ETH
Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zurich

D MATH

Michael Vollenweider

Benchmarking Framework for Gene Regulatory Network Inference Methods

Bachelor Thesis

Seminar for Statistics
Swiss Federal Institute of Technology (ETH) Zurich

Supervision

Magali Champion

July 2022

Contents

Abstract

Gene regulatory networks (GRNs) describe sets of interactions between genetic materials that govern gene expression. A robust understanding of such interactions elucidates how cells adapt, develop, and respond to environmental stimuli, which significantly aids the investigation of multiple biological and biomedical problems. Via GRN inference methods, the abstract graphical representation of GRNs can be approximated using various data sources, which have become increasingly abundant thanks to technological advances within the last two decades. While great effort has gone into studying GRN inference and considerable progress has been made, the problem is far from being solved in its entirety. To engender the effective development of methods, efficient comparison to prevailing state-of-the-art algorithms is crucial. For that purpose, this Bachelor's thesis adapts and extends existing work by Pratapa et al. (2020) to provide a reproducible benchmarking framework for GRN inference on bulk gene expression data (`https://github.com/michavol/benchmarking_GRN_inference`, (Vollenweider, 2022)). It begins with an introductory overview of available data types with a conceptual classification and technical description of a few selected methods, continues with the explanation of multiple evaluation metrics considered by the benchmarking framework, and concludes with a demonstration of how corresponding results may be employed to compare the performance and viability of different algorithms.

Chapter 1

Introduction

This section aims to elucidate the background needed for understanding the problem of gene regulatory network (GRN) inference and its implications. First, a biological description of GRNs is given, then the mathematical modeling of biological networks is discussed, and lastly, the goal and structure of this thesis are further outlined.

1.1 Gene Regulatory Networks (GRNs)

The discovery of deoxyribonucleic acid (DNA) engendered a deeper understanding of how biological systems function at a molecular level. It is heritable information that explains the vast diversity of life with only four nucleotides as its building blocks; adenine (A), guanine (G), cytosine (C), and thymine (T). Combinations of specific pairs of the four letters (A with T, and G with C) form a double-stranded helix that provides genetic instructions for the functioning, development, reproduction, and growth of all discovered organisms and many viruses. The process in which the information in DNA is dynamically read is called gene expression and is constituted of two steps. First, via transcription, a region of DNA is used to create an RNA molecule, a single-stranded polymer of identical nucleotides as in DNA but with T replaced by uracil (U). Any such region which encodes the synthesis of an RNA molecule is called a gene. Then, via translation, this RNA molecule is utilized to synthesize proteins, large macromolecules made from chains of amino acids, which are essential for many biological functions. Because this process is fundamental in the field of Biology, it is coined the central dogma of molecular biology (Crick, 1970). It needs to be noted that not all RNAs are necessarily translated, nor are all DNA molecules transcribed. For instance, only approximately 2% of the human genome codes for proteins (Carneiro, 2013). For better adaptation, development and responsiveness to environmental stimuli, organisms have evolved to regulate the expression of genes (increase or decrease the production of their products) via a wide range of mechanisms. The associated interaction structures are collectively called gene regulatory networks (GRN). To better understand how GRNs may be modeled, the mechanisms of gene regulation are investigated further, then the idea of GRN inference is introduced, and lastly, it is explained how already identified GRNs can be applied in research.

1.1.1 Mechanisms of Gene Regulation

Here, only a brief description of key points of gene expression is given. For more detailed information, the reader is referred to the comprehensive literature on the matter (Uzman, 2002; Blakely, 2004). For transcription, the enzyme RNA polymerase (RNAP) attaches near a gene on a DNA molecule, unzips the double-stranded helix, and creates an RNA copy of the gene, so-called messenger RNA (mRNA). The frequency at which RNAPs are recruited to perform transcription is mainly regulated by transcription factor (TF) proteins and epigenetic modifications. TF proteins bind to specific DNA regions to either promote (as an activator) or block (as a repressor) the recruitment of RNAPs. They are gene products themselves, and their effect can therefore be interpreted as

indirect gene-to-gene interaction. Epigenetic modifications are chemical changes of the DNA itself that can affect the accessibility to transcriptional machinery. Also translation is tightly regulated; proteins and RNAs bind to mRNA targets to block or promote their activity. It is believed that this kind of post-transcriptional regulation is as prevalent as transcriptional controls (Hogan et al., 2008). This means that while a gene may not influence the mRNA production of another gene, it may still significantly impact its protein expression, which must be kept in mind when working with various gene expression data.

1.1.2 GRN Inference

The previous discussion established that transcription and translation are complex processes that involve interactions of several different molecular players. Accurately measuring these interactions is infeasible within living cells. However, measuring the abundance of the system's components, such as mRNA and protein levels, has been made possible through technological advances within the last two decades (Huynh-Thu and Sanguinetti, 2018). Especially measuring mRNA concentrations becomes increasingly scalable and affordable (Lowe et al., 2017). Hence, significant efforts have been made to infer graphical representations of GRNs from these measurements. A statistical method that performs such inference is called a GRN inference method. From now on also the graphical abstraction of a GRN will be referred to as a GRN. In what ways GRNs can be predicted from which kinds of data will be investigated later in this thesis.

1.1.3 Usage of GRNs

After obtaining a GRN via some inference method, it can be used in several ways to help solve different biological and biomedical problems. Following Emmert-Streib et al. (2014) some examples are discussed:

- **Causal Map of Molecular Interactions**: One can use GRNs to derive new hypotheses about molecular interactions, which can then be analyzed experimentally via ChIP-chip or gene expression experiments. ChIP-chip experiments rely on chromatin immunoprecipitation (ChIP) and DNA microarray (chip) technology to measure interactions between proteins and DNA in vivo, that is, within a living organism or cell (Aparicio et al., 2004). As inference methods usually rank predicted interactions by importance or statistical significance, GRNs are very effective at narrowing down the number of potential interactions.

- **Experimental Design and Perturbation Experiments**: GRNs can also be employed to aid the experimental design of new experiments. For instance, when an experiment generates observational data by measuring the system's state, the signal about specific pathways (series of interactions among molecules, here genes) may not be strong enough for statistical inference. The prior knowledge of the underlying network can be used to design perturbation or intervention experiments that enhance the signal strength.

- **Networks as Biomarkers**: GRNs can also be used for biomarker-based diagnostics, prediction, and prognosis. Instead of individual genes as markers, network-based biomarkers are considered, which can be seen as statistical measures that explicitly capture the interaction structure between genes.

- **Comparative network analysis**: Once many GRNs from different diseases and physiological conditions are available, one can compare these networks statistically to understand such phenotypes better.

- **Network Medicine and Drug Design**: Lastly, the design of drugs can significantly benefit from information encoded in GRNs. Network Medicine employs GRNs to identify plausible drug targets within a signaling pathway relevant for some conditions. The systematic approach reduces the required resources for initial drug targeting and questions the cause at the molecular level.

1.2 Modelling of Biological Networks

To effectively work with biological networks computationally, one needs to find a model that is complex enough to capture essential properties of theirs while also simple enough such that it remains convenient to work with mathematically. The following paragraphs elucidate some mathematical considerations, such as the graphical representation of networks, their interpretation, and some important associated concepts; causality, high-dimensionality, and sparsity.

1.2.1 Graphical Representation

GRNs may be represented by graphs, where vertices encode different genes and edges their pairwise interactions. Depending on the information that needs to be contained in the representation, one can choose from multiple types of graphs. Here the focus lies on the structural properties of different graphs, not their interpretations, which will be further discussed in later sections. The definitions closely follow the book "Introduction to Graph Theory" (Trudeau, 1994).

- **Undirected Graph**: An undirected simple graph is the ordered pair $G = (V, E)$, where V is a set of vertices and $E \subseteq \{\{x, y\} | x, y \in V, \text{ and } x \neq y\}$ a set of unordered pairs of vertices, the edges. When loops are also allowed, that is $E \subseteq \{\{x, y\} | x, y \in V\}$, it can be called an undirected simple graph permitting loops. The number of vertices (order) is denoted by $|V|$ and the number of edges (size) by $|E|$. When used to model a biological network, such a graph does not contain any information about the type or direction of interaction between genes. In literature, such representation of a GRN is also called a Gene Co-Expression network (van Dam et al., 2018).
- **Directed Graph**: For directed graphs, edges also have an orientation and can be represented by ordered pairs of vertices. A directed graph that does not contain any cycles is referred to as a directed acyclic graph (DAG). These graphs can describe the direction of an interaction between two genes but not its regulatory function, that is, whether it is activation or repression.
- **Partially Directed Graphs**: Partially directed graphs (or mixed graphs) may contain directed and undirected edges. They can be represented by the ordered triple $G = (V, E, A)$, where V has the vertices, E the undirected edges, and A the directed edges. A mixed graph without cycles is called a partially directed acyclic graph (PDAG). This kind of graph can be helpful for GRN inference if the direction of some interactions is known but others are not.
- **Weighted Graphs**: The above graphs can be extended with edge weights, a number assigned to each edge. In the biological context, weights on edges can be employed to encode different types and strengths of interaction. A negative edge weight for an edge between genes g_1 and g_2 may indicate that g_1 is a repressor of g_2, for instance. Conversely, a positive weight may imply that g_1 activates g_2. Furthermore, the absolute value of the edge weight can be used to express the importance of that interaction in the network.

