Improving TabPFN's Synthetic Data Generation by Integrating Causal Structure

Davide Tugnoli[1,2] **Andrea De Lorenzo**[3] **Marco Virgolin**[4] **Giovanni Cinà**[5,6]

[1]Department of Mathematics, Informatics and Geosciences, University of Trieste, Trieste, Italy
[2]InSilicoTrials Technologies, Riva Grumula 2, 34123 Trieste, Italy
[3]Department of Engineering and Architecture, University of Trieste, Trieste, Italy
[4]InSilicoTrials Technologies BV, The Netherlands
[5]Medical Informatics Dept., Amsterdam UMC, The Netherlands
[6]Institute for Logic, Language and Computation, University of Amsterdam, The Netherlands

Abstract

Synthetic tabular data generation addresses data scarcity and privacy constraints in a variety of domains. Tabular Prior-Data Fitted Network (TabPFN), a recent foundation model for tabular data, has been shown capable of generating high-quality synthetic tabular data. However, TabPFN is autoregressive: features are generated sequentially by conditioning on the previous ones, depending on the order in which they appear in the input data. We demonstrate that when the feature order conflicts with causal structure, the model produces spurious correlations that impair its ability to generate synthetic data and preserve causal effects. We address this limitation by integrating causal structure into TabPFN's generation process through two complementary approaches: Directed Acyclic Graph (DAG)-aware conditioning, which samples each variable given its causal parents, and a partially directed acyclic graph (PDAG)-based strategy for scenarios with partial causal knowledge. We evaluate these approaches on controlled benchmarks and six CSuite datasets, assessing structural fidelity, distributional quality, and Average Treatment Effect (ATE) preservation. Across most settings, DAG-aware conditioning improves the quality and stability of synthetic data relative to vanilla TabPFN. Under partial causal knowledge, the oracle partially directed acyclic graph (oracle-PDAG), which orients only the edges into the colliders, shows moderate gains, while the benefit of a Completed Partially Directed Acyclic Graph (CPDAG) discovered from data depends on how well the causal structure is recovered. These results indicate that reliable causal structure, even partial, can be injected into TabPFN at inference time, without parameter updates, to improve synthetic data quality.

1 INTRODUCTION

Synthetic tabular data generation is becoming an important approach to handle data scarcity and privacy concerns in many domains [Shi et al., 2025b]. This is especially relevant in fields such as healthcare, finance, and policy research, where privacy regulations and ethical constraints often limit access to real data. In pharmaceutical research, for example, synthetic data can be used to simulate drug effects for safety and efficacy, while protecting patient confidentiality. For tabular data, generation methods must avoid producing close copies of training samples, especially when data are limited.

However, generating reliable synthetic tabular data is challenging because real datasets usually contain complex causal relations among variables. Generation methods that ignore such dependencies may create spurious correlations that differ from the true data-generating process. When synthetic data are used to augment limited real data or simulate clinical scenarios, misleading dependencies can propagate to treatment effect estimates. In drug development, for instance, inaccurate estimation of treatment effects from flawed synthetic data could lead to costly trials on ineffective drugs or the exclusion of promising ones.

Recent foundation models for tabular data, such as Tabular Prior-Data Fitted Network (TabPFN) [Hollmann et al., 2025], have shown promising results by pre-training on millions of synthetic datasets derived from Structural Causal Models (SCMs) [Pearl, 2009]. Pre-training on such diverse datasets is particularly valuable when real data are limited, a common scenario in domains like healthcare. This advantage becomes even more important for synthetic data generation, which requires predicting all features rather than a single target variable as in traditional classification or regression tasks. Extensions of TabPFN[1] enable unsupervised, autoregressive generation of synthetic tabular data. However, TabPFN's autoregressive approach generates variables

[1]https://github.com/PriorLabs/tabpfn-extensions

Proceedings of the 42nd Conference on Uncertainty in Artificial Intelligence (UAI 2026), PMLR 337:6807–6850.

sequentially without explicitly accounting for causal structure. Although the model averages predictions over random permutations of the conditioning set, this mitigates but does not eliminate order sensitivity: during generation, each variable can only condition on features that appear earlier in the sequence. When the generation order conflicts with causal dependencies, particularly in the presence of colliders, the model may introduce spurious correlations that compromise the reliability of synthetic data.

This work studies the autoregressive nature of TabPFN-based synthetic data generation, identifies a fundamental limitation, and proposes methods to address it. Our contributions are as follows:

1. We demonstrate that TabPFN synthetic data quality depends heavily on the order of the features in the input data due to the absence of causal reasoning, with this sensitivity persisting even at large training sizes.
2. We propose causal conditioning strategies that leverage known, partially known, or discovered causal structure at inference time, without parameter updates. We show improvements over vanilla TabPFN for both full Directed Acyclic Graph (DAG) knowledge and the oracle partially directed acyclic graph (oracle-PDAG) across distributional quality metrics, while structure discovered from data has no consistent effect and its potential benefit depends on how well the causal structure is recovered.
3. We quantify how synthetic data errors propagate to treatment effect preservation, demonstrating substantial misestimation that could lead to incorrect decisions in applications such as drug development.

All code, datasets, and detailed numerical results are publicly available.[2]

2 RELATED WORK

Generative approaches to synthetic tabular data range from classical statistical methods to deep models based on Generative Adversarial Networks (GANs), Variational Autoencoders (VAEs), diffusion models, and autoregressive architectures [Shi et al., 2025b].

Early deep generative models adapted architectures from image generation. TGAN [Xu and Veeramachaneni, 2018] applied GANs to mixed continuous–categorical tables using Long Short-Term Memory-based generation, while CTGAN [Xu et al., 2019] addressed class imbalance and multimodal distributions. CTAB-GAN [Zhao et al., 2021] further improved semantic fidelity. More recently, diffusion models have achieved strong performance [Kotelnikov et al., 2023, Zhang et al., 2024, Shi et al., 2025a]. While effective for statistical fidelity, these methods generate all features simultaneously without explicit causal structure.

Autoregressive approaches factorize the joint distribution into sequential conditionals $p(x_j \mid x_{C(j)})$, where $C(j)$ denotes the conditioning set. Early work used Classification And Regression Trees (CART) to generate features sequentially [Reiter, 2005]. GReaT treats tabular rows as text sequences for language model generation [Borisov et al., 2023]. TabularARGN [Tiwald et al., 2025] randomizes feature orderings during training to learn conditionals across different subsets. Among foundation models, TabPFN [Hollmann et al., 2023, 2025] achieves state-of-the-art performance on small datasets via in-context learning. Pre-trained on millions of SCM-sampled datasets, it provides strong inductive biases without target data retraining. TabPFGen extends TabPFN to generative tasks via energy-based sampling rather than autoregressive conditioning [Ma et al., 2024], but neither approach explicitly conditions on causal graphs during generation.

Causal approaches integrate structural knowledge directly into generation. DECAF embeds SCMs into the architecture for interventional debiasing [Van Breugel et al., 2021]. Causal-TGAN structures its generator according to a causal graph [Wen et al., 2022]. DATGAN incorporates expert DAGs to enforce structural dependencies [Lederrey et al., 2022]. CA-GAN combines causal discovery with reinforcement learning [Nguyen et al., 2025]. CausalDiffTab is a diffusion model that learns a causal graph from the data and uses it to regularize generation [Zhang et al., 2025]. Unlike TabPFN, these models are trained from scratch per dataset and explicitly use DAG structure to guide generation.

Recent work has applied transformer-based meta-learning to causal effect estimation. Do-PFN [Robertson et al., 2025] adapts the PFN framework to predict conditional interventional distributions from observational data, taking covariates and treatment as input. CausalPFN [Balazadeh et al., 2025] trains a transformer on a causal prior to estimate expected potential outcomes, also conditioning on covariates and treatment. Both models predict outcomes under intervention but do not model the full joint distribution over all variables. MACE-TNP [Dhir et al., 2025] addresses a related task, estimating Bayesian model-averaged interventional distributions from observational data via neural processes; it similarly targets interventional queries rather than joint density modeling. To our knowledge, this is the first work to combine a foundation model with explicit causal structure for synthetic tabular data generation, addressing autoregressive ordering violations and collider bias through DAG-aware and PDAG-based conditioning strategies.

[2] https://github.com/DavideTugnoli/tabpfn-causal-synthetic

3 METHODOLOGY

3.1 TABPFN CAUSAL EXTENSIONS

TabPFN is a transformer-based model pre-trained on approximately 130 million synthetic datasets, each sampled from a randomly generated SCM with diverse graph structures, functional relationships, and noise distributions [Hollmann et al., 2025]. At inference time, it receives an entire dataset as context and produces predictions in a single forward pass via in-context learning, without retraining on the target data. For synthetic data generation, the model is applied autoregressively: features are generated sequentially following the column order of the input data. Denoting this ordering as π, each feature $x_{\pi(i)}$ at position i in the sequence is sampled from a conditional distribution

$$p\big(x_{\pi(i)} \mid \mathcal{C}(\pi(i))\big)\,, \tag{1}$$

where the conditioning context $\mathcal{C}(\pi(i)) = \{x_{\pi(0)}, \ldots, x_{\pi(i-1)}\}$ consists of all features preceding position i in the ordering. When no context is available ($i = 0$), the model conditions on random noise to approximate the marginal distribution.

Hollmann et al. [2025] recognize that the autoregressive nature induces a specific bias due to the local ordering of variables within the conditioning set $\mathcal{C}(\pi(i))$. To mitigate this issue, they propose averaging predictions over multiple random permutations of the conditioning set at inference time. However, this averaging strategy does not address the limitation imposed by the global ordering π. When π does not respect the causal structure, the model may still condition on descendant variables when generating their ancestors, inducing spurious correlations.

For instance, consider a collider structure $X \to Z \leftarrow Y$, where the causes X and Y are marginally independent. If the global ordering places the common effect Z before its parents, the model may, e.g., generate X conditioned on Z, then Y conditioned on both Z and X. Conditioning on the collider Z makes X and Y conditionally dependent. Sampling from exact conditionals would still recover their marginal independence, but estimation errors in the learned conditionals propagate this dependence to the synthetic data, introducing spurious correlations.

To address this limitation, we introduce causal conditioning strategies that align TabPFN's generative process with known or partially known causal structures. These strategies act only on TabPFN's input and differ only in the conditioning set. For each variable, we present the model with the target column and its conditioning columns and query it through the standard in-context interface, leaving the pre-trained network unchanged. Causal relationships between variables can be formally represented using a DAG $\mathcal{G} = (V, E)$, where nodes V correspond to variables and directed edges E indicate direct causal influences [Pearl, 2009]. When such a graph is available, we condition each variable only on its causal parents rather than on all previously generated features:

$$\mathcal{C}_{\mathrm{DAG}}(x_{\pi(i)}) = \{x_j : x_j \to x_{\pi(i)} \text{ in } \mathcal{G}\}. \tag{2}$$

Variables are generated following a topological ordering of $\mathcal{G}$, ensuring that all parents have been generated before each child.[3]

However, in most real-world scenarios, the full DAG is unknown, and a causal discovery algorithm can recover only a partial graph from data. Such a graph can be represented by a Completed Partially Directed Acyclic Graph (CPDAG) [Chickering, 2002], which includes directed edges where the orientation is uniquely determined and undirected edges where multiple orientations remain compatible with the data. The discovered CPDAG represents a realistic scenario, though its recovered orientations may contain errors. As a correct-by-construction reference, we also evaluate the oracle-PDAG, which orients only the edges into the true DAG's colliders. Derived directly from the true graph, it isolates the effect of these orientations from finite-sample discovery errors.

Due to the partially directed structure of either partial graph, Equation (2) cannot be readily applied. We define a generation ordering σ that places variables with known causal parents before the remaining ones, and propose the following hybrid conditioning strategy, applied to both graphs by letting $\mathcal{G}^*$ denote the discovered CPDAG or the oracle-PDAG:

$$\mathcal{C}(x_{\sigma(i)}) = \begin{cases} \mathrm{pa}_{\mathcal{G}^*}(x_{\sigma(i)}) & \text{if fully directed,} \\ \{x_{\sigma(0)}, \ldots, x_{\sigma(i-1)}\} & \text{otherwise,} \end{cases} \tag{3}$$

where "fully directed" means that $x_{\sigma(i)}$ has at least one directed adjacent edge and no undirected adjacent edges in $\mathcal{G}^*$. Variables satisfying this condition are generated according to their causal parents, while the rest revert to sequential conditioning on all predecessors in the ordering. When every edge in $\mathcal{G}^*$ is directed, Eq. (3) recovers the conditioning set of Eq. (2) for all non-isolated variables; when no edge is directed, it reduces to the conditioning set of Eq. (1). Because the ordering places oriented parents before their children, this fallback still conditions on them. It drops only the restriction of conditioning on the parents alone, without assuming an orientation for the undirected edges.

[3] A topological ordering alone does not imply DAG-aware conditioning: the vanilla strategy of Eq. (1) can also follow a topological ordering, but conditions on all predecessors rather than on causal parents only.

4 EXPERIMENTAL DESIGN

4.1 EVALUATION METRICS AND CONTROLS

We evaluate synthetic data quality using metrics widely adopted in benchmark frameworks such as SynthEval [Lautrup et al., 2025] and the Differential Privacy Synthetic Data Challenge organized by the National Institute of Standards and Technology [Ridgeway et al., 2021]. We focus on dependency structure, measured by the Correlation Matrix Difference (CMD), and use the k-Marginal Total Variation Distance (kMTVD) as a robustness check on pairwise fidelity. We additionally report the Nearest-Neighbor Adversarial Accuracy (NNAA) [Yale et al., 2020], a nearest-neighbor resemblance metric used in privacy-oriented evaluations of synthetic data, which measures how distinguishable synthetic and real data are, defined in Appendix A and reported with each comparison in the appendix. The CMD quantifies how well the overall pairwise dependency structure among variables is preserved. We compute mixed correlation matrices combining Cramér's V for categorical–categorical pairs, the correlation ratio η for categorical–numerical pairs, and Spearman's rank correlation for numerical–numerical pairs. We replace Pearson (the SynthEval default) with Spearman to capture monotonic relationships in datasets with nonlinear dependencies. The metric is computed as the Frobenius norm of the difference between real and synthetic correlation matrices, i.e., $\text{CMD} = \|C_{\text{real}} - C_{\text{synthetic}}\|_F$.

The kMTVD with $k = 2$ measures pairwise distributional fidelity. Continuous variables are discretized into $B = 20$ quantile-based bins. For two empirical distributions P and Q, the total variation distance is $\text{TVD} = \frac{1}{2}\sum_x |P(x) - Q(x)|$, and the metric is the mean TVD across all variable pairs.

We assess the statistical significance of differences between conditioning strategies using the Wilcoxon signed-rank test with Pratt's method for handling tied observations, applying Holm correction for prespecified comparisons. Effect sizes are quantified using the Hodges–Lehmann estimator, the median of pairwise averages of differences [Hodges Jr and Lehmann, 1963].

4.2 DATASETS

We conduct experiments on three dataset classes, ranging from fully controlled hand-crafted settings to public benchmark datasets and realistic clinical scenarios (Table 1).

4.2.1 Custom Collider SCM

We design a four-variable SCM containing a collider to evaluate TabPFN's sensitivity to causal structure under fully controlled conditions. Colliders (variables with multiple parent edges) pose a fundamental challenge for autoregressive generators because they induce conditional dependencies between otherwise independent variables.

The DAG of this SCM is defined as $X_0 \to X_1 \leftarrow X_2 \leftarrow X_3$, where X_1 is the collider node. Although X_0 and X_2 are marginally independent ($X_0 \perp\!\!\!\perp X_2$), conditioning on their common child X_1 induces dependence between them ($X_0 \not\perp\!\!\!\perp X_2 \mid X_1$). When an autoregressive generator conditions on X_1 while generating its parents, it correctly learns that X_0 and X_2 are dependent given X_1. However, sequential sampling from these conditionals propagates this dependence into the marginal distribution, creating spurious marginal correlation between variables that should be independent. We construct the SCM with near-deterministic linear relationships to amplify this collider bias effect and enable clear detection of spurious associations introduced by inappropriate conditioning strategies. Full structural equations appear in Appendix B.

4.2.2 CSuite Benchmark Datasets

Next, we consider the Microsoft CSuite benchmark [Geffner et al., 2024], a collection of hand-crafted SCMs designed for evaluating causal discovery and inference under known ground-truth graphs.

We select six datasets with at least four nodes and distinct structures, covering a range of causal scenarios: CSuite Nonlin Simpson (CNS) and CSuite Mixed Simpson (CMS) (four nodes each) are variants of Simpson's paradox with continuous or mixed-type variables; CSuite Symprod Simpson (CSS) (four nodes) is a nonlinear benchmark with a product interaction. CSuite Mixed Confounding (CMC) (twelve nodes) combines mixed data types and complex confounding structures, while CSuite Large Backdoor (CLB) and CSuite Weak Arrows (CWA) (nine nodes each) have large topologies with backdoor paths and weak causal effects, respectively. Each dataset includes predefined interventions for treatment effect estimation.

4.2.3 Simglucose Type 1 Diabetes Mellitus Simulator

Finally, we evaluate on a realistic dataset derived from the UVA/Padova Type 1 Diabetes Mellitus (T1DM) Simulator[4] [Dalla Man et al., 2009], a Food and Drug Administration (FDA)-approved model of glucose-insulin dynamics. The original simulator integrates a system of ordinary differential equations over time. Following the approach in Esponera and Cinà [2025], we use an acyclic reformulation in a static setting by removing temporal dependencies.

The dataset contains 38 variables, including physiological states, patient-specific parameters, and exogenous ac-

[4] https://github.com/jxx123/simglucose

Table 1: Overview of the datasets used in our experiments, from controlled hand-crafted SCMs to CSuite benchmarks and a realistic clinical simulator.

Dataset	Abbr.	Vars.	Structural role	Causal info
Custom collider SCM	CSM	4	Collider	Known true DAG
CSuite Nonlin Simpson	CNS	4	Simpson's paradox, continuous	Known true DAG
CSuite Symprod Simpson	CSS	4	Nonlinear product interaction	Known true DAG
CSuite Mixed Simpson	CMS	4	Simpson's paradox, mixed-type	Known true DAG
CSuite Mixed Confounding	CMC	12	Complex confounding, mixed-type	Known true DAG
CSuite Large Backdoor	CLB	9	Backdoor paths	Known true DAG
CSuite Weak Arrows	CWA	9	Weak causal effects	Known true DAG
Simglucose	SGL	38	Physiological (static T1DM)	Partial (full DAG unknown)

tions. Although the physiological equations define a clear causal progression among the endogenous states, the relations among patient-specific parameters are not known, i.e., we do not know the full DAG. These parameters are sampled from distributions based on clinical data and represent heterogeneous virtual patients. Because only partial causal knowledge is available, we test whether partial topological ordering still improves generation quality.

4.3 SYNTHETIC DATA QUALITY

To ensure reproducibility, we conduct all experiments using the official implementation of TabPFN (version 2.1.0). We compare three conditioning strategies to assess how causal structure affects synthetic data quality: *vanilla* TabPFN (which conditions sequentially on predecessors, Equation (1)), DAG-aware generation (which conditions on causal parents), and partially directed acyclic graph (PDAG)-based generation (which combines directed and undirected causal relationships). We conduct experiments on the Custom SCM (CSM), the six CSuite benchmark datasets, and Simglucose (SGL), using the default value of 3 permutations for the internal averaging mechanism described in Section 3.1.

To assess robustness across different data availability scenarios, we test multiple training sizes $N \in \{20, 50, 100, 200, 500\}$[5] with 100 resampling iterations per configuration. We strictly separate training and test data by creating fixed test sets of 2000 samples for each dataset. For the CSM and SGL, we generate 6000 samples total, select test samples once using a fixed random seed, and sample training data from the remaining 4000 samples. For CSuite benchmarks, we use the predefined test sets and sample training data from the remaining pool.

To study the susceptibility of TabPFN-based data generation to the input ordering, we evaluate three column orderings: *original ordering* (as provided in the dataset, or a random permutation if it coincides with topological or reverse topological), *topological ordering* (parents before children), and *reverse topological ordering* (children before parents), which maximizes causal violations, illustrating a potential worst-case scenario. We use the original ordering as the default-use baseline, since it reflects how the model is typically applied without causal knowledge, and verify its robustness to random column permutations in Appendix D.

For PDAG-based generation, we consider two settings: a realistic one, where CPDAGs are *discovered* from the training data via the order-independent Peter-Clark (PC)-stable algorithm ($\alpha = 0.05$) [Colombo and Maathuis, 2014]; and an ablation with full prior knowledge, the oracle-PDAG, which orients only the edges into the colliders of the true DAG and leaves the other edges undirected. The latter applies no further orientation rules (e.g., Meek's rules [Meek, 1995]), so in general it is not a CPDAG. For the custom SCM, we apply the Fisher–Z conditional independence test to assess edge dependencies within the PC algorithm. For CSuite datasets with categorical or mixed-type variables, we use a hybrid strategy: Kernel Conditional Independence (KCI) test for continuous pairs, G^2 test for discrete pairs, and k-Nearest Neighbors Conditional Mutual Information ($k = 5$, 500 permutations) for mixed pairs. For all strategies, we generate synthetic datasets of the same size as the test sets and evaluate them using CMD, kMTVD, and NNAA.

4.4 TREATMENT EFFECT PRESERVATION

We evaluate how well synthetic data preserve causal effects by measuring Average Treatment Effect (ATE) preservation. This addresses a practical scenario: given a small interventional dataset (e.g., from a pilot trial), we generate synthetic data and assess whether the treatment effect is preserved.

For all datasets, we construct fixed balanced test sets by sampling equally from both intervention arms with a fixed random seed, using generated interventional data for CSM and SGL and benchmark-provided data for CSuite. As both arms are sampled under the intervention, the ATE is read directly as the difference in their mean outcomes. For DAG-

[5] N denotes the number of samples provided to TabPFN as in-context learning data.

aware and PDAG-based generation, we apply a do-surgery disconnecting the treatment variable from its parents.

We test training sizes $N \in \{20, 50, 100, 200, 500, 1000\}$ with 100 resampling iterations per configuration, extending to larger sizes as effect estimation generally benefits from more samples. For each training size, we construct the training set by combining equal numbers of samples from both intervention arms into a single dataset, which is then provided to TabPFN as in-context data.

