Contents

1 Introduction

In the field of machine learning, causal discovery plays a central role in helping researchers and practitioners move beyond correlation to identify underlying cause-effect relationships. While associational relationships describe patterns or dependencies in the data (e.g., two variables increasing together), causal relationships imply that intervening on one variable will directly influence the other. This distinction matters because many decisions in fields like healthcare, economics, marketing and finance depend not just on predicting outcomes, but on understanding what actions will change them.

Directed Acyclic Graphs (DAGs) are the standard framework for encoding such structures, and a growing body of work has focused on developing algorithms to learn these graphs from observational data–data that is collected without actively intervening in the system. Unlike experimental or interventional data, where variables are deliberately manipulated to observe effects, observational data simply records what happens naturally, making it harder to distinguish causation from correlation. A recurring challenge, however, is how to evaluate whether these algorithms actually recover the correct DAG structure, especially when ground-truth causality is often unknown in real-world data. As a result, most causal discovery methods are benchmarked on **synthetic datasets** where the true DAG is known by construction.

Over the past decade, numerous algorithms have been developed to learn DAGs from data, ranging from constraint-based approaches like the Peter-Clark (PC) algorithm [13], to score-based methods like Greedy Equivalence Search (GES) [2], and more recent optimization-based techniques such as Non-combinatorial Optimization via Trace Exponential and Augmented Lagrangian for Structure learning (NOTEARS) [16], Directed Acyclic Graphs via M-matrices and Acyclicity characterization (DAGMA) [1], and Sparse Regression for Causation (SparseRC) [8]. In the literature, these algorithms are evaluated on synthetically generated data, where the ground truth is known. However, recent evidence from a 2021 paper by Reisach, Seiler, and Weichwald titled "Beware of the Simulated DAG! Causal Discovery Benchmarks May Be Easy to Game" [12] suggests that this standard evaluation strategy using synthetic datasets may be misleading.

The central claim of Reisach et al. (2021) was that synthetic benchmarks used in causal discovery often contain an overlooked statistical regularity: **variables later in the causal chain tend to have higher marginal variances**. This property, what they call *varsortability*, means that the variables can be partially ordered by their marginal variances in a way that aligns with the underlying causal structure. This alignment, they argued, makes causal discovery artificially easier, as some algorithms may implicitly exploit this property to reconstruct graph structure, and appear to

perform well on these benchmarks, even if they wouldn't be effective on real-world data without such patterns. When the synthetic datasets were standardized to remove varsortability, performance for some optimization-based algorithms declined significantly. The core issue raised in that paper is a methodological one. If benchmarks are structured in a way that makes the target too easy to infer, the results I get from evaluating algorithms on them may give a false sense of progress. This raised a larger concern of overestimating the capabilities of modern causal discovery methods due to the unintended simplicity of synthetic data.

This thesis builds upon that question through a two-phase research project. The first phase focused on replicating the key results of Reisach et al. (2021) to better understand the extent to which varsortability affects structure learning in traditional algorithms. I reimplemented their experimental setup using Erdős-Rényi (ER) and Scale-Free (SF) graphs with Gaussian, Gumbel, and Exponential noise. I demonstrated that marginal variances often increase along the causal order, and that when data is standardized, the performance of some algorithms drops sharply. I also evaluated the impact of standardization algorithms that do not rely on optimization, and found that their performance remained stable. This provided early evidence that not all algorithms are equally sensitive to the varsortability property.

The second phase of the thesis broadens the investigation. Despite the findings from Reisach et al. (2021), many recent papers published after 2021 continue to use synthetic data to benchmark causal discovery algorithms. This raises an important question about whether current research has internalized the critique. In this phase, I surveyed a set of post-2021 papers that benchmark structure learning algorithms on synthetic datasets. I categorized these papers based on whether their evaluation practices were likely affected by varsortability and identified which algorithms were used.

To empirically assess the robustness of modern algorithms under these conditions, I evaluated five causal discovery algorithms in Phase 2 - two of which are post-2021, two are traditional, along with a baseline algorithm that relies on variance sorting introduced by Reisach et al. (2021), PC (1991) [13], NOTEARS (2018) [16], Sort-and-Regression Baseline (2021) [12], DAGMA (2022) [1], and SparseRC (2023) [8] - on both raw and standardized synthetic datasets. I measured their performance using metrics like Structural Hamming Distance (SHD) [14], Structural Intervention Distance (SID) [11], and the Balanced Scoring Function (BSF) [5]. The goal was to determine whether performance differences arise between raw (potentially varsortable) and standardized (non-varsortable) datasets, and whether the newer algorithms are reliably robust across both.

This research is significant for two main reasons. First, it addresses a growing concern in

machine learning about the reliability of benchmarks and the extent to which they reflect real-world complexity. Second, it contributes to the ongoing development of causal discovery by helping identify which methods are robust to spurious signals and which are not. In doing so, it encourages a more critical approach to evaluation and opens the door to better benchmark design in the future.

The central research question guiding this project is – **To what extent are modern causal discovery algorithms affected by varsortability in synthetic data, and how does their performance change when this property is removed or controlled?** Answering this question will help clarify whether reported gains in structural recovery are due to actual algorithmic improvements or to artifacts in how I simulate and evaluate causal data. Ultimately, this project aims to push the field toward more trustworthy causal inference by tightening the link between evaluation and real-world applicability.

2 Related Work

A central challenge of causal discovery has been how to evaluate new algorithms fairly and rigorously. This challenge has only grown more pressing in light of methodological critiques, most notably the observation that synthetic data, when not carefully designed, can inadvertently favor certain algorithmic approaches. The starting point for this project was a close study of Reisach et al. (2021)'s 2021 paper [12], Beware of the Simulated DAG!, which argued that a hidden property in many synthetic datasets, varsortability, could cause inflated performance in DAG recovery tasks. That work cast doubt on the validity of many benchmarks, specifically for algorithms like NOTEARS that rely on continuous optimization and could be inadvertently learning marginal variance patterns rather than true causal structure.

In Phase 1 of this thesis, I directly engaged with the Reisach et al. (2021) critique by replicating their experiments. This work provided a hands-on understanding of how varsortability manifests in common data generation processes and confirmed their results regarding the sensitivity of algorithms like NOTEARS. This replication served as a foundation for the second phase of the project, where I analyze whether newer papers have addressed or repeated the same underlying issues.

Since that publication, numerous papers have continued to evaluate causal discovery algorithms using synthetic data. Many of these studies replicate earlier simulation setups, often using additive noise models (ANMs) or linear Structural Equation Models (SEM) over ER or SF graphs. The fact that these methods were published in top-tier venues despite overlooking this

issue underscores the importance of revisiting how I evaluate causal discovery algorithms. What distinguishes this project is its focus on those post-2021 papers that have either overlooked or insufficiently addressed the varsortability issue despite the growing awareness in the field. To structure this review, I have organized the literature into two categories: (1) papers using synthetic data without controlling for varsortability; and (2) papers that acknowledge or respond to the varsortability issue

2.1 Papers Using Synthetic Data Without Controlling for Varsortability

One clear example is Global Optimality in Bivariate Gradient-based DAG Learning (NeurIPS 2023) [6], which evaluates a proposed optimization technique using only synthetic data from bivariate SEMs with independent noise. While the paper offers a theoretical contribution in defining a global optimum in a constrained bivariate case, its empirical analysis rests on simulated data that may be vulnerable to the same marginal variance signal that Reisach et al. (2021) critiqued. There is no mention of data standardization or variance alignment, and the limited two-variable setting does not eliminate the potential for varsortability to play a role in recovery performance.

A more complex benchmark appears in Fast Scalable and Accurate Discovery of DAGs Using the Best Order Score Search and Grow Shrink Trees (NeurIPS 2023) [8]. Here, the authors combine pseudo-empirical noise distributions, derived from randomized fMRI cortical signals, with structural simulations. Despite its novel integration of neuroimaging priors, the study relies on simulated structural DAGs, and the paper does not explicitly address the possibility that marginal variances could correlate with the true causal ordering.

Verification and Search Algorithms for Causal DAGs (Neurips 2022) [3] takes a more abstract approach, and evaluates the correctness of DAG verification and search schemes using synthetic graphs of varying sizes. Again, the evaluation pipeline involves synthetic DAGs, but the authors do not reference the problem of variance-driven structure or control for it in their analysis.

Similarly, DAGMA: Learning DAGs via M-matrices and a Log-Determinant Acyclicity Characterization (Neurips 2022) [1] introduces a differentiable acyclicity constraint and tests the resulting algorithm on datasets generated by linear and non-linear SEMs. The graphs are synthetically generated, yet the potential for varsortability-induced biases is not examined. No standardization procedure is mentioned, and there is no indication that algorithm performance is tested under differing variance regimes.

Truncated Matrix Power Iteration for Differentiable DAG Learning (Neurips 2022) [15]

also relies on synthetic ER graphs to benchmark its proposed optimization strategy. Despite being a direct continuation of differentiable approaches like NOTEARS, this paper does not cite or engage with Reisach et al. (2021), and there is no mention of how marginal variance might influence results.

Together, these papers form the core of this project's analysis in this thesis. They each propose new methods or variants built on differentiable structure learning, yet none provide a robustness analysis around data standardization or marginal variance effects. This gap offers a clear opportunity for critical evaluation and replication.

2.2 Papers That Acknowledge or Respond to the Varsortability Issue

Not all recent work ignores this methodological concern. Learning DAGs from Data with Few Root Causes (NeurIPS 2023) [9] is an important exception. While the authors use synthetic data for initial evaluation, they explicitly cite the Reisach et al. (2021) [12] paper and acknowledge the need for caution in interpreting benchmark results. This awareness is reflected in their decision to examine scenarios with constrained causal root nodes, which can alter the marginal variance landscape.

Another key paper is Structure Learning with Continuous Optimization: A Sober Look and Beyond (2024) [10]. This paper takes a retrospective approach, reassessing the limitations of continuous optimization methods in light of critiques like Reisach et al. (2021)'s [12]. The authors explore potential failure modes beyond varsortability and suggest ways to improve robustness, including a discussion of non-uniform noise variances and non-convex optimization landscapes. Their perspective serves as a valuable counterpart to the algorithms being evaluated in this thesis.

Finally, unsuitability of NOTEARS for Causal Graph Discovery when Dealing with Dimensional Quantities (2022) [7] critiques NOTEARS for a related but distinct issue – its lack of scale invariance. The authors demonstrate that simple rescaling of input features can result in entirely different causal graphs being learned. This critique is closely related to varsortability, since both issues stem from the algorithm's sensitivity to marginal variances and other scale-dependent signals in the data.

2.3 Descriptions of Studied Algorithms

NOTEARS [16]. NOTEARS is a causal discovery algorithm that reformulates the problem of learning DAGs from observational data as a continuous optimization task. The main idea is to represent the DAG as a weighted adjacency matrix W and minimize a least-squares loss function that measures how well the observed data can be reconstructed from a linear structural equation model

parameterized by W. To guarantee acyclicity, NOTEARS introduces a differentiable constraint based on the trace of the matrix exponential, $h(W) = \mathrm{tr}(\exp(W \circ W)) - d = 0$, where d is the number of variables and $\circ$ denotes the Hadamard (element-wise) product. This formulation enables using gradient-based solvers and augmented Lagrangian methods to jointly minimize loss and enforce acyclicity. NOTEARS avoids discrete graph operations by relaxing the acyclicity constraint into a smooth surrogate, which allows for efficient and scalable learning of DAGs in high-dimensional linear settings, and can be extended to incorporate prior knowledge or additional constraints.

PC [13]. The PC algorithm is a constraint-based method for causal discovery that infers DAGs by systematically testing conditional independence relationships. It initializes a fully connected undirected graph and iteratively removes edges between variable pairs found to be conditionally independent given subsets of increasing size. Remaining edges are then oriented using logical rules to avoid cycles, which produces a partially directed acyclic graph representing the Markov equivalence class of the true DAG. The algorithm leverages the principle that conditional independencies imply the absence of direct causal links, and progressively refines the graph structure through statistical testing. While computationally intensive for large graphs, PC's non-parametric nature and theoretical guarantees have made it a benchmark in causal discovery.

DAGMA [1]. DAGMA is a differentiable causal discovery algorithm that enforces acyclicity through a log-determinant constraint, $\log\det(I - W \circ W) = 0$, where W is the weighted adjacency matrix and $\circ$ denotes the Hadamard product. This constraint ensures $(I - W)$ remains an M-matrix, which inherently guarantees acyclicity while avoiding the numerical instability of matrix exponentials used in NOTEARS. DAGMA minimizes a loss combining least-squares error and L1 regularization via augmented Lagrangian methods, and enables scalable optimization in high-dimensional settings. DAGMA eliminates explicit cycle checks by operating directly on M-matrices, which offers faster convergence and theoretical guarantees. However, its performance in high-varsortability regimes remains understudied in the original work [1].

SparseRC [8]. SparseRC is a combinatorial optimization-based algorithm that learns sparse linear causal structures through a two-phase process. First, estimating a weighted adjacency matrix W via L1-regularized least-squares regression $\min_W \|X - XW\|_F^2 + \lambda\|W\|_1$. Second, greedily selecting edges sorted by absolute weight magnitude while enforcing acyclicity. Unlike continuous methods, SparseRC decouples structure learning from cycle avoidance by first identifying candidate edges and then pruning them into a DAG. This approach risks amplifying scale-related properties as the initial regression phase may prioritize edges with weights correlated with marginal variances. The final DAG is constructed by iteratively adding edges (sorted by $|W_{ij}|$) to an initially empty graph and skipping any that introduce cycles. While computationally efficient, SparseRC's post-hoc acyclicity

enforcement makes it uniquely vulnerable to varsortability-driven biases, as variance patterns may disproportionately influence both the regression weights and the edge selection order. This is different than methods like DAGMA, which integrate acyclicity constraints directly into the optimization loop.

GOLEM [4]. GOLEM (Gradient-based Optimization for Learning Effective DAGs) is a continuous, optimization-based algorithm that formulates structure learning as a differentiable optimization problem. The main idea is to represent the DAG as a weighted adjacency matrix W and minimize a loss function that combines data fidelity (typically a Gaussian negative log-likelihood), sparsity regularization, and a differentiable acyclicity constraint, often using the log-determinant formulation. By relaxing the discrete acyclicity constraint into a smooth surrogate, GOLEM enables efficient gradient-based optimization and scales well to high-dimensional settings. Unlike combinatorial or constraint-based methods, GOLEM's framework allows for flexible integration of prior knowledge and is robust to various noise models.

GES [2]. GES is a score-based causal discovery algorithm that searches over Markov equivalence classes of DAGs to optimize a predefined score (e.g., BIC). It operates in two phases - a forward phase that greedily adds edges to maximize the score, followed by a backward phase that removes unnecessary edges. GES efficiently navigates the space of possible DAGs while maintaining acyclicity. Unlike constraint-based methods like PC, GES explicitly optimizes a global score function that balances model fit and complexity. Though computationally expensive for moderate-sized graphs, GES can converge to local optima and assumes the score function reliably reflects causal structure.

2.4 Algorithmic Paradigms in Causal Discovery

Causal structure learning algorithms can be broadly categorized based on how they formulate the search for a DAG. In this work, I primarily reference four conceptual groupings: continuous, combinatorial, optimization-based, and score-based. These categories are not mutually exclusive. Instead, they refer to different dimensions of algorithm design, and individual methods may belong to multiple categories. This framework helps in understanding how each algorithm approaches DAG discovery and provides context for analyzing their susceptibility to data clues like varsortability in the subsequent experiments. Table 1 summarizes the categories into which the different algorithms studied in this project fall.

Continuous vs. Combinatorial: This distinction refers to the nature of the search space and the operations used to explore it. Continuous algorithms formulate DAG discovery as a differentiable optimization problem. These methods typically relax the discrete DAG constraint into a smooth, continuous surrogate that enables the use of gradient-based solvers. Examples include NOTEARS and DAGMA. On the other hand, combinatorial algorithms operate in a discrete search space and

manipulate graphs via additions, deletions, or reversals of edges, often guided by statistical tests or scoring heuristics. Examples include PC, GES, and SparseRC.

Optimization-based vs. Score-based: This distinction describes the objective used during graph construction. Optimization-based methods explicitly define a loss function that quantifies fit to data and optimize it subject to a DAG constraint. These losses often include regression error and regularization terms, and the DAG constraint is imposed either during or after optimization. Continuous algorithms like NOTEARS and DAGMA, as well as combinatorial algorithms like SparseRC, are typically optimization-based. In contrast, score-based methods evaluate candidate graphs using predefined scores (e.g., BIC, AIC) and search for the best graph through combinatorial procedures like greedy search. GES is a typical example.

Algorithm	Continuous	Combinatorial	Optimization-based	Score/Constraint-based
NOTEARS	✓		✓	
DAGMA	✓		✓	
SparseRC		✓	✓	
GES		✓	✓	✓(Score)
PC		✓		✓(Constraint)
GOLEM-NV	✓		✓	

Table 1: Categorization of causal discovery algorithms by design paradigm

3 Methodology and Experimental Design

Both phases of the experimental framework have different methodological objectives and pipelines. Phase 1 aimed to replicate the findings of Reisach et al. (2021) to investigate the role of marginal variance in traditional causal discovery. Phase 2 expanded the evaluation to newer algorithms and more comprehensive testing under both raw and standardized conditions.

3.1 Phase 1: Replication of Varsortability and Initial Benchmarks

To replicate the empirical results from Reisach et al. (2021)'s work, the first step was to reproduce the conditions under which synthetic datasets exhibit varsortability.