Graphs of GRNs Predicted by Methods

All methods introduced in Section Chapter 3 are implemented in such a way that they output either an undirected or partially directed graph with non-negative edge weights. As they do not infer the type of interaction (activation vs. repression), there is no need for signed edge weights. The greater the edge weight, the more important a method considers the edge for describing the underlying data. One can control how many edges the predicted GRN ought to contain by thresholding its value.

1.2.2 Interpretation of Graphs

For graph topologies to become useful, the interpretation of edges, vertices, and weights must be further specified. Their biological meaning ought to be captured by mathematical models as

accurately as possible. Multiple approaches to this give rise to several methods for GRN inference. They will be conceptually structured and discussed in detail in Chapter 3. The following paragraphs discuss some noteworthy concepts naturally associated with GRN inference.

- **Causality**: When working with interpretations of edges, it is salient to distinguish between associational and causal concepts (Pearl, 2010). Associational concepts can be characterized by a joint distribution of observed variables and do not hold information about cause and effect. Examples include correlation, regression, dependence, and conditional probability. Contrastingly, causal concepts cannot be defined from the distribution alone as they deal with changing conditions induced by external interventions, which allows them to explain cause and effect. When interpreting directed edges of predicted networks, it is essential to keep this in mind; while they can encode causality, they may just as well represent an associational concept such as a conditional probability. All methods except for the PC algorithm, described in Chapter 3 are based on associational concepts.

- **High Dimensionality** arises as gene expression data is often recorded for several thousand genes but only for tens to hundreds of samples (Linde et al., 2015). This means that the problem of GRN inference is strongly under-determined ('curse of dimensionality') and that generally, no unique solution exists, which all network inference approaches need to take into account.

- **Sparsity** refers to the fact that genes in a GRN are usually connected to a few other genes only; the network size is not much greater than its order. For example, GRNs from the well-studied model organisms Escherichia coli, yeast, Arabidopsis, and Drosophila all show a mean of 1.5-2 transcriptional regulators per gene (Leclerc, 2008). It is believed that sparse networks are favored by natural selection as the network topology is more flexible and free to evolve. It is essential to know that there is more than one way to control the sparsity in a predicted graph. Either method-specific sparsity parameters can be varied to adjust how many edges will be assigned non-zero weights (parameters ρ_1 and ρ_2 for example for GLASSO discussed in Subsection 3.3.1) or already predicted weighted edges can be post-processed by thresholding the weights. The latter allows applying the same thresholding technique to predictions from diverse methods as it is independent of their model parameters and only relies on their ability to produce a weighted graph. This is why the parameters of the methods are chosen in such a way that rather non-sparse GRNs are predicted which can then be sparsified by thresholding the edge weights, which is especially important for a fair evaluation of their performance discussed in Section 4.2.

1.3 Scope and Goal

This thesis aims to achieve three objectives: 1. Give an introductory overview of available data types with a conceptual classification and technical description of methods. 2. Adapt and extend existing work to provide a reproducible benchmarking framework for GRN inference on bulk gene expression data (`https://github.com/michavol/benchmarking_GRN_inference`, (Vollenweider, 2022)); the meaning of 'bulk' will be elucidated in the next chapter. 3. Implement multiple methods and demonstrate how the evaluation metrics help analyze results and compare performance. The featured algorithms are chosen in such way that each conceptual category discussed in Chapter 3 is represented at least once and should not be seen as an extensive comparison of state-of-the-art methods. The data sources are picked based on popularity in GRN inference literature and availability of data simulators. It must be noted that as GRN inference is a quickly evolving and vast area of research, there is much more to investigate than the scope of this thesis allows.

Chapter 2

Types of Gene Expression Data

As for any inference problem, acquiring a deep understanding of the underlying data is salient. The intrinsic complexity of living organisms gives rise to multiple diverse sources of information and various ways to obtain data from them. With the steady increase in computational resources and swift innovation in the field of biotechnology, methods for making measurements of the genome (genetic information), proteome (complete set of proteins), and metabolome (complete set of small-molecule chemicals) are evolving rapidly. This chapter aims to outline what kinds of data can be leveraged for GRN inference in general, why data may need to be simulated, and what data is used for benchmarking the methods implemented for this thesis.

2.1 Data Types

As mentioned in Chapter 1, there are several biochemical processes one can measure to extract information from a sample. One technically feasible way to evaluate how much a gene is being expressed is to count how often it is transcribed corresponding mRNA (Lowe et al., 2017). This measurement is post-processed and represented as a numerical value, the gene expression value. It is important to note that mRNA expression does not necessarily correlate perfectly with protein expression. This is due to the degradation of mRNA and protein products (Belle et al., 2006). Furthermore, as mentioned in the Introduction, some mRNAs are not translated into proteins at all. Nevertheless, it gives a good approximation of how active a gene is. When measuring the gene expression values for p genes and n samples, this gives the data matrix $\mathbf{D} \in \mathbb{R}^{n \times p}$, where the row $(\mathbf{D})_{i,:}$ describes the gene expression values of all genes in the i-th measurement and the column $(\mathbf{D})_{:,k}$ the gene expression values of n measurements for gene g_k. While genotype information, i.e., DNA sequences or metabolome measurements, can be exploited for network inference (de la Fuente, 2013), this thesis will focus on the more commonly used gene expression values only. This section outlines different types of gene expression datasets, including bulk, single-cell, time-series, and perturbation data.

2.1.1 Bulk vs. Single-Cell Data

No biological cell is identical to another. A cell's location and its alterations in the genome are unique. However, until a few years ago, most sequencing techniques were applied to tissue samples or cell populations. This means that the heterogeneity of cells is obscured by averaging. The resulting measurements are sometimes referred to as bulk data. In contrast, single-cell methods measure the expression of genes in individual cells. This allows them to dissect the heterogeneity of cells (sin, 2014). This increased level of detail can be highly beneficial for inferring GRNs. However, it also comes with increased complexity, labor intensity, and costs (Navarro, 2021).

2.1.2 Time-Series and Perturbation Data

For time-series data, the gene expression of a sample is measured repeatedly at different points in time. This engenders several methods to incorporate temporal information (Navarro, 2021). When obtaining perturbation data, the sample is perturbed in some way. This can include chemical changes in the environment or gene editing. One out of many examples are gene knockouts, where a target gene is deactivated (Hall et al., 2009). Double, triple or quadruple knockouts can be used to study the effect of making multiple genes inoperative simultaneously. Of course, time-series and perturbation data can also be combined (Omranian et al., 2016).

2.2 Simulated vs. Real Data

Ultimately, biologists want to apply GRN inference methods to real data. Hence, developing methods that work well on realistic data is desirable. However, to quantify "work well", it is crucial to know the ground truth network. While many pathways have been experimentally identified for well-known model organisms and made publicly available (KEGG: `http://www.kegg.jp/` or `http://www.genome.jp/kegg/`), they only provide small parts of large unknown networks (Kanehisa et al., 2017). Furthermore, experimental verification of pathways is laborious and not scalable. Therefore, inference methods are often benchmarked using simulated gene expression data for which the underlying network is known. This allows for objective performance evaluation and comparison between different results. However, it is challenging to accurately simulate such complex biological behavior. It needs to be ensured that the assumptions made for the simulation are appropriate and that the resulting data is approximately representative of real systems.

2.3 Data for Benchmarking

For benchmarking in this thesis, three different sources of data are considered. Two gene expression simulators, GeneNetWeaver (Schaffter et al., 2011) and SERGIO (Dibaeinia and Sinha, 2020), and real biological data from The Cancer Genome Atlas (TCGA) (`https://www.cancer.gov/about-nci/organization/ccg/research/structural-genomics/tcga`).