The ground-truth ATE on the test set is computed as $\text{ATE}_{\text{test}} = \mathbb{E}[Y \mid \text{do}(X = x_1)] - \mathbb{E}[Y \mid \text{do}(X = x_0)]$, where x_1 and x_0 denote the two prespecified intervention values and Y the predefined outcome variable for each dataset. Synthetic data are then generated using the three conditioning strategies, and the corresponding ATE on the synthetic data is computed as $\text{ATE}_{\text{synthetic}} = \mathbb{E}[\hat{Y} \mid \text{do}(X = x_1)] - \mathbb{E}[\hat{Y} \mid \text{do}(X = x_0)]$, where the expectation here is taken over the synthetic data. Our evaluation metric is the absolute ATE difference, $\Delta_{\text{ATE}} = |\text{ATE}_{\text{test}} - \text{ATE}_{\text{synthetic}}|$. Lower values indicate better preservation of causal effects. In comparative analyses, we test whether one method produces significantly lower absolute errors than another.

5 RESULTS

5.1 SYNTHETIC DATA QUALITY

Topological Column Ordering Improves Vanilla TabPFN Vanilla TabPFN is sensitive to feature ordering across all datasets, with sensitivity decreasing at larger training sizes (Figure A3). Topological ordering yields significant improvements in CMD for 26 out of 40 combinations of dataset and training size (Figure 1), with 3 degradations on CSS at training sizes $N \leq 100$. Datasets CSM and SGL show the greatest improvements in CMD: largest at small training sizes for CSM, and increasing with training size for SGL. The kMTVD and NNAA results are consistent with these trends (Figure A6). Table 2 reports correlation coefficients for variable pairs that should be independent in CSM, confirming that vanilla TabPFN introduces spurious correlations up to $|\rho| \approx 0.15$ while DAG-aware generation does not.

Conversely, reverse topological ordering predominantly degrades performance across metrics (Appendix G). This confirms that column ordering significantly impacts TabPFN's data generation quality.

DAG-Aware Generation Outperforms Vanilla TabPFN DAG-aware generation shows significant improvements in CMD for 24 of 35 configurations (Figure 2, left), compared to vanilla TabPFN with original ordering, with 2 degradations on CSS at $N = 20$ and $N = 100$. Improvements are strongest on CSM, CWA, and CMC, with improvement sizes decreasing at larger training sizes. The kMTVD and NNAA results also show widespread improvements across datasets (Figure A9), with one degradation on CLB in kMTVD. We also compare against five external tabular generators, all trained per dataset, including two causal-aware baselines given the true DAG. On the custom SCM, DAG-aware TabPFN still outperforms all five (Appendix K).

Comparing vanilla TabPFN and DAG-aware generation under the same topological ordering isolates the contribution of causal conditioning from the choice of feature ordering. DAG-aware generation still significantly improves CMD on 20 of 35 configurations with no degradations (Figure 3), while kMTVD and NNAA improve less but rarely degrade (Figure A10).

PDAG-Based Generation Improves Over Vanilla TabPFN When Sufficiently Oriented The oracle-PDAG shows 15 significant CMD improvements and 3 significant degradations (Figure 2, right), two on CLB at $N = 20$ and $N = 50$, and one on CSS at $N = 20$. Conversely, the discovered CPDAG shows no significant improvements and 7 significant degradations in CMD (Figure A11). Three of the seven degradations occur on CMC, where PC orients most discovered edges with low accuracy (Table A2). Most configurations (28 of 35) show no significant difference from vanilla TabPFN. The kMTVD and NNAA results are consistent with these patterns (Figures A12 and A13).

Table 2: Mean Pearson correlation coefficients for independent variable pairs in the custom collider SCM, at the smallest and largest training sizes. Values close to zero indicate correct independence preservation; the best value per column and training size is in **bold**. Standard deviations across 100 repetitions in parentheses. Intermediate training sizes are reported in Table A1.

Method	$\rho(X_0, X_3)$	$\rho(X_0, X_2)$
Train size $N = 20$		
Vanilla original	-0.149 (0.203)	-0.150 (0.202)
Vanilla topological	-0.018 (0.110)	-0.018 (0.109)
Vanilla reverse top.	-0.127 (0.195)	-0.123 (0.197)
DAG-aware	**0.004** (0.021)	**0.004** (0.021)
Oracle-PDAG	-0.018 (0.109)	-0.018 (0.110)
Discovered CPDAG	-0.141 (0.212)	-0.141 (0.210)
Train size $N = 500$		
Vanilla original	-0.028 (0.043)	-0.028 (0.043)
Vanilla topological	-0.003 (0.025)	-0.003 (0.025)
Vanilla reverse top.	-0.033 (0.040)	-0.032 (0.040)
DAG-aware	**0.001** (0.023)	**0.001** (0.023)
Oracle-PDAG	-0.003 (0.025)	-0.003 (0.025)
Discovered CPDAG	-0.013 (0.096)	-0.012 (0.100)
Test set	0.022 –	0.022 –

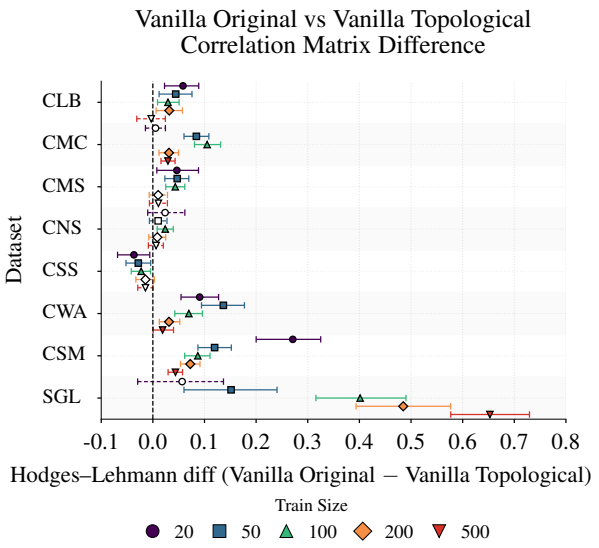

Figure 1: Hodges–Lehmann estimates comparing vanilla TabPFN with original ordering versus topological ordering in CMD. Positive values indicate that topological ordering achieves lower metric values (i.e., better synthetic data quality). Filled markers with solid error bars indicate significance at $p < 0.05$ (Holm correction).

5.2 TREATMENT EFFECT PRESERVATION

Vanilla TabPFN with Topological Ordering Improves ATE Preservation Topological ordering yields 18 significant improvements in ATE preservation across 42 configurations (Figure 4), with 2 degradations, one on CMC and one on CSS, at 20 samples. CSM shows the greatest improvements, particularly at smaller training sizes. On SGL, topological ordering produces 5 significant improvements out of 6 training sizes (Figure A18), with larger effect sizes than other datasets. Reverse topological ordering produces large degradations on CSM (Figure A19).

DAG-Aware Generation Improves ATE Preservation DAG-aware generation shows 26 significant improvements across 42 configurations with 2 degradations, one on CMC and one on CSS, at 20 samples (Figure 5, left), compared to 18 improvements for vanilla topological ordering. CSM and CMS show significant improvements across all training sizes, with the largest reductions at 20 samples. CNS shows significant improvements at most training sizes. Under the same topological ordering, DAG-aware generation still achieves 15 of 42 significant improvements in ATE preservation, with no degradations (Figure A20).

PDAG-Based Generation Improves ATE Preservation When Sufficiently Oriented The oracle-PDAG shows 23 significant improvements across 42 configurations, with one degradation on CMC at $N = 20$ and one on CSS at $N = 200$ (Figure 5, right). Improvements are strongest on CNS (across all training sizes), and on CMS and CSM (at smaller training sizes). The discovered CPDAG shows no consistent effects, with one improvement and one degradation across 42 configurations (Figure A21 and Table A4).

6 DISCUSSION

We demonstrated that vanilla TabPFN, which generates features autoregressively in the input column order, is sensitive to that ordering. When the column order conflicts with the causal structure, the model conditions on descendants or colliders. While this is not a problem in the population limit, for small sample sizes the estimation error in learned conditionals introduces spurious correlations that degrade both synthetic data quality and treatment effect preservation. We addressed this limitation by injecting causal structure into the generation process, using the full DAG when it is available and partial causal knowledge otherwise. For the partial case we evaluated both the oracle-PDAG and CPDAGs discovered from data. DAG-aware generation yields the most consistent improvements, the oracle-PDAG shows moderate gains, and a discovered CPDAG has no consistent effect, with a potential benefit that depends on how well the causal structure is recovered.

Reordering columns topologically improves both distributional quality and treatment effect preservation, while reversing causal direction worsens both. DAG-aware generation conditions each variable only on its causal parents rather than on all previously generated features; this produces higher-quality synthetic data with minimal degradations across datasets and metrics. The single degradation in kMTVD occurs on CLB, whose sparse causal structure (most nodes have a single parent) limits the conditioning context available to the model.

Oracle-PDAGs orienting only edges into colliders show improvements in synthetic data quality and ATE preservation. However, CPDAGs discovered from data often leave many edges unoriented, causing the model to fall back to vanilla sequential conditioning; when the recovered graph is inaccurate, they can even degrade performance. When PC recovers enough correct structure from the data, by contrast, discovered-CPDAG generation improves over vanilla TabPFN (Appendix L). Beyond sufficient edge orientation, the position of colliders within the causal graph also matters. When these are central (as in CMC), generating oriented variables first allows the remaining variables to follow the causal direction, producing improvements. When they are at the end of a causal chain (as in CLB), generating oriented variables first effectively reverses the generation order relative to the causal direction, leading to degradations.

Causal conditioning preserves causal effects better than

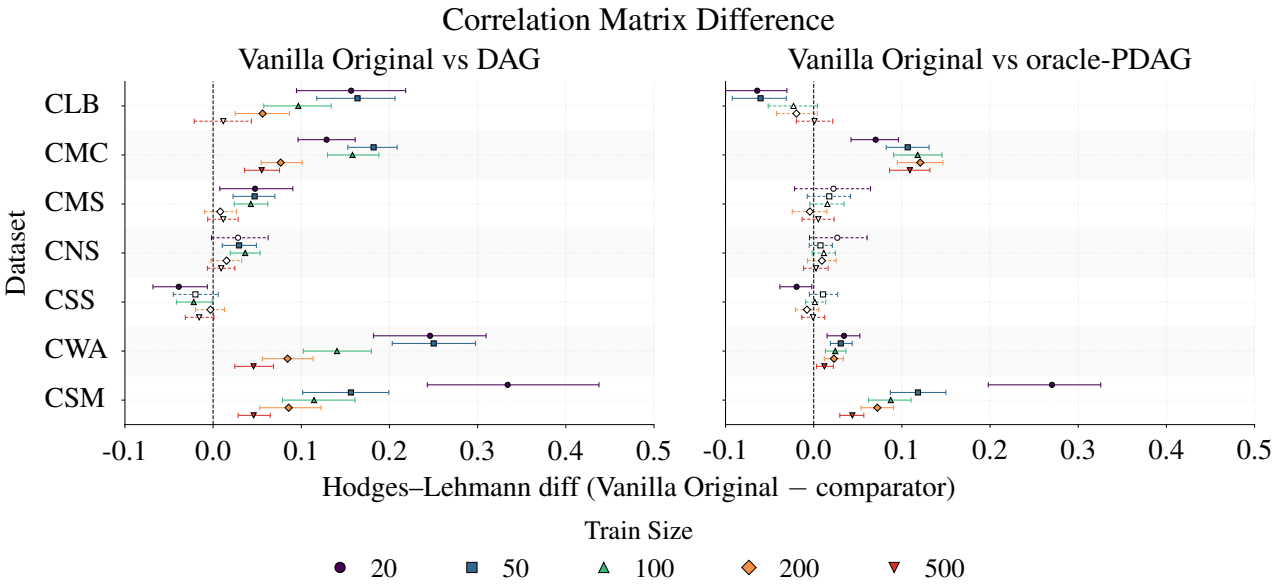

Figure 2: Hodges–Lehmann estimates comparing vanilla TabPFN with original ordering versus DAG-aware generation (left) and versus the oracle-PDAG (right) in CMD. Positive values indicate that the respective method achieves lower metric values (i.e., better synthetic data quality). Filled markers with solid error bars indicate significance at $p < 0.05$ (Holm correction).

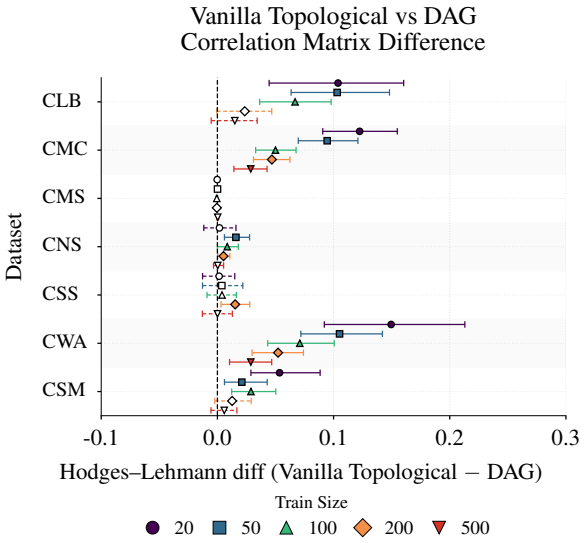

Figure 3: Hodges–Lehmann estimates comparing DAG-aware generation and vanilla TabPFN when both use topological ordering, in CMD. With feature ordering held fixed, this isolates the contribution of parent-based conditioning from the ordering itself. Positive values indicate that DAG-aware generation achieves lower metric values (i.e., better synthetic data quality). Filled markers with solid error bars indicate significance at $p < 0.05$ (Holm correction).

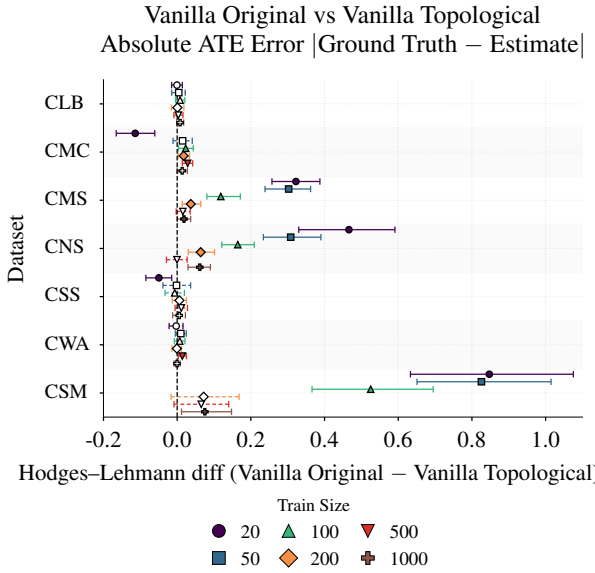

Figure 4: Hodges–Lehmann estimates of the reduction in absolute ATE error (Δ_{ATE}) when comparing vanilla TabPFN with original ordering versus topological ordering on CSuite and CSM datasets. Positive values indicate smaller errors (closer to ground truth) for topological ordering; negative values indicate larger errors. Filled markers with solid error bars indicate significance at $p < 0.05$ (Holm correction).

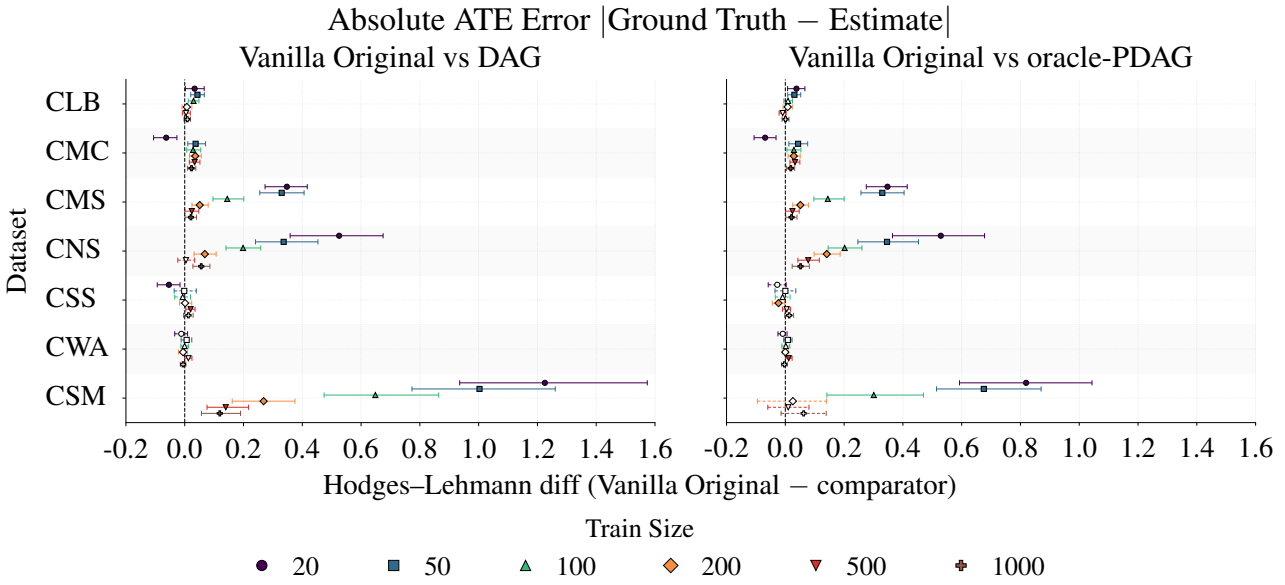

Figure 5: Hodges–Lehmann estimates of the reduction in absolute ATE error (Δ_{ATE}) when comparing vanilla TabPFN with original ordering versus DAG-aware generation (left) and versus the oracle-PDAG (right). Positive values indicate smaller errors (closer to ground truth) for the respective method (DAG-aware or oracle-PDAG); negative values indicate larger errors. Filled markers with solid error bars indicate significance at $p < 0.05$ (Holm correction).

vanilla TabPFN, particularly with limited training data. Under interventions, it also preserves more of the real data's empirical conditional independencies (Appendix R). On CSM, DAG-aware generation reduces the median absolute ATE error from 1.23 for vanilla TabPFN to 0.186 units at 20 samples, with improvements holding at larger training sizes. Table A5 reports the absolute ATE scale in native units for every dataset and training size. Since the CSM is near-deterministic ($\sigma = 10^{-5}$), we verify that the benefits of causal conditioning hold under higher noise ($\sigma = 10^{-2}$, Appendix S). The CMC degradation at $N = 20$ in the ATE experiments also occurs with vanilla TabPFN under topological ordering, which does not modify the generation mechanism. This indicates that the effect is a model property rather than a consequence of causal conditioning.

Our findings extend to SGL, a realistic dataset derived from an FDA-approved T1DM simulator whose partial causal knowledge comes from validated physiological mechanisms. On this dataset, partial topological ordering improves synthetic data quality (Figure 1). Unlike the other datasets, these improvements increase with training size, which we attribute to the interplay between sample size and data dimensionality: with 38 variables, the model needs sufficient context to benefit from correct ordering. In the smaller SCMs (4 to 12 variables), the model already benefits from correct ordering at small N, but this advantage diminishes as data increases, because vanilla TabPFN can then estimate the relevant conditionals well enough on its own. On SGL, the same ordering also better preserves treatment effects (Figure A18), which highlights the relevance to real-world medical applications, where accurate treatment effect preservation from synthetic data can support early-stage research while reducing the need to share sensitive patient data.

6.1 LIMITATIONS AND FUTURE WORK

This work is subject to the following limitations. First, DAG-aware generation assumes full knowledge of the causal structure, which is rarely available in practice. PDAG-based generation shows that partial causal information can still be valuable, but its effectiveness depends on collider placement in ways that cannot be predicted without knowledge of the true graph. Second, causal discovery relied solely on PC-stable. A preliminary evaluation with ReX [Renero et al., 2026], which returns fully oriented DAGs, yielded worse synthetic data quality since incorrect orientations propagate into the conditioning strategy (Appendix T). Third, CMD assumes monotonic dependencies and is uninformative for SCMs with multiplicative equations, as in CSS. Fourth, the interventional analysis focused exclusively on the ATE; other causal estimands or downstream tasks may respond differently to causal conditioning. Finally, our study focused primarily on TabPFN. Future work should explore ordering strategies for incomplete causal knowledge, evaluate a broader range of causal discovery algorithms, select evaluation metrics based on dependency structure, and test these findings on other neural autoregressive architectures, such as TabularARGN [Tiwald et al., 2025].

Acknowledgements

We thank the anonymous reviewers for their constructive comments, which helped improve the paper, and Ameen Abu-Hanna, Otto Nyberg, and Juliette Ortholand for their feedback on an earlier version of this manuscript. We acknowledge ISCRA for awarding this project access to the LEONARDO supercomputer, owned by the EuroHPC Joint Undertaking, hosted by CINECA (Italy). We also thank the University of Trieste for providing additional computational resources.

References

Vahid Balazadeh, Hamidreza Kamkari, Valentin Thomas, Junwei Ma, Bingru Li, Jesse C Cresswell, and Rahul Krishnan. CausalPFN: Amortized causal effect estimation via in-context learning. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*, 2025.

Vadim Borisov, Kathrin Seßler, Tobias Leemann, Martin Pawelczyk, and Gjergji Kasneci. Language models are realistic tabular data generators. In *The Eleventh International Conference on Learning Representations*, 2023.

David Maxwell Chickering. Optimal structure identification with greedy search. *Journal of Machine Learning Research*, 3:507–554, 2002.

Diego Colombo and Marloes H Maathuis. Order-independent constraint-based causal structure learning. *Journal of Machine Learning Research*, 15(116):3921–3962, 2014.

Chiara Dalla Man, Marc D. Breton, and Claudio Cobelli. Physical activity into the meal glucose—insulin model of type 1 diabetes: In silico studies. *Journal of Diabetes Science and Technology*, 3(1):56–67, 2009. doi: 10.1177/193229680900300107. URL https://doi.org/10.1177/193229680900300107. PMID: 20046650.

