mathbf{U}$). Moreover, we denote by $\mathbb{U}^{\mathcal{P}} = \{\mathbf{U}_i\}_{i=1}$ the collection of all undirected components of the PDAG $\mathcal{P}$. By construction, all nodes within the same undirected component share the same parent set in the corresponding CPDAG, as established in Lem. 1.

This property motivates the introduction of a component-wise parent deletion operator. For a CPDAG $\mathcal{P}^c$, let DELETEPA($\mathbf{U}^{\mathcal{P}^c}$) denote the operator that removes *all* directed edges from the parents of every node in $\mathbf{U}^{\mathcal{P}^c}$.

One might surmise that such a deletion would require iteratively checking the validity condition in Prop. 1 for all $U \in \mathbf{U}^{\mathcal{P}^c}$. Somewhat surprisingly, this is unnecessary; component-wise parent deletion is always valid and incurs no additional validity-checking cost.

Theorem 2. DELETEPA($\mathbf{U}^{\mathcal{P}^c}$) *is valid for any CPDAG* $\mathcal{P}^c$.

By decomposability (Def. 2) and score-equivalence (Def. 1), the score change induced by an operator depends only on the local scores of nodes whose parent sets are modified. In particular, applying DELETEPA($\mathbf{U}$) reduces the parent set of each $U \in \mathbf{U}$ to the empty set, while leaving the local scores of all other nodes unchanged. This yields a simple closed-form expression for the resulting score improvement.

Corollary 3 (Increase in local score)**.** *For any score-equivalent decomposable scoring criterion S, the increase in score that results from applying a valid operator* DELETEPA($\mathbf{U}$) *to a CPDAG* $\mathcal{P}^c$ *is:*

$$\delta \triangleq \sum_{U \in \mathbf{U}} \left[s(U, \emptyset; \mathbf{D}) - s(U, \mathtt{Pa}_U^{\mathcal{P}^c}; \mathbf{D}) \right]. \tag{4}$$

Note that a positive value of the score increase δ does not necessarily imply that applying DELETEPA($U, \mathbf{H} = \emptyset$) individually to each $U \in \mathbf{U}$ yields a positive score increase.

While this may seem like overkill, it is not necessarily so, since score-based methods often favor insertion over deletion and subsequent search steps can recover useful edges incrementally. This tendency is reflected in Fig. 2, the deletion-priority strategy of XGES-0, and the conservative insertion strategy of LGES. Thus, deleting first and then reconstructing meaningful edges can be more practical than identifying and removing unnecessary edges one by one.

Component-wise parent deletion perturbation. After running SEARCH$_1$ in the main algorithm (Alg. 1), the subroutine PERTURB (Alg. 2) iterates over the operations DELETEPA($\mathbf{U}$) for all undirected components in the current CPDAG. For each operation, we create a copy of the current CPDAG, apply the parent deletion, convert the resulting PDAG to a CPDAG using PDAG-TO-CPDAG[4], and pass the resulting graph to the main routine. The main routine then resumes SEARCH$_2$ on the modified copy. If the resulting score is worse than that of the current CPDAG before perturbation, the copy is discarded and the algorithm proceeds to the next parent deletion. Otherwise, the copy replaces the current CPDAG, and the procedure restarts from this updated CPDAG, enumerating all newly available parent deletions. The algorithm terminates once all deletions associated with a given CPDAG have been evaluated.

Scheduling perturbation strength. Motivated by the classical ILS principle of gradually weakening perturbation strength, we additionally consider a multi-phase strategy. The procedure consists of three phases: (i) component-wise parent deletions DELETEPA($\mathbf{U}$); (ii) single-node parent deletions DELETEPA($*, \mathbf{H} = \emptyset$);[5] and (iii) milder perturbations via single-edge deletions DELETE($*, *, \mathbf{H} = \emptyset$). Note that we fix $\mathbf{H} = \emptyset$, motivated by the theoretical result of Cor. 2, which shows that choosing non-empty $\mathbf{H}$ leads to collateral parental growth. These phases correspond to progressively smaller neighborhood modifications. This scheduling stabilizes the search trajectory and empirically results in improved performance. We defer the theoretical discussion of our methodological design to Appendix C.

Correctness. The correctness of Alg. 1—its ability to identify the optimal solution in the large-sample limit—follows from the validity of the proposed operators and the correctness guarantee of the underlying search procedure. Once the operators are valid, correctness no longer depends on the perturbation regime; rather, it follows from the correctness of the underlying search algorithm.

[4] That is, first sample a consistent extension using PDAG-TO-DAG [Dor and Tarsi, 1992], and then obtain the corresponding CPDAG via DAG-TO-CPDAG [Chickering, 2002a].

[5] In the single-node phase, we consider only nodes with at least two parents, as one-parent cases reduce to single-edge deletions.

Table 2: Variant names are highlighted in gray. OPS [Chickering, 2002b] simultaneously considers deletion and insertion operations without prioritization. Since BOSS operates in the DAG space, we set SEARCH$_2$ for BOSS-based variants to GES. For all other methods, we use the same procedure for SEARCH$_1$ and SEARCH$_2$.

Family	Original baseline			Single-edge deletion, DELETE			DELETEPA over components			Scheduled DELETEPA + DELETE		
	Variant	SEARCH$_1$	SEARCH$_2$	Variant	SEARCH$_1$	SEARCH$_2$	Variant	SEARCH$_1$	SEARCH$_2$	Variant	SEARCH$_1$	SEARCH$_2$
GES	GES	GES	-	GES-D	GES	GES	GES-DP	GES	GES	GES-DP+	GES	GES
OPS	OPS	OPS	-	OPS-D	OPS	OPS	OPS-DP	OPS	OPS	OPS-DP+	OPS	OPS
LGES	LGES	LGES	-	LGES+	LGES	LGES	LGES-DP	LGES	LGES	LGES-DP+	LGES	LGES
XGES	XGES-0	XGES-0	-	XGES	XGES-0	XGES-0	XGES-DP	XGES-0	XGES-0	XGES-DP+	XGES-0	XGES-0
BOSS	BOSS	BOSS	-	BOSS-D	BOSS	GES	BOSS-DP	BOSS	GES	BOSS-DP+	BOSS	GES

5 EXPERIMENTS

After describing the experimental setup, we investigate how performance varies with respect to key factors (e.g., graph density and sample size) across various datasets.

5.1 EVALUATION SETUP

We first conduct experiments on Erdős–Rényi (ER) DAGs. We let d denote the number of nodes and let ρ denote the density factor (expected degree per node).

To evaluate performance under heterogeneous graph structures, we further consider scale-free DAGs, which naturally exhibit skewed degree distributions characterized by a few high-degree nodes. For each parameter setting, we generate 30 random DAGs.

Finally, to further validate our findings, we evaluate our method on four biological benchmark networks derived from realistic gene expression datasets [Scutari, 2010]: ARTH150 [Opgen-Rhein and Strimmer, 2007], ECOLI70 [Schäfer and Strimmer, 2005], MAGIC-IRRI [Bandillo et al., 2013], and MAGIC-NIAB [Mackay et al., 2014].

For each synthetic DAG generated under an ER or scale-free model, we draw n samples from linear-Gaussian structural equation models. For the four biological benchmarks, we use the DAG structures and all model parameters provided by `bnlearn` [Scutari, 2010]. That is, both the structures and all parameters are fixed. Further details on the data generation process are provided in Appendix E.1.

Evaluation metrics. We primarily evaluate performance using the Structural Hamming Distance (SHD) and the F1 score. SHD quantifies the distance between the CPDAG recovered by an algorithm and the true CPDAG, while the F1 score measures the harmonic mean of precision and recall. To assess the computational efficiency of the proposed method, we additionally report the runtime of each variant. We defer detailed descriptions of SHD, the F1 score, and the runtime measurement to Appendix E.2.

Baseline algorithms. Table 2 summarizes the search routines and perturbation strategies for each method variant. We consider score-based methods, including the GES family, as well as the order-based method BOSS. Since LGES has two variants, LGES (CONS) and LGES (SAFE), we evaluate both. When a variant applies parent deletion to undirected components (Alg. 2), we append the suffix -DP to the method name. Similarly, when we employ the three-phase scheduled perturbation strategy, we append the suffix -DP+. Detailed descriptions of all methods are provided in Appendix B.

5.2 MAIN EXPERIMENTAL RESULTS

Dense condition performance. We first examine the effect of the proposed perturbations by varying ρ while fixing $d = 100$, $n = 10^4$, and the BIC hyperparameter $\lambda = 2$.

As illustrated in Fig. 4, performance deteriorates as ρ increases. Notably, the degradation in F1 (middle) is primarily driven by a sharp decline in precision (bottom), indicating that denser graphs are more prone to over-insertion errors. In this regime, the DP and DP+ variants improve performance across all baselines. Moreover, these advantages become more pronounced as ρ increases, suggesting that the proposed perturbation strategies enable the search to escape local optima even in dense graphs.

The relatively smaller improvement observed for BOSS is due to the fact that BOSS is less affected by parent-size overestimation. As shown in Fig. 1, BOSS exhibits a smaller degree of parent-size inflation than the other methods and therefore has less room for improvement from perturbations.

Performance under varying sample sizes. Fig. 5 (left) shows the F1 of the GES variants as a function of the sample size $n \in \{10^2, 200, 500, 10^3, 10^4, 10^5, 10^6\}$ while fixing $\rho = 5$, $d = 100$, and $\lambda = 2$. For GES and GES-D, F1 scores do not increase monotonically and may even decrease, whereas the DP and DP+ variants exhibit steadily improving F1 scores as the sample size increases. Interestingly, when $n \leq 10^3$, GES-D outperforms GES-DP. We hypothesize that component-wise parent deletion may be too strong in landscapes where local optima are closely spaced. Overall, GES-DP+ outperforms all others across all sample sizes.

High-dimensional condition performance. Fig. 5 (right) shows the F1 scores of the GES variants as a function of

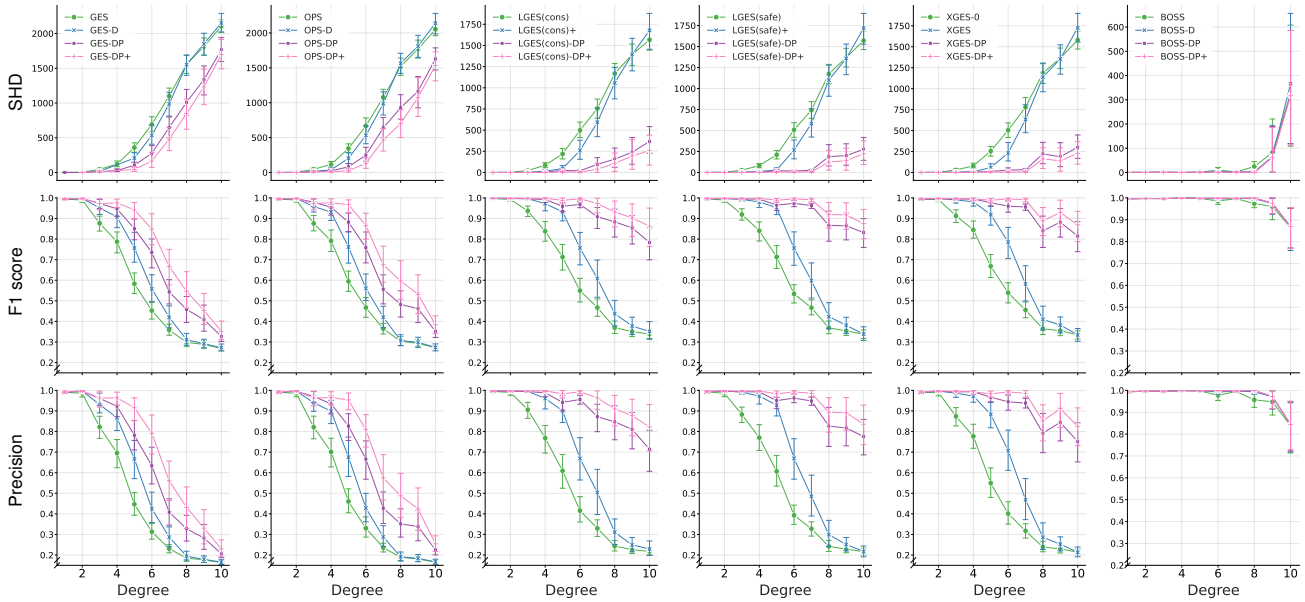


Figure 4: SHD(↓), F1 score(↑), and Precision(↑) for the GES, OPS, LGES (CONS), LGES (SAFE), XGES, and BOSS variants (from left to right) as the expected degree ρ varies. Colors indicate the four compared variant groups: **non-ILS** variants using only SEARCH_1, **single-edge deletion** variants, **DP** variants, and **DP+** variants. The vertical lines are computed as bootstrapped 95% confidence intervals over random seeds [Waskom, 2021]. The y-axes for F1 score and Precision are truncated below 0.2 for visibility.

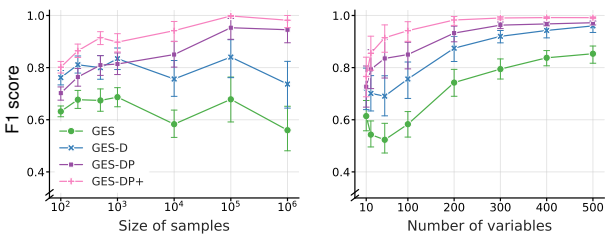


Figure 5: F1 scores as the sample size and the number of variables vary. The x-axis of the sample size plot (left) is on a log scale. The y-axes are truncated below 0.3 for visibility.

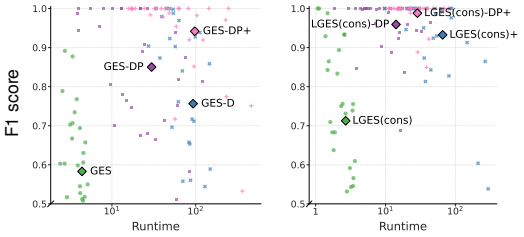
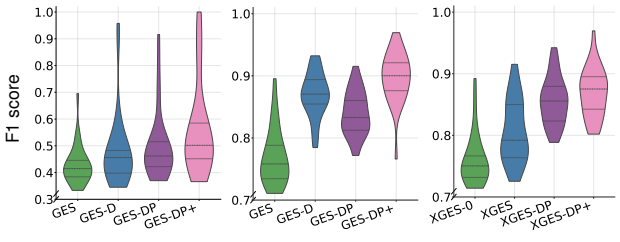


Figure 6: Runtime (sec) versus F1, with logarithmic x-axes. The y-axes are truncated at 0.5 for visibility. Points represent individual seeds, and diamonds (⋄) indicate mean performance.

the number of nodes d, as d varies from 10 to 500, while fixing $\rho = 5$, $n = 10^4$, and $\lambda = 2$. GES-DP and GES-DP+ outperform both GES and GES-D. Note that the increasing trend in F1 scores arises because, under a fixed ρ in the ER setting, the edge probability scales as $\frac{\rho}{d-1}$. Consequently, as the graph becomes relatively sparse, single-edge deletion variants may outperform DP variants due to their more localized perturbations, in line with our heuristic.

Runtime analysis. Fig. 6 illustrates runtime versus F1 score for GES and LGES variants. We set $d = 100$, $\rho = 5$, $n = 10^4$, and $\lambda = 2$. GES has the lowest runtime, as it does not incorporate the ILS procedure. GES-DP runs faster than GES-D, as expected, since (i) it avoids iterating over a potentially exponential number of $\mathbf{H}$ candidates based on Cor. 2, and (ii) its perturbation loop scales with the number of undirected components rather than edges, i.e., $\mathcal{O}(d)$ versus $\mathcal{O}(d^2)$. Although GES-DP+ incurs additional overhead due to its three-phase design, it has runtime comparable to GES-D. Similarly, LGES-DP+ is faster than LGES-D, despite the three-phase structure due to the same computational advantages of the DP perturbation. See Appendix E.3.5 for a detailed discussion of the runtime results.

Grid configurations performance. For a comprehensive analysis, we evaluate all variants listed in Table 2 across a total of 81 grid configurations, varying $d \in \{50, 100, 200\}$, $\rho \in \{2, 3, 5\}$, $n \in \{10^3, 10^4, 10^5\}$ and $\lambda \in \{1, 2, 4\}$. The DP+ variants consistently outperform other methods. Moreover, this performance gap becomes more pronounced in relatively dense regimes (i.e., $\rho = 5$). The DP variants consistently outperform all non-ILS methods. Compared with the single-edge deletion strategy, they usually match its performance and occasionally trail slightly, while offering



(a) Scale-free (b) ARTH150

Figure 7: Violin plots where the horizontal lines denote the 75_{th}, 50_{th}, and 25_{th} percentiles. The y-axes are truncated at 0.3 in (a) and 0.7 in (b), respectively.

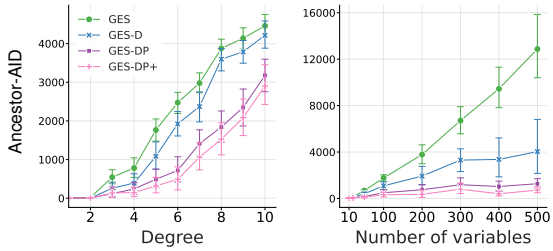


Figure 8: Ancestor-AID on ER DAGs with varying expected degree and varying number of variables. We set $n = 10^4$ and $\lambda = 2$ for both, with $d = 100$ on the left and $\rho = 5$ on the right.