2.3.1 GeneNetWeaver Simulator

The GeneNetWeaver (`http://gnw.sourceforge.net/`) is a software that was employed for generating benchmarking datasets for the Dialogue for Reverse Engineering Assessments and Methods (DREAM) challenges. There were three DREAM challenges for which gene regulatory networks needed to be inferred; DREAM3, DREAM4, and DREAM5. Their goal was to explore the reverse engineering of GRNs from different kinds of datasets. Participating teams were given gene expression data without the corresponding ground truth information, submitted their predicted networks and were then ranked based on some performance criteria. After the challenges were completed, multiple papers used the generated datasets to compare algorithms to the state-of-the-art (Huynh-Thu et al., 2010; Moerman et al., 2019; Schaffter et al., 2011). The following paragraphs describe how GeneNetWeaver is able to simulate gene expression data.

Extraction of Real Modules

Biological networks have specific graph-theoretic properties regarding their modularity, density, and structure. Hence, one cannot generate a random graph and expect it to behave similarly to a GRN (Van den Bulcke et al., 2006). Therefore, GeneNetWeaver starts by deriving modules from known in vivo network structures. This way, the graph properties may be inferred and used in the simulation.

Dynamical Model

Having created an underlying network structure, a dynamical, thermodynamics-based approach is used for modeling the biochemical processes of transcription and translation. Garcia et al. (2011) describe the noiseless model of these processes as such:

$$F_i^{RNA}(\mathbf{x}, \mathbf{y}) = \frac{dx_i}{dt} = m_i * f_i(\mathbf{y}) - \lambda_i^{RNA} * x_i, \tag{2.1}$$

$$F_i^{Prot}(\mathbf{x}, \mathbf{y}) = \frac{dy_i}{dt} = r_i * x_i - \lambda_i^{Prot} * y_i, \tag{2.2}$$

where F_i^{RNA} and F_i^{Prot} denote the rate of change of mRNA and protein concentration of gene g_i, respectively. The state variables $\mathbf{x}$ and $\mathbf{y}$ describe the mRNA and protein concentration levels. For each gene g_i, m_i is the maximum transcription rate, λ_i^{RNA} and λ_i^{Prot} the degradation rates of mRNA and protein molecules, and r_i the translation rate. $f_i(\cdot)$ evaluates the relative activation of the gene (Marbach et al., 2010). The molecular noise due to thermal fluctuations and the stochasticity of gene expression (Becskei and Serrano, 2000) is incorporated into the model via the chemical Langevin equation (CLE) (Gillespie, 2000). Moreover, realistic gene expression data does not only contain molecular but also technical noise. The degree thereof depends on the technology that was used for recording it. Different models of experimental noise were implemented (et al., 2005).

Experiments

GeneNetWeaver can simulate several genetic experiments. These include knockouts, knockdowns, dual knockouts, multifactorial perturbations, and custom perturbations. For benchmarking in this thesis, the focus lies on multifactorial perturbation experiments. There the basal activation of all genes is simultaneously and randomly perturbed. The aim is to imitate the expression of multiple organisms of the same type (Schaffter et al., 2011). In total, ten such perturbation datasets $\mathbf{D} \in \mathbb{R}^{n \times p}$ from different ground truth networks are used for evaluation. Five of which were simulated with $n = 5$, $p = 5$. The others with $n = 100$, $p = 100$. These are the exact same datasets used in the DREAM4 competition (Marbach et al., 2009).

2.3.2 SERGIO Simulator

A more recent data addition to the collection of gene expression simulators is SERGIO (**S**ingle-cell **E**xp**R**ession of **G**enes **I**n silic**O**). Unlike GeneNetWeaver which can only generate bulk data, it can also simulate single-cell transcriptomics data with (dynamic) and without (steady-state) differentiation of cells. Differentiation refers to cells becoming more specific types of cells due to permanent changes in gene expression (Rosenberg and Rosenberg, 2012). For benchmarking, only steady-state data is used as the selected methods are incapable of exploiting dynamic information.

Method

In contrast to the GeneNetWeaver, SERGIO (`https://github.com/PayamDiba/SERGIO`) does not explicitly model the translation processes and does not rely on a thermodynamics-based model. SERGIO also assumes the existence of a set of master regulator genes. Their gene expression cannot be influenced by other genes, i.e. their nodes in the network have outgoing edges only, and is controlled by constant production and decay rates. All other genes have a production rate which depends on their respective regulators and a constant decay rate. In the following, $[g_i]$ denotes the mRNA concentration produced by gene g_i and $R_{g_i}(\cdot)$ a regulatory function with the concentration of regulators as its arguments. For master regulators, $R_{g_i}(\cdot)$ is constant. Furthermore, for each gene, λ_{g_i} is the decay constant and $\phi_{g_i}(\cdot)$ stochastic molecular noise dependent on the concentration of gene g_i and its regulators. When $\mathcal{G} = \{g_1, g_2, ...\}$ is the set of genes, let $\mathcal{M} = \{m_1, m_2, ...\} \subseteq \mathcal{G}$ be

the set of master regulators, $\mathcal{N} = \mathcal{G} \setminus \mathcal{M} = \{n_1, n_2, ...\}$ all other genes and $\mathcal{R}_n \subseteq \mathcal{G} = \{r_{n1}, r_{n2}, ...\}$ the set of regulators of gene g_n. Then, the following equations model the underlying dynamics:

$$\frac{d[m]}{dt} = R_m - \lambda_m[m] + \phi_m([m]); \text{ for all } m \in \mathcal{M}, \tag{2.3}$$

$$\frac{d[n]}{dt} = \sum_{r \in \mathcal{R}_n} R_n([r]) - \lambda_n[n] + \phi_n([n], [r_{n1}], [r_{n2}], ...]); \text{ for all } n \in \mathcal{N}. \tag{2.4}$$

Dibaeinia and Sinha (2020) provide further details on the stochastic noise term and how to solve the stochastic differential equations. To imitate different cell types, SERGIO runs separate simulations using different constant production rates for the master regulators $\mathcal{M}$. Then for each cell type, simulated gene expression data can be extracted by randomly sampling from the steady-state region. As for the GeneNetWeaver, technical noise such as outlier genes, library size effects, and dropouts can also be incorporated (Zappia et al., 2017).

Datasets

For benchmarking, datasets with simulation parameter from the Table 2.1 below are used. For each, the simulation was repeated 15 times. Furthermore, the single-cell data can be interpreted as bulk data by averaging all gene expressions of a single-cell type. This gives the following 90 datasets:

Dataset ID	**Species**	**#Cell Types**	**#Cells**	$\lvert\mathcal{G}\rvert$	$\lvert\mathcal{M}\rvert$	**#Edges**
DS-SERGIO-1:1-15	E. coli	9	2,700	100	10	258
DS-SERGIO-2:1-15	yeast	9	2,700	400	37	1,155
DS-SERGIO-3:1-15	E. coli	9	2,700	1200	127	2,713
DS-SERGIO-1-bulk:1-15	E. coli	9	9	100	10	258
DS-SERGIO-2-bulk:1-15	yeast	9	9	400	37	1,155
DS-SERGIO-3-bulk:1-15	E. coli	9	9	1200	127	2,713

Table 2.1: Parameters for SERGIO datasets.

2.3.3 TCGA Data

Ultimately, the goal remains to apply GRN inference methods to real biological data. To analyse how well the selected methods can deal with such complex data, a dataset from The Cancer Genome Atlas (TCGA) is considered (`https://www.cancer.gov/about-nci/organization/ccg/research/structural-genomics/tcga`).

The Cancer Genome Atlas

On 26-27 October 2012, in Santa Cruz, California, TCGA launched the Pan-Cancer analysis project. The aim is to "generate, quality control, merge, analyze and interpret molecular profiles at the DNA, RNA, protein and epigenetic levels" for multiple tumor types and their subtypes (Weinstein, 2013). For data generation, tumor and germline samples are processed by the Biospecimen Core Resource to obtain purified DNA, RNA, and protein preparations. Then, these are profiled at Genome Characterization Centers and Genome Sequencing Centers. The updated data is then stored at the TCGA Data Coordinating Center. The resulting datasets help analyze the relevance of molecular changes in cancer and relate various types.

KIPAN Cohort

The KIPAN cohort is a combination of the Kidney Renal Papillary Cell Carcinoma (KIRP), Kidney Renal Clear Cell Carcinoma (KIRC), and Kidney Chromophobe (KICH) cohorts (`https://portal.gdc.cancer.gov/projects`). For benchmarking, a preprocessed RNAseq dataset with $p = 15,172$ and $n = 887$ was used.

