SAMPLE, ESTIMATE, AGGREGATE: A RECIPE FOR CAUSAL DISCOVERY FOUNDATION MODELS

Menghua Wu
Department of Computer Science
Massachusetts Institute of Technology
Cambridge, MA, USA
rmwu@mit.edu

Yujia Bao
Accenture
Mountain View, CA, USA
yujia.bao@accenture.com

Regina Barzilay & Tommi Jaakkola
Department of Computer Science
Massachusetts Institute of Technology
Cambridge, MA, USA
{regina,tommi}@csail.mit.edu

ABSTRACT

Causal discovery, the task of inferring causal structure from data, promises to accelerate scientific research, inform policy making, and more. However, the per-dataset nature of existing causal discovery algorithms renders them slow, data hungry, and brittle. Inspired by foundation models, we propose a causal discovery framework where a deep learning model is pretrained to resolve predictions from classical discovery algorithms run over smaller subsets of variables. This method is enabled by the observations that the outputs from classical algorithms are fast to compute for small problems, informative of (marginal) data structure, and their structure outputs as objects remain comparable across datasets. Our method achieves state-of-the-art performance on synthetic and realistic datasets, generalizes to data generating mechanisms not seen during training, and offers inference speeds that are orders of magnitude faster than existing models.

1 INTRODUCTION

A fundamental aspect of scientific research is to discover and validate causal hypotheses involving variables of interest. Given observations of these variables, the goal of causal discovery algorithms is to extract such hypotheses in the form of directed graphs, in which edges denote causal relationships Spirtes et al. (2001). In much of basic science, however, classical statistics remain the de facto basis of data analysis Replogle et al. (2022). Key barriers to widespread adoption of causal discovery algorithms include their high data requirements and their computational intractability on larger problems.

Current causal discovery algorithms follow two primary approaches that differ in their treatment of the underlying causal graph. Discrete optimization algorithms explore the super-exponential space of graphs by proposing and evaluating changes to a working graph Glymour et al. (2019). While these methods are quite fast on small graphs, the combinatorial space renders them intractable for exploring larger structures, with hundreds or thousands of nodes. Furthermore, their correctness is tied to hypothesis tests, whose results can be erroneous for noisy datasets, especially as graph sizes increase. More recently, a number of works have reframed the discrete graph search as a continuous optimization over weighted adjacency matrices Zheng et al. (2018); Brouillard et al. (2020). Continuous optimization algorithms often require that a generative model be fit to the full data distribution, a difficult task when the number of variables increases but the data remain sparse.

In this work, we present SEA: Sample, Estimate, Aggregate, a blueprint for developing causal discovery foundation models that enable fast inference on new datasets, perform well in low data regimes, and generalize to causal mechanisms beyond those seen during training. Our approach is motivated by two observations. First, while classical causal discovery algorithms scale poorly, their

bottleneck lies in the exponential search space, rather than individual independence tests. Second, in many cases, statistics like global correlation or inverse covariance are indeed strong indicators for a graph's overall connectivity. Therefore, we propose to leverage 1) the estimates of classical algorithms over small subgraphs, and 2) global graph-level statistics, as inputs to a deep learning model, pretrained to resolve these statistical descriptors into causal graphs.

Theoretically, we prove that given only marginal estimates over subgraphs, it is possible to recover causally sound global graphs; and that our proposed model has the capacity to recapitulate such reasoning. Empirically, we implement instantiations of SEA using an axial-attention based model Ho et al. (2020) which takes as input, inverse covariance and estimates from the classical FCI or GIES algorithms Spirtes et al. (1995); Hauser & Bühlmann (2012). We conduct thorough comparison to three classical baselines and five deep learning approaches. SEA attains the state-of-the-art results on synthetic and real-world causal discovery tasks, while providing 10-1000x faster inference. While these experimental results reflect specific algorithms and architectures, the overall framework accepts any combination of sampling heuristics, classical causal discovery algorithms, and statistical features. To summarize, our contributions are as follows.

1. To the best of our knowledge, we are the first to propose a method for building fast, robust, and generalizable foundation models for causal discovery.
2. We show that both our overall framework and specific architecture have the capacity to reproduce causally sound graphs from their inputs.
3. We attain state-of-the-art results on synthetic and realistic settings, and we provide extensive experimental analysis of our model.

2 BACKGROUND AND RELATED WORK

2.1 CAUSAL GRAPHICAL MODELS

A causal graphical model is a directed, acyclic graph $G = (V, E)$, where each node $i \in V$ corresponds to a random variable $X_i \in X$ and each edge $(i, j) \in E$ represents a causal relationship from $X_i \to X_j$. We assume that the data distribution P_X is Markov to G,

$$\forall i \in V, X_i \perp\!\!\!\perp V \setminus (X_{\delta_i} \cup X_{\pi_i}) \mid X_{\pi_j} \tag{1}$$

where δ_i is the set of descendants of node i in G, and π_i is the set of parents of i. In addition, we assume that P_X is minimal and faithful – that is, P_X does not include any independence relationships beyond those implied by applying the Markov condition to G Spirtes et al. (2001). Causal graphical models allow us to perform *interventions* on nodes i by setting conditional $P(X_i \mid X_{\pi_i})$ to a different distribution $\tilde{P}(X_i \mid X_{\pi_i})$.

2.2 CAUSAL DISCOVERY

Given a dataset $D \sim P_X$, the goal of causal discovery is to recover G. There are two main challenges. First, the number of possible graphs is super-exponential in the number of nodes N, so causal discovery algorithms must navigate this combinatorial search space efficiently. In terms of graph size, the limits of current algorithms range from tens of nodes Hägele et al. (2023); Lorch et al. (2022) to hundreds of nodes Lopez et al. (2022), where simplifying assumptions regarding the graph structure are often made in the latter case. Second, depending on data availability and the underlying data generation process, causal discovery algorithms may or may not be able to recover G in practice. In fact, many algorithms are only analyzed in the infinite-data regime and require at least thousands of data samples for reasonable empirical performance Spirtes et al. (2001); Brouillard et al. (2020). For detailed discussion regarding the two main approaches to graphical causal discovery, see Appendix A.

Causal discovery is also used in the context of pairwise relationships Zhang & Hyvarinen (2012); Monti et al. (2019), as well as for non-stationary Huang et al. (2020) and time-series Löwe et al. (2022) data. This work focuses on causal discovery for stationary graphs.

2.3 FOUNDATION MODELS

The concept of foundation models has revolutionized the machine learning workflow in a variety of disciplines: instead of training domain-specific models from scratch, we can query a pretrained,

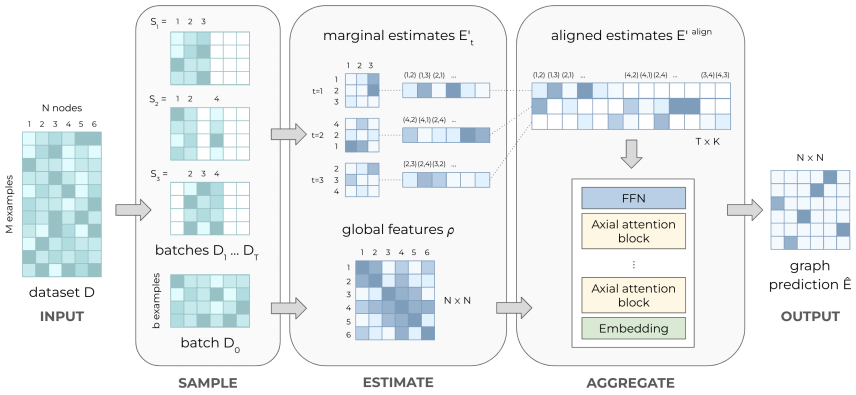

Figure 1: An overview of the SEA inference procedure. Given a new dataset, we 1) sample batches and subsets of nodes, 2) estimate marginal graphs and global statistics over these batches, and 3) aggregate these features using a pretrained model to obtain the underlying causal graph. Raw data are depicted in green, and graph-related features are depicted in blue.

general-purpose "foundation" model Radford et al. (2021); Brown et al. (2020); Bommasani et al. (2022). Recent work has explored foundation models for causal inference Zhang et al. (2023), but this method addresses inference rather than discovery, so it assumes knowledge of the causal graphs.

3 METHODS

We present SEA, a framework for developing fast and scalable causal discovery foundation models. Our framework combines global statistics and marginal estimates (the outputs of classical causal discovery algorithms on subsets of nodes) as inputs to a deep learning model, pretrained to aggregate these features into causal graphs (Section 3.1). As proofs of concept, we describe instantiations of SEA using specific algorithms and architectures (Sections 3.2, 3.3). Finally, we prove that our algorithm has the capacity to produce sound causal graphs, in theory and in our model (Section 3.4).

3.1 CAUSAL DISCOVERY FRAMEWORK

SEA is a causal discovery framework that learns to resolve statistical features and estimates of marginal graphs into a global causal graph. The inference procedure is depicted in Figure 1. Specifically, given a new dataset $D \in \mathbb{R}^{M \times N}$ faithful to graph $G = (V, E)$, we apply the following stages.

Sample: takes as input dataset D; and outputs data batches $\{D_0, D_1, \ldots, D_T\}$ and node subsets $\{S_1, \ldots, S_T\}$.

1. Sample $T + 1$ batches of $b \ll M$ observations uniformly at random from D.
2. Compute selection scores $\alpha \in (0, 1)^{N \times N}$ over D_0 (e.g. inverse covariance).
3. Sample T node subsets of size k. Each subset $S_t \subseteq V$ is constructed iteratively, with additional nodes sampled one at a time with probability proportional to $\sum_{j \in S_t} \alpha_{i,j}$ (C.6).

Estimate: takes as inputs data batches and node subsets; and outputs global statistics ρ and marginal estimates $\{E'_1, \ldots, E'_T\}$.

1. Compute global statistics $\rho \in \mathbb{R}^{N \times N}$ over D_0.
2. Run causal discovery algorithm f to obtain marginal estimates $f(D_t[S_t]) = E'_t$ for $t = 1 \ldots T$.

We use $D_t[S_t]$ to denote the observations in D_t that correspond only to the variables in S_t. Each estimate E'_t is a $k \times k$ adjacency matrix, corresponding to the k nodes in S_t.

Aggregate: takes as inputs global statistics, marginal estimates, and node subsets. A pretrained aggregator model outputs the predicted global causal graph $\hat{E} \in (0, 1)^{N \times N}$.

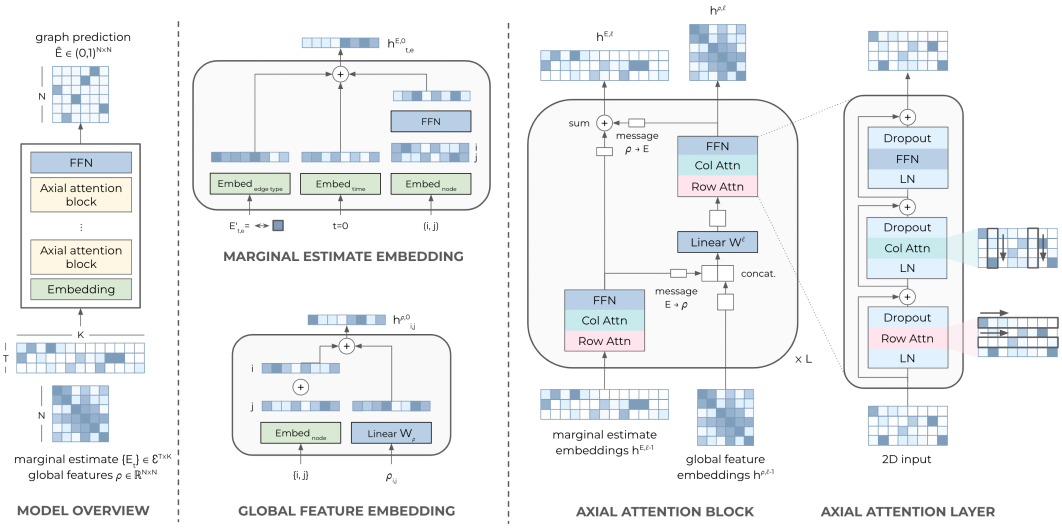

Figure 2: Aggregator architecture. Marginal graph estimates and global statistics are first embedded into the model dimension, and 1D positional embeddings are added along both rows and columns. These combined embeddings pass through a series of axial attention blocks, which attend to the graph estimates and the global features. The final layer global features pass through a feedforward network to predict the causal graph.

The aggregator's architecture is agnostic to the assumptions of the underlying causal discovery algorithm f (e.g. (non)linearity, observational vs. interventional). Therefore, we can freely swap out f and the training datasets to train aggregators with different implicit assumptions, without modifying the architecture itself. Here, our aggregator is pretrained over a diverse set of graph sizes, structures, and data generating mechanisms (details in Appendix C.1).

3.2 Model Architecture

The core module of SEA is the aggregator (Figure 2), implemented as a sequence of axial attention blocks. The aggregator takes as input global statistics $\rho \in \mathbb{R}^{N \times N}$, marginal estimates $E'_{1...T} \in \mathcal{E}^{T \times k \times k}$, and node subsets $S_{1...T} \in [N]^{T \times k}$, where $\mathcal{E}$ is the set of output edge types for the causal discovery algorithm f. The output is a graph prediction $\hat{E} \in \{0,1\}^{N \times N}$, supervised by the ground truth E. Our model is trained with cross entropy loss and L2 weight regularization. Please see Appendix C.3 for details.

Our model was implemented with 4 layers with 8 attention heads and hidden dimension 512. Our model was trained using the AdamW optimizer with a learning rate of 1e-4 Loshchilov & Hutter (2019). Please see C.3 for additional details regarding hyperparameters.

3.3 Implementation Details

We computed inverse covariance as the global statistic and selection score, due to its relationship to partial correlation and ease of computation. For comparison to additional statistics, see D.4. We chose the constraint-based FCI algorithm in the observational setting Spirtes et al. (2001), and the score-based GIES algorithm in the interventional setting Hauser & Bühlmann (2012). For discussion regarding alternative algorithms, see C.5. We sample batches of size $b = 500$ over $k = 5$ nodes each (analysis in D.3).

3.4 Theoretical Analyses

Our theoretical contributions span two aspects.

1. We formalize the notion of marginal estimates and prove that given sufficient marginal estimates, it is possible to recover a pattern faithful to the global causal graph (Theorem

B.5). We provide bounds on the number of marginal estimates required and motivate global statistics as an efficient means to reduce this bound (Propositions B.9, B.10).
2. We show that our proposed axial attention has the capacity to recapitulate the reasoning required for this task. In particular, we show that a stack of 3 axial attention blocks can recover the skeleton and v-structures in $O(N)$ width (Theorem B.13).

Please refer to Appendix B for the formal theorems and proofs. Our proofs assume only that the edge estimates are correct. We do not impose any functional requirements, and any appropriate independence test may be used. We discuss robustness and stability in B.4.

4 EXPERIMENTAL SETUP

We pretrained SEA models on synthetic training datasets only and ran inference on held-out testing datasets, which include both seen and unseen causal mechanisms. All baselines were trained and/or run from scratch on each testing dataset using their published code and hyperparameters. We evaluate our model across diverse synthetic datasets, simulated mRNA datasets Dibaeinia & Sinha (2020), and a real protein expression dataset Sachs et al. (2005). We include details on synthetic data generation in Appendix C.1 and Appendix D.2.1 (simulated mRNA).

4.1 CAUSAL DISCOVERY METRICS

Our causal discovery experiments consider both discrete and continuous metrics. In addition to standard metrics like SHD Tsamardinos et al. (2006), we include continuous metrics, as neural networks can be notoriously uncalibrated Guo et al. (2017), and arbitrary discretization thresholds reflect an incomplete picture of model performance Schaeffer et al. (2023). For all continuous metrics, we exclude the diagonal from evaluation, since several baselines manually set it to zero Brouillard et al. (2020); Lopez et al. (2022).

SHD: Structural Hamming distance is the minimum number of edge edits required to match two graphs Tsamardinos et al. (2006). Discretization thresholds are as published.

mAP, AUC: We compute the area under the precision-recall / ROC curves per edge and average over the graph. The mean random guessing baseline depends on the positive rate.

Edge orientation accuracy: We compute the accuracy of edge orientations as

$$\text{EdgeAcc} = \frac{\sum_{(i,j)\in E} \mathbb{1}\{P(i,j) > P(j,i)\}}{\|E\|}. \tag{2}$$

Since this quantity is normalized by the size of E, it is invariant to the positive rate. In contrast to edge orientation F1 Geffner et al. (2022), this quantity is also invariant to the assignment of forward/reverse edges as positive/negative.

4.2 BASELINES

We consider several deep learning and classical baselines. The following deep learning baselines all start by fitting a generative model to the data.

DCDI Brouillard et al. (2020) extracts the underlying graph as a model parameter. The G and DSF variants use Gaussian or deep sigmoidal flow likelihoods, respectively. **DCD-FG** Lopez et al. (2022) follows DCDI-G, but factorizes the graph into a product of two low-rank matrices for scalability.

DECI Geffner et al. (2022) takes a Bayesian approach and extracts the graph as a parameter.

DIFFAN Sanchez et al. (2023) uses the trained model's Hessian to obtain a topological ordering, followed by a classical pruning algorithm.

AVICI Lorch et al. (2022) uses an amortized inference approach to estimate $P(G \mid D)$ over a class of data-generating mechanisms via variational inference.

The following classical baselines (ablations) quantify the causal discovery utility of our inputs.

INVCOV computes inverse covariance over 2000 examples. This does *not* orient edges, but it is a strong connectivity baseline. We discretize based on ground truth (oracle) E.