Data Generation: The data consists primarily of generated synthetic datasets based on DAG models and covers a variety of conditions under which the causal discovery algorithms are tested. The data generation process employs two graph types, Erdös-Rényi (ER-k) and Scale-Free (SF-k), which exhibit distinct structural properties. Each graph class is parameterized by a notation k, which indicates the average density of the graph. In an ER graph, each edge is included with a fixed probability independent of the others. The parameter k in ER-k denotes the average number of edges per node. For example, an ER-2 graph with d nodes will have, on average, 2d edges distributed randomly. In contrast to ER graphs, SF graphs follow a preferential attachment mechanism, where

nodes with higher degrees are more likely to receive additional connections. This results in a graph with a few highly connected "hub" nodes and many nodes with relatively few connections. The parameter k in SF-k specifies the density, with k influencing the degree of preferential attachment. SF-2 and ER-2 graphs were generated to explore differences in sparsity and density.

Three graph sizes d = {10, 30, 50} and three noise types (Gaussian, Exponential, Gumbel) are combined to simulate diverse scenarios. The edges in these graphs were assigned weights sampled independently from a bimodal uniform distribution, Unif$((2, 0.5)(0.5, 2))$. Each dataset includes both raw and standardized versions where the raw data preserves the original scale of variances, leading to high varsortability, whereas the standardized data neutralizes marginal variances by rescaling all variables to unit variance, reducing varsortability to v = 0.5. A total of 180 DAGs are generated. Table 2 summarizes the combinations of properties to generate the DAGs.

Property	**Description**
Graph Types	ER-2, SF-2
Node Counts (d)	10, 30, 50 nodes
Edge Weights	Sampled from Unif$((-2, -0.5) \cup (0.5, 2))$
Noise Types	Gaussian, Exponential, Gumbel
Samples per Graph (n)	1000
Data Versions	Raw and Standardized (zero mean, unit variance)
Repetitions	10 per parameter combination
Total Graphs	90 graphs
Total Datasets	180 datasets (raw and standardized for each graph)

Table 2: Configuration of synthetic dataset generation used across both phases

Varsortability Calculation: Following Reisach et al. (2021), I computed a varsortability score for each DAG to quantify the alignment between marginal variances and the topological order of the DAG. This metric provides a principled way to evaluate how closely a dataset exhibits the conditions under which marginal variance can act as a proxy for causal direction.

Given a data matrix $X \in \mathbb{R}^{n \times d}$ and a DAG with adjacency matrix $W \in \mathbb{R}^{d \times d}$, let E be the binary edge matrix where $E_{ij} = 1$ if there is a directed edge from node i to node j. Let $\mathrm{var}(X_i)$ denote the sample variance of variable i.

Varsortability is computed over all directed paths in the DAG. For each directed path from i to j, I check whether $\mathrm{var}(X_i) < \mathrm{var}(X_j)$. To account for numerical precision, a tolerance ϵ is used, and partial credit is given if variances are nearly equal.

$$\text{Varsortability} = \frac{1}{N_{\text{paths}}} \sum_{\text{all directed paths } i \rightsquigarrow j} \left[\mathbb{I}\left(\mathrm{var}(X_i) < \mathrm{var}(X_j) - \epsilon\right) + \frac{1}{2}\mathbb{I}\left(|\,\mathrm{var}(X_i) - \mathrm{var}(X_j)| \leq \epsilon\right) \right]$$

where N_{paths} is the total number of directed paths in the DAG, and $\mathbb{I}(\cdot)$ is the indicator function. The pseudocode implementation of the varsortability score is as follows:

```
----------------------------------------
Input: Data matrix X, Adjacency matrix W, tolerance tol
Output: Varsortability score in [0,1]
1. Compute binary edge matrix E = (W != 0)
2. Initialize path matrix Ek = E (paths of length 1)
3. Compute variances: var = variance(X_i) for each variable
4. Initialize path counters: total_paths = 0, correct_paths = 0
5. For path_length from 1 to d-1:
    a. total_paths += number of paths in Ek
    b. For each path i -> ... -> j in Ek:
        i. If var_i/var_j < 1 - tol:  # Equivalent to var_i < var_j
            correct_paths += 1
        ii. Else if |var_i/var_j - 1| <= tol:  # Nearly equal variances
            correct_paths += 0.5
    c. Update Ek = Ek * E (matrix multiplication for longer paths)
6. Return score = correct_paths / total_paths
----------------------------------------
```

A high varsortability score (close to 1) means that, for most directed paths in the DAG, the variance increases along the causal direction, indicating strong alignment between marginal variance and causal order. A low varsortability score (near 0.5) shows little to no correlation between variance and causal structure, so marginal variances provide little information about the true DAG. This metric plays a central role in interpreting why certain algorithms may perform well under raw data conditions, especially when their mechanisms implicitly or explicitly leverage marginal variances.

Sortnregress Baseline: As a preliminary point of comparison, I included a simple heuristic method proposed by Reisach et al. (2021), known as the sort-and-regress (Sortnregress) algorithm. This baseline is included as this method provides insight into how much of the graph structure can be recovered using only marginal variances, without any model-based assumptions. It simply works by sorting variables in ascending order of their marginal variances and regressing each variable on its predecessors in the sorted list. The intuition is that, under strong varsortability, this naive approach may recover a substantial portion of the true causal structure purely by exploiting variance ordering. The Sortnregress algorithm logic is presented in the following pseudocode.

```
----------------------------------------
```

```
Input: Data matrix X (n_samples × d variables)
Output: Estimated adjacency matrix W (d × d)
1. Compute marginal variances of each variable in X
2. Sort variable indices by increasing variance → order = [v_1, v_2, ..., v_d]
3. Initialize W as zero matrix
4. For k in 1 to d-1:
    a. Target variable = order[k]
    b. Covariates = order[0:k]  # Variables with lower variance
    c. Fit OLS regression: target ~ covariates → get coefficients
    d. Compute weights = absolute OLS coefficients
    e. Fit weighted Lasso regression using BIC for sparsity:
       - Features: covariates * weights
       - Target: original target
    f. Store selected coefficients in W[covariates, target]
5. Return W
----------------------------------------
```

In Phase 1, I evaluated five causal discovery algorithms, **NOTEARS, PC, GES, GOLEM, and the Sortnregress baseline**, covering both continuous and combinatorial categories. These algorithms were assessed using SHD and SID metrics, to be consistent with the evaluation framework of Reisach et al. (2021).

3.2 Phase 2: Evaluation of Post-2021 Algorithms and Robustness Analysis

Building on the insights from Phase 1, Phase 2 was designed to evaluate two recently proposed algorithms, DAGMA and SparseRC, using the same dataset of synthetic DAGs. These algorithms were selected as the primary focus of evaluation due to their novel formulations and relevance to ongoing developments in continuous, optimization-based causal discovery. To contextualize their performance, three baseline methods from Phase 1 were included - NOTEARS (a continuous, optimization-based algorithm), PC (a combinatorial, constraint-based algorithm), and Sortnregress (a naive variance-driven heuristic). The primary objective was to assess algorithm robustness on raw versus standardized data, with attention to which algorithmic paradigms are more susceptible to changes in marginal variance.

Codebase Setup: A major part of the work in Phase 2 involved integrating codebases from various authors into a unified experimental pipeline. All datasets from Phase 1 were stored in a consistent format with fields for graph structure, raw data, standardized data, and metadata such as node count and noise type.

Each run was conducted using a fixed random seed (`seed=1`) to ensure full reproducibility across experimental repetitions. Simulations, training, and evaluations were executed using the UMass Unity computing cluster.

For standardized datasets with low varsortability at the 0.5 mark, slight jitter was introduced to the plotted varsortability values. This visual adjustment helped prevent overlapping points in scatter plots and facilitated more precise visualization of performance trends.

The specific implementation details per method are as follows:

- **PC and GES** were implemented using the `causallearn` Python library. `https://causal-learn.readthedocs.io/en/latest/`
- **GOLEM** was implemented using the authors' original code. `https://github.com/ignavierng/golem`
- **DAGMA** was run using the official PyTorch implementation provided by the authors. [`https://dagma.readthedocs.io/en/latest/`]
- **SparseRC** was integrated from its original JAX codebase. `https://github.com/pmisiakos/SparseRC`
- **NOTEARS** was implemented using the authors' original code. `https://github.com/xunzheng/notears`
- **Sortnregress** was adapted from the original codebase provided by Reisach et al. (2021). `https://github.com/Scriddie/Varsortability`

Evaluation: For each method and dataset, I computed the SHD, SID, and BSF scores. Each algorithm was evaluated on both the raw and standardized versions of each dataset. Results show clear differences in sensitivity to data standardization, which are discussed in the next section.

4 Results

4.1 Phase 1

The first step in Phase 1 was to quantify the extent of varsortability in the synthetic datasets. Table 3 reports the minimum, mean, and maximum varsortability scores across all datasets, grouped by graph type and noise distribution.

Across all combinations, raw datasets consistently exhibit high varsortability scores, with means ranging from 0.95 to 1.00. This indicates strong alignment between marginal variance and the true causal order, and that in the raw form, the data carries a statistical pattern that may serve as a shortcut

for structure learning algorithms. Whereas, standardized datasets yield uniform varsortability scores of 0.5, as expected, with removed variance differences across variables and eliminating this alignment.

These results confirm that synthetic data generated using typical practices can encode spurious cues via marginal variance, which may mislead certain algorithms into appearing more accurate than they are.

Graph	Noise	Min (raw)	Min (std)	Mean (raw)	Mean (std)	Max (raw)	Max (std)
ER-2	Exponential	0.95	0.5	0.99	0.5	1.00	0.5
ER-2	Gaussian	0.87	0.5	0.97	0.5	1.00	0.5
ER-2	Gumbel	0.90	0.5	0.98	0.5	1.00	0.5
SF-2	Exponential	0.86	0.5	0.99	0.5	1.00	0.5
SF-2	Gaussian	0.91	0.5	0.99	0.5	1.00	0.5
SF-2	Gumbel	0.80	0.5	0.98	0.5	1.00	0.5

Table 3: **Varsortability Scores for Raw and Standardized Data across Graph and Noise Types.** Average varsortability is consistently high in raw data, which supports the hypothesis that synthetic benchmarks often encode causal order through marginal variance. Standardized data exhibits uniform scores of 0.5, indicating low alignment.

With the varsortability pattern established, the next step was to evaluate how different structure learning algorithms perform on raw versus standardized datasets. Figure 1 and 2 show boxplots for SHD and SID across the five algorithms.

A clear pattern emerges when results are grouped by algorithm type. Combinatorial methods like PC and GES show consistent performance between raw and standardized datasets. Their SHD and SID scores remain stable, indicating that they do not rely on marginal variance cues and are robust to changes in data scale. These methods use conditional independence tests or score-based heuristics that are inherently scale-invariant.

In contrast, continuous optimization-based algorithms such as NOTEARS and GOLEM show a marked drop in performance after standardization. On raw data, both methods exhibit low SHD and SID, but these metrics worsen substantially when applied to standardized data. This suggests that their apparent success in the raw setting may come from exploiting the marginal variance pattern rather than learning true causal structure. The Sortnregress baseline follows an even more extreme version of this trend. Its performance is strong on raw data but collapses after standardization, which is consistent with its explicit dependence on variance ordering.

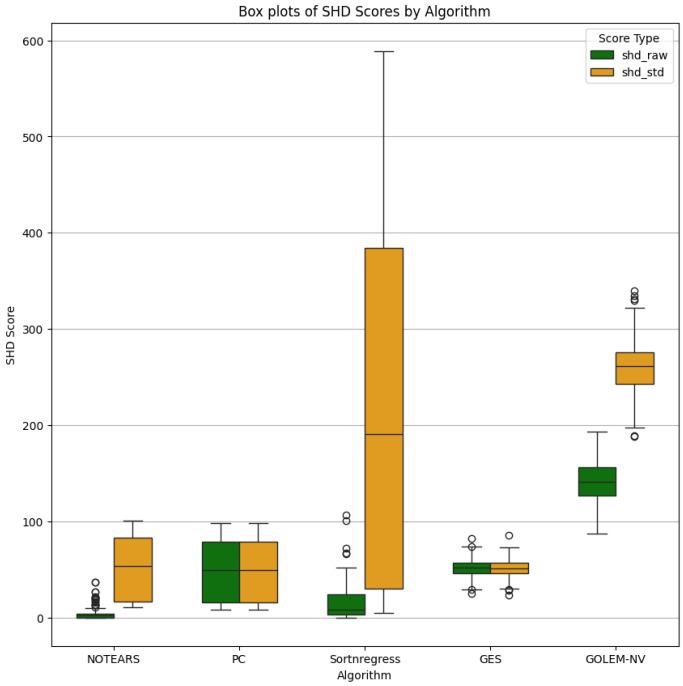

Figure 1: **SHD across raw and standardized synthetic datasets for Phase 1.** Continuous methods such as NOTEARS and GOLEM show a steep increase in SHD when the data are standardized, while combinatorial methods like PC and GES remain robust. The Sortnregress baseline performs well only when marginal variances align with the causal ordering.

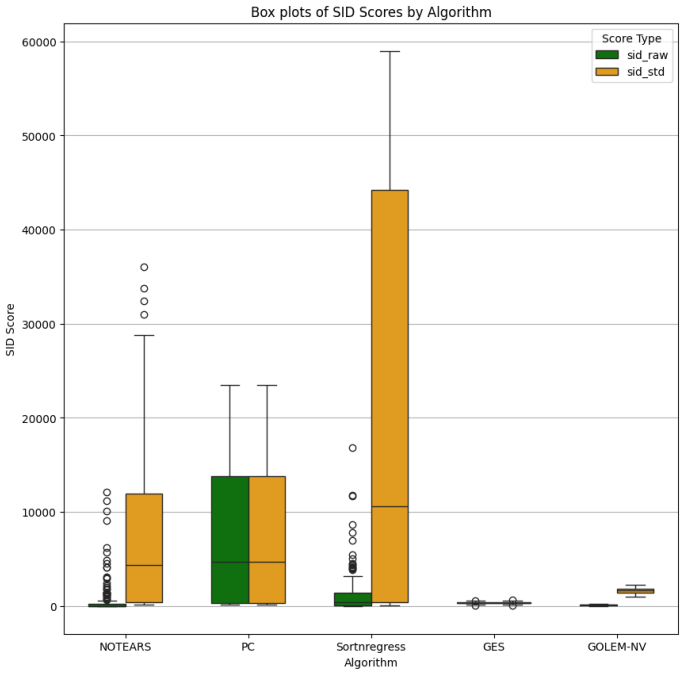

Figure 2: **SID across raw and standardized synthetic datasets for Phase 1.** Continuous approaches like NOTEARS, GOLEM exhibit an increase in SID under standardization, whereas combinatorial ones like PC and GES stay relatively stable. The Sortnregress baseline achieves low SID only when the marginal variances reflect the true causal ordering.

Phase 1 confirms Reisach et al.'s (2021) finding that algorithms exploit synthetic datasets' marginal-variance patterns as shortcuts. Continuous optimization methods excel when these patterns exist but fail after standardization, suggesting they fit variance structures, not causal graphs. Combinatorial methods remain stable across scaling and demonstrate robustness to superficial cues via conditional independence or score-based criteria.

4.2 Phase 2

4.2.1 Metric-by-Metric Analysis with Respect to Varsortability

In Phase 2, I extended the analysis to newer continuous optimization-based algorithms, DAGMA and SparseRC, alongside three baselines from Phase 1 - NOTEARS, PC, and Sortnregress. I evaluated the performance using SHD, SID, and an additional BSF score.

SHD measures the total number of edge insertions, deletions, or flips required to transform an estimated graph into the true DAG. A perfect reconstruction scores 0, and higher values indicate increasing structural inaccuracy.

According to the results presented in Section A.1.1, in the raw-data SHD vs. varsortability plots, each continuous optimization method (NOTEARS, DAGMA, SparseRC) and the Sortnregress baseline show low scores in high varsortability scenarios around 0.90 to 1.00. These methods benefit directly from the marginal-variance signal. As that alignment strengthens, SHD drops from double-digit errors (e.g., 8–22 at low varsortability) to near zero, indicating perfect or near-perfect recovery. This pattern reflects how continuous methods exploit scale. Their smooth optimization landscapes, constrained by acyclicity, become easier to navigate when marginal variances correlate with causal depth, providing an implicit ordering cue.

PC, by contrast, exhibits a flat SHD profile across the same range. Its error remains consistent, about 12–20 on 10-node graphs, 45–55 on 30-node graphs, and 80–95 on 50-node graphs, regardless of varsortability. Because it relies solely on scale-invariant conditional independence tests, it cannot exploit marginal variance, and thus neither benefits from nor collapses without that shortcut.

After standardization compresses varsortability to a narrow band around 0.5 (with small horizontal jitter added to separate overlapping points), the contrast becomes starker. SHD for variance-sensitive methods rises sharply, thus neither benefiting from nor collapsing - approximately 14–24 errors on 10-node graphs, 50–60 on 30-node graphs, and upwards of 100 on 50-node graphs. The structural signal is lost, and without it, these algorithms essentially guess. PC's SHD remains unchanged, with all points closely tracking the identity diagonal in the raw-vs-standardized scatter, showing its robustness.

SID quantifies the number of interventional queries on which the estimated graph's predictions diverge from the true DAG. A score of 0 indicates perfect agreement in causal effect estimation. Higher scores reflect more interventional inaccuracies.

