THE UNIVERSITY OF CHICAGO

# CAUSAL DISCOVERY AND OPTIMAL EXPERIMENTAL DESIGN FOR GENOME-SCALE BIOLOGICAL NETWORK RECOVERY

A DISSERTATION SUBMITTED TO
THE FACULTY OF THE DIVISION OF THE PHYSICAL SCIENCES DIVISION
IN CANDIDACY FOR THE DEGREE OF
MASTERS

DEPARTMENT OF COMPUTER SCIENCE

BY
ASHKA SHAH

CHICAGO, ILLINOIS
03/06/2023

# TABLE OF CONTENTS

# ABSTRACT

Causal discovery of genome-scale networks is important for identifying pathways from genes to phenotypes –e.g. cell function, disease, drug resistance and others. Causal learners based on graphical models rely on interventional samples to orient edges in the network. However, these models have not been shown to scale up the size of the genome, which are on the order of $1e3$-$1e4$ genes. We introduce a new learner, SP-GIES, that jointly learns from interventional and observational datasets and achieves almost 4x speedup against an existing learner for 1,000 node networks. SP-GIES achieves an AUC-PR score of 0.91 on 1,000 node networks, and scales up to 2,000 node networks – this is 4x larger than existing works. We also show how SP-GIES improves downstream optimal experimental design strategies for selecting interventional experiments to perform on the system. This is an important step forward in realizing causal discovery at scale via autonomous experimental design.

# CHAPTER 1

# INTRODUCTION

A biological network describes causal interactions between variables in a biological system. These variables can be genes, transcription factors, proteins and metabolites. Interactions between these variables, along with external environmental factors, maps the genome of a species (or genotype) to an observable trait (or phenotype) (See Fig. 1.1). Predicting phenotypes from genotypes is one of the core challenges in systems biology (Pigliucci [2010], Ritchie et al. [2015], Lewis et al. [2012]). Genotype-phenotype models are useful for predicting cell function, disease, drug resistance, and many other problems related to biology and public health. In this paper we focus on biological network recovery of the first layer in the interaction hierarchy (the gene regulatory network) – which we name a "genome-scale network" since the number of variables corresponds to the number of genes in a genome. Recovery of genome-scale networks is important for understanding drivers for phenotypic changes and for identifying new drug targets. Moreover, a better understanding of genome-scale networks enables more accurate downstream phenotype prediction models including those that use machine learning (See latent topics in Fig. 1.1).

Biological networks are inferred from experimental data. Recovery of biological networks is a reverse engineering problem: given experimental data, what is the network that gives rise to the data? Existing methods for network recovery fall into two primary categories: pairwise information theoretic methods, and graphical model methods. The advantages of the former are that they are embarrassingly parallel and have been shown to be effective on large-scale biological datasets (Faith et al. [2007], Lachmann et al. [2016], Margolin et al. [2006]). The main disadvantage of these methods is that they can only learn from one data distribution – e.g. they cannot incorporate new experimental data after an intervention has been applied to a system. The second category of network recovery methods are graphical models, often called structure learners. The advantages of these are that they have an

immediate causal interpretation (the direction of an edge implies cause and effect) and as a result a suite of methods have been built to jointly learn from observational and interventional data distributions (Hauser and Bühlmann [2012], Wang et al. [2017], Tong and Koller [2001]) – we call these "joint learners". This ability to jointly learn from different distributions takes a statistical model (e.g a Bayesian Network) and converts it into a causal model (e.g a Causal Bayesian Network). The disadvantages of graphical models are that they do not scale well with the number of nodes in the graph (the graph search space is combinatorial), and joint learners do not have parallel implementations. This means that joint structure learners have not been evaluated on datasets at scale – at most we observed evaluations on 500 node networks. Given that the genomes of most species are on the order of $1e3$-$1e4$ genes, there is a need to scale up these joint structure learners to larger networks. In this paper we introduce a new hybrid structure learner, named SP-GIES, that leverages the advantages of both categories of methods.

A causal model is preferable to a statistical model because directed causal edges in a network allow us to identify pathways, or mechanisms of action, from genotype to phenotype. In applications of biology and medicine, researchers seek to model both functional outcomes and the mechanisms leading to outcomes so that these can be understood and manipulated through therapeutics. Given this priority, there is a demand for causal models in this application space. This motivates the need to incorporate data from interventions on the system. We estimate the space of possible interventions in the environment and gene spaces to be $1e4$-$1e5$ depending on the species under investigation. A brute force sampling of this space of interventions is experimentally expensive and wasteful. Instead, there is a need to design feedback loops that choose advantageous interventions based on domain knowledge and learned knowledge. Work in optimal experimental design (OED) uses expected gain to choose interventions on variables in Bayesian Networks to drive causal discovery (Tong and Koller [2001], Hauser and Bühlmann [2014], Agrawal et al. [2019], Ghassami et al. [2018]).