Table 1: Causal discovery results on simulated mRNA data. Each setting encompasses 5 distinct scale-free graphs. Data were generated via SERGIO Dibaeinia & Sinha (2020).

N	Model	Hill coef. $= 2.0$, $E = N$				Hill coef. $= 2.0$, $E = 2N$			
		mAP ↑	AUC ↑	EA ↑	SHD ↓	mAP ↑	AUC ↑	EA ↑	SHD ↓
10	DCDI-G	0.48	0.73	0.70	16	0.32	0.57	0.59	26
	DCDI-DSF	0.63	0.84	0.81	18	0.44	0.64	0.64	26
	DCD-FG	0.59	0.82	0.79	81	0.43	0.69	0.67	73
	AVICI	0.58	0.85	0.81	6	0.22	0.44	0.36	17
	INVCOV	0.25	0.44	—	13	0.33	0.54	—	22
	CORR	0.44	0.89	—	9	0.35	0.67	—	20
	SEA (FCI)	**0.92**	**0.98**	**0.92**	**2**	**0.76**	**0.90**	**0.85**	**9**
20	DCDI-G	0.48	0.86	0.90	37	0.31	0.65	0.72	55
	DCDI-DSF	0.45	0.92	**0.94**	52	0.40	0.71	0.74	55
	DCD-FG	0.34	0.87	0.66	361	0.36	0.77	0.67	343
	AVICI	0.32	0.78	0.76	19	0.17	0.54	0.55	37
	INVCOV	0.27	0.56	—	24	0.30	0.62	—	47
	CORR	0.35	0.93	—	24	0.29	0.76	—	49
	SEA (FCI)	**0.54**	**0.94**	0.83	**17**	**0.50**	**0.85**	**0.78**	**31**

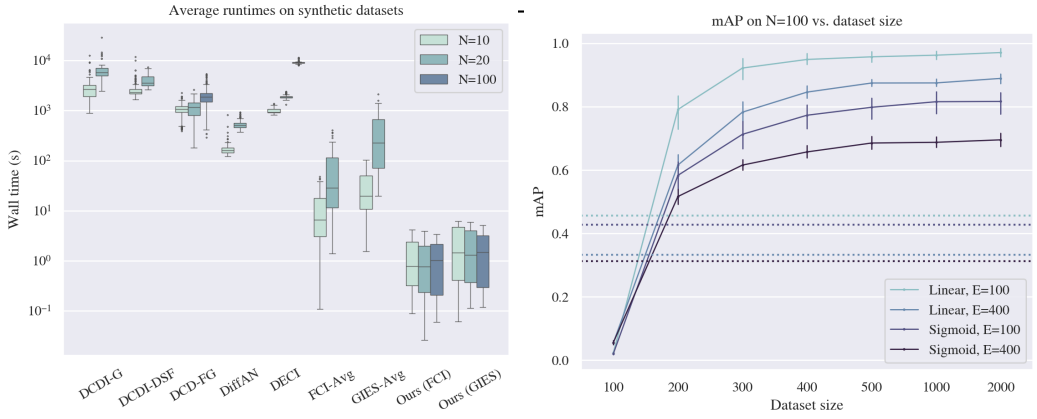

Figure 3: Analysis. Left: Average wall time required to run each model on a single dataset. The y-axis is plotted in log scale. Right: Performance of our model (FCI) vs. total dataset size. Error bars indicate 95% confidence interval across 5 datasets of each setting. Dashed lines indicate the INVCOV estimate on 500 points. We set batch size $b = \min(500, M)$ and node subset size $k = 5$.

FCI-AVG, GIES-AVG run the FCI and GIES algorithms, respectively, over *all* nodes, on 100 batches with 500 examples each. We take the mean $P((i, j))$ over all batches. This procedure yielded higher performance compared to running the algorithm only once, over a larger batch.

5 RESULTS

We highlight our simulated mRNA experiments and analysis on synthetic settings. Full synthetic and realistic experiments, as well as ablation studies, may be found in Appendix D.

We consistently outperform baselines on simulated mRNA data (Table 1). Furthermore, on a wide range of synthetic datasets with $N = 10, 20, 100$, our model is orders of magnitude faster than other continuous optimization methods (Figure 3). Finally, one of the main advantages of foundation models is that they enable high levels of performance in low resource scenarios. Figure 3 shows that SEA (FCI) only requires around $M = 500$ data samples for decent performance on graphs with $N = 100$ nodes. This is in contrast to existing continuous optimization methods, which require thousands to tens of thousands of samples to fit completely.

6 CONCLUSION

In this work, we introduced SEA, a framework for designing causal discovery foundation models. SEA is motivated by the idea that classical discovery algorithms provide powerful descriptors of data that are fast to compute and robust across datasets. Given these statistics, we train a deep learning model to reproduce faithful causal graphs. Theoretically, we demonstrated that it is possible to produce sound causal graphs from marginal estimates, and that our model has the capacity to do so. Empirically, we implemented two proofs of concept of SEA that perform well across a variety of causal discovery tasks. We hope that this work will inspire a new avenue of research in generalizable and scalable causal discovery algorithms.

ACKNOWLEDGEMENTS

We thank Bowen Jing, Felix Faltings, Sean Murphy, Jiaqi Zhang, and Romain Lopez for helpful discussions.

This material is based upon work supported by the National Science Foundation Graduate Research Fellowship under Grant No. 1745302. We would like to acknowledge support from the NSF Expeditions grant (award 1918839: Collaborative Research: Understanding the World Through Code), Machine Learning for Pharmaceutical Discovery and Synthesis (MLPDS) consortium, and the Abdul Latif Jameel Clinic for Machine Learning in Health.

REFERENCES

Rishi Bommasani, Drew A. Hudson, Ehsan Adeli, Russ Altman, Simran Arora, Sydney von Arx, Michael S. Bernstein, Jeannette Bohg, Antoine Bosselut, Emma Brunskill, Erik Brynjolfsson, Shyamal Buch, Dallas Card, Rodrigo Castellon, Niladri Chatterji, Annie Chen, Kathleen Creel, Jared Quincy Davis, Dora Demszky, Chris Donahue, Moussa Doumbouya, Esin Durmus, Stefano Ermon, John Etchemendy, Kawin Ethayarajh, Li Fei-Fei, Chelsea Finn, Trevor Gale, Lauren Gillespie, Karan Goel, Noah Goodman, Shelby Grossman, Neel Guha, Tatsunori Hashimoto, Peter Henderson, John Hewitt, Daniel E. Ho, Jenny Hong, Kyle Hsu, Jing Huang, Thomas Icard, Saahil Jain, Dan Jurafsky, Pratyusha Kalluri, Siddharth Karamcheti, Geoff Keeling, Fereshte Khani, Omar Khattab, Pang Wei Koh, Mark Krass, Ranjay Krishna, Rohith Kuditipudi, Ananya Kumar, Faisal Ladhak, Mina Lee, Tony Lee, Jure Leskovec, Isabelle Levent, Xiang Lisa Li, Xuechen Li, Tengyu Ma, Ali Malik, Christopher D. Manning, Suvir Mirchandani, Eric Mitchell, Zanele Munyikwa, Suraj Nair, Avanika Narayan, Deepak Narayanan, Ben Newman, Allen Nie, Juan Carlos Niebles, Hamed Nilforoshan, Julian Nyarko, Giray Ogut, Laurel Orr, Isabel Papadimitriou, Joon Sung Park, Chris Piech, Eva Portelance, Christopher Potts, Aditi Raghunathan, Rob Reich, Hongyu Ren, Frieda Rong, Yusuf Roohani, Camilo Ruiz, Jack Ryan, Christopher Ré, Dorsa Sadigh, Shiori Sagawa, Keshav Santhanam, Andy Shih, Krishnan Srinivasan, Alex Tamkin, Rohan Taori, Armin W. Thomas, Florian Tramèr, Rose E. Wang, William Wang, Bohan Wu, Jiajun Wu, Yuhuai Wu, Sang Michael Xie, Michihiro Yasunaga, Jiaxuan You, Matei Zaharia, Michael Zhang, Tianyi Zhang, Xikun Zhang, Yuhui Zhang, Lucia Zheng, Kaitlyn Zhou, and Percy Liang. On the opportunities and risks of foundation models, 2022.

Philippe Brouillard, Sébastien Lachapelle, Alexandre Lacoste, Simon Lacoste-Julien, and Alexandre Drouin. Differentiable causal discovery from interventional data, 2020.

Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. Language models are few-shot learners, 2020.

Peter Bühlmann, Jonas Peters, and Jan Ernest. CAM: Causal additive models, high-dimensional order search and penalized regression. *The Annals of Statistics*, 42(6):2526 – 2556, 2014. doi: 10.1214/14-AOS1260.

David Maxwell Chickering. Optimal structure identification with greedy search. 3:507–554, November 2002.

Payam Dibaeinia and Saurabh Sinha. Sergio: A single-cell expression simulator guided by gene regulatory networks. *Cell Systems*, 11(3):252–271.e11, 2020. ISSN 2405-4712. doi: https://doi.org/10.1016/j.cels.2020.08.003.

Tomas Geffner, Javier Antoran, Adam Foster, Wenbo Gong, Chao Ma, Emre Kiciman, Amit Sharma, Angus Lamb, Martin Kukla, Nick Pawlowski, Miltiadis Allamanis, and Cheng Zhang. Deep end-to-end causal inference, 2022.

Clark Glymour, Kun Zhang, and Peter Spirtes. Review of causal discovery methods based on graphical models. *Frontiers in Genetics*, 10, 2019. ISSN 1664-8021. doi: 10.3389/fgene.2019.00524.

Chuan Guo, Geoff Pleiss, Yu Sun, and Kilian Q. Weinberger. On calibration of modern neural networks. In *International Conference on Machine Learning*, 2017.

Alain Hauser and Peter Bühlmann. Characterization and greedy learning of interventional markov equivalence classes of directed acyclic graphs. 2012.

Jonathan Ho, Nal Kalchbrenner, Dirk Weissenborn, and Tim Salimans. Axial attention in multidimensional transformers, 2020.

Kurt Hornik, Maxwell Stinchcombe, and Halbert White. Multilayer feedforward networks are universal approximators. *Neural Networks*, 2(5):359–366, 1989. ISSN 0893-6080. doi: https://doi.org/10.1016/0893-6080(89)90020-8.

Biwei Huang, Kun Zhang, Jiji Zhang, Joseph Ramsey, Ruben Sanchez-Romero, Clark Glymour, and Bernhard Schölkopf. Causal discovery from heterogeneous/nonstationary data with independent changes, 2020.

Alexander Hägele, Jonas Rothfuss, Lars Lorch, Vignesh Ram Somnath, Bernhard Schölkopf, and Andreas Krause. Bacadi: Bayesian causal discovery with unknown interventions, 2023.

Diviyan Kalainathan, Olivier Goudet, and Ritik Dutta. Causal discovery toolbox: Uncovering causal relationships in python. *Journal of Machine Learning Research*, 21(37):1–5, 2020.

Nan Rosemary Ke, Silvia Chiappa, Jane Wang, Anirudh Goyal, Jorg Bornschein, Melanie Rey, Theophane Weber, Matthew Botvinic, Michael Mozer, and Danilo Jimenez Rezende. Learning to induce causal structure, 2022.

Sébastien Lachapelle, Philippe Brouillard, Tristan Deleu, and Simon Lacoste-Julien. Gradient-based neural dag learning, 2020.

Wai-Yin Lam, Bryan Andrews, and Joseph Ramsey. Greedy relaxations of the sparsest permutation algorithm. In James Cussens and Kun Zhang (eds.), *Proceedings of the Thirty-Eighth Conference on Uncertainty in Artificial Intelligence*, volume 180 of *Proceedings of Machine Learning Research*, pp. 1052–1062. PMLR, 01–05 Aug 2022.

Olivier Ledoit and Michael Wolf. A well-conditioned estimator for large-dimensional covariance matrices. *Journal of Multivariate Analysis*, 88(2):365–411, 2004. ISSN 0047-259X. doi: https://doi.org/10.1016/S0047-259X(03)00096-4.

Phillip Lippe, Taco Cohen, and Efstratios Gavves. Efficient neural causal discovery without acyclicity constraints. In *International Conference on Learning Representations*, 2022.

Romain Lopez, Jan-Christian Hütter, Jonathan K. Pritchard, and Aviv Regev. Large-scale differentiable causal discovery of factor graphs. In *Advances in Neural Information Processing Systems*, 2022.

Lars Lorch, Scott Sussex, Jonas Rothfuss, Andreas Krause, and Bernhard Schölkopf. Amortized inference for causal structure learning, 2022.

Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization, 2019.

Sindy Löwe, David Madras, Richard Zemel, and Max Welling. Amortized causal discovery: Learning to infer causal graphs from time-series data, 2022.

Ricardo Pio Monti, Kun Zhang, and Aapo Hyvarinen. Causal discovery with general non-linear relationships using non-linear ica, 2019.

Joris M. Mooij, Sara Magliacane, and Tom Claassen. Joint causal inference from multiple contexts. 2020.

Jorge Pérez, Javier Marinković, and Pablo Barceló. On the turing completeness of modern neural network architectures. In *International Conference on Learning Representations*, 2019.

Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, Gretchen Krueger, and Ilya Sutskever. Learning transferable visual models from natural language supervision, 2021.

Roshan M Rao, Jason Liu, Robert Verkuil, Joshua Meier, John Canny, Pieter Abbeel, Tom Sercu, and Alexander Rives. Msa transformer. In Marina Meila and Tong Zhang (eds.), *Proceedings of the 38th International Conference on Machine Learning*, volume 139 of *Proceedings of Machine Learning Research*, pp. 8844–8856. PMLR, 18–24 Jul 2021.

Patrik Reizinger, Yash Sharma, Matthias Bethge, Bernhard Schölkopf, Ferenc Huszár, and Wieland Brendel. Jacobian-based causal discovery with nonlinear ICA. *Transactions on Machine Learning Research*, 2023. ISSN 2835-8856.

J. M. Replogle, R. A. Saunders, A. N. Pogson, J. A. Hussmann, A. Lenail, A. Guna, L. Mascibroda, E. J. Wagner, K. Adelman, G. Lithwick-Yanai, N. Iremadze, F. Oberstrass, D. Lipson, J. L. Bonnar, M. Jost, T. M. Norman, and J. S. Weissman. Mapping information-rich genotype-phenotype landscapes with genome-scale Perturb-seq. *Cell*, 185(14):2559–2575, Jul 2022.

Karen Sachs, Omar Perez, Dana Pe'er, Douglas A. Lauffenburger, and Garry P. Nolan. Causal protein-signaling networks derived from multiparameter single-cell data. *Science*, 308(5721): 523–529, 2005. doi: 10.1126/science.1105809.

Pedro Sanchez, Xiao Liu, Alison Q. O'Neil, and Sotirios A. Tsaftaris. Diffusion models for causal discovery via topological ordering. In *The Eleventh International Conference on Learning Representations, ICLR 2023, Kigali, Rwanda, May 1-5, 2023*. OpenReview.net, 2023.

Rylan Schaeffer, Brando Miranda, and Oluwasanmi Koyejo. Are emergent abilities of large language models a mirage? *Advances in Neural Information Processing Systems*, abs/2304.15004, 2023.

Gideon Schwarz. Estimating the Dimension of a Model. *The Annals of Statistics*, 6(2):461 – 464, 1978. doi: 10.1214/aos/1176344136.

Shohei Shimizu, Patrik O. Hoyer, Aapo Hyvarinen, and Antti Kerminen. A linear non-gaussian acyclic model for causal discovery. *Journal of Machine Learning Research*, 7(72):2003–2030, 2006.

Peter Spirtes, Clark Glymour, and Richard Scheines. Causality from probability. In *Conference Proceedings: Advanced Computing for the Social Sciences*, 1990.

Peter Spirtes, Christopher Meek, and Thomas Richardson. Causal inference in the presence of latent variables and selection bias. In *Proceedings of the Eleventh Conference on Uncertainty in Artificial Intelligence*, UAI'95, pp. 499–506, San Francisco, CA, USA, 1995. Morgan Kaufmann Publishers Inc. ISBN 1558603859.

Peter Spirtes, Clark Glymour, and Richard Scheines. *Causation, Prediction, and Search*. MIT Press, 2001. doi: https://doi.org/10.7551/mitpress/1754.001.0001.

G'abor J. Sz'ekely, Maria L. Rizzo, and Nail K. Bakirov. Measuring and testing dependence by correlation of distances. *Annals of Statistics*, 35:2769–2794, 2007.

Ioannis Tsamardinos, Laura E Brown, and Constantin F Aliferis. The max-min hill-climbing bayesian network structure learning algorithm. *Machine learning*, 65(1):31–78, 2006.

Tom S. Verma and Judea Pearl. On the equivalence of causal models. In *Proceedings of the Sixth Conference on Uncertainty in Artificial Intelligence*, 1990.

Chulhee Yun, Srinadh Bhojanapalli, Ankit Singh Rawat, Sashank J. Reddi, and Sanjiv Kumar. Are transformers universal approximators of sequence-to-sequence functions? *CoRR*, abs/1912.10077, 2019.

Jiaqi Zhang, Joel Jennings, Cheng Zhang, and Chao Ma. Towards causal foundation model: on duality between causal inference and attention, 2023.

Kun Zhang and Aapo Hyvarinen. On the identifiability of the post-nonlinear causal model, 2012.

Xun Zheng, Bryon Aragam, Pradeep Ravikumar, and Eric P. Xing. Dags with no tears: Continuous optimization for structure learning, 2018.

