EFFICIENT DIFFERENTIABLE DISCOVERY OF CAUSAL ORDER

Mathieu Chevalley[1,2†] **Arash Mehrjou**[1,3] **Patrick Schwab**[1]
[1]GSK.ai [2]ETH Zürich [3] MPI for Intelligent Systems

ABSTRACT

Spurious correlations arise when AI models capture statistical dependencies that do not reflect the true causal structure of the underlying reality, leading to unreliable predictions and unsafe decision-making, particularly in high-stakes domains. While causal discovery methods exist to infer causal structure from data, many are computationally expensive and non-differentiable, limiting their integration into modern AI systems. In this work, we introduce a differentiable approach to causal ordering that allows causal discovery to be seamlessly incorporated as a module within existing machine learning pipelines. Our method builds upon Intersort (Chevalley et al., 2025), a score-based algorithm for discovering causal order in Directed Acyclic Graphs (DAGs) using interventional data. To enable differentiable optimization, we develop a continuous relaxation of Intersort using differentiable sorting and ranking techniques, allowing causal constraints to be directly integrated into gradient-based learning frameworks. By incorporating causal discovery as a regularizer, our approach encourages models to rely on causal relationships rather than spurious correlations, ultimately improving their robustness and trustworthiness when actions are taken based on the learned model. Empirical results demonstrate that enforcing causal order as an inductive bias enhances model generalization and interpretability, making AI systems more reliable and safer for real-world deployment.

1 INTRODUCTION

Machine learning models often exploit spurious correlations and shortcut learning strategies rather than learning true underlying causal relationships (Geirhos et al., 2020). This reliance on statistical patterns, rather than causal mechanisms, leads to poor generalization, particularly when applied to out-of-distribution data. Addressing this issue requires embedding causal reasoning directly into learning frameworks, enabling models to leverage causal structure rather than coincidental associations.

Causal discovery provides a pathway toward such robust learning by identifying the causal ordering of variables within a system. However, existing methods for causal discovery are either computationally intractable at scale or non-differentiable (Spirtes et al., 2000; Heinze-Deml et al., 2018), making them difficult to integrate into modern deep learning pipelines. Recently, Chevalley et al. (2025) introduced a novel score-based approach to infer causal order using interventional data, but its reliance on combinatorial optimization over the permutahedron renders it impractical for large-scale datasets and gradient-based learning frameworks.

To address these challenges, we introduce DiffIntersort, a differentiable and scalable reformulation of Intersort. By leveraging continuous relaxations (Cuturi, 2013), including differentiable sorting and ranking, we make causal order discovery computationally efficient and seamlessly integrable into modern machine learning models. This allows causal order to function as a regularizer, encouraging models to align with causal mechanisms rather than spurious correlations.

To evaluate the potential of causal order regularization, we integrate our regularizer into a causal discovery algorithm. Our empirical evaluations on diverse simulated datasets—including linear, random Fourier features, gene regulatory networks (GRNs) and neural network models—demonstrate that the proposed regularized algorithm significantly outperforms baseline methods such as GIES (Hauser & Bühlmann, 2012) and DCDI (Brouillard et al., 2020) on RFF and GRN data. Moreover, we demonstrate that our approach exhibits robustness across different data distributions and noise

† Correspondence to m.chevalley97@gmail.com

types. The algorithm efficiently scales with large datasets, maintaining consistent performance regardless of data size. These results underscore the potential of using our differentiable score to improve performance and generalizability in a scalable manner.

More broadly, this work contributes to the growing vision of integrating causal reasoning into deep learning. By embedding causal order as an inductive bias in differentiable models, we move beyond purely associational representations toward models that align with causal mechanisms. This approach has the potential to improve model generalization, robustness to interventions, and interpretability across domains such as genomics, neuroscience, and reinforcement learning. We believe that enabling scalable differentiable causal discovery is a key step toward imbuing machine learning models with a more principled understanding of cause and effect.

2 METHOD

We here present our methodological contribution, introducing a differentiable score on causal orders, and then describing how to use it as a regularizer. Detailed theoretical definitation and notations can be found in Appendix A.

2.1 DIFFERENTIABLE SCORE

While Intersort demonstrates cutting-edge results in discerning causal order among variables, its primary drawback is the substantial computational cost, which restricts its application to small-scale problems. The authors of the original paper acknowledged this limitation, confining their evaluation to a mere 30 nodes Chevalley et al. (2025). A covariate set of this size is prohibitively small for many real-world problems, such as those in genomics and climate change, where tens of thousands of variables are considered. We aim to enhance the scalability of Intersort through a differentiable objective function. This not only facilitates scaling to a considerably larger number of variables but even more importantly enables the integration of this algorithm in end-to-end gradient-based model training. In the subsequent sections, we initially revisit the fundamental score that underpins Intersort. Following this, we proceed to present a differentiable formulation, DiffIntersort, that addresses these shortcomings.

Intersort score– Given an observational distribution $P_X^{\mathcal{C},(\emptyset)}$ and a set of interventional distributions $\mathcal{P}_{int} = \{P_X^{\mathcal{C},do(X_k:=\tilde{N}_k)}, k \in \mathcal{I}\}$, $\mathcal{I} \subseteq V$, Chevalley et al. (2025) define the following score for a permutation π, for some statistical distance $D : \mathcal{P}(M) \times \mathcal{P}(M) \to [0, \infty)$, $\epsilon > 0$, $c > \epsilon$:

$$S(\pi, \epsilon, D, \mathcal{I}, P_X^{\mathcal{C},(\emptyset)}, \mathcal{P}_{int}, c) = \sum_{\pi(i)<\pi(j), i\in\mathcal{I}, j\in V} \left(D\left(P_{X_j}^{\mathcal{C},(\emptyset)}, P_{X_j}^{\mathcal{C},do(X_i:=\tilde{N}_i)}\right) - \epsilon\right) + c \cdot d \cdot \mathbf{1}_{D\left(P_{X_j}^{\mathcal{C},(\emptyset)}, P_{X_j}^{\mathcal{C},do(X_i:=\tilde{N}_i)}\right) > \epsilon} \tag{1}$$

Intuitively, the summation measures how well the causal order aligns with strong causal effects. The second term's rescaling by a factor of d ensures that effects exceeding ϵ will prioritize ordering constraints, enforcing $\pi(i) < \pi(j)$ by amplifying their relative importance compared to effects smaller than ϵ.

DiffIntersort score– To make Intersort differentiable, we reparameterize the ordering of variables using a potential vector $\boldsymbol{p} \in \mathbb{R}^d$, where the relative values of $\boldsymbol{p}$ determine the causal ordering. This allows us to construct a soft permutation matrix, which we optimize using the Sinkhorn operator to maintain differentiability. Specifically, we parameterize the ordering of the variable as determined by a permutation of the variables π through a potential $\boldsymbol{p} \in \mathbb{R}^d$ such that $\pi(i) < \pi(j) \iff p_i > p_j$. We write the permutation matrix associated to $\boldsymbol{p}$ as $\boldsymbol{\sigma}(\boldsymbol{p})$, which is a $d \times d$ binary matrix, where $\boldsymbol{\sigma}(\boldsymbol{p})_{ij} = 1$ if $\pi(i) = j$. We define $(\mathrm{grad}(\boldsymbol{p}))_{ij} = p_i - p_j$, which is nonnegative if and only if $\pi(i) < \pi(j)$ is in the associated topological order. Applying the element-wise Step function produces $(\mathrm{Step}(\mathrm{grad}(\boldsymbol{p})))_{ij} = 1_{p_i - p_j > 0}$ which is a matrix of the possible edges according to the potential $\boldsymbol{p}$.

We aim to rewrite the score such that it is parameterized by a potential $\boldsymbol{p}$. Building the matrix $\mathbf{D} \in \mathbb{R}^{d\times d}$ as

$$\mathbf{D}_{ij} = \begin{cases} D\left(\left(P_{X_j}^{\mathcal{C},(\emptyset)}, P_{X_j}^{\mathcal{C},do(X_i:=\tilde{N}_i)}\right) - \epsilon\right) & \\ +c \cdot d \cdot \mathbf{1}_{D\left(P_{X_j}^{\mathcal{C},(\emptyset)}, P_{X_j}^{\mathcal{C},do(X_i:=\tilde{N}_i)}\right) > \epsilon} & \text{if} \quad i \in \mathcal{I} \\ 0 & \text{if} \quad i \notin \mathcal{I} \end{cases} \tag{2}$$

we can write the score in terms of the potential instead of permutation as follows:

$$S(\boldsymbol{p}, \epsilon, D, \mathcal{I}, P_X^{\mathcal{C},(\emptyset)}, \mathcal{P}_{int}, c) = \langle \mathbf{D}, \text{Step}(\text{grad}(\boldsymbol{p})) \rangle_F . \tag{3}$$

The relationship between the potential and permutation is clarified through the following theoretical result.

Theorem 2.1. *Let* $\mathbb{P} = \arg\max_{\boldsymbol{p}} S(\boldsymbol{p}, \epsilon, D, \mathcal{I}, P_X^{\mathcal{C},(\emptyset)}, \mathcal{P}_{int}, c)$ *s.t.* $\boldsymbol{p}_i \neq \boldsymbol{p}_j \forall i, j \in V$*, be the set of potentials that maximize the score, such that no two entries of the potentials are equal.* $\Pi = \arg\max_{\pi} S(\pi, \epsilon, D, \mathcal{I}, P_X^{\mathcal{C},(\emptyset)}, \mathcal{P}_{int}, c)$ *be the set of permutations that maximize the Intersort score. For every* $\pi \in \Pi$*, there is a set* $\bar{p} \subset \mathbb{P}$ *such that* $\forall \boldsymbol{p} \in \bar{p} : \pi(i) < \pi(j) \iff p_i > p_j$*.*

The proof can be found in the appendix in Appendix F. This score is still not practically useful as it provides non-informative gradients for $\boldsymbol{p}$. To remedy this, inspired by Annadani et al. (2023) we define $\boldsymbol{L} \in \{0, 1\}^{d \times d}$ as a matrix with upper triangular part to be 1, and vector $\boldsymbol{o} = [1, \ldots, d]^T$. They propose the formulation

$$\text{Step}(\text{grad}(\boldsymbol{p})) = \boldsymbol{\sigma}(\boldsymbol{p}) \boldsymbol{L} \boldsymbol{\sigma}(\boldsymbol{p})^T. \tag{4}$$

and $\boldsymbol{\sigma}(\boldsymbol{p})$ is equivalent to the following optimization problem

$$\boldsymbol{\sigma}(\boldsymbol{p}) = \underset{\boldsymbol{\sigma}' \in \boldsymbol{\Sigma}_d}{\arg\max} \boldsymbol{p}^T (\boldsymbol{\sigma}' \boldsymbol{o}) \tag{5}$$

where $\boldsymbol{\Sigma}_d$ represents the space of all d dimensional permutation matrices. The reformulation of the permutation as an optimization problem over the set $\boldsymbol{\Sigma}_d$ can be further rewritten as

$$\boldsymbol{\sigma} = \underset{\boldsymbol{\sigma}' \in \Sigma_d}{\arg\max} \langle \boldsymbol{\sigma}', \boldsymbol{M} \rangle_F \tag{6}$$

where $M = \boldsymbol{p}\boldsymbol{o}^T$. Mena et al. (2018) demonstrated that this non-differentiable $\arg\max$ problem can be reformulated by regularizing it with the entropy and solving this smooth problem with the Sinkhorn algorithm. Specifically, they showed that $\mathcal{S}(\boldsymbol{M}/t) = \arg\max_{\boldsymbol{\sigma}' \in \mathcal{B}_d} \langle \boldsymbol{\sigma}', \boldsymbol{M} \rangle + tH(\boldsymbol{\sigma}')$, where $H(\cdot)$ denotes the entropy function and the parameter t controls the smothness of the approximation and $\mathcal{S}(\boldsymbol{M})$ is the Sinkhorn operator. The Sinkhorn operator on a matrix $\boldsymbol{M}$ involves a sequence of alternating row and column normalizations, known as Sinkhorn iterations. We refer readers to the original paper (Sinkhorn, 1964) and further applications (Adams & Zemel, 2011) of Sinkhorn operator for detailed presentation. Furthermore, we have that the regularized solution converges to the solution of Equation (6) as $t \to 0$, shown by $\lim_{t \to 0} \mathcal{S}(\boldsymbol{M}/t)$. We note here that other differentiable approximation for the permutation matrix could be used. See Appendix D.5 for a review of the differentiable sorting and ranking literature.