Anish Dhir, Cristiana Diaconu, Valentinian Mihai Lungu, James Requeima, Richard E Turner, and Mark van der Wilk. Estimating interventional distributions with uncertain causal graphs through meta-learning. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*, 2025.

Ana Esponera and Giovanni Cinà. Inducing causal structure applied to glucose prediction for T1DM patients. In Biwei Huang and Mathias Drton, editors, *Proceedings of the Fourth Conference on Causal Learning and Reasoning*, volume 275 of *Proceedings of Machine Learning Research*, pages 920–946. PMLR, May 2025. URL https://proceedings.mlr.press/v275/esponera25a.html. 07–09 May.

Tomas Geffner, Javier Antoran, Adam Foster, Wenbo Gong, Chao Ma, Emre Kiciman, Amit Sharma, Angus Lamb, Martin Kukla, Nick Pawlowski, et al. Deep end-to-end causal inference. *Transactions on Machine Learning Research*, 2024.

Joseph L Hodges Jr and Erich L Lehmann. Estimates of Location Based on Rank Tests. *The Annals of Mathematical Statistics*, 34(2):598–611, 1963. doi: 10.1214/aoms/1177704172. URL https://doi.org/10.1214/aoms/1177704172.

Noah Hollmann, Samuel Müller, Katharina Eggensperger, and Frank Hutter. TabPFN: A transformer that solves small tabular classification problems in a second. In *The Eleventh International Conference on Learning Representations*, 2023.

Noah Hollmann, Samuel Müller, Lennart Purucker, Arjun Krishnakumar, Max Körfer, Shi Bin Hoo, Robin Tibor Schirrmeister, and Frank Hutter. Accurate predictions on small data with a tabular foundation model. *Nature*, 637 (8045):319–326, 2025.

Akim Kotelnikov, Dmitry Baranchuk, Ivan Rubachev, and Artem Babenko. TabDDPM: Modelling tabular data with diffusion models. In *International Conference on Machine Learning*, pages 17564–17579. PMLR, 2023.

Anton D Lautrup, Tobias Hyrup, Arthur Zimek, and Peter Schneider-Kamp. SynthEval: a framework for detailed utility and privacy evaluation of tabular synthetic data. *Data Mining and Knowledge Discovery*, 39(1):6, 2025.

Gael Lederrey, Tim Hillel, and Michel Bierlaire. DATGAN: Integrating expert knowledge into deep learning for synthetic tabular data. *arXiv preprint arXiv:2203.03489*, 2022.

Junwei Ma, Apoorv Dankar, George Stein, Guangwei Yu, and Anthony Caterini. TabPFGen–Tabular data generation with TabPFN. *arXiv preprint arXiv:2406.05216*, 2024.

Christopher Meek. Causal inference and causal explanation with background knowledge. In *Proceedings of the Eleventh Conference on Uncertainty in Artificial Intelligence (UAI)*, pages 403–410, 1995.

Tu Anh Hoang Nguyen, Dang Nguyen, Tri-Nhan Vo, Thuc Duy Le, and Sunil Gupta. Causal-aware generative adversarial networks with reinforcement learning. *arXiv preprint arXiv:2510.24046*, 2025.

Judea Pearl. *Causality: Models, Reasoning, and Inference*. Cambridge University Press, Cambridge, UK, 2nd edition, 2009. ISBN 978-0521895606.

Jerome P Reiter. Using CART to generate partially synthetic public use microdata. *Journal of Official Statistics*, 21(3): 441–462, 2005.

Jesús Renero, Roberto Maestre, and Idoia Ochoa. ReX: Causal discovery based on machine learning and explainability techniques. *Pattern Recognition*, 172:112491, 2026.

Diane Ridgeway, Mary F Theofanos, Terese W Manley, and Christine Task. Challenge design and lessons learned from the 2018 differential privacy challenges, 2021.

Jake Robertson, Arik Reuter, Siyuan Guo, Noah Hollmann, Frank Hutter, and Bernhard Schölkopf. Do-PFN: In-context learning for causal effect estimation. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*, 2025.

Juntong Shi, Minkai Xu, Harper Hua, Hengrui Zhang, Stefano Ermon, and Jure Leskovec. TabDiff: a mixed-type diffusion model for tabular data generation. In *The Thirteenth International Conference on Learning Representations*, 2025a.

Ruxue Shi, Yili Wang, Mengnan Du, Xu Shen, Yi Chang, and Xin Wang. A comprehensive survey of synthetic tabular data generation. *arXiv preprint arXiv:2504.16506*, 2025b.

Paul Tiwald, Ivona Krchova, Andrey Sidorenko, Mariana Vargas Vieyra, Mario Scriminaci, and Michael Platzer. TabularARGN: A flexible and efficient auto-regressive framework for generating high-fidelity synthetic data. *arXiv preprint arXiv:2501.12012*, 2025.

Boris Van Breugel, Trent Kyono, Jeroen Berrevoets, and Mihaela Van der Schaar. DECAF: Generating fair synthetic data using causally-aware generative networks. *Advances in Neural Information Processing Systems*, 34: 22221–22233, 2021.

Bingyang Wen, Yupeng Cao, Fan Yang, Koduvayur Subbalakshmi, and Rajarathnam Chandramouli. Causal-TGAN: Modeling tabular data using causally-aware GAN. In *ICLR 2022 Workshop on Deep Generative Models for Highly Structured Data*, 2022.

Lei Xu and Kalyan Veeramachaneni. Synthesizing tabular data using generative adversarial networks. *arXiv preprint arXiv:1811.11264*, 2018.

Lei Xu, Maria Skoularidou, Alfredo Cuesta-Infante, and Kalyan Veeramachaneni. Modeling tabular data using conditional GAN. *Advances in Neural Information Processing Systems*, 32, 2019.

Andrew Yale, Saloni Dash, Ritik Dutta, Isabelle Guyon, Adrien Pavao, and Kristin P Bennett. Generation and evaluation of privacy preserving synthetic health data. *Neurocomputing*, 416:244–255, 2020.

Hengrui Zhang, Jiani Zhang, Zhengyuan Shen, Balasubramaniam Srinivasan, Xiao Qin, Christos Faloutsos, Huzefa Rangwala, and George Karypis. Mixed-type tabular data synthesis with score-based diffusion in latent space. In *The Twelfth International Conference on Learning Representations*, 2024.

Jia-Chen Zhang, Zheng Zhou, Yu-Jie Xiong, Chun-Ming Xia, and Fei Dai. CausalDiffTab: Mixed-type causal-aware diffusion for tabular data generation. *arXiv preprint arXiv:2506.14206*, 2025.

Zilong Zhao, Aditya Kunar, Robert Birke, and Lydia Y Chen. CTAB-GAN: Effective table data synthesizing. In *Asian Conference on Machine Learning*, pages 97–112. PMLR, 2021.

Improving TabPFN's Synthetic Data Generation by Integrating Causal Structure (Supplementary Material)

Davide Tugnoli[1,2] **Andrea De Lorenzo**[3] **Marco Virgolin**[4] **Giovanni Cinà**[5,6]

[1]Department of Mathematics, Informatics and Geosciences, University of Trieste, Trieste, Italy
[2]InSilicoTrials Technologies, Riva Grumula 2, 34123 Trieste, Italy
[3]Department of Engineering and Architecture, University of Trieste, Trieste, Italy
[4]InSilicoTrials Technologies BV, The Netherlands
[5]Medical Informatics Dept., Amsterdam UMC, The Netherlands
[6]Institute for Logic, Language and Computation, University of Amsterdam, The Netherlands

A NEAREST-NEIGHBOR ADVERSARIAL ACCURACY

The NNAA [Yale et al., 2020] is a nearest-neighbor metric used in privacy-oriented evaluations of synthetic data. It quantifies the distinguishability between synthetic and real data based on nearest-neighbor distances. By construction, unlike the CMD, which compares pairwise association, and the kMTVD with $k = 2$, which compares two-way joint distributions, the NNAA assesses similarity at the level of individual records in the full joint space, capturing aspects of fidelity that neither pairwise metric can detect. We use SynthEval's implementation with the Gower distance [Lautrup et al., 2025]. For each real sample i, the distance $d_{TS}(i)$ to its nearest synthetic neighbor is compared with the distance $d_{TT}(i)$ to its nearest other real sample. Similarly, for each synthetic sample j, the distance $d_{ST}(j)$ to its nearest real neighbor is compared with the distance $d_{SS}(j)$ to its nearest other synthetic sample. The adversarial accuracy is

$$\begin{aligned}\mathrm{NNAA} = \frac{1}{2}\Bigg[\frac{1}{n}\sum_{i=1}^{n}\mathbf{1}\big(d_{TS}(i) > d_{TT}(i)\big) \\ +\frac{1}{n}\sum_{j=1}^{n}\mathbf{1}\big(d_{ST}(j) > d_{SS}(j)\big)\Bigg],\end{aligned} \tag{4}$$

where $\mathbf{1}(\cdot)$ denotes the indicator function, and n is the number of samples in both the real and synthetic datasets. Values near 0.5 indicate that synthetic and real data are hard to distinguish. A value above 0.5 means they are easy to tell apart. A value below 0.5 means the synthetic samples are unusually close to the real ones. Since our interest is indistinguishability, we treat both sides equally and report each result as the distance from this ideal value, $|\mathrm{NNAA} - 0.5|$. Lower values mean greater indistinguishability. The raw NNAA values are available in the code repository for analyses that separate the two cases. In our experiments, the median NNAA lies above 0.5 in every dataset and generation method. Because we measure NNAA against a held-out test set that the model never sees during generation, this closeness reflects fidelity to the unseen real distribution.

However, the model does see the training records. We therefore ask whether causal conditioning brings the synthetic data closer to them than vanilla TabPFN does. To test this, we compute the same NNAA against the training set. The difference between the two values is the privacy loss [Yale et al., 2020]. A value near 0 means the synthetic data are similarly close to the training and held-out records, while positive values indicate greater proximity to the training records. Figures A1 and A2 report it as vanilla TabPFN minus each method. DAG-aware generation lowers it in most settings and never significantly increases it. The oracle-PDAG also lowers it in many settings, though less consistently. The discovered CPDAG shows no clear effect.

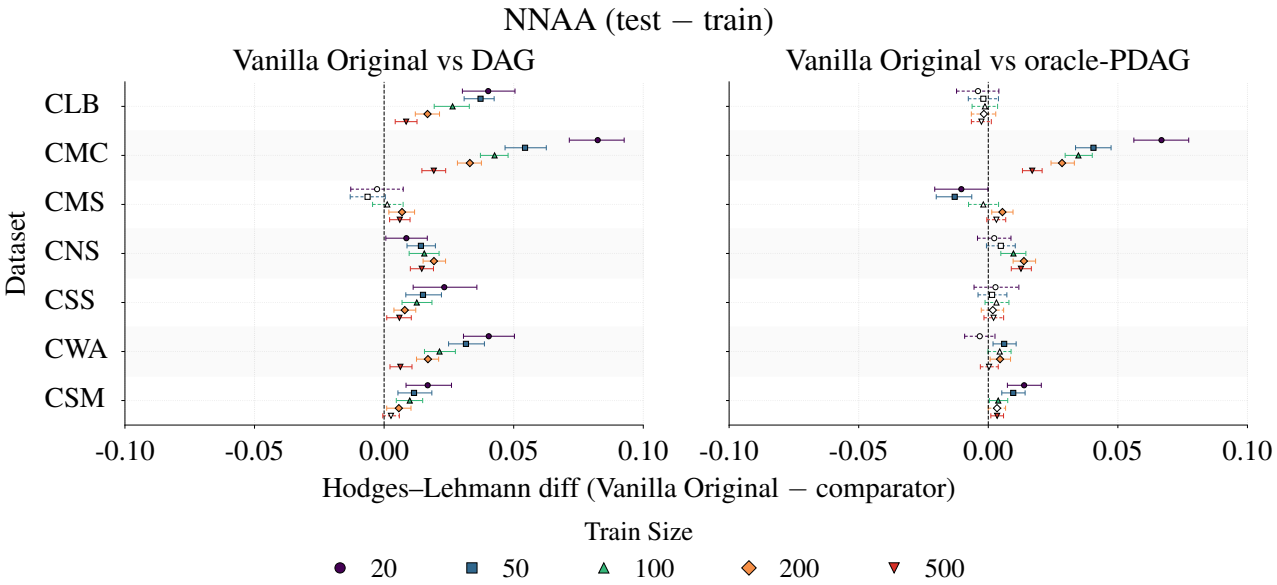

Figure A1: Hodges–Lehmann estimates comparing vanilla TabPFN with DAG-aware generation (left) and the oracle-PDAG (right) in the privacy loss (the difference between NNAA on the held-out test set and NNAA on the training set). Positive values indicate that the method's synthetic data are no closer to the training records than vanilla's. Filled markers with solid error bars indicate significance at $p < 0.05$ (Holm correction).

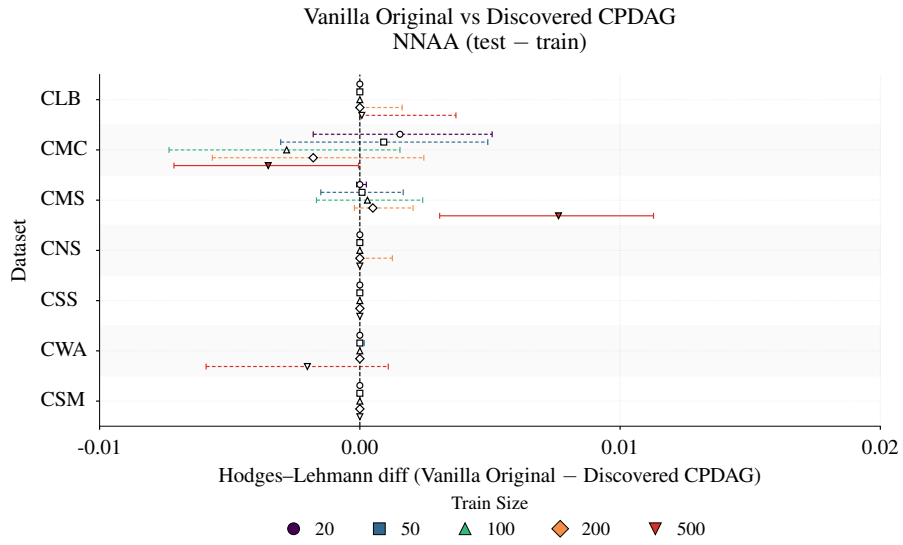

Figure A2: Hodges–Lehmann estimates comparing vanilla TabPFN with the discovered CPDAG in the privacy loss (the difference between NNAA on the held-out test set and NNAA on the training set). Positive values indicate that the synthetic data are no closer to the training records than vanilla's. Filled markers with solid error bars indicate significance at $p < 0.05$ (Holm correction).

B CUSTOM SCM DEFINITION

The custom collider SCM has causal structure $X_0 \to X_1 \leftarrow X_2 \leftarrow X_3$, with structural equations:

$$X_0 \sim \mathcal{N}(0, 1) \tag{5}$$
$$X_3 \sim \mathcal{N}(0, 1) \tag{6}$$
$$X_2 = 0.5\, X_3 + \epsilon_2, \quad \epsilon_2 \sim \mathcal{N}(0, \sigma^2) \tag{7}$$
$$X_1 = 5.0\, X_0 + 10.0\, X_2 + \epsilon_1, \quad \epsilon_1 \sim \mathcal{N}(0, \sigma^2) \tag{8}$$

where $\sigma = 10^{-5}$. Variables X_0 and X_3 are independent root nodes, and X_0 and X_2 are d-separated since X_2 depends only on X_3.

C ORDER SENSITIVITY ACROSS DATASETS

To quantify how much feature ordering affects synthetic data quality, we measure the variability of each metric across the three orderings (original, topological, reverse topological).

For each dataset and training size, we compute:

- The metric value under each of the three orderings
- The range: maximum value minus minimum value
- Bootstrap confidence intervals for the median range (1000 resamples, $\alpha = 0.05$)

A large range indicates that the model is highly sensitive to feature ordering. A small range indicates consistent performance regardless of ordering.

Figure A3 shows results for CMC. This dataset exemplifies a general pattern observed across all evaluated datasets: ordering sensitivity decreases with training size but persists even at larger sample sizes. Results for additional datasets are available in the code repository.

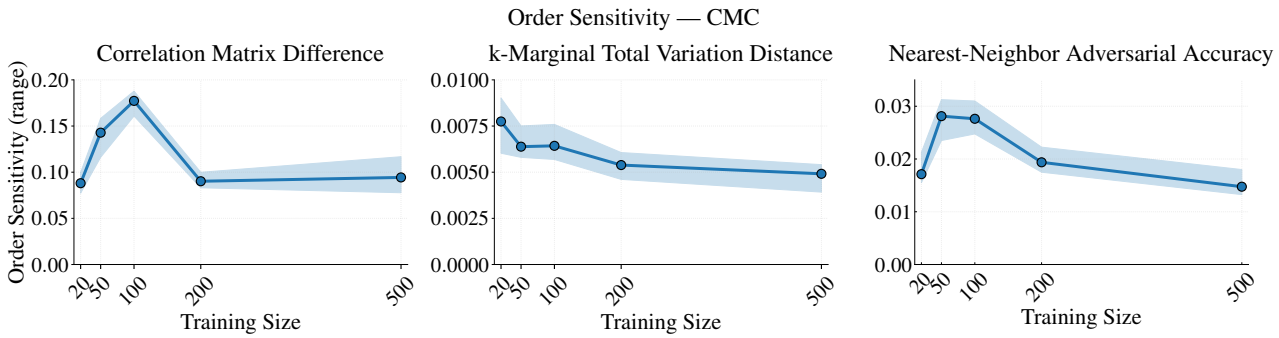

Figure A3: Order sensitivity for vanilla TabPFN on CMC. Y-axis shows the range of CMD (left), kMTVD ($k = 2$, center), and NNAA (right) values across three feature orderings: original, topological, and reverse topological. Shaded regions show 95 % bootstrap confidence intervals.

D RANDOM-ORDER SENSITIVITY OF THE VANILLA BASELINE

Throughout the paper we use each dataset's original column ordering as the vanilla baseline, since it matches the common use of TabPFN for synthetic data generation, where the practitioner feeds the data as-is without a causally informed ordering. As an additional check that our conclusions do not depend on this particular ordering, we also compare against vanilla TabPFN under random column orderings. For each dataset we sample a fixed pool of 10 random permutations and cyclically assign one to each of the 100 paired repetitions. Vanilla TabPFN generates with the assigned ordering on the same cached

training split, and the generated data are evaluated with the paper's paired protocol (Wilcoxon signed-rank with Pratt ties and Holm correction).

Figure A4 compares DAG-aware generation against vanilla TabPFN with random orderings on the seven datasets of the main comparison (35 dataset–size combinations). DAG-aware generation achieves 26 significant improvements and no degradations in CMD, and 31 improvements with 1 degradation (CLB at $N = 500$) in kMTVD. In NNAA, it achieves 28 improvements with no degradations (Figure A5). These results confirm that the advantage of DAG-aware generation holds whether the vanilla baseline uses the original ordering or random ones.

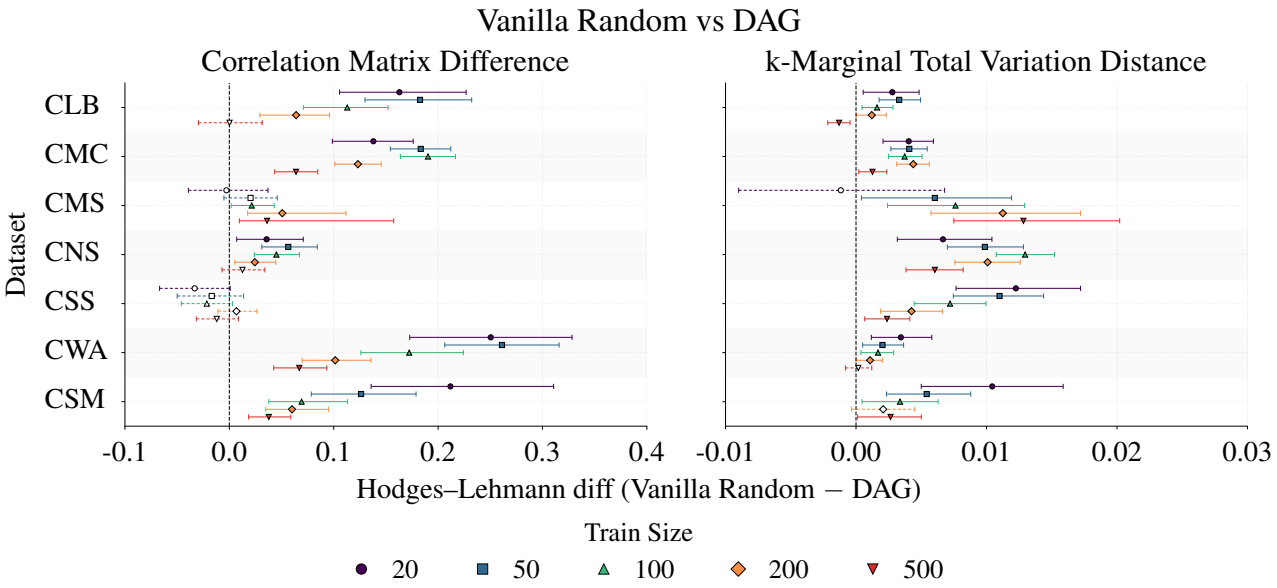

Figure A4: Hodges–Lehmann estimates comparing vanilla TabPFN with random column orderings and DAG-aware generation with topological ordering, in CMD (left) and kMTVD ($k = 2$, right). Positive values indicate that DAG-aware generation achieves lower metric values (i.e., better synthetic data quality). Filled markers with solid error bars indicate significance at $p < 0.05$ (Holm correction).

E VANILLA TABPFN WITH TOPOLOGICAL ORDERING RESULTS

Figure A6 reports kMTVD and NNAA results for vanilla TabPFN with topological ordering. In kMTVD, topological ordering yields 30 significant improvements and 1 degradation on CLB at 500 samples (left panel). In NNAA, it yields 27 significant improvements with no degradations (right panel).