Yujia Zheng, Biwei Huang, Wei Chen, Joseph Ramsey, Mingming Gong, Ruichu Cai, Shohei Shimizu, Peter Spirtes, and Kun Zhang. Causal-learn: Causal discovery in python. *arXiv preprint arXiv:2307.16405*, 2023.

A Related work, extended

Discrete optimization methods make atomic changes to a proposal graph until a stopping criterion is met. Constraint-based algorithms identify edges based on conditional independence tests, and their correctness is inseparable from the empirical results of those tests Glymour et al. (2019), whose statistical power depends directly on dataset size. These include the FCI and PC algorithms in the observational case Spirtes et al. (1995), and the JCI algorithm in the interventional case Mooij et al. (2020).

Score-based methods also make iterative modifications to a working graph, but their goal is to maximize a continuous score over the discrete space of all valid graphs, with the true graph at the optimum. Due to the intractable search space, these methods often make decisions based on greedy heuristics. Classic examples include GES Chickering (2002), GIES Hauser & Bühlmann (2012), CAM Bühlmann et al. (2014), and LiNGAM Shimizu et al. (2006).

Continuous optimization approaches recast the combinatorial space of graphs into a continuous space of weighted adjacency matrices. Many of these works train a generative model to learn the empirical data distribution, which is parameterized through the adjacency matrix Zheng et al. (2018); Lachapelle et al. (2020); Brouillard et al. (2020). Others focus on properties related to the empirical data distribution, such as a relationship between the underlying graph and the Jacobian of the learned model Reizinger et al. (2023), or between the Hessian of the data log-likelihood and the topological ordering of the nodes Sanchez et al. (2023). While these methods bypass the combinatorial search over discrete graphs, they still require copious data and time to train accurate generative models of the data distributions.

Finally, most similar to this work, amortized inference approaches Ke et al. (2022); Lorch et al. (2022) frame causal discovery as a supervised learning problem within a class of data-generating mechanisms. However, since they operate on raw observations, they do not scale to large datasets or generalize well across different functional classes.

B Proofs and derivations

Our theoretical contributions focus on two primary directions.

1. We formalize the notion of marginal estimates and prove that given sufficient marginal estimates, it is possible to recover a pattern faithful to the global causal graph. We provide lower bounds on the number of marginal estimates required for such a task, and motivate global statistics as an efficient means to reduce this bound.
2. We show that our proposed axial attention has the capacity to recapitulate the reasoning required for marginal estimate resolution. We provide realistic, finite bounds on the width and depth required for this task.

Before these formal discussions, we start with a toy example to provide intuition regarding marginal estimates and constraint-based causal discovery algorithms.

B.1 Toy example: Resolving marginal graphs

Consider the Y-shaped graph with four nodes in Figure 4. Suppose we run the PC algorithm on all subsets of three nodes, and we would like to recover the result of the PC algorithm on the full graph. We illustrate how one might resolve the marginal graph estimates. The PC algorithm consists of the following steps Spirtes et al. (2001).

1. Start from the fully connected, undirected graph on N nodes.
2. Remove all edges (i, j) where $X_i \perp\!\!\!\perp X_j$.
3. For each edge (i, j) and subsets $S \subseteq [N] \setminus \{i, j\}$ of increasing size $n = 1, 2, \ldots, d$, where d is the maximum degree in G, and all $k \in S$ are connected to either i or j: if $X_i \perp\!\!\!\perp X_j \mid S$, remove edge (i, j).
4. For each triplet (i, j, k), such that only edges (i, k) and (j, k) remain, if k was not in the set S that eliminated edge (i, j), then orient the "v-structure" as $i \to k \leftarrow j$.

5. (Orientation propagation) If $i \to j$, edge (j, k) remains, and edge (i, k) has been removed, orient $j \to k$. If there is a directed path $i \rightsquigarrow j$ and an undirected edge (i, j), then orient $i \to j$.

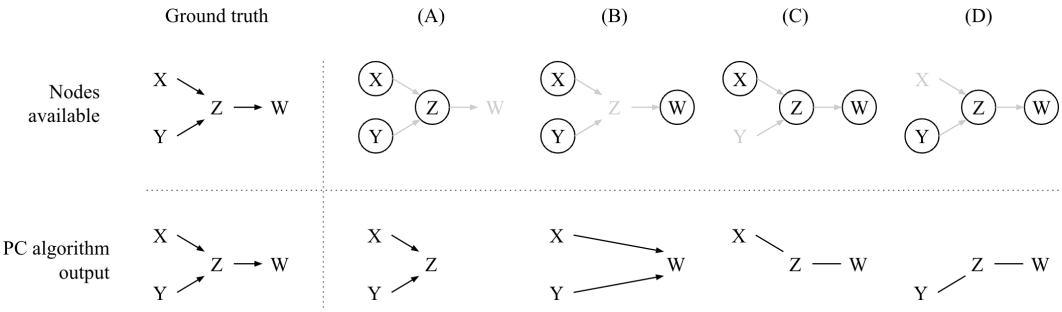

Figure 4: Resolving marginal graphs. Subsets of nodes revealed to the PC algorithm (circled in row 1) and its outputs (row 2).

In each of the four cases, the PC algorithm estimates the respective graphs as follows.

(A) We remove edge (X, Y) via (2) and orient the v-structure.
(B) We remove edge (X, Y) via (2) and orient the v-structure.
(C) We remove edge (X, W) via (3) by conditioning on Z. There are no v-structures, so the edges remain undirected.
(D) We remove edge (Y, W) via (3) by conditioning on Z. There are no v-structures, so the edges remain undirected.

The outputs (A-D) admit the full PC algorithm output as the only consistent graph on four nodes.

- X and Y are unconditionally independent, so no subset will reveal an edge between (X, Y).
- There are no edges between (X, W) and (Y, W). Otherwise, (C) and (D) would yield the undirected triangle.
- X, Y, Z must be oriented as $X \to Z \leftarrow Y$. Paths $X \to Z \to Y$ and $X \leftarrow Z \leftarrow Y$ would induce an (X, Y) edge in (B). Reversing orientations $X \leftarrow Z \to Y$ would contradict (A).
- (Y, Z) must be oriented as $Y \to Z$. Otherwise, (A) would remain unoriented.

B.2 Resolving marginal estimates into global graphs

B.2.1 Preliminaries

Classical results have characterized the Markov equivalency class of directed acyclic graphs. Two graphs are observationally equivalent if they have the same skeleton and v-structures Verma & Pearl (1990). Thus, a pattern P is *faithful* to a graph G if and only if they share the same skeletons and v-structures Spirtes et al. (1990).

Definition B.1. Let $G = (V, E)$ be a directed acyclic graph. A *pattern* P is a set of directed and undirected edges over V.

Definition B.2 (Theorem 3.4 from Spirtes et al. (2001))**.** If pattern P is *faithful* to some directed acyclic graph, then P is faithful to G if and only if

1. for all vertices X, Y of G, X and Y are adjacent if and only if X and Y are dependent conditional on every set of vertices of G that does not include X or Y; and
2. for all vertices X, Y, Z, such that X is adjacent to Y and Y is adjacent to Z and X and Z are not adjacent, $X \to Y \leftarrow Z$ is a subgraph of G if and only if X, Z are dependent conditional on every set containing Y but not X or Z.

Given data faithful to G, a number of classical constraint-based algorithms produce patterns that are faithful to G. We denote this set of algorithms as $\mathcal{F}$.

Theorem B.3 (Theorem 5.1 from Spirtes et al. (2001))**.** *If the input to the PC, SGS, PC-1, PC-2, PC*, or IG algorithms faithful to directed acyclic graph G, the output is a pattern that represents the faithful indistinguishability class of G.*

The algorithms in $\mathcal{F}$ are sound and complete *if* there are no unobserved confounders.

B.2.2 MARGINAL ESTIMATES

Let P_V be a probability distribution that is Markov, minimal, and faithful to G. Let $D \in \mathbb{R}^{M \times N} \sim P_V$ be a dataset of M observations over all $N = |V|$ nodes.

Consider a subset $S \subseteq V$. Let $D[S]$ denote the subset of D over S,

$$D[S] = \{x_{i,v} : v \in S\}_{i=1}^{N}, \tag{3}$$

and let $G[S]$ denote the subgraph of G induced by S

$$G[S] = (S, \{(i,j) : i,j \in S, (i,j) \in E\}. \tag{4}$$

If we apply any $f \in \mathcal{F}$ to $D[S]$, the results are *not* necessarily faithful to $G[S]$, as now there may be latent confounders in $V \setminus S$ (by construction). We introduce the term *marginal estimate* to denote the resultant pattern that, while not faithful to $G[S]$, is still informative.

Definition B.4 (Marginal estimate)**.** A pattern E' is a *marginal estimate* of $G[S]$ if and only if

1. for all vertices X, Y of S, X and Y are adjacent if and only if X and Y are dependent conditional on every set of vertices of S that does not include X or Y; and
2. for all vertices X, Y, Z, such that X is adjacent to Y and Y is adjacent to Z and X and Z are not adjacent, $X \to Y \leftarrow Z$ is a subgraph of S if and only if X, Z are dependent conditional on every set containing Y but not X or Z.

B.2.3 MARGINAL ESTIMATE RESOLUTION

We claim that given marginal estimates on sufficient subsets of nodes, it is always possible to recover a pattern faithful to the entire graph. First we construct a mapping from marginal estimates to the desired pattern, and then we provide tighter bounds on the number of estimates required.

Theorem B.5 (Marginal estimate resolution)**.** *Let $G = (V, E)$ be a directed acyclic graph with maximum degree d. For $S \subseteq V$, let E'_S denote the marginal estimate over S. Let $\mathcal{S}_d$ denote the superset that contains all subsets $S \subseteq V$ of size at most d. There exists a mapping from $\{E'_S\}_{S \in \mathcal{S}_{d+2}}$ to pattern E^*, faithful to G.*

Theorem B.5 formalizes the intuition that it requires at most d independence tests to check whether two nodes are independent, and to estimate the full graph structure, it suffices to run a causal discovery algorithm on subsets of $d + 2$.

We will show that the following algorithm produces the desired answer. On a high level, lines 3-8 recover the undirected "skeleton" graph of E^*, lines 9-15 recover the v-structures, and line 16 references step 5 in Section B.1.

Remark B.6. In the PC algorithm (Spirtes et al. (2001), B.1), its derivatives, and Algorithm 1, there is no need to consider separating sets with cardinality greater than maximum degree d, since the maximum number of independence tests required to separate any node from the rest of the graph is equal to number of its parents plus its children (due to the Markov assumption).

Lemma B.7. *The undirected skeleton of E^* is equivalent to the undirected skeleton of E'*

$$C^* \coloneqq \{\{i,j\} \mid (i,j) \in E^* \text{ or } (j,i) \in E^*\} = \{\{i,j\} \mid (i,j) \in E' \text{ or } (j,i) \in E'\} \coloneqq C'. \tag{5}$$

That is, $\{i,j\} \in C^* \iff \{i,j\} \in C'$.

Proof. It is equivalent to show that $\{i,j\} \notin C^* \iff \{i,j\} \notin C'$

$\Rightarrow$ If $\{i,j\} \notin C*$, then there must exist a separating set S in G of at most size d such that $i \perp\!\!\!\perp j \mid S$. Then $S \cup \{i,j\}$ is a set of at most size $d+2$, where $\{i,j\} \notin C'_{S \cup \{i,j\}}$. Thus, $\{i,j\}$ would have been removed from C' in line 6 of Algorithm 1.

$\Leftarrow$ If $\{i,j\} \notin C'$, let S be a separating set in $\mathcal{S}_{d+2}$ such that $\{i,j\} \notin C'_{S \cup \{i,j\}}$ and $i \perp\!\!\!\perp j \mid S$. S is also a separating set in G, and conditioning on S removes $\{i,j\}$ from C^*. □

Algorithm 1 Resolve marginal estimates of $f \in \mathcal{F}$

Input: Data $\mathcal{D}_G$ faithful to G
Initialize $E' \leftarrow K_N$ as the complete undirected graph on N nodes.
for $S \in \mathcal{S}_{d+2}$ **do**
  Compute $E'_S = f(\mathcal{D}_{G[S]})$
  for $(i, j) \notin E'_S$ **do**
    Remove (i, j) from E'
  end for
end for
for $E'_S \in \{E'_S\}_{\mathcal{S}_{d+2}}$ **do**
  for v-structure $i \to j \leftarrow k$ in E'_S **do**
    if $\{i, j\}, \{j, k\} \in E'$ and $\{i, k\} \notin E'$ **then**
      Assign orientation $i \to j \leftarrow k$ in E'
    end if
  end for
end for
Propagate orientations in E' (optional).

Lemma B.8. *A v-structure $i \to j \leftarrow k$ exists in E^* if and only if there exists the same v-structure in E'.*

Proof. The PCI algorithm orients v-structures $i \to j \leftarrow k$ in E^* if there is an edge between $\{i, j\}$ and $\{j, k\}$ but not $\{i, k\}$; and if j was not in the conditioning set that removed $\{i, k\}$. Algorithm 1 orients v-structures $i \to j \leftarrow k$ in E' if they are oriented as such in any E'_S; and if $\{i, j\}, \{j, k\} \in E', \{i, k\} \notin E'$

$\Rightarrow$ Suppose for contradiction that $i \to j \leftarrow k$ is oriented as a v-structure in E^*, but not in E'. There are two cases.

1. No E'_S contains the undirected path $i - j - k$. If either $i - j$ or $j - k$ are missing from any E'_S, then E^* would not contain (i, j) or (k, j). Otherwise, if all S contain $\{i, k\}$, then E^* would not be missing $\{i, k\}$ (Lemma B.7).
2. In every E'_S that contains $i - j - k$, j is in the conditioning set that removed $\{i, k\}$, i.e. $i \perp\!\!\!\perp k \mid S, S \ni j$. This would violate the faithfulness property, as j is neither a parent of i or k in E^*, and the outputs of the PC algorithm are faithful to the equivalence class of G (Theorem 5.1 Spirtes et al. (2001)).

$\Leftarrow$ Suppose for contradiction that $i \to j \leftarrow k$ is oriented as a v-structure in E', but not in E^*. By Lemma B.7, the path $i - j - k$ must exist in E^*. There are two cases.

1. If $i \to j \to k$ or $i \leftarrow j \leftarrow k$, then j must be in the conditioning set that removes $\{i, k\}$, so no E'_S containing $\{i, j, k\}$ would orient them as v-structures.
2. If j is the root of a fork $i \leftarrow j \to k$, then as the parent of both i and k, j must be in the conditioning set that removes $\{i, k\}$, so no E'_S containing $\{i, j, k\}$ would orient them as v-structures.

Therefore, all v-structures in E' are also v-structures in E^*. □

Proof of Theorem B.5. Given data that is faithful to G, Algorithm 1 produces a pattern E' with the same connectivity and v-structures as E^*. Any additional orientations in both patterns are propagated using identical, deterministic procedures, so $E' = E^*$. □

This proof presents a deterministic but inefficient algorithm for resolving marginal subgraph estimates. In reality, it is possible to recover the undirected skeleton and the v-structures of G without checking all subsets $S \in \mathcal{S}_{d+2}$.

Proposition B.9 (Skeleton bounds)**.** *Let $G = (V, E)$ be a directed acyclic graph with maximum degree d. It is possible to recover the undirected skeleton $C = \{\{i, j\} : (i, j) \in E\}$ in $O(N^2)$ estimates over subsets of size $d + 2$.*

Proof. Following Lemma B.7, an edge (i, j) is not present in C if it is not present in any of the size $d + 2$ estimates. Therefore, every pair of nodes $\{i, j\}$ requires only a single estimate of size $d + 2$, so it is possible to recover C in $\binom{N}{2}$ estimates. □

If we only leverage marginal estimates, we must check at least $\binom{N}{2}$ subsets to cover each edge at least once. However, we can often approximate the skeleton via global statistics such as correlation, allowing us to use marginal estimates more efficiently, towards answering orientation questions.

Proposition B.10 (V-structures bounds)**.** *Let $G = (V, E)$ be a directed acyclic graph with maximum degree d and ν v-structures. It is possible to identify all v-structures in $O(\nu)$ estimates over subsets of at most size $d + 2$.*

Proof. Each v-structure $i \to j \leftarrow k$ falls under two cases.

1. $i \perp\!\!\!\perp k$ unconditionally. Then an estimate over $\{i, j, k\}$ will identify the v-structure.
2. $i \perp\!\!\!\perp k \mid S$, where $j \notin S \subset V$. Then an estimate over $S \cup \{i, j, k\}$ will identify the v-structure. Note that $|S| \leq d + 2$ since the degree of i is at least $|S| + 1$.

Therefore, each v-structure only requires one estimate, and it is possible to identify all v-structures in $O(\nu)$ estimates. □

There are three takeaways from this section.

1. If we exhaustively run a constraint-based algorithm on all subsets of size $d + 2$, it is trivial to recover the estimate of the full graph. However, this is no more efficient than running the causal discovery algorithm on the full graph.
2. In theory, it is possible to recover the undirected graph in $O(N^2)$ estimates, and the v-structures in $O(\nu)$ estimates. However, we may not know the appropriate subsets ahead of time.
3. In practice, if we have a surrogate for connectivity, such as the global statistics used in SEA, then we can vastly reduce the number of estimates used to eliminate edges from consideration, and more effectively focus on sampling subsets for orientation determination.