In practice, we approximate the limit with a value of $t > 0$ and a certain number of iterations T, which results in a differentiable and doubly stochastic matrix in the d-dimensional Birkhoff polytope $\mathcal{B}_d$, the convex hull of "hard" permutation matrices. In our experiments, we use $t = 0.05$ and $T = 500$. After applying the Sinkhorn operator to obtain a differentiable approximation of the permutation matrix, we use the Hungarian algorithm Kuhn (1955) to project it back to a valid binary permutation, ensuring consistency with the discrete causal ordering while maintaining differentiability through the straight-through estimator (Bengio et al., 2013). The resulting binary matrix is denoted as $\mathcal{S}_{\text{bin}}^T(\boldsymbol{p}\boldsymbol{o}^T/t)$ with "bin" emphasizing a binary-valued matrix. As a result, the score becomes differentiable and can be differentiated through the iterations of the Sinkhorn operator. By replacing the non-differentiable part of Equation (2) with this matrix, the complete form of the differentiable score (we call it DiffIntersort) is derived as

$$S(\boldsymbol{p}, \epsilon, D, \mathcal{I}, P_X^{\mathcal{C},(\emptyset)}, \mathcal{P}_{int}, t, T) = \left\langle \mathbf{D}, \left(\mathcal{S}_{\text{bin}}^T \left(\frac{\boldsymbol{p}\boldsymbol{o}^T}{t} \right) \boldsymbol{L} \mathcal{S}_{\text{bin}}^T \left(\frac{\boldsymbol{p}\boldsymbol{o}^T}{t} \right)^T \right) \right\rangle_F . \tag{7}$$

For the rest of the paper, we drop the subscript "bin" and use $S(\boldsymbol{p})$ for conciseness. The maximizers of the DiffIntersort score and the Intersort score are equal for $t \to 0$ and $T \to \infty$ (Theorem 2.1). The DiffIntersort score $S(\mathbf{p})$ can be maximized with respect to the potential vector $\mathbf{p}$ using gradient descent algorithms. This allows us to find the ordering of the variables that is best aligned with the interventional data, according to the statistical distances captured in $\mathbf{D}$.

2.2 DIFFINTERSORT AS A CAUSAL REGULARIZER

We now look at a potential application of our DiffIntersort score, beyond causal order learning as in Chevalley et al. (2025). In particular, we want to evaluate its potential use as a causal regularizer in a differentiable learning model. To test that idea, we now proceed to use the score as a regularizer in a simple causal discovery algorithm. We emphasize here that the goal is not to present a new causal discovery algorithm *per se*, but to evaluate the usefulness of the DiffIntersort regularizer in causal tasks. Let us consider a dataset $\mathbf{X} \in \mathbb{R}^{n \times d}$ consisting of n observations of d variables $\{X_1, X_2, \ldots, X_n\}$. Our goal is to recover the causal structure and ordering of the variables from both observational and interventional data. Let $S(\mathbf{p})$ be the DiffIntersort score, which measures the consistency of the ordering induced by $\mathbf{p}$ with the interventional data. A regularized causal discovery objective can then be formulated as the following regularized optimization problem

$$\min_{\theta, \mathbf{p}} \quad \mathcal{L}_{\text{fit}}(\theta, \mathbf{p}) + \lambda S(\mathbf{p}), \tag{8}$$

where θ represents the parameters of the causal mechanisms (e.g., weight matrices in linear models), and $\mathcal{L}_{\text{fit}}(\theta, \mathbf{p})$ is the fitting loss that measures how well the model with parameters θ explains the observed data. The regularization ensures that the potential vector $\mathbf{p}$ also minimizes the DiffIntersort score, thus enforcing a causal ordering consistent with the interventional data.The regularization parameter $\lambda > 0$ controls the trade-off between fitting the data and enforcing the causal ordering through the DiffIntersort score.

As an example, a linear causal model can be constructed as

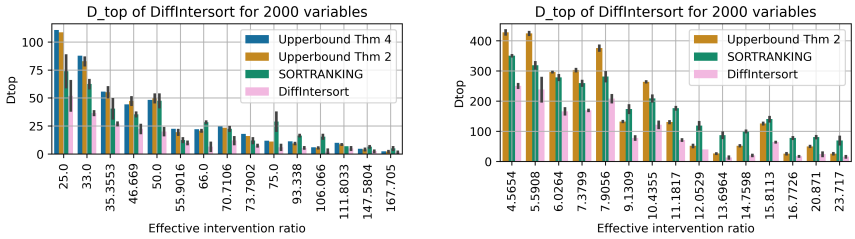
$$X_j = \sum_{i=1}^{d} W_{ji} X_i + b_j + N_j, \tag{9}$$

where W_{ji} are the entries of the weight matrix $\mathbf{W} \in \mathbb{R}^{d \times d}$, b_j is the bias term, and N_j is a noise term. To enforce the causal ordering induced by $\mathbf{p}$, we use the permuted upper-triangular matrix $\boldsymbol{M}_{\mathbf{p}} = \mathcal{S}_{\text{bin}}^T(\boldsymbol{p}\boldsymbol{o}^T/t)\boldsymbol{L}\mathcal{S}_{\text{bin}}^T(\boldsymbol{p}\boldsymbol{o}^T/t)^T$, which is a $d \times d$ matrix with $d(d-1)/2$ entries equal to 1. The matrix represents the possible locations of edges in the graph according to the causal ordering $\mathbf{p}$. By element-wise multiplication $\tilde{\mathbf{W}} = \mathbf{W} \circ \boldsymbol{M}_{\mathbf{p}}^T$, matrix $\boldsymbol{M}_{\mathbf{p}}$ acts as a mask to ensure that variable X_j may depend on variables preceding it in the causal ordering. The predicted values can be written in terms of the entries of $\tilde{\mathbf{W}}$ as $\hat{X}_j = \sum_{i=1}^{d} \tilde{\mathbf{W}}_{ji} X_i + b_j$. We described our fitting loss and learning algorithm in details in Appendix C. Our model can be extended to more complex parameterizations beyond the linear case, such as by adapting existing causal discovery methods that optimize over the permutahedron (see Appendix D.3 for a review). However, we adopt this simpler model to isolate and clearly assess the impact of regularization, independent of performance gains that may arise from a more complex model.

3 EMPIRICAL RESULTS

We next evaluate the proposed DiffIntersort differentiable score both in its effectiveness in deriving the causal order of a system, as well as it usefulness as a differentiable regularizer in a causal discovery model.

We first evaluate the DiffIntersort score in it ability to recover the causal order in simulated graphs and distance matrices. We here reproduce the experiment of (Chevalley et al., 2025). We compare the top order divergence of DiffIntersort to SORTRANKING, and to Intersort for 5 and 30 variables, and the upper-bounds of Thm 2 and Thm 4 derived in (Chevalley et al., 2025). Intersort does not scale beyond 30 variables (see Appendix I.1 for an analysis of the training time scaling). The upper bounds act as a sanity check, providing a measure of how close the approximate solution is to the true optimum of the score. We evaluate on both Erdős-Rényi distribution (Erdős et al., 1960) and scale-free network modeled by the Barabasi-Albert distribution Albert & Barabási (2002), with varying edge densities and intervention coverage. The results are reported in Figure 1 for 2000 variables and in Figures 6 and 7 for 5, 30, 100 and 1000 variables. It is crucial that our score be optimizable up to at least 2000 variables, as it is a common scale in real world datasets such as single-cell transcriptomics (Replogle et al., 2022). As studied previously, we initialize DiffIntersort with the solution of SORTRANKING, and use Adam (Kingma, 2014) to optimize the score. As observed, DiffIntersort fulfills the upper-bounds for all settings, even at large scale, which allows us to not reject the hypothesis that DiffInterosrt finds an optimum of the score. At large scale, it also outperforms SORTRANKING in



(a) Simulation ER with 2000 variables (b) Simulation SF with 2000 variables

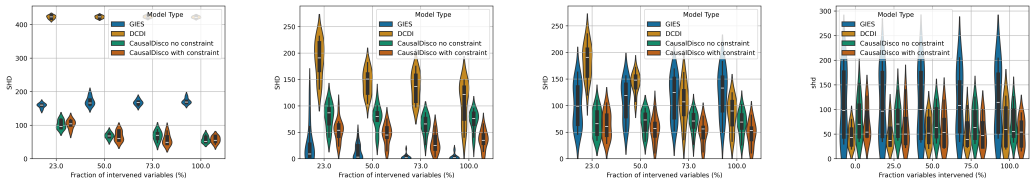
Figure 1: Simulation and comparison between the bounds of Theorems 2 and 4 of Chevalley et al. (2025) for Erdős-Rényi (ER, left) and scale-free networks (SF, right) with 2000 variables. We compare the causal order obtained by maximizing our proposed DiffIntersort score and the output of SORTRANKING. For each setting, we draw 1 graph per setting, following an ER distribution with a probability of edges per variable p_e in $\{0.0001, 0.00005, 0.00002\}$ and following a Barabasi-Albert SF distribution, with an average edge per variable in $\{1, 2, 3\}$. A setting is the tuple (p_{int}, p_e), where $p_e = \frac{2\mathrm{E}(\#\text{edges})}{d(d-1)}$ for the SF distribution. For each graph, we run the algorithm on 1 configuration, where each configuration corresponds to a draw of the targeted variables following p_{int}. We have $p_{int} \in \{0.25, 0.33, 0.5, 0.66, 0.75\}$. Settings are ordered on the x-axis following the effective intervention ratio $\frac{p_{int}}{\sqrt{p_e}}$ (Chevalley et al., 2025).

almost all settings. Those results validate our proposed approach of solving the Intersort problem in a continuous and differentiable framework, and guarantees that it is not limited by scale.

We now evaluate our method, DiffIntersort, on simulated data and compare its performance to various baseline methods. We follow the experimental setup of Chevalley et al. (2025) to ensure a fair and consistent evaluation across different domains. See Appendix G for details about the synthetic data generation. Specifically, we generate graphs from an Erdős-Rényi distribution (Erdős et al., 1960) with an expected number of edges per variable $c \in \{1, 2\}$. Data is simulated using both linear relationships and random Fourier features (RFF) additive functions to capture non-linear dependencies. In addition to these synthetic datasets, we apply our models to simulated single-cell RNA sequencing data generated using the SERGIO tool (Dibaeinia & Sinha, 2020), utilizing the code provided by Lorch et al. (2022) (MIT License, v1.0.5). We also test our method on neural network functional data following the setup of Brouillard et al. (2020), using the implementation from Nazaret et al. (2023) (MIT License, v0.1.0). To assess the impact of interventions, we vary the ratio of intervened variables in the set $25\%, 50\%, 75\%, 100\%$. All datasets are standardized based on the mean and variance of the observational data to eliminate the Varsortability artifact identified by Reisach et al. (2021). For the linear and RFF domains, the noise distribution is chosen uniformly at random from the following options: uniform Gaussian (noise scale independent of the parents), heteroscedastic Gaussian (noise scale functionally dependent on the parents), and Laplace distribution. In the neural network domain, the noise distribution is Gaussian with a fixed variance. We conduct experiments on 10 simulated datasets for each domain and each ratio of intervened variables. The observational datasets contain 5,000 samples, and each intervention dataset comprises 100 samples, mirroring the sample sizes typically found in real single-cell transcriptomics studies (Replogle et al., 2022).

We compare the performance of DiffIntersort and SORTRANKING (Chevalley et al., 2025) as measured by the top order divergence D_{top} on 100 variables in Figure 8. For the DiffIntersort score and the Intersort score, we use the same parameters as in Chevalley et al. (2025): $\epsilon = 0.3$ for linear, RFF and NN data, and $\epsilon = 0.5$ for GRN data, and $c = 1.0$. We use the Wasserstein distance (Villani et al., 2009) for the statistical metric. Results for 10 and 30 variables, additionally compared to Intersort, can be found in the appendix in Figure 10. As can be observed, the performance of the two algorithms is close. This demonstrates that the optimizing DiffIntersort can be solved at scale using continuously differentiable optimization also on realistic synthetic data.

We evaluate our regularized causal discovery method on synthetic datasets from four domains: linear structural equation models (SEMs), gene regulatory networks (GRNs), random Fourier features (RFFs), and neural networks (NNs). For each model type, we consider variable sizes of 10, 30, and 100 to assess scalability and performance across different problem dimensions. We use two evaluation metrics: Structural Hamming Distance (SHD) (Tsamardinos et al., 2006) and Structural Intervention Distance (SID) (Peters & Bühlmann, 2015) to compare inferred graphs to the true causal graphs. We compare to two baselines, namely GIES (Hauser & Bühlmann, 2012) and DCDI (Brouillard



(a) GRN, 30 vars, SHD (b) Linear, 30 vars, SHD (c) RFF, 30 vars, SHD (d) NN, 30 vars, SHD

Figure 2: Comparison of SHD (lower is better) for GRN, Linear, RFF, and Neural Network data with 30 variables. Our method (CausalDisco with and without constraint) achieves lower SHD values compared to baseline methods on GRN and RFF data. GIES outperforms on the linear data and DCDI performs slightly better on NN data.

et al., 2020). We note that those two baselines do not scale to 100 variables. For our model, we compare the performance of our proposed causal discovery model with and without the DiffIntersort constraint (i.e. $\lambda = 0$). For the regularized model, we use a high value of $\lambda = 100.0$, as we do not observe a negative effect of over-regularizing, and we thus ensure that the learn potential is close to the optimla of the DiffIntersort score (see Appendix I.2 for an analysis). We present the results for the SHD metrics at 30 variables in Figure 2. The results for 10 and 30 variables for SHD can be found in the appendix in Figure 12. The results for SID can be found in Figure 15 in the appendix.