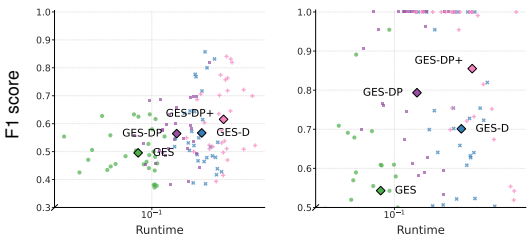
substantially reduced runtime. Overall, the proposed perturbation strategies demonstrate stable performance across diverse settings, with particularly strong advantages in dense graphs. Full results are reported in Appendix E.4.

Scale-free graph performance. We set $d = 50$, $m = 4$, $n = 10^3$, and $\lambda = 2$. As shown in Fig. 7a, the performance ranking GES-DP+ $>$ GES-DP $>$ GES-D $>$ GES is maintained across all percentiles. Further results for additional density settings across all baseline algorithms are provided in Appendix E.3.9.

Performance on biologically motivated benchmarks. We present results on ARTH150 in Fig. 7b as a representative biological benchmark, since it is the largest network among the four biological datasets. DP+ achieves the best performance for both GES and XGES. While GES-D slightly outperforms GES-DP, XGES-DP surpasses XGES. Although GES-DP occasionally exhibits marginally worse performance, it remains markedly faster. Additional results across all baseline algorithms on the four biological network benchmarks are provided in Appendix E.3.10. Overall, these representative findings illustrate that the proposed parent deletion perturbations remain effective in applied settings.

5.3 ADDITIONAL EXPERIMENTAL RESULTS

Evaluation with Adjustment Identification Distance. We additionally evaluate performance using the Adjustment Identification Distance (AID) [Henckel et al., 2024], which measures the dissimilarity between the learned and true graphs in terms of their implications for adjustment-based causal effect estimation. Among the AID variants, we report Ancestor-AID. As shown in Fig. 8, GES-DP+ and GES-DP consistently achieve lower Ancestor-AID scores than GES and GES-D. Further details on the metric and full results for all baselines are provided in Appendices E.2 and E.3.8.



(a) tanh-uniform (b) Linear-Gumbel

Figure 9: Runtime (sec) vs. F1 where the x-axes are on log scales. The y-axes are truncated at 0.3 in (a) and 0.5 in (b), respectively.

Performance under assumption violations. To evaluate the robustness of our method beyond the standard linear-Gaussian setting, we conducted additional experiments under (i) nonlinear (tanh) additive uniform noise models, and (ii) linear additive Gumbel noise models. As shown in Fig. 9, GES-DP+ consistently achieves the highest F1 across both settings, while GES-DP outperforms GES and GES-D. These results suggest that the proposed perturbation strategy remains effective even when the assumptions are violated. Full results for all baselines are provided in Appendix E.3.11.

6 CONCLUSION

We investigated how to effectively escape dense local optima in score-based causal discovery. To this end, we introduced *parent deletion*, a simple yet powerful perturbation operator. We first formalized single-node parent deletion and characterized its validity conditions, and then extended it to a component-wise formulation that operates on undirected components. This extension provably avoids collateral parental growth and incurs no additional validity-checking cost, while enabling substantially larger and more effective neighborhood exploration.

Future work. Future work includes investigating parent deletion beyond causal sufficiency [Zhang, 2006, 2008a,b], designing efficient states and operators for such settings [Claassen and Bucur, 2022], and extending our approach to MEC-size-aware perturbations [He et al., 2015, Wienöbst et al., 2021b, 2023, Wang et al., 2024, 2025, Liu et al., 2025], generalized score regimes [Huang et al., 2018], and MECs characterized by observational and interventional data [He and Geng, 2008, Hauser and Bühlmann, 2012].

Acknowledgements

This work was supported by LG AI Research. We thank the anonymous reviewers for constructive comments to improve the manuscript. This work was also supported by the IITP (RS-2022-II220953/50%) and NRF (RS-2023-00211904/50%) grants funded by the Korean government.

References

Silvia Acid and Luis M De Campos. Searching for Bayesian network structures in the space of restricted acyclic partially directed graphs. *Journal of artificial intelligence research*, 18:445–490, 2003.

Juan I Alonso, Luis de la Ossa, Jose A Gámez, and Jose M Puerta. On the use of local search heuristics to improve GES-based Bayesian network learning. *Applied Soft Computing*, 64:366–376, 2018.

Juan I Alonso-Barba, Luis delaOssa, Jose A Gámez, and Jose M Puerta. Scaling up the greedy equivalence search algorithm by constraining the search space of equivalence classes. *International journal of approximate reasoning*, 54(4):429–451, 2013.

Steen A Andersson, David Madigan, and Michael D Perlman. A characterization of Markov equivalence classes for acyclic digraphs. *The Annals of Statistics*, 25(2):505–541, 1997.

Bryan Andrews, Joseph Ramsey, Ruben Sanchez Romero, Jazmin Camchong, and Erich Kummerfeld. Fast scalable and accurate discovery of dags using the best order score search and grow shrink trees. *Advances in Neural Information Processing Systems*, 36:63945–63956, 2023.

Nonoy Bandillo, Chitra Raghavan, Pauline Andrea Muyco, Ma Anna Lynn Sevilla, Irish T Lobina, Christine Jade Dilla-Ermita, Chih-Wei Tung, Susan McCouch, Michael Thomson, Ramil Mauleon, et al. Multi-parent advanced generation inter-cross (magic) populations in rice: progress and potential for genetics research and breeding. *Rice*, 6(1):11, 2013.

Albert-László Barabási and Réka Albert. Emergence of scaling in random networks. *science*, 286(5439):509–512, 1999.

Elias Bareinboim, Juan D Correa, Duligur Ibeling, and Thomas Icard. On Pearl's hierarchy and the foundations of causal inference. In *Probabilistic and causal inference: the works of Judea Pearl*, pages 507–556. Association for Computing Machinery, 2022.

Kevin Bello, Bryon Aragam, and Pradeep Ravikumar. Dagma: Learning dags via m-matrices and a log-determinant acyclicity characterization. *Advances in Neural Information Processing Systems*, 35:8226–8239, 2022.

David Maxwell Chickering. Learning Bayesian networks is NP-complete. In *Learning from data: Artificial intelligence and statistics V*, pages 121–130. Springer, 1996.

David Maxwell Chickering. Learning equivalence classes of Bayesian-network structures. *Journal of machine learning research*, 2(Feb):445–498, 2002a.

David Maxwell Chickering. Optimal structure identification with greedy search. *Journal of machine learning research*, 3(Nov):507–554, 2002b.

David Maxwell Chickering. Statistically efficient greedy equivalence search. In *Proceedings of the Thirty-Sixth Conference on Uncertainty in Artificial Intelligence*, pages 241–249. Pmlr, 2020.

David Maxwell Chickering and Christopher Meek. Selective greedy equivalence search: finding optimal Bayesian networks using a polynomial number of score evaluations. In *Proceedings of the Thirty-First Conference on Uncertainty in Artificial Intelligence*, pages 211–219, 2015.

David Maxwell Chickering, Dan Geiger, and David Heckerman. Learning bayesian networks: Search methods and experimental results. In Doug Fisher and Hans-Joachim Lenz, editors, *Pre-proceedings of the Fifth International Workshop on Artificial Intelligence and Statistics*, volume R0 of *Proceedings of Machine Learning Research*, pages 112–128. PMLR, 04–07 Jan 1995. URL https://proceedings.mlr.press/r0/chickering95a.html. Reissued by PMLR on 01 May 2022.

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

Tom Claassen and Ioan G. Bucur. Greedy equivalence search in the presence of latent confounders. In James Cussens and Kun Zhang, editors, *Proceedings of the Thirty-Eighth Conference on Uncertainty in Artificial Intelligence*, volume 180 of *Proceedings of Machine Learning Research*, pages 443–452. PMLR, 01–05 Aug 2022. URL https://proceedings.mlr.press/v180/claassen22a.html.

Cassio P De Campos and Qiang Ji. Efficient structure learning of Bayesian networks using constraints. *The Journal of Machine Learning Research*, 12:663–689, 2011.

Luis M De Campos, Juan Manuel Fernández-Luna, and Jose Miguel Puerta. Local search methods for learning Bayesian networks using a modified neighborhood in the space of dags. In *Ibero-American Conference on Artificial Intelligence*, pages 182–192. Springer, 2002.

Luis M De Campos, Juan M Fernández-Luna, and J Miguel Puerta. An iterated local search algorithm for learning Bayesian networks with restarts based on conditional independence tests. *International Journal of Intelligent Systems*, 18(2):221–235, 2003.

Dorit Dor and Michael Tarsi. A simple algorithm to construct a consistent extension of a partially oriented graph. *Technicial Report R-185, Cognitive Systems Laboratory, UCLA*, page 45, 1992.

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

Peter I Frazier. A tutorial on bayesian optimization. *arXiv preprint arXiv:1807.02811*, 2018.

Nir Friedman, Iftach Nachman, and Dana Pe'er. Learning bayesian network structure from massive datasets: The "sparse candidate" algorithm. In *Proceedings of the Fifteenth Conference on Uncertainty in Artificial Intelligence*, pages 206–215, 1999.

Juan L Gamella, Armeen Taeb, Christina Heinze-Deml, and Peter Bühlmann. Characterization and greedy learning of gaussian structural causal models under unknown interventions. *arXiv preprint arXiv:2211.14897*, 2022.

Dan Geiger, David Heckerman, Henry King, and Christopher Meek. Stratified exponential families: graphical models and model selection. *Annals of statistics*, pages 505–529, 2001.

Clark Glymour, Kun Zhang, and Peter Spirtes. Review of causal discovery methods based on graphical models. *Frontiers in genetics*, 10:524, 2019.

Dominique MA Haughton. On the choice of a model to fit data from an exponential family. *The annals of statistics*, pages 342–355, 1988.

Alain Hauser and Peter Bühlmann. Characterization and greedy learning of interventional Markov equivalence classes of directed acyclic graphs. *The Journal of Machine Learning Research*, 13(1):2409–2464, 2012.

Yang-Bo He and Zhi Geng. Active learning of causal networks with intervention experiments and optimal designs. *Journal of Machine Learning Research*, 9(11), 2008.

Yangbo He, Jinzhu Jia, and Bin Yu. Counting and exploring sizes of Markov equivalence classes of directed acyclic graphs. *Journal of Machine Learning Research*, 16(79): 2589–2609, 2015.

David Heckerman, Dan Geiger, and David M Chickering. Learning Bayesian networks: The combination of knowledge and statistical data. *Machine learning*, 20(3):197–243, 1995.

Leonard Henckel, Theo Würtzen, and Sebastian Weichwald. Adjustment identification distance: A gadjid for causal structure learning. In Negar Kiyavash and Joris M. Mooij, editors, *Proceedings of the Fortieth Conference on Uncertainty in Artificial Intelligence*, volume 244 of *Proceedings of Machine Learning Research*, pages 1569–1598. PMLR, 15–19 Jul 2024. URL https://proceedings.mlr.press/v244/henckel24a.html.

Inki Hong, Andrew B Kahng, and Byung-Ro Moon. Improved large-step markov chain variants for the symmetric TSP. *Journal of Heuristics*, 3(1):63–81, 1997.

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 (KDD)*, pages 1551–1560. ACM, 2018. doi: 10.1145/3219819.3220104.

Jack Kuipers, Polina Suter, and Giusi Moffa. Efficient sampling and structure learning of Bayesian networks. *Journal of Computational and Graphical Statistics*, 31(3): 639–650, 2022.

Lifu Liu, Shiyuan He, and Jianhua Guo. An improved clique-picking algorithm for counting Markov equivalent dags via super cliques transfer. In *Forty-second International Conference on Machine Learning*, 2025.

Xiaohan Liu, Xiaoguang Gao, Xinxin Ru, Xiangyuan Tan, and Zidong Wang. Improving greedy local search methods by switching the search space. *Applied Intelligence*, 53(19):22143–22160, 2023.

Helena Ramalhinho Lourenço, Olivier C Martin, and Thomas Stützle. Iterated local search. In *Handbook of metaheuristics*, pages 320–353. Springer, 2003.

Helena Ramalhinho Lourenço, Olivier C Martin, and Thomas Stützle. Iterated local search: Framework and applications. In *Handbook of metaheuristics*, pages 129–168. Springer, 2018.

Ian J Mackay, Pauline Bansept-Basler, Toby Barber, Alison R Bentley, James Cockram, Nick Gosman, Andy J Greenland, Richard Horsnell, Rhian Howells, Donal M O'Sullivan, et al. An eight-parent multiparent advanced generation inter-cross population for winter-sown wheat: creation, properties, and validation. *G3: Genes, Genomes, Genetics*, 4(9):1603–1610, 2014.

Dimitris Margaritis and Sebastian Thrun. Bayesian network induction via local neighborhoods. *Advances in neural information processing systems*, 12, 1999.

Christopher Meek. Causal inference and causal explanation with background knowledge. In *Proceedings of the*

Eleventh conference on Uncertainty in artificial intelligence, pages 403–410, 1995.

Christopher Meek. *Graphical Models: Selecting causal and statistical models*. PhD thesis, Carnegie Mellon University, 1997.

Preetam Nandy, Alain Hauser, and Marloes H Maathuis. High-dimensional consistency in score-based and hybrid structure learning. *The Annals of Statistics*, 46(6A):3151–3183, 2018.

Achille Nazaret and David Blei. Extremely greedy equivalence search. In *Uncertainty in Artificial Intelligence*, pages 2716–2745. PMLR, 2024.

Ignavier Ng, AmirEmad Ghassami, and Kun Zhang. On the role of sparsity and dag constraints for learning linear dags. *Advances in Neural Information Processing Systems*, 33:17943–17954, 2020.

Ignavier Ng, Biwei Huang, and Kun Zhang. Structure learning with continuous optimization: A sober look and beyond. In *Causal Learning and Reasoning*, pages 71–105. PMLR, 2024.

Jens D. Nielsen, Tomáš Kočka, and Jose M. Peña. On local optima in learning Bayesian networks. In *Proceedings of the 19th Conference on Uncertainty in Artificial Intelligence (UAI)*, pages 435–442, 2003.

Rainer Opgen-Rhein and Korbinian Strimmer. From correlation to causation networks: a simple approximate learning algorithm and its application to high-dimensional plant gene expression data. *BMC Systems Biology*, 1(1):37, 2007.

Pekka Parviainen and Mikko Koivisto. Finding optimal Bayesian networks using precedence constraints. *The Journal of Machine Learning Research*, 14(1):1387–1415, 2013.

Judea Pearl. Probabilistic reasoning in intelligent systems: networks of plausible inference, 1988.

Judea Pearl. Causal diagrams for empirical research. *Biometrika*, 82(4):669–710, 1995.

Judea Pearl. *Causality: Models, Reasoning, and Inference*. Cambridge University Press, New York, 2000. 2nd edition, 2009.

Emilija Perkovic. Identifying causal effects in maximally oriented partially directed acyclic graphs. In Jonas Peters and David Sontag, editors, *Proceedings of the 36th Conference on Uncertainty in Artificial Intelligence (UAI)*, volume 124 of *Proceedings of Machine Learning Research*, pages 530–539. PMLR, 03–06 Aug 2020. URL https://proceedings.mlr.press/v124/perkovic20a.html.

Emilija Perkovic, Johannes Textor, Markus Kalisch, and Marloes H Maathuis. Complete graphical characterization and construction of adjustment sets in Markov equivalence classes of ancestral graphs. *Journal of Machine Learning Research*, 18(220):1–62, 2018.

Jonas Peters, Peter Bühlmann, and Nicolai Meinshausen. Causal inference by using invariant prediction: identification and confidence intervals. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 78 (5):947–1012, 2016.

Joseph Ramsey, Madelyn Glymour, Ruben Sanchez-Romero, and Clark Glymour. A million variables and more: the fast greedy equivalence search algorithm for learning high-dimensional graphical causal models, with an application to functional magnetic resonance images. *International journal of data science and analytics*, 3(2): 121–129, 2017.

Joseph D Ramsey, Kun Zhang, Madelyn Glymour, Ruben Sanchez Romero, Biwei Huang, Imme Ebert-Uphoff, Savini Samarasinghe, Elizabeth A Barnes, and Clark Glymour. Tetrad—a toolbox for causal discovery. In *8th international workshop on climate informatics*, pages 1–4. National Center for Atmospheric Research, 2018.

Garvesh Raskutti and Caroline Uhler. Learning directed acyclic graph models based on sparsest permutations. *Stat*, 7(1):e183, 2018.

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

Felix L Rios, Giusi Moffa, and Jack Kuipers. Benchpress: A versatile platform for structure learning in causal and probabilistic graphical models. *Journal of Statistical Software*, 114:1–43, 2025.