2.3.4 Overview of Datasets

The following table provides an overview of all datasets considered for benchmarking in this thesis, where p denotes the number of genes, n the number of samples and k the number of edges in the ground truth network:

Source	Names	Type	p	n	k	Subsection
DREAM4	DS-D1:1-5	Sim. Multifactorial Bulk	10	10	15	2.3.1
	DS-D2:1-5	Sim. Multifactorial Bulk	100	100	250	2.3.1
SERGIO	DS-S1:1-15	Sim. Static Single-Cell	100	2700	250	2.3.2
	DS-S2:1-15	Sim. Static Single-Cell	400	2700	1155	2.3.2
	DS-S3:1-15	Sim. Static Single-Cell	1200	2700	2713	2.3.2
	DS-S1-bulk:1-15	Sim. Bulk	100	9	250	2.3.2
	DS-S2-bulk:1-15	Sim. Bulk	400	9	1155	2.3.2
	DS-S3-bulk:1-15	Sim. Bulk	1200	9	2713	2.3.2
TCGA	DS-TCGA-1	Real Cancer Data	15,172	887	?	2.3.3

Table 2.2: Overview of datasets.
(**p** = number of genes, **n** = number of samples, **k** = number of edges, Sim.=Simulated)

Chapter 3

GRN Inference Methods

With the swift advancement of high-throughput measurement techniques and a steady increase in computational resources, myriad ways to infer GRNs emerge. First, this chapter attempts to conceptually structure different approaches to tackling the problem. Then, a more technical description of an aforethought selection of methods is given. The table below offers a high-level characterization of all methods from this selection.

Type	Name	Concept	Graph	Parallel	Subsection
Data-Driven	CORR	Correlation	UG	No	3.2.1
	ARACNE	Mutual Information	UG	No	3.2.2
	GENIE3	Regression: Random Forest	PDG	Yes	3.2.3
	GRNBOOST2	Regression: Boosting	PDG	Yes	3.2.3
Probabilistic	GLASSO	Gaussian Graphical Model	UG	No	3.3.1
	PPCOR	Gaussian Graphical Model	UG	No	3.3.1
	GENENET	Gaussian Graphical Model	PDG	No	3.3.1
	ORDER_MCMC	Bayesian Networks: MCMC	PDG	Yes	3.3.2
	PC	Causal Model	PDG	Yes	3.3.3

Table 3.1: Overview of methods. All graphs have non-negative edge weights. (U=Undirected, D=Directed, P=Partially, G=Graph)

Except for CORR, which simply computes a correlation, the algorithms the implementations rely on are well-known and have been applied to network reconstruction problems. With this selection the aim is to exemplify diverse categories of methods and not to provide a comprehensive comparison of all existing methods. The evaluation and comparison of the methods follow in subsequent chapters.

3.1 Conceptual Landscape

Coherently classifying various types of GRN inference tools is challenging as the suitability of a method strongly depends on the given context. For instance, depending on the use case, either supervised or unsupervised learning of networks may be preferred. Furthermore, certain concepts only apply to certain kinds of data. Nevertheless, some fundamental ideas are worth taking a closer look at, regardless of the situation.

3.1.1 Overview

As it is done for many contemporary Machine Learning problems, Michailidis considers two categories for the high-level classification of GRN reconstruction methods (Michailidis, 2012); unsupervised and supervised inference. Unsupervised learning, also called de-novo inference when applied

to GRNs, assumes no prior knowledge about the edges of the network. In contrast, supervised learning uses information about the already partially known network. For example, when a well-known model organism is studied, specific pathways have been experimentally verified and can be incorporated when predicting the remaining network. A common approach is to start with the set of verified edges as training data and predict edge labels for unlabeled edges in their proximity. The predicted relationships can then be included in the training set to predict neighboring edges again. This is repeated until all edges are labeled (Bleakley et al., 2007). Another way of introducing domain knowledge is to combine statistical models with kinetic parameters of reaction mechanisms (Mourão et al., 2011). While this can improve prediction accuracy, it generally comes with significant scalability issues. The benchmarking framework for this thesis is developed for handling unsupervised learning algorithms.

3.1.2 Basic Concepts

Regardless of whether prior knowledge is used for network inference, GRN inference methods may be grouped into data-driven and probabilistic methods. Detailed descriptions of corresponding algorithms are given in Sections 3.2 and 3.3, respectively. When multiple models are aggregated to build a new one, it is called an ensemble method. The proposed categorization is intrinsically imperfect as many methods combine multiple concepts and cannot be assigned to a single category. Nevertheless, it helps to gain a better global understanding of the problem.

Data-driven methods construct a fully connected weighted graph directly from the data (Huynh-Thu and Sanguinetti, 2018). The weighted edges may be interpreted as gene-to-gene interactions and used to reconstruct a network using a threshold. There are various ways to obtain such weights. Three primary metrics are correlation, information-theoretic scores, and feature importance obtained via regression. Section 3.2 will take a closer look at such methods.

Probabilistic methods explicitly impose a probabilistic model on the data and reconstruct GRNs by optimizing some measure of fit or by employing Bayesian approaches. These include Gaussian graphical models (GGMs), Bayesian networks (BNs), and some causal models. Examples of such are discussed in more detail in Section 3.3.

Ensemble methods work by aggregating results of various techniques via some consensus methods. Bellot et al. differentiate between the homogeneous and heterogeneous scenarios (Bellot et al., 2019). In the former, inference algorithms that work on the same data type are combined. In the latter, results are obtained by aggregating predictions of methods that differ in what kind of data source they are applied to. One important homogeneous example of model averaging is bootstrap aggregating, also known as bagging (Efron, 1981). The idea is to improve stability and accuracy, reduce variance and avoid overfitting by averaging the predictions for N_{boot} randomly drawn replicate datasets from the original data. These sets have the same size as the original data from which they are uniformly sampled with replacement. Without parallelization, this multiplies the computational runtime by N_{boot}. Bagging is employed for several algorithms discussed in the following sections.

3.1.3 Data-Specific Concepts

Inference may also be classified by the type of data it is intended for. As described in Chapter 2, measurement techniques may produce, among others, bulk, single-cell, and time-series data. Sometimes the assumption that a single underlying network can explain all data is also inappropriate. This gives rise to multi-network models.

Bulk data contains less information than other richer types of data. Cellular heterogeneity is masked as it measures averaged expression across a population of cells. However, it is also the least expensive to obtain and the simplest in structure. This makes it convenient for demonstrating

the previously discussed basic concepts in GRN inference which are not data-specific. The methods from Table 3.1 can all be applied to this kind of data. Here, the datasets of type "Bulk" from Table 2.2 from Chapter 2.2 are considered as examples.

In contrast to bulk measurements, single-cell data offers more detailed information about individual cells and reveal cellular heterogeneity and subpopulation expression (Navarro, 2021). Several methods have been developed capable of exploiting this additional information. A few examples are PIDC (Chan et al., 2017), relying on partial information decomposition, SCNS (Woodhouse et al., 2018), via boolean network models, and SCENIC (et al., 2017), using random forests and motif analysis. The methods from Table 3.1 can also be applied to single-cell data. However, they are not designed to leverage the heterogeneity. Here, the datasets of type "Single-Cell" from Table 2.2 from Chapter 2 are used.

For time-series data, on a high-level, especially dynamic Bayesian and differential equation networks may be considered (Huynh-Thu and Sanguinetti, 2018). Dynamic Bayesian models characterize each time point as an individual Bayesian network. This allows them to resolve cycles, an intrinsic limitation of Bayesian modeling on static data. Another advantage is that rich literature on signal processing is available and may be repurposed for GRN inference. Differential equation networks model the underlying biochemical processes as differential equations and infer gene-to-gene interactions by solving them. The application of differential equations is well-studied and has been applied in various forms for GRN reconstruction (Fröhlich et al., 2017). The application of methods on time-series data goes beyond the scope of this thesis.

Huynh-Thu et al. consider two main categories of multi-network models (Huynh-Thu and Sanguinetti, 2018). The first assumes a scenario where data is generated under different but similar conditions. This requires the model to transfer information between various conditions as it can be assumed that the underlying network is somewhat similar. To achieve this, one can apply a shared diversity penalty (Niculescu-Mizil and Caruana, 2007; Chiquet et al., 2010) or a hierarchical Bayesian approach (Werhli and Husmeier, 2007; Penfold et al., 2012). In the second scenario, one assumes that the network varies over time. Generally, methods include finding time points for the network changes and identifying networks across periods. Also the application of these kinds of models are not further investigated in this thesis.

3.2 Data-Driven Methods

This section aims to characterize data-driven methods further; more precisely, methods that rely on correlation, information-theoretic scores, or regression. Specifically, for correlation networks, the self-implemented CORR algorithm is discussed. Regarding information theoretic scores, the ARACNE algorithm is elucidated (Margolin et al., 2006), and for regression-based methods, Genie3 (Huynh-Thu et al., 2010) and GRNBoost2 (Moerman et al., 2019) are looked at.