B.3 COMPUTATIONAL POWER OF THE AXIAL ATTENTION MODEL

Existing literature on the universality and computational power of vanilla Transformers Yun et al. (2019); Pérez et al. (2019) rely on generous assumptions regarding depth or precision. Here, we show that our axial attention-based model can implement the specific reasoning required to resolve marginal estimates under realistic conditions.

We prove that three blocks can recover the skeleton and v-structures in $O(N)$ width, and additional blocks have the capacity to propagate orientations. We first formalize the notion of a neural network architecture's capacity to "implement" an algorithm. Then we prove Theorem B.13 by construction.

Definition B.11. Let f be a map from finite sets Q to F, and let ϕ be a map from finite sets Q_Φ to F_Φ. We say ϕ *implements* f if there exists injection $g_{\text{in}} : Q \to Q_\Phi$ and surjection $g_{\text{out}} : F_\Phi \to F$ such that

$$\forall q \in Q, g_{\text{out}}(\phi(g_{\text{in}}(q))) = f(q). \tag{6}$$

Definition B.12. Let Q, F, Q_Φ, F_Φ be finite sets. Let f be a map from Q to F, and let Φ be a finite set of maps $\{\phi : Q_\Phi \to F_\Phi\}$. We say Φ has the *capacity* to implement f if and only if there exists at least one element $\phi \in \Phi$ that implements f.

Theorem B.13 (Model capacity)**.** *Given a graph G with N nodes, a stack of L axial attention blocks has the capacity to recover its skeleton and v-structures in $O(N)$ width, and propagate orientations on paths of $O(L)$ length.*

Proof. We consider axial attention blocks with dot-product attention and omit layer normalization from our analysis, as is common in the Transformer universality literature Yun et al. (2019). Our inputs $X \in \mathbb{R}^{d \times R \times C}$ consist of d-dimension embeddings over R rows and C columns. Since our axial attention only operates over one dimension at a time, we use $X_{\cdot,c}$ to denote a 1D sequence of length R, given a fixed column c, and $X_{r,\cdot}$ to denote a 1D sequence of length C, given a fixed row

r. A single axial attention layer (with one head) consists of two attention layers and a feedforward network,

$$\text{Attn}_{\text{row}}(X_{\cdot,c}) = X_{\cdot,c} + W_O W_V X_{\cdot,c} \cdot \sigma\left[(W_K X_{\cdot,c})^T W_Q X_{\cdot,c}\right], \tag{7}$$
$$X \leftarrow \text{Attn}_{\text{row}}(X)$$
$$\text{Attn}_{\text{col}}(X_{r,\cdot}) = X_{r,\cdot} + W_O W_V X_{r,\cdot} \cdot \sigma\left[(W_K X_{r,\cdot})^T W_Q X_{r,\cdot}\right], \tag{8}$$
$$X \leftarrow \text{Attn}_{\text{col}}(X)$$
$$\text{FFN}(X) = X + W_2 \cdot \text{ReLU}(W_1 \cdot X + b_1 \mathbf{1}_L^T) + b_2 \mathbf{1}_L^T, \tag{9}$$

where $W_O \in \mathbb{R}^{d\times d}, W_V, W_K, W_Q \in \mathbb{R}^{d\times d}, W_2 \in \mathbb{R}^{d\times m}, W_1 \in \mathbb{R}^{m\times d}, b_2 \in \mathbb{R}^d, b_1 \in \mathbb{R}^m$, and m is the hidden layer size of the feedforward network. For concision, we have omitted the r and c subscripts on the Ws, but the row and column attentions use different parameters. Any row or column attention can take on the identity mapping by setting W_O, W_V, W_K, W_Q to $d \times d$ matrices of zeros.

A single axial attention *block* consists of two axial attention layers ϕ_E and ϕ_ρ, connected via messages (Section 3.2)

$$h^{E,\ell} = \phi_{E,\ell}(h^{E,\ell-1})$$
$$h^{\rho,\ell-1} \leftarrow W_{\rho,\ell}\left[h^{\rho,\ell-1}, m^{E\to\rho,\ell}\right]$$
$$h^{\rho,\ell} = \phi_{\rho,\ell}(h^{\rho,\ell-1})$$
$$h^{E,\ell} \leftarrow h^{E,\ell} + m^{\rho\to E,\ell}$$

where h^ℓ denote the hidden representations of E and ρ at layer ℓ, and the outputs of the axial attention block are $h^{\rho,\ell}, h^{E,\ell}$.

We construct a stack of $L \geq 3$ axial attention blocks that implement Algorithm 1.

Model inputs Consider edge estimate $E'_{i,j} \in \mathcal{E}$ in a graph of size N. Let e_i, e_j denote the endpoints of (i, j). Outputs of the PC algorithm can be expressed by three endpoints: $\{\varnothing, \bullet, \blacktriangleright\}$. A directed edge from $i \to j$ has endpoints $(\bullet, \blacktriangleright)$, the reversed edge $i \leftarrow j$ has endpoints $(\blacktriangleright, \bullet)$, an undirected edge has endpoints $(\bullet, \bullet)$, and the lack of any edge between i, j has endpoints $(\varnothing, \varnothing)$.

Let one-hot$_N(i)$ denote the N-dimensional one-hot column vector where element i is 1. We define the embedding of (i, j) as a $d = 2N + 6$ dimensional vector,

$$g_{\text{in}}(E_{t,(i,j)}) = h^{E,0}_{(i,j)} = \left[\begin{array}{c} \text{one-hot}_3(e_i) \\ \hline \text{one-hot}_3(e_j) \\ \hline \text{one-hot}_N(i) \\ \hline \text{one-hot}_N(j) \end{array}\right]. \tag{10}$$

To recover graph structures from h^E, we simply read off the indices of non-zero entries (g_{out}). We can set $h^{\rho,0}$ to any $\mathbb{R}^{d\times N\times N}$ matrix, as we do not consider its values in this analysis and discard it during the first step.

Claim B.14. *(Consistency) The outputs of each step*

1. *are consistent with (10), and*
2. *are equivariant to the ordering of nodes in edges.*

For example, if (i, j) is oriented as $(\blacktriangleright, \bullet)$, then we expect (j, i) to be oriented $(\bullet, \blacktriangleright)$.

Step 1: Undirected skeleton We use the first axial attention block to recover the undirected skeleton C'. We set all attentions to the identity, set $W_{\rho,1} \in \mathbb{R}^{2d\times d}$ to a $d \times d$ zeros matrix, stacked on top of a $d \times d$ identity matrix (discard ρ), and set FFN$_E$ to the identity (inputs are positive). This yields

$$h^{\rho,0}_{i,j} = m^{E\to\rho,1}_{i,j} = \left[\begin{array}{c} P_{e_i}(\varnothing) \\ P_{e_i}(\bullet) \\ P_{e_i}(\blacktriangleright) \\ \hline \vdots \\ \hline \text{one-hot}_N(i) \\ \hline \text{one-hot}_N(j) \end{array}\right], \tag{11}$$

where $P_{e_i}(\cdot)$ is the frequency that endpoint $e_i = \cdot$ within the subsets sampled. FFNs with 1 hidden layer are universal approximators of continuous functions Hornik et al. (1989), so we use FFN_ρ to map

$$\text{FFN}_\rho(X_{i,u,v}) = \begin{cases} 0 & i \leq 6 \\ 0 & i > 6, X_{1,u,v} = 0 \\ -X_{i,u,v} & \text{otherwise,} \end{cases} \tag{12}$$

where $i \in [2N + 6]$ indexes into the feature dimension, and u, v index into the rows and columns. This allows us to remove edges not present in C' from consideration:

$$m^{\rho \to E,1} = h^{\rho,1}$$
$$h^{E,1}_{i,j} \leftarrow h^{E,1}_{i,j} + m^{\rho \to E,1}_{i,j} = \begin{cases} 0 & (i,j) \notin C' \\ h^{E,0}_{i,j} & \text{otherwise.} \end{cases} \tag{13}$$

This yields $(i, j) \in C'$ if and only if $h^{\rho,1}_{i,j} \neq \mathbf{0}$. We satisfy B.14 since our inputs are valid PC algorithm outputs for which $P_{e_i}(\varnothing) = P_{e_j}(\varnothing)$.

Step 2: V-structures The second and third axial attention blocks recover v-structures. We run the same procedure twice, once to capture v-structures that point towards the first node in an ordered pair, and one to capture v-structures that point towards the latter node.

We start with the first row attention over edge estimates, given a fixed subset t. We set the key and query attention matrices

$$W_K = k \cdot \begin{bmatrix} 0 & 0 & 1 & & & & & & \\ & & & 0 & 1 & 0 & & & \\ & \vdots & & & & & & & \\ & & & & & & I_N & & \\ & & & & & & & -I_N \end{bmatrix} \quad W_Q = k \cdot \begin{bmatrix} 0 & 0 & 1 & & & & & & \\ & & & 0 & 1 & 0 & & & \\ & \vdots & & & & & & & \\ & & & & & & I_N & & \\ & & & & & & & I_N \end{bmatrix} \tag{14}$$

where k is a large constant, I_N denotes the size N identity matrix, and all unmarked entries are 0s.

Recall that a v-structure is a pair of directed edges that share a target node. We claim that two edges $(i, j), (u, v)$ form a v-structure in E', pointing towards $i = u$, if this inner product takes on the maximum value

$$\left\langle (W_K h^{E,1})_{i,j}, (W_Q h^{E,1})_{u,v} \right\rangle = 3. \tag{15}$$

Suppose both edges (i, j) and (u, v) still remain in C'. There are two components to consider.

1. If $i = u$, then their shared node contributes $+1$ to the inner product (prior to scaling by k). If $j = v$, then the inner product accrues -1.
2. Nodes that do not share the same endpoint contribute 0 to the inner product. Of edges that share one node, only endpoints that match ▶ at the starting node, or • at the ending node contribute $+1$ to the inner product each. We provide some examples below.

(e_i, e_j)	(e_u, e_v)	contribution	note
(▶, •)	(•, ▶)	0	no shared node
(•, ▶)	(•, ▶)	0	wrong endpoints
(•, •)	(•, •)	1	one correct endpoint
(▶, •)	(▶, •)	2	v-structure

All edges with endpoints $\varnothing$ were "removed" in step 1, resulting in an inner product of zero, since their node embeddings were set to zero. We set k to some large constant (empirically, $k^2 = 1000$ is more than enough) to ensure that after softmax scaling, $\sigma_{e,e'} > 0$ only if e, e' form a v-structure.

Given ordered pair $e = (i, j)$, let $V_i \subset V$ denote the set of nodes that form a v-structure with e with shared node i. Note that V_i excludes j itself, since setting of W_K, W_Q exclude edges that share both nodes. We set W_V to the identity, and we multiply by attention weights σ to obtain

$$(W_V h^{E,1} \sigma)_{e=(i,j)} = \begin{bmatrix} \vdots \\ \hline \text{one-hot}_N(i) \\ \hline \alpha_j \cdot \text{binary}_N(V_j) \end{bmatrix} \tag{16}$$

where $\text{binary}_N(S)$ denotes the N-dimensional binary vector with ones at elements in S, and the scaling factor

$$\alpha_j = (1/\|V_j\|) \cdot \mathbb{1}\{\|V_j\| > 0\} \in [0, 1] \tag{17}$$

results from softmax normalization. We set

$$W_O = \begin{bmatrix} \mathbf{0}_{N+6} & \\ & 0.5 \cdot I_N \end{bmatrix} \tag{18}$$

to preserve the original endpoint values, and to distinguish between the edge's own node identity and newly recognized v-structures. To summarize, the output of this row attention layer is

$$\text{Attn}_{\text{row}}(X_{\cdot,c}) = X_{\cdot,c} + W_O W_V X_{\cdot,c} \cdot \sigma,$$

which is equal to its input $h^{E,1}$ plus additional positive values $\in (0, 0.5)$ in the last N positions that indicate the presence of v-structures that exist in the overall E'.

Our final step is to "copy" newly assigned edge directions into all the edges. We set the ϕ_E column attention, FFN_E and the ϕ_ρ attentions to the identity mapping. We also set $W_{\rho,2}$ to a $d \times d$ zeros matrix, stacked on top of a $d \times d$ identity matrix. This passes the output of the ϕ_E row attention, aggregated over subsets, directly to $\text{FFN}_{\phi,2}$.

For endpoint dimensions $\mathbf{e} = [6]$, we let $\text{FFN}_{\phi,2}$ implement

$$\text{FFN}_{\rho,2}(X_{\mathbf{e},u,v}) = \begin{cases} [0,0,1,0,1,0]^T - X_{\mathbf{e},u,v} & 0 < \sum_{i>N+6} X_{i,u,v} < 0.5 \\ 0 & \text{otherwise.} \end{cases} \tag{19}$$

Subtracting $X_{\mathbf{e},u,v}$ "erases" the original endpoints and replaces them with $(\blacktriangleright, \bullet)$ after the update

$$h^{E,1}_{i,j} \leftarrow h^{E,1}_{i,j} + m^{\rho \to E,1}_{i,j}.$$

The overall operation translates to checking whether *any* v-structure points towards i, and if so, assigning edge directions accordingly. For dimensions $i > 6$,

$$\text{FFN}_{\rho,2}(X_{i,u,v}) = \begin{cases} -X_{i,u,v} & X_{i,u,v} \leq 0.5 \\ 0 & \text{otherwise,} \end{cases} \tag{20}$$

effectively erasing the stored v-structures from the representation and remaining consistent to (10).

At this point, we have copied all v-structures once. However, our orientations are not necessarily symmetric. For example, given v-structure $i \to j \leftarrow k$, our model orients edges (j, i) and (j, k), but not (i, j) or (k, j).

The simplest way to symmetrize these edges (for the writer and the reader) is to run another axial attention block, in which we focus on v-structures that point towards the second node of a pair. The only changes are as follows.

- For W_K and W_Q, we swap columns 1-3 with 4-6, and columns 7 to $N + 6$ with the last N columns.
- $(h^{E,2}\sigma)_{i,j}$ sees the third and fourth blocks swapped.
- W_O swaps the $N \times N$ blocks that correspond to i and j's node embeddings.
- $\text{FFN}_{\rho,3}$ sets the endpoint embedding to $[0,1,0,0,0,1]^T - X_{\mathbf{e},u,v}$ if $i = 7, ..., N + 6$ sum to a value between 0 and 0.5.

The result is $h^{E,3}$ with all v-structures oriented symmetrically, satisfying B.14.

Step 3: Orientation propagation To propagate orientations, we would like to identify cases $(i, j), (i, k) \in E', (j, k) \notin E'$ with shared node i and corresponding endpoints $(\blacktriangleright, \bullet), (\bullet, \bullet)$. We use ϕ_E to identify triangles, and ϕ_ρ to identify edges $(i, j), (i, k) \in E'$ with the desired endpoints, while ignoring triangles.

Marginal layer The row attention in ϕ_E fixes a subset t and varies the edge (i, j).

Given edge (i, j), we want to extract all (i, k) that share node i. We set the key and query attention matrices to

$$W_K, W_Q = k \cdot \begin{bmatrix} 0 & 1 & 1 & 0 & 1 & 1 & & \\ \vdots & & & & & & & \\ & & & & & & I_N & \\ & & & & & & & \pm I_N \end{bmatrix}. \tag{21}$$

We set W_V to the identity to obtain

$$(W_V h^E \sigma)_{e=(i,k)} = \begin{bmatrix} \vdots \\ \hline \vdots \\ \hline \text{one-hot}_N(i) \\ \hline \alpha_k \cdot \text{binary}_N(V_k) \end{bmatrix}, \tag{22}$$

where V_k is the set of nodes k that share any edge with i. To distinguish between k and V_k, we again set W_o to the same as in (18). Finally, we set FFN_E to the identity and pass h^E directly to ϕ_ρ. To summarize, we have h^E equal to its input, with values $\in (0, 0.5)$ in the last N locations indicating 1-hop neighbors of each edge.

Global layer Now we would like to identify cases $(i, k), (j, k)$ with corresponding endpoints $(\bullet, \blacktriangleright), (\bullet, \bullet)$. We set the key and query attention matrices

$$W_K = k \cdot \begin{bmatrix} 0 & 0 & 1 & & \\ \vdots & & & & \\ & & & I_N & \\ & & & & I_N \end{bmatrix} \quad W_Q = k \cdot \begin{bmatrix} 0 & 1 & -1 & 0 & 1 & -1 & & \\ \vdots & & & & & & & \\ & & & & & & I_N & \\ & & & & & & & -I_N \end{bmatrix}. \tag{23}$$

The key allows us to check that endpoint i is directed, and the query allows us to check that (i, k) exists in C', and does not already point elsewhere. After softmax normalization, for sufficiently large k, we obtain $\sigma_{(i,j),(i,k)} > 0$ if and only if (i, k) should be oriented $(\bullet, \blacktriangleright)$, and the inner product attains the maximum possible value

$$\langle (W_K h^\rho)_{i,j}, (W_Q h^\rho)_{i,k} \rangle = 2. \tag{24}$$

We consider two components.

1. If the endpoints match our desired endpoints, we gain a $+1$ contribution to the inner product.
2. A match between the first nodes contributes $+1$. If the second node shares any overlap (either same edge, or a triangle), then a negative value would be added to the overall inner product.

Therefore, we can only attain the maximal inner product if only one edge is directed, and if there exists no triangle.

We set W_o to the same as in (18), and we add h^ρ to the input of the next ϕ_E. To summarize, we have h^ρ equal to its input, with values $\in (0, 0.5)$ in the last N locations indicating incoming edges.