Juliane Schäfer and Korbinian Strimmer. A shrinkage approach to large-scale covariance matrix estimation and implications for functional genomics. *Statistical applications in genetics and molecular biology*, 4(1), 2005.

Gideon Schwarz. Estimating the dimension of a model. *The annals of statistics*, pages 461–464, 1978.

Marco Scutari. Learning Bayesian networks with the bnlearn r package. *Journal of statistical software*, 35:1–22, 2010.

Marco Scutari. Bayesian networks, MAGIC populations and multiple trait prediction. Invited talk, 5th International Conference on Quantitative Genetics (ICQG); network distributed via the bnlearn Bayesian Network Repository, https://www.bnlearn.com/bnrepository/, 2016.

Marco Scutari, Phil Howell, David J Balding, and Ian Mackay. Multiple quantitative trait analysis using bayesian networks. *Genetics*, 198(1):129–137, 2014.

Ilya Shpitser, Tyler VanderWeele, and James M Robins. On the validity of covariate adjustment for estimating causal effects. In *Proceedings of the Twenty-Sixth Conference on Uncertainty in Artificial Intelligence*, pages 527–536. AUAI Press, 2010.

Liam Solus, Yuhao Wang, and Caroline Uhler. Consistency guarantees for greedy permutation-based causal inference algorithms. *Biometrika*, 108(4):795–814, 2021. doi: 10.1093/biomet/asaa104.

Peter Spirtes, Clark N Glymour, and Richard Scheines. *Causation, prediction, and search*. MIT press, 2000.

Thomas Stützle. Applying iterated local search to the permutation flow shop problem. Technical report, Technical Report AIDA-98-04, FG Intellektik, TU Darmstadt, 1998.

Eric D Taillard. Comparison of iterative searches for the quadratic assignment problem. *Location science*, 3(2): 87–105, 1995.

Ioannis Tsamardinos, Laura E Brown, and Constantin F Aliferis. The max-min hill-climbing Bayesian network structure learning algorithm. *Machine learning*, 65:31–78, 2006.

Benito van der Zander and Maciej Liśkiewicz. Finding minimal d-separators in linear time and applications. In *Uncertainty in Artificial Intelligence*, pages 637–647. PMLR, 2020.

Benito van der Zander, Maciej Liśkiewicz, and Johannes Textor. Constructing separators and adjustment sets in ancestral graphs. In *Proceedings of the UAI 2014 Conference on Causal Inference: Learning and Prediction-Volume 1274*, pages 11–24, 2014.

Thomas Verma and Judea Pearl. Equivalence and synthesis of causal models. In *Proceedings of the Sixth Conference on Uncertainty in Artificial Intelligence*, pages 220–227, 1991.

Tian-Zuo Wang, Wen-Bo Du, and Zhi-Hua Zhou. An efficient maximal ancestral graph listing algorithm. In *Forty-first International Conference on Machine Learning*, 2024.

Tian-Zuo Wang, Wen-Bo Du, and Zhi-Hua Zhou. Polynomial-delay MAG listing with novel locally complete orientation rules. In *Forty-second International Conference on Machine Learning*, 2025. URL https://openreview.net/forum?id=70voOlSPos.

Michael L Waskom. Seaborn: statistical data visualization. *Journal of open source software*, 6(60):3021, 2021.

Marcel Wienöbst, Max Bannach, and Maciej Liśkiewicz. Extendability of causal graphical models: Algorithms and computational complexity. In *Uncertainty in Artificial Intelligence*, pages 1248–1257. PMLR, 2021a.

Marcel Wienöbst, Max Bannach, and Maciej Liskiewicz. Polynomial-time algorithms for counting and sampling Markov equivalent dags. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pages 12198–12206, 2021b.

Marcel Wienöbst, Max Bannach, and Maciej Liśkiewicz. Polynomial-time algorithms for counting and sampling Markov equivalent dags with applications. *Journal of Machine Learning Research*, 24(213):1–45, 2023.

Marcel Wienöbst, Leonard Henckel, and Sebastian Weichwald. Embracing discrete search: A reasonable approach to causal structure learning. In *The Fourteenth International Conference on Learning Representations (ICLR)*, 2026.

Changhe Yuan and Brandon Malone. Learning optimal Bayesian networks: A shortest path perspective. *Journal of Artificial Intelligence Research*, 48:23–65, 2013.

Jiji Zhang. *Causal inference and reasoning in causally insufficient systems*. PhD thesis, Citeseer, 2006.

Jiji Zhang. On the completeness of orientation rules for causal discovery in the presence of latent confounders and selection bias. *Artificial Intelligence*, 172(16):1873–1896, 2008a. ISSN 0004-3702. doi: https://doi.org/10.1016/j.artint.2008.08.001. URL https://www.sciencedirect.com/science/article/pii/S0004370208001008.

Jiji Zhang. Causal reasoning with ancestral graphs. *Journal of Machine Learning Research*, 9:1437–1474, 2008b.

Xun Zheng, Bryon Aragam, Pradeep K Ravikumar, and Eric P Xing. DAGs with NO TEARS: Continuous optimization for structure learning. *Advances in Neural Information Processing Systems*, 31, 2018.

Yujia Zheng, Biwei Huang, Wei Chen, Joseph Ramsey, Mingming Gong, Ruichu Cai, Shohei Shimizu, Peter Spirtes, and Kun Zhang. Causal-learn: Causal discovery in python. *Journal of Machine Learning Research*, 25(60):1–8, 2024.

Breaking Bad: Component-Wise Parent Deletion for Score-Based Causal Discovery (Supplementary Material)

Min Woo Park[*1] **Taehui Yun**[*1] **YoungIn Jang**[1] **Yoonseok Yeom**[1] **Jonghwan Kim**[1] **Jiyeon Kang**[2]
Songseong Kim[2] **Hyemin Jung**[2] **Sangmin Lee**[2] **Jongseong Jang**[†2] **Sanghack Lee**[†1,3]

[1]Graduate School of Data Science, Seoul National University, Republic of Korea
[2] LG AI Research, Republic of Korea
[3]SNU-LG AI Research Center, Republic of Korea

CONTENTS

A PRELIMINARY RESULTS

We introduce preliminary results from Chickering [2002a,b], which are used throughout the paper.

A.1 PRELIMINARY RESULTS IN CHICKERING [2002A]

Lemma 2 (Proposition 18 in Chickering [2002a]). *Let $\mathcal{P}$ be any PDAG that admits a consistent extension and contains a compelled edge $X \to Y$. If there is an edge, either directed or undirected, between Y and some Z such that Z and X are not adjacent, then that edge is compelled as $Y \to Z$.*

Lemma 3 (Proposition 19 in Chickering [2002a]). *Let $\mathcal{P}$ be any PDAG that admits a consistent extension such that there is a directed path from X to Y consisting of compelled edges. If there is an edge between X and Y, it is compelled as $X \to Y$.*

Lemma 4 (Lemma 22 in Chickering [2002a]). *For any directed edge $X \to Y$ in a CPDAG, X is a parent of every node reachable by Y via undirected edges.*

Lemma 5 (Lemma 25 in Chickering [2002a]). *Let $\mathcal{P}^c$ denote a CPDAG, and let $\mathcal{P}^{c\prime}$ denote the PDAG that results from deleting a single edge (directed or undirected) between X and Y from $\mathcal{P}^c$. Consider any consistent extension $\mathcal{G}$ of $\mathcal{P}^c$, and the directed graph $\mathcal{G}'$ that results from deleting the (directed) edge between X and Y in $\mathcal{G}$ such that the result is a DAG with the same adjacencies as $\mathcal{P}^{c\prime}$. Any v-structure in $\mathcal{G}'$ but not in $\mathcal{P}^{c\prime}$, or any v-structure in $\mathcal{P}^{c\prime}$ but not in $\mathcal{G}'$, must be of the form $X \to Z \leftarrow Y$.*

Lemma 6 (Lemma 30 in Chickering [2002a]). *Let $\mathbf{X} = \{X_1, \cdots, X_n\}$ be the nodes from any undirected clique of size n within some undirected component of a CPDAG $\mathcal{P}^c$, and let π denote any total ordering of the nodes in $\mathbf{X}$. There exists a consistent extension of $\mathcal{P}^c$ for which (i) the edge orientations among the nodes in $\mathbf{X}$ are consistent with π, and (ii) any edge between X_i and a neighbor Y that is not in $\mathbf{X}$ (i.e., $Y \in \mathtt{Ne}_{X_i}^{\mathcal{P}^c} \setminus \mathbf{X}$) is oriented as $X_i \to Y$.*

A.2 PRELIMINARY RESULTS IN CHICKERING [2002B]

Lemma 7 (Proposition 31 in Chickering [2002b]). *Let $\mathcal{P}$ and $\mathcal{P}'$ denote any pair of PDAGs. Let $X \to Y \leftarrow Z$ be any v-structure in $\mathcal{P}'$ that is not in $\mathcal{P}$. Then, one of the following conditions must hold: (i) $X \notin \mathtt{Pa}_Y^{\mathcal{P}}$, (ii) $Z \notin \mathtt{Pa}_Y^{\mathcal{P}}$, or (iii) X and Z are adjacent in $\mathcal{P}$.*

Lemma 8 (Lemma 34 in Chickering [2002b]). *Let $\{X, Y, Z\}$ be any three nodes that form a clique of size three in PDAG $\mathcal{P}$. If any two of the edges in the clique are reversible, then the third edge is reversible as well.*

Definition 5 (Definition 12 in Chickering [2002b]). For non-adjacent nodes X and Y in $\mathcal{P}^c$, and for any subset $\mathbf{T} \subseteq \mathtt{Ne}_Y^{\mathcal{P}^c} \setminus \mathtt{Adj}_X^{\mathcal{P}^c}$, the $\textsc{Insert}(X, Y, \mathbf{T})$ operator modifies $\mathcal{P}^c$ by (i) inserting the directed edge $X \to Y$, and (ii) for each $T \in \mathbf{T}$, directing the previously undirected edge $T - Y$ as $T \to Y$.

Definition 6 (Definition 13 in Chickering [2002b]). For adjacent nodes X and Y in $\mathcal{P}^c$ connected by either $X - Y$ or $X \to Y$, and for any subset $\mathbf{H} \subseteq \mathtt{Ne}_Y^{\mathcal{P}^c} \cap \mathtt{Adj}_X^{\mathcal{P}^c}$, the $\textsc{Delete}(X, Y, \mathbf{H})$ operator modifies $\mathcal{P}^c$ by deleting the edge between X and Y, and for each $H \in \mathbf{H}$, (i) directing the previously undirected edge between Y and H as $Y \to H$ and (ii) directing any previously undirected edge between X and H as $X \to H$.

Theorem 3 (Theorem 15 in Chickering [2002b]). *Let $\mathcal{P}^c$ be any CPDAG, and let $\mathcal{P}^{c\prime}$ denote the result of applying an $\textsc{Insert}(X, Y, \mathbf{T})$ operator to $\mathcal{P}^c$. There exists a consistent extension $\mathcal{G}$ of $\mathcal{P}^c$ to which adding the edge $X \to Y$ results in a consistent extension $\mathcal{G}'$ of $\mathcal{P}^{c\prime}$ if and only if the following conditions hold in $\mathcal{P}^c$:*

(i) $(\mathtt{Ne}_Y^{\mathcal{P}^c} \cap \mathtt{Adj}_X^{\mathcal{P}^c}) \cup \mathbf{T}$ *is a clique.*

(ii) Every semi-directed path from Y to X contains a node in $(\mathtt{Ne}_Y^{\mathcal{P}^c} \cap \mathtt{Adj}_X^{\mathcal{P}^c}) \cup \mathbf{T}$, where a semi-directed path is defined as a directed path in which some edges are allowed to be undirected.

Theorem 4 (Theorem 17 in Chickering [2002b]). *Let $\mathcal{P}^c$ be any CPDAG that contains either $X \to Y$ or $X - Y$, and let $\mathcal{P}^{c\prime}$ denote the result of applying a $\textsc{Delete}(X, Y, \mathbf{H})$ operator to $\mathcal{P}^c$. There exists a consistent extension $\mathcal{G}$ of $\mathcal{P}^c$ to which deleting the edge $X \to Y$ results in a consistent extension $\mathcal{G}'$ of $\mathcal{P}^{c\prime}$ if and only if $(\mathtt{Ne}_Y^{\mathcal{P}^c} \cap \mathtt{Adj}_X^{\mathcal{P}^c}) \setminus \mathbf{H}$ is a clique in $\mathcal{P}^c$.*

B ADDITIONAL BACKGROUND

We provide a comprehensive overview of prior work, focusing on their key concepts and methodological foundations.

OPS [Chickering, 2002b]. In contrast to the two-phase GES approach, i.e., a forward phase with insertion only followed by a backward phase with deletion only, OPS considers insertion and deletion operations simultaneously. That is, among all valid insertion and deletion operations, OPS selects a greedy operation that maximally improves the score.

XGES-0 and XGES [Nazaret and Blei, 2024]. XGES-0 modifies the search strategy of GES by considering insertion, deletion, and reversal operations[1] simultaneously, while explicitly prioritizing deletions over other operations and reversals over insertions whenever such operations yield a non-decreasing score. While this deletion-first heuristic partially mitigates the over-insertion effect, Fig. 1 demonstrates that substantial over-insertion remains, motivating more principled perturbation strategies.

Building on XGES-0, XGES further escapes local optima by forcefully deleting candidate edges that may block progress. Upon convergence to a local optimum, XGES removes a candidate edge, reruns XGES-0 while prohibiting its reinsertion, and accepts the perturbation only if it yields a higher final score.

LGES and LGES+ [Ejaz and Bareinboim, 2025]. LGES revisits the greediness of XGES-0 from a complementary perspective, focusing on more cautious edge insertion. Instead of selecting the highest-scoring insertion, LGES avoids inserting edges between node pairs for which the score provides evidence of conditional independence. LGES has two variants, LGES (CONS) and LGES (SAFE), which differ in their insertion strategies[2].

[1] This operator reverses the direction of a target directed edge [Hauser and Bühlmann, 2012]. One can regard this operation as deleting the edge and then inserting the reversed edge.

[2] When we refer to LGES without specifying a variant, the statement applies regardless of the insertion strategy used.

- **LGES (CONS).** The first strategy, CONSERVATIVEINSERT, iterates over all valid $\text{INSERT}(X, Y, \mathbf{T})$ operations for a given non-adjacent pair $X, Y \in \mathbf{V}$. If any choice of $\mathbf{T}$ results in a decrease in the score, the procedure discards *all* $\text{INSERT}(X, Y, *)$ operators for that pair and proceeds to the next pair.
- **LGES (SAFE).** The second strategy, SAFEINSERT, first samples an arbitrary DAG $\mathcal{G} \in [\mathcal{P}^c]$. For each pair of non-adjacent nodes $X, Y \in \mathbf{V}$ in $\mathcal{G}$, it checks whether $\mathcal{G}$ has a higher score than the graph $\mathcal{G}' = \mathcal{G} \cup \{X \to Y\}$. If adding the edge does not improve the score, *all* $\text{INSERT}(X, Y, *)$ operators for that pair are discarded.

These *less greedy* insertion rules reduce the introduction of spurious adjacencies that may be difficult to remove in later phases, leading to improved finite-sample performance.

LGES+ is a variant that combines LGES with XGES-style ILS heuristics.

BOSS [Andrews et al., 2023]. BOSS (Best Order Score Search) is a *state-of-the-art* order-based causal discovery algorithm [Raskutti and Uhler, 2018, Solus et al., 2021] that searches over variable orderings rather than graph structures directly. Given a permutation, BOSS constructs a DAG by greedily selecting parent sets using the grow-shrink (GS) procedure [Margaritis and Thrun, 1999] and then scores the resulting graph. The search proceeds by iteratively relocating individual variables within the permutation to maximize the score, followed by an optional backward equivalence search to ensure asymptotic correctness. In this paper, we employ BOSS with the additional GES backward phase, initialized from the CPDAG corresponding to the DAG returned by BOSS.

Peter–Clark (PC) [Spirtes et al., 2000]. The Peter–Clark (PC) algorithm is a classical constraint-based method for causal discovery. It infers a CPDAG by performing a sequence of conditional independence tests to identify the skeleton of the graph, followed by a set of orientation rules—known as Meek's rules [Meek, 1995]—to direct edges while avoiding cycles and spurious v-structures. Unlike score-based approaches that optimize a global objective, PC relies on statistical tests and is therefore sensitive to the choice of significance level and the reliability of conditional independence testing.

C DISCUSSION ON METHODOLOGICAL CHOICES

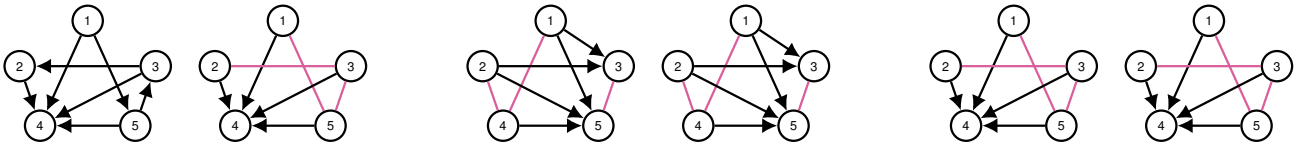
In this section, we provide a deeper perspective on the core design choices of our proposed framework, expanding upon the rationale presented in the main text. By contextualizing these decisions within the broader landscape of ILS and score-based causal discovery, we aim to highlight the principled reasoning that underpins our methodology.