3.2.1 Correlation-Based Methods

One of the first concepts that come to mind when trying to model the association between variables is correlation. In statistics, correlation often refers explicitly to describing a linear relationship between variables in terms of the Pearson correlation coefficient 2008. Other types of correlation exist, such as the Spearman correlation, which describes how monotonically two variables relate to each other (Mohr et al., 2022) and the Kendall rank correlation coefficient, which measures the ordinal association between two quantities (Stepanov, 2015). Furthermore, partial correlation quantifies the degree of association between two random variables when removing the effect of a set of controlling random variables (Fisher, 1924). As this kind of correlation is also strongly linked to the concept of Gaussian Graphical Models (GGMs), it will only be further discussed in Section 3.3.

Sample Pearson Correlation Coefficient

One common metric to measure the degree of a linear relationship, given sample data, is the so-called sample Pearson correlation coefficient. Given two non-zero vectors $\mathbf{v}, \mathbf{w} \in \mathbb{R}^n$, their sample correlation coefficient $r_{\mathbf{vw}} \in [-1, 1]$ is defined by:

$$r_{\mathbf{vw}} := \frac{\sum_{i=1}^{n}(v_i - \bar{\mathbf{v}})(w_i - \bar{\mathbf{w}})}{\sqrt{\sum_{i=1}^{n}(v_i - \bar{\mathbf{v}})^2 \sum_{i=1}^{n}(w_i - \bar{\mathbf{w}})^2}}. \tag{3.1}$$

Here $\bar{\mathbf{v}}$ and $\bar{\mathbf{w}}$ denote the arithmetic means of the entries of $\mathbf{v} := (v_i)_{1 \leq i \leq n}$ and $\mathbf{w} := (w_i)_{1 \leq i \leq n}$. Under the assumption of an uncorrelated bivariate normal distribution for $\mathbf{v}$ and $\mathbf{w}$, the statistical significance of the obtained value can be evaluated by applying a t-test. This is especially important for low numbers of samples. The following t-statistic can be used to determine a corresponding p-value (Mohr et al., 2022):

$$t = \frac{r_{\mathbf{vw}} \times \sqrt{n-2}}{\sqrt{1 - r_{\mathbf{vw}}}}. \tag{3.2}$$

CORR Algorithm

The `CORR` algorithm implements an inference method by using the concepts above. As it relies on basic theory and is computationally efficient, it acts as a baseline for other methods. One can represent the gene expression data as a matrix $\mathbf{D} \in \mathbb{R}^{n \times p}$, where n is the number of measurements taken and p the count of genes. Computing the sample correlation coefficient between any two measurement vectors $(\mathbf{D})_{i,:}$, $(\mathbf{D})_{j,:}$ using Equation (3.1), results in a correlation matrix $\mathbf{C} \in [-1, 1]^{p \times p}$. One can then calculate the corresponding p-values via Equation (3.2). To account for multiple testing, a Holm correction is made (Holm, 1979). Having obtained correlation coefficients and corresponding p-values for all possible edges, there are different ways to use them for predicting a GRN. Naturally, one can use correlation coefficients as edge weights and the p-value as a sparsity parameter. Any edge with a correlation coefficient with a p-value smaller than the p-value threshold is contained in the GRN, others not. The smaller the the threshold, the sparser the predicted network. However, as discussed in Subsection 1.2.2, for comparing the performance of diverse methods it can be desirable to first produce non-sparse networks which can then be thresholded with the edge weights. This is why a p-value threshold of 1 is chosen for the implementation, which also omits the need for its computation. For implementing this procedure in R, the `cor` function and `corr.test` from the package `psych` are used (Revelle, 2022).

Advantages & Disadvantages

Huynh-Thu and Sanguinetti (2018) point out that one significant advantage of using correlation as a metric for network inference is its simplicity and efficiency. The asymptotic runtime scales quadratically with the number of genes and linearly with the number of samples. However, this simplicity comes at a price. Correlation cannot identify confounding variables or differentiate between direct and indirect relationships between variables. Using partial correlation instead can alleviate this issue and is only studied more closely in Subsection 3.3 due to its connection to Gaussian graphical models. Furthermore, non-linear relationships may be missed (Huynh-Thu and Sanguinetti, 2018). Lastly, correlation is a symmetric score and can therefore only provide undirected edges.

3.2.2 Information Theoretic Scores

Ideas from information theory have also been employed for obtaining edge weights. One important concept is mutual information, which measures the degree of mutual dependence between two

random variables. By thresholding the mutual information, edges may be selected for constructing a so-called relevance network (Butte and Kohane, 2000). This basic idea can then be enriched with other information theory and statistics concepts. ARACNE (Margolin et al., 2006), for instance, further processes the relevance network by using the data processing inequality (DPI) to eliminate a majority of indirect interactions. Other examples include the context likelihood of relatedness (CLR) (Faith et al., 2007) and MRNET (Meyer et al., 2007) algorithms.

Mutual Information

Mutual information between two random variables X and Y is defined as:

$$MI(X;Y) = H(X) + H(Y) - H(X,Y),$$

where $H(X)$, $H(Y)$ denote the marginal entropies of X and Y, and $H(X,Y)$ the joint entropy of X and Y. For discrete X and Y, they are given by:

$$H(X) = -\sum_{x\in\mathcal{X}} p(x)\log p(x),$$

$$H(Y) = -\sum_{y\in\mathcal{Y}} p(y)\log p(y),$$

$$H(X,Y) = -\sum_{x\in\mathcal{X},y\in\mathcal{Y}} p(x,y)\log p(x,y),$$

where $\mathcal{X} = \{x_1,...,x_n\}$ and $\mathcal{Y} = \{y_1,...,y_m\}$ are the sample spaces of X and Y, $p(x)$ and $p(y)$ the probabilities of outcomes x and y and $p(x,y)$ the joint probability of outcomes x and y. While entropy is infinite for continuous variables (Margolin et al., 2006) due to the infinite sample space, the mutual information remains well-defined. By substituting the entropy H with the differential entropy it can be determined via:

$$MI(X;Y) = \int_{\mathcal{Y}}\int_{\mathcal{X}} p(x,y)\log\left(\frac{p(x,y)}{p(x)\,p(y)}\right)dxdy. \tag{3.3}$$

Data Processing Inequality

The DPI is an extensively used information-theoretic concept. Informally, it states that information cannot be increased by post-processing of any kind (Nalewajski, 2012). Assuming that random variables X, Y and Z form the Markov chain $X \to Y \to Z$, Y contains all of the information about X for determining Z. Then, the inequality asserts that Z cannot hold any more information about X than Y does. In terms of mutual information, this can be expressed by:

$$MI(X;Y) \geq MI(X;Z). \tag{3.4}$$

ARACNE Algorithm

Based on ideas from Markov networks literature (Andersen, 1991), the joint probability distribution of the expressions of all genes, $P(\{g_i\}), i = 1,...,p$, can be expressed as:

$$P(\{g_i\}) = \frac{1}{c}e^{-H(\{g_i\})}, \tag{3.5}$$

where p describes the number of genes, c is a non-negative normalization factor and $H(\{g_i\})$ is the Hamiltonian describing the system's statistics. The ARACNE algorithm approximates the interaction structure with the following Hamiltonian (Margolin et al., 2006):

$$H(\{g_i\}) = \sum_{i}^{p}\phi_i(g_i) + \sum_{i,j}^{p}\phi_{i,j}(g_i,g_j), \tag{3.6}$$

where the first-order potentials ϕ_i describe the marginal probabilities, $P(g_i)$, which can be directly estimated from the gene expression data. The potentials $\phi_{i,j}$ correspond to the pairwise gene expression profile mutual information $MI(g_i; g_j)$ as in (3.3). For their estimation, a Gaussian Kernel is employed (Beirlant et al., 1997). By randomly reshuffling the data and using a Monte Carlo simulation for different kernel widths, as in Butte and Kohane (2000), a p-value can be calculated and used to estimate a mutual information threshold, I_0. Indirect relationships in the obtained relevance network are pruned using the DPI. This is done by checking possible gene triplets and pruning the edge with the smallest mutual information; for any triplet of genes (g_i, g_j, g_k), the weakest edge, say (g_j, g_k), is removed if:

$$MI(g_j; g_k) < \min\{MI(g_i; g_j), MI(g_i; g_k)\} - \epsilon, \tag{3.7}$$

where $\epsilon \in \mathbb{R}$ is a sparsity parameter. The smaller ϵ, the sparser the predicted GRN as more pairs of genes fulfill the inequality (3.7). To produce non-sparse GRNs, an ϵ of 0.2 was picked. ARACNE is implemented using the R package `minet` (Meyer et al., 2008).