F SPURIOUS CORRELATIONS IN CUSTOM SCM

Table A1 reports Pearson correlation coefficients for variable pairs that should be independent according to the custom SCM causal structure (Appendix B). The test set baseline confirms the expected independence with correlations close to zero ($\rho \approx 0.02$). Here we report the intermediate training sizes $N \in \{50, 100, 200\}$; $N = 20, 500$ are in the main text (Table 2). Vanilla TabPFN with original and reverse topological orderings introduces substantial spurious correlations, particularly at small training sizes ($|\rho| \approx 0.15$ at $N = 20$). DAG-aware generation maintains near-zero correlations across all training sizes. Oracle-PDAG performs similarly to vanilla topological, while discovered CPDAG tracks vanilla original at the smallest training size (Table 2) and shows attenuated but more variable correlations at the sizes reported here, reflecting how undirected edges are resolved based on column order.

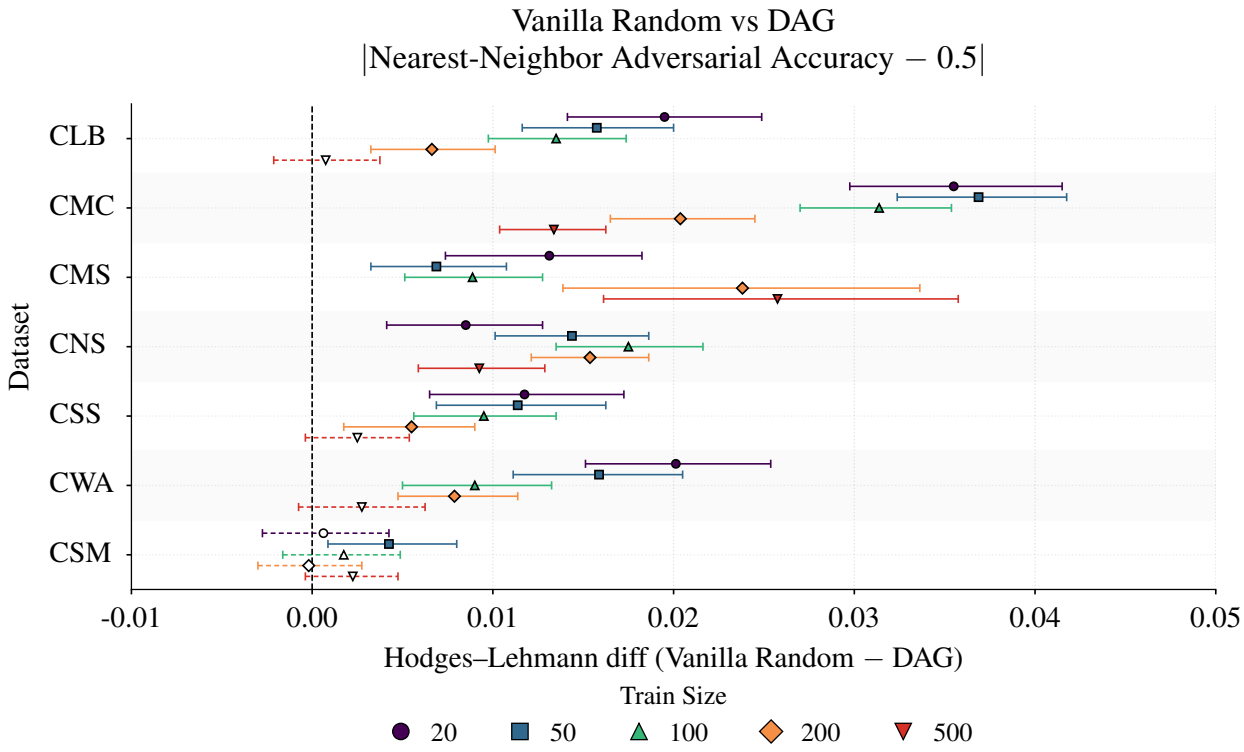

Figure A5: Hodges–Lehmann estimates comparing vanilla TabPFN with random column orderings and DAG-aware generation with topological ordering, in NNAA (reported as the distance from the ideal value, $|\text{NNAA} - 0.5|$). Positive values indicate that DAG-aware generation achieves lower values (i.e., greater indistinguishability between synthetic and real data). Filled markers with solid error bars indicate significance at $p < 0.05$ (Holm correction).

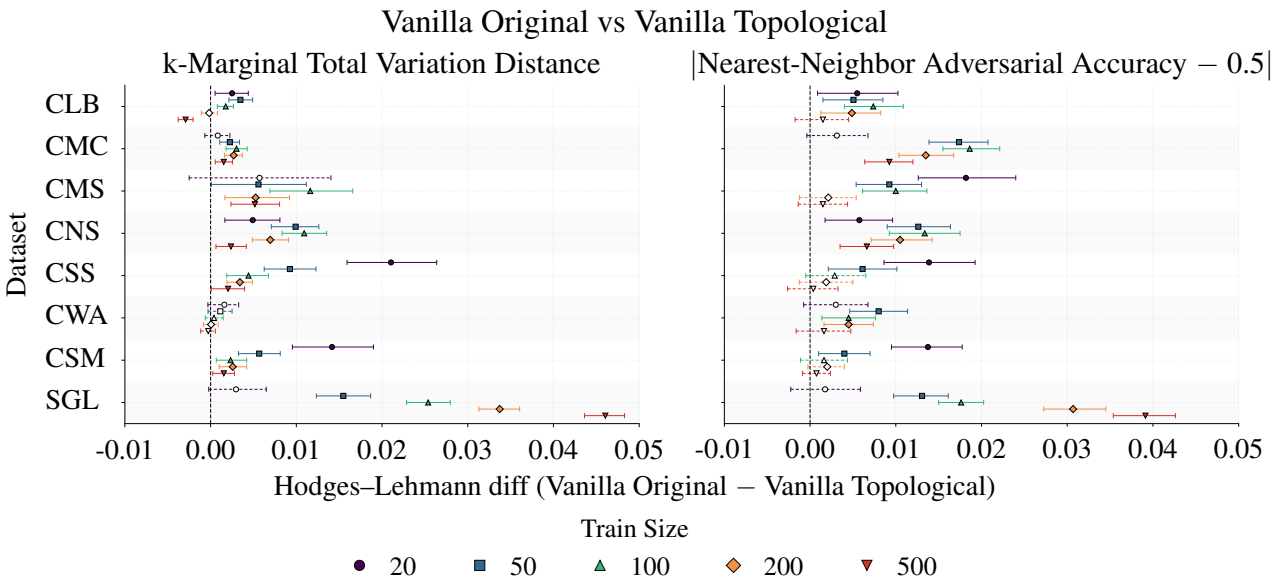

Figure A6: Hodges–Lehmann estimates comparing vanilla TabPFN with original ordering versus topological ordering in kMTVD ($k = 2$, left) and NNAA (right), reported as the distance from the ideal value, $|\text{NNAA} - 0.5|$. Positive values indicate that topological ordering achieves lower kMTVD (i.e., better pairwise fidelity) and lower $|\text{NNAA} - 0.5|$ (i.e., greater indistinguishability between synthetic and real data). Filled markers with solid error bars indicate significance at $p < 0.05$ (Holm correction).

Table A1: Mean Pearson correlation coefficients for independent variable pairs in the custom collider SCM. Values close to zero indicate correct independence preservation; the best value per column and training size is in **bold**. Standard deviations across 100 repetitions in parentheses.

Method	$\rho(X_0, X_3)$		$\rho(X_0, X_2)$	
Train size $N = 50$				
Vanilla original	−0.023	(0.143)	−0.025	(0.143)
Vanilla topological	0.005	(0.075)	0.006	(0.076)
Vanilla reverse top.	−0.030	(0.148)	−0.031	(0.146)
DAG-aware	**0.000**	(0.023)	**0.000**	(0.024)
Oracle-PDAG	0.006	(0.077)	0.005	(0.076)
Discovered CPDAG	−0.001	(0.178)	−0.002	(0.180)
Train size $N = 100$				
Vanilla original	−0.042	(0.114)	−0.043	(0.114)
Vanilla topological	−0.017	(0.064)	−0.017	(0.063)
Vanilla reverse top.	−0.047	(0.101)	−0.045	(0.101)
DAG-aware	**0.000**	(0.020)	**0.000**	(0.020)
Oracle-PDAG	−0.017	(0.063)	−0.017	(0.064)
Discovered CPDAG	−0.029	(0.140)	−0.029	(0.141)
Train size $N = 200$				
Vanilla original	−0.029	(0.104)	−0.035	(0.080)
Vanilla topological	−0.012	(0.042)	−0.012	(0.042)
Vanilla reverse top.	−0.036	(0.082)	−0.028	(0.112)
DAG-aware	**−0.001**	(0.025)	**−0.001**	(0.025)
Oracle-PDAG	−0.006	(0.077)	−0.012	(0.042)
Discovered CPDAG	**−0.001**	(0.152)	−0.005	(0.147)
Test set	0.022	–	0.022	–

G VANILLA TABPFN WITH REVERSE TOPOLOGICAL ORDERING RESULTS

This section reports results for vanilla TabPFN with reverse topological ordering. Reverse topological ordering shows predominantly negative effects across metrics, with 4 significant degradations in CMD and 17 in kMTVD (Figure A7, left and right, respectively), against 4 significant improvements in CMD and 1 in kMTVD. NNAA confirms this pattern with 10 significant degradations and 1 significant improvement (Figure A8).

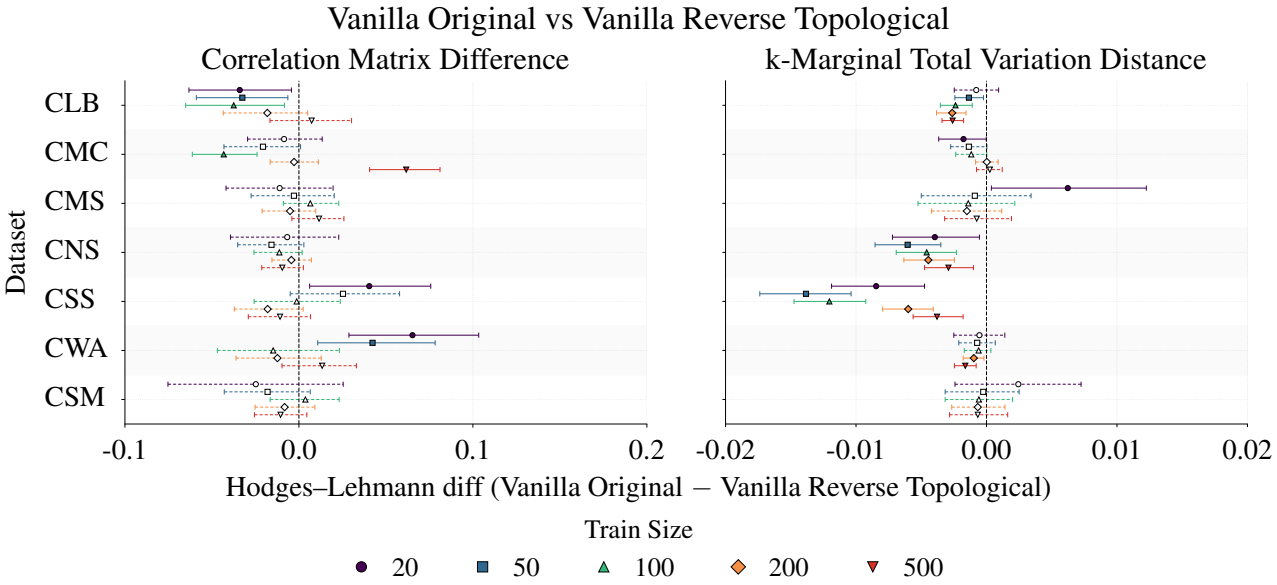

Figure A7: Hodges–Lehmann estimates comparing vanilla TabPFN with original ordering versus reverse topological ordering in CMD (left) and kMTVD ($k = 2$, right). Positive values indicate that reverse topological ordering achieves lower metric values (i.e., better synthetic data quality). Filled markers with solid error bars indicate significance at $p < 0.05$ (Holm correction).

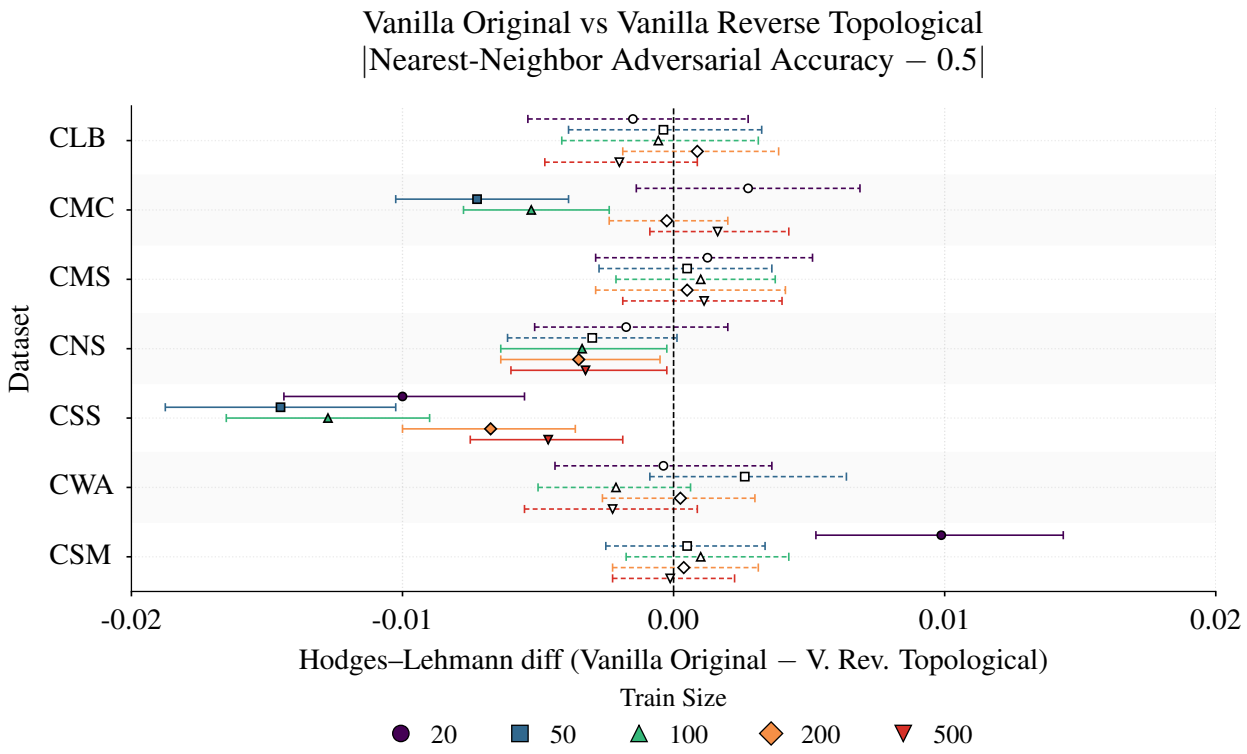

Figure A8: Hodges–Lehmann estimates comparing vanilla TabPFN with original ordering versus reverse topological ordering in NNAA (reported as the distance from the ideal value, $|\mathrm{NNAA} - 0.5|$). Positive values indicate that reverse topological ordering achieves lower values (i.e., greater indistinguishability between synthetic and real data). Filled markers with solid error bars indicate significance at $p < 0.05$ (Holm correction).

H DAG-AWARE GENERATION VS VANILLA: kMTVD AND NNAA RESULTS

Figure A9 reports kMTVD and NNAA results comparing vanilla TabPFN with original ordering versus DAG-aware generation. In kMTVD, DAG-aware generation yields 30 significant improvements and 1 degradation on CLB at 500 samples (left panel). In NNAA, it yields 29 significant improvements with no degradations (right panel).

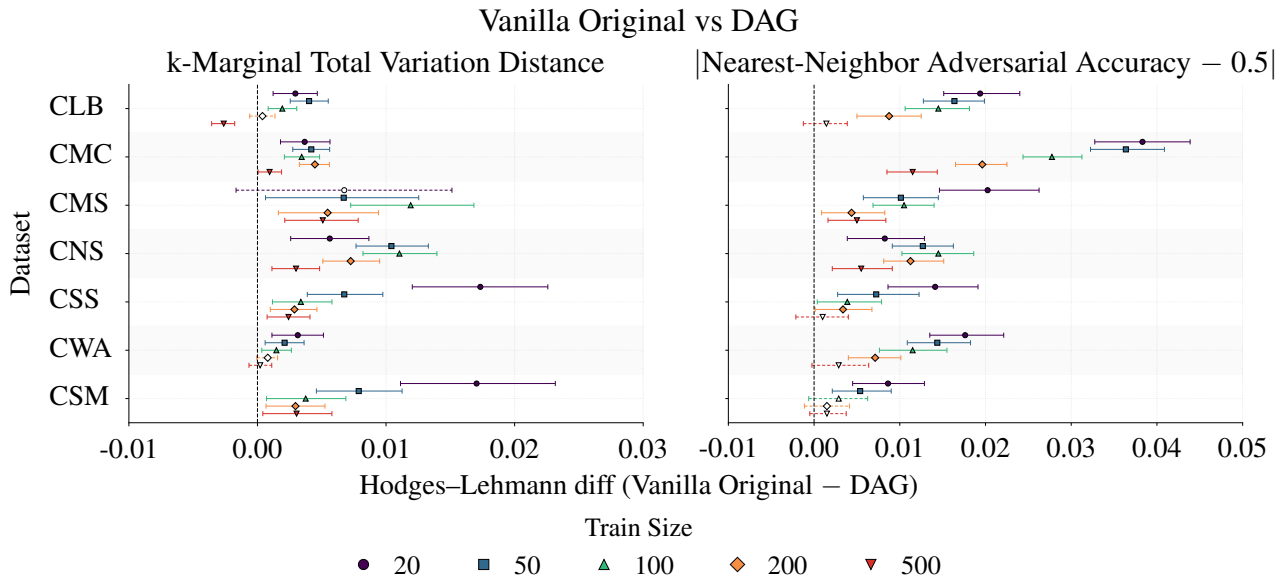

Figure A9: Hodges–Lehmann estimates comparing vanilla TabPFN with original ordering versus DAG-aware generation in kMTVD ($k = 2$, left) and NNAA (right), reported as the distance from the ideal value, $|\text{NNAA} - 0.5|$. Positive values indicate that DAG-aware generation achieves lower kMTVD (i.e., better pairwise fidelity) and lower $|\text{NNAA} - 0.5|$ (i.e., greater indistinguishability between synthetic and real data). Filled markers with solid error bars indicate significance at $p < 0.05$ (Holm correction).

I DAG-AWARE GENERATION VS VANILLA UNDER TOPOLOGICAL ORDERING

We compare DAG-aware generation and vanilla TabPFN when both use topological ordering, isolating the contribution of causal parent-based conditioning from the feature ordering itself.

DAG-aware generation shows 10 improvements and 2 degradations in kMTVD and 14 improvements and 1 degradation in NNAA (Figure A10).

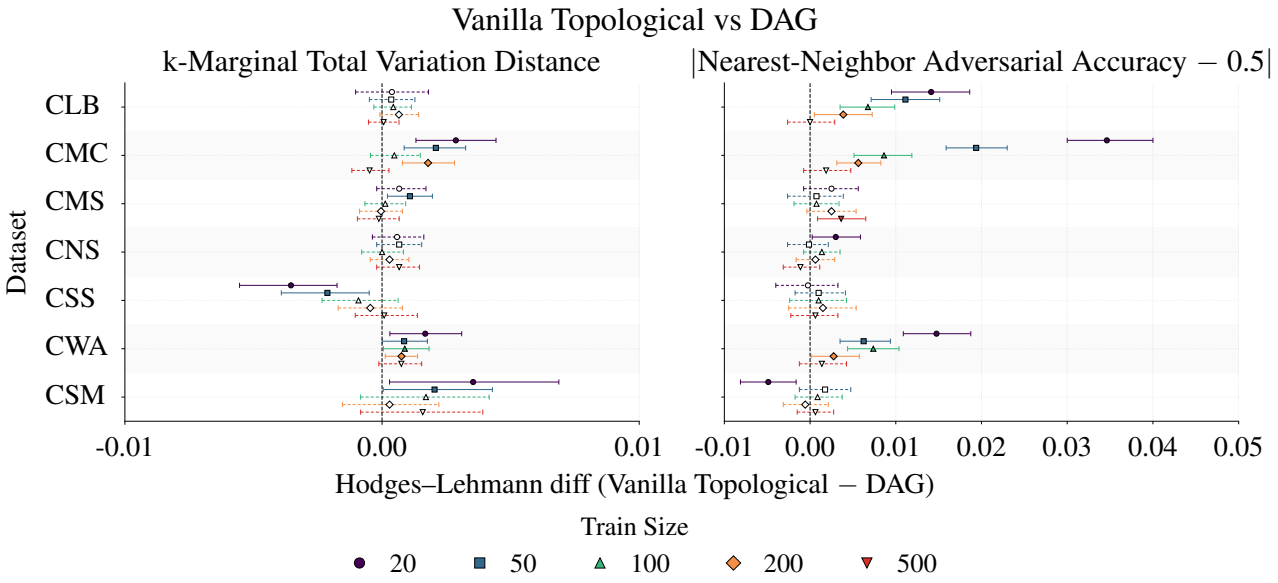

Figure A10: Hodges–Lehmann estimates comparing DAG-aware generation and vanilla TabPFN, both using topological ordering, in kMTVD ($k = 2$, left) and NNAA (right), reported as the distance from the ideal value, $|\mathrm{NNAA} - 0.5|$. Positive values indicate that DAG-aware generation achieves lower kMTVD (i.e., better pairwise fidelity) and lower $|\mathrm{NNAA} - 0.5|$ (i.e., greater indistinguishability between synthetic and real data). Filled markers with solid error bars indicate significance at $p < 0.05$ (Holm correction).

J PDAG-BASED GENERATION: CMD, kMTVD, AND NNAA RESULTS

Figure A11 reports CMD results for discovered CPDAG, showing 7 degradations with no improvements. Results for oracle-PDAG in CMD appear in Figure 2. Figure A12 reports kMTVD results for PDAG-based generation. In kMTVD, oracle-PDAG shows 19 significant improvements and 4 degradations, all on CLB; discovered CPDAG shows 1 improvement and 6 degradations. The single improvement occurs on CMS at $N = 500$, where PC orients $52\,\%$ of discovered edges with 0.67 direction precision (Table A2). Three of the six degradations occur on CMC, where PC orients most discovered edges with low accuracy. Figure A13 reports NNAA results. In NNAA, oracle-PDAG shows 19 improvements with no degradations; discovered CPDAG shows no improvements and 8 degradations.