As can be seen, the DiffIntersort constraint is consistently beneficial in terms of performance on both metrics, for all types of data and at all considered scales. This comparison validates the usefulness of inducing the interventional faithfulness inductive bias to a causal models via the DiffIntersort score. It also enforces generalizability across data settings. We expect that this approach may be applicable to other causal tasks of interest, in settings where a large set of single variable interventions are available. Compared to baselines, our model outperforms on the GRN and RFF data. GIES is the best model on linear data, and DCDI has a slightly better performance on NN data. GIES and DCDI do not scale to 100 variables but we would expect the results to be the same, as our algorithm has an F1 score that is almost unaffected by the number of variables (see Figure 9). The results on the F1 score also shows the robustness of our causal discovery model with the DiffIntersort constraint to the number of variables.

4 Conclusion

We addressed the scalability and differentiability limitations of Intersort, a score-based method for discovering causal orderings using interventional data. By reformulating the Intersort score with differentiable sorting—leveraging the Sinkhorn operator—we enabled scalable and differentiable optimization of causal orderings. This reformulation allows the Intersort score to function as a continuous regularizer in gradient-based learning frameworks, broadening its applicability to downstream causal tasks. Our approach preserves Intersort's theoretical advantages while significantly enhancing its practicality for large-scale problems. Empirical evaluations show that integrating the differentiable Intersort score into causal discovery improves performance over unregularized methods, particularly in complex settings with non-linear dependencies and large variable sets. The method remains robust across different data distributions and noise levels, scaling effectively without performance degradation.

Beyond causal discovery, our work contributes to a broader vision: integrating causal reasoning into modern machine learning pipelines. Differentiable causal ordering regularization has the potential to enhance model robustness, generalization, and interpretability. In genomics, it could help respect known gene regulatory hierarchies, reducing spurious correlations. In reinforcement learning, it could constrain policies to follow valid causal dependencies, improving sample efficiency. In interpretability research, enforcing causal order could lead to more reliable model explanations by aligning feature importance with causal influence. More broadly, this work suggests a new research direction: how can causal ordering serve as a foundation for more causally-aware deep learning models? By bridging interventional faithfulness with gradient-based learning, we move toward models that do more than capture statistical patterns—they reflect underlying causal processes. Future work may explore deeper theoretical guarantees, real-world applications, and architectures that natively integrate causal ordering constraints, shaping the future of causal learning at the intersection of representation learning and interpretability.

REFERENCES

Adams, R. P. and Zemel, R. S. Ranking via Sinkhorn propagation. *arXiv preprint arXiv:1106.1925*, 2011.

Albert, R. and Barabási, A.-L. Statistical mechanics of complex networks. *Reviews of modern physics*, 74(1):47, 2002.

Annadani, Y., Pawlowski, N., Jennings, J., Bauer, S., Zhang, C., and Gong, W. Bayesdag: Gradient-based posterior inference for causal discovery. *Advances in Neural Information Processing Systems*, 36:1738–1763, 2023.

Bengio, Y., Léonard, N., and Courville, A. Estimating or propagating gradients through stochastic neurons for conditional computation. *arXiv preprint arXiv:1308.3432*, 2013.

Blondel, M., Teboul, O., Berthet, Q., and Djolonga, J. Fast differentiable sorting and ranking. In *International Conference on Machine Learning*, pp. 950–959. PMLR, 2020.

Brouillard, P., Lachapelle, S., Lacoste, A., Lacoste-Julien, S., and Drouin, A. Differentiable causal discovery from interventional data. *Advances in Neural Information Processing Systems*, 33: 21865–21877, 2020.

Bühlmann, P., Peters, J., and Ernest, J. Cam: Causal additive models, high-dimensional order search and penalized regression. 2014.

Charpentier, B., Kibler, S., and Günnemann, S. Differentiable dag sampling. In *Proceedings of the 10th International Conference on Learning Representations (ICLR)*. 2022.

Chevalley, M., Schwab, P., and Mehrjou, A. Deriving Causal Order from Single-Variable Interventions: Guarantees & Algorithm. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=u63OVngeSp.

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

Cuturi, M. Sinkhorn distances: Lightspeed computation of optimal transport. *Advances in neural information processing systems*, 26, 2013.

Cuturi, M., Teboul, O., and Vert, J.-P. Differentiable ranking and sorting using optimal transport. *Advances in neural information processing systems*, 32, 2019.

Dibaeinia, P. and Sinha, S. Sergio: a single-cell expression simulator guided by gene regulatory networks. *Cell systems*, 11(3):252–271, 2020.

Erdős, P., Rényi, A., et al. On the evolution of random graphs. *Publ. Math. Inst. Hung. Acad. Sci*, 5 (1):17–60, 1960.

Faria, G. R. A., Martins, A., and Figueiredo, M. A. Differentiable causal discovery under latent interventions. In *Conference on Causal Learning and Reasoning*, pp. 253–274. PMLR, 2022.

Geirhos, R., Jacobsen, J.-H., Michaelis, C., Zemel, R., Brendel, W., Bethge, M., and Wichmann, F. A. Shortcut learning in deep neural networks. *Nature Machine Intelligence*, 2(11):665–673, 2020.

Grover, A., Wang, E., Zweig, A., and Ermon, S. Stochastic optimization of sorting networks via continuous relaxations. In *International Conference on Learning Representations*, 2019. URL https://openreview.net/forum?id=H1eSS3CcKX.

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

He, Y., Cui, P., Shen, Z., Xu, R., Liu, F., and Jiang, Y. Daring: Differentiable causal discovery with residual independence. In *Proceedings of the 27th ACM SIGKDD conference on knowledge discovery & data mining*, pp. 596–605, 2021.

Heinze-Deml, C., Maathuis, M. H., and Meinshausen, N. Causal structure learning. *Annual Review of Statistics and Its Application*, 5:371–391, 2018.

Kingma, D. P. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

Kuhn, H. W. The hungarian method for the assignment problem. *Naval research logistics quarterly*, 2(1-2):83–97, 1955.

Lorch, L., Sussex, S., Rothfuss, J., Krause, A., and Schölkopf, B. Amortized inference for causal structure learning. *arXiv preprint arXiv:2205.12934*, 2022.

Massidda, R., Landolfi, F., Cinquini, M., and Bacciu, D. Constraint-free structure learning with smooth acyclic orientations. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=KWO8LSUC5W.

Meinshausen, N. Causality from a distributional robustness point of view. In *2018 IEEE Data Science Workshop (DSW)*, pp. 6–10. IEEE, 2018.

Mena, G., Belanger, D., Linderman, S., and Snoek, J. Learning latent permutations with Gumbel-Sinkhorn networks. *arXiv preprint arXiv:1802.08665*, 2018.

Montagna, F., Mastakouri, A., Eulig, E., Noceti, N., Rosasco, L., Janzing, D., Aragam, B., and Locatello, F. Assumption violations in causal discovery and the robustness of score matching. In Oh, A., Naumann, T., Globerson, A., Saenko, K., Hardt, M., and Levine, S. (eds.), *Advances in Neural Information Processing Systems*, volume 36, pp. 47339–47378. Curran Associates, Inc., 2023a. URL https://proceedings.neurips.cc/paper_files/paper/2023/file/93ed74938a54a73b5e4c52bbaf42ca8e-Paper-Conference.pdf.

Montagna, F., Noceti, N., Rosasco, L., Zhang, K., and Locatello, F. Causal discovery with score matching on additive models with arbitrary noise. In van der Schaar, M., Zhang, C., and Janzing, D. (eds.), *Proceedings of the Second Conference on Causal Learning and Reasoning*, volume 213 of *Proceedings of Machine Learning Research*, pp. 726–751. PMLR, 11–14 Apr 2023b. URL https://proceedings.mlr.press/v213/montagna23a.html.

Nazaret, A., Hong, J., Azizi, E., and Blei, D. Stable differentiable causal discovery. *arXiv preprint arXiv:2311.10263*, 2023.

Pearl, J. *Causality*. Cambridge university press, 2009.

Peters, J. and Bühlmann, P. Structural intervention distance for evaluating causal graphs. *Neural computation*, 27(3):771–799, 2015.

Peters, J., Janzing, D., and Schölkopf, B. *Elements of causal inference: foundations and learning algorithms*. The MIT Press, 2017.

Petersen, F., Borgelt, C., Kuehne, H., and Deussen, O. Differentiable sorting networks for scalable sorting and ranking supervision. In *International Conference on Machine Learning*, pp. 8546–8555. PMLR, 2021.

Petersen, F., Borgelt, C., Kuehne, H., and Deussen, O. Monotonic differentiable sorting networks. In *International Conference on Learning Representations*, 2022. URL https://openreview.net/forum?id=IcUWShptD7d.

Prillo, S. and Eisenschlos, J. Softsort: A continuous relaxation for the argsort operator. In *International Conference on Machine Learning*, pp. 7793–7802. PMLR, 2020.

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

Replogle, J. M., Saunders, R. A., Pogson, A. N., Hussmann, J. A., Lenail, A., Guna, A., Mascibroda, L., Wagner, E. J., Adelman, K., Lithwick-Yanai, G., et al. Mapping information-rich genotype-phenotype landscapes with genome-scale perturb-seq. *Cell*, 185(14):2559–2575, 2022.

Rolland, P., Cevher, V., Kleindessner, M., Russell, C., Janzing, D., Schölkopf, B., and Locatello, F. Score matching enables causal discovery of nonlinear additive noise models. In *International Conference on Machine Learning*, pp. 18741–18753. PMLR, 2022.

Shahverdikondori, M., Mokhtarian, E., and Kiyavash, N. QWO: Speeding up permutation-based causal discovery in liGAMs. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*, 2024. URL https://openreview.net/forum?id=BptJGaPn9C.

Shen, X., Bühlmann, P., and Taeb, A. Causality-oriented robustness: exploiting general additive interventions. *arXiv preprint arXiv:2307.10299*, 2023.

Shimizu, S., Hoyer, P. O., Hyvärinen, A., Kerminen, A., and Jordan, M. A linear non-gaussian acyclic model for causal discovery. *Journal of Machine Learning Research*, 7(10), 2006.

Shimizu, S., Inazumi, T., Sogawa, Y., Hyvarinen, A., Kawahara, Y., Washio, T., Hoyer, P. O., Bollen, K., and Hoyer, P. Directlingam: A direct method for learning a linear non-gaussian structural equation model. *Journal of Machine Learning Research-JMLR*, 12(Apr):1225–1248, 2011.

Sinkhorn, R. A relationship between arbitrary positive matrices and doubly stochastic matrices. *The annals of mathematical statistics*, 35(2):876–879, 1964.

Solus, L., Wang, Y., and Uhler, C. Consistency guarantees for greedy permutation-based causal inference algorithms. *Biometrika*, 108(4):795–814, 2021.

Spirtes, P. An anytime algorithm for causal inference. In Richardson, T. S. and Jaakkola, T. S. (eds.), *Proceedings of the Eighth International Workshop on Artificial Intelligence and Statistics*, volume R3 of *Proceedings of Machine Learning Research*, pp. 278–285. PMLR, 04–07 Jan 2001. URL https://proceedings.mlr.press/r3/spirtes01a.html.

Spirtes, P., Glymour, C. N., Scheines, R., and Heckerman, D. *Causation, prediction, and search*. MIT press, 2000.

Squires, C., Wang, Y., and Uhler, C. Permutation-based causal structure learning with unknown intervention targets. In *Conference on Uncertainty in Artificial Intelligence*, pp. 1039–1048. PMLR, 2020.

Tsamardinos, I., Brown, L. E., and Aliferis, C. F. The max-min hill-climbing bayesian network structure learning algorithm. *Machine learning*, 65:31–78, 2006.

Villani, C. et al. *Optimal transport: old and new*, volume 338. Springer, 2009.

Wang, Y., Solus, L., Yang, K., and Uhler, C. Permutation-based causal inference algorithms with interventions. *Advances in Neural Information Processing Systems*, 30, 2017.

Waxman, D., Butler, K., and Djurić, P. M. Dagma-dce: Interpretable, non-parametric differentiable causal discovery. *IEEE Open Journal of Signal Processing*, 2024.

Wei, D., Gao, T., and Yu, Y. Dags with no fears: A closer look at continuous optimization for learning bayesian networks. In Larochelle, H., Ranzato, M., Hadsell, R., Balcan, M., and Lin, H. (eds.), *Advances in Neural Information Processing Systems*, volume 33, pp. 3895–3906. Curran Associates, Inc., 2020. URL https://proceedings.neurips.cc/paper_files/paper/2020/file/28a7602724ba16600d5ccc644c19bf18-Paper.pdf.