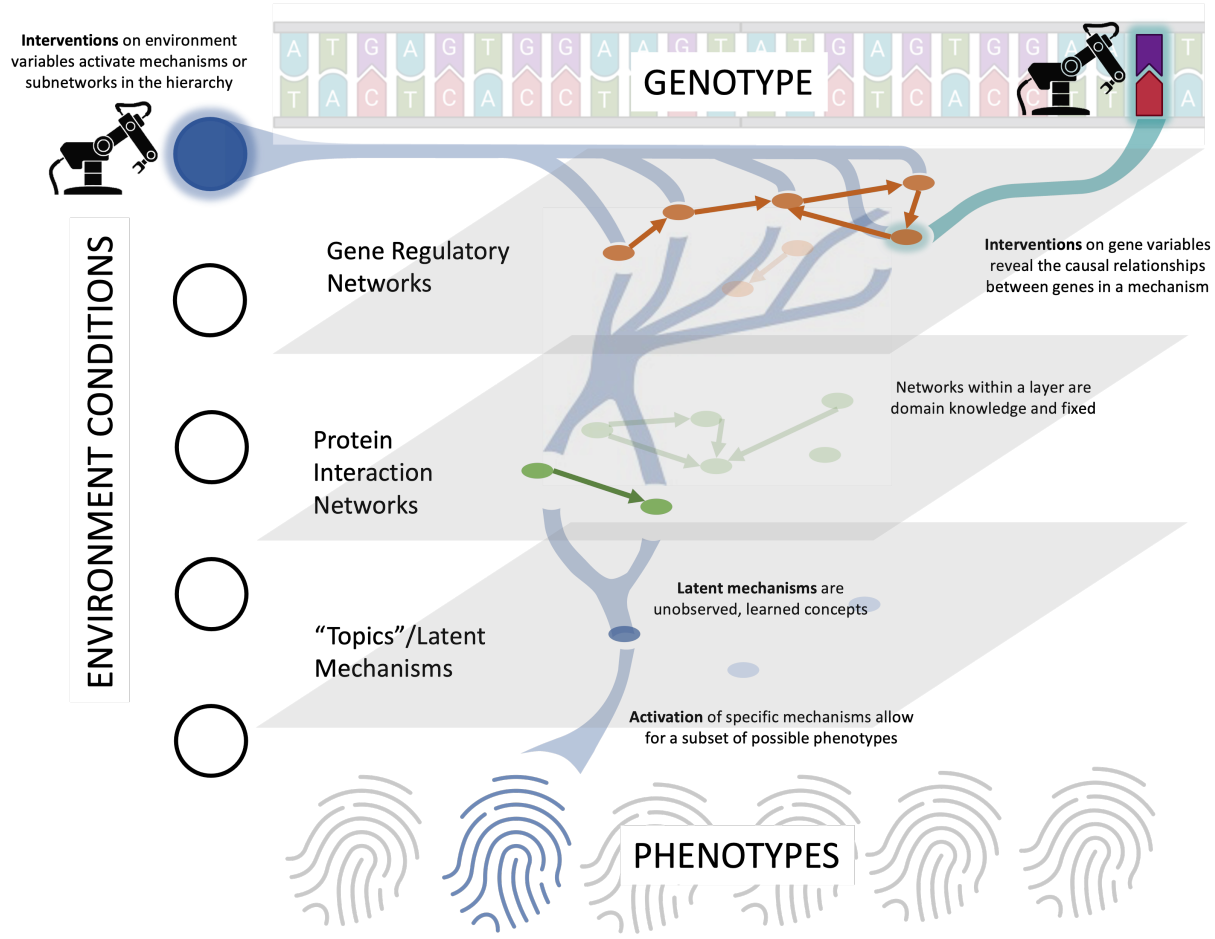


Figure 1.1: The hierarchy of biological networks which is partially known. Levels in this hierarchy can be represented as causal networks. Interventions activate mechanisms of action and reveal causal relationships within a mechanism. The space of possible interventions is large, motivating the need fgor autonomous design loops like AI driven experimental design and robotic laboratories. In theory, recovery of all networks allows for full genotype to phenotype mapping. However, in practice it is impossible to fully capture the data distribution and control for confounding variables. This motivates representing mechanisms of action as topic latent variables that can be learned via topic modeling.

This paradigm allows us to reason about how new inferred knowledge can connect to our existing domain knowledge, and provide a path forward for integrating AI into science domains.

The contributions of this work are as follows:

1. The implementation of a new hybrid structure learning method named Skeleton-Primed Greedy Interventional Equivalence Search (SP-GIES), based on the existing method GIES, that jointly learns from observational and interventional data. SP-GIES achieves better network recovery accuracy and a faster time to solution on large-scale networks compared to existing methods.

2. The application of SP-GIES to an optimal experimental design feedback loop that chooses optimal interventions on genes.

3. An analysis of the complexity of these methods and discussion of future directions, including unified libraries of causal algorithms, in order to achieve genome-scale network recovery for genotype to phenotype mapping with interventions.

## 1.1 Preliminaries

### *1.1.1 Causal Bayesian Networks*

Let $G = (V, E)$ be an acyclic graph defined by a set of vertices $V$ and directed edges $E$. The vertices of the graph represent random variables $X_1 ... X_p$. Under the Markov Assumption for Bayesian Networks, each variable $X_i$ is conditionally independent of its non-descendants given its parents. The joint distribution of a Bayesian network factorizes as $P(\mathbf{X}) = \prod_{i=1}^{p} P(X_i|Pa(X_i))$ [Spirtes et al., 2000]. The following definitions describe important properties of Bayesian Networks that are relevant for structure learning.

**Definition 1.1.1.** *Markov Equivalence Class (MEC)* The Markov Equivalence class (MEC)

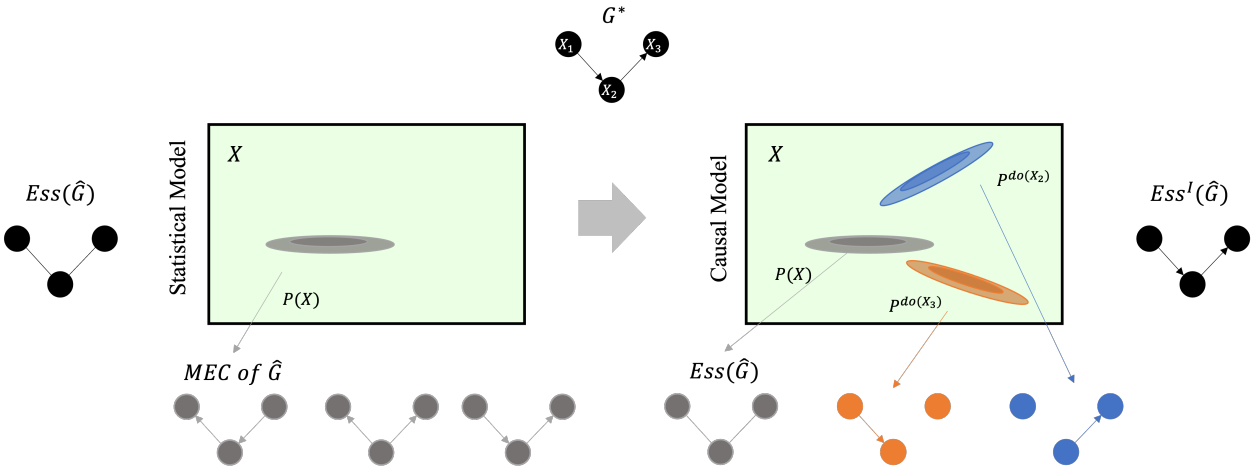


Figure 1.2: How to convert from purely statistical model (such as a Bayesian Network) to causal model (such as a Causal Bayesian Network) with interventional data. By modeling interventional distributions, the interventional essential graph $Ess^I(G)$ is closer to the true underlying causal graph compared to the essential graph $Ess(G)$. This figure is adapted from Schölkopf et al. [2021]

of a Bayesian Network $G$ consists of all directed acyclic graphs (DAGs) that share the same conditional independence relationships.

**Definition 1.1.2.** *Essential Graph (Ess)* An essential graph Ess(G) is a partially oriented graph that uniquely represents the MEC of a DAG. Directed arrows only exist on edges consistent across the equivalence class and all other edges are left undirected [Andersson et al., 1997].

A Causal Bayesian Network is a Bayesian Network where edges represent causal and effect relationships [Pearl, 1995a]. For example if a directed edge exists from $X_i$ to $X_j$, this means that no unobserved confounding variables are responsible for the their correlation and $X_i$ is a direct cause of $X_j$.

### *1.1.2 Structure Learning with Observational Data*

Constructing the graph structure from a set of instances (sampled values for each node $X_i$) is called structure learning. The following assumptions are made for learning a graph $\hat{G}$ from

data.

1. All random variables are observed, ie there are no hidden variables

2. **Causal Markov Assumption** - The data is generated from an underlying Bayesian Network $(G^*, \theta^*)$ over a set of random variables $X$

3. **Faithfulness Assumption** - The distribution $P^*$ over $X$ induced by $(G^*, \theta^*)$ satisfies no independencies beyond those implied by the structure of $G^*$

We are given a data set $D = \{X_1 ... X_p\}$ of $N$ samples from $P^*$ – this data is assumed to be independent and identically distributed (iid). The task is to learn a model $\hat{M} = (\hat{G}, \hat{\theta})$ that defines a distribution $\hat{P}$ that best fits true distribution $P^*$ [Koller and Friedman, 2009].

All graphs in the same MEC give rise to the same observational distribution. Therefore the best we can do with only observational data is recover the true graph's MEC. Several graphical model based methods learn the structure from observational data only. These are split into constraint-based and score-based methods. There are also pairwise information theoretic methods that learn from only observational data. We discuss these in Section 2.

### *1.1.3 Structure Learning with Interventional Data*

In addition to structure learning on observational data, we want to incorporate interventional datasets into our learning procedures. Intervening on a set of random variables $I \subseteq X$ removes incoming edges to the random variables $I$, and sets the joint distribution to a new interventional distribution $P^{do(X_i=x)}$. Here, we are setting node $X_i$ to a value $x$. In this paper, we assume a "hard" interventions, where a node is set to a constant value, because this mimics the types of interventions we are able to perform in biology applications – e.g. gene knockout experiments. This is contrast to a "soft" intervention which is samples $X_i = x$ from a distribution.