Orientation assignment Our final step is to assign our new edge orientations. Let the column attention take on the identity mapping. For endpoint dimensions $\mathbf{e} = (4, 5, 6)$, we let FFN_ρ implement

$$\text{FFN}_\rho(X_{\mathbf{e},u,v}) = \begin{cases} [0, 0, 1]^T - X_{\mathbf{e},u,v} & 0 < \sum_{i>N+6} X_{i,u,v} < 0.5 \\ 0 & \text{otherwise.} \end{cases} \tag{25}$$

This translates to checking whether any incoming edge points towards v, and if so, assigning the new edge direction accordingly. For dimensions $i > 6$,

$$\text{FFN}_\rho(X_{i,u,v}) = \begin{cases} 0 & X_{i,u,v} \leq 0.5 \\ X_{i,u,v} & \text{otherwise,} \end{cases} \tag{26}$$

effectively erasing the stored assignments from the representation. Thus, we are left with $h^{E,\ell}$ that conforms to the same format as the initial embedding in (10).

To symmetrize these edges, we run another axial attention block, in which we focus on paths that point towards the second node of a pair. The only changes are as follows.

- For ϕ_E layer W_K and W_Q (21), we swap I_N and $\pm I_N$.
- For ϕ_ρ layer W_K and W_Q (23), we swap I_N and $\pm I_N$.
- W_O swaps the $N \times N$ blocks that correspond to i and j's node embeddings.
- For FFN_ρ (25), we let $\mathbf{e} = (1, 2, 3)$ instead.

The result is h^E with symmetric 1-hop orientation propagation, satisfying B.14. We may repeat this procedure k times to capture k-hop paths.

To summarize, we used axial attention block 1 to recover the undirected skeleton C', blocks 2-3 to identify and copy v-structures in E', and all subsequent $L - 3$ layers to propagate orientations on paths up to $\lfloor (L-3)/2 \rfloor$ length. Overall, this particular construction requires $O(N)$ width for $O(L)$ paths.

□

Final remarks Information theoretically, it should be possible to encode the same information in $\log N$ space, and achieve $O(\log N)$ width. For ease of construction, we have allowed for wider networks than optimal.

On the other hand, if we increase the width and encode each edge symmetrically, e.g. $(e_i, e_j, e_j, e_i \mid i, j, j, i)$, we can reduce the number of blocks by half, since we no longer need to run each operation twice. However, attention weights scale quadratically, so we opted for an asymmetric construction.

Finally, a strict limitation of our model is that it only considers 1D pairwise interactions. In the graph layer, we cannot compare different edges' estimates at different times in a single step. In the feature layer, we cannot compare (i, j) to (j, i) in a single step either. However, the graph layer does enable us to compare all edges at once (sparsely), and the feature layer looks at a time-collapsed version of the whole graph. Therefore, though we opted for this design for computational efficiency, we have shown that it is able to capture significant graph reasoning.

B.4 Robustness and stability

We discuss the notion of stability informally, in the context of Spirtes et al. (2001). There are two cases in which our framework may receive erroneous inputs: low/noisy data settings, and functionally misspecified situations. We consider our framework's robustness to these cases, in terms of recovering the skeleton and orienting edges.

B.4.1 Data noise

In the case of noisy data, edges may be erroneously added, removed, or misdirected from marginal estimates E'. Our framework provides two avenues to mitigating such noise.

1. We observe that global statistics can be estimated reliably in low data scenarios. For example, Figure 7 suggests that 200 examples suffice to provide a robust estimate over 100 variables in our synthetic settings. Therefore, even if the marginal estimates are erroneous, the neural network can learn the skeleton from the global statistics.
2. Most classical causal discovery algorithms are not stable with respect to edge orientation assignment. That is, an error in a single edge may propagate throughout the graph. Empirically, we observe that the majority vote of GIES achieves reasonable accuracy even without any training, while FCI suffers in this assessment (Table 4). However both SEA (GIES) and SEA (FCI) achieve high edge accuracy. Therefore, while the underlying algorithms may not be stable with respect to edge orientation, our pretrained aggregator seems to be robust.

B.4.2 FUNCTIONAL MISSPECIFICATION

It is also possible that our global statistics and marginal estimates make misspecified assumptions regarding the data generating mechanisms. The degree of misspecification can vary case by case, so it is hard to provide any broad guarantees about the performance of our algorithm, in general. However, we can make the following observation.

If two variables are independent, $X_i \perp\!\!\!\perp X_j$, they are independent, e.g. under linear Gaussian assumptions. If X_i, X_j exhibit more complex functional dependencies, they may be erroneously deemed independent. Therefore, any systematic errors are necessarily one-sided, and the model can learn to recover the connectivity based on global statistics.

C EXPERIMENTAL DETAILS

C.1 SYNTHETIC DATA GENERATION

Synthetic datasets were generated using code from DCDI Brouillard et al. (2020), which extended the Causal Discovery Toolkit data generators to interventional data Kalainathan et al. (2020).

The synthetic datasets were constructed based on Table 2. For each synthetic dataset, we first sample a graph based on the desired topology and connectivity. Then we topologically sort the graph and sample observations starting from the root nodes, using random instantiations of the designated causal mechanism (details in C.1). We generated 90 training, 5 validation, and 5 testing datasets for each combination (8160 total). To evaluate our model's capacity to generalize to new functional classes, we reserve the Sigmoid and Polynomial causal mechanisms for testing only.

We considered the following causal mechanisms. Let y be the node in question, let X be its parents, let E be an independent noise variable (details below), and let W be randomly initialized weight matrices.

- Linear: $y = XW + E$.
- Polynomial: $y = W_0 + XW_1 + X^2W_2 + E$
- Sigmoid additive: $y = \sum_{i=1}^{d} W_i \cdot \text{sigmoid}(X_i) + E$
- Randomly initialized neural network (NN): $y = \text{Tanh}((X, E)W_{\text{in}})W_{\text{out}}$
- Randomly initialized neural network, additive (NN additive): $y = \text{Tanh}(XW_{\text{in}})W_{\text{out}} + E$

Root causal mechanisms, noise variables, and interventional distributions maintained the DCDI defaults.

- Root causal mechanisms were set to $\text{Uniform}(-2, 2)$.
- Noise was set to $E \sim 0.4 \cdot \mathcal{N}(0, \sigma^2)$ where $\sigma^2 \sim \text{Uniform}(1, 2)$.
- Interventions were applied to all nodes (one at a time) by setting their causal mechanisms to $\mathcal{N}(0, 1)$.

Ablation datasets with $N > 100$ nodes contained 100,000 points each (same as $N = 100$).

Table 2: Synthetic data generation settings. The symbol $*$ denotes training only, and † denotes testing only. We take the Cartesian product of all parameters for our settings. (Non)-additive refers to (non)-additive Gaussian noise. Details regarding the causal mechanisms can be found in C.1

Parameter	Values
Nodes (N)	$10, 20, 100$
Edges	$N, 2N^*, 3N^*, 4N$
Points	$1000N$
Interventions	N
Topology	Erdős-Rényi, Scale Free
Mechanism	Linear, NN additive, NN non-additive, Sigmoid additive†, Polynomial additive†

C.2 BASELINE DETAILS

We considered the following baselines. All baselines were run using official implementations published by the authors. All deep learning models were run on a single V100-PCIE-32GB GPU, except for DIFFAN, since we were unable to achieve consistent GPU and R support within a Docker environment using their codebase. For all models, we recorded only computation time (CPU and GPU) and omitted any file system-related time.

DCDI Brouillard et al. (2020) was trained on each of the $N = 10, 20$ datasets using their published hyperparameters. We denote the Gaussian and Deep Sigmoidal Flow versions as DCDI-G and DCDI-DSF respectively. DCDI could not scale to graphs with $N = 100$ due to memory constraints (did not fit on a 32GB V100 GPU).

DCD-FG Lopez et al. (2022) was trained on all of the test datasets using their published hyperparameters. We set the number of factors to $5, 10, 20$ for each of $N = 10, 20, 100$, based on their ablation studies. Due to numerical instability on $N = 100$, we clamped augmented Lagrangian multipliers μ and γ to 10 and stopped training if elements of the learned adjacency matrix reached `NaN` values. After discussion with the authors, we also tried adjusting the μ multiplier from 2 to 1.1, but the model did not converge within 48 hours.

DECI Geffner et al. (2022) was trained on all of the test datasets using their published hyperparameters. However, on all $N = 100$ cases, the model failed to produce any meaningful results (adjacency matrices nearly all remained 0s with AUCs of 0.5). Thus, we only report results on $N = 10, 20$.

DIFFAN Sanchez et al. (2023) was trained on the each of the $N = 10, 20$ datasets using their published hyperparameters. The authors write that "most hyperparameters are hard-coded into [the] constructor of the DIFFAN class and we verified they work across a wide set of datasets." We used the original, non-approximation version of their algorithm by maintaining `residue=True` in their codebase. We were unable to consistently run DIFFAN with both R and GPU support within a Docker container, and the authors did not respond to questions regarding reproducibility, so all models were trained on the CPU only. We observed approximately a 10x speedup in the < 5 cases that were able to complete running on the GPU.

C.3 NEURAL NETWORK DESIGN

C.3.1 MODEL ARCHITECTURE

We expand upon the architecture details, described in Section 3.2.

Embeddings We project global statistics into the model dimension via a learned linear projection matrix $W_\rho : \mathbb{R} \to \mathbb{R}^d$, and we embed edge types via a learned embedding $\text{ebd}_\mathcal{E} : \mathcal{E} \to \mathbb{R}^d$. To collect estimates of the same edge over all subsets, we align entries of $E'_{1\ldots T}$ into $E'^{\text{align}}_T \in \mathcal{E}^{T\times K}$

$$E'^{\text{align}}_{t,e=(i,j)} = \begin{cases} E'_{t,i,j} & \text{if } i \in S_t, j \in S_t \\ 0 & \text{otherwise} \end{cases} \tag{27}$$

where t indexes into the subsets, e indexes into the set of unique edges, and K is the number of unique edges.

We add learned 1D positional embeddings along both dimensions of each input,

$$\begin{aligned} \text{pos-ebd}(\rho_{i,j}) = &\ \text{ebd}_{\text{node}}(i') + \text{ebd}_{\text{node}}(j') \\ \text{pos-ebd}(E'^{\text{align}}_{t,e}) = &\ \text{ebd}_{\text{time}}(t) \\ &+ \text{FFN}([\text{ebd}_{\text{node}}(i'), \text{ebd}_{\text{node}}(j')]) \end{aligned}$$

where i', j' index into a random permutation on V for invariance to node permutation and graph size.[1] Due to the (a)symmetries of their inputs, $\text{pos-ebd}(\rho_{i,j})$ is symmetric, while $\text{pos-ebd}(E'^{\text{align}}_{t,e})$

[1]The sampling of S_t already provides invariance to node order. However, the mapping $i' = \sigma(V)_i$ allows us to avoid updating positional embeddings of lower order positions more than higher order ones, due to the mixing of graph sizes during training.

considers the node ordering. In summary, the inputs to our axial attention blocks are

$$h^{\rho}_{i,j} = (W_\rho \rho)_{i,j} + \text{pos-ebd}(\rho_{i,j}) \tag{28}$$

$$h^{E}_{t,e} = \text{ebd}_{\mathcal{E}}(E'^{\text{align}}_{t,e}) + \text{pos-ebd}(E'^{\text{align}}_{t,e}) \tag{29}$$

for $i, j \in [N]^2$, $t \in [T]$, $e \in [K]$.

Axial attention An axial attention block contains two axial attention layers (marginal estimates, global features) and a feed-forward network (Figure 2, right).

Given a 2D input, an axial attention layer attends first along the rows, then along the columns. For example, given a matrix of size `(R,C,d)`, one pass of the axial attention layer is equivalent to running standard self-attention along `C` with batch size `R`, followed by the reverse. For marginal estimates, `R` is the number of subsets T, and `C` is the number of unique edges K. For global features, `R` and `C` are both the total number of vertices N.

Following Rao et al. (2021), each self-attention mechanism is preceded by layer normalization and followed by dropout, with residual connections to the input,

$$x = x + \text{Dropout}(\text{Attn}(\text{LayerNorm}(x))). \tag{30}$$

We pass messages between the marginal and global layers to propagate information. Let $\phi_{E,\ell}$ be marginal layer ℓ, let $\phi_{\rho,\ell}$ be global layer ℓ, and let $h^{\cdot,\ell}$ denote the hidden representations out of layer ℓ.

The marginal to global message $m^{E\to\rho} \in \mathbb{R}^{N\times N\times d}$ contains representations of each edge averaged over subsets,

$$m^{E\to\rho,\ell}_{i,j} = \begin{cases} \frac{1}{T_e}\sum_t h^{E,\ell}_{t,e=(i,j)} & \text{if } \exists S_t, i, j \in S_t \\ \epsilon & \text{otherwise.} \end{cases} \tag{31}$$

where T_e is the number of S_t containing e, and missing entries are padded to learned constant ϵ. The global to marginal message $m^{\rho\to E} \in \mathbb{R}^{K\times d}$ is simply the hidden representation itself,

$$m^{\rho\to E,\ell}_{t,e=(i,j)} = h^{\rho,\ell}_{i,j}. \tag{32}$$

We incorporate these messages as follows.

$$h^{E,\ell} = \phi_{E,\ell}(h^{E,\ell-1}) \quad \text{(marginal feature)} \tag{33}$$

$$h^{\rho,\ell-1} \leftarrow W^\ell \left[h^{\rho,\ell-1}, m^{E\to\rho,\ell}\right] \quad \text{(marginal to global)} \tag{34}$$

$$h^{\rho,\ell} = \phi_{\rho,\ell}(h^{\rho,\ell-1}) \quad \text{(global feature)} \tag{35}$$

$$h^{E,\ell} \leftarrow h^{E,\ell} + m^{\rho\to E,\ell} \quad \text{(global to marginal)} \tag{36}$$

$W^\ell \in \mathbb{R}^{2d\times d}$ is a learned linear projection, and $[\cdot]$ denotes concatenation.

Graph prediction For each pair of vertices $i \neq j \in V$, we predict $e = 0, 1$, or 2 for no edge, $i \to j$, and $j \to i$ respectively. This constraint may be omitted for cyclic graphs. We do not additionally enforce that our predicted graphs are acyclic, similar in spirit to Lippe et al. (2022).

Given the output of the final axial attention block, h^ρ, we compute logits

$$z_{\{i,j\}} = \text{FFN}\left(\left[h^{\rho}_{i,j}, h^{\rho}_{j,i}\right]\right) \in \mathbb{R}^3 \tag{37}$$

which correspond to probabilities after softmax normalization.

C.3.2 HYPERPARAMETERS

Hyperparameters and architectural choices were selected by training the model on 20% of the the training and validation data for approximately 50k steps (several hours). We considered the following parameters in sequence.

- learned positional embedding vs. sinusoidal positional embedding

- number of layers $\times$ number of heads: $\{4, 8\} \times \{4, 8\}$
- learning rate $\eta = \{1e-4, 5e-5, 1e-5\}$

For our final model, we selected learned positional embeddings, 4 layers, 8 heads, and learning rate $\eta = 1e-4$.

C.4 Training and hardware details

The models were trained across 2 NVIDIA RTX A6000 GPUs and 60 CPU cores. We used the GPU exclusively for running the aggregator, and retained all classical algorithm execution on the CPUs (during data loading). The total pretraining time took approximately 14 hours for the final FCI model and 16 hours for the final GIES model.

For the scope of this paper, our models and datasets are fairly small. We did not scale further due to hardware constraints. Our primary bottlenecks to scaling up lay in availability of CPU cores and networking speed across nodes, rather than GPU memory or utilization.

We are able to run inference comfortably over $N = 500$ graphs with $T = 500$ subsets of $k = 5$ nodes each, on a single 32GB V100 GPU. For runtime analysis, we used a batch size of 1, with 1 data worker per dataset. Runtime could be further improved if we amortized the GPU utilization across batches.

C.5 Choice of classical causal discovery algorithm

We selected FCI Spirtes et al. (1995) as the underlying discovery algorithm in the observational setting over GES Chickering (2002) and GRaSP Lam et al. (2022) due to its superior downstream performance. We hypothesize this may be due to its richer output (ancestral graph) providing more signal to the Transformer model. We also tried Causal Additive Models Bühlmann et al. (2014), but its runtime was too slow for consistent GPU utilization. Observational algorithm implementations were provided by the causal-learn library Zheng et al. (2023). The code for running these alternative classical algorithms is available in our codebase.

We selected GIES as the underlying discovery algorithm in the interventional setting because a Python implementation was readily available at `https://github.com/juangamella/gies`.

We tried incorporating implementations from the Causal Discovery Toolbox via a Docker image Kalainathan et al. (2020), but there was excessive overhead associated with calling an R subroutine and reading/writing the inputs/results from disk.

Finally, we considered other independence tests for richer characterization, such as kernel-based methods. However, due to speed, we chose to remain with the default Fisherz conditional independence test for FCI, and BIC for GIES Schwarz (1978).

C.6 Sampling procedure

Selection scores: We consider three strategies for computing selection scores α. We include an empirical comparison of these strategies in Table 5.

1. Random selection: α is an $N \times N$ matrix of ones.
2. Global-statistic-based selection: $\alpha = \rho$.
3. Uncertainty-based selection: $\alpha = \hat{H}(E_t)$, where H denotes the information entropy

$$\alpha_{i,j} = -\sum_{e \in \{0,1,2\}} p(e) \log p(e). \tag{38}$$

Let $c^t_{i,j}$ be the number of times edge (i, j) was selected in $S_1 \dots S_{t-1}$, and let $\alpha^t = \alpha / \sqrt{c^t_{i,j}}$. We consider two strategies for selecting S_t based on α_t.