Yang, K., Katcoff, A., and Uhler, C. Characterizing and learning equivalence classes of causal dags under interventions. In *International Conference on Machine Learning*, pp. 5541–5550. PMLR, 2018.

Yu, Y., Gao, T., Yin, N., and Ji, Q. Dags with no curl: An efficient dag structure learning approach. In Meila, M. and Zhang, T. (eds.), *Proceedings of the 38th International Conference on Machine Learning*, volume 139 of *Proceedings of Machine Learning Research*, pp. 12156–12166. PMLR, 18–24 Jul 2021. URL https://proceedings.mlr.press/v139/yu21a.html.

Zantedeschi, V., Franceschi, L., Kaddour, J., Kusner, M., and Niculae, V. DAG learning on the permutahedron. In *The Eleventh International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=m9LCdYgN8-6.

Zheng, X., Aragam, B., Ravikumar, P., and Xing, E. P. DAGs with NO TEARS: Continuous Optimization for Structure Learning. In *Advances in Neural Information Processing Systems*, 2018.

Zheng, X., Dan, C., Aragam, B., Ravikumar, P., and Xing, E. P. Learning sparse nonparametric DAGs. In *International Conference on Artificial Intelligence and Statistics*, 2020.

A Definitions and Assumptions

In this section, we introduce notations and definitions that are used throughout the paper inspired by (Pearl, 2009; Peters et al., 2017).

Let $(\mathcal{M}, d)$ be a metric space, and let $\mathcal{P}(\mathcal{M})$ denote the set of probability measures over $\mathcal{M}$. We define D to be a statistical distance function $D : \mathcal{P}(\mathcal{M}) \times \mathcal{P}(\mathcal{M}) \to [0, \infty)$ that measures the divergence between probability distributions on $\mathcal{M}$. Consider a set of d random variables $\mathbf{X} = (X_1, X_2, \ldots, X_d)$ indexed by $V = \{1, 2, \ldots, d\}$, with joint distribution $P_{\mathbf{X}}$. We denote the marginal distribution of each variable as P_{X_i} for $i \in V$. A causal graph is a tuple $\mathcal{G} = (V, E)$ of nodes and edges that form a Directed Acyclic Graph (DAG), where V is the set of nodes (variables), and $E \subseteq V \times V$ is the set of directed edges representing causal relationships. An edge $(i, j) \in E$ indicates that variable X_i is a direct cause of variable X_j. Let $\mathbf{A}^{\mathcal{G}}$ be the adjacency matrix of $\mathcal{G}$, where $\mathbf{A}^{\mathcal{G}}_{ij} = 1$ if $(i, j) \in E$, and $\mathbf{A}^{\mathcal{G}}_{ij} = 0$ otherwise. For each node $j \in V$, the set of parents $\mathrm{Pa}(j)$ consists of all nodes with edges pointing to j, i.e., $\mathrm{Pa}(j) = \{i \in V \mid (i, j) \in E\}$. We denote the set of descendants of node i as $\mathrm{De}_{\mathcal{G}}(i)$, which includes all nodes reachable from i via directed paths. Similarly, the set of ancestors of i is denoted as $\mathrm{An}_{\mathcal{G}}(i)$.

An SCM $\mathcal{C} = (\mathbf{S}, P_N)$ consists of a set of structural assignments $\mathbf{S}$ and a joint distribution over exogenous noise variables P_N. Each variable X_j is assigned via a structural equation:

$$X_j = f_j\left(\mathbf{X}_{\mathrm{Pa}(j)}, N_j\right),$$

where N_j is an exogenous noise variable, and $\mathbf{X}_{\mathrm{Pa}(j)}$ are the parent variables of X_j. The exogenous variable need not be independent, potentially introducing confounding.

In our work, we focus on interventions that modify the structural assignments of certain variables. Specifically, we consider interventions where the structural assignment of a variable X_k is replaced by a new exogenous variable $\tilde{N}_k$, independent of its parents $X_k = \tilde{N}_k$.

Definition A.1. A *causal order* of the graph $\mathcal{G} = (V, E)$ is a permutation $\pi : V \to V$ such that for any edge $(i, j) \in E$, we have $\pi(i) < \pi(j)$. This ensures that causes precede their effects in the ordering (Peters et al., 2017). Multiple causal orders may satisfy the same DAG.

Since $\mathcal{G}$ is acyclic, at least one causal order exists, though it may not be unique. We denote the set of all valid causal orders consistent with $\mathcal{G}$ as Π^*.

Definition A.2. To measure the discrepancy between a proposed permutation π and the causal structure of the graph $\mathcal{G}$, we use the *top order divergence* (Rolland et al., 2022), defined as:

$$D_{\text{top}}(\mathcal{G}, \pi) = \sum_{\pi(i) > \pi(j)} \mathbf{A}^{\mathcal{G}}_{ij}.$$

This divergence counts the number of edges that are inconsistent with the ordering π, i.e., edges where the cause appears after the effect in the proposed ordering. For any causal order $\pi^* \in \Pi^*$, we have $D_{\text{top}}(\mathcal{G}, \pi^*) = 0$.

Assumption A.3 (Interventional Faithfulness)**.** Interventional faithfulness (Chevalley et al., 2025) assumes that all directed paths in the causal graph manifest as significant changes in the distribution under interventions as measured by a statistical distance. Specifically, if intervening on variable X_i leads to a detectable change in the distribution of variable X_j, then there must be a directed path from X_i to X_j in the causal graph $\mathcal{G}$. Conversely, if there is no directed path from X_i to X_j, then intervening on X_i does not affect the distribution of X_j beyond a significance threshold ϵ.

Interventional faithfulness allows us to use statistical divergences between marginal observational and interventional distributions to infer the causal ordering of variables. By assuming interventional faithfulness, we can relate changes observed under interventions to the underlying causal structure.

B Computational Complexity of DiffInterSort

DiffIntersort scales efficiently compared to combinatorial search methods. Its primary computational cost comes from (1) *Differentiable Score Computation:* $O(d^2)$ due to matrix operations; (2) *Sinkhorn*

Algorithm 1 DiffIntersort Causal Discovery Algorithm

for epoch $\leftarrow 1$ **to** max_epochs **do**
  $\mathcal{L}_{\text{DiffIntersort}} \leftarrow \lambda_2 S(\mathbf{p})$
  for each mini-batch $\mathbf{X}_{\text{batch}}$, interventions$_{\text{batch}}$ **do**
    Forward Pass: Compute predictions $\hat{\mathbf{X}} \leftarrow f(\mathbf{X}_{\text{batch}}; \theta, \mathbf{p})$
    Compute Fitting Loss $\mathcal{L}_{\text{fit}}(\theta, \mathbf{p})$:
      Compute MAE for observational data:
      $\mathcal{L}_{\text{obs}} \leftarrow \frac{1}{n^0} \sum_{i=1}^{n^0} \ell(\mathbf{x}_i, \hat{\mathbf{x}}_i)$
      Compute MAE for interventional data and environment invariance:
      $\mathcal{L}_{\text{int}} \leftarrow \gamma \sum_{e \in \mathcal{E}} \omega^e \left(\frac{1}{n^e} \sum_{i=1}^{n^e} \ell^e(\mathbf{x}_i, \hat{\mathbf{x}}_i) - \mathcal{L}_{\text{obs}} \right)$
      Total fitting loss:
      $\mathcal{L}_{\text{fit}} \leftarrow \mathcal{L}_{\text{obs}} + \mathcal{L}_{\text{int}}$
    Compute Regularization Loss:
      $\mathcal{L}_{\text{L1}} \leftarrow \lambda_1 \|\mathbf{W}\|_1$
    Compute Total Loss: $\mathcal{L} \leftarrow \mathcal{L}_{\text{fit}} + \mathcal{L}_{\text{L1}} + \mathcal{L}_{\text{DiffIntersort}}$
    Backward Pass: Compute gradients $\nabla_{\theta, \mathbf{p}} \mathcal{L}$
    Update Parameters: $\theta, \mathbf{p} \leftarrow \text{Optimizer}(\theta, \mathbf{p})$
  end for
end for
Return Causal edges and variable ordering

Operator: $O(d^2T)$, where T is the number of iterations (typically $T = 500$); (3) *Hungarian Algorithm:* Worst-case $O(d^3)$, but empirically $O(d^2)$ due to initialization with a near-optimal solution. Thus, the practical complexity is closer to $O(d^2T)$, significantly outperforming combinatorial methods like Intersort. Experiments confirm scalability to thousands of variables, making DiffIntersort well-suited for genomics and other high-dimensional applications.

C Detailed description of the Causal Discovery loss

Inspired by the fitting loss in Shen et al. (2023), we define the fitting loss $\mathcal{L}_{\text{fit}}(\theta)$ as:

$$\mathcal{L}_{\text{fit}}(\theta, \mathbf{p}) = \frac{1}{n^0} \sum_{i=1}^{n^0} \ell(\mathbf{x}_i, \hat{\mathbf{x}}_i; \theta, \mathbf{p}) + \gamma \sum_{e \in \mathcal{E}} \omega^e \left(\frac{1}{n^e} \sum_{i=1}^{n^e} \ell^e(\mathbf{x}_i, \hat{\mathbf{x}}_i; \theta, \mathbf{p}) - \frac{1}{n^0} \sum_{i=1}^{n^0} \ell^0(\mathbf{x}_i, \hat{\mathbf{x}}_i; \theta, \mathbf{p}) \right), \tag{10}$$

where $\ell(\mathbf{x}_i, \hat{\mathbf{x}}_i; \theta)$ is the mean absolute error (MAE) loss function for observational sample i, $\ell^e(\mathbf{x}_i, \hat{\mathbf{x}}_i; \theta)$ is the loss for samples in environment e. In our case, an environment corresponds to an intervention on one variable. $\gamma \geq 0$ is a parameter controlling the emphasis on invariance across environments. We use $\gamma = 0.5$. ω^e are weights for each environment. We set $\omega^e = \frac{1}{|\mathcal{E}|}$. $\mathcal{E}$ is the set of environments, with $0 \in \mathcal{E}$ denoting the reference observational environment. n^e is the number of samples in environment $e \in \mathcal{E}$. The loss encourages the model to fit the data in the reference environment while penalizing deviations in performance across different environments, promoting robustness to interventions. This should encourage the weights to corresponds to the true causal weights, as the equivalence between robustness and causality is well established (Meinshausen, 2018).

Combining the fitting loss and the regularization terms, the final loss function is:

$$\begin{aligned}\mathcal{L}(\theta, \mathbf{p}) =& \frac{1}{n^0} \sum_{i=1}^{n^0} \ell(\mathbf{x}_i, \hat{\mathbf{x}}_i; \theta, \mathbf{p}) \\ &+ \gamma \sum_{e \in \mathcal{E}} \omega^e \left(\frac{1}{n^e} \sum_{i=1}^{n^e} \ell^e(\mathbf{x}_i, \hat{\mathbf{x}}_i; \theta, \mathbf{p}) - \frac{1}{n^0} \sum_{i=1}^{n^0} \ell^0(\mathbf{x}_i, \hat{\mathbf{x}}_i; \theta, \mathbf{p}) \right) \\ &+ \lambda_1 \left\| \mathbf{W} \right\|_1 + \lambda_2 S(\mathbf{p}).\end{aligned} \tag{11}$$

This loss function includes all the components: (1) *Data Fitting Loss*: Measures how well the model predicts the observed data, adjusted for interventions; (2) *Environment Invariance Penalty*: Encourages the model to have consistent performance across different environments; (3) L_1 *Regularization*: Promotes sparsity in the weight matrix $\mathbf{W}$; (4) *DiffIntersort Regularization*: Incorporates interventional faithfulness by penalizing with the DiffIntersort score $S(\mathbf{p})$ Equation (7). We also note that no acyclicity constraint is needed as the weight matrix is enforced to be acyclic through the masking based on the causal order $\mathbf{p}$.

The full learning algorithm is described in Algorithm 1.

D Related Work

D.1 Causal Discovery Methods

Causal discovery aims to identify cause-and-effect relationships among variables, typically represented as Directed Acyclic Graphs (DAGs). Various methodologies have been developed to infer these structures from data, primarily categorized into constraint-based, score-based, and hybrid approaches.

D.1.1 Constraint-Based Methods

These methods rely on statistical tests to assess conditional independencies in the data. The PC algorithm (Spirtes et al., 2000) is a prominent example that iteratively removes edges between variables based on conditional independence tests, constructing a skeleton of the causal graph and then orienting the edges to form a DAG. Its extension, the Fast Causal Inference (FCI) (Spirtes, 2001) algorithm, accounts for latent confounders and selection bias, providing a more robust framework in complex scenario.