According to the results presented in Section A.1.2, the SID vs. varsortability plots mirror the SHD trends. For NOTEARS, DAGMA, SparseRC, and Sortnregress, SID drops significantly as varsortability increases toward 1.0. When the marginal-variance pattern is strong, these methods achieve near-zero SID, accurately predicting how interventions propagate through the graph. However, as varsortability decreases, SID rapidly inflates from around 80–100 incorrect interventions at low varsortability on 10-node graphs, to thousands on 30- and 50-node graphs. This sensitivity highlights the extent to which their performance depends on variance-driven edge orientation.

PC again shows no such dependence. Its SID remains flat across the raw varsortability range: a few hundred errors on 10-node graphs, a few thousand on 30-node graphs, and over ten thousand on 50-node graphs. It neither benefits from nor is harmed by the presence or absence of marginal variance alignment.

After standardization, SID scores for the variance-sensitive methods rise dramatically and plateau at high levels, with dozens to hundreds of incorrect interventions for 10-node graphs and thousands for larger graphs. No downward trend remains. Their SID scatter plots show points far above the y = x line, signaling significant degradation. In contrast, PC's SID points lie almost perfectly along the diagonal, showing that standardization has no measurable impact.

BSF evaluates both edge presence and absence by weighting each correct or incorrect decision by its rarity. Correctly identifying a true edge contributes +1/a, a true non-edge +1/i; false positives cost –1/i and missed edges cost –1/a (where a is the number of true edges and i the number of independencies). A perfect graph scores +1, a reversed graph –1, and a naïve empty or complete graph scores 0-making BSF a sensitive measure of both precision and recall.

According to the results presented in Section A.1.3, in the raw-data BSF vs. varsortability plots, NOTEARS, DAGMA, and SparseRC display sharp upward slopes. When varsortability is low (0.90–0.95), BSF scores begin in the 0–0.2 range; as it approaches 1.0, BSF climbs to 0.8–1.0. This reflects a dual benefit. These methods not only orient edges more accurately (as seen in SHD) but also avoid false positives by leveraging the variance pattern. Sortnregress shows a similar but noisier rise, with BSF improving from 0.1 to 0.7. PC, again unaffected by marginal variance, maintains a flat BSF profile around 0.2 across the varsortability spectrum.

Once the data is standardized and varsortability collapses to 0.5, all variance-sensitive methods suffer a breakdown. BSF scores fall sharply, often below zero for 30- and 50-node graphs, because both types of errors spike. True edges are missed, and spurious ones are added. The remaining variance

jitter contributes only noise. These methods no longer outperform even a trivial predictor. Their BSF points in the scatter plots lie well below the diagonal, indicating a collapse in overall structural quality. PC, as before, holds steady with BSF values between 0.15 and 0.25, tracking the diagonal exactly.

4.2.2 Distributional Behavior

To complement the metric-by-metric analysis, I examined the full distribution of scores using Kernel Density Estimates (KDEs) for SHD, SID, and BSF presented in Section A.2. These density plots provide a more complete picture of how each algorithm's performance varies and how those distributions change after standardization.

For the continuous optimization methods (NOTEARS, DAGMA, SparseRC) and the Sortnregress baseline, the distributions on raw data are sharply peaked near the ideal. SHD and SID densities are concentrated around low-error regions, and BSF densities approach +1. This indicates not only strong performance but also low variance across runs. These methods reliably perform well when the marginal variance pattern is present. The Sortnregress baseline, although simpler, shows a similar pattern in high-varsortability settings, further emphasizing the predictive power of marginal variance.

After standardization, the distributions for these same methods shift dramatically. SHD densities move from low-error regions to broader, higher-error peaks centered around 15–25 for 10 nodes, 50–60 for 30 nodes, and often above 80 for 50 nodes. SID densities stretch across hundreds or thousands of incorrect interventions, and BSF collapses toward 0 or even negative values. These shifts are not subtle. They reflect a fundamental collapse in edge recovery and interventional reliability once the global variance cue is removed. The increase in distributional spread also suggests instability. These methods not only fail more often post-standardization, but fail inconsistently across datasets.

PC, in contrast, exhibits remarkable distributional stability. Its raw and standardized KDE curves for all three metrics-SHD, SID, and BSF-overlap almost perfectly. SHD and SID distributions remain centered at moderate error values, while BSF hovers steadily around 0.2–0.3. This invariance demonstrates that PC's performance is governed by structural properties in the data rather than superficial patterns in scale. Its robustness holds across all graph sizes.

These KDE results reinforce the broader story. Continuous optimization methods can achieve excellent results, but only when the marginal variance pattern is intact. Once that pattern is neutralized through standardization, their performance distributions collapse toward chance. PC, by contrast, performs modestly but consistently and shows no dependence on whether or not the data contains a variance-based shortcut.

4.2.3 Aggregate Performance Summary

To consolidate the findings across all metrics and graph sizes, I present a high-level comparison of each algorithm's behavior on raw versus standardized data. These results, shown in Figures 3 (SHD), 4 (SID), 5 (BSF), and Tables 4 (SHD), 5 (SID), and 6 (BSF), illustrate how the presence or absence of the marginal variance signal shapes overall performance.

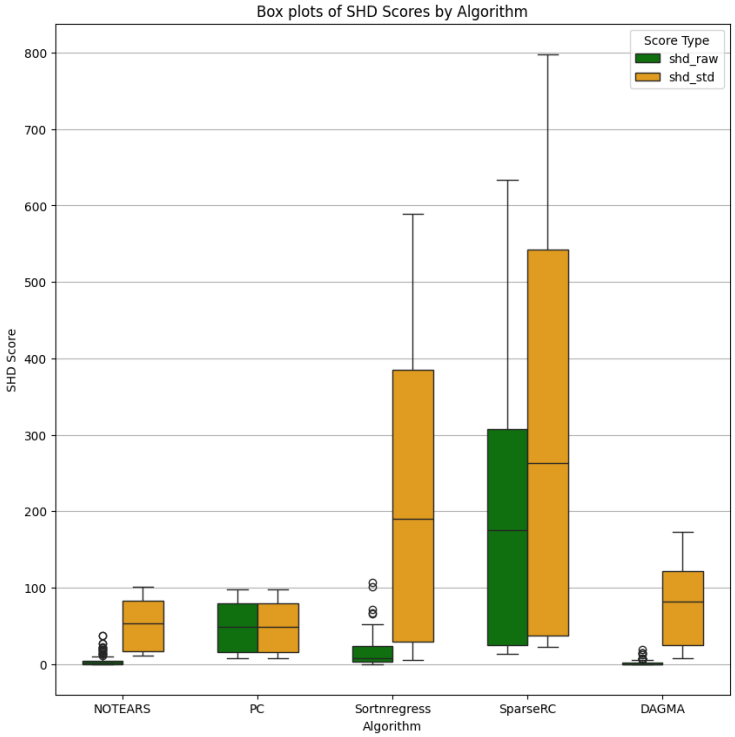

Figure 3: Aggregate SHD across raw and standardized data

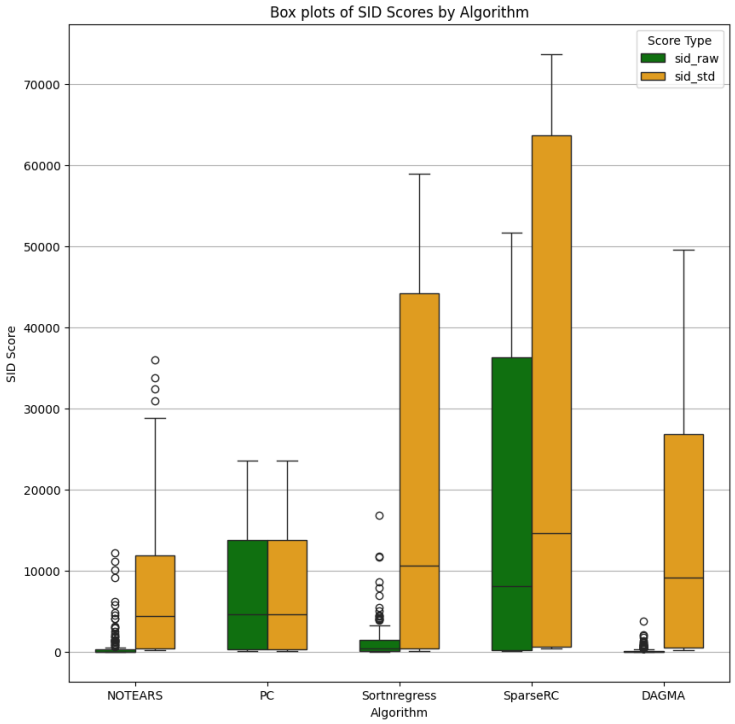

Figure 4: Aggregate SID across raw and standardized data

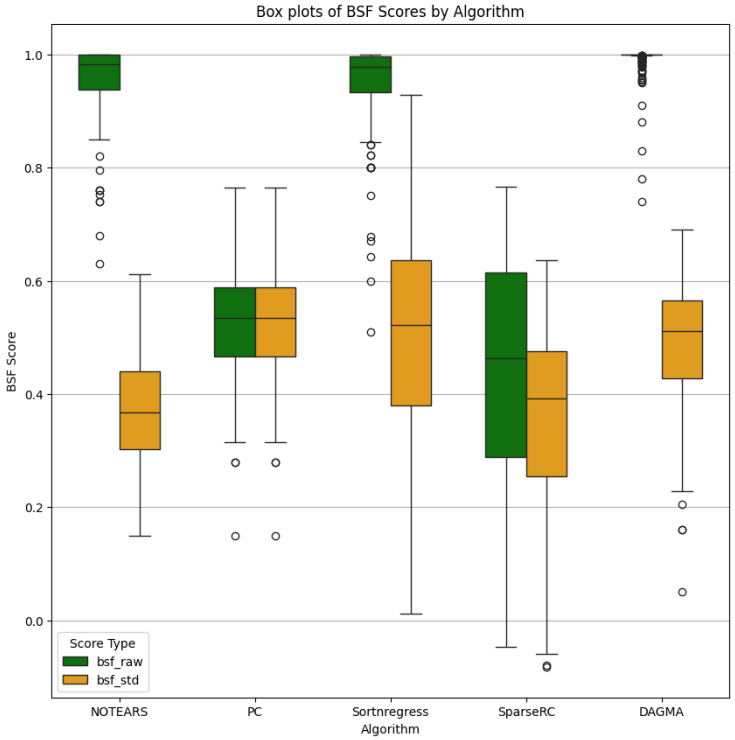

Figure 5: Aggregate BSF across raw and standardized data

Table 4: Aggregate SHD summary statistics for raw and standardized data

Algorithm	**SHD (Raw)**					**SHD (Standardized)**				
	Mean	Median	Std	Min	Max	Mean	Median	Std	Min	Max
DAGMA	1.47	0.0	3.01	0	19	79.85	82.0	48.44	8	173
NOTEARS	3.99	2.0	6.43	0	37	48.62	53.5	30.69	11	101
PC	48.17	49.0	28.32	8	98	48.17	49.0	28.32	8	98
Sortnregress	15.39	8.0	18.08	0	107	217.08	190.5	178.31	5	589
SparseRC	202.43	175.0	171.79	13	634	301.47	263.5	242.92	23	798

Table 5: Aggregate SID summary statistics for raw and standardized data

Algorithm	**SID (Raw)**					**SID (Standardized)**				
	Mean	Median	Std	Min	Max	Mean	Median	Std	Min	Max
DAGMA	170.59	0.0	435.06	0	3724	14441.46	9164.0	14904.75	153	49539
NOTEARS	642.18	9.0	1832.41	0	12152	6929.54	4379.0	8156.35	144	36015
PC	6884.27	4654.5	6722.95	117	23520	6884.27	4654.5	6722.95	117	23520
Sortnregress	1192.77	420.5	2194.54	0	16807	19639.06	10628.5	21027.18	63	58996
SparseRC	16875.44	8033.0	18778.07	54	51695	26886.79	14572.5	28088.91	360	73745

Table 6: Aggregate BSF summary statistics for raw and standardized data

Algorithm	**BSF (Raw)**					**BSF (Standardized)**				
	Mean	Median	Std	Min	Max	Mean	Median	Std	Min	Max
DAGMA	0.99	1.00	0.03	0.74	1.00	0.49	0.51	0.12	0.05	0.69
NOTEARS	0.95	0.98	0.07	0.63	1.00	0.37	0.37	0.09	0.15	0.61
PC	0.53	0.53	0.10	0.15	0.76	0.53	0.53	0.10	0.15	0.76
Sortnregress	0.95	0.98	0.08	0.51	1.00	0.50	0.52	0.18	0.01	0.93
SparseRC	0.44	0.46	0.20	-0.05	0.77	0.35	0.39	0.16	-0.08	0.64

Looking first at SHD, DAGMA and NOTEARS achieve near-zero median errors on raw data (0 and 2 edges, respectively), with extremely tight interquartile ranges-indicating highly reliable recovery when varsortability is high. However, after standardization, both methods exhibit severe degradation.

Median SHD jumps to 82 for DAGMA and 53.5 for NOTEARS, with upper whiskers reaching as high as 174 and 101 edges, respectively. This is a full order-of-magnitude increase in structural error. SparseRC and Sortnregress follow the same pattern: Sortnregress rises from a median of 8 errors to 190, and SparseRC from 175 to 264. In contrast, PC maintains a median SHD of 49 before and after standardization, with no significant shift in its error distribution.

SID reveals the same dependency. On raw data, the variance-sensitive methods show moderate SID values, often in the sub-50 or sub-500 range. But after standardization, SID spikes into the tens of thousands for DAGMA, NOTEARS, SparseRC, and Sortnregress. Their interventional predictions collapse without the variance cue. PC's SID, however, remains stable in the low-thousands across both conditions.

The BSF further highlights this divide. On raw data, DAGMA and NOTEARS reach near-perfect median BSF scores (0.99), Sortnregress follows closely (0.98), and SparseRC performs moderately well (0.46). After standardization, all four methods see steep declines: BSF drops to 0.51 for DAGMA, 0.37 for NOTEARS, 0.52 for Sortnregress, and 0.39 for SparseRC. Many runs score near or below zero, reflecting worse-than-naive performance. PC, by contrast, holds steady at 0.53 in both raw and standardized settings, showing no loss in its ability to balance edge detection and rejection.

5 Conclusion and Future Work

This study demonstrates that the strong performance of modern continuous optimization-based causal discovery algorithms on synthetic benchmarks is largely driven by their exploitation of a marginal-variance "shortcut." Across the three complementary metrics, SHD, SID, BSF, I observed that methods such as NOTEARS, DAGMA, SparseRC, and even the simple Sortnregress baseline achieve near-perfect graph recovery when the underlying DAGs exhibit high varsortability. However, their performance collapses once this variance pattern is neutralized through standardization. The drop is systematic, substantial, and consistent across graph sizes and evaluation metrics.

In contrast, the classic PC and GES algorithms, which rely on scale-invariant conditional independence tests, maintain moderate, stable performance regardless of the presence or absence of a variance signal. Aggregate box plots and kernel density estimates further support this finding. Only PC and GES show consistent error distributions under both raw and standardized conditions. Continuous methods, by comparison, revert to near-chance performance when deprived of the implicit orientation cues encoded in marginal variances. These results highlight a key limitation in current synthetic benchmarking practices - high accuracy on raw data may reflect alignment with superficial statistical artifacts, rather than genuine causal structure learning.

Looking ahead, I plan to expand this evaluation framework by incorporating several post-2021 algorithms, including the Truncated Matrix Power Iteration method [15] introduced in 2022 and the bivariate-constraint gradient learner method [6] introduced in 2023. These methods will be tested using the same synthetic pipeline to assess whether they, too, are sensitive to varsortability.

Beyond synthetic data, I will extend the evaluation to real-world causal inference tasks using datasets from the UMass Amherst Knowledge Discovery Lab's benchmark suite. Unlike synthetic graphs, these datasets often contain meaningful variance patterns that reflect true measurement scales rather than design artifacts. Testing algorithms under these conditions will help determine whether their reliance on marginal variance persists outside controlled simulations.

Through this broader evaluation, the goal is to advance causal discovery benchmarking practices toward standards that measure structural learning ability, not just sensitivity to data preprocessing. By identifying and controlling for statistical shortcuts like varsortability, I can better assess which algorithms truly uncover causal structure and which merely exploit the surface patterns of synthetic data.