Advantages & Disadvantages

The ARACNE algorithm is widely-used for several reasons. With a quadratic computational complexity in the number of samples and genes, it is only slightly inferior to correlation-based methods. This makes it possible to also use ARACNE for inferring complex mammalian cell networks (Margolin et al., 2006). Being able to identify indirect interactions can circumvent one of the major flaws of the CORR method. Another advantage is that the information-theoretic method is not limited to constructing trees but is also capable of reproducing cycles. However, ARACNE cannot be employed to infer causal information due to the symmetry of mutual information. Furthermore, especially for low numbers of samples, the estimation of mutual information may not be robust to the presence of noise in the data (Huynh-Thu and Sanguinetti, 2018).

3.2.3 Regression-Based Methods

Another popular approach to GRN reconstruction is using regression. One can employ penalized linear regression methods such as the LASSO (**L**east **A**bsolute **S**hrinkage and **S**election **O**perator) (Tibshirani, 2013) and all its derivatives, but also non-linear regression methods such as decision trees (Breiman, 2001). Specifically, GENIE3 (Huynh-Thu et al., 2010) and GRNBoost2 (Moerman et al., 2019) will be looked at. More recently, also neural nets have shown promising results for inference on single-cell data (LeCun et al., 2015; Fan and Ma, 2021; Shrivastava et al., 2022).

Regression for GRNs

The idea of regression is to infer a model f which can predict some continuous response variable Y, given m inputs $X^{(1)}, X^{(2)}, ..., X^{(m)}$. For learning f, n observations of the input-output pairs are used. In the context of GRNs, gene expression data can be represented as a matrix $\mathbf{D} \in \mathbb{R}^{n \times p}$, where n is the number of measurements taken and p the count of genes. A natural approach is to learn to predict the gene expression vector $(\mathbf{D})_{:,i}$ of each gene g_i by using expression measurements $(\mathbf{D})_{:,j}$ of all other genes g_j for $j \in \{1, 2, ..., p\} \setminus \{i\}$. Let the collection of these measurements be denoted by $(\mathbf{D})_{:,-i} \in \mathbb{R}^{n \times (p-1)}$. Hence, one needs to find suitable models f_i for $i = 1, ..., p$ such that:

$$(\mathbf{D})_{:,i} = f_i((\mathbf{D})_{:,-i}, \boldsymbol{\beta}_i) + \boldsymbol{e_i}, \tag{3.8}$$

where $\boldsymbol{\beta}_i$ are unknown model parameters and $\boldsymbol{e}_i \in \mathbb{R}^n$ a vector of error terms. This results in having to solve p regression problems. Some loss function is minimized for the training of regression problems. The learned parameters $\boldsymbol{\beta}_i$ can then be used to reconstruct the network. In the following paragraphs, the application of linear regression, concepts of decision trees, random forest and boosting, and two tree-based methods are described.

Linear Regression

For linear regression, f_i from Equation (3.8) becomes a matrix-vector multiplication:

$$f_i((\mathbf{D})_{:,-i}, \boldsymbol{\beta}_i) = (\mathbf{D})_{:,-i}\boldsymbol{\beta}_i. \tag{3.9}$$

In addition to the LASSO, examples of methods that can be applied include the Dantzig selector (Candes and Tao, 2007), ridge regression (Hoerl and Kennard, 1970), elastic net (Zou and Hastie, 2015), and extensions of the LASSO (Tibshirani, 2013; Omranian et al., 2016). With the LASSO, $\boldsymbol{\beta}_i$ from (3.9) can then be estimated via:

$$\boldsymbol{\beta}_i = \min_{\boldsymbol{\beta} \in \mathbb{R}^{p-1}} \left\{ \|(\mathbf{D})_{:,i} - (\mathbf{D})_{:,-i}\boldsymbol{\beta}\|_2^2 + \lambda_1 \|\boldsymbol{\beta}\|_1 \right\}, \tag{3.10}$$

where $\|\boldsymbol{\beta}\|_1$ describes theL_1-norm of $\boldsymbol{\beta}$, which is the sum of the absolute value of all its entries. The L_1 regularization makes the LASSO a sparse estimator as it forces regression coefficients towards zero (Tibshirani, 2013). This is especially useful for GRN inference as biological networks are known to be sparse. Furthermore, the problem from (3.10) is a convex optimization problem and can therefore be efficiently computed using the LARS algorithm for example (Bühlmann and van de Geer, 2011). As the entries of $\boldsymbol{\beta}_i$ are tightly linked to the concept of partial correlation and Gaussian Graphical Models, the LASSO will be further discussed in Section 3.3 (Waldorp and Marsman, 2020).

Decision Trees

Alternatively f_i from (3.8) can be described with decision tress. The idea is to split the data points with binary tests recursively. Each test is represented as a node in a binary tree and based on a single regressor variable, a gene. The goal of each test is to minimize the variance of the gene expression of the response gene in the resulting subsets of samples. Usually, such a test is a simple comparison between the input variable value and a threshold parameter. During training, these thresholds are learned for each node in the decision tree. Then, the tree leaves contain the predicted numerical value for the response variable.

Bootstrap Aggregating: Bagging

Hastie et al. (2013) point out that decision trees suffer from high variance and instability (Hastie et al., 2013). Because of the model's hierarchical nature, errors in the top split can propagate down to nodes below. One way to alleviate this issue is to use ensemble methods to reduce the variance. As discussed in Section 3.1.2, an example for such technique is bagging. In the context of decision trees, several trees are grown for different random samples of the training set. The final prediction of the response variable is the mean of all decision tree regressions. Conveniently, the internal bootstrapping allows easy estimation of feature importances. When, in addition to bagging, only a random subset of all features is used as potential splitting nodes, it becomes a random forest (Breiman, 2001). Parameters that need to be defined for the training of a random forest include N_{min}, the minimal number of samples that a leaf node must contain, K, the number of input variables chosen at random for each node of a tree, and T, the count of trees in the ensemble (Huynh-Thu et al., 2010).

GENIE3 Algorithm

GENIE3, developed by Huynh-Thu et al. (2010), uses random forests to tackle the regression problem from 3.8 (Huynh-Thu et al., 2010). The target gene's expression is normalized to have a unit variance in the training sample before learning the trees to avoid bias. Having learned the tree model, variable importances $w_{i,j} > 0, (i, j = 1, ..., p)$ are computed, where $w_{i,j}$ represents the degree of relationship between genes g_i and g_j and can be used as edge weights for GRN inference. Its computation relies on evaluating how much a node representing g_i reduces the variance of the response value of g_j. The resulting network contains directed edges because the weights do not

need to be symmetric. Moreover, because they are non-negative, only unweighted edges can be inferred.

When using GENIE3, the option exists to define a restricted set of candidate regulators. This is especially advisable if certain genes are already known to produce transcription factors; proteins which control the rate of transcription of other genes. Let n_{TF} be the number of candidate regulators. It also acts as a sparsity parameter, as allowing more potential regulators can lead to denser network topologies. K is usually chosen to be between $\sqrt{n_{TF}}$ and n_{TF}. For the other model parameters, the authors report that $T = 500$ already leads to good performance and that n_{min} only has a small impact on the results. $n_{min} = 1$ is ideal for performance but can be increased to save computational time.

For benchmarking, GENIE3 is implemented using the R package `GENIE3` (Huynh-Thu et al., 2010). Its computational complexity amounts to $O(pTKn \log n)$. For all datasets the algorithm runs in parallel with all available cores (see Section 4.1 for hardware details) and parameters are set to, $n_{TF} = n$, $K = \sqrt{n}$ and $n_{min} = 1$. Setting $n_{TF} = n$ allows all genes to be potential regulators and leads to a denser network.

GRNBoost2 Algorithm

Another type of ensemble method is gradient boosting, which additively combines weak learners to create a predictor with lower bias and variance. In contrast to bagging, weak learners are sequentially produced during the training phase. After each addition of a weak learner, higher weightage is assigned to incorrectly classified samples to improve performance. For example, the well-established gradient boosting machines (GBMs) have already been applied to GRN inference problems (Awek and Arodz, 2013). GRNBoost2 is a stochastic variation on GBMs, equipped with an early-stopping strategy (Moerman et al., 2019). It sequentially grows decision trees using randomly chosen 90% of the original data and evaluates a loss function with the remaining 10%. This way, it can be checked how much adding a decision tree improves performance and whether an early-stopping criterion is met. The algorithm is implemented using the python package `arboreto` (Moerman et al., 2019). Several parameters need to be specified:

- `learning_rate = 0.01`

  Low values encourage large ensembles.

- `n_estimators = 5000`

  Upper bound for the number of estimators.