D.1.2 Score-Based Methods

These approaches assign scores to different graph structures based on how well they fit the data and search for the graph with the optimal score. The Greedy Equivalence Search (GES) (Chickering, 2002) algorithm begins with an empty graph and incrementally adds edges to maximize a chosen score, such as the Bayesian Information Criterion (BIC). The Greedy Interventional Equivalence Search (GIES) (Hauser & Bühlmann, 2012) extends GES by incorporating interventional data, enhancing its ability to uncover causal directions that are indistinguishable using observational data alone.

D.1.3 Functional Causal Model-Based Methods

These methods assume specific functional relationships between variables. For instance, the Linear Non-Gaussian Acyclic Model (LiNGAM) (Shimizu et al., 2006) assumes that the data-generating process is linear with non-Gaussian noise, enabling the identification of causal directions that are not identifiable under Gaussian assumptions.

D.2 Differentiable Causal Discovery Methods

Differentiable causal discovery methods have gained prominence due to their ability to integrate seamlessly with gradient-based optimization frameworks. A notable example is the NOTEARS

(Zheng et al., 2018; 2020) algorithm, which formulates the structure learning problem as a continuous optimization task. It introduces a smooth characterization of the acyclicity constraint, enabling the use of standard numerical optimization techniques. However, enforcing this acyclicity constraint can be computationally intensive, especially for large-scale problems.

Building upon NOTEARS, several methods have been proposed to improve efficiency and scalability. For instance, DAGs with No Fears (Wei et al., 2020) re-examines the continuous optimization framework, addressing limitations in the original formulation and proposing enhancements to the optimization process. Similarly, DAG-NoCurl (Yu et al., 2021) introduces a no-curl constraint to ensure acyclicity, offering an alternative approach to the acyclicity enforcement in NOTEARS. Additionally, the Differentiable Causal Discovery from Interventional Data (DCDI) (Brouillard et al., 2020) method leverages interventional data to enhance identifiability and employs neural networks to model complex causal relationships. Several other prominent differentiable methods have been proposed in this line of research, including Differentiable Causal Discovery from Interventional Data (DCDI) Brouillard et al. (2020), Stable Differentiable Causal Discovery (SDCD) (Nazaret et al., 2023), Differentiable Causal Discovery Under Latent Interventions Faria et al. (2022), Differentiable Causal Discovery with Residual Independence (DARING) He et al. (2021), and Dagma-DCE Waxman et al. (2024).

Despite these advancements, enforcing acyclicity constraints remains a challenge, often leading to increased computational complexity and potential convergence issues.

D.3 Permutation-Based Methods

To address the challenges associated with acyclicity constraints, permutation-based methods have been developed, focusing on learning over the topological ordering of the variables. By optimizing over the permutahedron—the convex hull of all permutation vectors—these methods inherently ensure acyclicity without the need for explicit constraints.

Key developments include:

- **Greedy Sparsest Permutation (GSP)**: This method associates a score to each permutation of variables and performs a greedy search to find the permutation that leads to the sparsest DAG, effectively learning the causal structure by identifying the optimal variable ordering Solus et al. (2021).
- **Permutation-Based Causal Inference with Interventions**: Extending GSP, IGSP (Wang et al., 2017; Yang et al., 2018; Squires et al., 2020) incorporates interventional data into the permutation-based framework, enhancing the identifiability of causal structures by leveraging additional experimental information.
- **DAG Learning on the Permutahedron**: This method formulates DAG learning as an optimization problem over the permutahedron, guaranteeing the learning of a valid DAG and allowing for end-to-end training without preprocessing steps Zantedeschi et al. (2023).
- **COSMO**: Massidda et al. (2024) introduced COSMO, a constraint-free continuous optimization scheme for acyclic structure learning. At its core, COSMO employs a novel differentiable approximation of an orientation matrix parameterized by a single priority vector, enabling the learning of a smooth orientation matrix and the resulting acyclic adjacency matrix without explicitly evaluating acyclicity at any step. This approach ensures convergence to an acyclic solution and offers improved scalability due to its asymptotically faster computations.
- **QWO**: Shahverdikondori et al. (2024) introduced a novel method to enhance the efficiency of computing the optimal DAG for a given permutation, significantly speeding up permutation-based causal discovery in Linear Gaussian Acyclic Models.
- **BayesDAG**: Annadani et al. (2023) introduced BayesDAG, a framework that employs gradient-based posterior inference for causal discovery. This method utilizes stochastic gradient Markov Chain Monte Carlo (SG-MCMC) and variational inference to sample from the posterior distribution of DAGs, allowing for uncertainty quantification in the inferred causal structures. BayesDAG is applicable to both linear and nonlinear causal models, providing flexibility in modeling complex data-generating processes.

- **DP-DAG**: DP-DAG (Charpentier et al., 2022) is a differentiable probabilistic model designed for efficient DAG sampling suitable for continuous optimization. The method samples a DAG by first determining a linear ordering of nodes and then sampling edges consistent with this ordering. This approach ensures the generation of valid DAGs throughout the training process and eliminates the need for complex augmented Lagrangian optimization schemes. Additionally, the authors propose VI-DP-DAG, which combines DP-DAG with variational inference to approximate the posterior probability over DAG edges given observed data.

In summary, while differentiable causal discovery methods like NOTEARS have advanced the field by enabling continuous optimization, permutation-based methods provide a compelling alternative by focusing on learning variable orderings. This approach inherently satisfies acyclicity, offering advantages in efficiency and scalability. This approach has benefited from advances in differentiable ranking and sorting, allowing continous and differentiable relaxations of causal discovery over the permutahedron.

D.4 Causal Ordering discovery

Causal ordering, which involves finding the causal order of the variables, is a foundational step in causal discovery. Even though it does not identify the exact graph, it can facilitate subsequent edge recovery using techniques like penalized regression (Bühlmann et al., 2014; Shimizu et al., 2011). Moreover, even without full causal graph identification, a valid causal order allows for the construction of a fully connected graph that accurately describes interventional distributions (Peters & Bühlmann, 2015; Bühlmann et al., 2014).

Recent studies have highlighted that sorting variables by variance can recover causal order in simulated datasets (Reisach et al., 2021). Building on this insight, several algorithms have been developed to infer causal order from observational data, employing methods such as score matching (Rolland et al., 2022; Montagna et al., 2023a;b).

In the context of interventional data, proposed a rule-based algorithm to infer causal order. Intersort improved on this idea by introducing a score-based method to derive the causal, which leverages optimization tools for enhanced scalability. Additionally, Intersort provides theoretical results that upper-bound the expected error of the algorithm, particularly in scenarios where only a subset of variables is intervened upon.

D.5 Differentiable sorting and ranking

Differentiable sorting and ranking techniques have emerged as essential tools for integrating ranking and sorting operations into end-to-end learning frameworks. Traditional sorting operators, being non-differentiable, posed significant challenges in gradient-based optimization. To address this, Grover et al. (2019) introduced NeuralSort, a continuous relaxation of the sorting operator, enabling differentiable approximations of permutation matrices . Cuturi et al. (2019) further advanced this field by framing ranking and sorting as optimal transport problems, employing entropic regularization and Sinkhorn iterations to approximate ranks and sorted values.

Subsequent works have explored improvements in efficiency and applicability. Prillo & Eisenschlos (2020) proposed SoftSort, a simple yet effective continuous relaxation of the argsort operator, offering state-of-the-art performance with computational efficiency. Blondel et al. (2020) introduced fast differentiable sorting and ranking operators with $\mathcal{O}(n \log n)$ complexity, achieved by projecting inputs onto the permutahedron and employing isotonic optimization.

Extensions have also focused on stability and scalability. Petersen et al. (2021) presented differentiable sorting networks by relaxing conditional swap operations, addressing challenges such as vanishing gradients in large datasets. Building on this, Petersen et al. (2022) proposed monotonic differentiable sorting networks, introducing sigmoid-based relaxations to ensure gradient correctness and robustness.

E Intersort Optimization Details

Chevalley et al. (2025) propose the Intersort algorithm to optimize the score. Specifically, the Intersort algorithm consists of two steps. The first step, SORTRANKING, finds an initial by ordering the

observed statistical $D\left(P_{X_j}^{\mathcal{C},(\emptyset)}, P_{X_j}^{\mathcal{C},do(X_i:=\tilde{N}_i)}\right)$ distances from highest to lowest, adding an edge to the solution if it does not create a cycle. When the significance threshold ϵ is reached, the algorithm stops and returns the topological order of the built graph as an initial solution for the second step. This runtime complexity of this algorithm is $\mathcal{O}(d \cdot |\mathcal{I}| \log(d \cdot |\mathcal{I}|))$. The second step, LOCALSEARCH, iteratively searches in a close neighborhood in permutation space for a higher scoring solution, until the score cannot be improved anymore. For each iteration, the complexity is $\mathcal{O}(d^2)$, and the number of iterations is approximately $\mathcal{O}(d)$.

F PROOFS

Proof of Theorem 2.1. First, let us recall that we have $\boldsymbol{p} \in \mathbb{R}^d$, and $\pi \in \{0, 1, \ldots, d\}^d$, where $\forall i, j \in \{0, 1, \ldots, d\}, \pi_i \neq \pi_j$. We thus trivially have that any permutation π can be represented by a potential $\boldsymbol{p}$, by $\boldsymbol{p}_i = -\pi_i \forall i \in \{0, 1, \ldots, d\}$. We now have to prove that if $\pi \in \Pi$, then the corresponding potential $\boldsymbol{p}_\pi \in \mathbb{P}$. Let $s = \max_\pi S(\pi, \epsilon, D, \mathcal{I}, P_X^{\mathcal{C},(\emptyset)}, \mathcal{P}_{int}, c)$ be the maximum achievable score. The sum of the score is over the elements of $\mathbf{D}_{ij}$ where $\pi_i < \pi_j$. For all these pairs of indices, we also have that $p_{\pi_i} > p_{\pi_j}$, and thus for all those pairs, we also have $(\text{Step}(\text{grad}(\boldsymbol{p}_\pi)))_{ij} = 1$. This exactly corresponds to the elements that are non-zero and thus contribute to the sum in Equation (3). Thus we have that $S(\boldsymbol{p}_\pi) = s$, and as such $\boldsymbol{p}_\pi \in \mathbb{P}$, which concludes the proof. □

G DETAILS OF EMPIRICAL EVALUATION

We here describe the setting of our synthetic evaluation. We follow the setup of Chevalley et al. (2025), which was based on the setup and implementation of Lorch et al. (2022).

G.1 LINEAR AND RANDOM FOURIER FEATURE (RFF) DOMAINS

Each causal variable x_j is modeled in terms of its parents $x_{\text{pa}(j)}$ using the equation:

$$x_j \leftarrow f_j(x_{\text{Pa}_j^{\mathcal{G}}}, \epsilon_j) = f_j(x_{\text{Pa}_j^{\mathcal{G}}}) + h_j(x_{\text{Pa}_j^{\mathcal{G}}})\epsilon_j,$$

where ϵ_j denotes additive noise, potentially heteroscedastic. The noise scale $h_j(x)$ is specified as:

$$h_j(x) = \log(1 + \exp(g_j(x))),$$

with $g_j(x)$ being a nonlinear function. For heteroscedastic noise, random Fourier features are used, configured with a length scale of 10.0 and output scale of 2.0.

Interventions fix the value of the intervened variable to a constant drawn from a signed Uniform distribution over $[1.0, 5.0]$.

G.1.1 DOMAIN-SPECIFIC MODELING

- **Linear Domain:** Causal functions are linear transformations:

  $$f_j(x_{\text{Pa}_j^{\mathcal{G}}}) = w_j^\top x_{\text{Pa}_j^{\mathcal{G}}} + b_j,$$

  where w_j and b_j are sampled independently. Specifically, w_j is drawn from a signed Uniform distribution over $[1, 3]$, and b_j is sampled from a Uniform distribution over $[-3, 3]$.

- **RFF Domain:** Causal functions are modeled using a Gaussian Process (GP) approximation via random Fourier features:

  $$f_j(x_{\text{Pa}_j^{\mathcal{G}}}) = b_j + c_j \sqrt{\frac{2}{M}} \sum_{m=1}^{M} \alpha^{(m)} \cos\left(\frac{1}{\ell_j}\omega^{(m)} \cdot x_{\text{Pa}_j^{\mathcal{G}}} + \delta^{(m)}\right),$$

  where $\alpha^{(m)} \sim \mathcal{N}(0, 1)$, $\omega^{(m)} \sim \mathcal{N}(0, \mathbf{I})$, and $\delta^{(m)} \sim \text{Uniform}(0, 2\pi)$. Parameters b_j, c_j, and ℓ_j are sampled independently: ℓ_j from Uniform([7.0, 10.0]), c_j from Uniform([10.0, 20.0]), b_j from Uniform([-3, 3]), and $M = 100$.

G.2 SIMULATION OF SINGLE-CELL GENE EXPRESSION DATA

Realistic single-cell RNA sequencing data is generated using the SERGIO simulator (Dibaeinia & Sinha, 2020). SERGIO models gene expression as snapshots from the steady state of a dynamical system governed by the chemical Langevin equation. Gene interactions are defined by a causal graph G, with variability introduced through master regulator (MR) rates. Cell types are distinguished by differences in MR rates, which affect noise and expression profiles.