Greedy selection: Throughout our experiments, we used a greedy algorithm for subset selection. We normalize probabilities to 1 before the constructing each Categorical. Initialize

$$S_t \leftarrow \{i : i \sim \text{Categorical}(\alpha^t_1 \dots \alpha^t_N)\}. \tag{39}$$

where $\alpha_i^t = \sum_{j \neq i \in V} \alpha_{i,j}^t$. While $|S_t| < k$, update

$$S_t \leftarrow S_t \cup \{j : j \sim \text{Categorical}(\alpha_{1,S_t}^t \dots \alpha_{N,S_t}^t)) \tag{40}$$

where

$$\alpha_{j,S_t} = \begin{cases} \sum_{i \in S_t} \alpha_{i,j}^t & j \notin S_t \\ 0 & \text{otherwise.} \end{cases} \tag{41}$$

Subset selection: We also considered the following subset-level selection procedure, and observed minor performance gain for significantly longer runtime (linear program takes around 1 second per batch). Therefore, we opted for the greedy method instead.

We solve the following integer linear program to select a subset S_t of size k that maximizes $\sum_{i \in S_t} \alpha_{i,j}^t$. Let $\nu_i \in \{0, 1\}$ denote the selection of node i, and let $\epsilon_{i,j} \in \{0, 1\}$ denote the selection of edge (i, j). Our objective is to

$$\begin{array}{lll} \text{maximize} & \sum_{i,j} a_{i,j}^t \cdot \epsilon_{i,j} & \\ \text{subject to} & \sum_i \nu_i = k & \text{subset size} \\ & \epsilon_{i,j} \geq \nu_i + \nu_j - 1 & \text{node-edge consistency} \\ & \epsilon_{i,j} \leq \nu_i & \\ & \epsilon_{i,j} \leq \nu_j, & \\ & \nu_i \in \{0, 1\} & \\ & \epsilon_{i,j} \in \{0, 1\} & \end{array}$$

for $i, j \in V \times V$, $i \in V$. S_t is the set of non-zero indices in ν.

The final algorithm used the greedy selection strategy, with the first half of batches sampled according to global statistics, and the latter half sampled randomly, with visit counts shared. This strategy was selected heuristically, and we did not observe significant improvements or drops in performance when switching to other strategies (e.g. all greedy statistics-based, greedy uncertainty-based, linear program uncertainty-based, etc.)

D Empirical analyses

D.1 Additional results on synthetic data

For completeness, we include additional results and analysis.

SEA significantly outperforms the baselines across a variety of graph sizes and causal mechanisms in Table 3, and we maintain high performance even for causal mechanisms beyond the training set (Sigmoid, Polynomial). Our pretrained aggregator consistently improves upon the performance of its individual inputs (INVCOV, FCI-AVG, GIES-AVG), demonstrating the value in learning such a model. In terms of edge orientation accuracy, SEA outperforms the baselines in all settings (Table 4). We have omitted INVCOV from this comparison since it does not orient edges.

Table 5 compares the heuristics-based greedy sampler (inverse covariance + random) with the model uncertainty-based greedy sampler. Runtimes are plotted in Figure 6. The latter was run on CPU only, since it was non-trivial to access the GPU within a PyTorch data loader. We ran a forward pass to obtain an updated selection score every 10 batches, so this accrued over 10 times the number of forward passes, all on CPU. With proper engineering, this model-based sampler is expected to be much more efficient than reported. Still, it is faster than nearly all baselines.

Table 6 and Figure 5 show that the current implementations of SEA can generalize to graphs up to $4\times$ larger than those seen during training. With respect to larger graphs, there are two minor issues with the current implementation. We set an insufficient maximum subset positional embedding size of 500, and we did not sample random starting subset indices to ensure that higher-order embeddings are updated equally. We anticipate that increasing the limit on the number of subsets and ensuring that all embeddings are sufficiently learned will improve the generalization capacity on larger graphs. Nonetheless, our current model already obtains reasonable performance on larger graphs, out of the box.

Table 3: Causal discovery results on synthetic datasets. Each setting encompasses 5 distinct Erdős-Rényi graphs. The symbol † indicates that SEA was not pretrained on this setting. Runtimes are plotted in Figure 3. Details regarding baselines can be found in C.2.

N	E	Model	Linear		NN add.		NN non-add.		Sigmoid†		Polynomial†	
			mAP ↑	SHD ↓	mAP ↑	SHD ↓	mAP ↑	SHD ↓	mAP ↑	SHD ↓	mAP ↑	SHD ↓
10	10	DCDI-G	0.74	2.8	0.79	**2.2**	0.89	**1.0**	0.46	5.8	0.41	8.9
		DCDI-DSF	0.82	2.0	0.57	3.0	0.50	4.2	0.38	6.3	0.29	11.2
		DCD-FG	0.45	20.4	0.41	21.2	0.59	19.2	0.40	19.8	0.50	18.5
		DIFFAN	0.25	14.0	0.32	13.6	0.12	21.8	0.24	12.0	0.20	15.0
		DECI	0.18	19.4	0.16	13.8	0.23	16.2	0.29	13.9	0.46	7.8
		INVCOV	0.49	11.0	0.45	11.4	0.36	13.6	0.44	11.4	0.45	10.9
		FCI-AVG	0.52	10.0	0.38	8.2	0.40	9.8	0.56	9.1	0.41	10.0
		GIES-AVG	0.81	3.6	0.61	6.0	0.71	4.8	0.70	5.9	0.61	7.1
		SEA (F)	0.97	1.6	**0.95**	2.4	**0.92**	2.8	0.83	**3.7**	0.69	6.7
		SEA (G)	**0.99**	**1.2**	0.94	2.6	0.91	3.2	**0.85**	4.0	**0.70**	**5.8**
20	80	DCDI-G	0.46	44.0	0.41	61.6	**0.82**	37.4	0.48	44.2	0.37	59.7
		DCDI-DSF	0.48	41.2	0.44	**60.0**	0.74	**28.4**	0.48	43.6	0.38	**57.6**
		DCD-FG	0.32	171.8	0.33	156.0	0.41	162.2	0.47	80.1	0.49	79.8
		DIFFAN	0.21	127.2	0.19	153.6	0.18	144.6	0.22	116.8	0.18	157.1
		DECI	0.25	87.2	0.29	104.4	0.26	79.6	0.31	71.0	0.43	58.9
		INVCOV	0.35	94.2	0.27	107.8	0.30	100.2	0.34	91.7	0.32	94.4
		FCI-AVG	0.30	75.8	0.31	80.2	0.30	74.4	0.41	72.3	0.34	76.6
		GIES-AVG	0.41	70.0	0.44	75.2	0.46	67.4	0.50	65.6	0.49	68.1
		SEA (F)	0.86	29.6	0.55	73.6	0.72	51.8	**0.77**	**42.8**	0.61	61.8
		SEA (G)	**0.89**	**26.8**	**0.58**	71.4	0.73	50.6	0.76	45.0	**0.65**	60.1
100	400	DCD-FG	0.05	3068	0.07	3428	0.10	3510	0.13	3601	0.12	3316
		INVCOV	0.25	557.0	0.09	667.8	0.14	639.0	0.27	514.7	0.20	539.4
		SEA (F)	0.90	122.0	**0.28**	**361.2**	0.60	273.2	**0.69**	226.9	**0.38**	**327.0**
		SEA (G)	**0.91**	**116.6**	0.27	364.4	**0.61**	**266.8**	**0.69**	**218.3**	**0.38**	328.0

Due to the scope of this project and computing resources, we did not train very "big" models in the modern sense. There is much space to scale, both in terms of model architecture and the datasets covered. Table 7 probes the generalization limits of the two implementations of SEA in this paper.

We identified that our models, trained primarily on additive noise, achieve reasonable performance, but do not generalize reliably to causal mechanisms with multiplicative noise. For example, we tested additional datasets with the following mechanisms (same format as C.1).

- Sigmoid mix: $y = \sum_{i=1}^{d} W_i \cdot \text{sigmoid}(X_i) \times E$
- Polynomial mix: $y = (W_0 + XW_1 + X^2W_2) \times E$

We anticipate that incorporating these data into the training set would alleviate some of this gap (just as training on synthetic mRNA data enabled us to perform well there, despite its non-standard data distributions). However, we did not have time to test this hypothesis empirically.

DCDI learns a new generative model over each dataset, and its more powerful, deep sigmoidal flow variant seems to perform well in some (but not all) of these harder cases.

Tables 8 and 9 report the full results on $N = 100$ graphs.

Tables 10 and 11 report our results on scale-free graphs.

D.2 ADDITIONAL REALISTIC EXPERIMENTS

We report results on the real Sachs protein dataset Sachs et al. (2005) in Table 12. The relative performance of each model differs based on metric. SEA performs comparably, while maintaining fast inference speeds. However, despite the popularity of this dataset in causal discovery literature

Table 4: Synthetic experiments, edge direction accuracy (higher is better). All standard deviations were within 0.2. The symbol † indicates that SEA was not pretrained on this setting.

N	E	Model	Linear	NN add	NN	Sig.†	Poly.†
10	10	DCDI-G	0.74	0.80	0.85	0.41	0.44
		DCDI-DSF	0.79	0.62	0.68	0.38	0.39
		DCD-FG	0.50	0.47	0.70	0.43	0.54
		DIFFAN	0.61	0.55	0.26	0.53	0.47
		DECI	0.50	0.43	0.62	0.63	0.75
		FCI-AVG	0.52	0.43	0.41	0.55	0.40
		GIES-AVG	0.76	0.49	0.69	0.67	0.63
		SEA (FCI)	0.92	**0.92**	**0.94**	0.76	0.71
		SEA (GIES)	**0.94**	0.88	0.93	**0.84**	**0.79**
20	80	DCDI-G	0.47	0.43	0.82	0.40	0.24
		DCDI-DSF	0.50	0.49	0.78	0.41	0.28
		DCD-FG	0.58	0.65	0.75	0.62	0.48
		DIFFAN	0.46	0.28	0.36	0.45	0.21
		DECI	0.30	0.47	0.35	0.48	0.57
		FCI-AVG	0.19	0.19	0.22	0.33	0.23
		GIES-AVG	0.56	0.73	0.59	0.62	0.61
		SEA (FCI)	**0.93**	**0.90**	**0.93**	**0.85**	**0.89**
		SEA (GIES)	0.92	0.88	0.92	0.84	**0.89**
100	400	DCD-FG	0.46	0.60	0.70	0.67	0.53
		SEA (FCI)	0.93	0.90	0.91	**0.87**	0.82
		SEA (GIES)	**0.94**	**0.91**	**0.92**	**0.87**	**0.84**

Table 5: Comparison between heuristics-based sampler (random and inverse covariance) vs. model confidence-based sampler. Details regarding the samplers can be found in C.6. The suffix -L indicates the greedy confidence-based sampler. Each setting encompasses 5 distinct Erdős-Rényi graphs. The symbol † indicates that SEA was not pretrained on this setting. Bold indicates best of all models considered (including baselines not pictured).

N	E	Model	Linear			NN add.			NN non-add.			Sigmoid†			Polynomial†		
			mAP	EA	SHD	mAP	EA	SHD	mAP	EA	SHD	mAP	EA	SHD	mAP	EA	SHD
10	10	SEA(F)	0.97	0.92	1.6	**0.95**	**0.92**	2.4	**0.92**	0.94	2.8	0.83	0.76	3.7	0.69	0.71	6.7
		SEA(G)	**0.99**	**0.94**	1.2	0.94	0.88	2.6	0.91	0.93	3.2	0.85	0.84	4.0	0.70	0.79	**5.8**
		SEA(F)-L	0.97	0.93	**1.0**	0.95	0.87	2.4	**0.92**	**0.98**	3.4	0.84	0.77	3.9	**0.70**	0.79	**5.8**
		SEA(G)-L	0.98	0.93	1.4	0.94	0.91	2.8	0.91	0.94	4.0	**0.88**	**0.84**	**3.6**	0.70	**0.80**	**5.8**
10	40	SEA(F)	0.90	0.87	14.4	0.91	0.94	11.2	0.87	0.86	16.0	**0.81**	0.85	22.7	0.81	0.92	33.4
		SEA(G)	**0.94**	**0.91**	**12.8**	0.91	**0.95**	**10.4**	**0.89**	**0.89**	17.2	0.81	**0.87**	24.5	0.89	0.93	29.5
		SEA(F)-L	0.91	0.90	15.6	**0.91**	0.92	15.8	0.88	0.86	14.2	0.81	0.84	23.2	0.82	0.93	33.8
		SEA(G)-L	0.93	0.91	13.4	0.91	0.93	**10.4**	0.88	0.85	16.2	0.79	0.83	25.5	**0.90**	**0.94**	28.3
20	20	SEA(F)	0.97	0.92	3.2	0.94	0.97	3.2	0.84	0.93	7.2	0.84	0.85	7.6	**0.71**	**0.80**	10.2
		SEA(G)	0.97	0.89	3.0	**0.94**	0.95	3.4	0.83	0.94	7.8	0.84	0.83	8.1	0.69	0.78	10.1
		SEA(F)-L	0.97	**0.92**	2.8	0.93	0.95	3.8	**0.85**	0.94	6.8	**0.85**	0.85	**7.5**	0.67	0.78	**9.9**
		SEA(G)-L	**0.97**	0.90	**2.6**	**0.94**	**0.98**	3.4	0.83	**0.97**	7.0	0.84	0.84	7.9	0.67	0.79	10.6
20	80	SEA(F)	0.86	**0.93**	29.6	0.55	0.90	73.6	0.72	**0.93**	51.8	**0.77**	0.85	42.8	0.61	0.89	61.8
		SEA(G)	**0.89**	0.92	**26.8**	**0.58**	0.88	71.4	0.73	0.92	50.6	0.76	0.84	45.0	**0.65**	**0.89**	60.1
		SEA(F)-L	0.86	0.92	32.0	0.55	**0.90**	74.0	0.74	0.93	49.2	0.76	**0.87**	**41.8**	0.59	0.88	62.3
		SEA(G)-L	**0.89**	0.92	28.4	0.58	0.89	71.6	0.75	0.92	49.4	0.75	0.85	45.7	**0.65**	0.88	60.6

(due to lack of better alternatives), biological networks are known to be time-resolved and cyclic, so the validity of the ground truth "consensus" graph has been questioned by experts Mooij et al. (2020).

Table 6: Scaling to synthetic graphs, larger than those seen in training. Each setting encompasses 5 distinct Erdős-Rényi graphs. For all analysis in this table, we took $T = 500$ subsets of nodes, with $b = 500$ examples per batch. Here, the mean AUC values are artificially high due to the high negative rates, as actual edges scale linearly as N, while the number of possible edges scales quadratically.

N	Model	Linear, $E = N$				Linear, $E = 4N$			
		mAP ↑	AUC ↑	SHD ↓	EdgeAcc ↑	mAP ↑	AUC ↑	SHD ↓	EdgeAcc ↑
100	INVCOV	0.43	0.99	116.8	—	0.30	0.93	511.8	—
	CORR	0.42	0.99	113.0	—	0.19	0.80	579.4	—
	SEA (FCI)	0.97	1.00	11.6	0.93	0.88	0.98	129.0	0.94
	SEA (GIES)	0.97	1.00	12.8	0.91	0.91	0.99	104.6	0.95
200	INVCOV	0.45	1.00	218.4	—	0.33	0.96	999.6	—
	CORR	0.42	0.99	223.0	—	0.18	0.86	1183.5	—
	SEA (FCI)	0.91	1.00	49.9	0.87	0.82	0.97	327.4	0.92
	SEA (GIES)	0.95	1.00	35.4	0.91	0.86	0.98	271.9	0.92
300	INVCOV	0.46	1.00	308.3	—	0.35	0.98	1444.7	—
	CORR	0.42	1.00	326.2	—	0.20	0.89	1710.4	—
	SEA (FCI)	0.80	1.00	121.1	0.78	0.70	0.95	693.1	0.86
	SEA (GIES)	0.88	1.00	88.9	0.84	0.78	0.96	556.1	0.87
400	INVCOV	0.47	1.00	417.7	—	0.36	0.98	1882.7	—
	CORR	0.42	1.00	445.4	—	0.20	0.91	2269.3	—
	SEA (FCI)	0.49	0.93	313.9	0.61	0.56	0.90	1103.1	0.75
	SEA (GIES)	0.70	0.99	225.9	0.71	0.70	0.94	871.6	0.80
500	INVCOV	0.47	1.00	504.5	—	0.38	0.99	2299.8	—
	CORR	0.42	1.00	543.3	—	0.21	0.93	2789.5	—
	SEA (FCI)	0.27	0.90	757.6	0.51	0.29	0.86	1823.6	0.56
	SEA (GIES)	0.41	0.98	485.3	0.57	0.48	0.92	1653.5	0.67

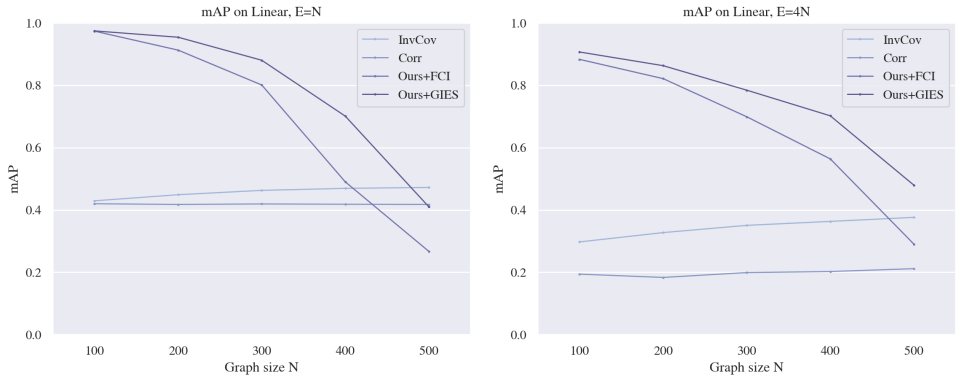

Figure 5: mAP on graphs larger than seen during training. Due to an insufficient maximum number of subset embeddings, we were only able to sample 500 batches, which appears to be too few for larger graphs. These values correspond to the numbers in Table 6.