A Appendix: Phase 2 Results Visualizations

A.1 Metric-by-Metric Plots

A.1.1 Structural Hamming Distance

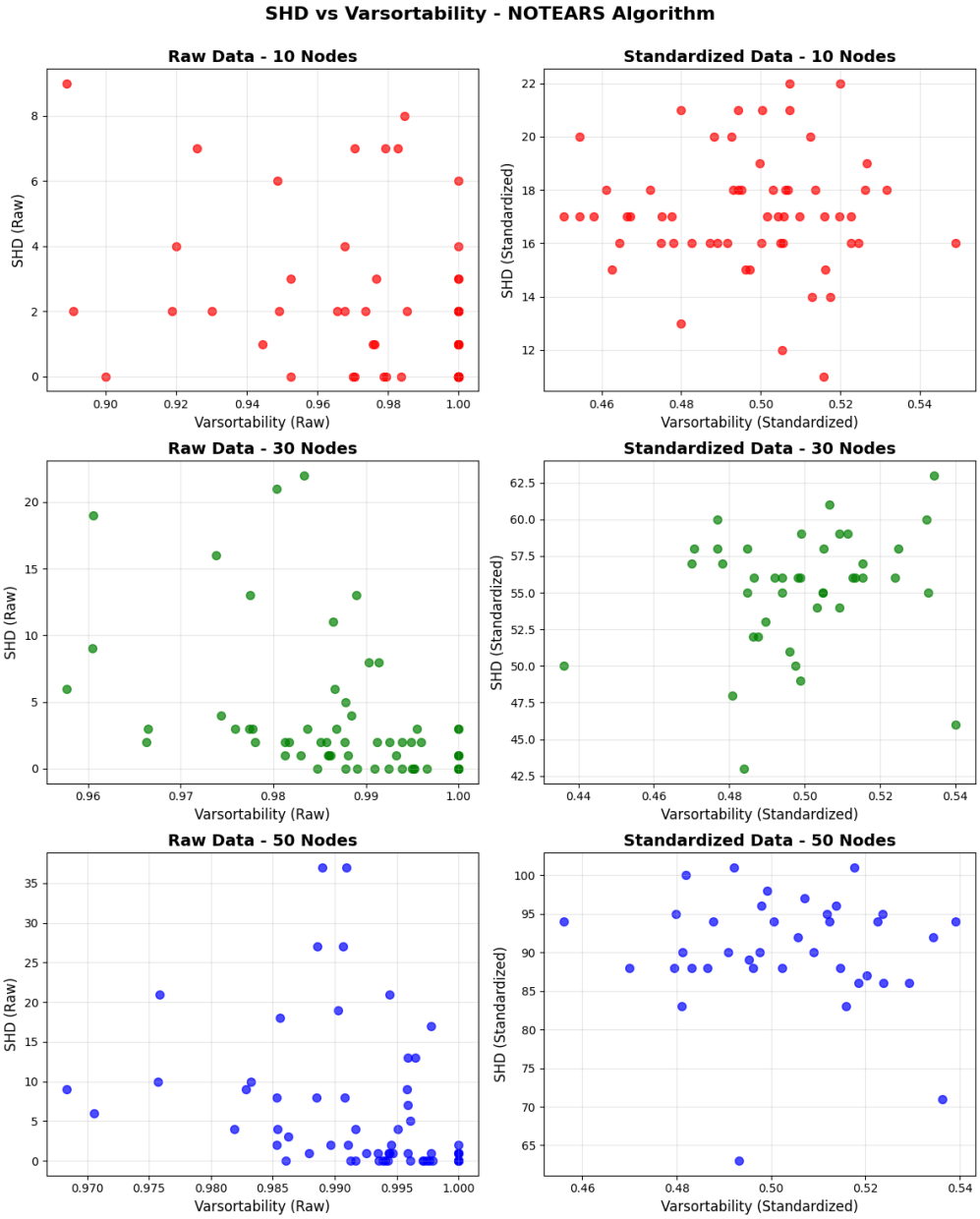

Figure 6: SHD vs. varsortability for NOTEARS

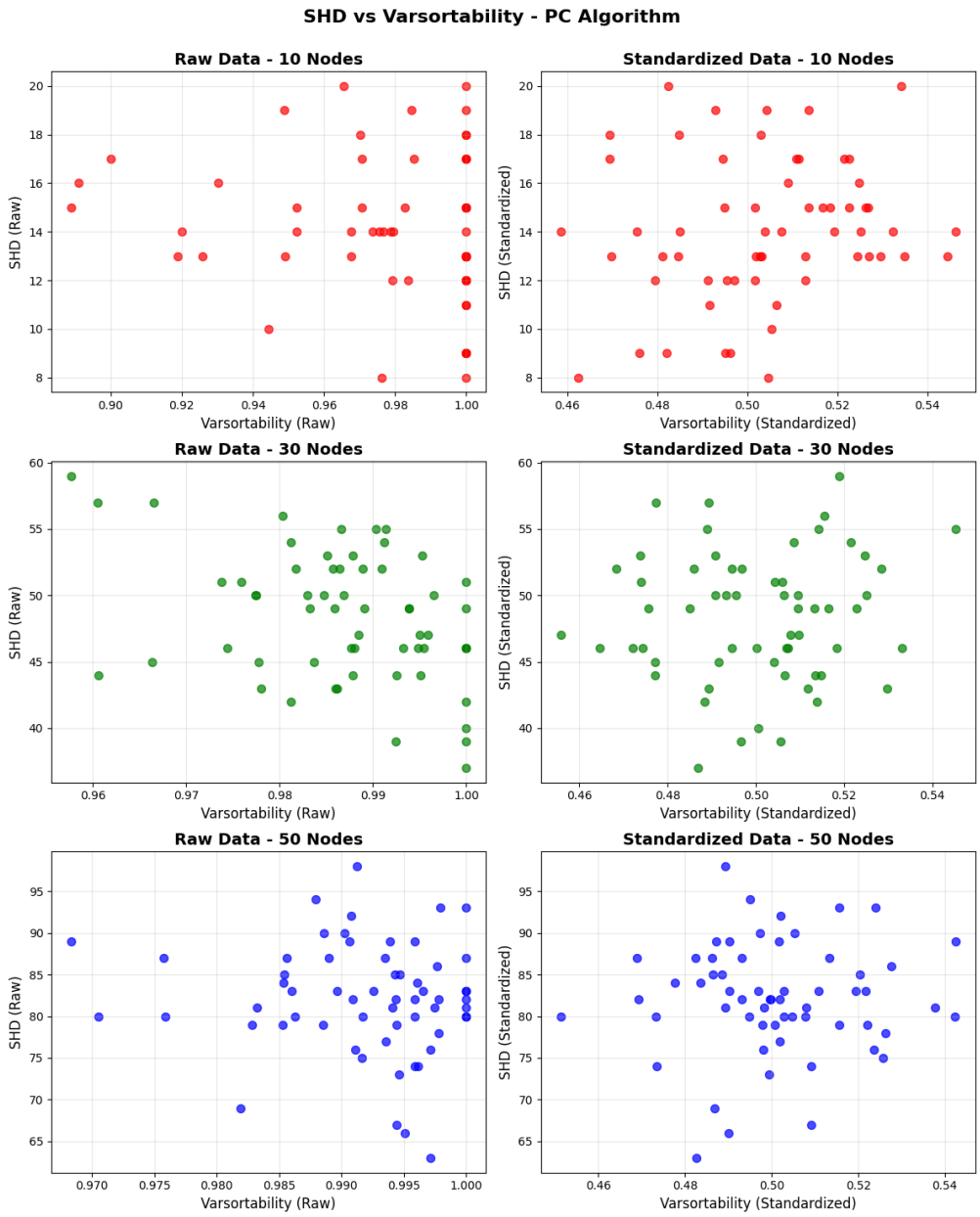

Figure 7: SHD vs. varsortability for DAGMA

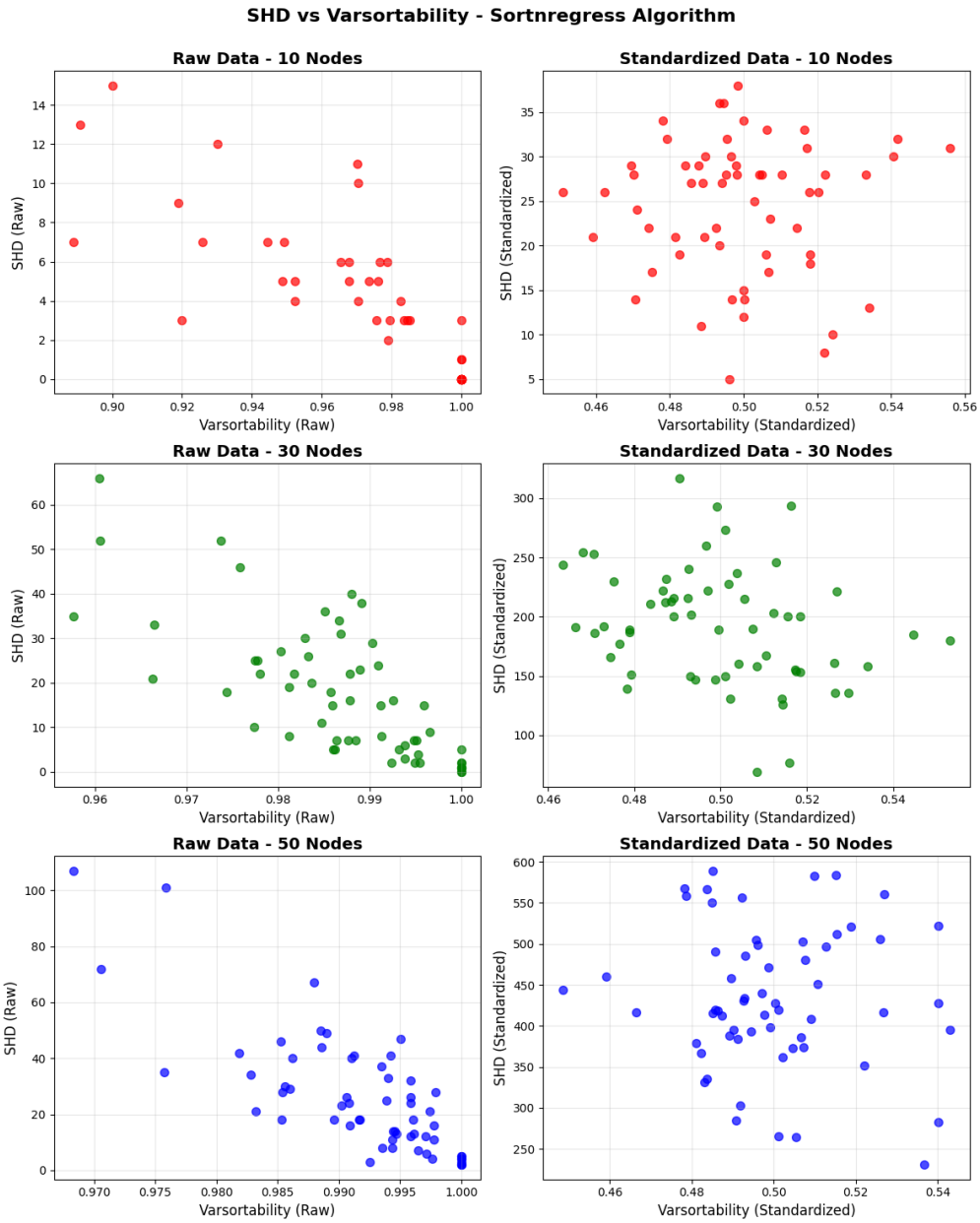

Figure 8: SHD vs. varsortability for SparseRC

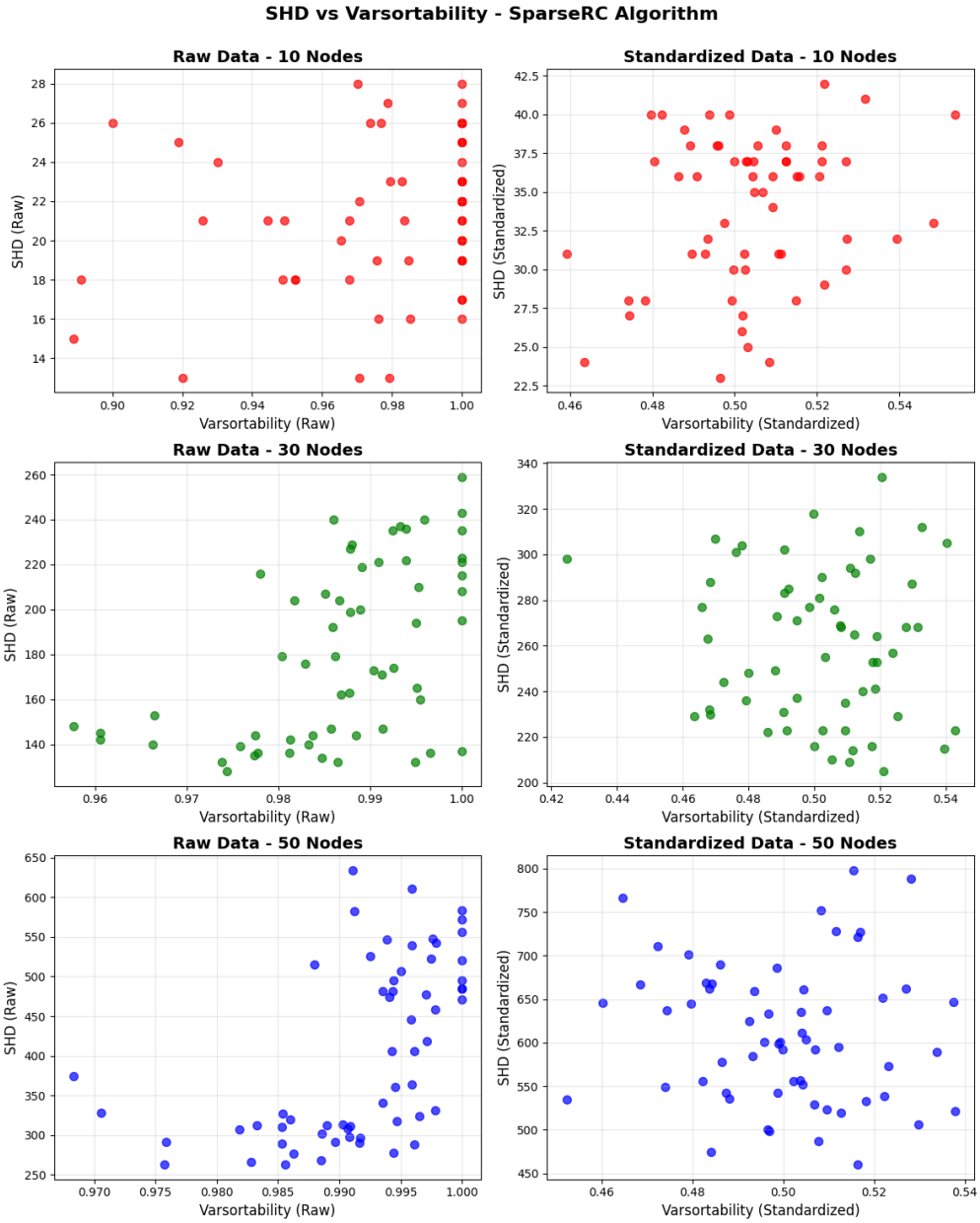

Figure 9: SHD vs. varsortability for PC

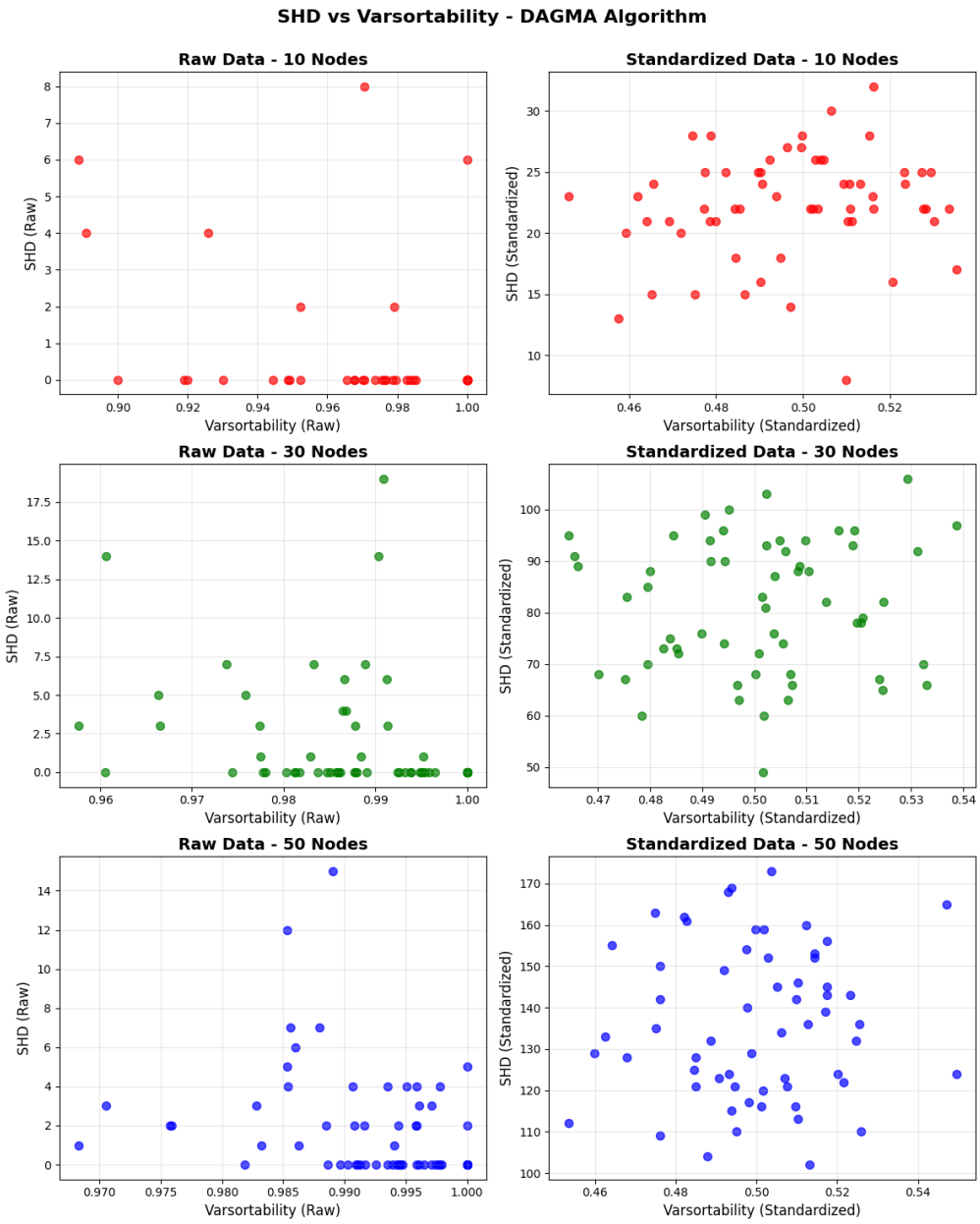

Figure 10: SHD vs. varsortability for Sortnregress

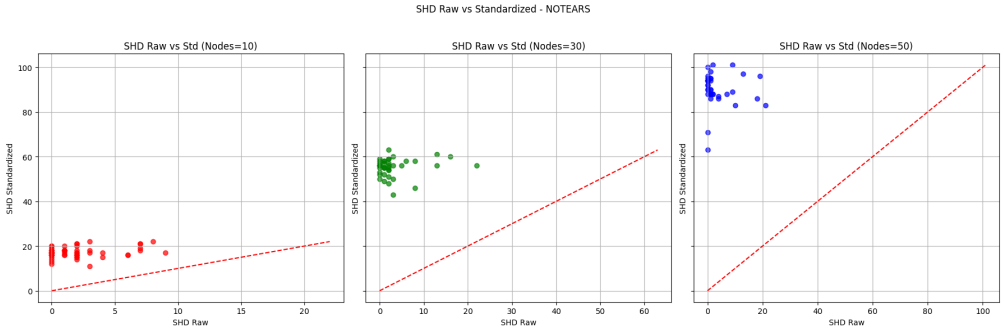

Figure 11: Standardized vs. raw SHD for NOTEARS

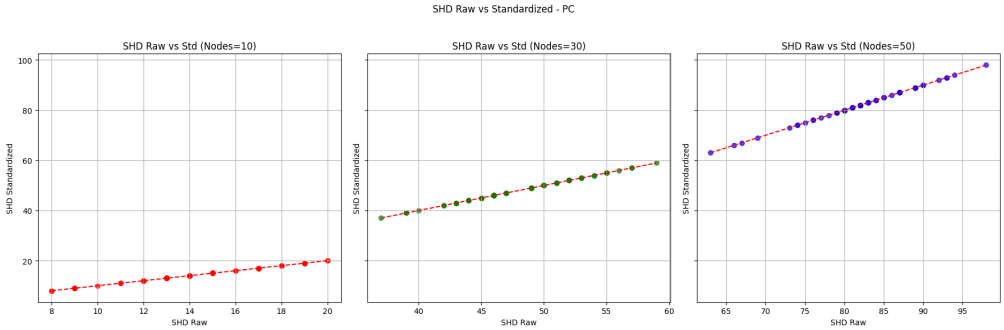

Figure 12: Standardized vs. raw SHD for DAGMA