G.2.1 SIMULATION PARAMETERS

Simulations cover $c = 5$ cell types and d genes. Key parameters include:

- Interaction strengths k: Uniform($[1.0, 5.0]$),
- MR production rates b: Uniform($[1.0, 3.0]$),
- Hill coefficients: $\gamma = 2.0$,
- Decay rates: $\lambda = 0.8$,
- Noise scale: $\zeta = 1.0$.

Interventions correspond to gene knockouts, where expression is fixed at 0. Technical noise is not simulated.

G.3 SIMULATION OF NEURAL NETWORK-BASED DATA

To simulate data for causal discovery, random fully connected neural networks (MLPs) are used to define conditional distributions.

G.3.1 NEURAL NETWORK SPECIFICATION

Each MLP has a single hidden layer of 10 neurons and uses ReLU activation. The MLP maps inputs $x_{\text{Pa}_j^{\mathcal{G}}}$ to a scalar output representing the mean μ of a conditional Gaussian:

$$p_j(x_j|x_{\text{Pa}_j^{\mathcal{G}}}) \sim \mathcal{N}(\mu = \text{MLP}(x_{\text{Pa}_j^{\mathcal{G}}}), \sigma = 1.0).$$

G.3.2 INTERVENTIONAL DATA GENERATION

Interventions alter the distribution of affected nodes. For an intervened node, the distribution is set to:

$$p_j(x_j|\text{do}(x_j)) \sim \mathcal{N}(2, 1.0),$$

independent of the MLP, to simulate intervention effects.

H HYPERPARAMETERS

Table 1: Hyperparameters for the DiffIntersort Causal Discovery Algorithm

Parameter	**Value**	**Description**
λ_1	0.01	L1 regularization coefficient for the weight matrix $\mathbf{W}$.
λ_2	100.0	Regularization parameter for the DiffIntersort regularization
`scaling` c	Dimension dependent	Scaling factor for the distance matrix (see Table 2).
`n_iter`	2000	Maximum total number of iterations for the optimization process.
`lr_int`	Dimension dependent	Learning rate for the permutation optimizer parameters (see Table 2).
`n_iter_sinkhorn`	500	Number of iterations for the Sinkhorn normalization process.
`t_sinkhorn`	0.05	Temperature parameter for the Sinkhorn normalization.
`eps`	0.3 or 0.5 for GRN	Epsilon value for the distance matrix.
`p_scale`	0.001	Initial scaling factor for the initialization of the permutation vector $\mathbf{p}$.
Number batches	3	Number of mini-batches per iterations.
γ	0.5	Parameter controlling the emphasis on invariance across environments.
`betas`	(0.9, 0.99)	Beta parameters for the Adam optimizer of the potential.
`lr_weights`	1e-3	Learning rate for the data fitting parameters.

Table 2: Configuration Parameters for Different Dimensions

Dimension	**Learning Rate (lr)**	**Scaling** c
3	0.05	0.1
10	0.05	0.5
30	0.01	1.0
100	0.001	1.0
1000	0.0005	1.0
2000	0.0001	1.0

I ADDITIONAL EXPERIMENTS

I.1 TRAIN TIME SCALING

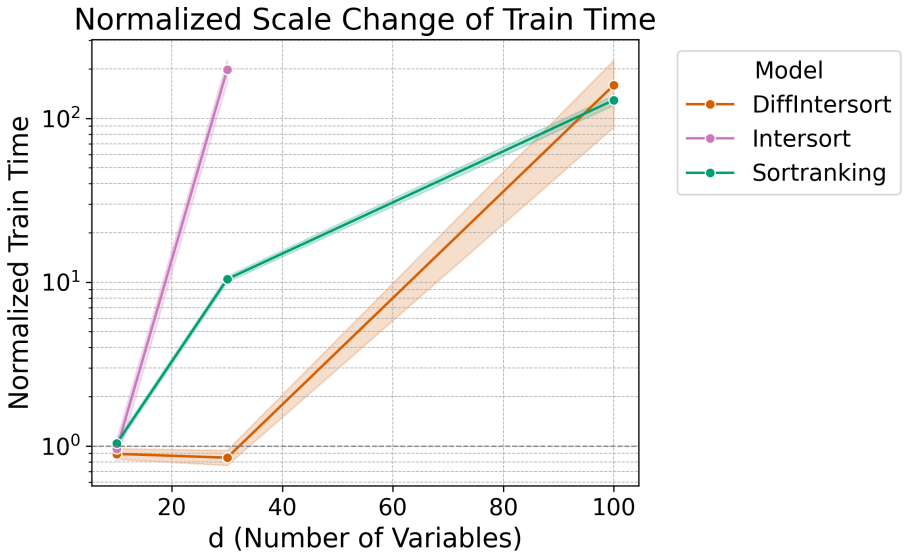


Figure 3: Normalized scale change in training time (y-axis) for Intersort, DiffIntersort and SORTRANKING across different values of d (x-axis) at 50% intervention fraction on the Neural Network data. The training time for each model is normalized to start at 1 for the smallest d, illustrating the relative growth in computational cost as the number of variables (d) increases. The plot uses a logarithmic y-scale to highlight differences in scaling behavior between models. A reference line at 1 indicates the baseline training time for $d = 10$.

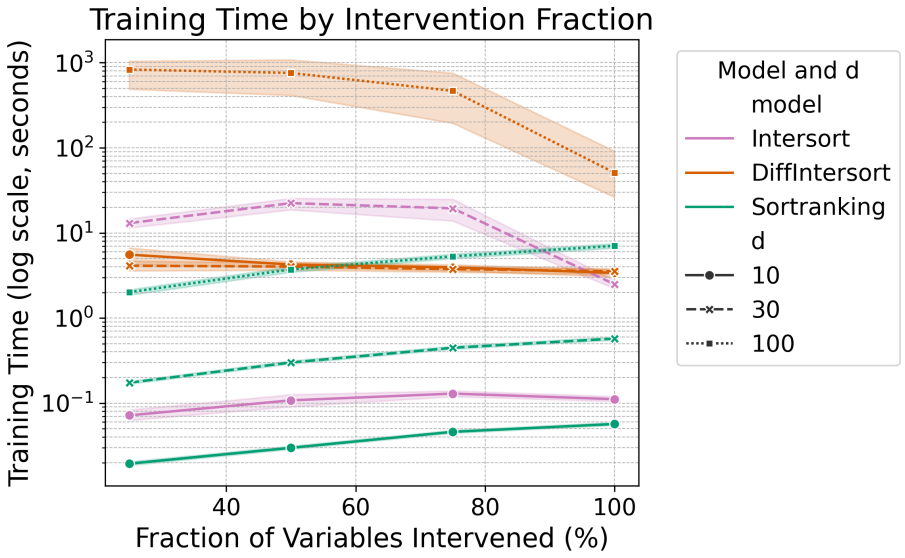


Figure 4: Training time (y-axis, log scale) for Intersort, DiffIntersort and SORTRANKING across different values of d at various intervention fraction (x-axis) on the Neural Network data.

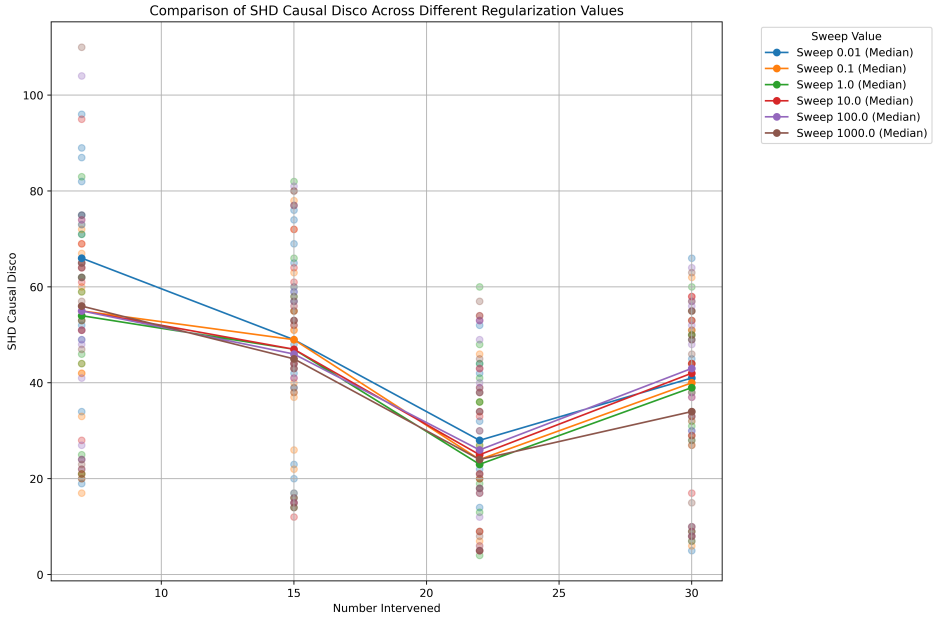
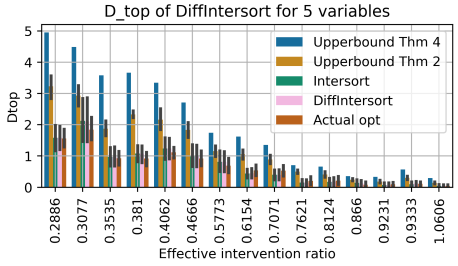


Figure 5: Comparison of SHD Causal Disco across different regularization values (Sweep) on Linear data with 30 variables. The x-axis represents the number of interventions, while the y-axis shows the Structural Hamming Distance (SHD) for causal discovery. Transparent scatter points indicate individual data samples, while solid lines connect the median SHD values at each intervention level for each sweep value. Lower SHD values indicate better causal structure recovery. The plot highlights how different regularization strengths impact performance across varying intervention numbers.

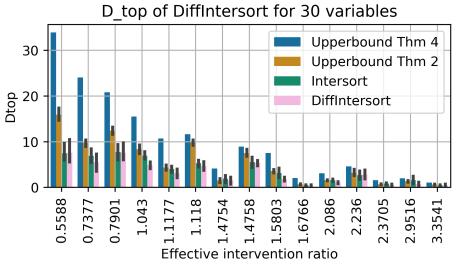
I.2 Regularization Sweep

We test different values for the regularization strength of DiffIntersort in causal discovery (see Figure 5). We observed that there does not seem to be major differences and that there are no risks of over-regularization. We thus use a value of $\lambda_2 = 100.0$ for all experiments.

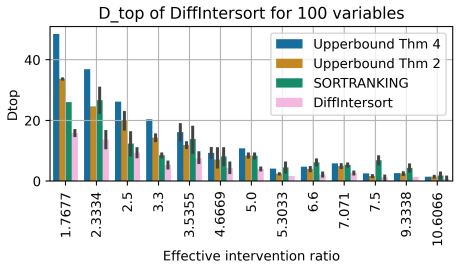
I.3 SIMULATED DISTANCE MATRICES



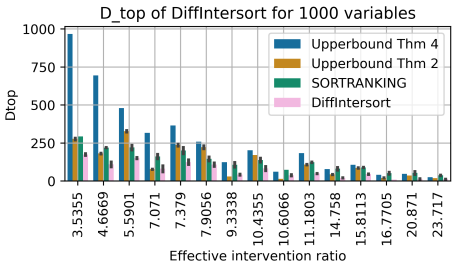
(a) Simulation ER with 5 variables



(b) Simulation ER with 30 variables

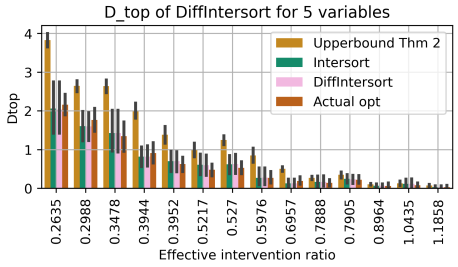


(c) Simulation ER with 100 variables

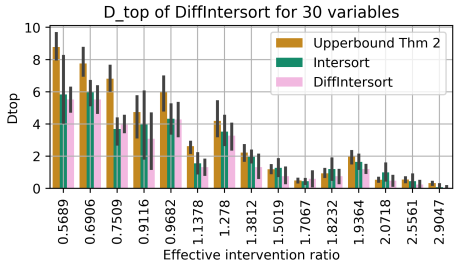


(d) Simulation ER with 1000 variables

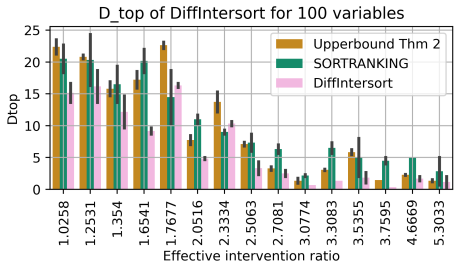
Figure 6: Comparison of performance on simulated ER graphs in terms of D_{top} divergence between the two bounds of (Chevalley et al., 2025), DiffIntersort, Intersort, and SORTRANKING. For each setting, we draw multiple graphs, where a setting is the tuple (p_{int}, p_e). Then, for each graph, we run the algorithm on multiple configurations, where a configuration corresponds to a set of intervened variables following p_{int}. We have $p_{int} \in \{0.25, 0.33, 0.5, 0.66, 0.75\}$ for all scales. For 5 variables, $p_e \in \{0.5, 0.66, 0.75\}$. For 30, $p_e \in \{0.05, 0.1, 0.2\}$. For 1000 variables, $p_e \in \{0.005, 0.002, 0.001\}$. For 20000 variables settings, $p_e \in \{0.0001, 0.00005, 0.00002\}$. These probabilities approximately correspond to an average of 1, 2, or 3 edges per variable.