Table A2 reports graph discovery quality for the PC-stable algorithm. Direction recall does not exceed 0.32 across all datasets and training sizes. On most datasets, PC leaves the majority of edges undirected, causing the hybrid strategy to fall back to vanilla conditioning. On CMC, PC orients up to $86\,\%$ of discovered edges, but direction precision drops to 0.19 at $N = 500$, leading the strategy to condition on incorrect parents and producing the degradations observed in the main results.

Table A2: PC-stable graph discovery quality on CSuite datasets and the custom SCM, averaged over 100 repetitions. Skeleton recall measures the fraction of true edges recovered. Direction recall measures the fraction of true edge orientations correctly identified. Oriented fraction reports the proportion of discovered edges that PC orients. Direction precision measures the fraction of oriented edges with correct orientation (– indicates no oriented edges).

Dataset	N	Skel. rec.	Dir. rec.	Orient. frac.	Dir. prec.
CSM	20	0.66	0.00	0.02	0.00
	50	0.68	0.00	0.05	0.00
	100	0.68	0.00	0.03	0.00
	200	0.69	0.00	0.06	0.00
	500	0.68	0.00	0.03	0.00
CLB	20	0.08	0.00	0.01	0.00
	50	0.11	0.00	0.05	0.00
	100	0.11	0.00	0.11	0.02
	200	0.12	0.00	0.18	0.01
	500	0.11	0.00	0.19	0.01
CMC	20	0.08	0.01	0.15	0.60
	50	0.19	0.07	0.42	0.63
	100	0.28	0.07	0.56	0.31
	200	0.34	0.09	0.72	0.25
	500	0.40	0.11	0.86	0.19
CMS	20	0.33	0.00	0.01	0.50
	50	0.56	0.01	0.03	0.75
	100	0.59	0.06	0.13	0.81
	200	0.67	0.15	0.25	0.89
	500	0.81	0.32	0.52	0.67
CNS	20	0.24	0.00	0.00	–
	50	0.26	0.00	0.03	0.00
	100	0.28	0.00	0.06	0.04
	200	0.29	0.00	0.07	0.00
	500	0.44	0.00	0.00	–
CSS	20	0.16	0.00	0.00	–
	50	0.30	0.00	0.04	0.00
	100	0.31	0.00	0.01	0.00
	200	0.33	0.00	0.00	–
	500	0.56	0.02	0.05	0.40
CWA	20	0.05	0.00	0.00	–
	50	0.24	0.01	0.10	0.34
	100	0.33	0.01	0.06	0.43
	200	0.38	0.02	0.12	0.35
	500	0.40	0.07	0.23	0.48

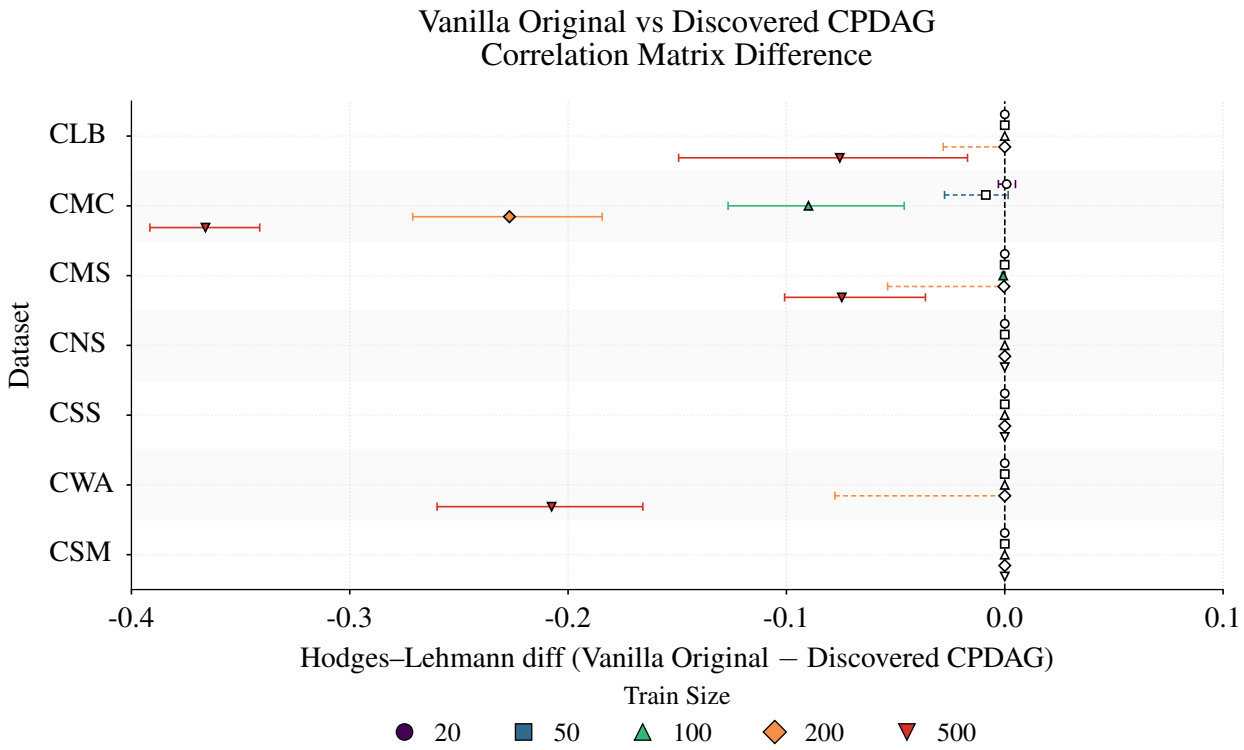

Figure A11: Hodges–Lehmann estimates comparing generation with the discovered CPDAG and vanilla TabPFN with original ordering in CMD. Positive values indicate that it achieves lower metric values (i.e., better synthetic data quality). Filled markers with solid error bars indicate significance at $p < 0.05$ (Holm correction).

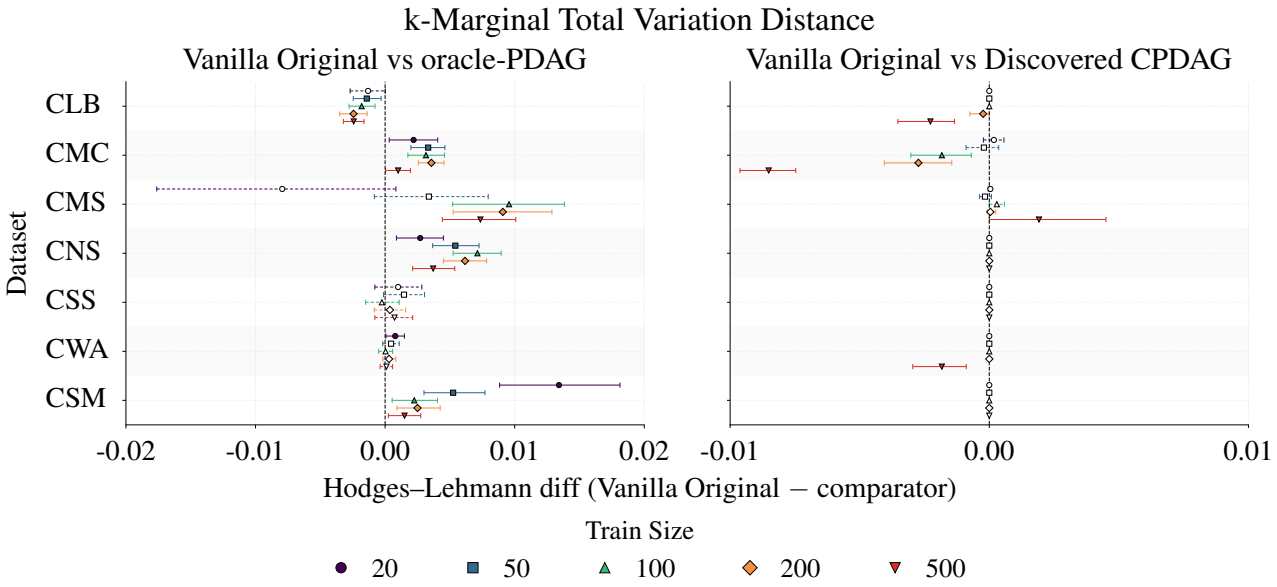

Figure A12: Hodges–Lehmann estimates comparing PDAG-based generation and vanilla TabPFN with original ordering in kMTVD ($k = 2$), with oracle-PDAG (left) and discovered CPDAG (right). Positive values indicate that PDAG-based generation achieves lower metric values (i.e., better synthetic data quality). Filled markers with solid error bars indicate significance at $p < 0.05$ (Holm correction).

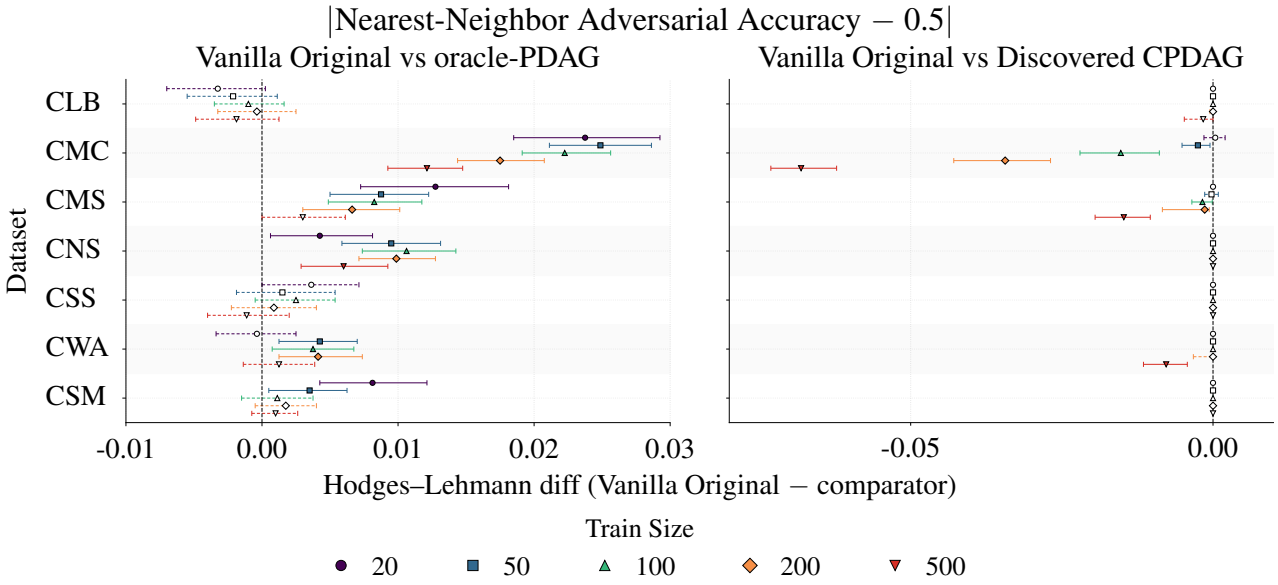

Figure A13: Hodges–Lehmann estimates comparing PDAG-based generation and vanilla TabPFN with original ordering in NNAA (reported as the distance from the ideal value, $|\mathrm{NNAA} - 0.5|$), with oracle-PDAG (left) and discovered CPDAG (right). Positive values indicate that PDAG-based generation achieves lower values (i.e., greater indistinguishability between synthetic and real data). Filled markers with solid error bars indicate significance at $p < 0.05$ (Holm correction).

K COMPARISON WITH EXTERNAL TABULAR GENERATORS

We compare TabPFN-based generation against five external tabular generators. TabularARGN is an autoregressive generator trained over randomized feature orderings, CTGAN is a GAN-based generator, DATGAN and DECAF are causal-aware generators to which we supply the true custom SCM DAG, and CausalDiffTab is a causal-aware diffusion model [Zhang et al., 2025] run with its official defaults. Each baseline is trained per dataset on the same cached training splits used by the comparison experiments and evaluated with the same metrics on the same global test sets, with 100 repetitions per combination paired one-to-one with the TabPFN runs. TabPFN-based generation, by contrast, requires no training. The comparison targets the small-sample setting ($N \leq 500$) in which our method operates.

TabularARGN, the closest practical autoregressive baseline, was run on five datasets and five training sizes: CLB, CMC, CSS, the custom SCM, and its noise variant CSMn2 ($\sigma = 10^{-2}$, Appendix S). TabPFN achieves better median CMD, kMTVD, and $|\mathrm{NNAA} - 0.5|$ in all 25 dataset–size combinations, for both vanilla generation with original ordering and DAG-aware generation, and every paired comparison is significant (Wilcoxon-Pratt with Holm correction).

We run the full five-generator comparison on the custom SCM, the controlled collider setting that isolates the failure mode, while the broader multi-dataset check above is limited to TabularARGN, the only autoregressive baseline and thus the most directly comparable to TabPFN. Table A3 reports the custom SCM comparison for all five baselines. DAG-aware TabPFN achieves significantly lower values in all 75 paired contrasts (five baselines, three metrics, five training sizes), and the paired median difference favors TabPFN in all 150 contrasts overall. Against vanilla TabPFN, 68 of 75 contrasts are significant. The seven non-significant ones all concern CMD for the two causal-aware baselines that approach vanilla's correlation quality at small and medium training sizes (DATGAN at $N \leq 200$ and CausalDiffTab at $50 \leq N \leq 200$). Even when supplied with the true DAG and trained on the target data, no external generator outperforms vanilla TabPFN in any of these paired comparisons. Our repository provides modular code for testing further models.

Table A3: Median synthetic data quality on the custom SCM over 100 repetitions per combination, paired one-to-one with the TabPFN runs. Lower is better for all metrics. NNAA is reported as the distance from the ideal value, $|\mathrm{NNAA} - 0.5|$. † marks generators that receive the true custom SCM DAG. CausalDiffTab estimates causal structure internally (official defaults). TabularARGN and CTGAN receive no causal information. All external baselines are trained on each training split, whereas the TabPFN rows use inference-time conditioning only. The best value per column is in **bold**.

	Training size N				
Method	20	50	100	200	500
CMD					
TabularARGN	2.202	2.210	2.209	2.212	2.214
CTGAN	1.841	1.828	1.868	2.063	2.128
DATGAN†	0.544	0.278	0.240	0.180	0.261
DECAF†	2.216	2.259	3.494	3.485	2.796
CausalDiffTab	1.048	0.313	0.231	0.195	0.207
TabPFN vanilla (original)	0.471	0.285	0.196	0.154	0.120
TabPFN DAG-aware†	**0.130**	**0.135**	**0.085**	**0.086**	**0.074**
kMTVD					
TabularARGN	0.862	0.631	0.522	0.472	0.449
CTGAN	0.517	0.496	0.502	0.598	0.593
DATGAN†	0.503	0.459	0.446	0.435	0.413
DECAF†	0.997	0.996	0.992	0.992	0.990
CausalDiffTab	0.524	0.343	0.285	0.265	0.231
TabPFN vanilla (original)	0.277	0.244	0.225	0.221	0.214
TabPFN DAG-aware†	**0.260**	**0.235**	**0.222**	**0.218**	**0.213**
$\lvert\mathrm{NNAA} - 0.5\rvert$					
TabularARGN	0.491	0.481	0.476	0.474	0.472
CTGAN	0.469	0.470	0.471	0.475	0.475
DATGAN†	0.457	0.447	0.431	0.409	0.419
DECAF†	0.500	0.500	0.500	0.500	0.500
CausalDiffTab	0.466	0.237	0.136	0.096	0.058
TabPFN vanilla (original)	0.055	0.023	0.013	0.011	0.007
TabPFN DAG-aware†	**0.046**	**0.016**	**0.009**	**0.010**	**0.006**

L DISCOVERED CPDAG IN A PC-RECOVERABLE REGIME

The custom SCM was designed to expose TabPFN's autoregressive vulnerability. As a side effect, its near-deterministic mechanisms ($\sigma = 10^{-5}$) make the DAG-implied conditional independencies hard to detect with PC's Fisher–Z tests. Each variable is almost a deterministic function of its parents, so correlations among connected variables approach one and the partial correlations underlying the test degenerate at finite samples. Its weak discovered-CPDAG results therefore reflect a setting where causal discovery fails, not the quality of the generation itself. To isolate what happens when PC succeeds, we keep the same DAG and structural coefficients and increase only the SCM noise to $\sigma = 0.2$ (denoted CSMr in the figures), restoring the conditional variability the test requires while leaving the causal structure unchanged. In a separate discovery screening over 120 runs, PC recovers the exact CPDAG in $98.3\,\%$ of runs at $N = 200$ and $99.2\,\%$ at $N = 500$, and recovery at these training sizes remains high at $\sigma = 0.1$ and $\sigma = 0.5$.

In this setting causal discovery helps synthetic data generation. Discovered-CPDAG generation significantly improves over vanilla original in CMD at all five training sizes and in kMTVD at four of five ($N \geq 50$), closely matching the oracle-PDAG reference for $N \geq 100$ (Figures A14 to A16). NNAA differences are never significant. To verify that this result is not specific to $\sigma = 0.2$, we repeat the same paired comparison at $\sigma = 0.1$ and $\sigma = 0.5$. Figure A17 shows the CMD improvement of the discovered CPDAG over vanilla original alongside the fraction of runs in which PC recovers the exact CPDAG. The improvement rises with the recovery rate at every noise level. At $\sigma = 0.1$ with $N \leq 50$, where PC recovers the exact CPDAG in about a third of the runs or fewer, the improvement is close to zero. Where recovery is high, the improvement is clearly positive, and it shrinks at large N as vanilla TabPFN improves on its own.

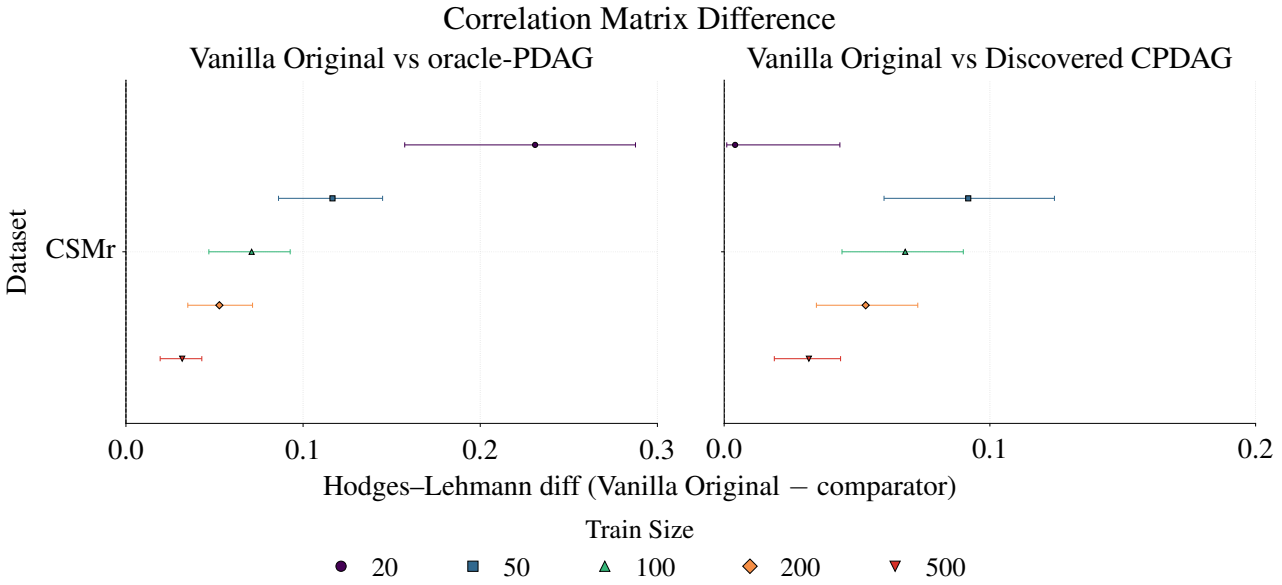

Figure A14: Hodges–Lehmann estimates comparing vanilla TabPFN with original ordering versus oracle-PDAG (left) and discovered CPDAG (right) on the PC-recoverable custom SCM variant (CSMr, $\sigma = 0.2$), in CMD. Positive values indicate that PDAG-based generation achieves lower metric values (i.e., better synthetic data quality). Filled markers with solid error bars indicate significance at $p < 0.05$ (Holm correction).

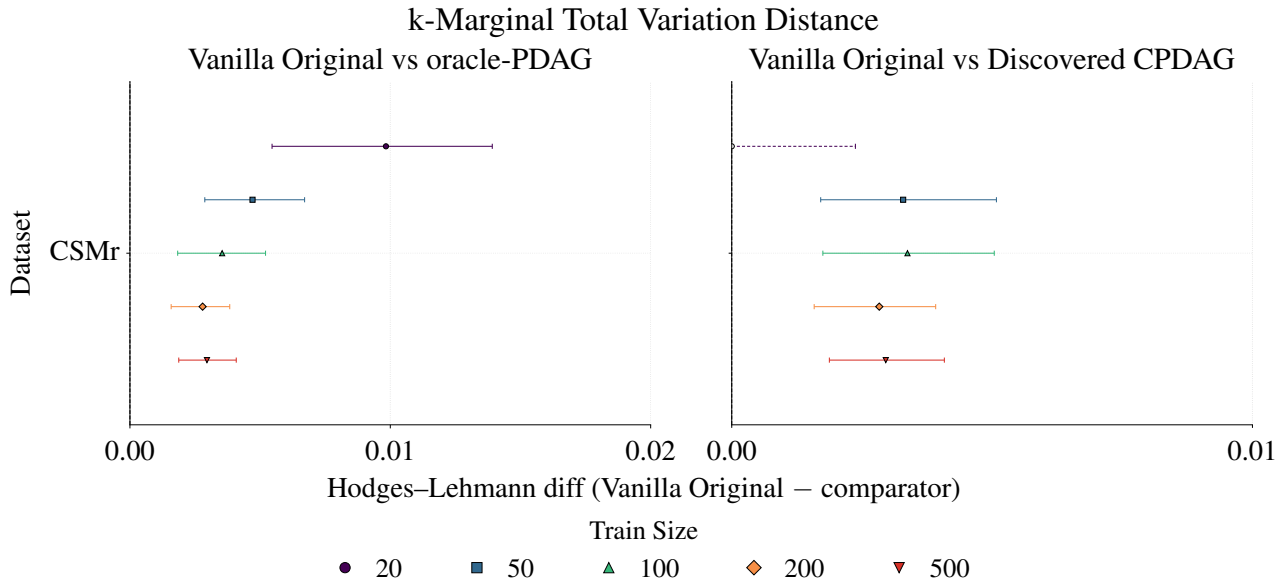

Figure A15: Hodges–Lehmann estimates comparing vanilla TabPFN with original ordering versus oracle-PDAG (left) and discovered CPDAG (right) on the PC-recoverable custom SCM variant (CSMr, $\sigma = 0.2$), in kMTVD ($k = 2$). Positive values indicate that PDAG-based generation achieves lower metric values (i.e., better synthetic data quality). Filled markers with solid error bars indicate significance at $p < 0.05$ (Holm correction).