Figure 13: Standardized vs. raw SHD for SparseRC

Figure 14: Standardized vs. raw SHD for PC

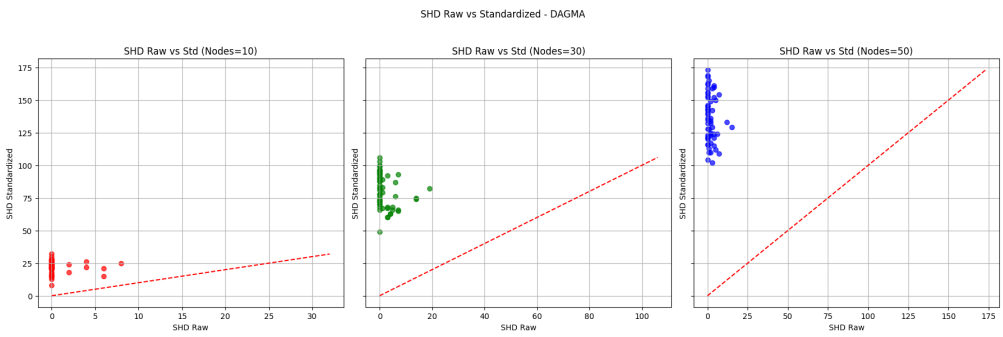

Figure 15: Standardized vs. raw SHD for Sortnregress

A.1.2 Structural Intervention Distance

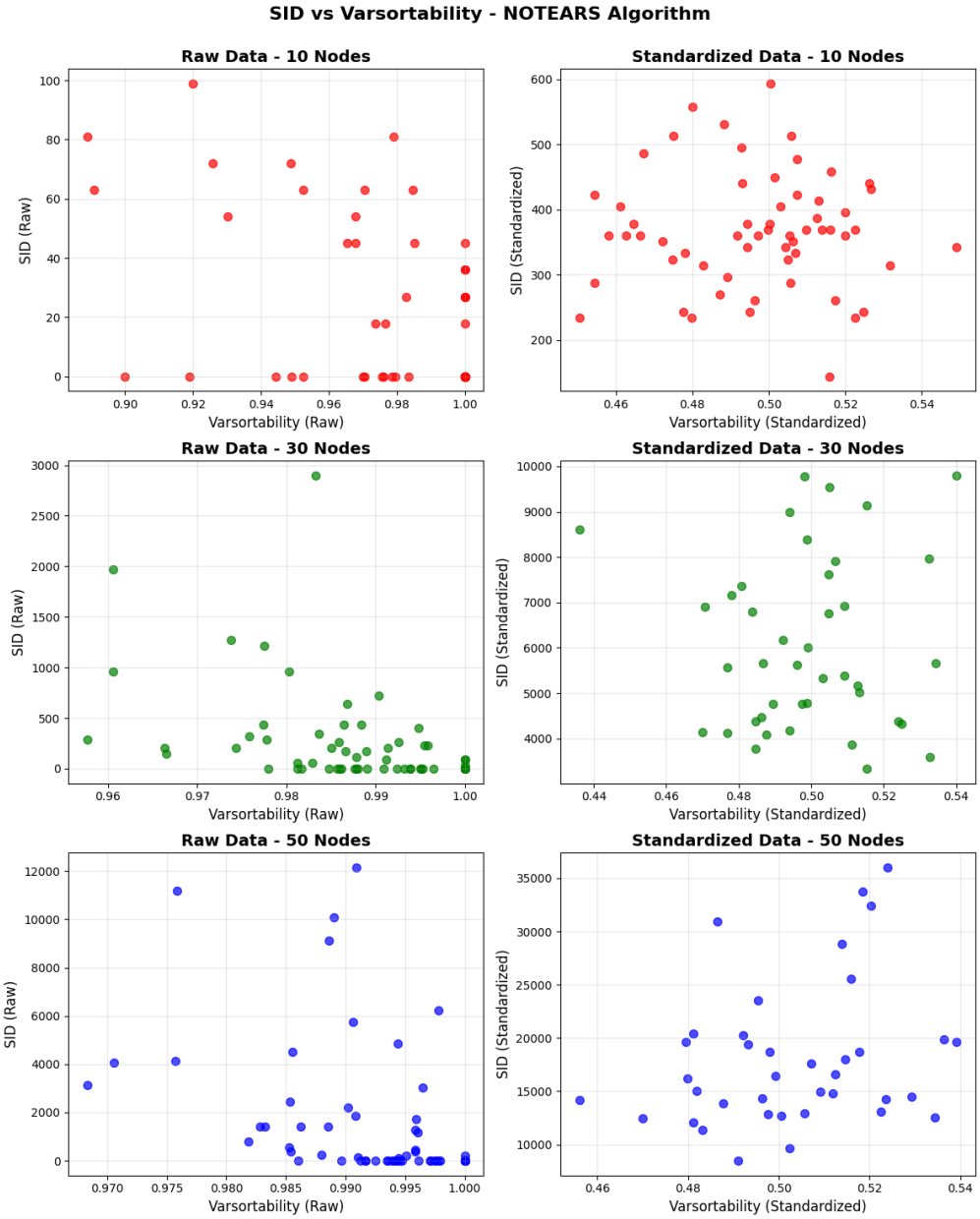

Figure 16: SID vs. varsortability for NOTEARS

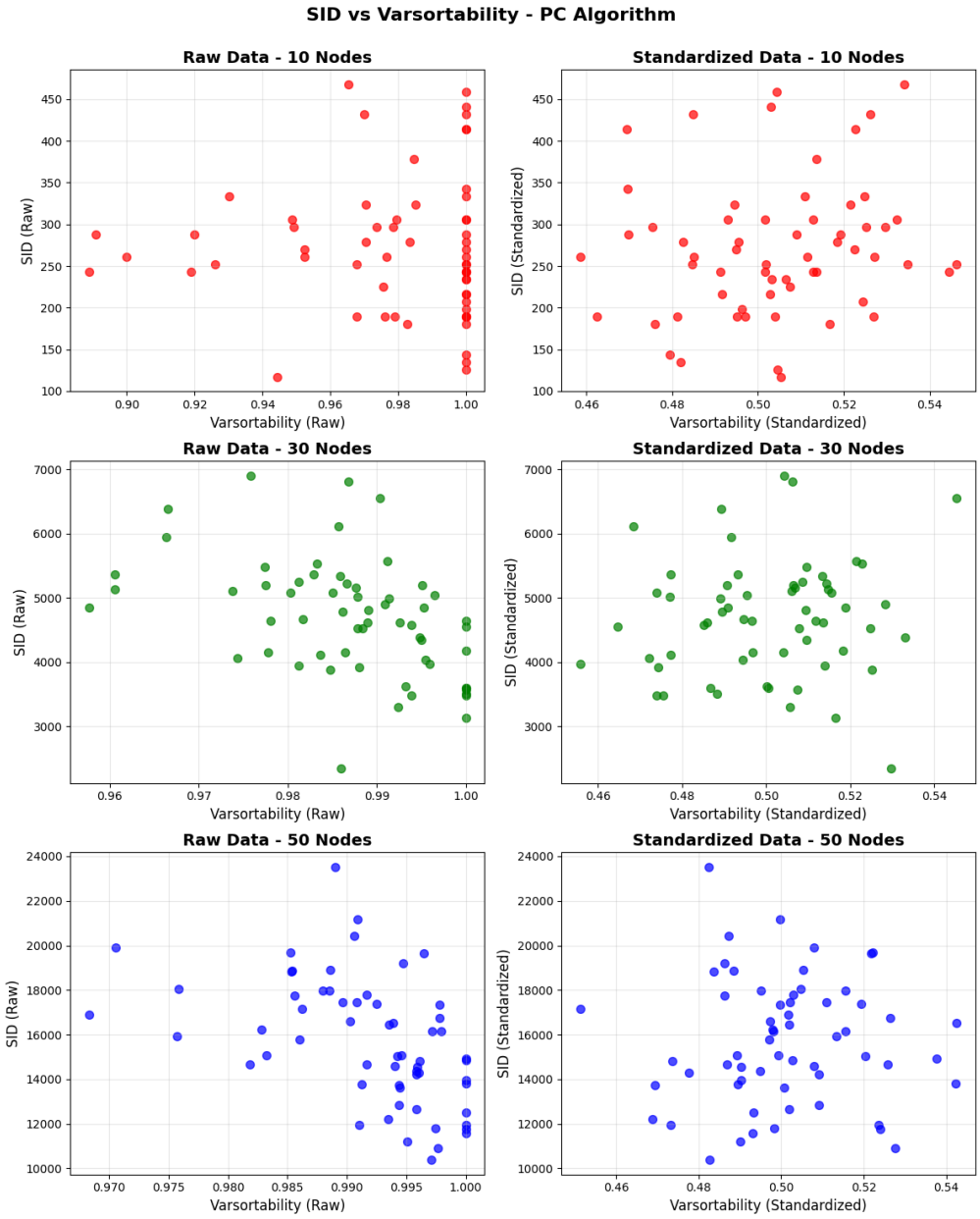

Figure 17: SID vs. varsortability for DAGMA

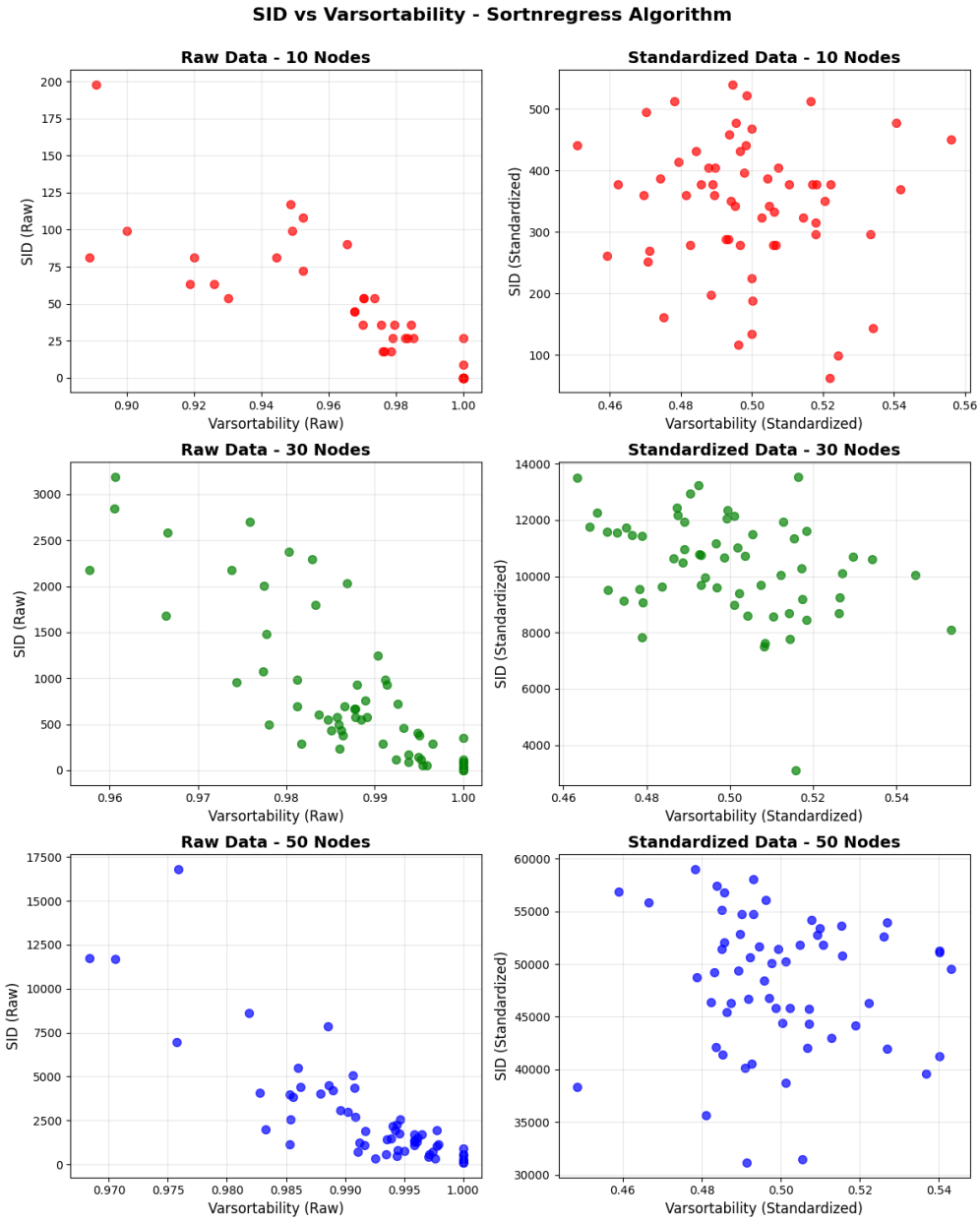

Figure 18: SID vs. varsortability for SparseRC

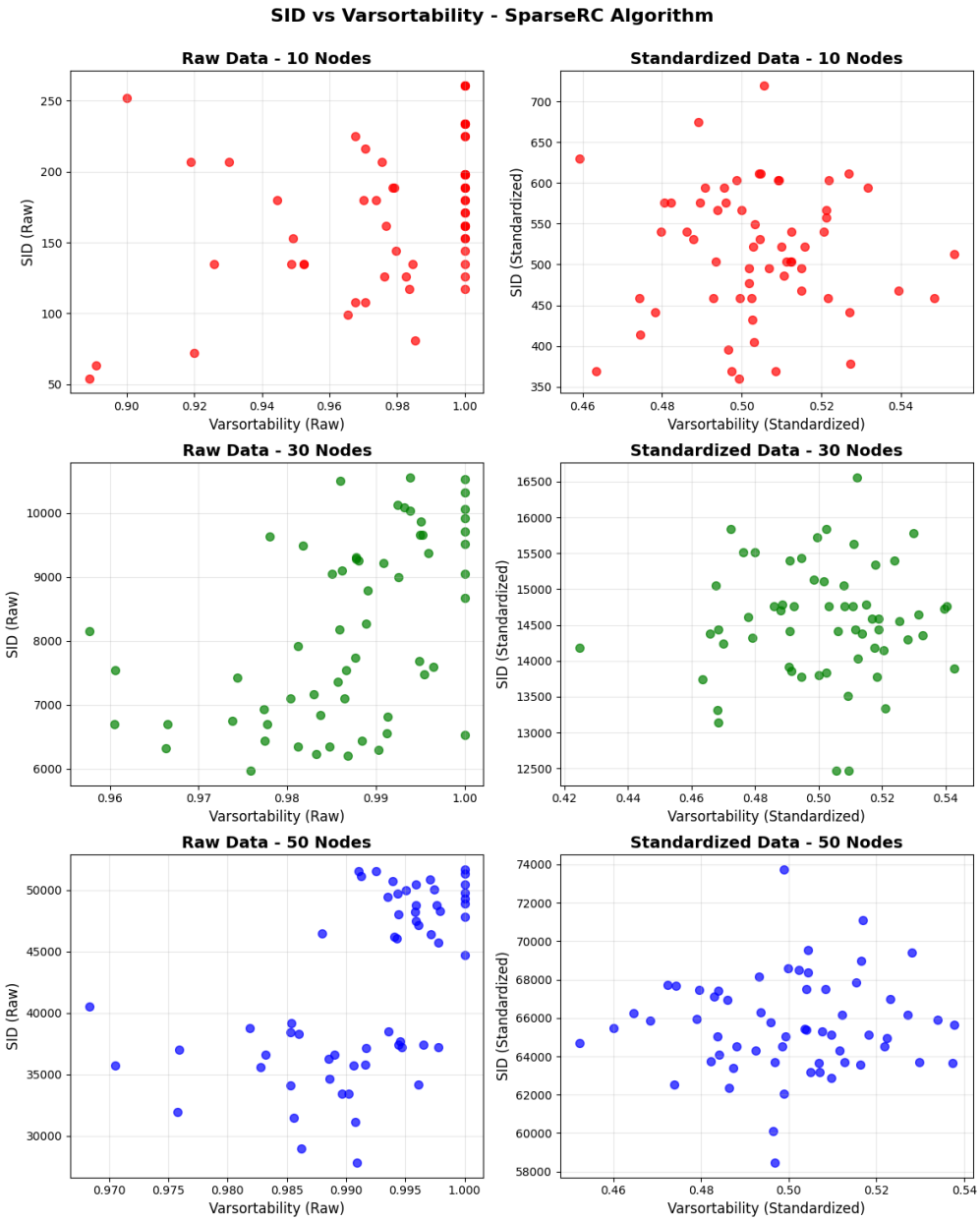

Figure 19: SID vs. varsortability for PC

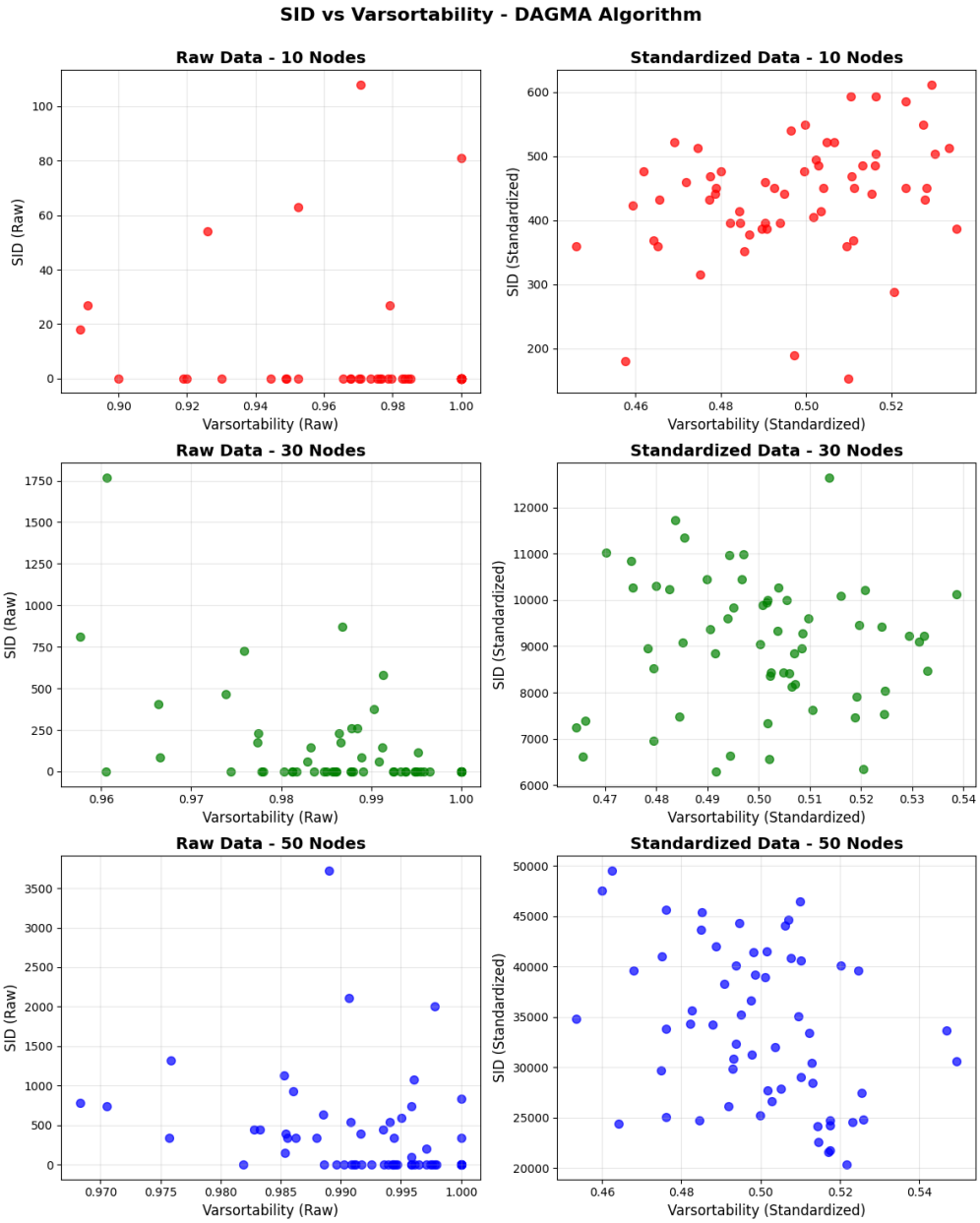

Figure 20: SID vs. varsortability for Sortnregress

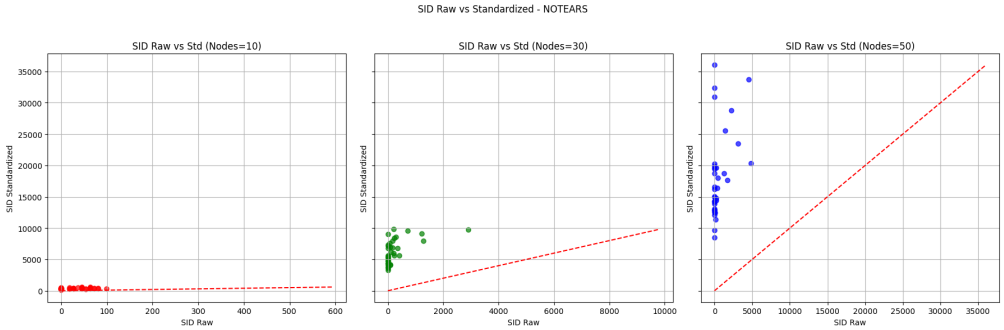

Figure 21: Standardized vs. raw SID for NOTEARS