Theoretical scope and search dynamics. Our theoretical results (Prop. 1 and Thm. 2) focus on guaranteeing the validity of the proposed DELETEPA operator. While mapping out the complete search dynamics—such as providing formal bounds for escaping specific local optima—remains a challenging open problem in non-convex combinatorial optimization, our validity proofs establish a critical foundation.

They ensure that every perturbation keeps the search within valid graph states. This theoretical guarantee enables the ILS framework to explore the search space aggressively. The substantial and consistent performance improvements observed across a wide variety of synthetic and realistic graphs suggest that these validity-preserving operators can effectively disrupt the dense local optima that typically trap conventional score-based methods.

Principled coarse-to-fine scheduling. The DP+ variant employs a sequential scheduling strategy, moving from component-wise deletions to single-node deletions and finally to single-edge deletions. This design follows the well-established coarse-to-fine search paradigm in combinatorial optimization [Lourenço et al., 2003]. It is also consistent with prior observations in ILS that there is no *a priori* single best perturbation strategy across problem instances [Taillard, 1995, Hong et al., 1997, Stützle, 1998].

By initiating the search with coarse, macro-level disruptions (component-wise), the algorithm effectively escapes deep, suboptimal basins of attraction. Subsequent finer-grained operations (single-node and single-edge) then facilitate delicate refinements within the newly discovered promising regions. This hierarchical sequence systematically balances broad exploration of the search space with targeted exploitation, providing a highly robust default configuration that generalizes across diverse domains without the need for exhaustive hyperparameter tuning.



(a) $\mathcal{G}^*$ (b) $\mathcal{P}^c$ (c) Stuck CPDAGs (d) DP variants

Figure 10: Reversible edges are highlighted in pink. (a, b) The true DAG and its corresponding CPDAG. (c, d) For both the GES and XGES variants, the single-edge deletion methods (i.e., GES-D and XGES) get stuck at the same incorrect CPDAG and accept no further updates, whereas the DP variants successfully escape the local optimum and recover the ground-truth CPDAG.

The strategic role of the $\mathbf{H} = \emptyset$ constraint. In the single-node and single-edge deletion phases of DP+, we intentionally enforce $\mathbf{H} = \emptyset$. Although this setting restricts the creation of new v-structures during the perturbation step, it acts as an important safeguard. As demonstrated in Cor. 2, permitting arbitrary $\mathbf{H}$ sets during aggressive deletion operations frequently triggers unintended *collateral parental growth*, which directly undermines the primary goal of simplifying overly dense graph structures. It is crucial to note that this restriction is strictly temporary.

The subsequent local search phase (i.e., SEARCH_2) operates without this constraint, considering an exponential number of non-empty $\mathbf{H}$ sets to identify essential v-structures and traverse MECs. Through this design, our framework separates responsibilities: perturbations safely dismantle dense local artifacts, while the subsequent greedy search rigorously rebuilds and refines the structural details based on evidence in the data.

A case study where single-edge deletions fail. We illustrate a case in which single-edge perturbations (GES-D and XGES) cannot escape a local optimum, whereas component-wise parent deletions (GES-DP and XGES-DP) can.

- **Single-edge deletions (Fig. 10c).** GES-D and XGES yield the same *incorrect* CPDAG, with a BIC score of 364.33 and SHD$= 8$.
- **Component-wise parent deletions (Fig. 10d).** GES-DP and XGES-DP recover the ground truth, with a BIC score of 377.52 and zero SHD.

The score gap of $+13.19$ BIC units is identical for the GES and XGES bases at $n = 2{,}000$ and $\lambda = 2$.

The escape mechanism is as follows. In GES and XGES-0 (SEARCH_1), the CPDAG over-parents the receiving component $\{3, 5\}$ with directed edges from $\{1, 2, 4\}$. Component-wise parent deletion removes all these incoming edges in one move; a subsequent phase then reorients the graph so that node 4 becomes the common child of $\{1, 2, 3, 5\}$, forming four collider structures, as in the ground truth. In contrast, no sequence of single-edge edits can achieve this transition, since every individual edit strictly worsens the BIC score.

D OMITTED PROOFS

In this section, we provide detailed proofs of the statements presented in the main body of the paper.

D.1 PROOF OF COROLLARY 1

Proof of Cor. 1. By the validity condition of $\text{DELETEPA}(X, \mathbf{H})$ in Prop. 1, $\texttt{Ne}_X^{\mathcal{P}^c} \setminus \mathbf{H}$ is a clique; hence $\texttt{Ne}_X^{\mathcal{P}^c} \setminus \mathbf{H} \cup \{X\}$ also forms a clique. According to Lem. 6, one can construct a DAG $\mathcal{G}$ whose directed edges are consistent with the ordering $\pi = \{\texttt{Ne}_X^{\mathcal{P}^c} \setminus \mathbf{H} \prec X \prec \mathbf{H}\}$. This means $\texttt{Pa}_X^{\mathcal{G}} = \texttt{Pa}_X^{\mathcal{P}^c} \cup (\texttt{Ne}_X^{\mathcal{P}^c} \setminus \mathbf{H})$. After applying the operator, the resulting DAG $\mathcal{G}'$ has $\texttt{Pa}_X^{\mathcal{G}'} = \texttt{Ne}_X^{\mathcal{P}^c} \setminus \mathbf{H}$.

By score equivalence (Def. 1), $\mathcal{G}$ and $\mathcal{G}'$ represent the score of corresponding CPDAGs, respectively. Therefore, the score before applying the operation is $s(X, \texttt{Pa}_X^{\mathcal{P}^c} \cup (\texttt{Ne}_X^{\mathcal{P}^c} \setminus \mathbf{H}))$, and the score after applying the operation is $s(X, \texttt{Ne}_X^{\mathcal{P}^c} \setminus \mathbf{H})$. This concludes the proof. □

D.2 PROOF OF PROPOSITION 1

Lemma 9. *Let $\mathcal{P}^c$ denote a CPDAG, and let $\mathcal{P}^{c\prime}$ denote the PDAG obtained by applying* DELETEPA$(X, \mathbf{H})$ *to $\mathcal{P}^c$. Consider any consistent extension $\mathcal{G}$ of $\mathcal{P}^c$, and the directed graph $\mathcal{G}'$ that results from deleting the edges between $\texttt{Pa}_X^{\mathcal{P}^c}$ and X from $\mathcal{G}$. Then $\mathcal{G}'$ is a consistent extension of $\mathcal{P}^{c\prime}$ if and only if the set of reversible children of X in $\mathcal{G}$ is equal to $\mathbf{H}$.*

Proof of Lem. 9. From Lem. 7 and the fact that DELETEPA$(X, \mathbf{H})$ does not undirect any directed edges, it follows that v-structures present in $\mathcal{P}^c$ but absent in $\mathcal{P}^{c\prime}$ are precisely the v-structures present in $\mathcal{G}$ but absent in $\mathcal{G}'$. Furthermore, the v-structures eliminated by applying DELETEPA$(X, \mathbf{H})$ are exactly those that involve any edge between $P \in \texttt{Pa}_X^{\mathcal{P}^c}$ and X. Therefore, it suffices to show that the additional v-structures introduced by the two modifications coincide *if and only if* the set of reversible children of X in $\mathcal{G}$ is equal to $\mathbf{H}$.

For the sake of contradiction, suppose that an additional v-structure introduced by DELETEPA$(X, \mathbf{H})$ in $\mathcal{P}^{c\prime}$ is of the form $A \to D \leftarrow B$, where $D \in \mathbf{H}$ and $\{A, B\} \cap (\texttt{Pa}_X^{\mathcal{P}^c} \cup \{X\}) \neq \emptyset$. Without loss of generality, assume $B = X$ and $A \notin \{X\} \cup \texttt{Pa}_X^{\mathcal{P}^c}$. By the definition of DELETEPA$(X, \mathbf{H})$, the edge $A \to D$ must already be present in $\mathcal{P}^c$. Since the v-structure $A \to D \leftarrow X$ does not exist in $\mathcal{P}^c$, and the adjacency between A and X is not created by DELETEPA$(X, \mathbf{H})$, it follows that the edge between D and X must be undirected in $\mathcal{P}^c$. However, by Lem. 2, the edge cannot remain undirected in $\mathcal{P}^{c\prime}$, yielding a contradiction.

Therefore, it suffices to consider the case in which the additional v-structures are of the form $P \to D \leftarrow X$ where $P \in \texttt{Pa}_X^{\mathcal{P}^c}$, and either both edges already exist in $\mathcal{P}^c$ or $D \in \mathbf{H}$. For the sake of contradiction, suppose that the former case holds, i.e., $D \leftarrow X$ is a compelled edge. Since $D \leftarrow X$ and $P \to X$ are compelled in $\mathcal{P}^c$, it follows from Lem. 3 that $P \to D$ is also compelled in $\mathcal{P}^c$. Consequently, the v-structure $P \to D \leftarrow X$ must already exist in $\mathcal{P}^{c\prime}$, contradicting the assumption that it is introduced by DELETEPA$(X, \mathbf{H})$. Hence, it must be the case that $D \in \mathbf{H}$. Since $\mathbf{H} \subseteq \texttt{Ne}_X^{\mathcal{P}^c}$, we conclude that $D \in \mathbf{H}$ is a reversible child of X in $\mathcal{G}$. □

Proof of Prop. 1. **(If)** Let $\mathcal{P}^{c\prime}$ denote the PDAG obtained by applying DELETEPA$(X, \mathbf{H})$ to $\mathcal{P}^c$. Assume that $\texttt{Ne}_X^{\mathcal{P}^c} \setminus \mathbf{H}$ is a clique of undirected edges. By definition of $\texttt{Ne}_X^{\mathcal{P}^c}$, it follows that $\texttt{Ne}_X^{\mathcal{P}^c} \setminus \mathbf{H} \cup \{X\}$ also forms a clique. According to Lem. 6, we can construct a consistent extension $\mathcal{G}$ from $\mathcal{P}^c$ in which the directed edges are consistent with the ordering $\pi = \{\texttt{Ne}_X^{\mathcal{P}^c} \setminus \mathbf{H} \prec X \prec \mathbf{H}\}$. Clearly, in $\mathcal{G}$, the reversible children of X are precisely the nodes in $\mathbf{H}$. Furthermore, any node in $\texttt{Ne}_X^{\mathcal{P}^c}$ becomes a child of $\texttt{Pa}_X^{\mathcal{P}^c}$ by Lem. 1. Thus, applying Lem. 9, the resulting graph $\mathcal{G}'$ obtained by deleting the directed edges from $\mathcal{G}$ is a consistent extension of $\mathcal{P}^{c\prime}$.

(Only if) Suppose that $\texttt{Ne}_X^{\mathcal{P}^c} \setminus \mathbf{H}$ does not form a clique in $\mathcal{P}^c$ but $\mathcal{P}^{c\prime}$ admits a consistent extension $\mathcal{G}'$. From Lem. 9, we conclude that the set of reversible children of X is precisely $\mathbf{H}$. Hence, every node in $\texttt{Ne}_X^{\mathcal{P}^c} \setminus \mathbf{H}$ must be a parent of X in $\mathcal{G}$. Since $\texttt{Ne}_X^{\mathcal{P}^c} \setminus \mathbf{H}$ is not a clique, there exist two non-adjacent nodes in this set that, together with X, form a v-structure in $\mathcal{G}$. This contradicts the fact that both corresponding edges are reversible. □

D.3 PROOF OF THEOREM 2

Proof of Thm. 2. Let $\mathcal{P}^c$ denote a CPDAG, and let $\mathcal{P}^{c\prime}$ denote the resulting graph obtained by applying DELETEPA$(\mathbf{U}^{\mathcal{P}^c})$ to $\mathcal{P}^c$. Consider any consistent extension $\mathcal{G}$ of $\mathcal{P}^c$, and the graph $\mathcal{G}'$ that results from deleting the edges between $\texttt{Pa}_X^{\mathcal{P}^c}$ and $\mathbf{U}^{\mathcal{P}^c}$ for all $X \in \mathbf{U}^{\mathcal{P}^c}$ in $\mathcal{G}$. Clearly, $\mathcal{G}'$ and $\mathcal{P}^{c\prime}$ contain the same adjacencies. Therefore, it suffices to show that $\mathcal{G}'$ and $\mathcal{P}^{c\prime}$ have the same set of v-structures (Thm. 1). From Lem. 5, any v-structure that is in one but not the other must have the form $P \to D \leftarrow Z$ for any nodes P and Z where $P \in \texttt{Pa}_X^{\mathcal{P}^c}$ for arbitrary $X \in \mathbf{U}^{\mathcal{P}^c}$ and $Z \in \mathbf{U}^{\mathcal{P}^c} \setminus \{X\}$.

Suppose the v-structure $P \to D \leftarrow Z$ exists in $\mathcal{G}'$ but not in $\mathcal{P}^{c\prime}$. If the edge $D \leftarrow Z$ is compelled in $\mathcal{P}^c$, then, since $D \leftarrow Z$ and $P \to Z$ are compelled in $\mathcal{P}^c$, it follows from Lem. 3 that $P \to D$ is also compelled in $\mathcal{P}^c$. Consequently, the v-structure $P \to D \leftarrow Z$ must exist in $\mathcal{P}^{c\prime}$, yielding a contradiction. Otherwise, if $D \leftarrow Z$ is a reversible edge, then $D \in \mathbf{U}^{\mathcal{P}^c}$. Note that the existence of $P \to D$ in $\mathcal{P}^c$ is guaranteed by Lem. 4. Since we remove all edges between $\texttt{Pa}_X^{\mathcal{P}^c}$ and $\mathbf{U}^{\mathcal{P}^c}$, the existence of the directed edge $P \to D$ results in a contradiction.

Conversely, suppose that the v-structure $P \to D \leftarrow Z$ exists in $\mathcal{P}^{c\prime}$. Both edges $P \to D$ and $D \leftarrow Z$ are directed in $\mathcal{P}^c$, since the only difference between $\mathcal{P}^c$ and $\mathcal{P}^{c\prime}$ is the presence of edges between $\texttt{Pa}_X^{\mathcal{P}^c}$ and $\mathbf{U}^{\mathcal{P}^c}$. Consequently, these edges have the same orientation in both $\mathcal{G}$ and $\mathcal{G}'$, and thus the v-structure must also exist in $\mathcal{G}'$. □

D.4 PROOF OF COROLLARY 3

Proof of Cor. 3. It follows immediately from subtracting the score of $\mathcal{G} \in [\mathcal{P}^c]$ from the score of $\mathcal{G}'$, where $\mathcal{G}$ is any consistent extension of $\mathcal{P}^c$, and $\mathcal{G}'$ denotes the DAG that results from the removing the edges from the parents of $\mathbf{U}$ in $\mathcal{P}^c$. □

E EXPERIMENTAL DETAILS

The simulations were conducted on a Linux server equipped with an Intel® Xeon® Gold 6430 processor (32 cores per socket, 128 logical cores total) running at 2.1 GHz and 64 GB of RAM. No GPUs were used during the simulations.

Baseline implementation. We implement the proposed methods (e.g., DELETEPA, DP variants) in the programming language adopted in previous work whenever possible. In particular, GES, OPS and XGES are evaluated using the C++ implementation provided by Nazaret and Blei [2024][3]. For BOSS, we employ Tetrad [Ramsey et al., 2018][4], and apply the backward phase using our C++ implementation. LGES is originally implemented in Python[5], which can be computationally inefficient when evaluating a wide range of experimental settings due to language overhead. To enable efficient large-scale evaluation, we reimplement LGES in C++ along with its proposed variants. Finally, we use the Python `causal-learn` [Zheng et al., 2024][6] implementation of PC.

E.1 DETAILS ON DATA GENERATING PROCESSES

Erdős–Rényi DAGs. We generate Erdős–Rényi (ER) graphs by independently sampling undirected edges between each pair of nodes with a fixed probability $\frac{\rho}{d-1}$. This process yields random graphs with homogeneous degree distributions, serving as a standard baseline for synthetic causal graph generation. To obtain a DAG $\mathcal{G}^*$, we first assign a random ordering π and then orient each undirected edge according to this ordering. Specifically, for any undirected edge V_i — V_j, we direct the edge following the specified ordering, i.e., $V_{\min(\pi(i),\pi(j))} \to V_{\max(\pi(i),\pi(j))}$.

Scale-free DAGs. We generate scale-free DAGs using the Barabási–Albert (BA) model [Barabási and Albert, 1999], which is a standard generative model for networks exhibiting heterogeneous degree distributions. The BA model constructs an undirected graph through a preferential attachment mechanism, where nodes are added sequentially and each newly added node connects to m existing nodes. As a result, nodes that are added earlier tend to accumulate more connections, forming high-degree structures. The attachment parameter m controls the graph density, with larger values leading to denser graphs. Following the experimental setting of Wienöbst et al. [2026], we set $m = 4$ in the main experiments. To evaluate different density regimes, we additionally consider $m \in \{2, 6\}$. To obtain a DAG $\mathcal{G}^*$, we first assign a random ordering and orient each undirected edge according to this ordering as in ER.