**Definition 1.1.3.** *Interventional Distribution* The joint distribution after a hard intervention is:

$$P^{do(X_i=x)} = \begin{cases} \prod_{j \neq i}^{p} P(X_j | Pa(X_j) & if X_i = x \\ 0 & otherwise \end{cases}$$

Jointly learning on observational and interventional data allows us to correctly isolate causal relationships and orient edges in the graph. An estimated $Ess(G)$ from observational data can be further refined into an I-essential graph ($Ess^I(G)$). This is a partially oriented DAG that represents an interventional Markov Equivalence class (I-MEC) (see Fig. 1.2). There are several existing methods that jointly estimate structure given observational and interventional data. See Vowels et al. [2021] for a full list of these structure learners.

### *1.1.4 Optimal Experimental Design*

We have access to new interventional data by sampling the system, but we wish prioritize the interventional experiments that will recover the true causal graph the fastest. This is done via maximization of an expected utility function, $U$, over the set of potential interventions: $\hat{I} = \arg\max_I \mathbb{E}_{G|D}[U]$ where $U$ is a utility function like mutual information, number of oriented edges, etc.

Our goal is to recover the underlying Causal Bayesian Network given a mix of interventional and observational samples, and some prior information about the causal edges in the graph. We cast this as a Bayesian Inference problem over the space of DAGs. We have a prior $P(G)$ which encodes any prior structural knowledge about the underlying DAG. Applying Bayes Theorem gives us the posterior distribution $P(G|D) \propto P(D|G)P(G)$. The likelihood is $P(D|G) = \int_\theta P(D, \theta|G)d\theta = \int_\theta P(D|\theta, G)P(\theta|G)d\theta$, where we have marginalized out the parameters of the graph [Tong and Koller, 2001]. $D$ is a mix of observational and interventional data. The posterior distribution is sampled via the joint structure learners described

in the previous section. After sampling new data from the chosen intervention using OED, the new data is concatenated to the existing dataset and the posterior is re-sampled using the structure learners.

# CHAPTER 2

# RELATED WORK

Table 2.1 and 2.2 provide summary of the related work in biological network recovery and optimal experimental design respectively.

Table 2.1: Properties of all the learners covered in this paper. Methods are split row-wise by the nature of the learner. Data type refers to observational and/or interventional data capability of the learners. Evaluation dataset lists the benchmarks used for evaluation for each paper. Finally, the table also lists the maximum number of nodes the algorithm was evaluated on and the complexity of the algorithm as reported in the corresponding papers. $p$ is the number of nodes, $n$ is the number of samples. $k$ is the maximal degree of any node in the graph. For IGSP, we did not find a complexity analysis in the paper.

| | | Data Type | | Evaluation dataset | | | Scaling | |
|---|---|---|---|---|---|---|---|---|
| Paper | | Observ. | Interv. | Random | DREAM | Large-Scale | Max # of nodes | Worst case runtime |
| Pairwise Info. Theoretic | *CLR* Faith et al. [2007] | ✓ | | | | ✓ | 4,345 | $O(p^3n^3)$ |
| | *ARACNE-AP* Lachmann et al. [2016] | ✓ | | | | ✓ | 1,331 | $O(p^3 + p^2n^2)$ |
| | *Pinna et. al* Pinna et al. [2010] | | ✓ | ✓ | ✓ | | 100 | $O(p^3 + np)$ |
| Graphical Models | *PC* Zarebavani et al. [2019],Le et al. [2016], Madsen et al. [2017] | ✓ | | ✓ | ✓ | ✓ | 5,361 | $O(p^{k+2})$ |
| | *GIES* Hauser and Bühlmann [2012] | ✓ | ✓ | ✓ | ✓ | | 500 | $O(2^p)$ |
| | *IGSP* Wang et al. [2017] | ✓ | ✓ | | | | 24 | $O(2^p)$ |
| | *MCMC Mallows* Rau et al. [2013] | ✓ | ✓ | | ✓ | | 10 | $O(2^p)$ |
| Hybrid | *SP-GIES* [this paper] | ✓ | ✓ | ✓ | ✓ | ✓ | 2,000 | $O(2^p)$ |

Table 2.2: Properties of the OED methods covered in the paper. OED criteria refers to the utility function used for selection of next experiments. Note that the the complexity listed here corresponds only to choosing the next intervention or sets of interventions. Each algorithm here also has the cost of sampling from the posterior distribution which is equivalent to the complexity of the learners in Table 2.1

| | OED criteria | | Evaluation dataset | | | Scaling | |
|---|---|---|---|---|---|---|---|
| Paper | Info. theory | Edge orient. | Random | DREAM | Large-Scale | Max # of nodes | Big O |
| Ness et al. [2017] | | ✓ | | ✓ | | 17 | $O(\|E\| * p!)$ |
| Hauser and Bühlmann [2014] | | ✓ | | | | 40 | polynomial in $p$ |
| Tong and Koller [2001] | ✓ | | | | | 12 | $O(pn)$ |
| *ABCD* [Agrawal et al., 2019] | ✓ | | ✓ | ✓ | | 10 | $O(p^2n)$ |
| *BED* [Ghassami et al., 2018] | ✓ | | ✓ | ✓ | | 100 | polynomial in $p$ |

### *2.0.1 Structure Learning With Interventional Data for Biology Network Recovery*

Pairwise information theoretic learners use calculated information theoretic metrics to estimate the dependence of two variables. A threshold is applied to the metric to obtain a

network representation of the system. Pinna et al. [2010] use a deviation matrix (difference between intervened samples and steady state samples) to estimate an initial network. This algorithm won the DREAM4 network recovery challenge and was the state-of-the-art for gene regulatory network reconstruction at the time. Faith et al. [2007] introduce CLR (context likelihood of relatedness) which calculates the pairwise B-spline mutual information between genes in the E. coli K-12 gene regulatory network. CLR is implemented in MATLAB, with a fast parallel kernel for calculating the pairwise mutual information. Margolin et al. [2006] developed ARACNE (Algorithm for the Reconstruction of Accurate Cellular Networks) which is a similar pairwise mutual information based network recovery algorithm. ARACNE additionally employs a DPI (data processing inequality) algorithm that removes indirect interactions. ARACNE was more recently upgraded to ARACNE-AP [Lachmann et al., 2016], which is implemented in Java, to handle adaptive partitioning to calculate the mutual information – this achieved a 200x computational performance improvement.

Graphical model based learners reveal a much finer structure than pairwise methods because they are based on Bayesian Network. The PC algorithm is a constraint-based structure learner that only handles observational datasets. PC learns a graph by performing conditional independence testing on pairs of nodes conditioned on other nodes. Since its introduction, many parallel implementations and optimizations of the PC algorithm have been added (Le et al. [2016], Madsen et al. [2017]) including one that allows for GPU parallelism (Zarebavani et al. [2019]). Existing structure learners that jointly learn from observational and interventional data are either score-based or hybrid score/constraint-based methods. Rau et al. [2013] propose an algorithm called MCMC-Mallows based on Markov Chain Monte Carlo sampling over the space of potential Gaussian DAGs and optimizes the best scoring DAG. MCMC-Mallows was evaluated on the DREAM4 networks, and scored better than Pinna et al. [2010] on two of the networks. Hauser and Bühlmann [2012] proposed Greedy Equivalence Intervention Search (GIES) which is a greedy score-based approach that is an

extension of the original GES learner. The time complexity of a step of GIES is polynomial in $p$ (the number of vertices in the graph) ; otherwise, it is in the worst case exponential – however the authors note and show empirically that GIES is more efficient than this worst case. GIES was evaluated on DREAM4 and achieved scores in the top third of all participants. Intervention Greedy Sparsest Permutation (IGSP, Wang et al. [2017]) is a hybrid score/constraint, and non-parametric extension of the GSP learner. IGSP associates a DAG to every permutation of random variables and greedily updates the DAG by transposing elements of the permutation. IGSP performed comparably to GIES on protein signaling data [Sachs et al., 2005] and better than GIES on a real gene expression dataset with 24 genes.