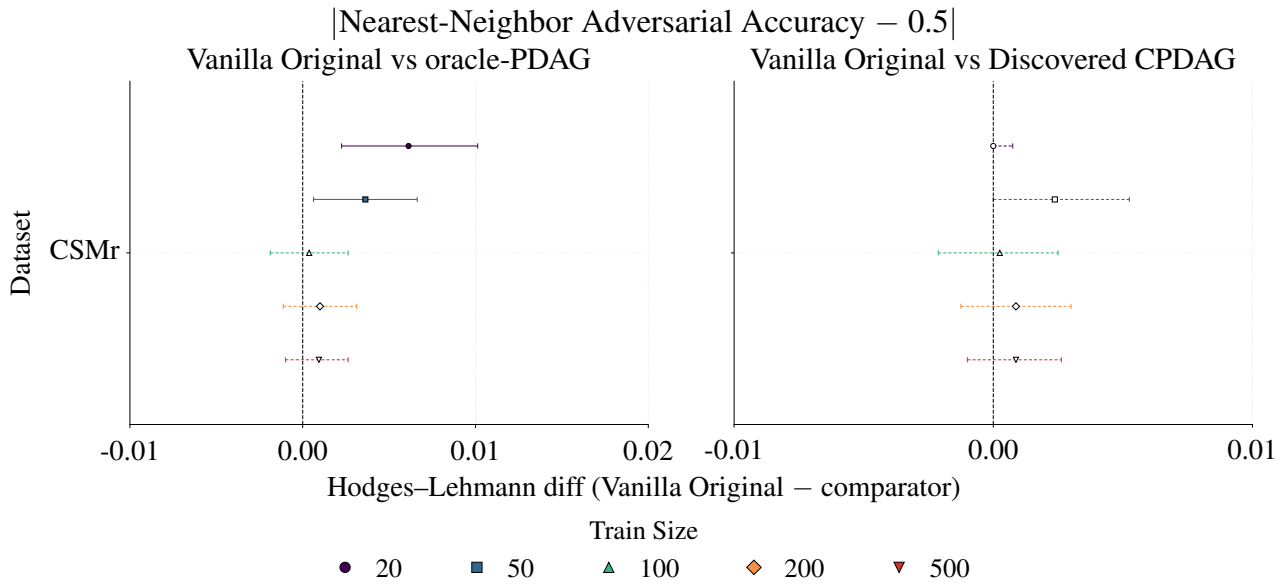

Figure A16: Hodges–Lehmann estimates comparing vanilla TabPFN with original ordering versus oracle-PDAG (left) and discovered CPDAG (right) on the PC-recoverable custom SCM variant (CSMr, $\sigma = 0.2$), in NNAA (reported as the distance from the ideal value, $|\mathrm{NNAA} - 0.5|$). Positive values indicate that PDAG-based generation achieves lower values (i.e., greater indistinguishability between synthetic and real data). Filled markers with solid error bars indicate significance at $p < 0.05$ (Holm correction).

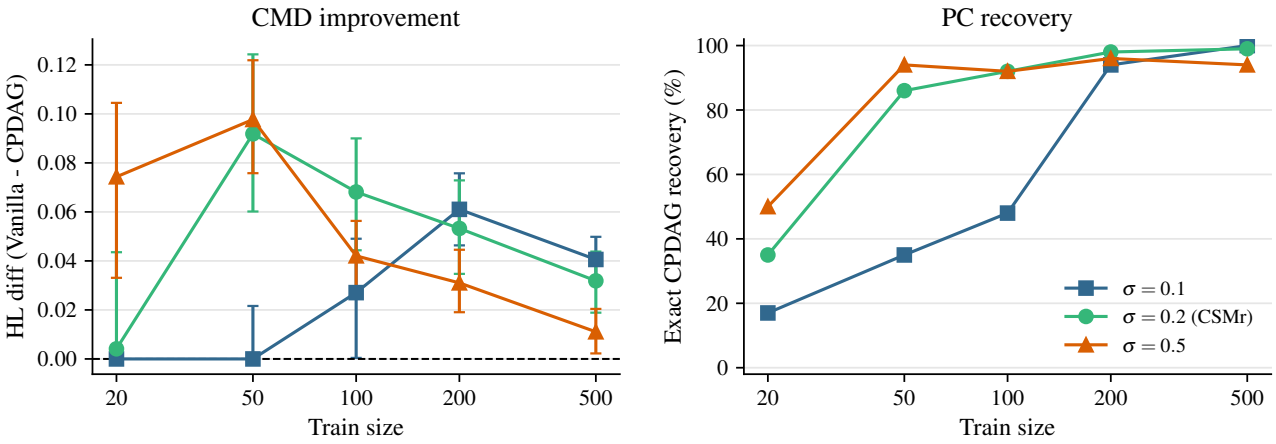

Figure A17: Discovered-CPDAG generation versus PC recovery on the custom SCM at noise levels $\sigma \in \{0.1, 0.2, 0.5\}$. The left panel shows the paired Hodges–Lehmann CMD improvement of discovered-CPDAG generation over vanilla TabPFN with original ordering as a function of training size, over 100 paired repetitions per combination. Values above zero favor the discovered CPDAG. Error bars show Holm-adjusted confidence intervals. The right panel shows the fraction of the same runs in which PC recovers the exact CPDAG.

M VANILLA TABPFN WITH TOPOLOGICAL ORDERING RESULTS ON ATE PRESERVATION: SGL

Figure A18 reports ATE preservation results for vanilla TabPFN with topological ordering on SGL. Topological ordering produces 5 significant improvements out of 6 training sizes, with larger effect sizes than the CSuite datasets (Figure 4).

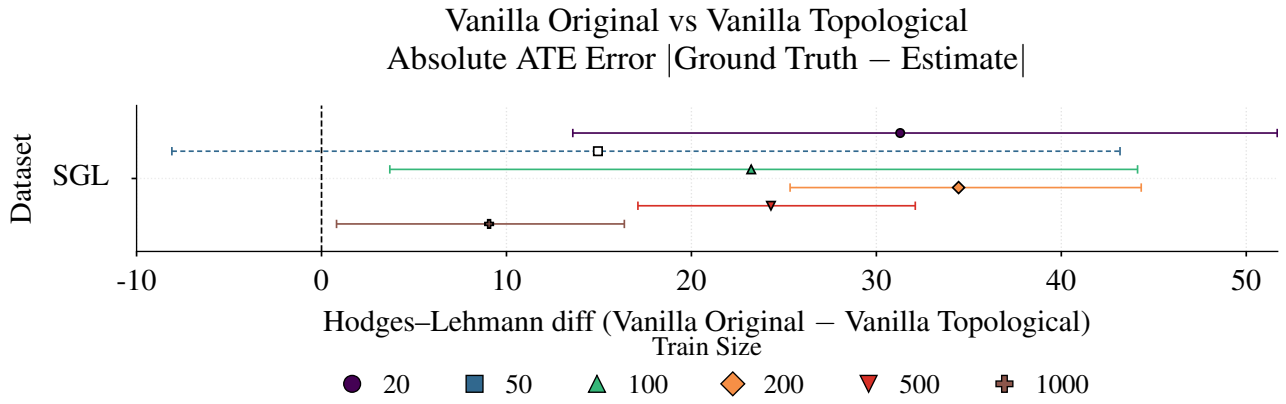

Figure A18: Hodges–Lehmann estimates of the reduction in absolute ATE error (Δ_{ATE}) when comparing vanilla TabPFN with original ordering versus topological ordering on SGL. Positive values indicate smaller errors (closer to ground truth) for topological ordering; negative values indicate larger errors. Filled markers with solid error bars indicate significance at $p < 0.05$ (Holm correction).

N VANILLA TABPFN WITH REVERSE TOPOLOGICAL ORDERING RESULTS ON ATE PRESERVATION

Figure A19 reports results for reverse topological ordering. Reverse topological ordering shows 8 significant improvements and 7 degradations. The largest degradation occurs on CSM at 20 samples.

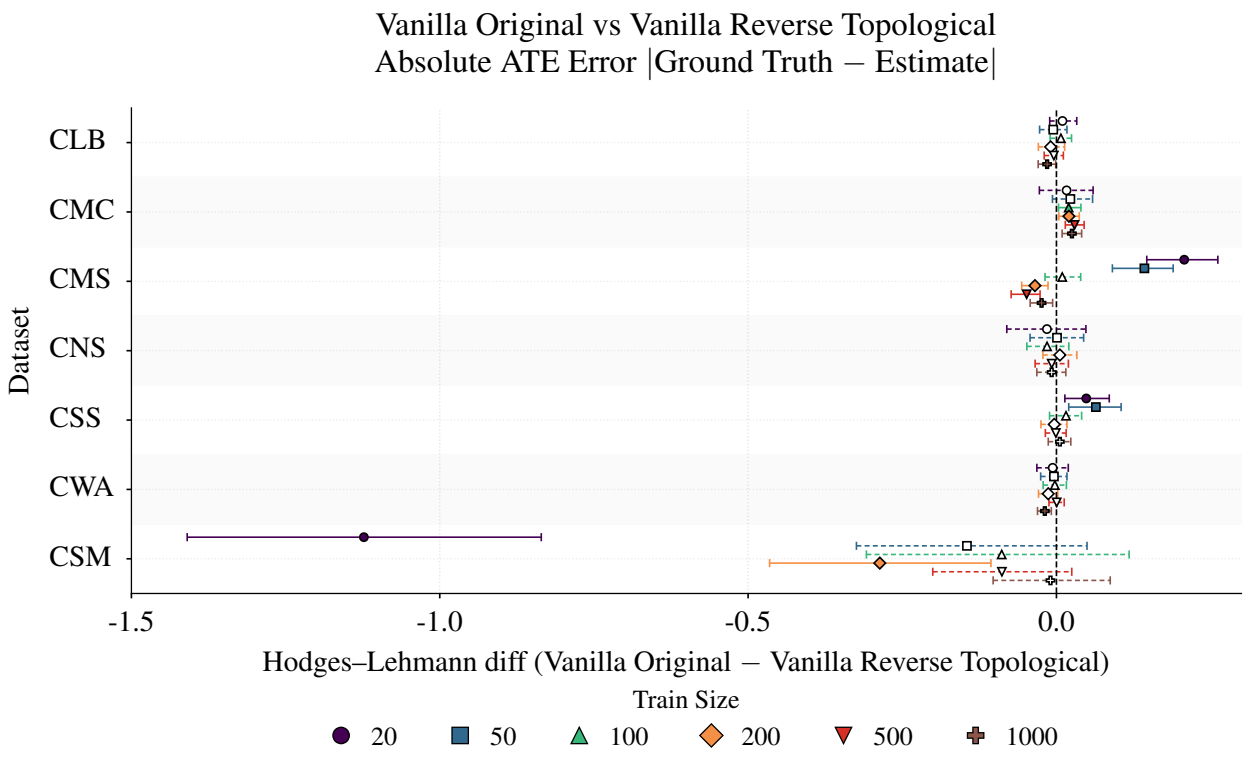

Figure A19: Hodges–Lehmann estimates of the reduction in absolute ATE error (Δ_{ATE}) when comparing vanilla TabPFN with original ordering versus reverse topological ordering. Positive values indicate smaller errors (closer to ground truth) for reverse topological ordering; negative values indicate larger errors. Filled markers with solid error bars indicate significance at $p < 0.05$ (Holm correction).

O DAG-AWARE GENERATION VS VANILLA UNDER TOPOLOGICAL ORDERING ON ATE PRESERVATION

We compare DAG-aware generation and vanilla TabPFN on ATE preservation when both use topological ordering, isolating the contribution of causal parent-based conditioning from the feature ordering itself.

DAG-aware generation shows 15 significant improvements with no degradations (Figure A20). CSM shows significant improvements across most training sizes, confirming that causal parent conditioning provides benefits beyond feature ordering alone.

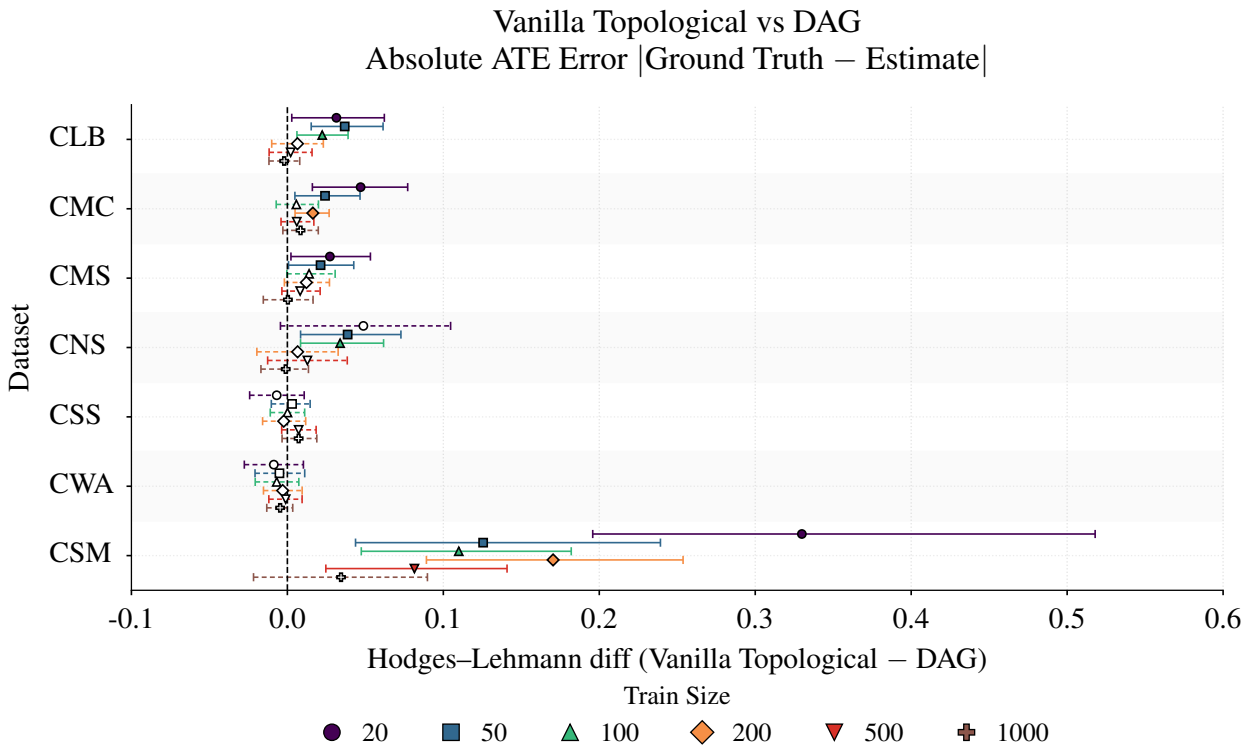

Figure A20: Hodges–Lehmann estimates of the reduction in absolute ATE error (Δ_{ATE}) when comparing vanilla TabPFN with topological ordering versus DAG-aware generation. Positive values indicate smaller errors (closer to ground truth) for DAG-aware generation; negative values indicate larger errors. Filled markers with solid error bars indicate significance at $p < 0.05$ (Holm correction).

P DISCOVERED CPDAG RESULTS ON ATE PRESERVATION

Figure A21 reports ATE preservation results for discovered CPDAG. The discovered CPDAG shows one significant improvement on CMC at $N = 1000$ and one degradation on CWA at $N = 1000$. Table A4 reports graph discovery quality against the mutilated DAG (the causal graph with edges into the treatment variable removed), which is the appropriate reference for interventional data. On CMC at $N = 1000$, PC orients $88\,\%$ of discovered edges with direction precision 0.24, providing enough correct structure around the treatment–outcome path to improve ATE estimation. On CWA at $N = 1000$, PC orients only $28\,\%$ of edges with direction precision 0.02, causing the few oriented edges to introduce incorrect conditioning around the treatment–outcome relationship.

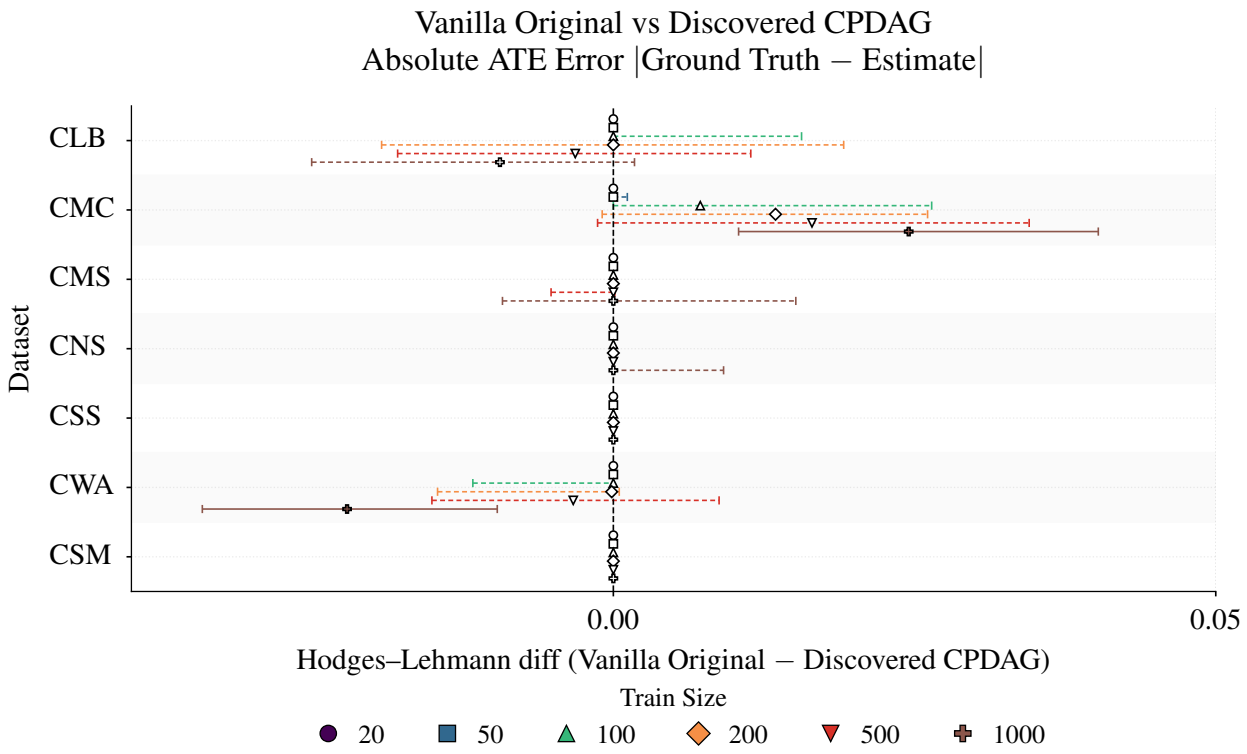

Figure A21: Hodges–Lehmann estimates of the reduction in absolute ATE error (Δ_{ATE}) when comparing vanilla TabPFN with original ordering versus discovered CPDAG. Positive values indicate smaller errors (closer to ground truth) for discovered CPDAG; negative values indicate larger errors. Filled markers with solid error bars indicate significance at $p < 0.05$ (Holm correction).

Table A4: PC-stable graph discovery quality on CSuite datasets and the custom SCM for the ATE experiments, averaged over 100 repetitions. Metrics are computed against the mutilated DAG (edges into the treatment variable removed). Skeleton recall measures the fraction of true edges recovered. Direction recall measures the fraction of true edge orientations correctly identified. Oriented fraction reports the proportion of discovered edges that PC orients. Direction precision measures the fraction of oriented edges with correct orientation (– indicates no oriented edges).

Dataset	N	Skel. rec.	Dir. rec.	Orient. frac.	Dir. prec.
CSM	20	0.68	0.00	0.03	0.00
	50	0.68	0.00	0.05	0.00
	100	0.68	0.00	0.03	0.00
	200	0.67	0.00	0.01	0.00
	500	0.67	0.00	0.01	0.00
	1000	0.67	0.00	0.00	–
CLB	20	0.19	0.00	0.01	0.50
	50	0.53	0.01	0.08	0.29
	100	0.78	0.05	0.24	0.21
	200	0.91	0.04	0.28	0.13
	500	1.00	0.04	0.32	0.07
	1000	1.00	0.03	0.31	0.03
CMC	20	0.07	0.01	0.17	0.30
	50	0.18	0.08	0.39	0.65
	100	0.22	0.11	0.60	0.52
	200	0.24	0.14	0.75	0.44
	500	0.33	0.18	0.81	0.39
	1000	0.38	0.15	0.88	0.24
CMS	20	0.27	0.01	0.19	0.20
	50	0.43	0.02	0.14	0.23
	100	0.46	0.01	0.14	0.14
	200	0.64	0.02	0.38	0.07
	500	0.82	0.02	0.61	0.04
	1000	0.88	0.03	0.71	0.03
CNS	20	0.31	0.00	0.02	0.50
	50	0.40	0.00	0.00	–
	100	0.56	0.00	0.06	0.13
	200	0.64	0.01	0.10	0.14
	500	0.71	0.00	0.20	0.00
	1000	0.83	0.00	0.59	0.00
CSS	20	0.38	0.01	0.05	0.67
	50	0.50	0.09	0.17	1.00
	100	0.67	0.06	0.12	0.75
	200	0.67	0.00	0.02	0.00
	500	0.67	0.00	0.00	–
	1000	0.67	0.00	0.01	0.00
CWA	20	0.14	0.01	0.10	0.40
	50	0.43	0.02	0.14	0.31
	100	0.50	0.01	0.15	0.12
	200	0.53	0.01	0.19	0.11
	500	0.58	0.01	0.29	0.02
	1000	0.58	0.01	0.28	0.02

Q ABSOLUTE SCALE OF ATE ERRORS

The ATE-preservation forest plots report paired Hodges–Lehmann reductions in the absolute ATE error Δ_{ATE} without an explicit reference scale. Table A5 complements them by reporting, for each dataset and training size, the absolute value of the true test-set ATE together with the median Δ_{ATE} of vanilla TabPFN with original ordering and of the primary comparator. All values are expressed in the native outcome units of each dataset. SGL values are in mg/dL. On the SGL simulator, topological ordering lowers the median ATE error from 39.6–84.9 mg/dL (vanilla TabPFN) to 15.9–64.4 mg/dL, a paired Hodges–Lehmann reduction of 9.06–34.4 mg/dL significant at every training size except $N = 50$. These errors remain large relative to the true test-set ATE of 2.29 mg/dL.