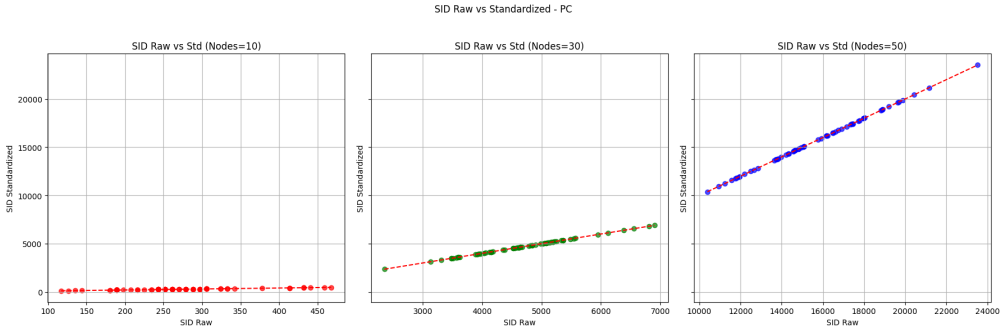

Figure 22: Standardized vs. raw SID for DAGMA

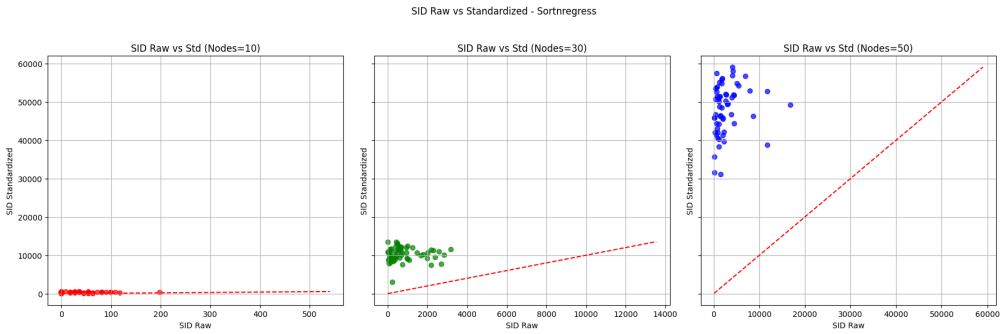

Figure 23: Standardized vs. raw SID for SparseRC

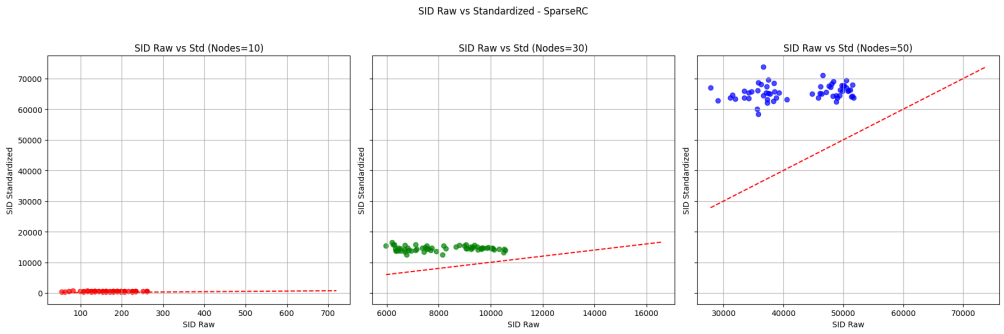

Figure 24: Standardized vs. raw SID for PC

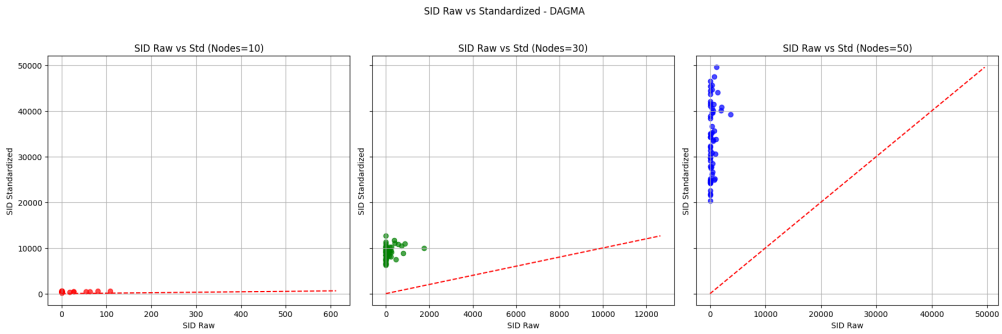

Figure 25: Standardized vs. raw SID for Sortnregress

A.1.3 Balanced Scoring Function

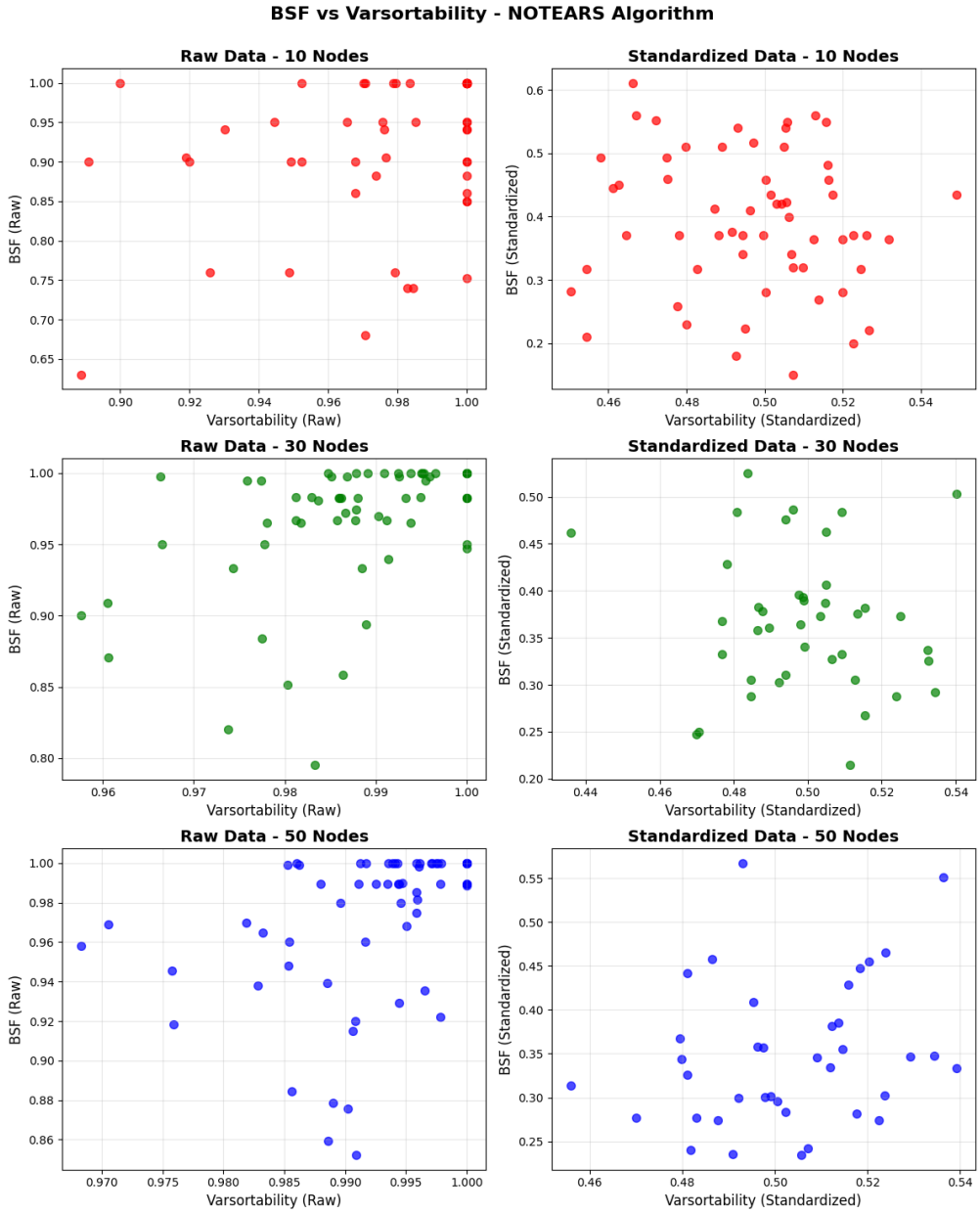

Figure 26: BSF vs. varsortability for NOTEARS

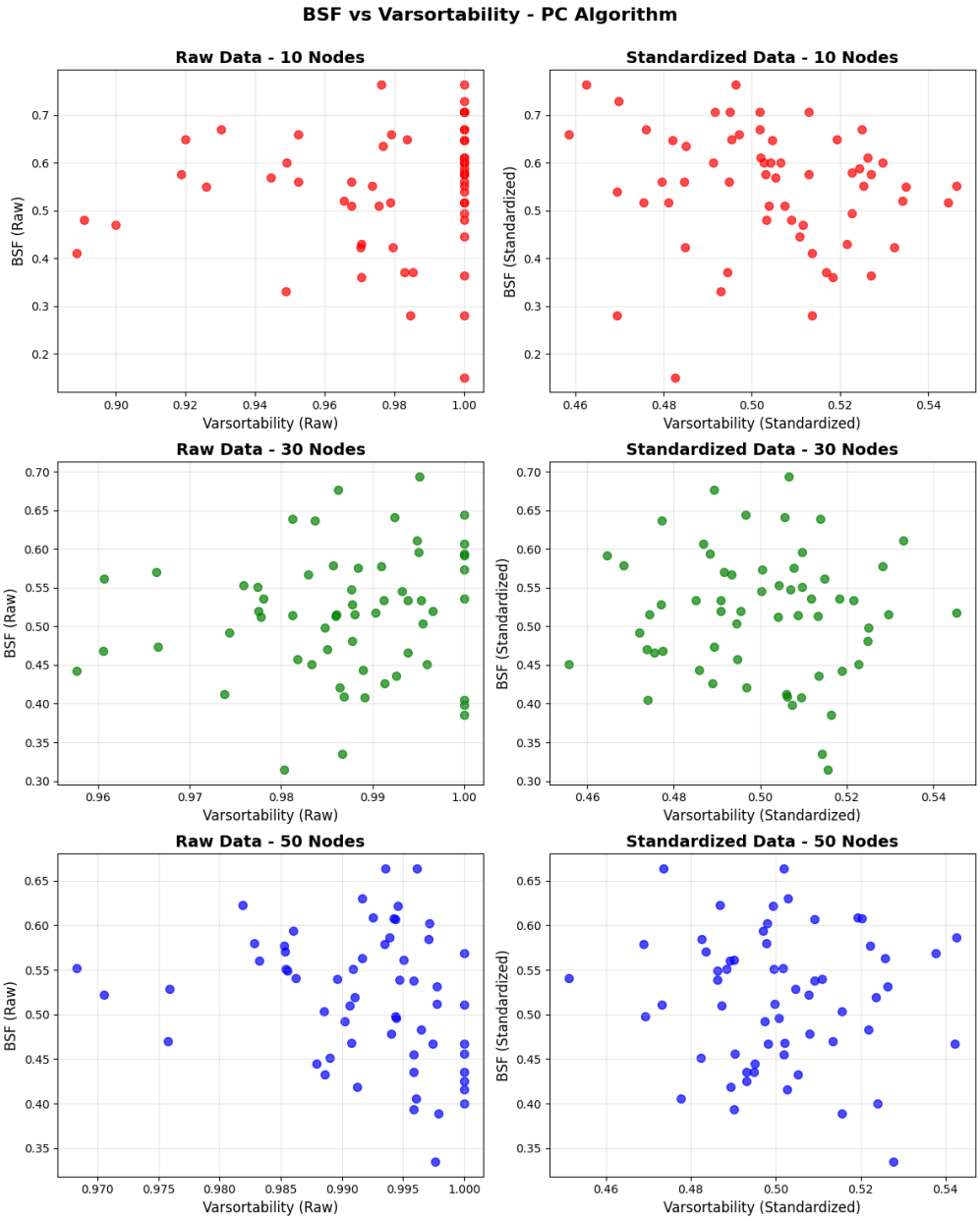

Figure 27: BSF vs. varsortability for DAGMA

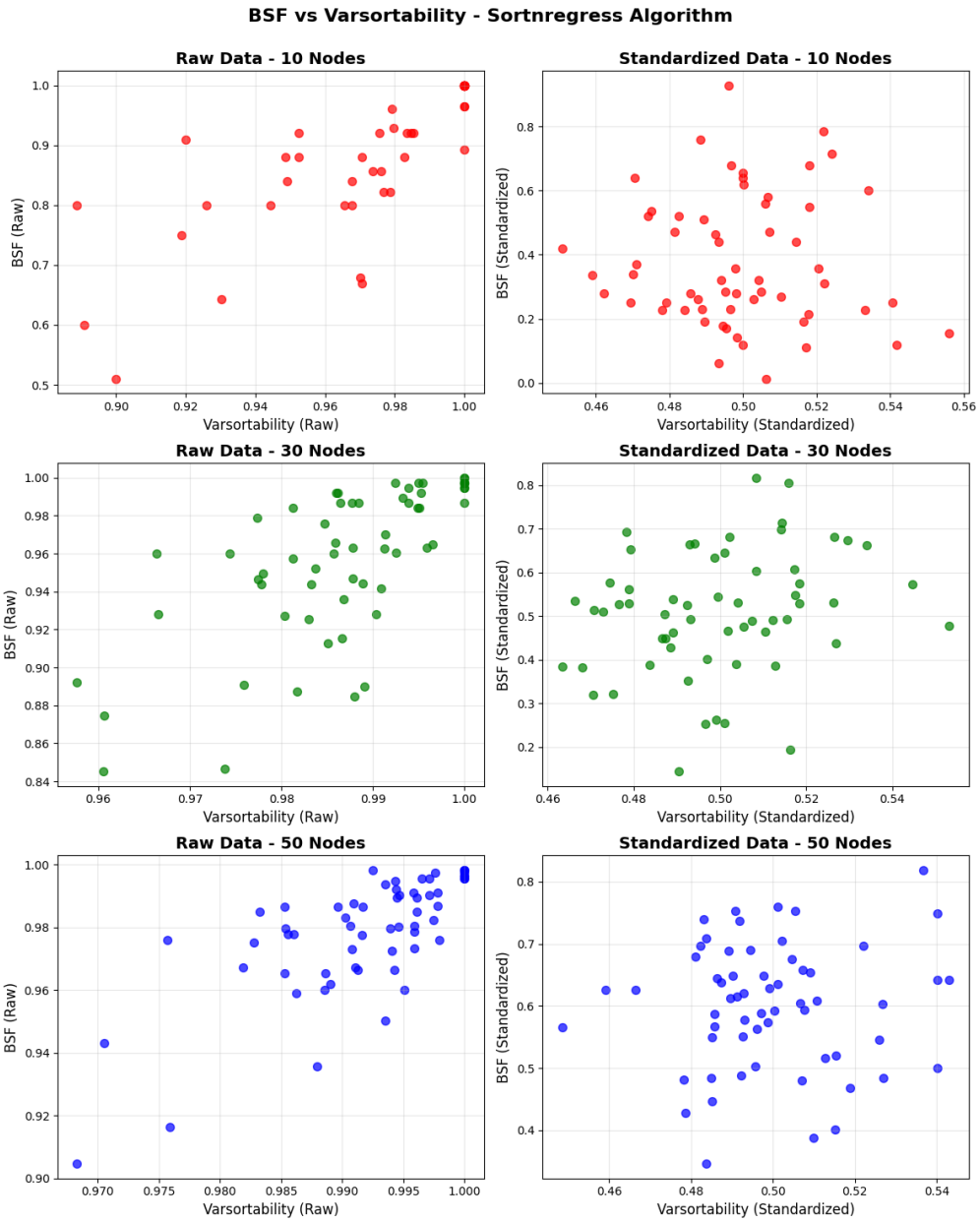

Figure 28: BSF vs. varsortability for SparseRC

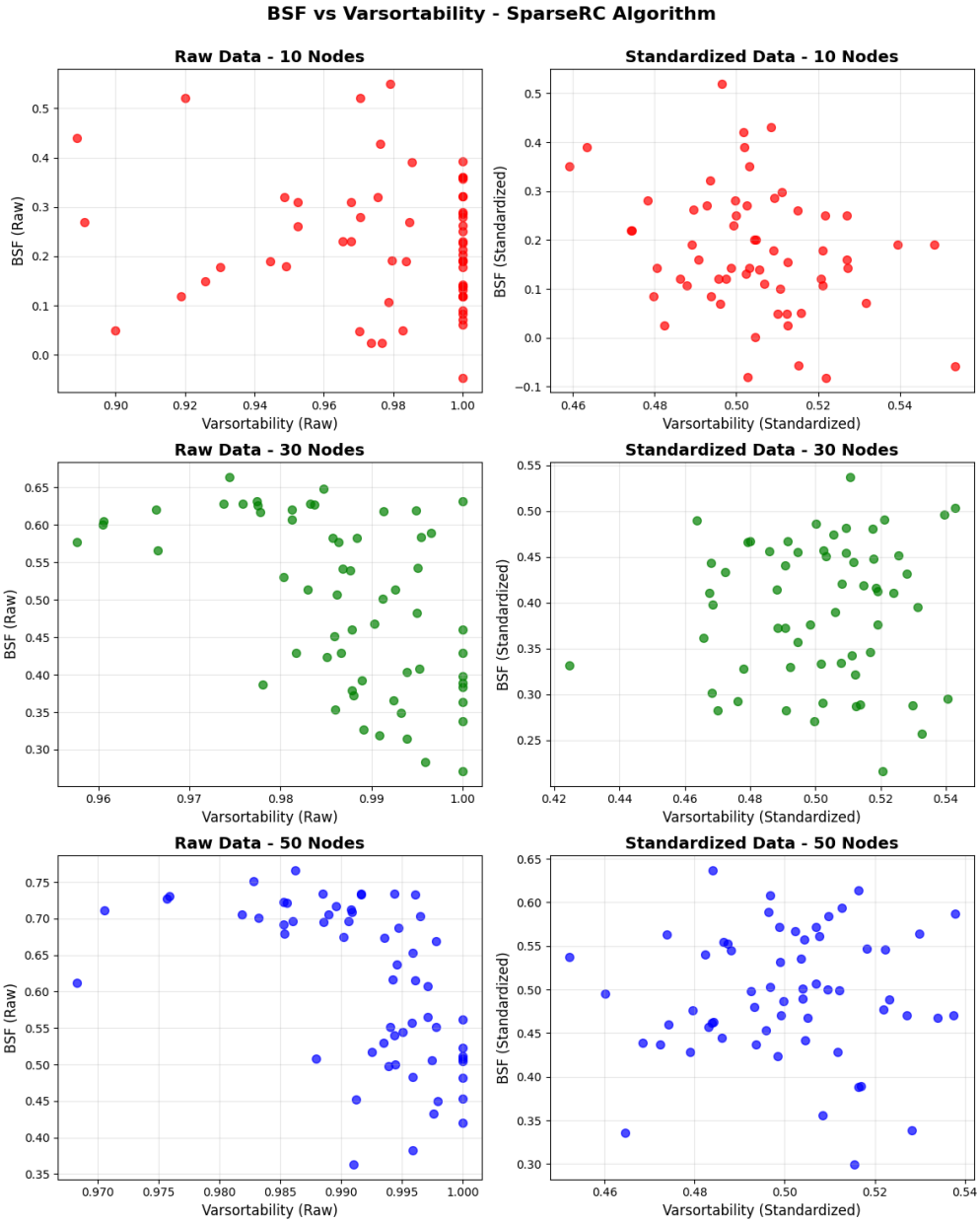