### *2.0.2 Optimal Experimental Design for Biology Network Recovery*

One way to choose the optimal intervention is to choose the set of interventions that maximizes the mutual information or leads to the greatest decrease in entropy. Tong and Koller [2001] sample a set of orderings from the distribution over graphs and parameters. They then use this set of orderings to compute entropy terms and select the intervention with the lowest expected posterior loss. Agrawal et al. [2019] extend the work of Tong and Koller [2001] with the ABCD strategy – a greedy implementation of batched experimental design. They use a utility function based on the expected entropy decrease of an intervention. This requires calculating expectations over the graph and parameter spaces, however, they are able to make a tractable algorithm by using bootstrapping and weighted importance sampling.

Another way to choose the optimal intervention is to choose the intervention that leads to the maximal number of oriented edges. Ness et al. [2017] use optimal experimental design to recover protein signaling networks [Sachs et al., 2005]. They use a utility function based on the expected information gain of an intervention given the observational MEC and other interventions in the batch. This algorithm, however, has factorial dependence on batch size. Ghassami et al. [2018] use the expected number of oriented edges of an essential graph as

the utility function. One drawback of this work as that the essential graph of the ground truth network was given as input to the algorithm, rather than learned from the data. Since access to $Ess(G^*)$ is rarely ever given in real world problems, the results of this evaluation are not relevant to the type of causal discovery we are interested in. Hauser and Bühlmann [2014] similarly propose a utility function based on the number of oriented edges of a skeleton graph.

# CHAPTER 3

# SP-GIES

Here we describe our hybrid algorithm Skeleton-Primed GIES (SP-GIES). A skeleton is an undirected graph with edges that correspond to edges in a DAG. The algorithm is a simple sequential use of a pairwise learner to estimate the skeleton of the data using only observational samples, and the joint graphical model structure learner GIES to orient those edges. The two step algorithm is as follows:

1. Use ARACNE,CLR or PC to generate a skeleton with only observational samples
2. Use the output of (1) to restrict the possible edge set for the GIES learner. Jointly learn from observational and interventional data using GIES.

For (1), if the PC algorithm is used then the input to (2) is an Ess(G) which is represented by a partially oriented DAG or PDAG. We chose GIES for (2) since it has an open source implementation that is part of the widely used pcalg library in R. For scaling studies, we chose to use the R implementation of PC algorithm. This is because CLR and ARACNE are implemented in MATLAB and Java respectively. Since GIES only has an R implementation, we chose to use the R implementation of PC for our scaling studies.

While (1) is easily parallelizable on multiple processors or deployable to a GPU, the bottleneck of the computation is in the GIES step. Still in Fig. 3.1, we see that SP-GIES is able to reach a faster time to solution than GIES. To understand why this is, we investigated the GIES algorithm. GIES is a greedy score-based structure learner. The score function is the Bayesian Information Criteria which is based on the maximum likelihood estimate of the graph given the current data. Starting from an empty essential graph, in the forward phase of the algorithm an edge is added to $Ess(G)$ such that the new essential graph has the maximal score. The algorithm iterates through all possible edges until it finds the corresponding essential graph with the highest score. The backward phase is similar,

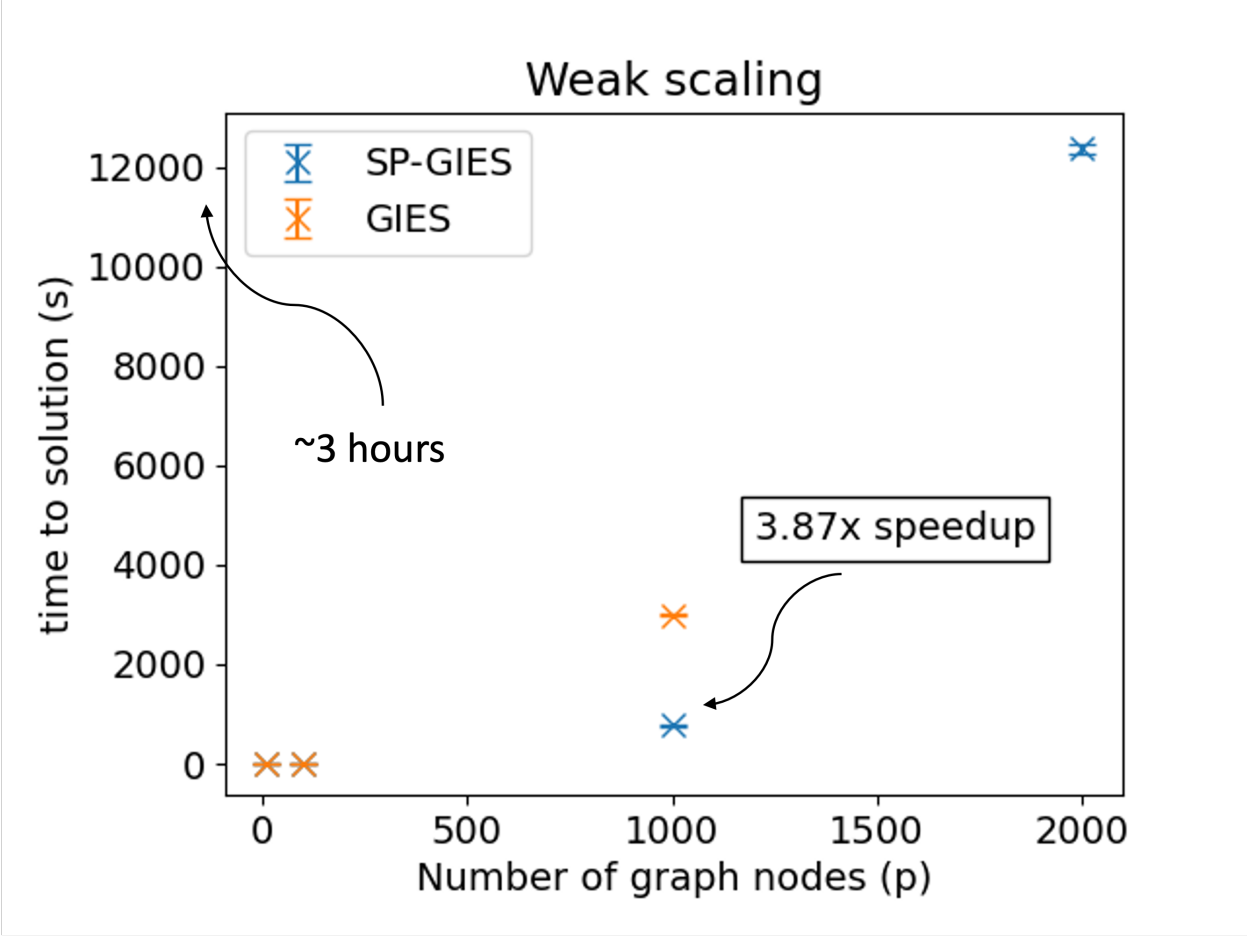


Figure 3.1: A weak scaling study comparing the SP-GIES and GIES algorithms. The number of samples is fixed to $n$=1,000. Data points are averaged over 3 runs. The study was performed on a 2.8 Ghz AMD EPYC Milan 7543P 32 core CPU and a NVIDIA A100 GPU. The GIES 2,000 node run exceeded 8 hours and it not plotted here.

except an edge is removed instead of added. By restricting the set of possible edges to the GIES algorithm by first estimating the skeleton, we narrow the size of the search space for both phases – this is what results in the speedup. Moreover, since we use a fast GPU implementation of the PC algorithm (Zarebavani et al. [2019]), the overhead of estimating the skeleton first is minimal.