Simulated data generation. Given a generated causal graph structure (i.e., ER or scale-free), we simulate data from a linear-Gaussian structural equation model. To ensure faithfulness and avoid weak dependencies, the edge coefficients in $\mathbf{w}_V$ are sampled independently from a uniform distribution $\texttt{Unif}([-2, -0.5] \cup [0.5, 2])$, following the same setting as in Ng et al. [2020] and Ejaz and Bareinboim [2025].

In some cases, the resulting weight matrix $[\mathbf{w}_V]_{V \in \mathbf{V}}$ becomes (nearly) singular, which may lead to numerical instability during data generation. In such cases, we apply ℓ_1 normalization to each row of the weight matrix with a scaling factor, i.e., $\tilde{\mathbf{w}}_V \triangleq 0.9 \cdot \mathbf{w}_V / \|\mathbf{w}_V\|_1$. The noise terms ε_V are sampled independently from $\mathcal{N}(\mu_V, \sigma_V^2)$, with means μ_V drawn from $\mathcal{N}(0, 1)$ and variances σ_V^2 drawn from $\texttt{Unif}([0.1, 0.5])$. We define $v = \tilde{\mathbf{w}}_V^\top \mathbf{pa}_V^{\mathcal{G}^*} + \varepsilon_V$[7] and draw n samples from p^* using `sempler`[8] [Gamella et al., 2022].

[3] https://github.com/ANazaret/XGES
[4] https://github.com/cmu-phil/tetrad
[5] https://github.com/CausalAILab/lges
[6] https://github.com/py-why/causal-learn
[7] Here $\mathbf{pa}_V^{\mathcal{G}}$ denotes the value assignments of $\mathbf{Pa}_V^{\mathcal{G}}$.
[8] https://github.com/juangamella/sempler

Table 3: Summary of biologically motivated benchmark networks. Here, d denotes the number of nodes, $|\mathbf{E}|$ the number of edges, and $\rho \triangleq 2|\mathbf{E}|/d$ the average degree.

Dataset	d	$\lvert\mathbf{E}\rvert$	ρ
ARTH150	107	150	2.80
ECOLI70	46	70	3.04
MAGIC-NIAB	44	66	3.00
MAGIC-IRRI	64	102	3.19

Biologically motivated benchmark datasets. We leverage the biological networks from the `GeneNet`[9] package, distributed through the `bnlearn`[10] repository [Scutari, 2010].

- **ARTH150 [Opgen-Rhein and Strimmer, 2007].** ARTH150 was originally constructed from high-dimensional plant gene expression data and serves as a benchmark for discovery algorithms.
- **ECOLI70 [Schäfer and Strimmer, 2005].** ECOLI70 is derived from microarray gene expression data of *Escherichia coli*, adapted from the `GeneNet` package.
- **MAGIC-NIAB [Mackay et al., 2014, Scutari et al., 2014].** MAGIC-IRRI and MAGIC-NIAB are derived from Multi-parent Advanced Generation Inter-Cross (MAGIC) populations. MAGIC-NIAB encodes the genetic and phenotypic structure of an eight-parent winter-wheat MAGIC population developed at the National Institute of Agricultural Botany (NIAB).
- **MAGIC-IRRI [Bandillo et al., 2013, Scutari, 2016].** MAGIC-IRRI is built from a rice MAGIC population from the International Rice Research Institute (IRRI) as an example of multiple-trait modeling in plant genetics.

The sizes of the biological networks are summarized in Table 3.

Biological data generation. We retain the structural parameter settings from the `bnlearn` repository. For each variable $V \in \mathbf{V}$, the network fixes an intercept β_V, edge coefficients $\mathbf{w}_V$ for its parents $\mathtt{Pa}_V^{\mathcal{G}}$, and a residual variance σ_V^2, all of which are held fixed across replicates. Then, n samples are generated according to $v = \mathbf{w}_V^\top \mathbf{pa}_V^{\mathcal{G}} + \beta_V + \varepsilon_V$ with $\varepsilon_V \sim \mathcal{N}(0, \sigma_V^2)$. We repeat this procedure with 30 random seeds; only the realized noise varies.

E.2 DETAILS ON METRICS

Structural Hamming Distance. We compute Structural Hamming Distance (SHD) at the CPDAG level, i.e., between the predicted CPDAG $\hat{\mathcal{P}} = \langle \mathbf{V}, \hat{\mathbf{E}} \rangle$ and the CPDAG $\mathcal{P}^* = \langle \mathbf{V}, \mathbf{E}^* \rangle$ corresponding to the true DAG $\mathcal{G}^*$ [Peters et al., 2016]. Accordingly, the ground-truth DAG $\mathcal{G}^*$ is first converted to its corresponding CPDAG using DAG-TO-CPDAG [Chickering, 2002a]. The SHD measures the minimum number of elementary edge operations required to transform $\hat{\mathcal{P}}$ into $\mathcal{P}^*$.

For each unordered node pair $\langle V_1, V_2 \rangle$, if an adjacency exists in exactly one of $\hat{\mathcal{P}}$ and $\mathcal{P}^*$, a cost of 1 is incurred (missing or extra edge). If an adjacency exists in both graphs but differs in orientation or type (e.g., $V_1 \to V_2$ vs. $V_1 \leftarrow V_2$; or $V_1 \to V_2$ vs. $V_1 — V_2$), an additional cost of 1 is incurred. If both the adjacency and its orientation and type match, no cost is added. The SHD is obtained by summing these costs over all node pairs.

Precision, recall and F1 score. We adopt the mixed-graph true/false positive scoring rule proposed by Rios et al. [2025]. For each predicted edge $E \in \hat{\mathbf{E}}$, we assign edge-wise true-positive and false-positive scores as follows:

$$\mathtt{TP}_E = \begin{cases} 1 & \text{if } E \text{ appears in } \mathbf{E}^* \text{ with the same orientation or type} \\ \frac{1}{2} & \text{if } E \text{ appears in } \mathbf{E}^* \text{ but with a different orientation or type} \\ 0 & \text{if no orientation or type of } E \text{ appears in } \mathbf{E}^* \end{cases} \tag{5}$$

[9] https://github.com/cran/GeneNet
[10] https://www.bnlearn.com/bnrepository

$$\mathtt{FP}_E = \begin{cases} 1 & \text{if no orientation or type of } E \text{ appears in } \mathbf{E}^* \\ \frac{1}{2} & \text{if } E \text{ appears in } \mathbf{E}^* \text{ but with a different orientation or type} \\ 0 & \text{if } E \text{ appears in } \mathbf{E}^* \text{ exactly.} \end{cases} \tag{6}$$

The aggregated counts of true positives (TP), false positives (FP) and false negatives (FN) are then defined as:

$$\mathtt{TP} = \sum_{E \in \hat{\mathbf{E}}} \mathtt{TP}_E \qquad \mathtt{FP} = \sum_{E \in \hat{\mathbf{E}}} \mathtt{FP}_E \qquad \mathtt{FN} = |\mathbf{E}^*| - \mathtt{TP}. \tag{7}$$

Put simply, this scoring rule assigns full credit to an exactly correct edge, half credit when the adjacency is correct but the orientation or edge type is incorrect, and zero credit to incorrect adjacencies. Finally, precision (PR), recall (RE), and the F1 score ($\mathtt{F}_1$) are defined as follows:

$$\mathtt{PR} = \frac{\mathtt{TP}}{\mathtt{TP} + \mathtt{FP}} \qquad \mathtt{RE} = \frac{\mathtt{TP}}{\mathtt{TP} + \mathtt{FN}} \qquad \mathtt{F}_1 = \frac{2\,\mathtt{PR} \cdot \mathtt{RE}}{\mathtt{PR} + \mathtt{RE}} = \frac{2\,\mathtt{TP}}{2\,\mathtt{TP} + \mathtt{FP} + \mathtt{FN}}. \tag{8}$$

Adjustment Identification Distance. We additionally report the Adjustment Identification Distance (AID) [Henckel et al., 2024] in Appendix E.3.8. AID reflects the dissimilarity between the learned graph and the true graph when used to estimate causal effects with an adjustment set [Pearl, 1995, Shpitser et al., 2010]. We compute the AID between the learned CPDAG $\hat{\mathcal{P}}$ and the true CPDAG $\mathcal{P}^*$ [Perkovic et al., 2018, Perkovic, 2020].

While AID can be computed using various adjustment strategies (e.g., parents or constructive back-door sets [van der Zander et al., 2014, van der Zander and Liśkiewicz, 2020]), we specifically report the *Ancestor-AID*, which counts *mistakes* using the full set of treatment ancestors as the adjustment set if V_2 is a descendant of V_1; otherwise it infers the observational density.

Ancestor-AID is measured across all $d(d-1)$ possible ordered node pairs $\langle V_1, V_2 \rangle$ (i.e., each pair corresponds to $P(v_2 \mid do(v_1))$) in the graph. To measure the number of mistakes, we first classify each pair into one of three disjoint categories based on the claim made by $\hat{\mathcal{P}}$: (1) non-identifiable, (2) non-descendant, or (3) identifiable with a proposed adjustment set $\mathtt{An}(V_1)_{\hat{\mathcal{P}}} \setminus \{V_1\}$. The subsequent evaluation against $\mathcal{P}^*$ inherently yields at most one of the following mutually exclusive mistakes:

(i) ***Identifiability mismatch:*** *$\hat{\mathcal{P}}$ incorrectly claims that the causal effect $P(v_2 \mid do(v_1))$ is not identifiable, whereas it is identifiable in $\mathcal{P}^*$.*

(ii) ***Descendant misclassification:*** *$\hat{\mathcal{P}}$ incorrectly identifies V_2 as a non-descendant of V_1 (predicting no causal effect), whereas $V_2 \in \mathtt{De}(V_1)_{\mathcal{P}^*}$.*

(iii) ***Invalid adjustment set:*** *$\hat{\mathcal{P}}$ proposes the full ancestors $\mathtt{An}(V_1)_{\hat{\mathcal{P}}} \setminus \{V_1\}$ as a valid adjustment set, but this set fails to satisfy the adjustment criterion in $\mathcal{P}^*$.*

To compute the Ancestor-AID, we utilize the Python package `gadjid`[11], which wraps the highly efficient core algorithm implemented in Rust.

Runtime analyses. We report the wall-clock time (in seconds) for all algorithms, measuring only the causal discovery process while excluding data loading and disk I/O operations. For BOSS, the total runtime is computed as the sum of the initial forward graph construction time (Tetrad in Java) and the subsequent search time (in C++). For PC (`causal-learn` in Python), we measure the execution time required to produce the final PDAG, including all independence tests.

[11] https://github.com/CausalDisco/gadjid

E.3 OMITTED EXPERIMENTAL RESULTS

E.3.1 Performance under Varying Density Factor

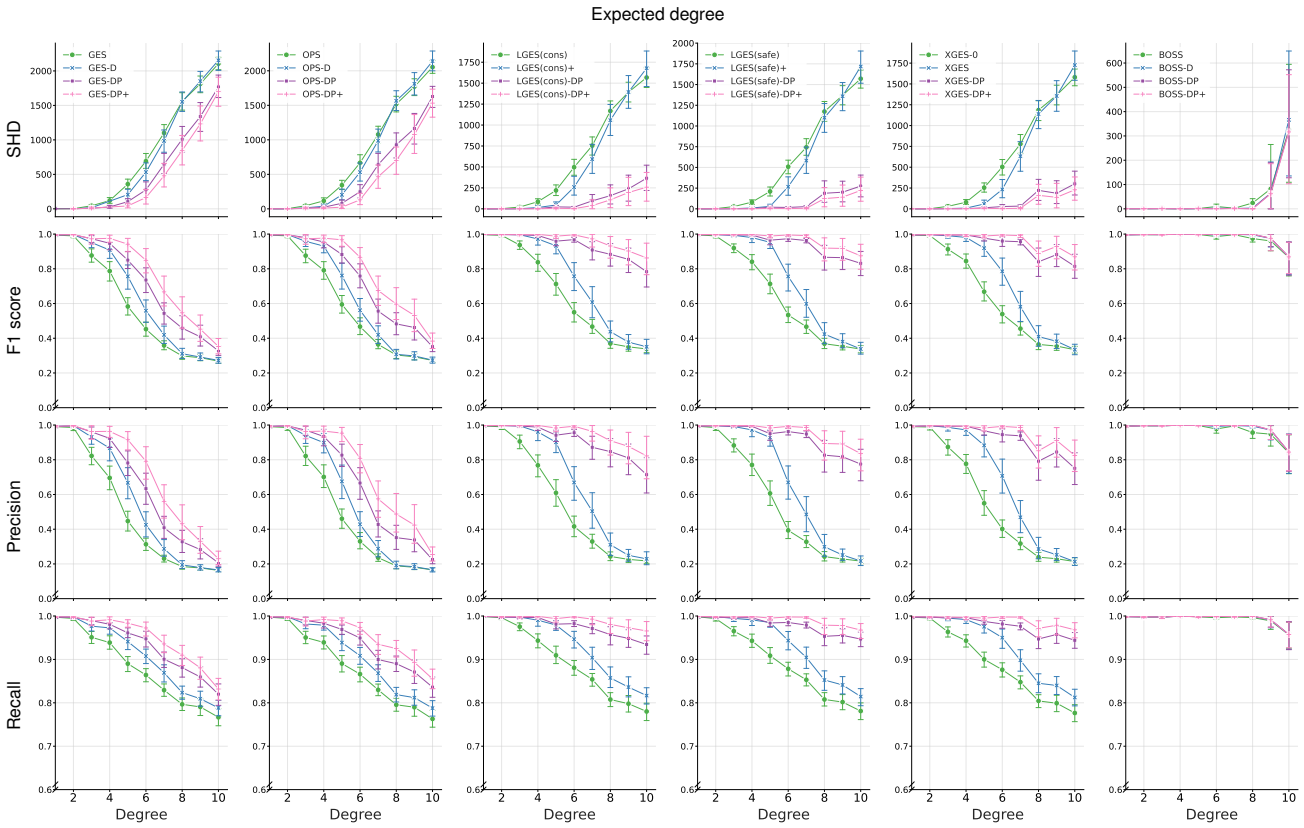


Figure 11: SHD(↓), F1 score(↑), Precision(↑), and Recall(↑) for GES, OPS, LGES (CONS), LGES (SAFE), XGES, and BOSS variants (from left to right) as **the expected degree ρ varies**. Colors indicate the four compared variant groups: **non-ILS** variants, **single-edge deletion** variants, **DP** variants, and **DP+** variants. The vertical lines are computed as bootstrapped 95% confidence intervals over random seeds. The y-axes for Recall are truncated below 0.6 for visibility.

Fig. 11 presents the full comparison across all baselines as the expected degree ρ varies over $\rho \in [1, 10]$, while fixing $d = 100$, $n = 10^4$, and $\lambda = 2$.

- In relatively dense graphs, both precision and recall decrease; however, the magnitude of degradation differs substantially. Precision drops sharply—falling to around 0.2 in some settings—whereas recall remains comparatively higher, typically above 0.8. As a result, the overall decline in F1 score is primarily driven by the pronounced reduction in precision, indicating amplified over-insertion errors in dense graphs.

  Despite differences in absolute performance across baselines, the relative behavior of the parent deletion perturbation strategies remains stable, i.e., DP and DP+ variants consistently maintain higher precision and F1 scores than other strategies.

- In contrast, under relatively sparse graphs, all variants achieve near-perfect precision and high F1 scores, resulting in only marginal differences in F1 scores.