Table A5: Absolute-scale reference for ATE preservation over 100 paired repetitions per combination. $|\text{ATE}_{\text{test}}|$ is the absolute true ATE on the global test set. The error columns report the median absolute ATE error (Δ_{ATE}) of vanilla TabPFN with original ordering and of the primary comparator. The comparator is DAG-aware generation with topological ordering, except on SGL, where it is vanilla TabPFN with topological ordering because no full DAG is available. HL is the paired Hodges–Lehmann estimate of the error reduction (vanilla minus comparator). The significance flag comes from the paired Wilcoxon signed-rank test (Pratt ties, Holm correction) and matches the corresponding forest plots. A set flag with a negative HL marks a significant worsening.

Dataset	N	$\lvert\text{ATE}_{\text{test}}\rvert$	Vanilla Δ_{ATE}	Comp. Δ_{ATE}	HL	Sig.
CSM	20	4.98	1.23	0.186	1.23	Y
	50	4.98	1.15	0.183	1.00	Y
	100	4.98	0.826	0.163	0.649	Y
	200	4.98	0.343	0.162	0.268	Y
	500	4.98	0.274	0.152	0.14	Y
	1000	4.98	0.243	0.151	0.119	Y
CLB	20	0.593	0.402	0.328	0.0336	Y
	50	0.593	0.271	0.221	0.0433	Y
	100	0.593	0.14	0.111	0.03	Y
	200	0.593	0.0961	0.0782	0.007 25	N
	500	0.593	0.0556	0.0533	0.004 43	N
	1000	0.593	0.0405	0.0301	0.0087	N
CMC	20	1.03	0.389	0.479	−0.063	Y
	50	1.03	0.173	0.105	0.037	Y
	100	1.03	0.129	0.0873	0.0286	Y
	200	1.03	0.0926	0.0554	0.0357	Y
	500	1.03	0.0894	0.0453	0.0337	Y
	1000	1.03	0.0701	0.0527	0.0231	Y
CMS	20	0.727	0.491	0.133	0.348	Y
	50	0.727	0.445	0.0742	0.33	Y
	100	0.727	0.192	0.0677	0.145	Y
	200	0.727	0.094	0.0496	0.0511	Y
	500	0.727	0.0705	0.0496	0.0242	Y
	1000	0.727	0.081	0.0565	0.0211	Y
CNS	20	1.96	0.73	0.289	0.525	Y
	50	1.96	0.419	0.123	0.337	Y
	100	1.96	0.278	0.101	0.199	Y
	200	1.96	0.178	0.142	0.0684	Y
	500	1.96	0.129	0.126	0.004 24	N
	1000	1.96	0.106	0.0519	0.0562	Y

Table A5 (continued)

Dataset	N	$\lvert\text{ATE}_{\text{test}}\rvert$	Vanilla Δ_{ATE}	Comp. Δ_{ATE}	HL	Sig.
CSS	20	0.0371	0.212	0.262	−0.0538	Y
	50	0.0371	0.237	0.219	−0.001 67	N
	100	0.0371	0.137	0.151	−0.006 83	N
	200	0.0371	0.0901	0.0849	0.001 35	N
	500	0.0371	0.0673	0.0458	0.0201	Y
	1000	0.0371	0.0577	0.0446	0.0126	N
CWA	20	1.08	0.491	0.523	−0.0118	N
	50	1.08	0.144	0.152	0.006 59	N
	100	1.08	0.1	0.0901	−0.000 419	N
	200	1.08	0.0779	0.0853	−0.005 65	N
	500	1.08	0.0605	0.0454	0.0124	N
	1000	1.08	0.0274	0.0352	−0.005 62	N
SGL	20	2.29	84.9	50.7	31.3	Y
	50	2.29	65.2	64.4	14.9	N
	100	2.29	79.0	51.8	23.2	Y
	200	2.29	58.9	23.9	34.4	Y
	500	2.29	39.6	15.9	24.3	Y
	1000	2.29	43.5	28.1	9.06	Y

R EMPIRICAL CONDITIONAL-INDEPENDENCE PRESERVATION

We test whether the synthetic data preserve the conditional-independence (CI) structure of the real data. For each training split we identify the CI statements that are not rejected on the real data, test the same statements on the matched synthetic data, and report the fraction that remains non-rejected. We use the same conditional-independence test as the discovery pipeline for each dataset. We evaluate 100 paired repetitions per dataset and training size and compare conditions with the paper's paired protocol (Wilcoxon signed-rank with Pratt ties, Holm correction).

Under observational generation, preservation is high for every condition. Median fractions range from 0.75 to 1.00 across datasets and training sizes, and $82\,\%$ of combinations are at or above 0.89. The paired differences between DAG-aware and vanilla generation are small (absolute Hodges–Lehmann effects below 0.05) and show no consistent direction.

Under interventional generation, causal conditioning matters (Table A6). DAG-aware generation preserves significantly more empirical CIs than vanilla TabPFN in 4 of 6 training sizes on CSM (median Hodges–Lehmann effect $+0.11$) and 5 of 6 on CSMn2 ($+0.23$), and it is never significantly lower on either. CSMn2 is the custom SCM with $\sigma = 10^{-2}$ evaluated in Appendix S. Oracle-PDAG follows the same pattern (4 and 5 of 6), while discovered CPDAG does so in only 0 and 1 of 6, reflecting discovery quality rather than the PDAG-based conditioning rule. On CMS, both DAG-aware and vanilla generation preserve almost all CIs, with no significant difference between them. On the remaining CSuite datasets vanilla and DAG-aware generation are essentially at ceiling, and the few significant differences involving DAG-aware generation are small (absolute Hodges–Lehmann effects below 0.011). They all occur at large training sizes ($N \geq 200$). Two of them fall on CLB, where the ATE improvement of DAG-aware generation in the same interventional runs is also not significant at those sizes, reflecting the same large-sample convergence of vanilla TabPFN. The diagnostic confirms the paper's central message that under interventional generation causal-structure-aware conditioning reliably preserves the empirical independence structure of the real data.

Table A6: Median fraction of empirical conditional independencies preserved under interventional generation, over 100 paired repetitions per combination. Reference statements are the CIs not rejected on the real training data, using for each dataset the same conditional-independence test as the discovery pipeline. The same statements are then tested on the synthetic data. * marks medians whose paired difference from vanilla original is significant (Wilcoxon-Pratt with Holm correction within each dataset–size family). A starred median below the corresponding vanilla value indicates a significant decrease relative to vanilla. When the displayed medians coincide, the direction of the difference is not visible at this precision.

Dataset	N	Vanilla	DAG-aware	Oracle-PDAG	Disc. CPDAG
CSM	20	0.91	0.91	0.91	0.91
	50	0.89	0.89	0.89*	0.82
	100	0.78	0.89*	0.89*	0.78
	200	0.78	0.89*	0.89*	0.78
	500	0.67	0.89*	0.67	0.67
	1000	0.63	0.75*	0.75*	0.59
CSMn2	20	0.91	0.91	0.91	0.91
	50	0.67	0.89*	0.89*	0.73*
	100	0.67	0.89*	0.89*	0.67
	200	0.67	0.89*	0.89*	0.67
	500	0.67	0.89*	0.89*	0.67
	1000	0.67	1.00*	1.00*	0.67
CLB	20	1.00	1.00	1.00	1.00
	50	0.97	0.97	0.98	0.97
	100	0.96	0.96	0.96	0.96
	200	0.97	0.96*	0.98*	0.97
	500	0.99	0.98*	0.99*	0.98*
	1000	0.99	0.99	0.99	0.96*
CMC	20	1.00	1.00	1.00	1.00
	50	1.00	1.00	1.00	1.00
	100	1.00	1.00	1.00	1.00
	200	1.00	1.00	1.00	1.00
	500	1.00	0.99*	1.00*	1.00*
	1000	0.99	0.99*	0.99*	0.99*
CMS	20	1.00	1.00	1.00	1.00
	50	1.00	1.00	1.00	1.00
	100	1.00	0.91	0.91	1.00
	200	0.89	0.88	0.88	0.85
	500	0.80	0.78	0.80	0.67*
	1000	1.00	1.00	1.00	0.50*
CNS	20	0.95	0.96	0.95	0.95
	50	1.00	1.00	1.00	1.00
	100	1.00	1.00	1.00	0.88
	200	1.00	1.00*	1.00	1.00
	500	1.00	1.00	1.00	1.00
	1000	1.00	1.00	1.00	0.83*
CSS	20	1.00	1.00	1.00	1.00
	50	1.00	1.00	1.00	1.00
	100	0.93	0.90	0.93	0.93
	200	1.00	1.00	1.00	1.00
	500	1.00	1.00	1.00	1.00
	1000	1.00	1.00	1.00	1.00

Table A6 (continued)

Dataset	N	Vanilla	DAG-aware	Oracle-PDAG	Disc. CPDAG
CWA	20	1.00	1.00	1.00	1.00
	50	0.96	0.96	0.96	0.96
	100	0.95	0.95	0.95	0.96*
	200	0.95	0.94	0.97*	0.95
	500	0.95	0.95	0.98*	0.95
	1000	0.97	0.96*	0.97	0.94*

S ROBUSTNESS TO HIGHER NOISE

To verify that the findings reported in the main text are not specific to the near-deterministic regime ($\sigma = 10^{-5}$), we repeat our experiments on the custom SCM with $\sigma = 10^{-2}$ (denoted CSMn2).

SYNTHETIC DATA QUALITY

DAG-aware generation shows significant improvements over vanilla TabPFN across all training sizes in both CMD and kMTVD (Figure A22). In NNAA, improvements are significant at $N = 20$ and $N = 50$ but not at larger training sizes (Figure A23), consistent with the $\sigma = 10^{-5}$ results.

Oracle-PDAG shows the same pattern: significant improvements in CMD and kMTVD across all training sizes (Figures A24 and A25), with NNAA significant only at small N (Figure A26). Discovered CPDAG shows no meaningful differences across metrics and training sizes, consistent with the $\sigma = 10^{-5}$ results.

These results confirm that the benefits of causal conditioning extend beyond the near-deterministic regime.

Table A7 confirms that spurious correlations under $\sigma = 10^{-2}$ are comparable in magnitude to those observed under $\sigma = 10^{-5}$ (Tables A1 and 2). Oracle-PDAG shows reduced correlations under higher noise, suggesting increased robustness to imprecise conditioning from undirected edges.

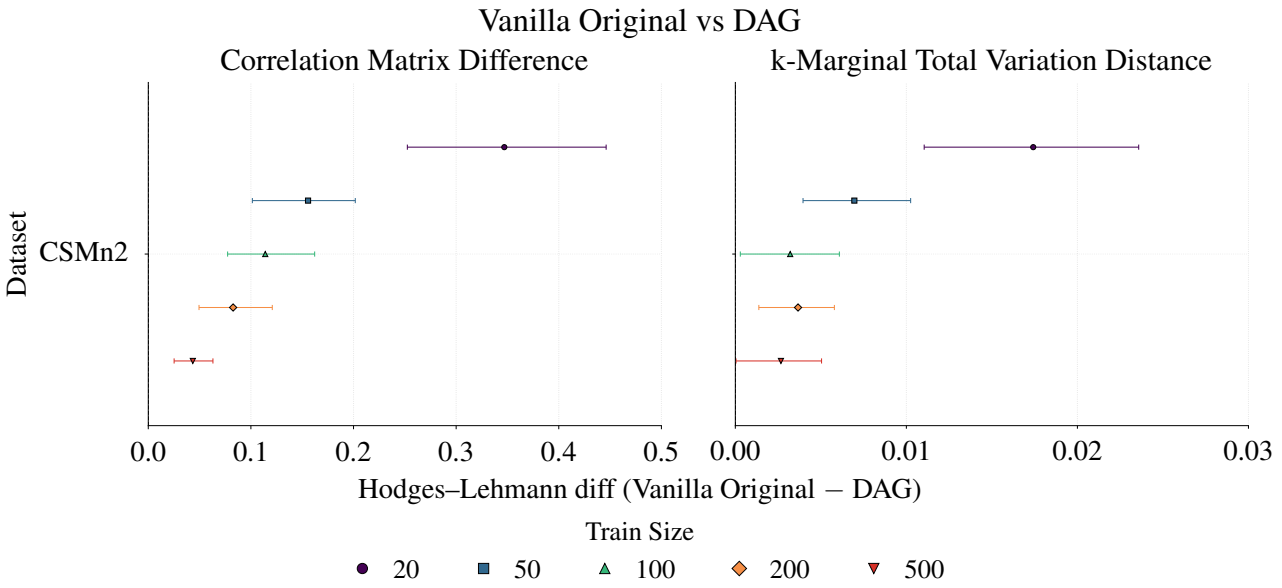

Figure A22: Hodges–Lehmann estimates comparing DAG-aware generation and vanilla TabPFN with original ordering on the custom SCM with $\sigma = 10^{-2}$, in CMD (left) and kMTVD ($k = 2$, right). Positive values indicate that DAG-aware generation achieves lower metric values (i.e., better synthetic data quality). Filled markers with solid error bars indicate significance at $p < 0.05$ (Holm correction).

Table A7: Mean Pearson correlation coefficients for independent variable pairs in the custom collider SCM with $\sigma = 10^{-2}$. Values close to zero indicate correct independence preservation; the best value per column and training size is in **bold**. Standard deviations across 100 repetitions in parentheses.

Method	$\rho(X_0, X_3)$		$\rho(X_0, X_2)$	
Train size $N = 20$				
Vanilla original	-0.147	(0.205)	-0.145	(0.206)
Vanilla topological	-0.015	(0.116)	-0.014	(0.115)
Vanilla reverse top.	-0.121	(0.202)	-0.119	(0.204)
DAG-aware	0.004	(0.021)	$\mathbf{0.003}$	(0.021)
Oracle-PDAG	$\mathbf{0.003}$	(0.023)	0.004	(0.021)
Discovered CPDAG	-0.136	(0.215)	-0.134	(0.216)
Train size $N = 50$				
Vanilla original	-0.026	(0.144)	-0.026	(0.144)
Vanilla topological	0.005	(0.075)	0.006	(0.076)
Vanilla reverse top.	-0.033	(0.148)	-0.034	(0.146)
DAG-aware	$\mathbf{0.000}$	(0.023)	$\mathbf{0.000}$	(0.024)
Oracle-PDAG	-0.001	(0.024)	$\mathbf{0.000}$	(0.023)
Discovered CPDAG	0.005	(0.194)	0.005	(0.193)
Train size $N = 100$				
Vanilla original	-0.043	(0.113)	-0.043	(0.113)
Vanilla topological	-0.017	(0.064)	-0.017	(0.063)
Vanilla reverse top.	-0.049	(0.102)	-0.047	(0.102)
DAG-aware	$\mathbf{0.000}$	(0.020)	0.001	(0.020)
Oracle-PDAG	-0.002	(0.025)	$\mathbf{0.000}$	(0.020)
Discovered CPDAG	-0.022	(0.143)	-0.022	(0.145)
Train size $N = 200$				
Vanilla original	-0.029	(0.099)	-0.034	(0.080)
Vanilla topological	-0.012	(0.042)	-0.012	(0.042)
Vanilla reverse top.	-0.035	(0.082)	-0.028	(0.109)
DAG-aware	$\mathbf{-0.002}$	(0.025)	$\mathbf{-0.001}$	(0.026)
Oracle-PDAG	-0.003	(0.029)	-0.002	(0.025)
Discovered CPDAG	0.019	(0.178)	0.017	(0.175)
Train size $N = 500$				
Vanilla original	-0.027	(0.043)	-0.027	(0.042)
Vanilla topological	-0.004	(0.025)	-0.003	(0.025)
Vanilla reverse top.	-0.033	(0.039)	-0.032	(0.039)
DAG-aware	$\mathbf{0.000}$	(0.023)	0.001	(0.023)
Oracle-PDAG	$\mathbf{0.000}$	(0.023)	$\mathbf{0.000}$	(0.023)
Discovered CPDAG	0.016	(0.146)	0.019	(0.155)
Test set	0.022	–	0.022	–

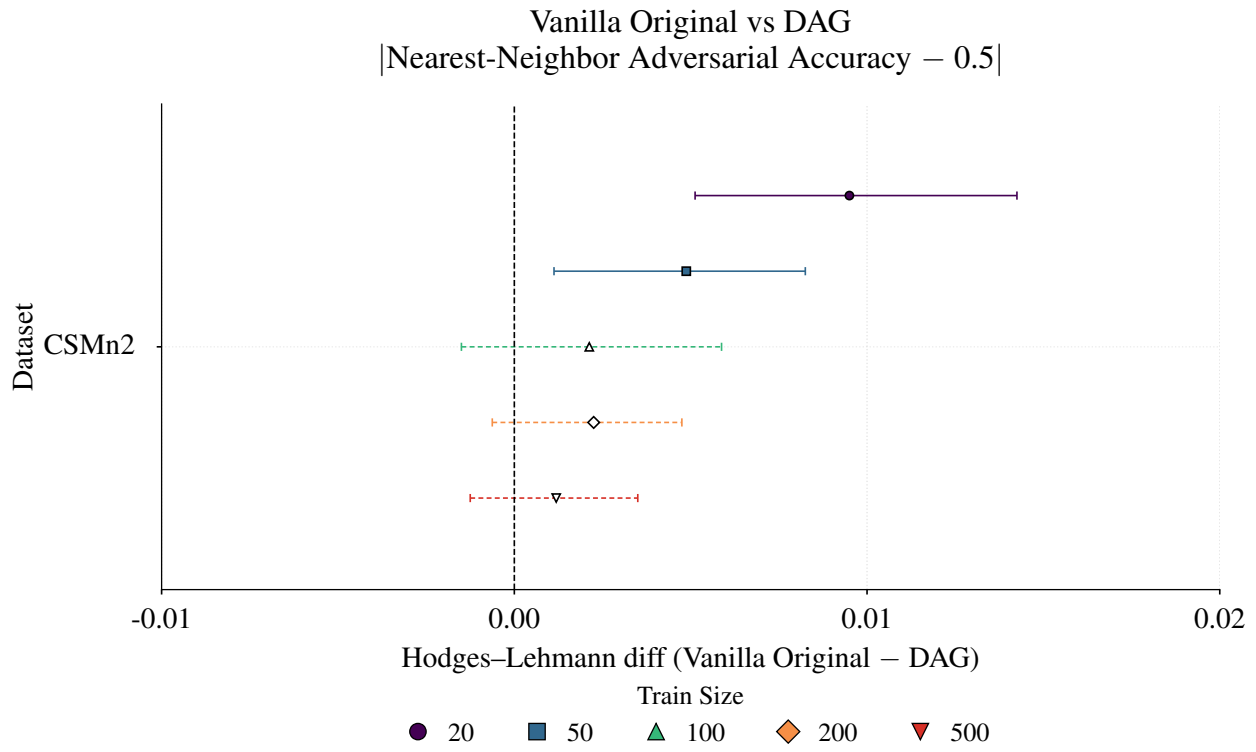

Figure A23: Hodges–Lehmann estimates comparing DAG-aware generation and vanilla TabPFN on the custom SCM with $\sigma = 10^{-2}$, in NNAA (reported as the distance from the ideal value, $|\mathrm{NNAA} - 0.5|$). Positive values indicate that DAG-aware generation achieves lower values (i.e., greater indistinguishability between synthetic and real data). Filled markers with solid error bars indicate significance at $p < 0.05$ (Holm correction).

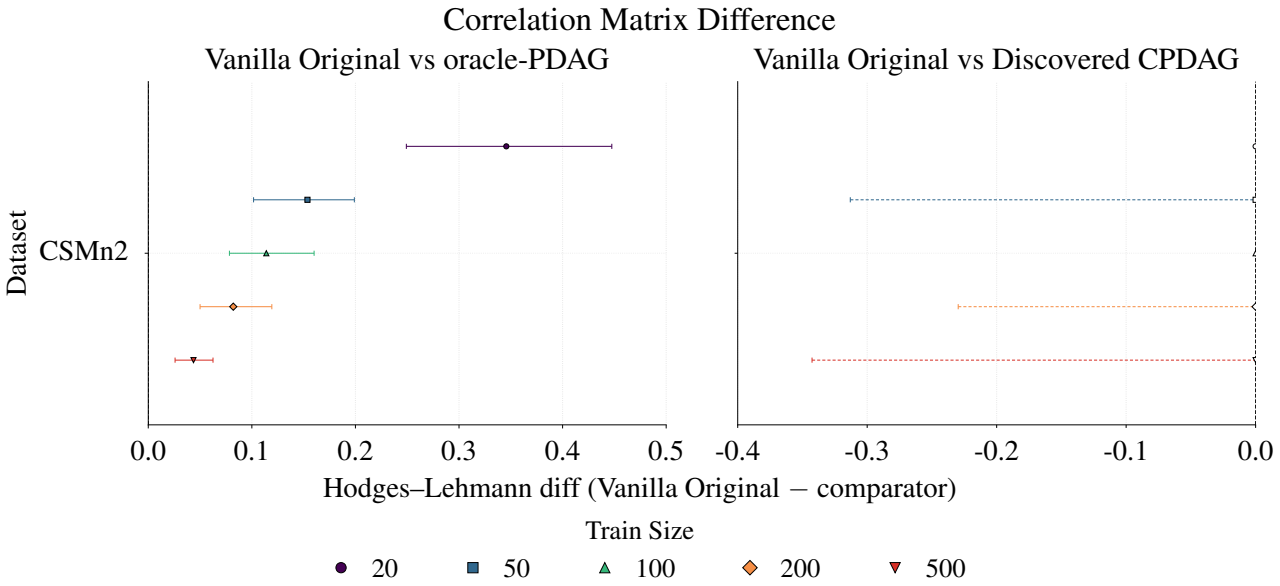

Figure A24: Hodges–Lehmann estimates comparing PDAG-based generation and vanilla TabPFN on the custom SCM with $\sigma = 10^{-2}$, in CMD, with oracle-PDAG (left) and discovered CPDAG (right). Positive values indicate that PDAG-based generation achieves lower metric values (i.e., better synthetic data quality). Filled markers with solid error bars indicate significance at $p < 0.05$ (Holm correction).