Table 2.1 shows that all joint learners have a worst case exponential complexity – this attributed to the combinatorial search space over graphs. However, the average empirical complexity is related to the topology of the network i.e, the size of the largest clique. Since

the average complexity is difficult to calculate, we show in Figure 3.1 that SP-GIES reaches a faster time to solution as the size of the network increases. For most biological datasets, $p$ is larger than $n$ since the size of the genome is large and experimental data is difficult to collect. We ran a scaling study for SP-GIES using a similar study as Hauser and Bühlmann [2012] except we were able to scale up to $p$=2,000 nodes versus the $p$=500 nodes in the GIES paper. At 2,000 nodes, the GIES algorithm failed to find a solution in 8 hours. We see that SP-GIES is able to reach a faster time to solution than GIES because of the restriction of the edge set discussed previously.

# CHAPTER 4

# RESULTS

Each learner was evaluated on three metrics: Structural Hamming Distance (SHD), Structural Interventional Distance (SID) and the AUC-PR. The SHD is the L1 error between the generated DAG and the ground truth DAG. The SID counts the incorrect interventional distributions as defined by Eq. 1.1.3. The AUC-PR is the area under the precision-recall curve. All three of these metrics are common evaluations for causal discovery. However, we note that SHD and AUC-PR are biased towards empty graphs. Suppose a structure learner $A$ learns a graph $G_A$ with no edges and with $SHD_A = |E|$. Another learner $B$ learns a graph $G_B$ with $SHD_B > |E|$, but correctly learns some of the causal edges in the true graph. According to the SHD, $G_A$ is better and we may posit that $A$ is the better learner for the data. However, an argument can be made for $G_B$ because it captures more causal relationships than $G_A$. This means that a lower SHD may not always indicate a model that best fits the data distribution. A similar argument can be made for the AUC-PR. The drawback of the SID is the computational complexity is quadratic in the number of nodes for sparse networks, and cubic in the number of nodes for dense networks. This is shown empirically up to 50 nodes by Peters and Bühlmann [2015].

Datasets at various scales were generated or collated for evaluation of SP-GIES against existing learners. Random networks were generated using the Networkx Python library. Data distributions are multinomial Gaussians. Interventional data is generated from these graphs using the GaussDAG class from the CausalDAG Python package Chandler Squires [2018]. The DREAM4 insilico challenge provides observational and interventional knockout data of gene regulatory networks generated from stochastic ODEs. The dataset provides one interventional sample per node in the graph. The ground truth networks are subnetworks of the larger known E.coli gene regulatory network, however the samples are generated synthetically. The RegulonDB dataset is provided by Faith et al. [2007]. The network is the

Table 4.1: Evaluation of learners on random networks of size 10. Three different types of random networks were generated: Erdös Renyi Erdős et al. [1960], Scale free Barabási and Bonabeau [2003], and Small world Watts and Strogatz [1998]. $p$ and $k$ refer to parameters used to generate the graphs, not the number of nodes and degree as used previously. Results are averaged over 30 random graphs, each learned with 100 data samples. The superscript on the algorithms refers to the data type used (O for observational, OI for mixed observational and interventional). For SP-GIES, ARACNE, CLR, and then PC were used for skeleton estimation for Erdös Renyi, Scale Free and Small World respectively since these were the best performers.

| Random Networks size 10 | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| Algorithm | Erdös Renyi ($p = 0.5$) | | | Scale Free ($k = 2$) | | | Small World ($p = 0.5, k = 2$) | | |
| | SHD | SID | AUC-PR | SHD | SID | AUC-PR | SHD | SID | AUC-PR |
| $PC^O$ | 22.77 | 70.73 | 0.35 | 11.27 | 61.00 | 0.62 | 8.23 | 36.33 | 0.59 |
| $CLR^O$ | 22.67 | 71.43 | 0.34 | 17.27 | 68.30 | 0.35 | 10.83 | 51.17 | 0.43 |
| ARACNE-$AP^O$ | 21.83 | 79.00 | 0.36 | 16.63 | 76.70 | 0.37 | 10.10 | 52.80 | 0.43 |
| $GES^O$ | 23.27 | 53.34 | 0.47 | **11.20** | **28.57** | **0.69** | 10.23 | **17.27** | 0.67 |
| $GIES^{OI}$ | 23.93 | **39.23** | 0.59 | 14.13 | 36.43 | 0.68 | 21.30 | 17.70 | 0.55 |
| $Pinna^{OI}$ | 22.67 | 57.60 | 0.26 | 18.00 | 56.93 | 0.28 | 13.97 | 29.73 | 0.13 |
| $IGSP^{OI}$ | **20.00** | 67.27 | 0.38 | 15.00 | 53.23 | 0.41 | 6.73 | 22.43 | 0.55 |
| SP-$GIES^{OI}$ | 24 | 39.87 | 0.59 | 11.34 | 48.53 | 0.64 | **5.90** | 26.53 | **0.68** |
| NULL | 21.00 | 66.83 | **0.61** | 16.00 | 61.30 | 0.58 | 10.00 | 30.00 | 0.55 |

Table 4.2: Evaluation of learners on the DREAM4 networks of size 10. The dataset contains 11 observational samples and 10 interventional samples. Interventional samples are gene-knockout experiments. For SP-GIES, ARACNE was used for skeleton estimation, except for Network 4 which used CLR. Note that for Network 3, we used adaptive learning in the GIES subroutine (adaptive="triples") to achieve the best performance for the SP-GIES learner. We did not see significant improvement with adaptive learning for the other networks.

| DREAM4 insilico size 10 | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Algorithm | Network 1 | | | Network 2 | | | Network 3 | | | Network 4 | | | Network 5 | | |
| | SHD | SID | AUC-PR | SHD | SID | AUC-PR | SHD | SID | AUC-PR | SHD | SID | AUC-PR | SHD | SID | AUC-PR |
| $PC^O$ | 14 | 39 | 0.31 | 12 | 56 | 0.43 | 16 | 64 | 0.26 | 14 | 45 | 0.16 | 11 | 57 | **0.59** |
| $CLR^O$ | 13 | 36 | 0.45 | **10** | 56 | 0.53 | 14 | 62 | 0.36 | 14 | 47 | 0.19 | 15 | 62 | 0.33 |
| ARACNE-$AP^O$ | **11** | 28 | 0.56 | 12 | 56 | 0.45 | 13 | 61 | 0.43 | 15 | 45 | 0.06 | 12 | 57 | 0.46 |
| $GES^O$ | 13 | 29 | 0.49 | 15 | 52 | 0.30 | 17 | 64 | 0.15 | 19 | 48 | 0.10 | 11 | 51 | 0.14 |
| $GIES^{OI}$ | 13 | 31 | 0.51 | 12 | 49 | 0.51 | 16 | **40** | 0.49 | 15 | **26** | 0.34 | 9 | 39 | 0.37 |
| $Pinna^{OI}$ | **11** | **22** | 0.49 | 11 | 56 | 0.08 | 14 | 61 | 0.57 | **12** | 42 | 0.37 | 11 | 38 | 0.47 |
| $IGSP^{OI}$ | 13 | 27 | 0.07 | 13 | 54 | 0.08 | 16 | 65 | 0.07 | 15 | 38 | 0.06 | 13 | 48 | 0.19 |
| SP-$GIES^{OI}$ | **11** | 23 | **0.63** | **10** | **34** | **0.6** | **12** | 47 | **0.63** | **12** | 35 | 0.33 | **6** | **30** | 0.45 |
| NULL | 12 | 27 | 0.57 | 11 | 53 | 0.58 | 14 | 61 | 0.57 | **12** | 36 | **0.56** | 11 | 46 | 0.56 |

current estimated E.coli K12 regulatory network and contains 1146 nodes, 3175 edges and 524 real experimental samples. The RegulonDB contains environmental interventions and gene knockout interventions. Existing learners currently only model gene variables, therefore

the environmental interventional data is treated as observational data. Future work includes explicitly modeling environmental condition variables.