E.3.2 Performance under Varying Numbers of Variables

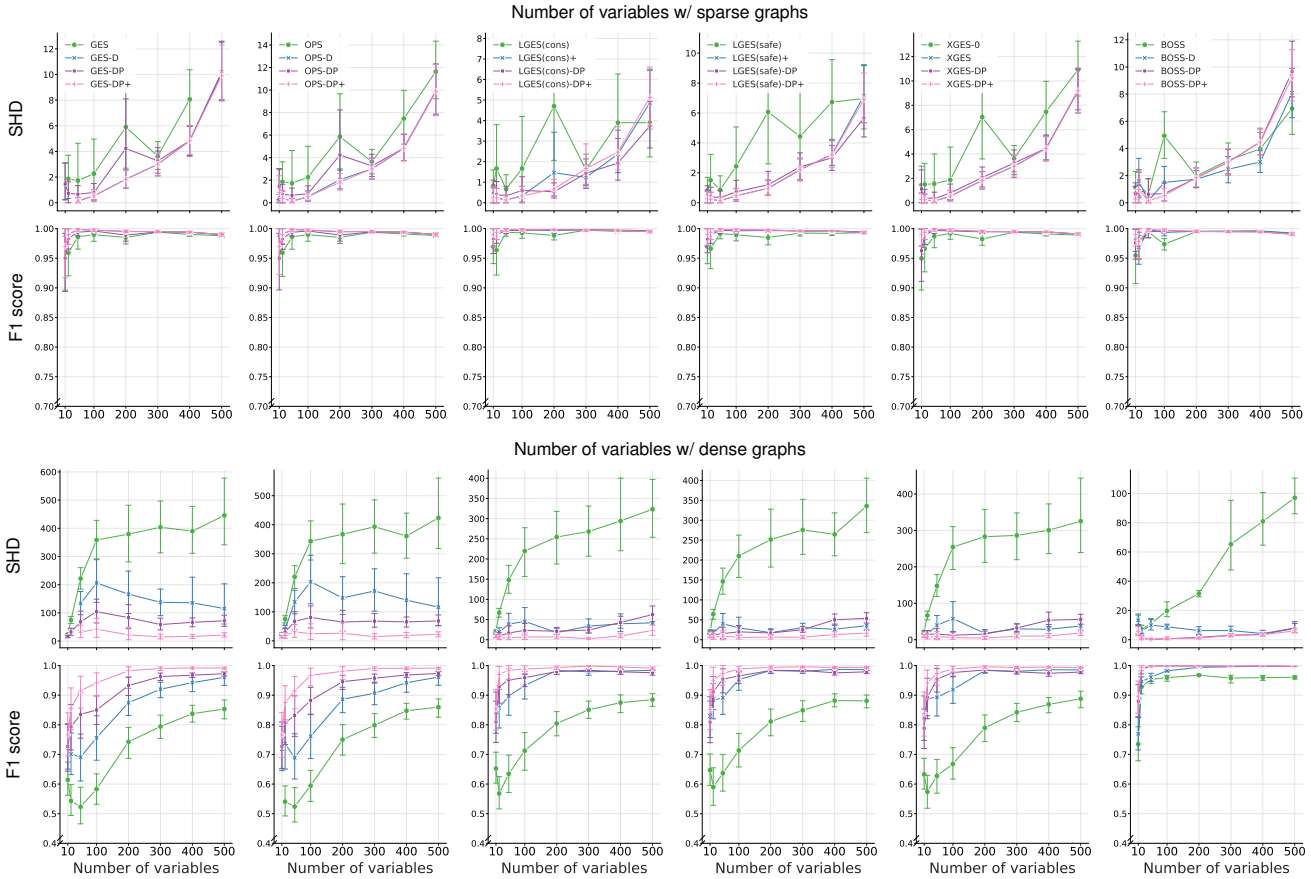


Figure 12: The top row corresponds to $\rho = 2$ (sparse), and the bottom row corresponds to $\rho = 5$ (dense). SHD(↓) and F1 score(↑) for the GES, OPS, LGES (CONS), LGES (SAFE), XGES, and BOSS variants (from left to right) as **the number of variables** d **varies**. Colors indicate the four compared variant groups: **non-ILS** variants, **single-edge deletion** variants, **DP** variants, and **DP+** variants. The vertical lines are computed as bootstrapped 95% confidence intervals over random seeds. The y-axes for F1 scores are truncated at 0.7 and 0.4 for visibility, respectively.

Fig. 12 presents the full comparison across baselines as the number of variables d increases over $d \in \{10, 20, 50, 100, 200, 300, 400, 500\}$, while fixing $n = 10^4$ and $\lambda = 2$, under both relatively sparse ($\rho = 2$) and dense graphs ($\rho = 5$).

- In denser graphs, DP and DP+ variants maintain clear advantages over other variants in moderate dimensions. However, as d becomes very large (i.e., $d = 500$), the performance gap gradually narrows, and DP variants can even exhibit slightly lower F1 than the single-edge deletion variant. This trend is consistent with the fact that, under fixed ρ, the edge probability scales as $\frac{\rho}{d-1}$. Consequently, even in the relatively high expected-degree regime, **increasing** d **makes the graph sparser**, reducing the benefit of parent deletion-based perturbations.
- In relatively sparse graphs, performance differences diminish more rapidly as d grows, since localized perturbations are already sufficient to escape shallow local optima.

Taken together, the density and dimensionality experiments (Secs. E.3.1 and E.3.2) indicate that the relative benefit of parent deletion depends jointly on graph density and dimensionality. Nevertheless, the parent deletion-based perturbation strategies remain consistently competitive across scales and retain clear advantages in moderately to highly dense graphs, where search complexity is more pronounced.

E.3.3 Performance under Varying Sample Sizes

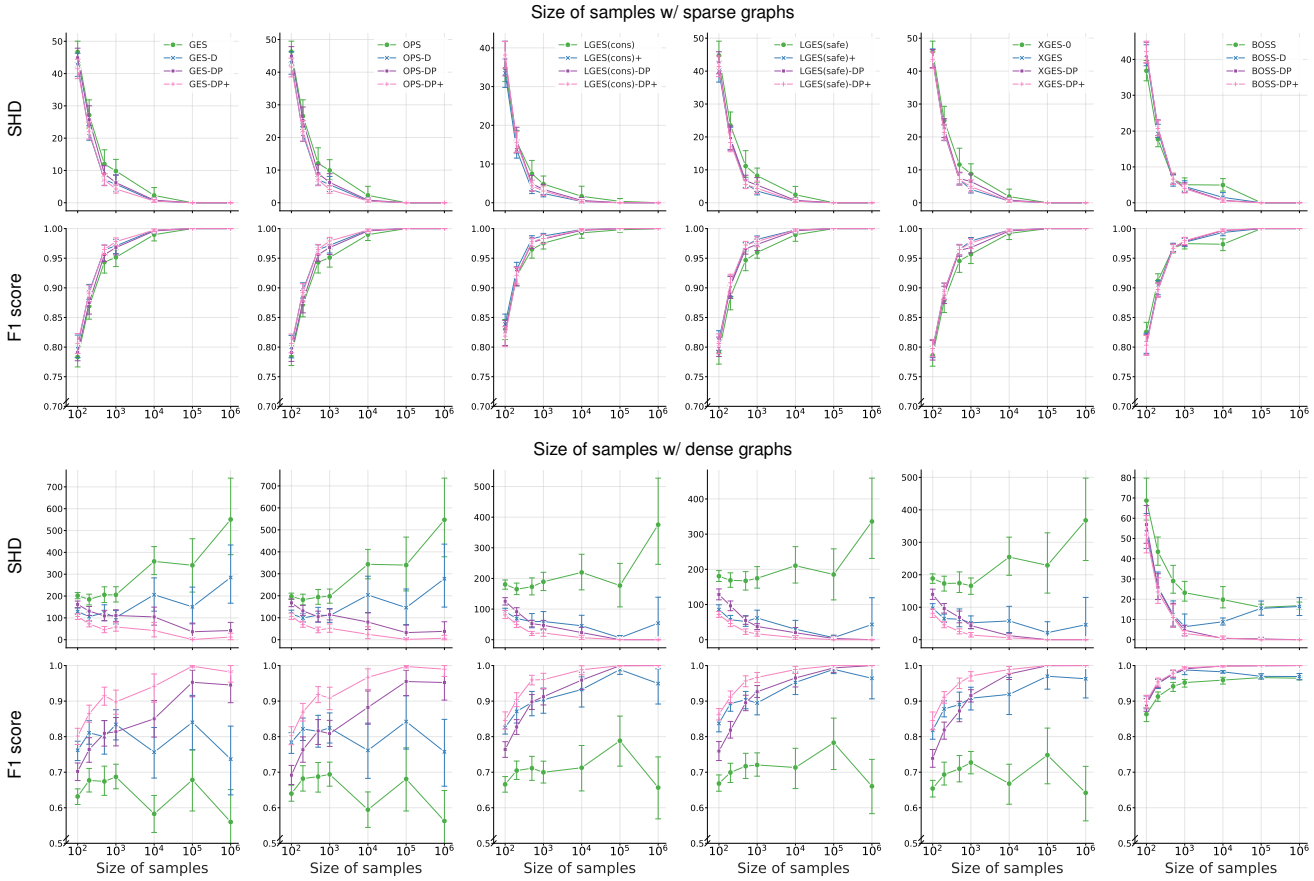


Figure 13: The top row corresponds to $\rho = 2$ (sparse), and the bottom row corresponds to $\rho = 5$ (dense). SHD(↓) and F1 score(↑) as **the sample size n varies**. Colors indicate the four compared variant groups: **non-ILS** variants, **single-edge deletion** variants, **DP** variants, and **DP+** variants. The vertical lines are computed as bootstrapped 95% confidence intervals over random seeds. The y-axes for F1 scores are truncated at 0.7 and 0.5 for visibility, respectively.

Fig. 13 reports SHD and F1 scores as the sample size n increases over $n \in \{10^2, 200, 500, 10^3, 10^4, 10^5, 10^6\}$, while fixing $d = 100$ and $\lambda = 2$, under relatively sparse ($\rho = 2$) and dense graphs ($\rho = 5$).

- In denser graphs, non-ILS and single-edge deletion variants often fail to exhibit monotonic improvement and may even degrade as n increases. In contrast, DP and particularly DP+ demonstrate more stable and steadily improving performance, consistently achieving lower SHD and higher F1 scores across all baselines.

  The performance gap becomes increasingly pronounced with larger sample sizes, suggesting that parent deletion-based perturbations may more effectively leverage additional information in complex search landscapes. In other words, DP and DP+ appear to benefit more from the increased signal, which may reduce the likelihood of convergence to suboptimal local structure.

- In relatively sparse graphs, all strategies attain similarly high F1 scores with minimal differences in SHD and F1 scores. The relative gains from perturbation strategies are therefore less pronounced, reflecting the reduced prevalence of severe local optima in relatively sparse graphs. Most variants achieve higher F1 scores in sparse settings, whereas BOSS exhibits better and more stable performance in dense graphs.

E.3.4 Robustness to BIC Hyperparameters

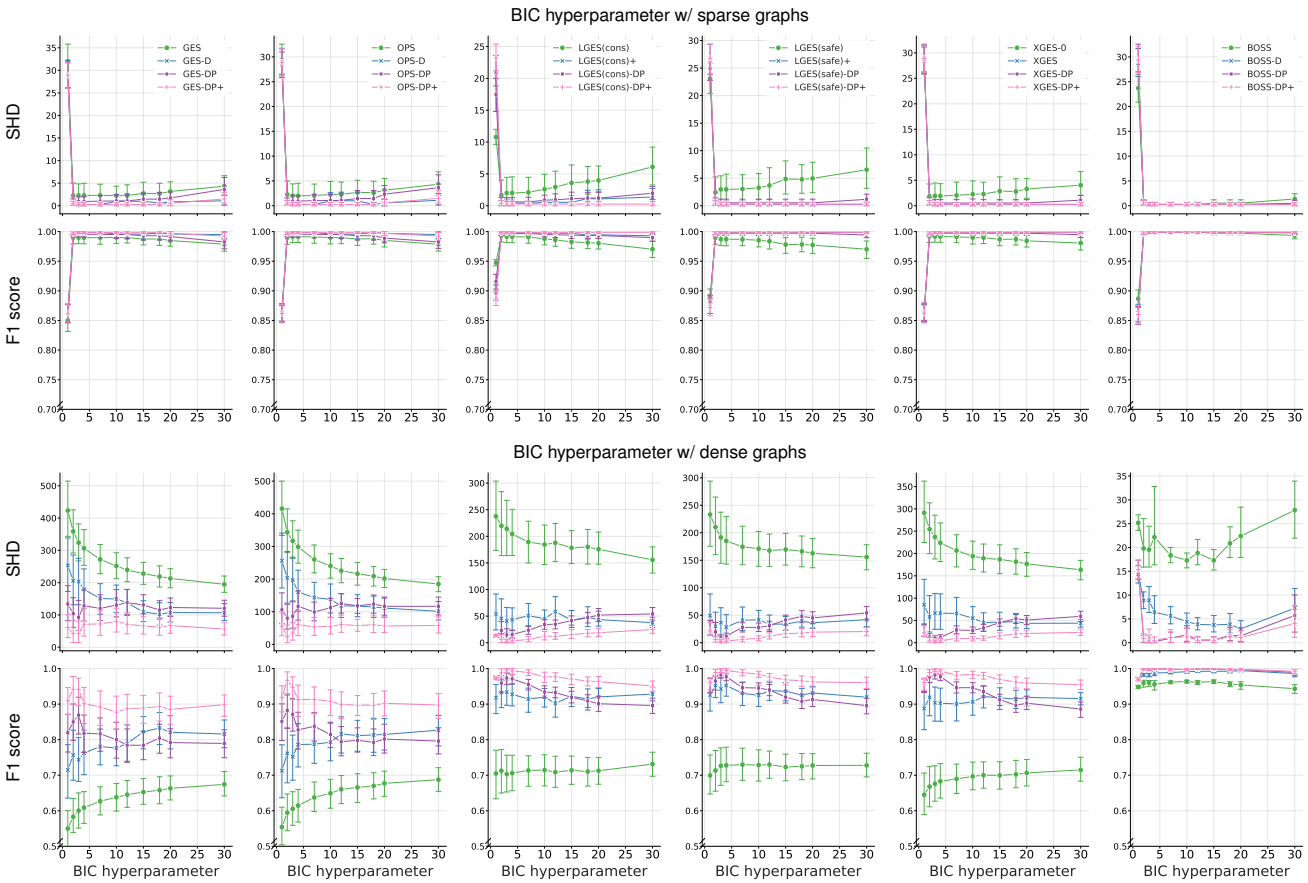


Figure 14: The top row corresponds to $\rho = 2$ (sparse), and the bottom row corresponds to $\rho = 5$ (dense). SHD(↓) and F1 score(↑) as **the BIC hyperparameter λ varies**. Colors indicate the four compared variant groups: **non-ILS** variants, **single-edge deletion** variants, **DP** variants, and **DP+** variants. The vertical lines are computed as bootstrapped 95% confidence intervals over random seeds. The y-axes for F1 scores are truncated at 0.7 and 0.5 for visibility, respectively.

To evaluate robustness to the score hyperparameter, we gradually vary $\lambda \in \{1, 2, 3, 4, 7, 10, 12, 15, 18, 20, 30\}$ while fixing $d = 100$ and $n = 10^4$ under both relatively sparse ($\rho = 2$) and dense graphs ($\rho = 5$). Fig. 14 demonstrates the results.

- In denser graphs, DP and DP+ outperform other strategies across a wide range of λ values, including the settings $\lambda \in \{1, 2\}$ used in established works such as Andrews et al. [2023], Nazaret and Blei [2024] and Ejaz and Bareinboim [2025]. **When $\lambda \geq 15$, DP may exhibit slightly lower F1 scores than the single-edge deletion strategy**, while consistently requiring substantially less runtime; the runtime advantage will be addressed in the following section. In contrast, DP+ maintains the lowest SHD and the highest F1 score across the entire range of λ.
- In sparse graphs, DP, DP+, and the single-edge deletion strategy all achieve high F1 scores, with no meaningful differences.

As λ increases, SHD and F1 scores often improve, although this trend is not universal. In the BIC score, λ acts as a penalty on parent set size, discouraging overly complex structures. While larger λ values can suppress excessive parent growth and reduce over-insertion errors, overly strong regularization may hinder the recovery of true dependencies.

Consequently, the optimal choice of λ is instance-specific. Thus, one may adopt a principled hyperparameter-selection strategy, such as Bayesian optimization [Frazier, 2018], to determine an appropriate value of λ.

E.3.5 Runtime Analyses

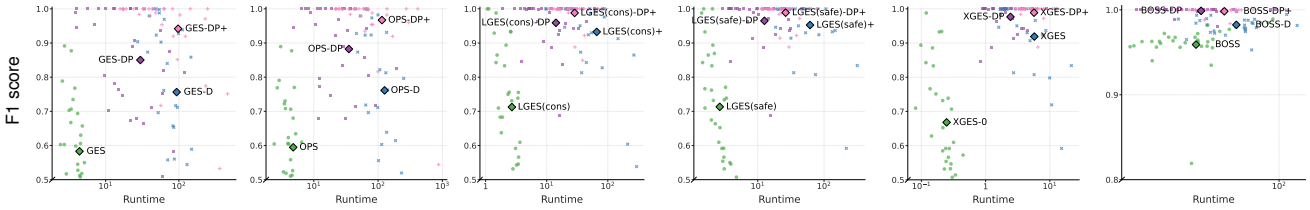


Figure 15: **Runtime versus F1 score** in relatively dense graphs ($\rho = 5$). Each panel corresponds to GES, OPS, LGES (CONS), LGES (SAFE), XGES, and BOSS variants (from left to right) as **the number of variables d varies**. Colors indicate the four compared variant groups: **non-ILS** variants, **single-edge deletion** variants, **DP** variants, and **DP+** variants. Points represent individual seeds, and diamonds ($\diamond$) indicate mean performance. The y-axes for F1 scores are truncated at 0.8 for BOSS and 0.5 for others.

We report the wall-clock time (in seconds) for all algorithms across diverse experimental settings. Runtimes are reported on a log scale.

- **RT vs. F1 in dense graphs (Fig. 15).** We present runtime versus F1 score with $d = 100$, $n = 10^4$, $\rho = 5$, $\lambda = 2$. We set $\rho = 5$ (relatively dense) to ensure a fair comparison.