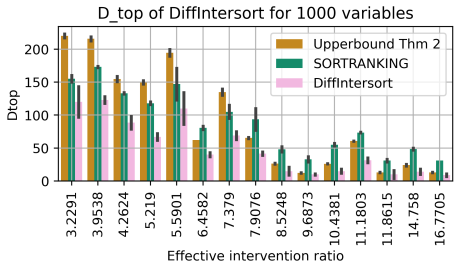
(a) Simulation SF with 5 variables



(b) Simulation SF with 30 variables



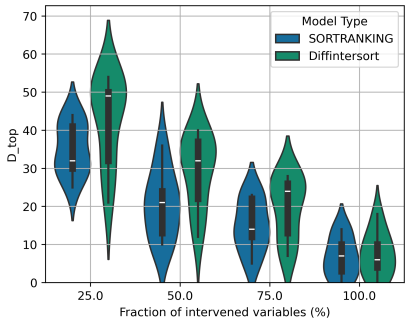
(c) Simulation SF with 100 variables



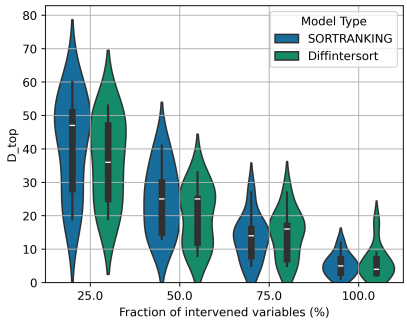
(d) Simulation SF with 1000 variables

Figure 7: Comparison of performance on simulated SF graphs in terms of D_{top} divergence between the two bounds of (Chevalley et al., 2025), DiffIntersort, Intersort and SORTRANKING. For each setting, we draw multiple graphs, where a setting is the tuple (p_{int}, p_e). The networks follow a Barabasi-Albert SF distribution, with average edge per variable in $\{1, 2, 3\}$. A setting is the tuple (p_{int}, p_e), where $p_e = \frac{2\mathrm{E}(\#edges)}{d(d-1)}$. Then, for each graph, we run the algorithm on multiple configurations, where a configuration corresponds to a set of intervened variables following p_{int}. We have $p_{int} \in \{0.25, 0.33, 0.5, 0.66, 0.75\}$ for all scales.

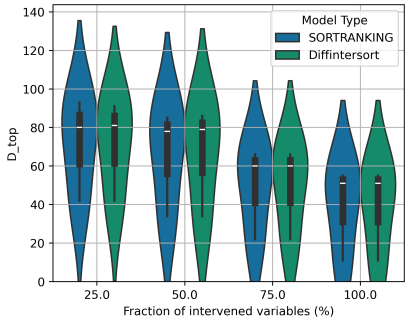
I.4 SIMULATED DATA



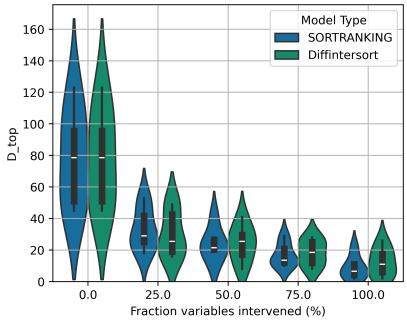
(a) Linear 100 variables



(b) RFF 100 variables

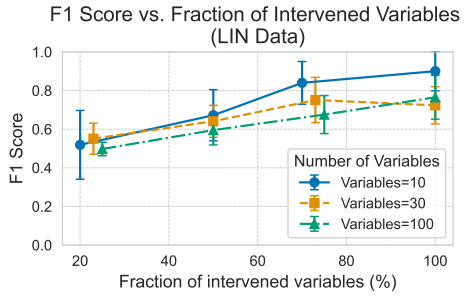


(c) GRN 100 variables

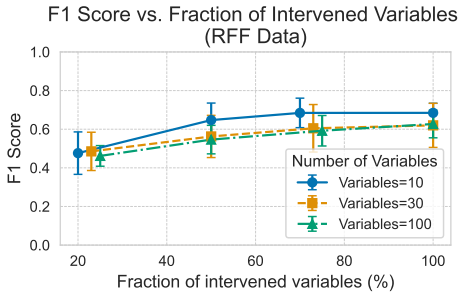


(d) NN 100 variables

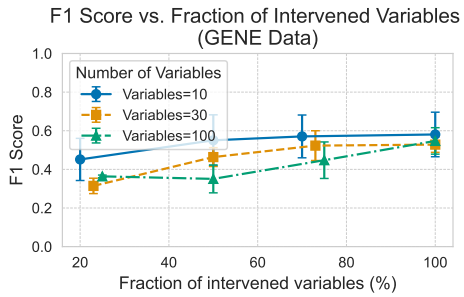
Figure 8: Top order diverge scores (lower is better) assessing the quality of the derived causal order, comparing our method based on the DiffIntersort score to SORTRANKING on 100 variables, for various types of data.



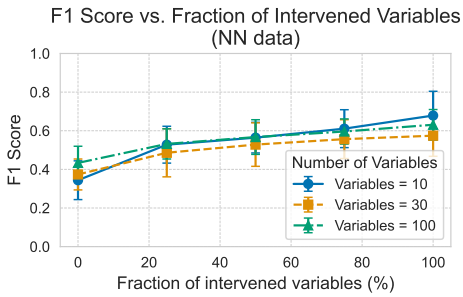
(a) Linear



(b) RFF

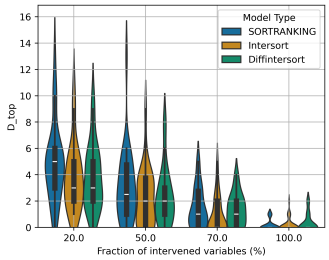


(c) GRN

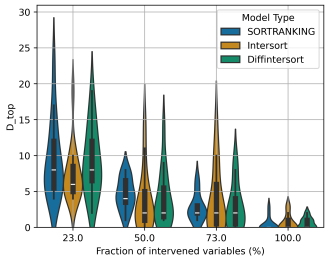


(d) NN

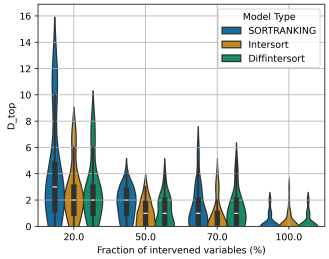
Figure 9: F1 score of our algorithm with DiffIntersort constraint for the four considered data types over the fraction of intervened variables for 10, 30, and 100 variables. As can be observed, the performance is consistent across the scale of the number of variables as there is no major drop in performance at 100 variables compared to 10 and 30 variables.



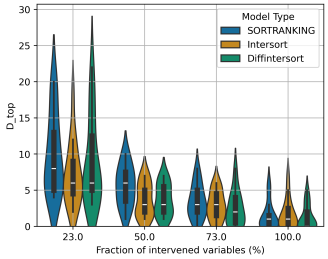
(a) Linear 10 variables



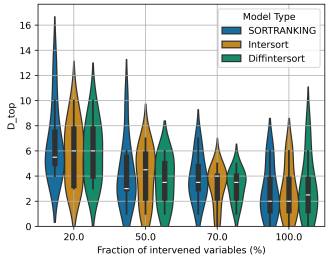
(b) Linear 30 variables



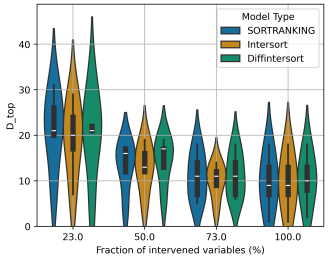
(c) RFF 10 variables



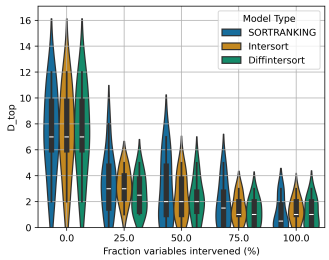
(d) RFF 30 variables



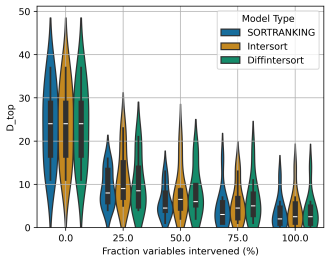
(e) GRN 10 variables



(f) GRN 30 variables

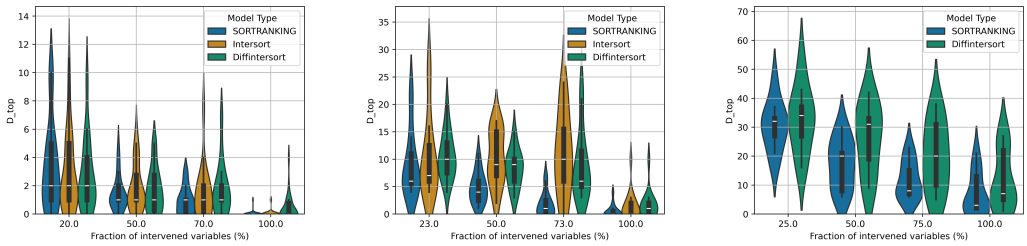


(g) NN 10 variables

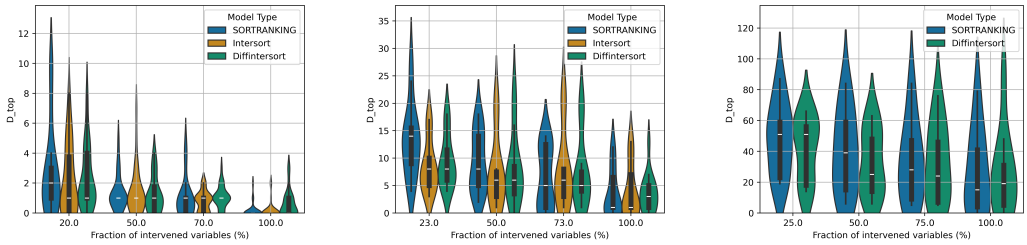


(h) NN 30 variables

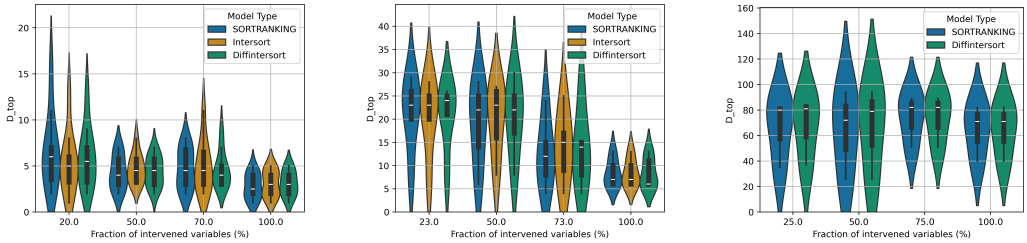
Figure 10: Top order diverge scores (lower is better) assessing the quality of the derived causal order, comparing our method based on the DiffIntersort score to SORTRANKING and Intersort on 10 and 30 variables, for various types of data.