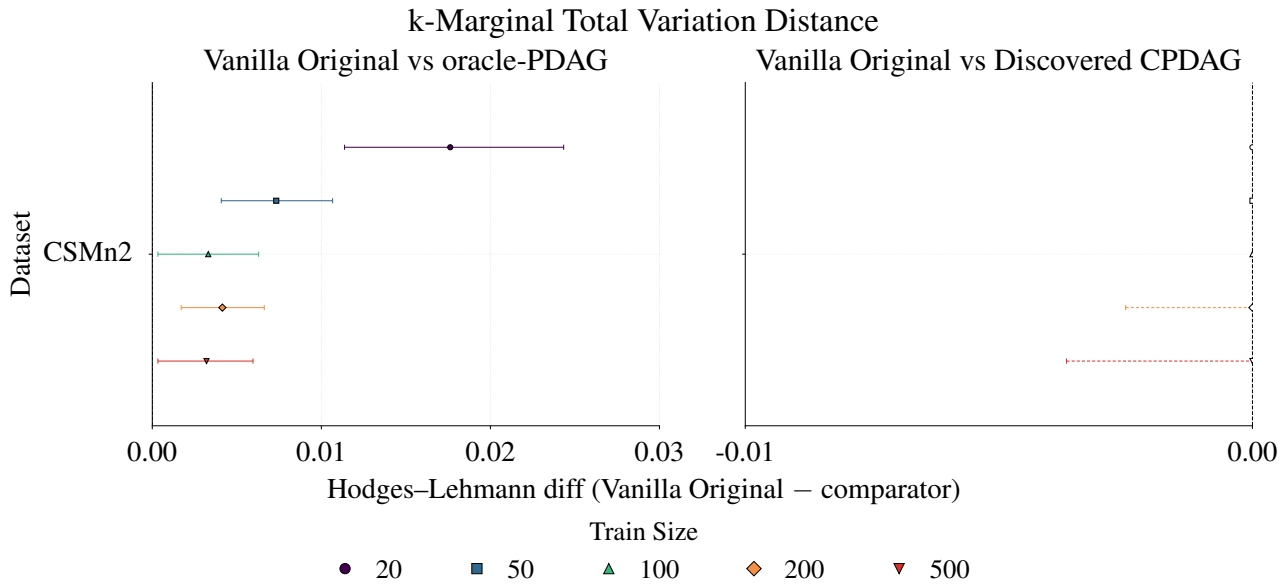

Figure A25: Hodges–Lehmann estimates comparing PDAG-based generation and vanilla TabPFN with original ordering on the custom SCM with $\sigma = 10^{-2}$, in kMTVD ($k = 2$), with oracle-PDAG (left) and discovered CPDAG (right). Positive values indicate that PDAG-based generation achieves lower metric values (i.e., better synthetic data quality). Filled markers with solid error bars indicate significance at $p < 0.05$ (Holm correction).

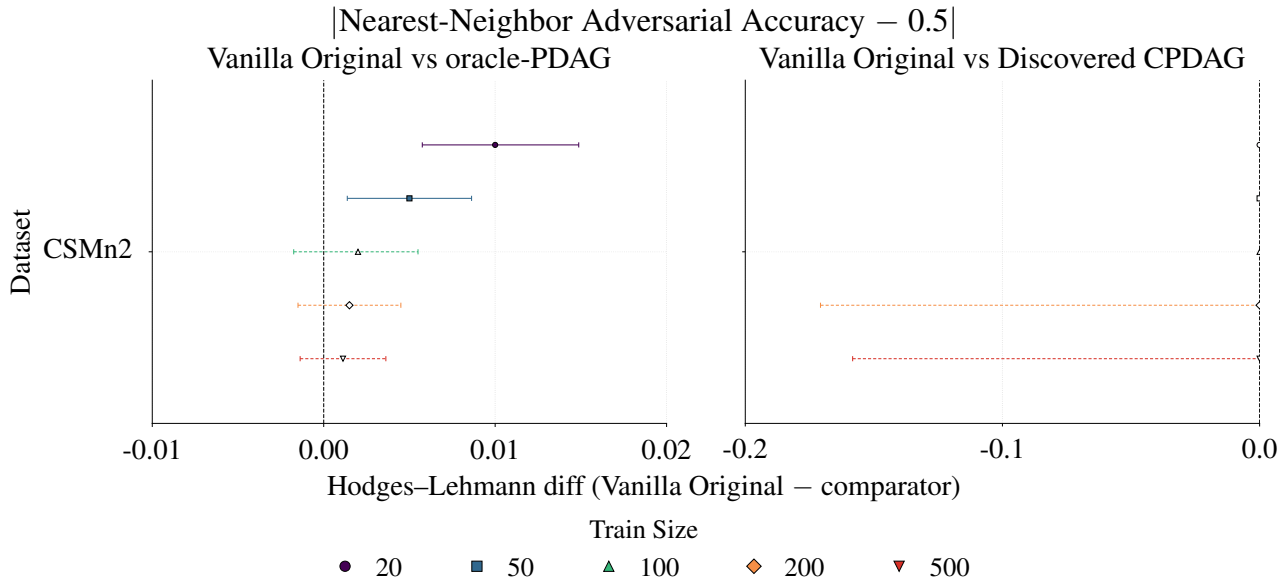

Figure A26: Hodges–Lehmann estimates comparing PDAG-based generation and vanilla TabPFN on the custom SCM with $\sigma = 10^{-2}$, in NNAA (reported as the distance from the ideal value, $|\mathrm{NNAA} - 0.5|$), with oracle-PDAG (left) and discovered CPDAG (right). Positive values indicate that PDAG-based generation achieves lower values (i.e., greater indistinguishability between synthetic and real data). Filled markers with solid error bars indicate significance at $p < 0.05$ (Holm correction).

TREATMENT EFFECT PRESERVATION

DAG-aware generation significantly improves ATE preservation across $N \in \{20, 50, 100, 200, 500\}$ compared to vanilla TabPFN (Figure A27). At $N = 1000$, the direction remains favorable but not statistically significant. Oracle-PDAG shows the same pattern (Figure A28). Discovered CPDAG shows no differences across training sizes.

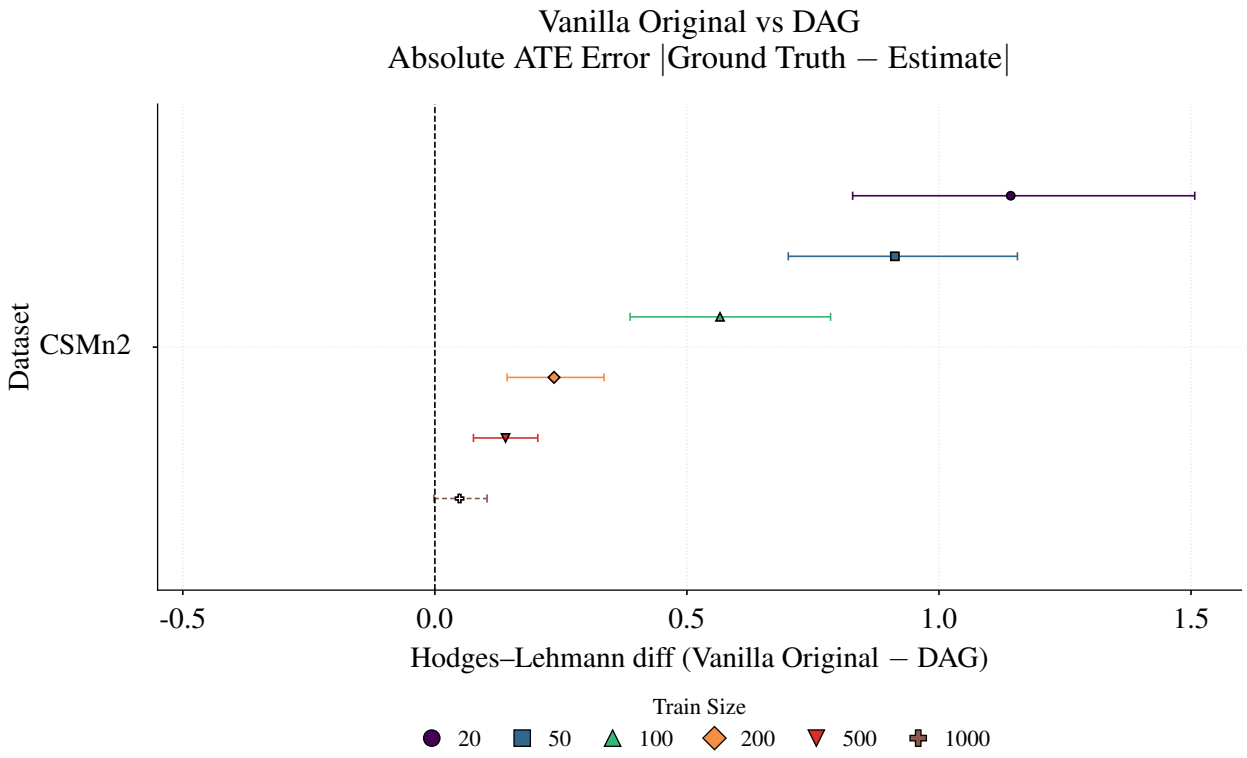

Figure A27: Hodges–Lehmann estimates of the reduction in absolute ATE error (Δ_{ATE}) when comparing vanilla TabPFN with original ordering versus DAG-aware generation on the custom SCM with $\sigma = 10^{-2}$. Positive values indicate smaller errors (closer to ground truth) for DAG-aware generation; negative values indicate larger errors. Filled markers with solid error bars indicate significance at $p < 0.05$ (Holm correction).

T CAUSAL DISCOVERY WITH REX

To assess the sensitivity of our approach to the choice of causal discovery algorithm, we tested ReX [Renero et al., 2026], a method that employs machine learning regressors for skeleton discovery and exploits additive noise model assumptions to orient edges, on the four CSuite datasets that satisfy these assumptions (CLB, CNS, CSS, CWA). We ran ReX with default hyperparameters, as preliminary tests showed no meaningful difference with tuning at these sample sizes. Given the consistent degradations observed across all distributional metrics, we did not extend the evaluation to ATE preservation.

Table A8 reports graph quality metrics averaged over 100 repetitions. ReX exhibits low direction precision at small and medium training sizes, particularly on CLB, CNS, and CWA (below 50 % for $N \leq 100$). Graph quality improves at $N = 500$ for CLB (0.96 skeleton recall, 0.95 direction precision) and CSS (1.00 direction precision, 0.60 recall). CNS achieves high direction precision at $N = 500$ but skeleton recall remains at 0.50. CWA maintains 0.75 direction precision and 0.57 recall even at $N = 500$.

Using ReX-discovered DAGs for causal conditioning leads to significant degradations compared to vanilla TabPFN across nearly all dataset–training size combinations: 15 significant degradations and one significant improvement (CSS at $N = 500$, where ReX reaches 1.00 direction precision) in CMD, and 20 significant degradations with no improvements in both kMTVD and NNAA (Figures A29 and A30).

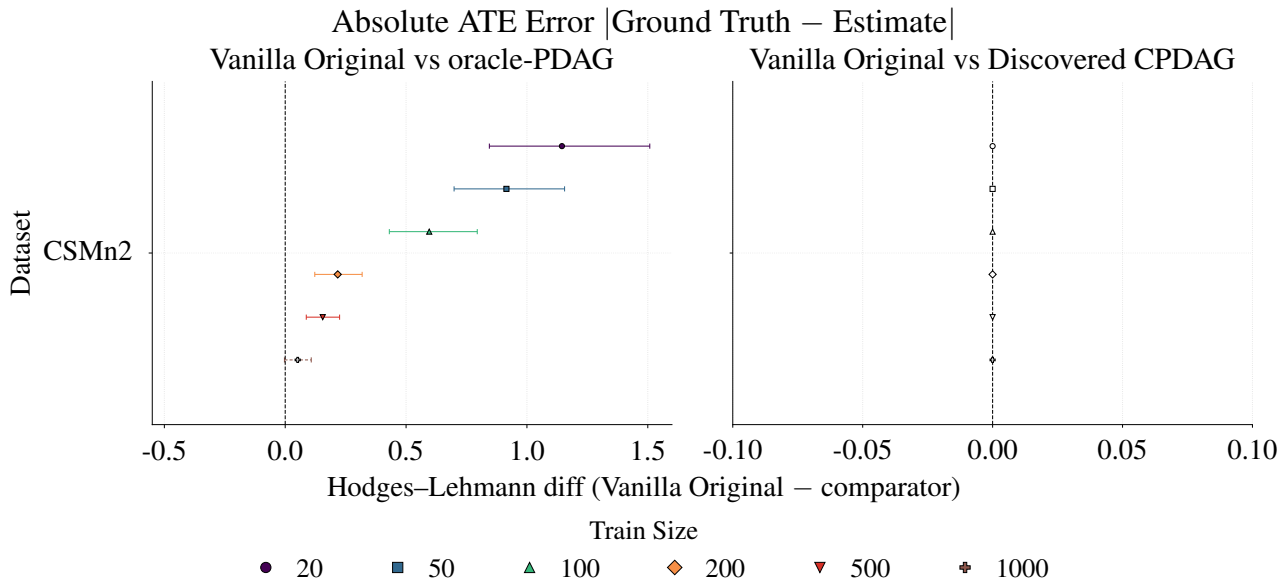

Figure A28: Hodges–Lehmann estimates of the reduction in absolute ATE error (Δ_{ATE}) when comparing vanilla TabPFN with original ordering versus PDAG-based generation on the custom SCM with $\sigma = 10^{-2}$, with oracle-PDAG (left) and discovered CPDAG (right). Positive values indicate smaller errors (closer to ground truth) for the respective method (oracle-PDAG or discovered CPDAG); negative values indicate larger errors. Filled markers with solid error bars indicate significance at $p < 0.05$ (Holm correction).

These results confirm that the quality of the discovered causal graph is critical: incorrect graphs actively harm synthetic data quality. This motivates the use of conservative algorithms such as PC combined with our hybrid conditioning strategy, which falls back to vanilla sequential generation for undirected edges rather than committing to potentially wrong directions.

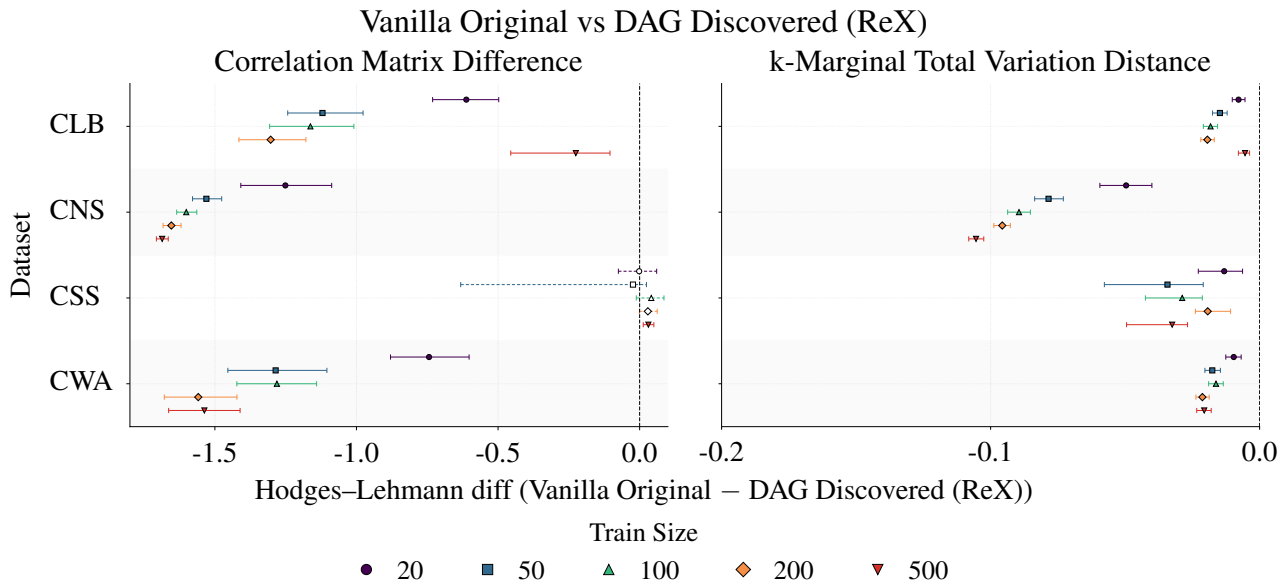

Figure A29: Hodges–Lehmann estimates comparing DAG-aware generation with ReX-discovered graphs and vanilla TabPFN with original ordering, in CMD (left) and kMTVD ($k = 2$, right). Positive values indicate that the ReX-based approach achieves lower metric values (i.e., better synthetic data quality). Filled markers with solid error bars indicate significance at $p < 0.05$ (Holm correction).

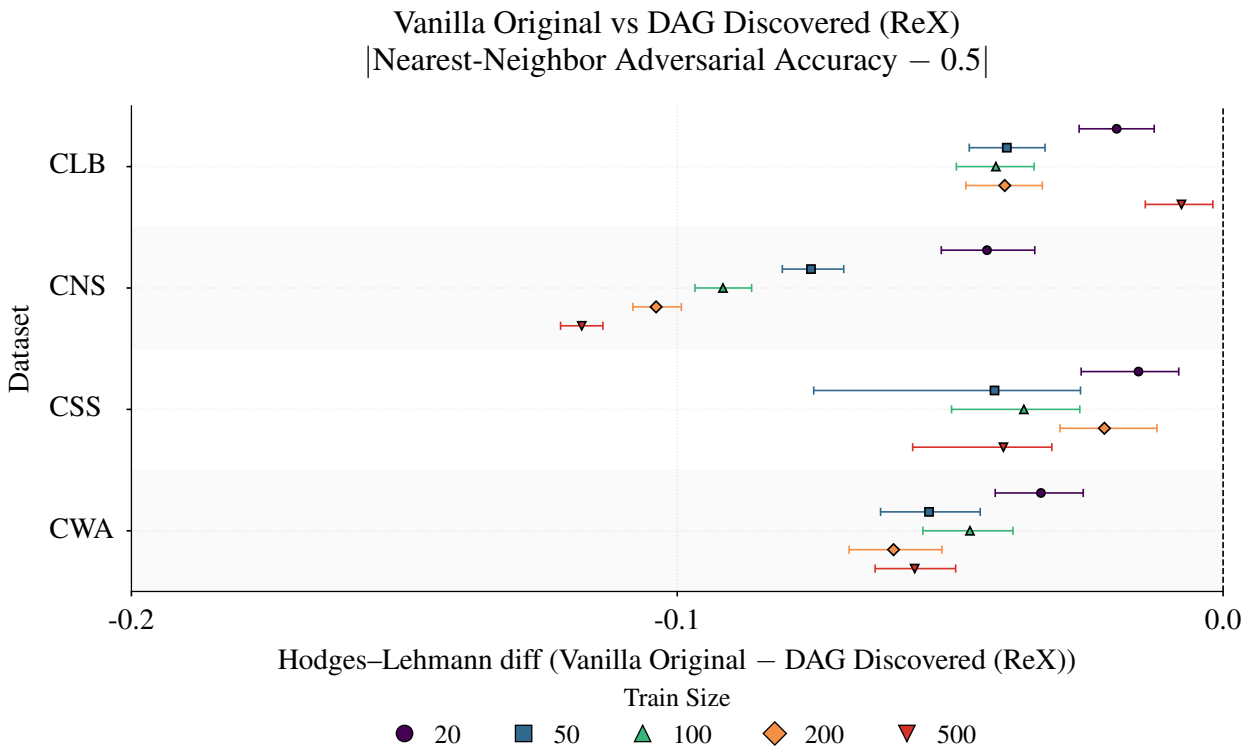

Figure A30: Hodges–Lehmann estimates comparing DAG-aware generation with ReX-discovered graphs and vanilla TabPFN with original ordering, in NNAA (reported as the distance from the ideal value, $|\mathrm{NNAA} - 0.5|$). Positive values indicate that the ReX-based approach achieves lower values (i.e., greater indistinguishability between synthetic and real data). Filled markers with solid error bars indicate significance at $p < 0.05$ (Holm correction).

Table A8: ReX graph quality on CSuite datasets, averaged over 100 repetitions. Since ReX returns fully oriented DAGs, all discovered edges are directed. Skeleton recall measures the fraction of true edges recovered. Direction precision measures the fraction of correctly oriented edges among recovered true-skeleton edges.

Dataset	N	Skel. recall	Dir. prec.
CLB	20	0.70	0.47
	50	0.73	0.45
	100	0.86	0.47
	200	0.91	0.62
	500	0.96	0.95
CNS	20	0.57	0.36
	50	0.53	0.36
	100	0.51	0.32
	200	0.51	0.68
	500	0.50	0.99
CSS	20	0.51	0.57
	50	0.47	0.65
	100	0.51	0.88
	200	0.63	0.99
	500	0.60	1.00
CWA	20	0.56	0.44
	50	0.55	0.41
	100	0.62	0.39
	200	0.58	0.51
	500	0.57	0.75