Figure 29: BSF vs. varsortability for PC

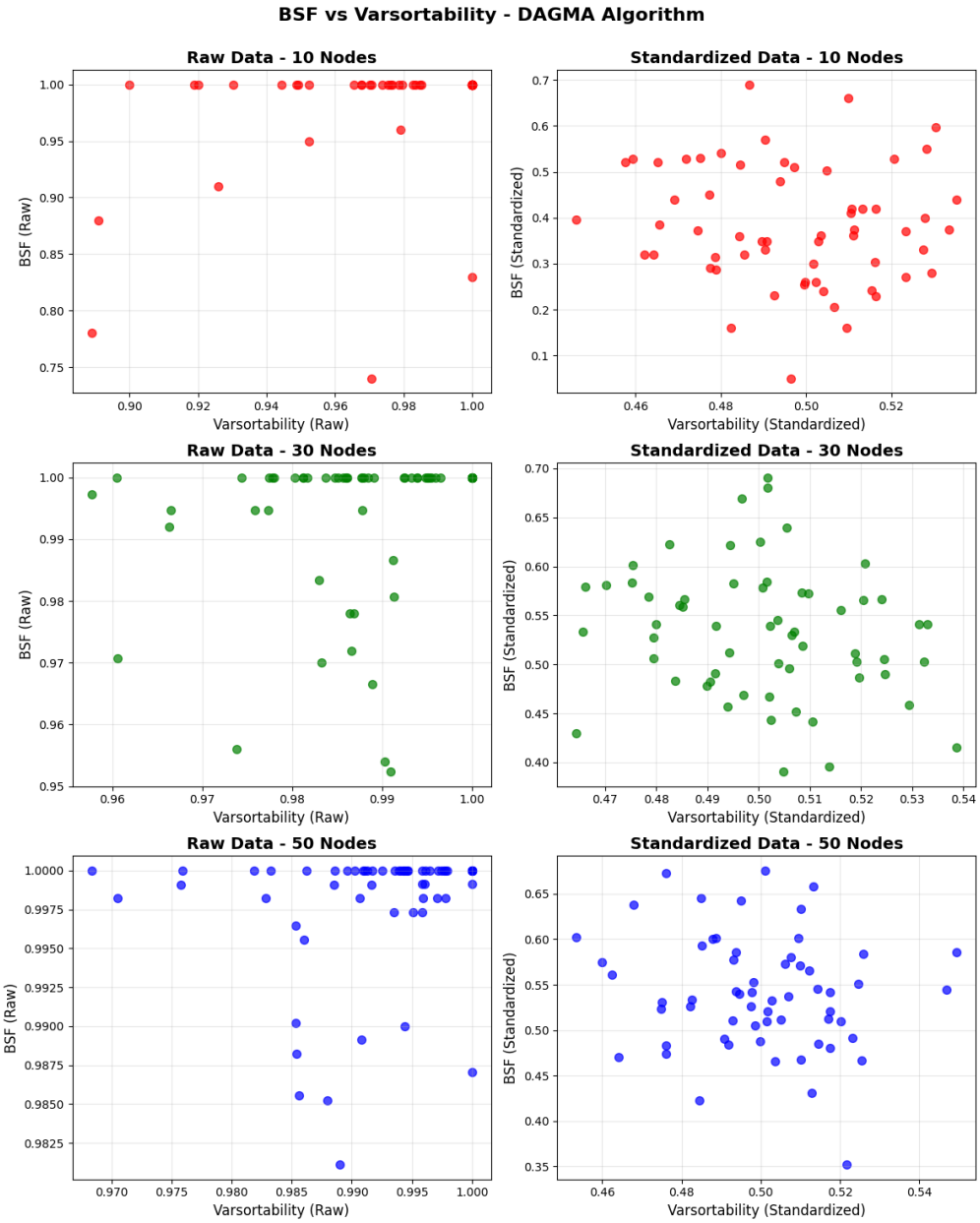

Figure 30: BSF vs. varsortability for Sortnregress

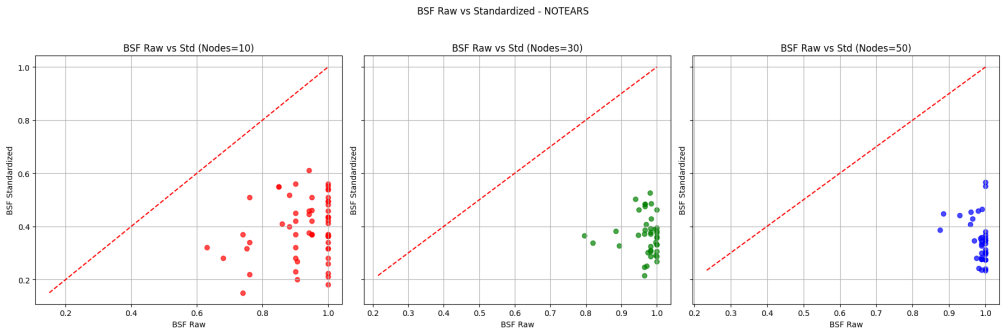

Figure 31: Standardized vs. raw BSF for NOTEARS

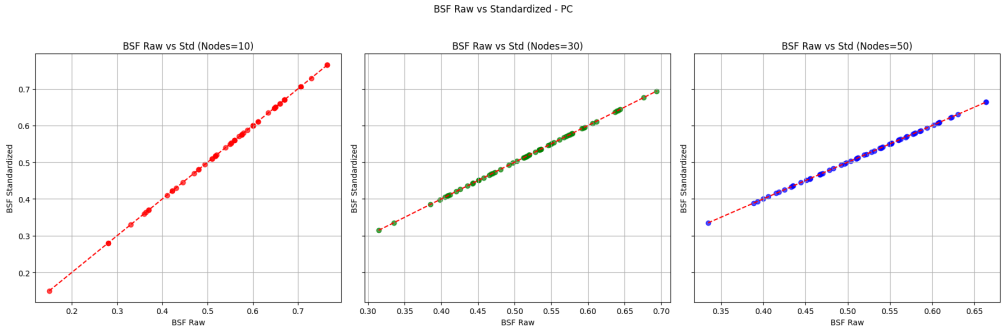

Figure 32: Standardized vs. raw BSF for DAGMA

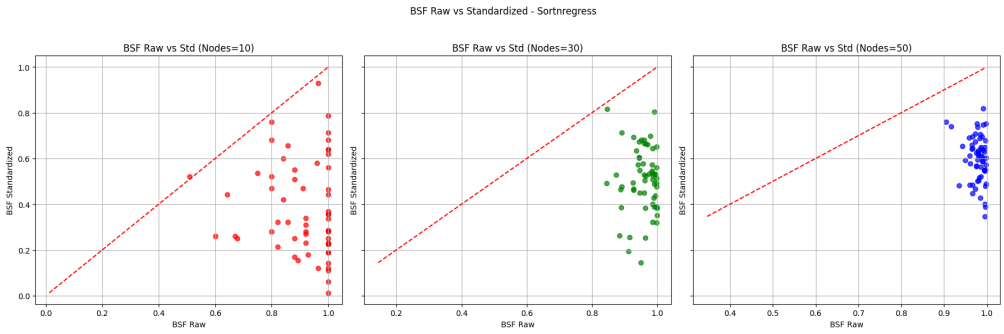

Figure 33: Standardized vs. raw BSF for SparseRC

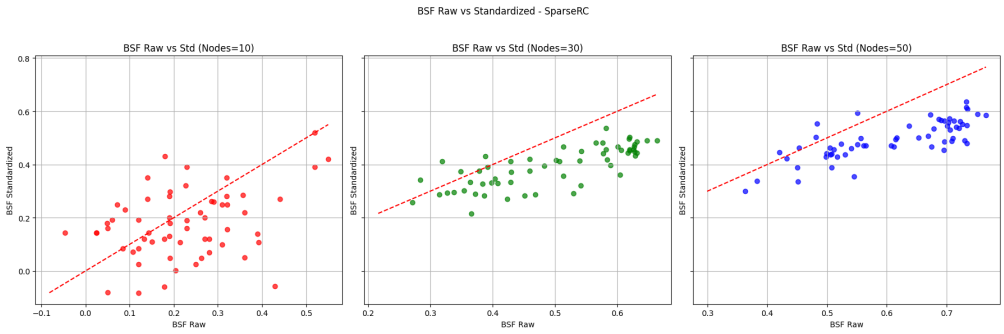

Figure 34: Standardized vs. raw BSF for PC

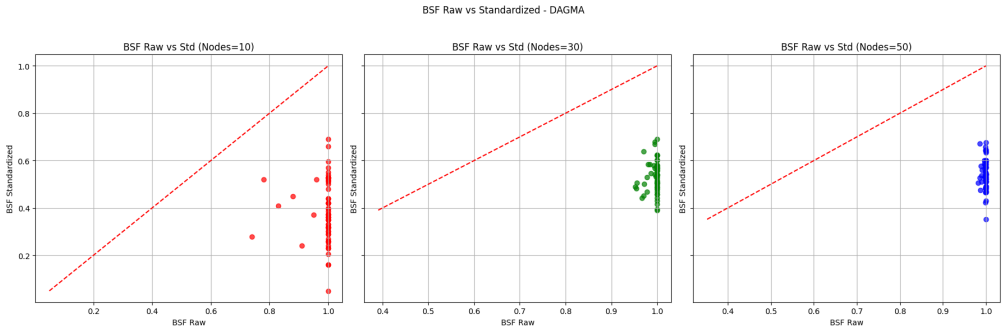

Figure 35: Standardized vs. raw BSF for Sortnregress

A.2 Kernel Density Estimate Plots

A.2.1 Structural Hamming Distance

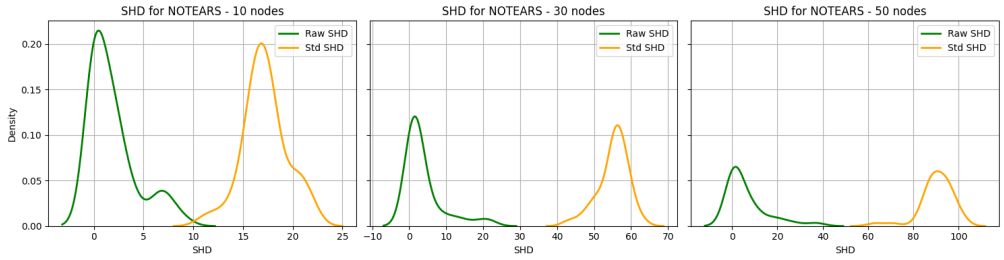

Figure 36: KDE plot for SHD (NOTEARS)