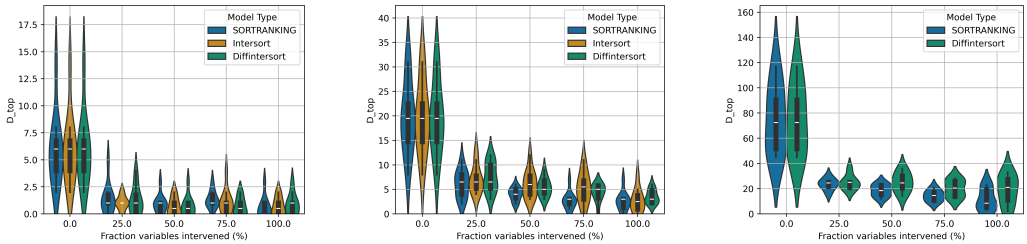
(a) Linear 10 variables (b) Linear 30 variables (c) Linear 100 variables



(d) RFF 10 variables (e) RFF 30 variables (f) RFF 100 variables

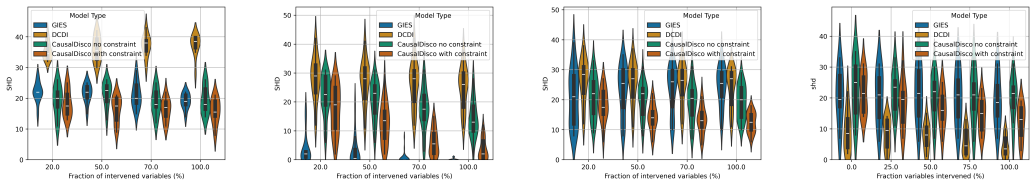


(g) GRN 10 variables (h) GRN 30 variables (i) GRN 100 variables

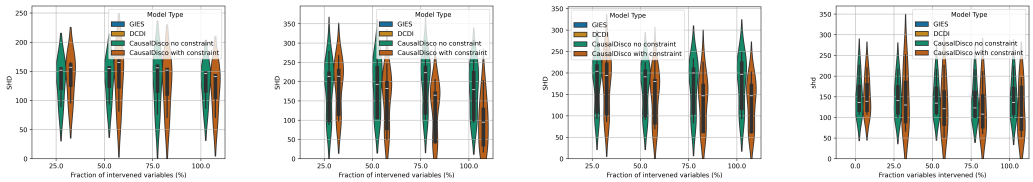


(j) NN 10 variables (k) NN 30 variables (l) NN 100 variables

Figure 11: Top order diverge scores (lower is better) assessing the quality of the derived causal order, comparing our method based on the DiffIntersort score to SORTRANKING and Intersort on 10 and 30 variables, for various types of data for a scale free network distribution.

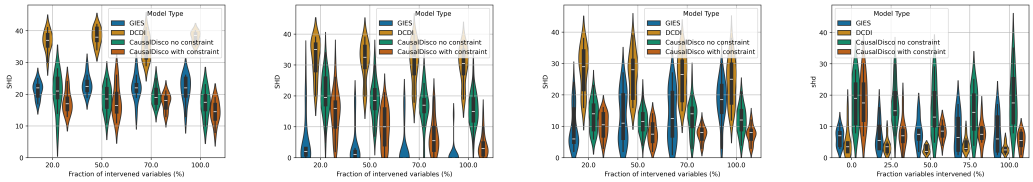


(a) GRN, 10 vars, SHD (b) Linear, 10 vars, SHD (c) RFF, 10 vars, SHD (d) NN, 10 vars, SHD

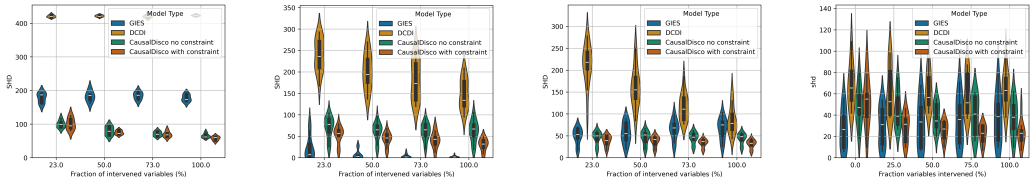


(e) GRN, 100 vars, SHD (f) Linear, 100 vars, SHD (g) RFF, 100 vars, SHD (h) NN, 100 vars, SHD

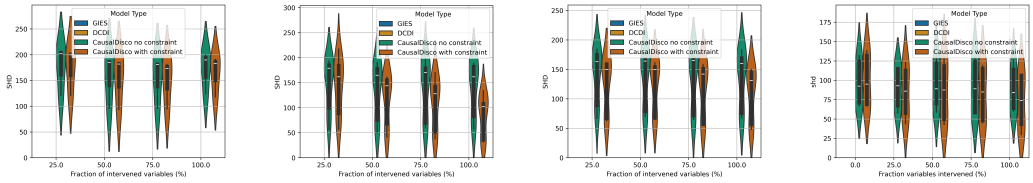
Figure 12: Comparison of Structural Hamming Distance (SHD) for Gene, Linear, RFF, and Neural Network models with varying numbers of variables.



(a) GRN, 10 vars, SHD (b) Linear, 10 vars, SHD (c) RFF, 10 vars, SHD (d) NN, 10 vars, SHD

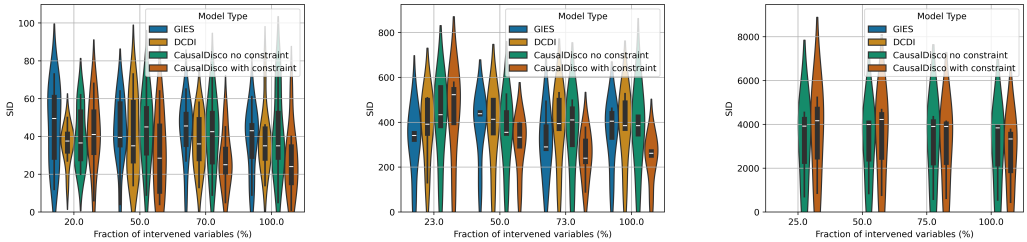


(e) GRN, 30 vars, SHD (f) Linear, 30 vars, SHD (g) RFF, 30 vars, SHD (h) NN, 30 vars, SHD

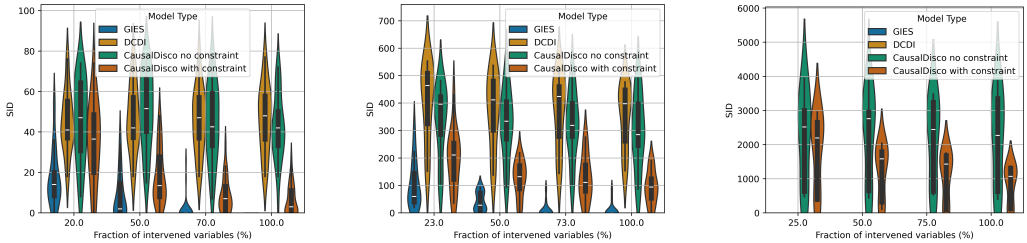


(i) GRN, 100 vars, SHD (j) Linear, 100 vars, SHD (k) RFF, 100 vars, SHD (l) NN, 100 vars, SHD

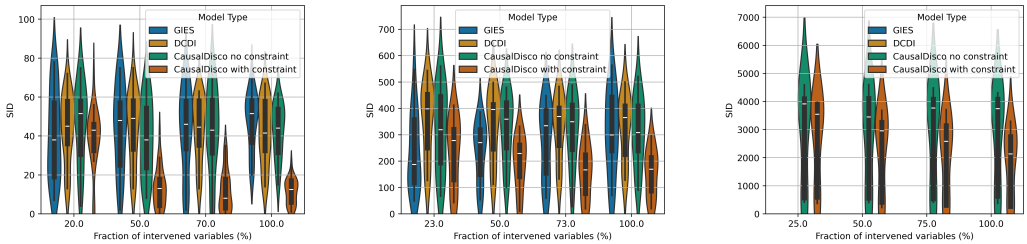
Figure 13: Comparison of Structural Hamming Distance (SHD) for Gene, Linear, RFF, and Neural Network models with varying numbers of variables for a scale-free network distribution.



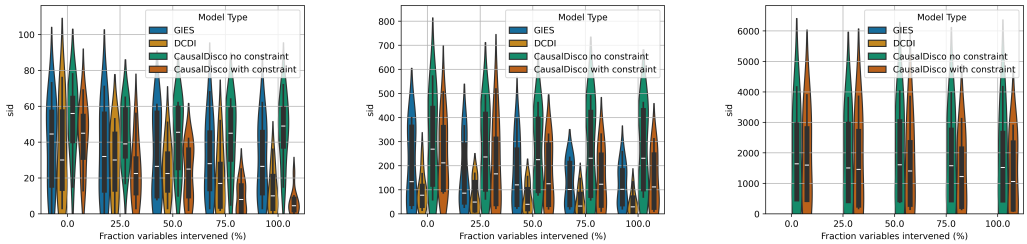
(a) GRN, 10 vars, SID (b) GRN, 30 vars, SID (c) GRN, 100 vars, SID



(d) Linear, 10 vars, SID (e) Linear, 30 vars, SID (f) Linear, 100 vars, SID

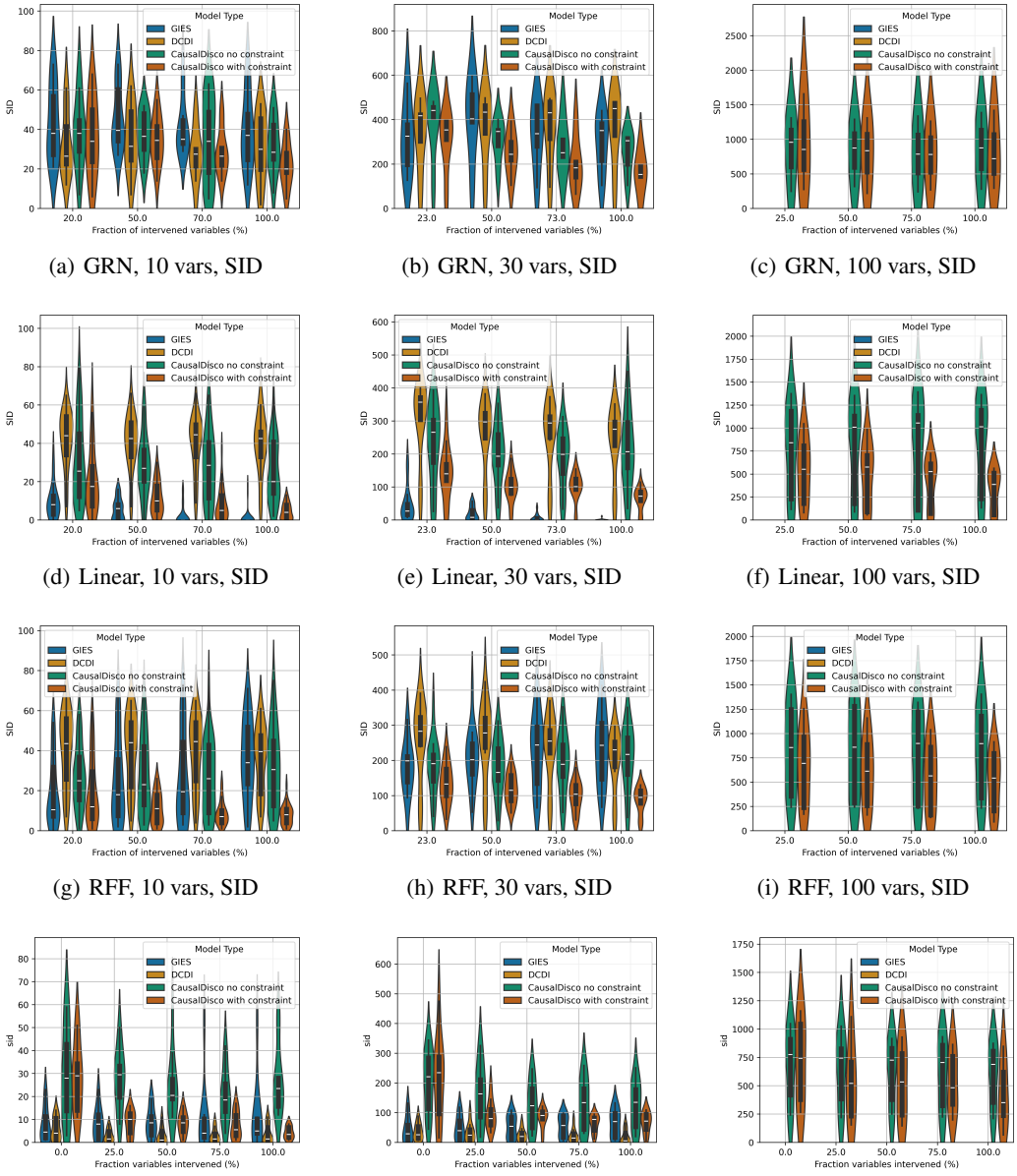


(g) RFF, 10 vars, SID (h) RFF, 30 vars, SID (i) RFF, 100 vars, SID



(j) NN, 10 vars, SID (k) NN, 30 vars, SID (l) NN, 100 vars, SID

Figure 14: Comparison SID (lower is better) for GRN, Linear, RFF, and Neural Network models with varying numbers of variables.



(a) GRN, 10 vars, SID (b) GRN, 30 vars, SID (c) GRN, 100 vars, SID

(d) Linear, 10 vars, SID (e) Linear, 30 vars, SID (f) Linear, 100 vars, SID

(g) RFF, 10 vars, SID (h) RFF, 30 vars, SID (i) RFF, 100 vars, SID

(j) NN, 10 vars, SID (k) NN, 30 vars, SID (l) NN, 100 vars, SID

Figure 15: Comparison SID (lower is better) for GRN, Linear, RFF, and Neural Network models with varying numbers of variables for a scale-free network distribution.