Table 4.3: Evaluation of learners on large-scale networks. Pinna et al. [2010] requires one interventional sample per gene, since the RegulonDB dataset does not provide this we did not evaluate Pinna on this dataset. The SID calculation for CLR on the random 1,000 node network resulted in an out of memory error on our compute node. $GES^{O}$ required setting the maximum degree to 10 in order to achieve convergence

| Evaluation of Large Scale Networks | | | | | | |
|---|---|---|---|---|---|---|
| Algorithm | RegulonDB 1146 nodes, 3179 edges | | | Small world 1000 nodes, 1000 edges | | |
| | SHD | SID | AUC-PR | SHD | SID | AUC-PR |
| ARACNE-AP$^{O}$ | 3,752 | 25,979 | 0.04 | 1,000 | 3,883 | 0.50 |
| CLR$^{O}$ | **3,095** | **18,069** | 0.30 | 2,704 | OOM error | 0.45 |
| PC$^{O}$ | 3,963 | 23,908 | 0.01 | 467 | 2,447 | 0.82 |
| GES$^{O}$ | 8,712 | 74,224 | 0.005 | 1,523 | **879** | 0.88 |
| GIES$^{OI}$ | 8,355 | 80,580 | 0.006 | 1,623 | 1,097 | 0.84 |
| Pinna$^{OI}$ | x | x | x | 16,577 | 4,334 | 0.002 |
| IGSP$^{OI}$ | | | | | | |
| SP-GIES$^{OI}$ | 3,154 | 22,114 | 0.10 | **341** | 975 | **0.91** |
| NULL | 3,179 | 18,294 | **0.50** | 1,000 | 3,883 | 0.50 |

Results of our evaluation on random networks, DREAM4 gene regulatory networks, and genome-scale networks are shown in Table 4.1, Table 4.2, and Table 4.3 respectively. As a reference, we also include the scores for a network without any dependencies (no edges), named NULL. Many real world networks, including gene regulatory networks, are sparse. As a result the NULL learner often appears to perform very well since the true networks lie close to an empty network in combinatorial space. For random networks of size 10, GIES outperforms other methods across all network types. For DREAM4, SP-GIES most frequently gets the best score across all metric types over all five networks. We see that the addition of interventional data improves the performance of algorithms that can handle both data types (namely GIES, and SP-GIES).

For the RegulonDB network, the best scoring method is CLR. This is likely because the CLR method calculates a threshold value to prune edges so that the learner achieves 60%

precision compared to the ground truth network – this makes it an unfair comparison since the ground truth network must be known a priori. We used CLR for the skeleton estimation of SP-GIES. An unexpected result is that SP-GIES appears to worsen the estimate of the graph. Nandy et al. [2018] show that the search path of GES may have to leave the search space determined by a skeleton, even though the true CPDAG lies within these search spaces. Therefore restricting search space of the GIES algorithm is not entirely controllable and can lead to misguided optimization. Still, we see that priming the GIES algorithm with the CLR skeleton produces better results than GIES on its own.

There are a couple reasons the joint learners may not be performing better here. First, the experimental gene-knockout samples only contain 6 different types of knockouts out of the potential 1,146 genes. It's possible that more coverage over the intervention space will improve the joint learners. Second, many of the experimental samples come from condition interventions which are not modelled here. Still compared to GIES, SP-GIES achieves better network recovery on the RegulonDB network. SP-GIES also outperforms all other methods on the 1,000 node random networks for the SHD and AUC-PR metrics. Overall, SP-GIES performs better than GIES on larger scale networks and on DREAM4 networks. As these are both relevant datasets for genome-scale network recovery, we view this as a promising step forward.

For many of the networks we evaluated here, the NULL graph achieves the best score. As described previously, the SHD and AUC-PR are sometimes misleading metrics for this use case. Interestingly, the NULL graph never scores best for the SID metric. This could indicate that causal network recovery methods should be evaluated with the SID rather than other metrics. However, more analysis is needed to understand the validity of this and is outside the scope of this paper.

To understand the performance gains of SP-GIES over GIES on optimal experimental design, we evaluated both algorithms with two different OED criteria on the DREAM4 insilico

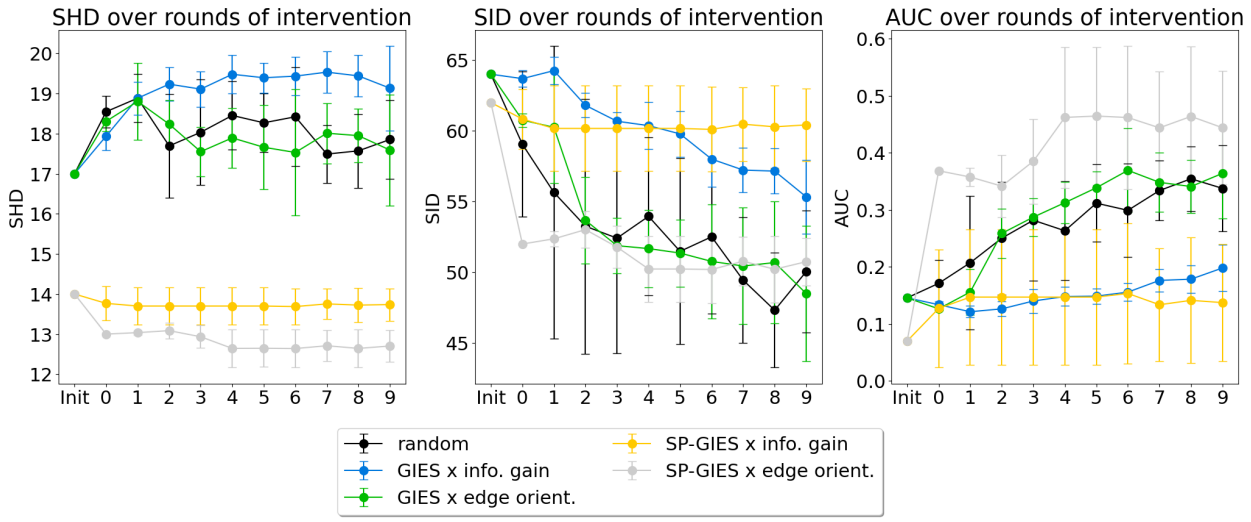


Figure 4.1: An evaluation of OED strategies for DREAM4 10 node network #3 over 10 rounds of intervention. At each round of intervention, a new interventional data sample is added to the current dataset corresponding to the chosen intervention. GIES and SP-GIES are joint learners. Metrics are averaged over 30 runs.

10 node dataset. We limited this evaluation to small networks since the time complexity of OED includes posterior sampling and optimal gene selection for each round of intervention. We follow the framework of Agrawal et al. [2019], which includes bootstrap sampling of the posterior distribution. An example result for a DREAM4 insilico network is shown in Fig. 4.1. We observe two trends here. First, the edge orientation strategy achieves better performance than the information gain strategy. Second, the SP-GIES learner boosts the initial performance of the learner, however, it easily stagnates and additional interventional samples do not continue to improve the algorithm at the same rate as the GIES based strategies. Still the SP-GIES x OED runs achieve better performance than GIES x OED, and there is some improvement in accuracy with the addition of interventional data. We speculate that stagnation may happen with the SP-GIES joint learner because by fixing the skeleton, we also restrict the search space of the learner. This may lead to the GIES step of the algorithm getting stuck in a local minima. This motivates incorporating some level of uncertainty into the skeleton, or randomly including sets of edges outside the skeleton into the search space – we leave this to future work.