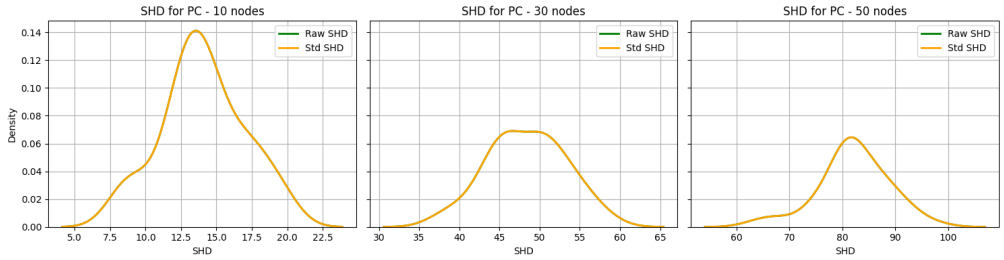

Figure 37: KDE plot for SHD (DAGMA)

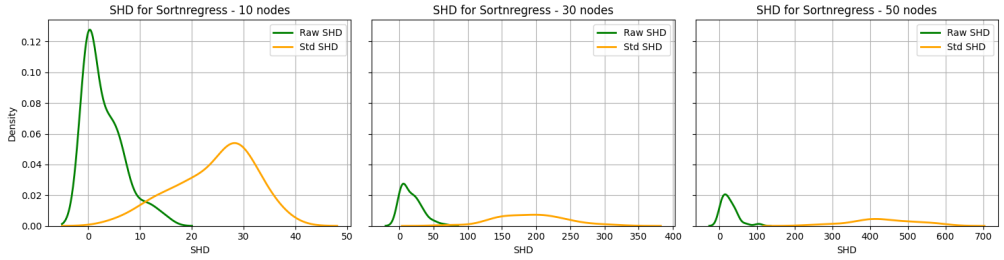

Figure 38: KDE plot for SHD (SparseRC)

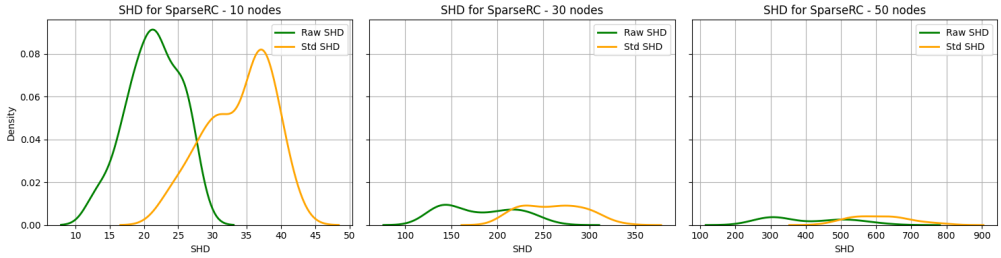

Figure 39: KDE plot for SHD (PC)

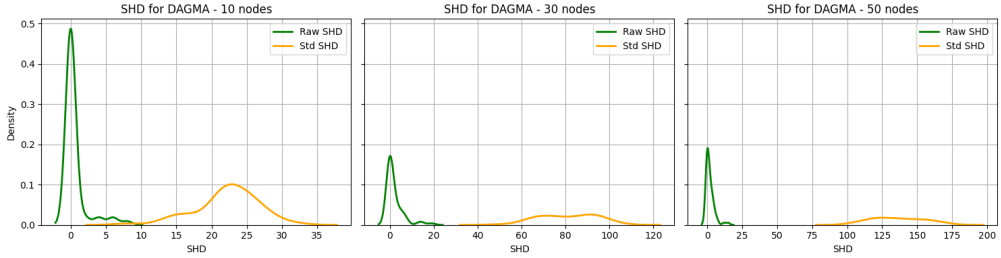

Figure 40: KDE plot for SHD (Sortnregress)

A.2.2 Structural Intervention Distance

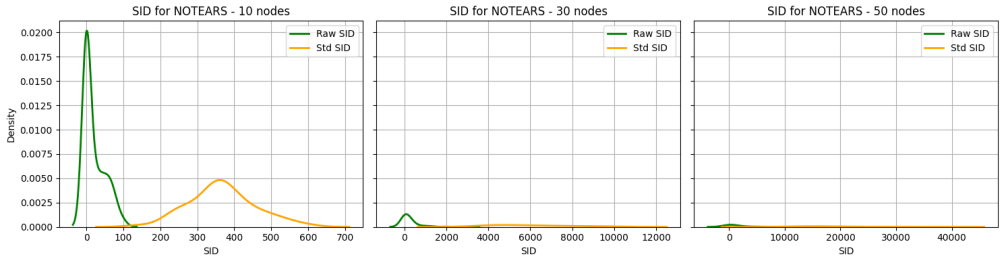

Figure 41: KDE plot for SID (NOTEARS)

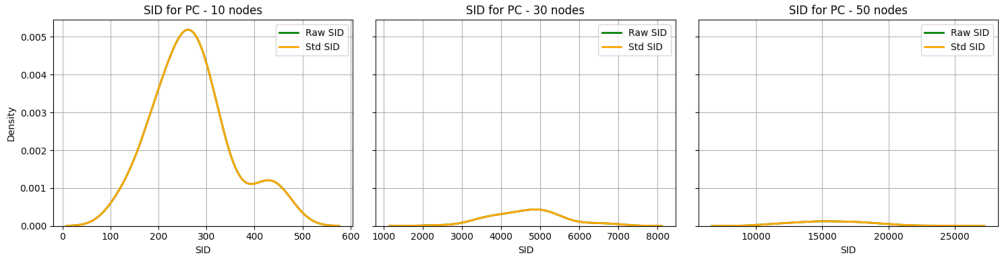

Figure 42: KDE plot for SID (DAGMA)

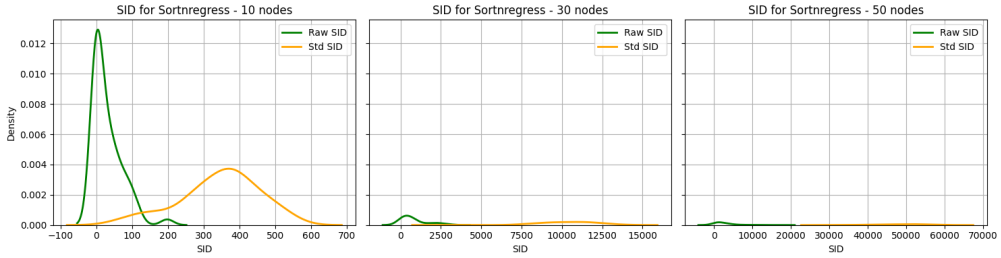

Figure 43: KDE plot for SID (SparseRC)

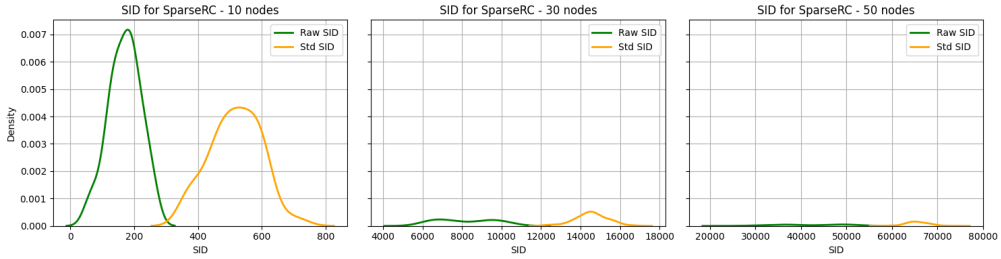

Figure 44: KDE plot for SID (PC)

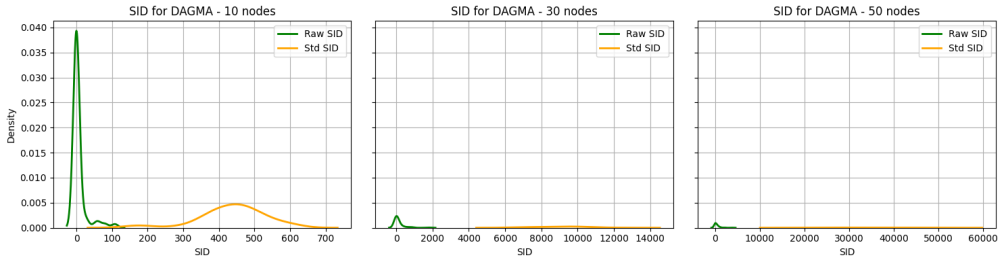

Figure 45: KDE plot for SID (Sortnregress)

A.2.3 Balanced Scoring Function

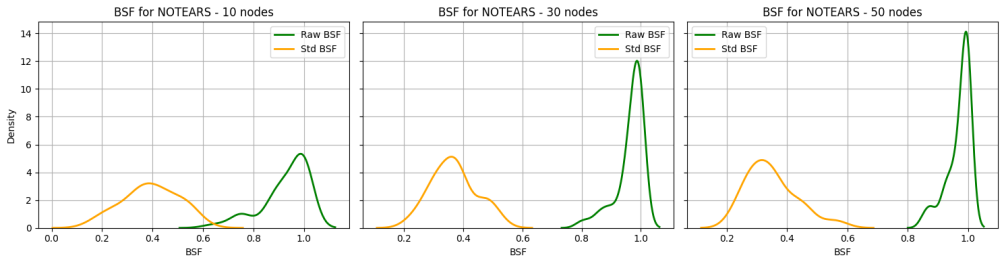

Figure 46: KDE plot for BSF (NOTEARS)

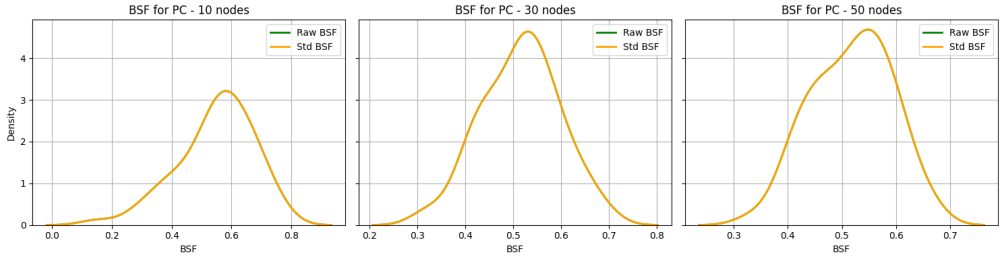

Figure 47: KDE plot for BSF (DAGMA)

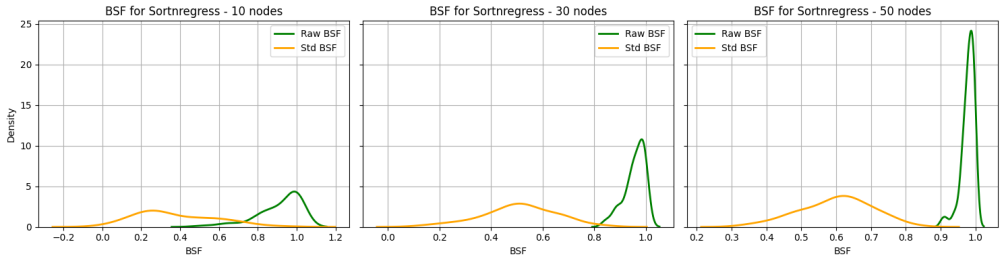

Figure 48: KDE plot for BSF (SparseRC)

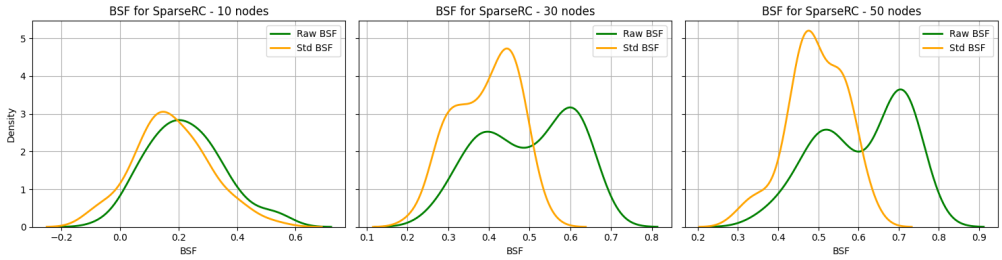

Figure 49: KDE plot for BSF (PC)

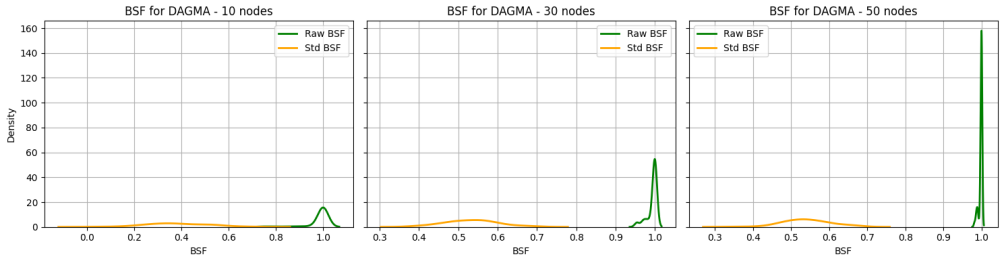

Figure 50: KDE plot for BSF (Sortnregress)

References

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

[2] David Maxwell Chickering. Optimal structure identification with greedy search. *J. Mach. Learn. Res.*, 3(null):507–554, March 2003.

[3] Davin Choo, Kirankumar Shiragur, and Arnab Bhattacharyya. Verification and search algorithms for causal dags. In S. Koyejo, S. Mohamed, A. Agarwal, D. Belgrave, K. Cho, and A. Oh, editors, *Advances in Neural Information Processing Systems*, volume 35, pages 12787–12799. Curran Associates, Inc., 2022.

[4] Tom Claassen, Joris Mooij, and Tom Heskes. Learning sparse causal models is not np-hard. *arXiv preprint arXiv:1309.6824*, 2013.

[5] Anthony C Constantinou. Evaluating structure learning algorithms with a balanced scoring function. *arXiv preprint arXiv:1905.12666*, 2019.

[6] Chang Deng, Kevin Bello, Pradeep Ravikumar, and Bryon Aragam. Global optimality in bivariate gradient-based dag learning. In A. Oh, T. Naumann, A. Globerson, K. Saenko, M. Hardt, and S. Levine, editors, *Advances in Neural Information Processing Systems*, volume 36, pages 17929–17968. Curran Associates, Inc., 2023.

[7] Marcus Kaiser and Maksim Sipos. Unsuitability of notears for causal graph discovery when dealing with dimensional quantities. *Neural Processing Letters*, 54(3):1587–1595, 2022.

[8] Panagiotis Misiakos, Chris Wendler, and Markus Püschel. Learning dags from data with few root causes. In A. Oh, T. Naumann, A. Globerson, K. Saenko, M. Hardt, and S. Levine, editors, *Advances in Neural Information Processing Systems*, volume 36, pages 16865–16888. Curran Associates, Inc., 2023.

[9] Panagiotis Misiakos, Chris Wendler, and Markus Püschel. Learning dags from data with few root causes. *Advances in Neural Information Processing Systems*, 36:16865–16888, 2023.

[10] 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.

[11] Jonas Peters and Peter Bühlmann. Structural intervention distance (sid) for evaluating causal graphs. *arXiv preprint arXiv:1306.1043*, 2013.

[12] 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.

[13] Peter Spirtes, Clark Glymour, and Richard Scheines. *Causation, Prediction, and Search*. The MIT Press, 01 2001.

[14] Ioannis Tsamardinos, Constantin F Aliferis, Alexander R Statnikov, and Er Statnikov. Algorithms for large scale markov blanket discovery. In *FLAIRS*, volume 2, pages 376–81, 2003.

[15] Zhen Zhang, Ignavier Ng, Dong Gong, Yuhang Liu, Ehsan Abbasnejad, Mingming Gong, Kun Zhang, and Javen Qinfeng Shi. Truncated matrix power iteration for differentiable dag learning. *Advances in Neural Information Processing Systems*, 35:18390–18402, 2022.

[16] 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.