D.2.1 SIMULATED MRNA DATA GENERATION

We generated mRNA data using the SERGIO simulator Dibaeinia & Sinha (2020). We sampled datasets with the Hill coefficient set to $\{0.25, 0.5, 1, 2, 4\}$ for training, and 2 for testing (2 was default). We set the decay rate to the default 0.8, and the noise parameter to the default of 1.0. We sampled 400 graphs for each of $N = \{10, 20\}$ and $E = \{N, 2N\}$.

Table 7: Generalization limits of our current implementations. Each setting represents 5 distinct graphs. Our models were not pretrained on multiplicative noise, so they do not generalize reliably to these cases. While much slower, DCDI variants learn the data distribution from scratch each time, so they seem to perform well in some of these cases. We anticipate that training on multiplicative data would alleviate this generalization gap, but we did not have time to test this empirically.

N	E	Model	Sigmoid mix†			Poly. mix†			Sigmoid mix (SF)†			Poly. mix (SF)†		
			mAP ↑	EA ↑	SHD ↓	mAP ↑	EA ↑	SHD ↓	mAP ↑	EA ↑	SHD ↓	mAP ↑	EA ↑	SHD ↓
10	10	DCDI-G	0.84	0.81	0.9	0.11	0.00	10.2	0.67	0.92	15.2	0.12	0.00	10.6
		DCDI-DSF	**0.96**	**0.91**	**0.3**	**0.39**	0.32	**7.3**	0.81	**0.98**	13.6	**0.39**	0.36	**9.8**
		DCD-FG	0.52	0.61	18.0	0.11	0.00	10.2	0.57	0.66	15.5	0.12	0.00	10.6
		DIFFAN	0.14	0.31	19.9	0.11	0.09	14.0	0.10	0.22	17.8	0.11	0.14	17.2
		DECI	0.17	0.43	19.1	0.11	0.07	14.1	0.20	0.53	15.0	0.12	0.06	13.6
		INVCOV	0.34	0.51	12.1	0.18	0.51	16.4	0.31	0.56	11.4	0.16	0.50	16.1
		FCI-AVG	0.56	0.54	9.1	0.11	0.00	10.2	0.53	0.52	8.1	0.12	0.01	10.6
		GIES-AVG	0.91	0.88	3.1	0.22	0.38	10.2	**0.93**	0.88	**2.3**	0.22	0.37	10.6
		SEA (FCI)	0.58	0.51	5.7	0.25	**0.53**	10.2	0.67	0.56	4.5	0.20	**0.46**	10.6
		SEA (GIES)	0.41	0.39	8.1	0.18	0.51	10.2	0.48	0.48	5.4	0.21	**0.46**	10.6
10	40	DCDI-G	0.58	0.35	25.0	0.44	0.00	39.8	0.76	0.83	22.0	0.31	0.00	28.2
		DCDI-DSF	**0.79**	0.46	**12.2**	0.48	0.09	35.6	**0.92**	**0.88**	**15.8**	0.35	0.07	**26.4**
		DCD-FG	0.60	0.40	26.8	0.44	0.00	39.8	0.52	0.46	22.1	0.31	0.00	28.2
		DIFFAN	0.38	0.32	31.8	0.40	0.32	**30.0**	0.26	0.30	35.0	0.28	0.29	29.3
		DECI	0.46	0.39	27.5	0.43	0.03	39.5	0.35	0.51	24.9	0.30	0.05	30.1
		INVCOV	0.48	0.51	38.7	0.48	0.54	40.3	0.40	0.51	32.5	0.34	0.48	38.2
		FCI-AVG	0.44	0.18	39.7	0.44	0.01	39.8	0.43	0.30	25.2	0.32	0.01	28.2
		GIES-AVG	0.43	0.37	38.2	0.50	0.48	39.8	0.48	0.45	22.8	0.36	0.39	28.2
		SEA (FCI)	0.56	0.58	28.6	**0.81**	0.85	39.7	0.62	0.67	17.3	0.58	**0.85**	28.2
		SEA (GIES)	0.58	**0.65**	30.2	0.80	**0.86**	39.5	0.64	0.69	16.7	**0.60**	0.81	28.2
20	20	DCDI-G	0.57	0.72	6.2	0.06	0.01	21.7	0.47	**0.98**	45.4	0.06	0.01	**20.7**
		DCDI-DSF	**0.91**	**0.94**	**1.1**	**0.21**	0.46	34.6	0.56	0.97	40.6	**0.29**	**0.80**	83.0
		DCD-FG	0.50	0.68	62.1	0.06	0.00	24.3	0.63	0.81	256.6	0.06	0.11	186.4
		DIFFAN	0.08	0.26	47.5	0.06	0.15	51.5	0.08	0.30	42.6	0.06	0.22	53.7
		DECI	0.17	0.58	38.0	0.06	0.05	36.0	0.18	0.60	32.9	0.06	0.03	38.9
		INVCOV	0.31	0.59	22.3	0.09	0.52	35.9	0.24	0.44	24.5	0.07	0.50	35.4
		FCI-AVG	0.64	0.69	17.4	0.06	0.01	**21.3**	0.58	0.66	15.3	0.06	0.00	21.0
		GIES-AVG	0.82	0.75	8.5	0.15	0.44	**21.3**	**0.83**	0.78	**7.8**	0.12	0.42	21.0
		SEA (FCI)	0.61	0.60	8.5	0.12	**0.55**	**21.3**	0.63	0.56	9.0	0.11	0.55	21.1
		SEA (GIES)	0.41	0.48	13.0	0.11	0.55	21.6	0.50	0.49	12.3	0.11	0.52	21.3
20	80	DCDI-G	0.60	0.59	32.3	0.21	0.05	86.4	0.54	0.84	78.6	0.18	0.14	74.8
		DCDI-DSF	**0.89**	**0.81**	**8.4**	0.24	0.25	102.3	**0.65**	**0.89**	60.1	0.27	0.60	201.6
		DCD-FG	0.46	0.72	222.2	0.21	0.00	81.8	0.52	0.76	202.2	0.18	0.15	225.9
		DIFFAN	0.18	0.30	151.1	0.19	0.31	130.8	0.15	0.30	137.0	0.16	0.31	127.9
		DECI	0.31	0.43	70.5	0.20	0.03	89.0	0.25	0.41	66.5	0.17	0.02	79.0
		INVCOV	0.30	0.51	98.1	0.22	0.51	115.4	0.27	0.53	89.1	0.19	0.49	108.1
		FCI-AVG	0.37	0.30	76.0	0.21	0.01	79.3	0.38	0.34	59.6	0.18	0.01	**66.5**
		GIES-AVG	0.54	0.64	68.4	0.23	0.37	79.3	0.55	0.65	50.7	0.19	0.36	**66.5**
		SEA (FCI)	0.61	0.64	52.6	0.41	**0.87**	79.3	0.62	0.70	**36.7**	0.34	0.82	66.7
		SEA (GIES)	0.53	0.64	58.1	**0.43**	0.86	**79.0**	0.59	0.74	41.4	**0.35**	**0.82**	66.8

These data distributions are quite different from typical synthetic datasets, as they simulate steady-state measurements and the data are lower bounded at 0 (gene counts).

Table 8: Causal discovery results on synthetic datasets with 100 nodes, continuous metrics. Each setting encompasses 5 distinct Erdős-Rényi graphs. The symbol † indicates that the model was not trained on this setting. All standard deviations were within 0.1.

N	E	Model	Linear		NN add.		NN non-add.		Sigmoid†		Polynomial†	
			mAP ↑	AUC ↑	mAP ↑	AUC ↑	mAP ↑	AUC ↑	mAP ↑	AUC ↑	mAP ↑	AUC ↑
100	100	DCD-FG	0.11	0.75	0.12	0.71	0.18	0.73	0.20	0.72	0.06	0.60
		INVCOV	0.40	0.99	0.22	0.94	0.16	0.87	0.40	0.97	0.36	0.90
		SEA (FCI)	0.96	1.00	0.83	0.97	0.75	0.97	0.79	0.97	0.56	0.88
		SEA (GIES)	0.97	1.00	0.82	0.98	0.74	0.96	0.80	0.97	0.54	0.85
100	400	DCD-FG	0.05	0.59	0.07	0.64	0.10	0.72	0.13	0.72	0.12	0.64
		INVCOV	0.25	0.91	0.09	0.62	0.14	0.77	0.27	0.86	0.20	0.67
		SEA (FCI)	0.90	0.99	0.28	0.82	0.60	0.92	0.69	0.92	0.38	0.80
		SEA (GIES)	0.91	0.99	0.27	0.82	0.61	0.92	0.69	0.91	0.38	0.78

Table 9: Causal discovery results on synthetic datasets with 100 nodes, discrete metrics. Each setting encompasses 5 distinct Erdős-Rényi graphs. The symbol † indicates that the model was not trained on this setting.

N	E	Model	Linear		NN add.		NN non-add.		Sigmoid†		Polynomial†	
			EA ↑	SHD ↓	EA ↑	SHD ↓	EA ↑	SHD ↓	EA ↑	SHD ↓	EA ↑	SHD ↓
100	100	DCD-FG	0.63	3075.8	0.58	2965.0	0.60	2544.4	0.59	3808.0	0.34	1927.9
		INVCOV	—	124.4	—	130.0	—	158.8	—	112.3	—	106.3
		SEA (FCI)	0.91	13.4	0.90	34.4	0.91	47.2	0.78	40.3	0.69	59.2
		SEA (GIES)	0.91	13.6	0.93	32.8	0.91	45.8	0.78	38.6	0.68	60.3
100	400	DCD-FG	0.46	3068.2	0.60	3428.8	0.70	3510.8	0.67	3601.8	0.53	3316.7
		INVCOV	—	557.0	—	667.8	—	639.0	—	514.7	—	539.4
		SEA (FCI)	0.93	122.0	0.90	361.2	0.91	273.2	0.87	226.9	0.82	327.0
		SEA (GIES)	0.94	116.6	0.91	364.4	0.92	266.8	0.87	218.3	0.84	328.0

D.3 TRADITIONAL ALGORITHM PARAMETERS

We investigated model performance with respect to the settings of our graph estimation parameters. Our model is sensitive to the size of batches used to estimate global features and marginal graphs (Figure 7). In particular, at least 250 points are required per batch for an acceptable level of performance. Our model is not particularly sensitive to the number of batches sampled (Figure 8), or to the number of variables sampled in each subset (Figure 9).

D.4 CHOICE OF GLOBAL STATISTIC

We selected inverse covariance as our global feature due to its ease of computation and its relationship to partial correlation. For context, we also provide the performance analysis of several alternatives. Tables 13 and 14 compare the results of different graph-level statistics on our synthetic datasets. Discretization thresholds for SHD were obtained by computing the p^{th} quantile of the computed values, where $p = 1 - (E/N)$. This is not entirely fair, as no other baseline receives the same calibration, but these ablation studies only seek to compare state-of-the-art causal discovery methods with the "best" possible (oracle) statistical alternatives.

Table 10: Causal discovery results on synthetic scale-free datasets, continuous metrics. Each setting encompasses 5 distinct scale-free graphs. The symbol † indicates that SEA was not pretrained on this setting. Details regarding baselines can be found in C.2.

N	E	Model	Linear		NN add.		NN non-add.		Sigmoid†		Polynomial†	
			mAP ↑	AUC ↑	mAP ↑	AUC ↑	mAP ↑	AUC ↑	mAP ↑	AUC ↑	mAP ↑	AUC ↑
10	10	DCDI-G	0.54	0.90	0.59	0.88	0.69	0.89	0.48	0.77	0.50	0.73
		DCDI-DSF	0.70	0.92	0.71	0.88	0.36	0.83	0.46	0.75	0.49	0.76
		DCD-FG	0.56	0.76	0.47	0.72	0.50	0.73	0.44	0.68	0.57	0.75
		DIFFAN	0.25	0.73	0.15	0.66	0.16	0.62	0.31	0.75	0.24	0.63
		DECI	0.17	0.65	0.17	0.67	0.20	0.72	0.27	0.73	0.49	0.82
		INVCOV	0.38	0.57	0.26	0.50	0.29	0.52	0.27	0.46	0.08	0.13
		FCI-AVG	0.56	0.80	0.51	0.80	0.43	0.74	0.60	0.82	0.34	0.68
		GIES-AVG	0.87	0.98	0.61	0.94	0.69	0.94	0.75	0.96	0.71	0.91
		SEA (FCI)	0.94	**0.99**	0.93	**0.98**	**0.93**	0.98	0.81	0.97	**0.76**	**0.91**
		SEA (GIES)	**0.95**	**0.99**	**0.94**	**0.98**	0.92	**0.98**	**0.85**	**0.98**	0.74	0.90
10	40	DCDI-G	0.70	0.85	0.74	0.85	**0.88**	**0.91**	0.56	0.66	0.53	0.64
		DCDI-DSF	0.74	0.87	0.73	0.84	0.71	0.90	0.56	0.69	0.51	0.63
		DCD-FG	0.37	0.58	0.45	0.61	0.45	0.58	0.49	0.63	0.63	0.73
		DIFFAN	0.29	0.50	0.25	0.38	0.28	0.46	0.31	0.53	0.27	0.44
		DECI	0.30	0.51	0.41	0.65	0.33	0.51	0.38	0.60	0.59	0.77
		INVCOV	0.36	0.48	0.34	0.49	0.37	0.48	0.39	0.54	0.26	0.34
		FCI-AVG	0.47	0.64	0.41	0.60	0.40	0.58	0.48	0.64	0.41	0.59
		GIES-AVG	0.43	0.68	0.43	0.63	0.44	0.61	0.49	0.69	0.59	0.71
		SEA (FCI)	**0.93**	0.96	0.84	0.92	0.81	0.90	**0.81**	**0.89**	0.73	0.84
		SEA (GIES)	0.92	**0.96**	**0.84**	**0.93**	0.83	0.90	0.79	0.88	**0.78**	**0.87**
20	20	DCDI-G	0.41	0.95	0.50	0.94	0.69	0.96	0.37	0.83	0.37	0.77
		DCDI-DSF	0.48	0.95	0.55	0.93	0.33	0.90	0.37	0.79	0.35	0.82
		DCD-FG	0.51	0.87	0.39	0.83	0.48	0.84	0.56	0.84	0.50	0.84
		DIFFAN	0.27	0.80	0.11	0.65	0.11	0.66	0.26	0.77	0.12	0.69
		DECI	0.13	0.69	0.15	0.71	0.15	0.73	0.15	0.71	0.25	0.79
		INVCOV	0.30	0.57	0.26	0.53	0.21	0.53	0.24	0.49	0.03	0.10
		FCI-AVG	0.63	0.84	0.44	0.78	0.43	0.79	0.60	0.86	0.47	0.78
		GIES-AVG	0.82	0.99	0.58	0.95	0.57	0.96	0.75	0.98	0.61	0.90
		SEA (FCI)	**0.95**	**1.00**	0.91	**0.98**	0.87	**0.98**	**0.84**	**0.98**	0.70	**0.92**
		SEA (GIES)	0.93	1.00	**0.91**	0.98	**0.88**	0.98	0.82	0.98	**0.70**	0.91
20	80	DCDI-G	0.62	0.88	0.61	**0.89**	**0.76**	**0.94**	0.44	0.76	0.36	0.60
		DCDI-DSF	0.58	0.87	0.55	0.86	0.58	0.92	0.43	0.78	0.35	0.66
		DCD-FG	0.38	0.70	0.30	0.69	0.48	0.80	0.48	0.75	0.53	0.73
		DIFFAN	0.18	0.55	0.15	0.44	0.16	0.53	0.19	0.56	0.15	0.38
		DECI	0.21	0.58	0.24	0.64	0.26	0.66	0.30	0.68	0.41	0.75
		INVCOV	0.27	0.52	0.22	0.51	0.25	0.54	0.27	0.51	0.12	0.30
		FCI-AVG	0.31	0.63	0.30	0.62	0.30	0.62	0.41	0.68	0.32	0.62
		GIES-AVG	0.51	0.87	0.43	0.78	0.47	0.81	0.52	0.82	0.47	0.73
		SEA (FCI)	0.92	**0.98**	**0.64**	0.89	0.71	0.90	0.73	0.90	0.59	0.81
		SEA (GIES)	**0.92**	**0.98**	0.63	0.89	0.73	0.91	**0.77**	**0.92**	**0.62**	**0.84**

CORR refers to global correlation,

$$\rho_{i,j} = \frac{\mathbb{E}\left(X_i X_j\right) - \mathbb{E}\left(X_i\right)\mathbb{E}\left(X_j\right)}{\sqrt{\mathbb{E}\left(X_i^2\right) - \mathbb{E}\left(X_i\right)^2} \cdot \sqrt{\mathbb{E}\left(X_j^2\right) - \mathbb{E}\left(X_j\right)^2}}. \tag{42}$$

D-CORR refers to distance correlation, computed between all pairs of variables. Distance correlation captures both linear and non-linear dependencies, and D-CORR$(X_i, X_j) = 0$ if and only if $X_i \perp\!\!\!\perp X_j$. Please refer to Sz'ekely et al. (2007) for the full derivation. Despite its power to capture non-linear

Table 11: Causal discovery results on synthetic scale-free datasets, discrete metrics. Each setting encompasses 5 distinct scale-free graphs. The symbol † indicates that SEA was not pretrained on this setting. Details regarding baselines can be found in C.2.