# CHAPTER 5

# DISCUSSION

Understanding genome-scale networks is important for building a causal mapping from genotype to phenotype. However, recovery of genome-scale networks from interventional datasets has not yet been realized because of the computational complexity of structure learners that jointly learn from mixed data. Existing methods like GIES, IGSP, MCMC Mallows and others are only evaluated on small networks up to 500 nodes and exhibit worst case exponential complexity. This makes subsequent optimal experimental design techniques computationally intractable since the complexity is then multiplied by the computation of choosing the optimal intervention. This is a huge issue in realizing causal discovery at scale via autonomous design loops.

In this paper, we note that structure learners built for observational datasets have parallel implementations on multiple processors and on GPUs. Our SP-GIES learner first estimates a skeleton using these parallel implementations. This restricts the edge set for the subsequent joint structure learner and improves the time to solution. However, SP-GIES is still bottlenecked by the scaling of the joint learner GIES – for example running network sizes of $1e4$ nodes did not complete in 24 hours. To realize recovery of genome-scale networks on the order of $1e3$-$1e4$, and to sample an interventional space on the order $1e4$-$1e5$, we must have breakthroughs in distributed parallel implementation of joint learners. Joint structure learners currently do not have parallel implementations because they are score-based learners (or hybrid constraint/score-based learners). At each step of the algorithm the candidate graph is scored, typically using a function of the likelihood $P(D|G)$ over the entire graph. As we saw in Section 1.1, the joint distribution of a DAG is a product of conditional probabilities which, for each node, depends on the parent nodes. As a result, the calculation of the likelihood is sequential and typically optimization over this space is done greedily. It is not intuitive how the graph may be partitioned into subgraphs for individual compute kernels.

Constraint based methods, on the other hand, use pairwise conditional independence testing to resolve edges. A naive approach to scaling is to generate a separate compute thread per calculation, however, one can group variables into blocks to further minimize computations and boost performance. This type of performance enhancement has not yet been realized by score-based learners, and this presents a roadblock for existing joint learners.

In evaluating existing methods and designing the SP-GIES method, we found that learners and OED strategies are implemented in a varied set of languages including Java, MATLAB, R, Python and C. This is due to the diverse nature of backgrounds interested in causal discovery of biological networks including from biology, statistics, computer science, machine learning, and representation learning. There has been a recent interest in unifying tools for causal discovery – for example CausalDiscoveryToolbox Kalainathan and Goudet [2019] , Pgmpy Ankan and Panda [2015] and CausalDAG Chandler Squires [2018] are Python libraries that provide varying support for graphical and/or causal modeling. However, in the case of CausalDiscoveryToolbox, the library is a Python wrapper for R code, which in turn is a wrapper for C code. In the case of CausalDAG, many of the implementations are incomplete. For Pgmpy, no joint interventional learners are implemented. To realize better parallel implementations of causal algorithms and OED strategies, we should encourage collaborations with the supercomputing community. Therefore, there is a need for exclusive Python or C implementations of these models and algorithms since these languages are popular in the supercomputing community.

# CHAPTER 6

# CONCLUSION

We present SP-GIES, a joint structure learner that leverages parallel observational learners to estimate a skeleton and initialize the GIES joint learner. We provide a systematic evaluation of SP-GIES against existing methods for datasets at various scales and with various metrics. We see that SP-GIES is able to provide better network recovery accuracy for larger scale networks up to 2,000 nodes and on biological networks up to 1,146 nodes. This scale of network recovery for joint learners has not been achieved to our knowledge. SP-GIES reaches a faster time to solution than existing method GIES and provides up to 3.87x speedup. We also show that SP-GIES improves subsequent OED strategies. Future work includes a distributed parallel implementation of SP-GIES, and incorporation of SP-GIES into an autonomous experimentation design loop.

# REFERENCES


Raj Agrawal, Chandler Squires, Karren Yang, Karthikeyan Shanmugam, and Caroline Uhler. Abcd-strategy: Budgeted experimental design for targeted causal structure discovery. In *The 22nd International Conference on Artificial Intelligence and Statistics*, pages 3400–3409. PMLR, 2019.

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

Ankur Ankan and Abinash Panda. pgmpy: Probabilistic graphical models using python. In *Proceedings of the 14th Python in Science Conference (SCIPY 2015)*. Citeseer, 2015.

Albert-László Barabási and Eric Bonabeau. Scale-free networks. *Scientific american*, 288 (5):60–69, 2003.

Rodolphe Barrangou and Jennifer A Doudna. Applications of crispr technologies in research and beyond. *Nature biotechnology*, 34(9):933–941, 2016.

Ivan Brugere, Brian Gallagher, and Tanya Y Berger-Wolf. Network structure inference, a survey: Motivations, methods, and applications. *ACM Computing Surveys (CSUR)*, 51 (2):1–39, 2018.

Chandler Squires. *`causaldag`: creation, manipulation, and learning of causal models*, 2018. URL `https://github.com/uhlerlab/causaldag`.

Anthony Coutant, Katherine Roper, Daniel Trejo-Banos, Dominique Bouthinon, Martin Carpenter, Jacek Grzebyta, Guillaume Santini, Henry Soldano, Mohamed Elati, Jan Ramon, et al. Closed-loop cycles of experiment design, execution, and learning accelerate systems biology model development in yeast. *Proceedings of the National Academy of Sciences*, 116(36):18142–18147, 2019.

Fernando M Delgado and Francisco Gómez-Vela. Computational methods for gene regulatory networks reconstruction and analysis: A review. *Artificial intelligence in medicine*, 95: 133–145, 2019.

Atray Dixit, Oren Parnas, Biyu Li, Jenny Chen, Charles P Fulco, Livnat Jerby-Arnon, Nemanja D Marjanovic, Danielle Dionne, Tyler Burks, Raktima Raychowdhury, et al. Perturb-seq: dissecting molecular circuits with scalable single-cell rna profiling of pooled genetic screens. *cell*, 167(7):1853–1866, 2016.

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

Jeremiah J Faith, Boris Hayete, Joshua T Thaden, Ilaria Mogno, Jamey Wierzbowski, Guillaume Cottarel, Simon Kasif, James J Collins, and Timothy S Gardner. Large-scale mapping and validation of escherichia coli transcriptional regulation from a compendium of expression profiles. *PLoS biology*, 5(1):e8, 2007.

Nikolaus Fortelny and Christoph Bock. Knowledge-primed neural networks enable biologically interpretable deep learning on single-cell sequencing data. *Genome biology*, 21(1): 1–36, 2020.

AmirEmad Ghassami, Saber Salehkaleybar, Negar Kiyavash, and Elias Bareinboim. Budgeted experiment design for causal structure learning. In *International Conference on Machine Learning*, pages 1724–1733. PMLR, 2018.

Marco Grzegorczyk. An introduction to gaussian bayesian networks. In *Systems biology in drug discovery and development*, pages 121–147. Springer, 2010.

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

Alain Hauser and Peter Bühlmann. Two optimal strategies for active learning of causal models from interventional data. *International Journal of Approximate Reasoning*, 55(4): 926–939, 2014.

Alain Hauser and Peter Bühlmann. Jointly interventional and observational data: estimation of interventional markov equivalence classes of directed acyclic graphs. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 77(1):291–318, 2015.

José Jiménez-Luna, Francesca Grisoni, and Gisbert Schneider. Drug discovery with explainable artificial intelligence. *Nature Machine Intelligence*, 2(10):573–584, 2020.

Diviyan Kalainathan and Olivier Goudet. Causal discovery toolbox: Uncover causal relationships in python. *arXiv preprint arXiv:1903.02278*, 2019.

George Em Karniadakis, Ioannis G Kevrekidis, Lu Lu, Paris Perdikaris, Sifan Wang, and Liu Yang. Physics-informed machine learning. *Nature Reviews Physics*, 3(6):422–440, 2021.

Daphne Koller and Nir Friedman. *Probabilistic graphical models: principles and techniques*. MIT press, 2009.