  Across all baselines, DP and DP+ achieve higher F1 scores than the other variants. In this regime, DP variants consistently require less runtime than single-edge deletion strategies for all baselines. The runtime of DP+ variants varies across baselines: for GES and XGES, DP+ variants take more time than the other variants, whereas for OPS, LGES, and BOSS, DP+ variants require less runtime than single-edge deletion strategies.

- **Runtime under varying factors (Fig. 16).** We further examine runtime while varying one key factor at a time: expected degree ρ, number of variables d, sample size n, and BIC hyperparameter λ. When varying one factor, the others are fixed to $\rho = 5$, $d = 100$, $n = 10^4$, and $\lambda = 2$.

  - **Runtime increases with** $\rho \in [1, 10]$, reflecting the increase in complexity of the search space. As shown in Fig. 1, score-based methods tend to introduce a large number of parents, resulting in more complex intermediate graphs. Consequently, additional time is required to traverse and refine these structures during perturbation. For GES, OPS, and BOSS, DP variants exhibit runtime comparable to single-edge deletion strategies, while DP+ variants generally incur the highest runtime. In contrast, for LGES and XGES, DP achieves lower runtime than other perturbation strategies, whereas DP+ and single-edge deletion variants show similar computational cost.
  - **As** $d \in \{10, 20, 50, 100, 200, 300, 400, 500\}$ **increases, runtime grows for all baseline algorithms** as the number of candidate operations expands with the number of variables. Nevertheless, DP maintains lower runtime than other perturbation strategies across most baselines, especially in denser graphs.
  - **Increasing** $n \in \{10^2, 200, 500, 10^3, 10^4, 10^5, 10^6\}$ **generally stabilizes or reduces runtime**, as large samples improve performance. DP and DP+ achieve stable or lower runtime with more samples, while single-edge deletion strategies require longer runtime with more samples in denser graphs. These results suggest that DP and DP+ leverage larger sample sizes more efficiently, consistent with the improved F1 trends observed in Fig. 13.
  - **As** $\lambda \in \{1, 2, 3, 4, 7, 10, 12, 15, 18, 20, 30\}$ **increases, runtime decreases** across most baseline algorithms, especially in denser graphs. Since larger λ penalizes large parent sets, it suppresses overly complex intermediate graphs and reduces the search space.

- **RT on scale-free and biological benchmarks (Fig. 17).** We examine runtime on both scale-free DAGs with density factor $m \in \{2, 4, 6\}$ and the biological benchmarks.

  As shown in Fig. 17, DP variants consistently achieve the lowest runtime among perturbation strategies. While DP+ variants generally incur higher runtime than single-edge deletion strategies, they remain competitive in runtime while maintaining higher F1 scores.

  Overall, these results indicate that parent deletion-based perturbations not only achieve high F1 scores but also maintain competitive runtime across various settings, including both synthetic and realistic datasets.

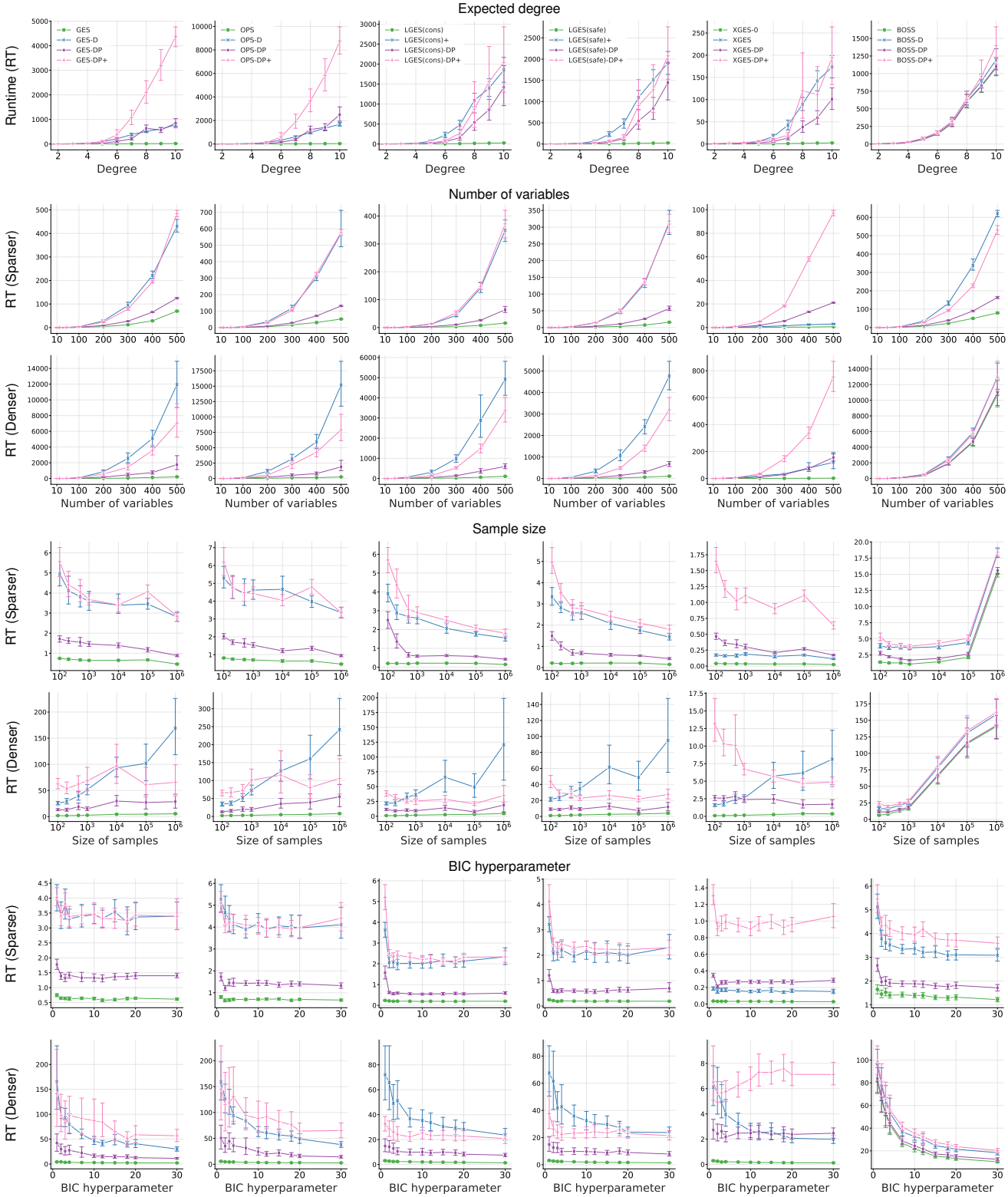


Figure 16: Comprehensive runtime analysis across **varying experimental factors** ρ, d, n, **and** λ. Colors indicate the four compared variant groups: **non-ILS** variants, **single-edge deletion** variants, **DP** variants, and **DP+** variants.

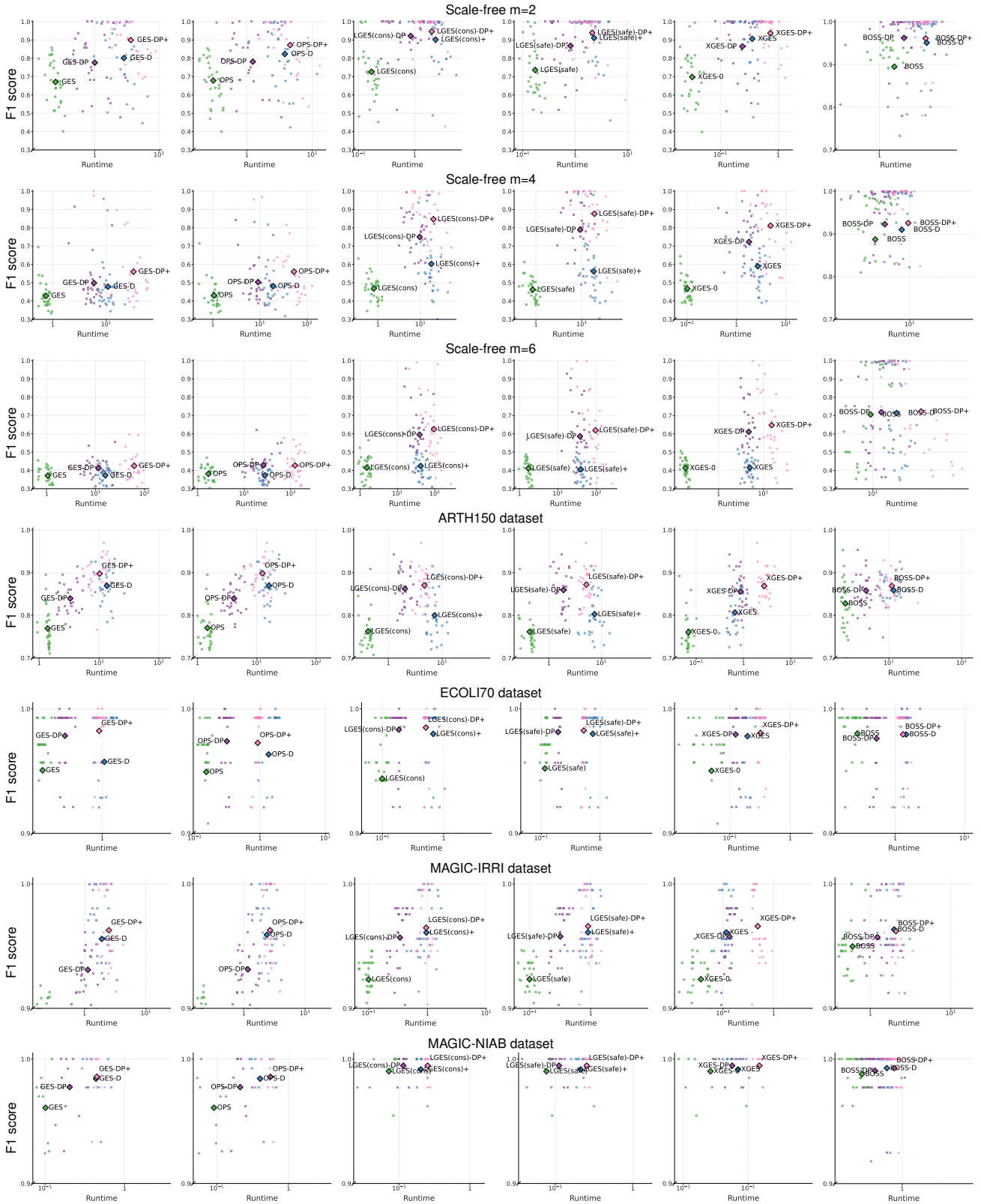


Figure 17: **Runtime versus F1 score** across scale-free DAGs with **varying degree parameter** m and biological benchmarks. The top three panels correspond to scale-free DAGs with $m = 2$, $m = 4$, and $m = 6$, respectively, while the bottom four panels report results on biological benchmarks. Colors indicate the four compared variant groups: **non-ILS** variants, **single-edge deletion** variants, **DP** variants, and **DP+** variants. Points represent individual seeds, and diamonds ($\diamond$) indicate mean performance. The y-axes for F1 scores are truncated at 0.3 for scale-free DAGs, 0.7 for ARTH150, and 0.9 for the other three benchmarks.

E.3.6 Comparison with Single-Node Parent Deletion Strategy

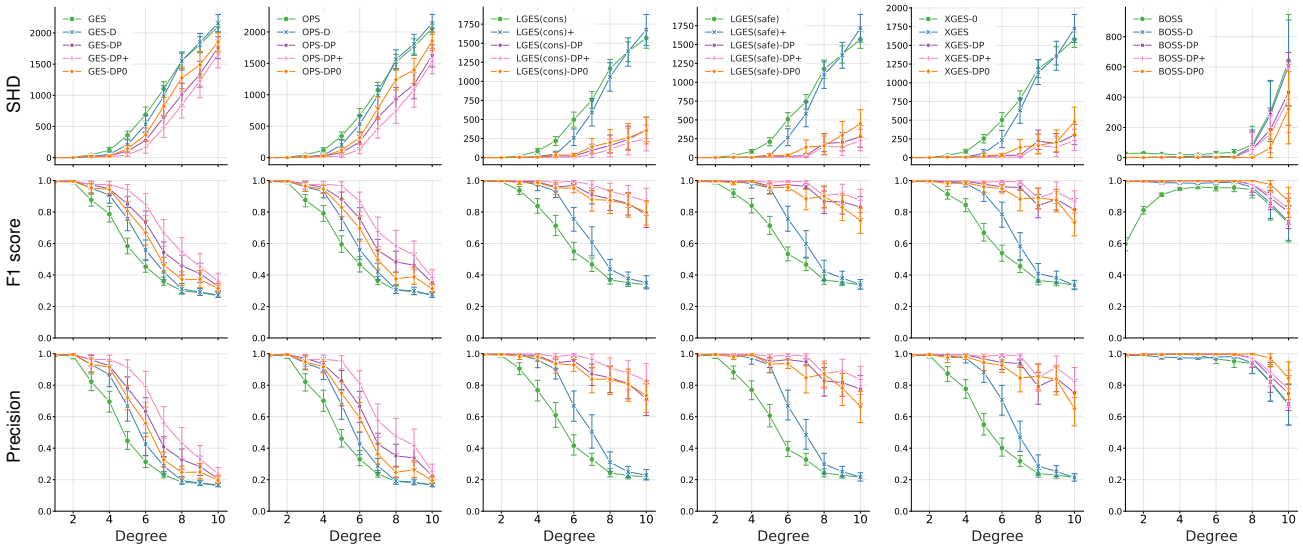


Figure 18: SHD(↓), F1 score(↑), and Precision(↑) for the GES, OPS, LGES (CONS), LGES (SAFE), XGES, and BOSS variants (from left to right) as **the expected degree ρ varies**. Colors indicate the five compared variant groups: **non-ILS** variants, **single-edge deletion** variants, **DP** variants, **DP+** variants, and **DP0** variants using a single-node parent deletion strategy. The vertical lines are computed as bootstrapped 95% confidence intervals over random seeds.

We evaluate a standalone single-node parent deletion variant (DP0) across all baselines. This experiment assesses whether deleting the parent edges of a single node is sufficient. We compare DP0 with the other variants under the setting $d = 100$, $n = 10^4$, and $\lambda = 2$, while varying the expected degree $\rho \in [1, 10]$. Fig. 18 shows the results.

DP0 variants consistently improve upon both the non-ILS baselines and single-edge deletion variants, demonstrating the effectiveness of parent deletion as a perturbation mechanism. However, DP variants, and particularly DP+ variants, generally achieve higher F1 scores, especially in denser settings, suggesting that component-wise parent deletion perturbations can more effectively escape local optima.

E.3.7 Comparison with PC Algorithm

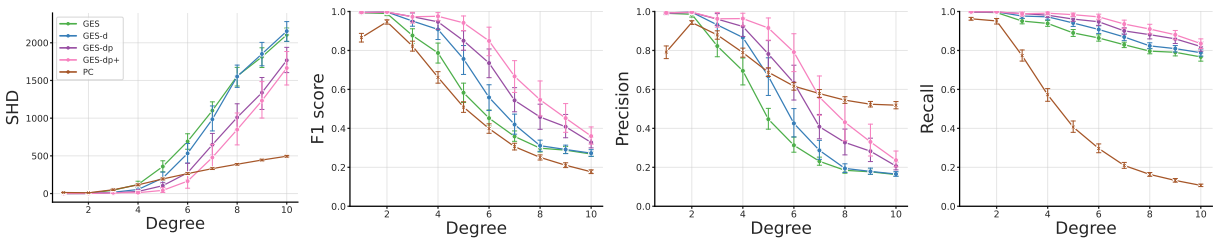


Figure 19: Comparison of SHD(↓), F1(↑), Precision(↑), and Recall(↑) for **PC** and four GES variants as ρ **varies**.

Constraint-based causal discovery methods such as Peter–Clark (PC) [Spirtes et al., 2000] and Sparsest Permutations (SP) [Raskutti and Uhler, 2018] recover the MEC by using conditional independence tests to infer the skeleton and orient edges. While there is no general claim about the relative performance of these methods, empirical analyses by Tsamardinos et al. [2006] and Ejaz and Bareinboim [2025] have reported that GES-family methods outperform PC across various sample sizes.

Nevertheless, for concreteness, we provide a brief comparison with PC, a representative constraint-based causal discovery

method. We set the significance level to $\alpha = 0.01$ for conditional independence tests. Fig. 19 shows the performance of GES variants and PC as ρ varies over $\rho \in [1, 10]$, while fixing $d = 100$, $n = 10^4$, and $\lambda = 2$.

While PC does not exhibit a drastic increase in SHD as ρ grows, its recall deteriorates substantially, leading to consistently lower F1 scores than *all* GES variants. This pattern suggests that PC tends to predict fewer edges than are present in the ground truth. This behavior appears to stem from the structural nature of the PC algorithm, which starts from a fully connected graph skeleton and iteratively removes edges through conditional independence testing. In contrast, score-based methods are more prone to over-insertion but generally achieve higher recall and, consequently, better overall F1 performance.