- `max_features = 0.1`

  Percentage of features used in every tree node.

- `subsample = 0.9`

  Percentage of samples used for training. If it is set to 1.0, then early-stopping is deactivated.

- `early_stop_window_length = 25`

  Size of the window over which loss function improvements are averaged.

Advantages & Disadvantages

Penalized linear regression methods are computationally cheap and therefore scalable. They can also benefit from a probabilistic interpretation as it will become clear in the next section. Using non-linear regression methods for GRN inference has multiple advantages as well. Due to their non-parametric nature, no assumption about the type of relationship between any variables needs to be made for tree-based methods. Furthermore, they are simple to use, have few parameters that need to be tuned, and can handle both continuous and discrete data. Another advantage of

tree-based methods over the other data-driven methods discussed in previous sections is the ability to infer directed edges. However, it needs to be noted that this direction does not rely on causal modeling, as discussed in Section 3.3.3. Nevertheless, the methods seem to be able to recover some causal information. Also, the computational cost remains to be reduced for GENIE3 to be applied to complex mammalian networks. Already for $n = 805, p = 4511, n_{TF} = 334$, the algorithm runs for about 20 hours on a 16GB RAM, Intel Xeon E5520 2.27 GHz computer. GRNBoost2 is more efficient and scalable (speedup factor via parallelization is greater) while achieving similar performance (Moerman et al., 2019).

3.3 Probabilistic Methods

When a graph encodes a conditional dependence structure for a set of random variables, it is referred to as a probabilistic graphical model (PGM) (Jordan, 2004). Some PGMs are of particular interest as they give rise to methods for GRN inference. More specifically, Gaussian graphical models (GGMs), Bayesian networks (BNs), and causal models will be examined.

3.3.1 Gaussian Graphical Models

The description of Gaussian graphical models closely follows the book "Statistics for High-Dimensional Data: Methods, Theory and Applications" (Bühlmann and van de Geer, 2011). An undirected graphical model is characterized by the pair (G, P) where $G = (V, E)$ is an undirected simple graph and P a joint probability distribution of the vertices $V = \{1, ..., p\}$ which fulfills the pairwise and global Markov property. The pairwise Markov property is fulfilled for P with respect to G if:

$$(j,k) \notin E \implies X^{(j)} \perp X^{(k)} | X^{(V \setminus \{j,k\})}, \tag{3.11}$$

where $\perp$ denotes statistical independence. The global Markov property states:

$$A \text{ and } B \text{ are separated by } C \implies X^{(A)} \perp X^{(B)} | X^{(C)}, \tag{3.12}$$

where A, B, C are disjoint subsets of V (Bühlmann and van de Geer, 2011). An undirected graphical model is called a Gaussian graphical model (GGM) when P is a multivariate Gaussian; $X = (X^{(1)}, ..., X^{(p)}) \sim \mathcal{N}_p(\boldsymbol{\mu}, \boldsymbol{\Sigma})$. In this case the pairwise and global Markov properties are equivalent (Bühlmann and van de Geer, 2011).

Conditional Independence, Partial Correlation and Linear Regression

A particularly useful result for GRN inference regarding GGMs is a property that relates the existence of edges, conditional independence and entries of the inverse of the covariance matrix $\boldsymbol{\Theta} = \boldsymbol{\Sigma}^{-1}$, also termed precision or concentration matrix, as follows (Lauritzen, 1996):

$$(j,k) \notin E \iff X^{(j)} \perp X^{(k)} | X^{(V \setminus \{j,k\})} \iff (\boldsymbol{\Theta})_{j,k} = 0. \tag{3.13}$$

Moreover, the scaled version of the precision matrix $\boldsymbol{\Theta}$, denoted by $\boldsymbol{C}$ corresponds to partial correlations (Lauritzen, 1996):

$$(\boldsymbol{C})_{j,k} = \frac{(\boldsymbol{\Theta})_{j,k}}{\sqrt{(\boldsymbol{\Theta})_{j,j}(\boldsymbol{\Theta})_{k,k}}}, \tag{3.14}$$

$$\boldsymbol{\rho}_{jk|V \setminus \{j,k\}} = -(\boldsymbol{C})_{j,k}, \tag{3.15}$$

where $\boldsymbol{\rho}_{jk|V \setminus \{j,k\}}$ is the partial correlation between $X^{(j)}$ and $X^{(k)}$ given $X^{(V \setminus \{j,k\})}$. This results in the following relation for GGMs:

$$(j,k) \text{ and } (k,j) \in E \iff (\boldsymbol{\Theta})_{j,k} \neq 0 \iff \boldsymbol{\rho}_{jk|V \setminus \{j,k\}} \neq 0. \tag{3.16}$$

Hence, the undirected graph of a GGM can be fully determined with either the precision matrix or partial correlations. Furthermore, Friedman et al. (2009) establish a connection to multiple linear regression (Friedman et al., 2009). Let $X = (Z, Y)$, where $Z = (X^{(1)}, ..., X^{(p-1)})$ and $Y = X^{(p)}$. Accordingly, $\mathbf{\Sigma}$ and $\mathbf{\Theta}$ are partitioned such that:

$$\mathbf{\Sigma} = \begin{bmatrix} \mathbf{\Sigma}_{ZZ} & \boldsymbol{\sigma}_{ZY} \\ \boldsymbol{\sigma}_{ZY}^T & \sigma_{YY} \end{bmatrix} \text{ and } \mathbf{\Theta} = \begin{bmatrix} \mathbf{\Theta}_{ZZ} & \boldsymbol{\theta}_{ZY} \\ \boldsymbol{\theta}_{ZY}^T & \theta_{YY} \end{bmatrix}, \tag{3.17}$$

where $\mathbf{\Sigma}_{ZZ}, \mathbf{\Theta}_{ZZ} \in \mathbb{R}^{(p-1)\times(p-1)}$, $\boldsymbol{\sigma}_{ZY}, \boldsymbol{\theta}_{ZY} \in \mathbb{R}^{p-1}$ and $\sigma_{YY}, \theta_{YY} \in \mathbb{R}$. Then the correspondence between the regression coefficients ($\boldsymbol{\beta} \in \mathbb{R}^{p-1}$) for the population multiple linear regression of Y on Z and the entries of the matrices from (3.17) can be stated as (Friedman et al., 2009):

$$\boldsymbol{\beta} = \mathbf{\Theta}_{ZZ}\boldsymbol{\sigma}_{ZY} = -\frac{\boldsymbol{\theta}_{ZY}}{\theta_{YY}}. \tag{3.18}$$

This holds for any analogous partition of $X = (Z, Y)$ where $Y = X^{(i)}$ and Z is a vector of all other variables. Let the corresponding regression coefficient vector be denoted by $\boldsymbol{\beta}^{(i)}$. Then this result can be combined with Equation (3.16) to obtain:

$$(j,k) \text{ and } (k,j) \in E \iff (\mathbf{\Theta})_{j,k} \neq 0 \iff \boldsymbol{\rho}_{jk|V\setminus\{j,k\}} \neq 0 \iff \boldsymbol{\beta}_k^{(j)} \neq 0 \text{ and } \boldsymbol{\beta}_j^{(k)} \neq 0. \tag{3.19}$$

GLASSO

The findings from above imply that GRN inference can be performed through estimating the precision matrix $\mathbf{\Theta}$ from given gene expression data. Under the assumption of a GGM with $X = (X^{(1)}, ..., X^{(p)}) \sim \mathcal{N}_p(\boldsymbol{\mu}, \mathbf{\Sigma})$, let the data $\mathbf{D} \in \mathbb{R}^{n\times p}$ be a realization of X, μ denote the population mean and $\mathbf{S}$ the empirical covariance matrix:

$$\mathbf{S} = \frac{1}{n}\sum_{i=1}^{n}((\mathbf{D})_{i,:} - \overline{(\mathbf{D})_{i,:}})((\mathbf{D})_{i,:} - \overline{(\mathbf{D})_{i,:}})^T, \tag{3.20}$$

where $\overline{(\mathbf{D})_{i,:}}$ is the arithmetic mean of all entries in a sample vector $(\mathbf{D})_{i,:}$. Then the log-likelihood of the data (ignoring constants) is (Friedman et al., 2009):

$$l(\mathbf{\Theta}) = \log\det\mathbf{\Theta} - \text{trace}(\boldsymbol{S}\mathbf{\Theta}), \tag{3.21}$$

where the maximum likelihood estimate of $\mathbf{\Sigma} = \mathbf{\Theta}^{-1}$ is the empirical covariance $\boldsymbol{S}$. However, $\boldsymbol{S}$ is generally not sparse and hence unsuitable for GRN inference. One way to estimate a sparse precision matrix is to use the graphical LASSO (GLASSO), a modified version of the LASSO as introduced in Subsection 3.2.3. GLASSO complements the likelihood from (3.21) with a L_1 regularization on $\mathbf{\Theta}$:

$$l_{LASSO}(\mathbf{\Theta}) = \log\det\mathbf{\Theta} - \text{trace}(\boldsymbol{S}\mathbf{\Theta}) - \rho\left\|\mathbf{\Theta}\right\|_1, \tag{3.22}$$

where $\rho \in \mathbb{R}$ is a regularization parameter. Such convex optimization problem can be solved with the R package `glasso` which employs a coordinate-descent algorithm (Friedman et al., 2008). The parameter ρ controls the sparsity of the predicted network. The larger it is, the more edges will be removed from the network as their edge weights are penalized more in 3.22. The algorithm is run twice for two different regularization parameters ρ_1, ρ_2, where $\rho_1 < \rho_2$. The first run with ρ_1 is used to precondition the regression coefficients for a final run with ρ_2. To predict non-sparse GRNs, $\rho_1 = 0.001$ and $\rho_2 = 0.002$ are chosen to be small.