Alexander Lachmann, Federico M Giorgi, Gonzalo Lopez, and Andrea Califano. Aracne-ap: gene network reverse engineering through adaptive partitioning inference of mutual information. *Bioinformatics*, 32(14):2233–2235, 2016.

Thuc Duy Le, Tao Hoang, Jiuyong Li, Lin Liu, Huawen Liu, and Shu Hu. A fast pc algorithm for high dimensional causal discovery with multi-core pcs. *IEEE/ACM transactions on computational biology and bioinformatics*, 16(5):1483–1495, 2016.

Nathan E Lewis, Harish Nagarajan, and Bernhard O Palsson. Constraining the metabolic genotype–phenotype relationship using a phylogeny of in silico methods. *Nature Reviews Microbiology*, 10(4):291–305, 2012.

Maxwell W Libbrecht and William Stafford Noble. Machine learning applications in genetics and genomics. *Nature Reviews Genetics*, 16(6):321–332, 2015.

Zachary C Lipton. The mythos of model interpretability: In machine learning, the concept of interpretability is both important and slippery. *Queue*, 16(3):31–57, 2018.

Jianzhu Ma, Michael Ku Yu, Samson Fong, Keiichiro Ono, Eric Sage, Barry Demchak, Roded Sharan, and Trey Ideker. Using deep learning to model the hierarchical structure and function of a cell. *Nature methods*, 15(4):290–298, 2018.

Anders L Madsen, Frank Jensen, Antonio Salmerón, Helge Langseth, and Thomas D Nielsen. A parallel algorithm for bayesian network structure learning from large data sets. *Knowledge-Based Systems*, 117:46–55, 2017.

Daniel Marbach, Thomas Schaffter, Claudio Mattiussi, and Dario Floreano. Generating realistic in silico gene networks for performance assessment of reverse engineering methods. *Journal of computational biology*, 16(2):229–239, 2009.

Adam A Margolin, Ilya Nemenman, Katia Basso, Chris Wiggins, Gustavo Stolovitzky, Riccardo Dalla Favera, and Andrea Califano. Aracne: an algorithm for the reconstruction of gene regulatory networks in a mammalian cellular context. In *BMC bioinformatics*, volume 7, pages 1–15. BioMed Central, 2006.

Sara Mohammad-Taheri, Robert Ness, Jeremy Zucker, and Olga Vitek. Do-calculus enables causal reasoning with latent variable models. *arXiv preprint arXiv:2102.06626*, 2021.

Kevin P Murphy. Active learning of causal bayes net structure. 2001.

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

Robert Osazuwa Ness, Karen Sachs, Parag Mallick, and Olga Vitek. A bayesian active learning experimental design for inferring signaling networks. In *International Conference on Research in Computational Molecular Biology*, pages 134–156. Springer, 2017.

Marcus Nguyen, S Wesley Long, Patrick F McDermott, Randall J Olsen, Robert Olson, Rick L Stevens, Gregory H Tyson, Shaohua Zhao, and James J Davis. Using machine learning to predict antimicrobial mics and associated genomic features for nontyphoidal salmonella. *Journal of clinical microbiology*, 57(2):e01260–18, 2019.

Judea Pearl. From bayesian networks to causal networks. In *Mathematical models for handling partial knowledge in artificial intelligence*, pages 157–182. Springer, 1995a.

Judea Pearl. Causal diagrams for empirical research. *Biometrika*, 82(4):669–688, 1995b.

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

Massimo Pigliucci. Genotype–phenotype mapping and the end of the 'genes as blueprint'metaphor. *Philosophical Transactions of the Royal Society B: Biological Sciences*, 365(1540):557–566, 2010.

Andrea Pinna, Nicola Soranzo, and Alberto De La Fuente. From knockouts to networks: establishing direct cause-effect relationships through graph analysis. *PloS one*, 5(10): e12912, 2010.

Andrea Rau, Florence Jaffrézic, and Grégory Nuel. Joint estimation of causal effects from observational and intervention gene expression data. *BMC systems biology*, 7(1):1–12, 2013.

Marylyn D Ritchie, Emily R Holzinger, Ruowang Li, Sarah A Pendergrass, and Dokyoon Kim. Methods of integrating data to uncover genotype–phenotype interactions. *Nature Reviews Genetics*, 16(2):85–97, 2015.

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.

John W Santerre, James J Davis, Fangfang Xia, and Rick Stevens. Machine learning for antimicrobial resistance. *arXiv preprint arXiv:1607.01224*, 2016.

Alberto Santos-Zavaleta, Heladia Salgado, Socorro Gama-Castro, Mishael Sánchez-Pérez, Laura Gómez-Romero, Daniela Ledezma-Tejeida, Jair Santiago García-Sotelo, Kevin Alquicira-Hernández, Luis José Muñiz-Rascado, Pablo Peña-Loredo, et al. Regulondb v 10.5: tackling challenges to unify classic and high throughput knowledge of gene regulation in e. coli k-12. *Nucleic acids research*, 47(D1):D212–D220, 2019.

Thomas Schaffter, Daniel Marbach, and Dario Floreano. Genenetweaver: in silico benchmark generation and performance profiling of network inference methods. *Bioinformatics*, 27 (16):2263–2270, 2011.

Bernhard Schölkopf, Francesco Locatello, Stefan Bauer, Nan Rosemary Ke, Nal Kalchbrenner, Anirudh Goyal, and Yoshua Bengio. Towards causal representation learning. *CoRR*, abs/2102.11107, 2021. URL `https://arxiv.org/abs/2102.11107`.

Burr Settles. Active learning literature survey. 2009.

P. Spirtes, C. Glymour, and R. Scheines. *Causation, Prediction, and Search*. MIT press, 2nd edition, 2000.

Gustavo Stolovitzky, DON Monroe, and Andrea Califano. Dialogue on reverse-engineering assessment and methods: the dream of high-throughput pathway inference. *Annals of the New York Academy of Sciences*, 1115(1):1–22, 2007.

Yuriy Sverchkov and Mark Craven. A review of active learning approaches to experimental design for uncovering biological networks. *PLoS computational biology*, 13(6):e1005466, 2017.

Daniel Yasumasa Takahashi, João Ricardo Sato, Carlos Eduardo Ferreira, and André Fujita. Discriminating different classes of biological networks by analyzing the graphs spectra distribution. *PloS one*, 7(12):e49949, 2012.

Simon Tong and Daphne Koller. Active learning for structure in bayesian networks. In *International joint conference on artificial intelligence*, volume 17, pages 863–869. Citeseer, 2001.

Matthew J Vowels, Necati Cihan Camgoz, and Richard Bowden. D'ya like dags? a survey on structure learning and causal discovery. *ACM Computing Surveys (CSUR)*, 2021.

Yuhao Wang, Liam Solus, Karren Yang, and Caroline Uhler. Permutation-based causal inference algorithms with interventions. *Advances in Neural Information Processing Systems*, 30, 2017.

Duncan J Watts and Steven H Strogatz. Collective dynamics of 'small-world'networks. *nature*, 393(6684):440–442, 1998.

Alireza Yazdani, Lu Lu, Maziar Raissi, and George Em Karniadakis. Systems biology informed deep learning for inferring parameters and hidden dynamics. *PLoS computational biology*, 16(11):e1007575, 2020.

Kevin Y Yip, Roger P Alexander, Koon-Kiu Yan, and Mark Gerstein. Improved reconstruction of in silico gene regulatory networks by integrating knockout and perturbation data. *PloS one*, 5(1):e8121, 2010.

Behrooz Zarebavani, Foad Jafarinejad, Matin Hashemi, and Saber Salehkaleybar. cupc: Cuda-based parallel pc algorithm for causal structure learning on gpu. *IEEE Transactions on Parallel and Distributed Systems*, 31(3):530–542, 2019.