E.3.8 Evaluation with Adjustment Identification Distance

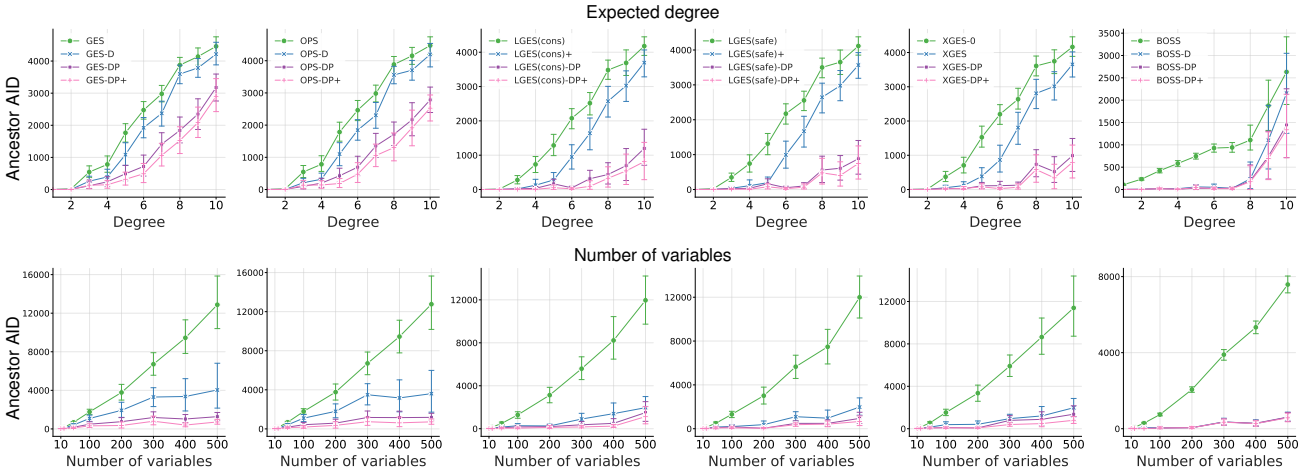


Figure 20: Ancestor-AID on ER DAGs with **varying expected degree** ρ and **varying number of variables** d. The top and bottom panels vary the expected degree and the number of variables, respectively. Colors indicate the four compared variant groups: **non-ILS** variants, **single-edge deletion** variants, **DP** variants, and **DP+** variants.

We evaluate Ancestor Adjustment Identification Distance (Ancestor-AID), which quantifies errors in causal effect estimation arising from incorrect covariate adjustment. This experiment aims to assess whether the proposed perturbation methods improve not only structural accuracy but also causal effect estimation.

We consider two sweeps: varying the expected degree $\rho \in [1, 10]$ while fixing $d = 100$, $n = 10^4$, and $\lambda = 2$; and varying the number of variables $d \in \{10, 20, 50, 100, 200, 300, 400, 500\}$ while fixing $\rho = 5$, $n = 10^4$, and $\lambda = 2$.

As shown in Fig. 20, Ancestor-AID increases for all methods as the graphs become denser (larger ρ) or higher-dimensional (larger d), indicating the growing difficulty of recovering the causal structure, with DP and DP+ variants increasing more gradually and remaining more stable than the other variants. Across all baselines, DP variants and DP+ variants consistently achieve lower Ancestor-AID scores than non-ILS methods and single-edge deletion variants. Moreover, the performance gap widens as the parameters increase. This relative performance is aligned with the trends observed under other evaluation metrics.

These results suggest that the improvements from parent deletion are not confined to raw structural accuracy but also yield **more reliable causal effect estimation**.

E.3.9 Results on Scale-Free DAGs

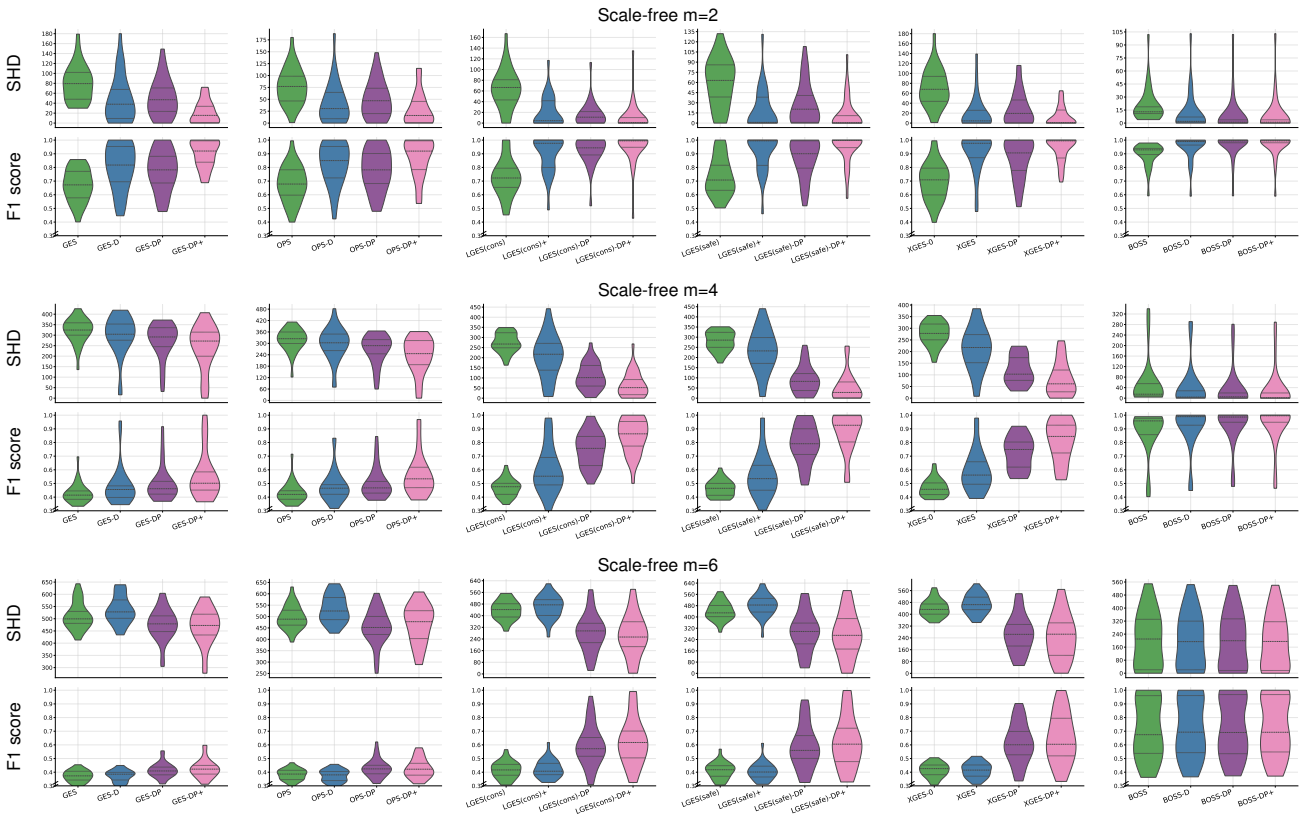


Figure 21: Violin plots for scale-free graphs **varying degree parameter** $m \in \{2, 4, 6\}$ where the horizontal lines denote the 75_{th}, 50_{th}, and 25_{th} percentiles. Colors indicate the four compared variant groups: **non-ILS** variants, **single-edge deletion** variants, **DP** variants, and **DP+** variants. The y-axes for F1 scores are truncated at 0.3 for visibility.

We further evaluate structural recovery on scale-free DAGs under multiple attachment settings $m \in \{2, 4, 6\}$, while fixing $d = 50$, $n = 10^3$, and $\lambda = 2$. As shown in Fig. 21, performance differs across density regimes induced by varying m. Since larger m produces denser graphs, recovery becomes more challenging: Consistent with the ER experiments, SHD tends to increase and F1 tends to decrease in denser regimes.

Nevertheless, DP and DP+ remain consistently competitive across all values of m. In particular, their relative advantage is more evident at $m = 6$, where high-degree nodes create more complex local structures. These results indicate that parent deletion-based perturbations are robust across a range of attachment configurations and heterogeneous degree distributions.

E.3.10 Results on Biologically Motivated Benchmarks

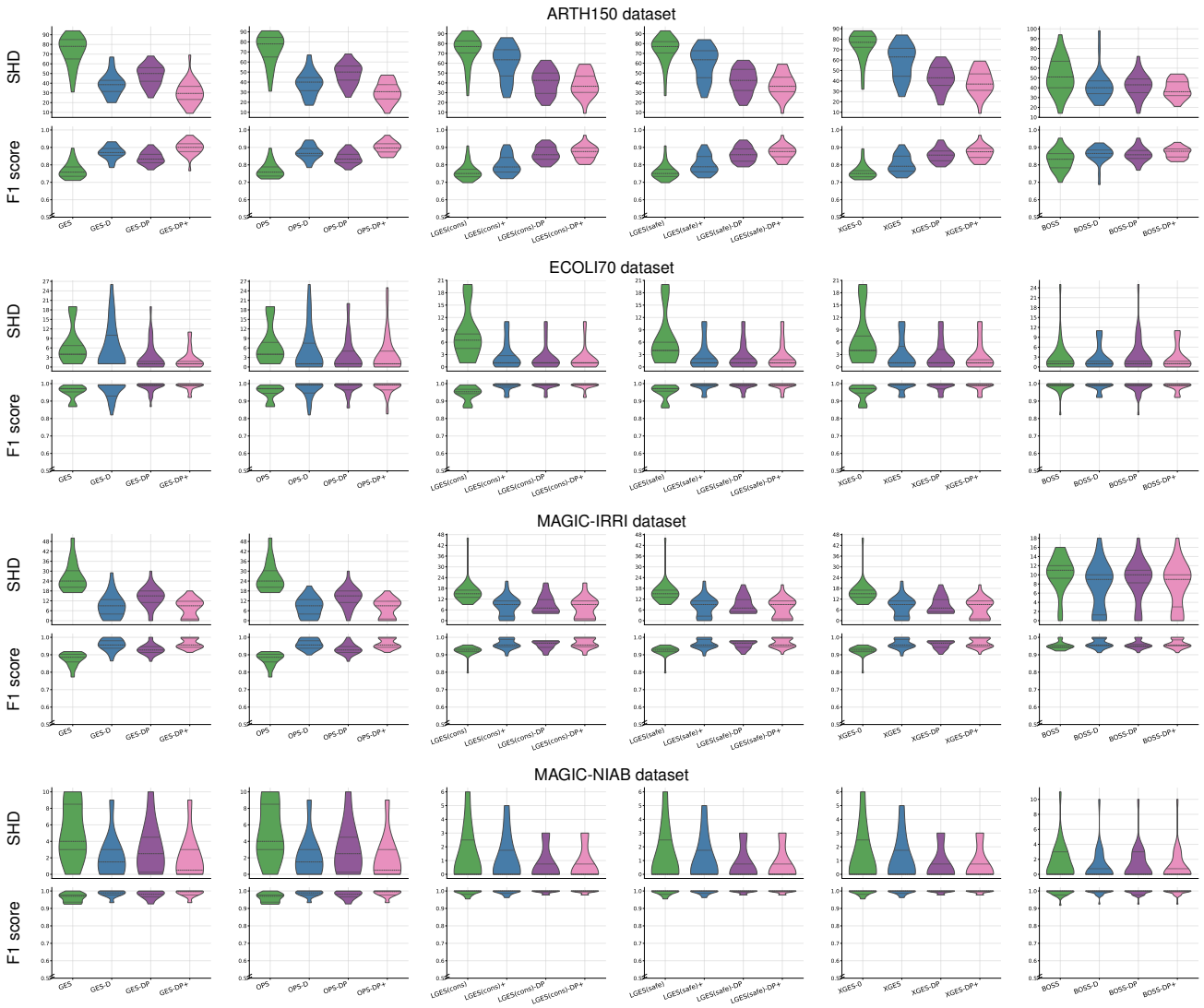


Figure 22: Violin plots on four biological benchmarks where the horizontal lines denote the 75_{th}, 50_{th}, and 25_{th} percentiles. Colors indicate the four compared variant groups: **non-ILS** variants, **single-edge deletion** variants, **DP** variants, and **DP+** variants. The y-axes for F1 scores are truncated at 0.5 for visibility.

Fig. 22 presents the full results on the four biological network benchmarks (ARTH150, ECOLI70, MAGIC-IRRI, and MAGIC-NIAB). Across baselines, DP and DP+ variants consistently achieve lower SHD and higher F1 scores than their non-perturbed counterparts. For GES and OPS, DP+ variants attain the best overall performance. For LGES, XGES, and BOSS, both DP and DP+ variants outperform single-edge deletion and non-ILS variants.

These results indicate the benefits of the proposed method in realistic benchmarks beyond synthetic data, supporting its practical applicability to real-world problems.

E.3.11 Results under Assumption Violations

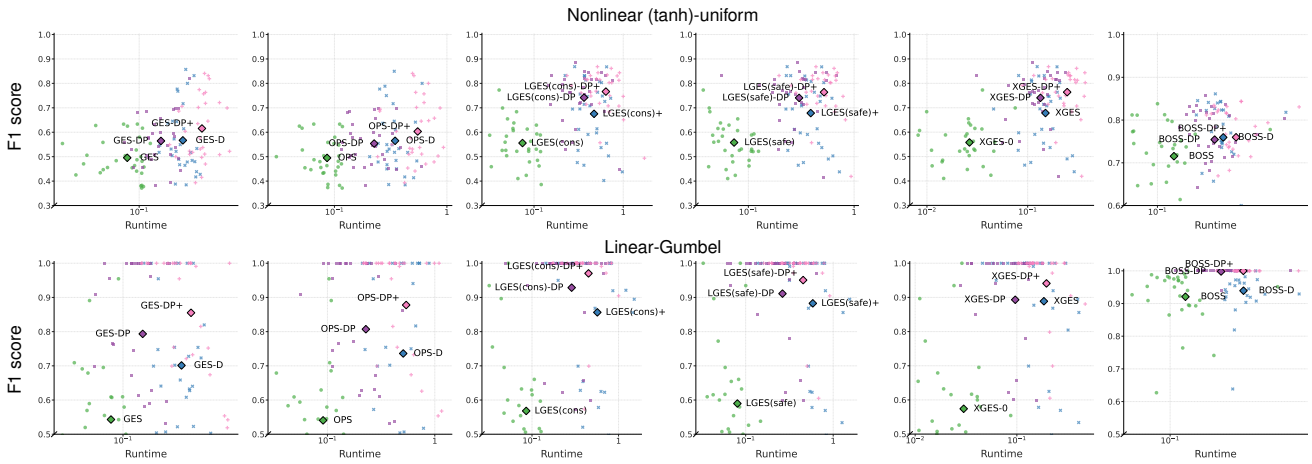


Figure 23: **Runtime versus F1 score** under a **nonlinear** (tanh) **additive uniform** noise model (top) and a **linear additive Gumbel** noise model (bottom), with $d = 20$, $\rho = 5$, and $n = 10^4$. Colors indicate the four compared variant groups: **non-ILS** variants, **single-edge deletion** variants, **DP** variants, and **DP+** variants. Points represent individual seeds, and diamonds ($\diamond$) indicate mean performance. The y-axes for F1 scores are truncated at 0.3 and 0.5 for visibility, respectively.

In practical applications, the true data-generating process is often unknown, and the linear-Gaussian assumptions on the underlying mechanism may therefore be misspecified. Nevertheless, our results shown in Fig. 23 demonstrate that the proposed methods remain beneficial under such assumption violations.

To evaluate the robustness of our method beyond the standard linear-Gaussian setting, we provide additional experiments under (i) nonlinear (tanh) additive uniform noise models, $v = \tanh(\mathbf{w}_V^\top \mathbf{pa}_V^{\mathcal{G}}) + \varepsilon_V$, where $\varepsilon_V \sim \texttt{Unif}([-1, 1])$; and (ii) linear additive Gumbel noise models, $v = \mathbf{w}_V^\top \mathbf{pa}_V^{\mathcal{G}} + \varepsilon_V$, where $\varepsilon_V \sim \texttt{Gumbel}(0, 1)$. For both settings, we generate ER DAGs with the number of variables $d = 20$ and expected degree $\rho = 5$, and sample size $n = 10^4$ for each DAG.

Across both settings, DP+ variants consistently achieve the highest F1 scores, while DP variants generally outperform both the non-ILS variants and single-edge deletion variants. These results suggest that the proposed perturbation strategies remain effective even when the linear-Gaussian assumption is violated.

E.4 GRID CONFIGURATION RESULTS

For comprehensive evaluation across diverse experimental settings, we assess **all variants** listed in Table 2 over a **total of 81** grid configurations, varying $d \in \{50, 100, 200\}$, $\rho \in \{2, 3, 5\}$, $n \in \{10^3, 10^4, 10^5\}$ and $\lambda \in \{1, 2, 4\}$. The complete results are reported in the following tables.

For clarity, we present separate tables for each baseline algorithm and each number of variables (e.g., GES variants with $d = 50$). Each algorithm is evaluated using SHD, F1 score, and runtime (RT). All reported values are the mean $\pm$ standard deviation over 30 random seeds. For ease of comparison, the best and second-best results are highlighted in **bold** and underlined, respectively. For runtime, the same highlighting rule is applied, excluding the non-ILS baseline (e.g., GES).