N	E	Model	Linear		NN add.		NN non-add.		Sigmoid†		Polynomial†	
			EA ↑	SHD ↓	EA ↑	SHD ↓	EA ↑	SHD ↓	EA ↑	SHD ↓	EA ↑	SHD ↓
10	10	DCDI-G	0.85	16.6	0.82	17.4	0.81	16.2	0.71	16.9	0.70	16.6
		DCDI-DSF	**0.89**	16.2	0.75	15.4	0.78	16.8	0.80	18.1	0.77	18.0
		DCD-FG	0.60	16.4	0.57	22.2	0.57	20.0	0.47	17.9	0.59	16.8
		DIFFAN	0.55	9.2	0.49	14.6	0.36	11.6	0.59	7.7	0.42	14.8
		DECI	0.51	17.4	0.55	17.4	0.58	12.0	0.62	12.6	0.74	8.2
		INVCOV	0.60	9.8	0.54	12.0	0.54	10.8	0.40	11.9	0.14	17.4
		FCI-AVG	0.62	8.4	0.51	7.8	0.45	8.0	0.61	9.2	0.34	9.6
		GIES-AVG	0.83	2.2	0.60	6.0	0.75	4.2	0.72	4.9	0.73	5.7
		SEA (FCI)	0.87	**1.2**	0.93	2.0	**0.96**	2.2	0.70	4.3	0.82	5.3
		SEA (GIES)	0.85	1.4	**0.96**	**1.8**	0.94	**2.0**	**0.86**	**3.4**	**0.83**	**5.0**
10	40	DCDI-G	0.79	24.0	0.77	27.8	0.82	19.6	0.65	31.4	0.61	32.6
		DCDI-DSF	0.84	22.8	0.78	24.4	0.83	20.4	0.71	31.6	0.62	33.3
		DCD-FG	0.36	24.8	0.41	25.2	0.38	25.6	0.41	23.2	0.54	18.5
		DIFFAN	0.40	29.8	0.28	37.0	0.38	32.6	0.45	28.0	0.33	32.7
		DECI	0.43	27.8	0.66	22.6	0.48	28.6	0.52	22.2	0.66	**13.3**
		INVCOV	0.42	36.2	0.50	35.2	0.48	39.2	0.59	33.7	0.34	50.8
		FCI-AVG	0.33	23.8	0.28	27.2	0.25	27.6	0.36	26.1	0.24	27.3
		GIES-AVG	0.46	21.8	0.50	24.4	0.48	25.2	0.52	22.7	0.64	22.5
		SEA (FCI)	0.88	7.0	0.95	15.2	**0.91**	15.0	**0.87**	**13.9**	0.91	19.4
		SEA (GIES)	**0.88**	**6.6**	**0.98**	**14.0**	0.88	**14.4**	0.87	14.1	**0.93**	19.1
20	20	DCDI-G	0.95	40.4	0.92	44.8	**0.96**	39.8	**0.88**	41.1	0.79	38.4
		DCDI-DSF	**0.95**	40.4	0.92	42.4	0.90	42.2	0.84	41.1	**0.83**	49.3
		DCD-FG	0.68	252.2	0.77	182.8	0.78	181.2	0.70	251.3	0.69	278.2
		DIFFAN	0.67	23.6	0.40	42.2	0.42	34.0	0.59	22.6	0.50	46.8
		DECI	0.50	42.0	0.54	43.0	0.57	40.0	0.51	34.7	0.65	25.3
		INVCOV	0.54	20.6	0.54	24.8	0.52	26.6	0.50	23.9	0.13	38.2
		FCI-AVG	0.67	13.8	0.52	17.4	0.53	17.8	0.65	16.7	0.50	18.9
		GIES-AVG	0.82	6.4	0.71	12.6	0.68	13.2	0.75	11.4	0.73	13.4
		SEA (FCI)	0.91	**2.6**	**0.96**	4.6	0.93	6.4	0.82	**6.7**	0.76	10.3
		SEA (GIES)	0.85	4.0	0.95	**3.6**	0.93	**6.2**	0.80	7.9	0.82	**9.9**
20	80	DCDI-G	0.83	93.0	0.89	104.2	**0.91**	67.8	0.78	82.5	0.60	79.7
		DCDI-DSF	0.84	103.2	0.85	94.8	0.89	63.8	0.78	84.4	0.64	82.6
		DCD-FG	0.63	188.2	0.70	187.2	0.78	190.2	0.71	217.3	0.71	234.7
		DIFFAN	0.42	110.6	0.30	144.6	0.40	118.8	0.41	99.7	0.21	149.4
		DECI	0.33	72.2	0.48	81.4	0.48	67.0	0.50	60.4	0.58	**47.2**
		INVCOV	0.50	85.4	0.48	94.2	0.53	86.6	0.51	87.2	0.33	116.8
		FCI-AVG	0.26	58.2	0.22	58.4	0.26	55.0	0.37	59.3	0.23	62.9
		GIES-AVG	0.65	48.4	0.71	53.6	0.68	48.0	0.64	53.6	0.61	54.9
		SEA (FCI)	0.92	19.0	**0.93**	**48.4**	0.88	38.4	0.85	37.1	0.86	49.8
		SEA (GIES)	**0.92**	**17.6**	**0.93**	49.2	0.89	**37.2**	**0.88**	**35.1**	**0.89**	48.1

dependencies, we opted not to use D-CORR because it is quite slow to compute between all pairs of variables.

INVCOV refers to inverse covariance, computed globally,

$$\rho = \mathbb{E}\left((X - \mathbb{E}(X))(X - \mathbb{E}(X))^T\right)^{-1}. \tag{43}$$

For graphs $N < 100$, inverse covariance was computed directly using NumPy. For graphs $N \geq 100$, inverse covariance was computed using Ledoit-Wolf shrinkage at inference time Ledoit & Wolf (2004).

Table 12: Causal discovery on (real) Sachs protein expression dataset. While INVCOV is a strong baseline for connectivity, it does not predict orientation. The relative performance of the methods depends on metric.

Model	mAP ↑	AUC ↑	SHD ↓	EdgeAcc ↑	Time (s) ↓
DCDI-G	0.17	0.55	21.0	0.20	2436.5
DCDI-DSF	0.20	0.59	20.0	0.20	1979.6
DCD-FG	**0.32**	0.59	27.0	0.35	951.4
DIFFAN	0.14	0.45	37.0	0.41	293.7
DECI	0.21	**0.62**	28.0	**0.53**	609.7
INVCOV	0.31	0.61	20.0	—	**0.002**
FCI-AVG	0.27	0.59	18.0	0.24	41.9
GIES-AVG	0.21	0.59	17.0	0.24	77.9
SEA (FCI)	0.23	0.54	24.0	0.47	3.2
SEA (GIES)	0.23	0.60	**14.0**	0.41	2.9

Table 13: Comparison of global statistics (continuous metrics). Each setting encompasses 5 distinct Erdős-Rényi graphs.

N	E	Model	Linear		NN add.		NN non-add.		Sigmoid		Polynomial	
			mAP ↑	AUC ↑	mAP ↑	AUC ↑	mAP ↑	AUC ↑	mAP ↑	AUC ↑	mAP ↑	AUC ↑
10	10	CORR	0.45	0.87	0.41	0.86	0.41	0.85	0.46	0.86	0.45	0.85
		D-CORR	0.42	0.86	0.41	0.87	0.40	0.87	0.43	0.86	0.45	0.89
		INVCOV	0.49	0.87	0.45	0.86	0.36	0.81	0.44	0.86	0.45	0.83
10	40	CORR	0.47	0.53	0.47	0.52	0.46	0.52	0.48	0.53	0.48	0.54
		D-CORR	0.46	0.53	0.46	0.51	0.46	0.54	0.48	0.53	0.47	0.54
		INVCOV	0.50	0.57	0.48	0.52	0.47	0.53	0.47	0.50	0.48	0.52
100	100	CORR	0.42	0.99	0.25	0.94	0.25	0.93	0.42	0.98	0.35	0.91
		D-CORR	0.41	0.99	0.25	0.96	0.26	0.96	0.41	0.98	0.37	0.94
		INVCOV	0.40	0.99	0.22	0.94	0.16	0.87	0.40	0.97	0.36	0.90
100	400	CORR	0.19	0.80	0.10	0.63	0.14	0.72	0.27	0.84	0.20	0.72
		D-CORR	0.19	0.80	0.10	0.63	0.14	0.75	0.26	0.84	0.21	0.74
		INVCOV	0.25	0.91	0.09	0.62	0.14	0.77	0.27	0.86	0.20	0.67

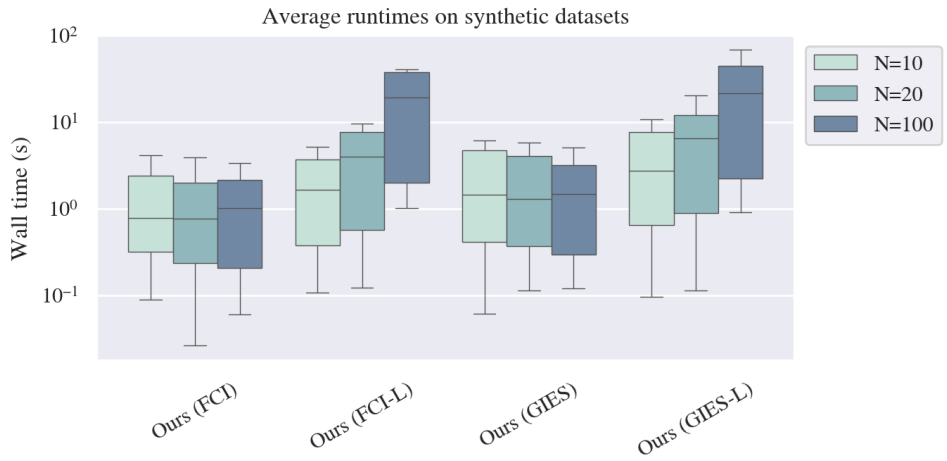

Figure 6: Runtime for heuristics-based greedy sampler vs. model uncertainty-based greedy sampler (suffix -L). For sampling, the model was run on CPU only, due to the difficulty of invoking GPU in the PyTorch data sampler.

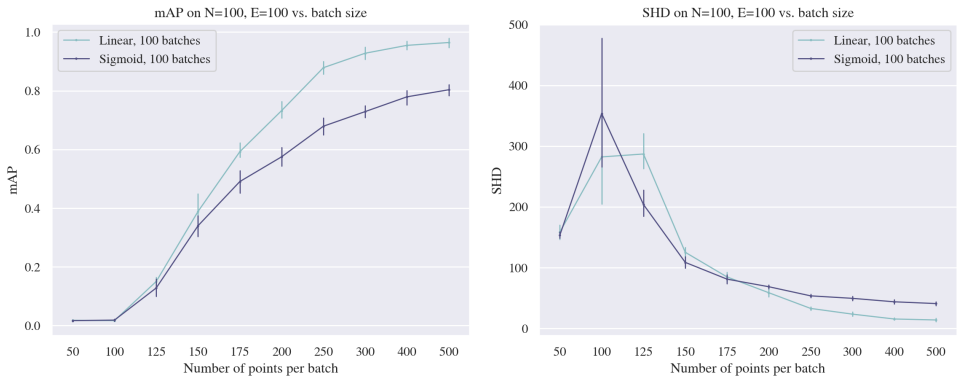

Figure 7: Performance of our model (GIES) as a function of traditional algorithm batch size. Error bars indicate 95% confidence interval across the 10 datasets of each setting. The global feature and marginal graph estimates are sensitive to batch size and require at least 250 points per batch to achieve an acceptable level of performance.

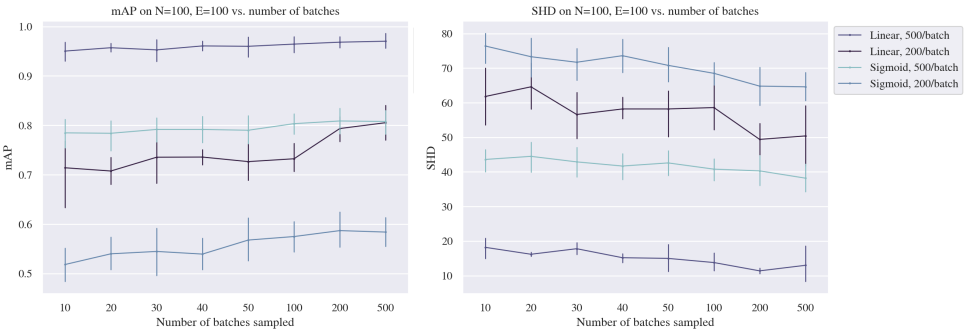

Figure 8: Performance of our model (GIES) as a function of number of batches sampled. Error bars indicate 95% confidence interval across the 10 datasets of each setting. Our model is relatively insensitive to the number of batches sampled, though more batches are beneficial in harder cases, e.g. sigmoid mechanism with additive noise or smaller batch size.

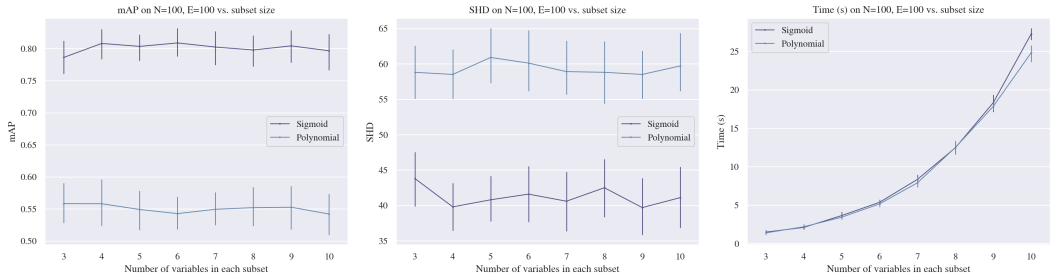

Figure 9: Performance of our model (GIES) as a function of subset size $|S|$ (number of variables sampled). Error bars indicate 95% confidence interval across the 10 datasets of each setting. Our model was trained on $|S| = 5$, but it is insensitive to the number of variables sampled per subset at inference. Runtime scales exponentially upwards.

Table 14: Comparison of global statistics (SHD). Discretization thresholds for SHD were obtained by computing the p^{th} quantile of the computed values, where $p = 1 - (E/N)$. Each setting encompasses 5 distinct Erdős-Rényi graphs.

N	E	Model	Linear	NN add.	NN non-add.	Sigmoid	Polynomial
10	10	CORR	$10.6_{\pm 2.8}$	$10.2_{\pm 4.6}$	$12.0_{\pm 1.9}$	$11.1_{\pm 4.3}$	$9.9_{\pm 2.8}$
		D-CORR	$10.4_{\pm 2.6}$	$9.8_{\pm 4.7}$	$12.2_{\pm 2.6}$	$10.8_{\pm 3.3}$	$10.2_{\pm 3.2}$
		INVCOV	$11.0_{\pm 2.8}$	$11.4_{\pm 5.5}$	$13.6_{\pm 2.9}$	$11.4_{\pm 4.1}$	$10.9_{\pm 3.5}$
10	40	CORR	$39.2_{\pm 2.4}$	$38.0_{\pm 1.8}$	$38.2_{\pm 0.7}$	$38.8_{\pm 3.3}$	$38.2_{\pm 2.0}$
		D-CORR	$38.8_{\pm 2.0}$	$38.8_{\pm 1.5}$	$37.0_{\pm 0.6}$	$38.9_{\pm 3.2}$	$38.0_{\pm 2.0}$
		INVCOV	$35.8_{\pm 2.3}$	$39.2_{\pm 1.5}$	$37.6_{\pm 2.7}$	$40.7_{\pm 2.2}$	$38.4_{\pm 1.2}$
100	100	CORR	$113.0_{\pm 4.9}$	$132.2_{\pm 18.0}$	$144.6_{\pm 5.2}$	$106.5_{\pm 11.5}$	$110.3_{\pm 6.1}$
		D-CORR	$113.8_{\pm 5.3}$	$133.2_{\pm 17.9}$	$144.2_{\pm 6.7}$	$108.5_{\pm 11.9}$	$109.5_{\pm 5.7}$
		INVCOV	$124.4_{\pm 8.1}$	$130.0_{\pm 17.2}$	$158.8_{\pm 6.2}$	$112.3_{\pm 14.8}$	$106.3_{\pm 4.6}$
100	400	CORR	$580.4_{\pm 24.5}$	$666.0_{\pm 13.5}$	$626.2_{\pm 23.4}$	$516.5_{\pm 18.5}$	$562.5_{\pm 20.1}$
		D-CORR	$578.2_{\pm 24.7}$	$665.4_{\pm 15.4}$	$626.6_{\pm 21.9}$	$522.3_{\pm 17.6}$	$557.2_{\pm 20.4}$
		INVCOV	$557.0_{\pm 11.7}$	$667.8_{\pm 15.4}$	$639.0_{\pm 9.7}$	$514.7_{\pm 23.1}$	$539.4_{\pm 18.4}$