PPCOR

A way to estimate an unregularized precision matrix is to compute the inverse of the empirical covariance matrix. The algorithm PPCOR (**P**airwise **P**artial **COR**relation) computes the

pseudo-inverse using the Moore-Penrose generalized matrix inverse (Penrose, 1955). Furthermore, it calculates the level of statistical significance for the obtained partial correlations (Kim, 2015), which can be thresholded for controlling the number of predicted edges. However, similar to CORR in Subsection 3.2.1, the threshold is set to 1 in order to predict a dense network. PPCOR is implemented using the R package `ppcor` (Kim, 2015).

GENENET

While it is known that correlation does not provide causal information, GENENET can approximate the causal structure of the ground truth network via a simple heuristic (Opgen-Rhein and Strimmer, 2007). Similar to PPCOR, it first computes a partial correlation network. Then multiple testing of the log-ratio of standardized partial variances is used to obtain a partial ordering of nodes. A specified number of strongest edges then defines a partially directed graph. The algorithm is implemented using the R package `genenet` (Opgen-Rhein and Strimmer, 2007). To predict a non-sparse GRN, all edges are chosen to be contained in the graph.

Advantages & Disadvantages

A significant advantage of the methods described above is their scalability. Their low computational cost allows them to be applied to complex biological networks. Furthermore, the statistical properties of partial correlation and its connection to Gaussian graphical models are well-understood (Bühlmann and van de Geer, 2011). The magnitude of the partial correlations also provides a natural way to define weights for gene-to-gene interactions. While GENENET can infer a directed graph, GGMs remain non-causal models and can at most heuristically approximate the regulatory structure of the underlying ground truth network.

3.3.2 Bayesian Networks

Unlike undirected graphical models discussed in the previous subsection, directed graphical models offer a way to represent directed relationships between genes. A well-known example is the Bayesian network (BN), in which edges represent conditional probabilities. More formally, a BN is a pair (G, P) where $G = (V, E)$ is a directed acyclic graph (DAG) with $V = \{1, ..., p\}$ representing random variables $X = \{X^{(1)}, ..., X^{(p)}\}$, and P a joint probability distribution for all vertices which factors into conditional probabilities as (Beerenwinkel and Siebourg, 2012):

$$P(X^{(1)}, ..., X^{(p)}) = \prod_{i=1}^{p} P(X^{(i)} | X^{\mathrm{pa}(i)}), \tag{3.23}$$

where $\mathrm{pa}(i)$ denotes the set of parents of vertex i in G; $X^{\mathrm{pa}(i)} = (X^{(1)}, ..., X^{(k)})$ when $\{1, ..., k\}$ are the parents of i in the graph. Bayesian networks also have the potential for causal inference as it will be further discussed in the next subsection. To quantify how well a graph structure G describes given data $\mathbf{D}$, one can consider the posterior probability $P(G|\mathbf{D})$. Via Bayes theorem it can be expressed as follows:

$$P(G|\mathbf{D}) = \frac{P(\mathbf{D}|G)P(G)}{\sum_{G' \in \mathcal{G}} P(\mathbf{D}|G')P(G')}, \tag{3.24}$$

where $P(\mathbf{D}|G)$ is the data likelihood, $P(G)$ is the prior distribution of graph topologies, and the denominator is the marginal data likelihood for the set of all possible graph structures $\mathcal{G}$. It needs to be noted that the factorization of the joint probability distribution is not necessarily unique. This means that multiple networks can have the same probability distribution, a property known as Markov equivalence (Barber, 2012). As the number of possible graphs explodes super-exponentially with the number of vertices, computing the marginal likelihood is intractable for larger graphs. One way to circumvent this difficulty is to perform approximate inference via sampling methods for which $P(G|\mathbf{D})$ is not explicitly computed but sampled from. Doing so requires understanding Markov Chain Monte Carlo (MCMC) and the Metropolis-Hastings algorithm. Their elucidation closely follows the book "Learning in Graphical Models" (Jordan, 1998).

Markov Chain Monte Carlo (MCMC)

Let $P(X)$ be an unknown target distribution, then MCMC finds a finite sample $X^{(1)}, ..., X^{(R)}$ of size $R \in \mathbb{N}$ which follows the distribution of X:

$$\{X^{(r)}\}_{r=1}^{R} \sim P(X). \tag{3.25}$$

It does so by constructing a Markov chain $\{X^{(m)}\}_m \supset \{X^{(r)}\}_{r=1}^{R}$ with transition matrix $\mathbf{T}$ where $(\boldsymbol{T})_{x,y} = P(X^{(m+1)} = x | X^{(m)} = y)$ for all $x, y \in S$, the common state space of random variables $X^{(i)}$ for $i \in \{1, ..., m\}$. If the Markov chain is ergodic (aperiodic and irreducible) and the detailed balance equations $(\boldsymbol{T})_{x,y} P(X = y) = (\boldsymbol{T})_{y,x} P(X = x)$ hold, then its unique stationary distribution corresponds to the unknown target distribution:

$$\lim_{m \to \infty} P(X^{(m)}) \to P_\infty(X) = P(X). \tag{3.26}$$

Metropolis-Hastings Algorithm

The Metropolis-Hastings algorithm aims to construct a Markov chain with such unique stationary distribution. Informally, it starts with some random sample $X^{(0)}$, proposes a new sample $X^{(1)}$ with a proposal distribution and accepts it with a certain probability. This is repeated until convergence. Let $(\boldsymbol{Q})_{x,y}$ be the proposal distribution and $(\boldsymbol{A})_{x,y} = A(X^{(m)} = x | X^{(m-1)} = y)$ the acceptance probability, then the transition matrix $\mathbf{T}$ of the Markov chain has entries $(\boldsymbol{T})_{x,y} = (\boldsymbol{Q})_{x,y}(\boldsymbol{A})_{x,y}$. Such that detailed balance is fulfilled, $(\boldsymbol{A})_{x,y}$ is set to:

$$(\boldsymbol{A})_{x,y} := \min \left\{ \frac{P(X = x)(\boldsymbol{Q})_{y,x}}{P(X = y)(\boldsymbol{Q})_{x,y}}, 1 \right\}. \tag{3.27}$$

Before convergence, Markov chain elements do not yet follow the target distribution. Hence, this phase is discarded when sampling from the Markov chain. When applied to Bayesian networks, each element in the Markov chain represents a DAG $G^{(m)}$. Hence, $\mathbf{Q}$ proposes a new graph given the previous, and the acceptance probability is set accordingly:

$$G^{(m)} \sim Q(G^{(m)} | G^{(m-1)}), \tag{3.28}$$

$$A(G^{(m)} | G^{(m-1)}) = \min \left\{ \frac{P(\mathbf{D} | G^{(m)}) P(G^{(m)}) Q(G^{(m-1)} | G^{(m)})}{P(\mathbf{D} | G^{(m-1)}) P(G^{(m-1)}) Q(G^{(m)} | G^{(m-1)})}, 1 \right\}, \tag{3.29}$$

where $\frac{Q(G^{(m-1)} | G^{(m)})}{Q(G^{(m)} | G^{(m-1)})}$ is called the Hastings ratio. For a Hastings ratio of 1, the Metropolis-Hastings algorithm becomes the Metropolis algorithm. A special instance of Metropolis-Hastings is Gibbs sampling. Instead of proposing new values for all entries of a sample, the proposal distribution only proposes one new coordinate based on all others. This is particularly useful for graphical models, as the conditional probabilities are much easier to sample from than the joint distribution.

ORDER_MCMC Algorithm

There are multiple adaptations of the Metropolis-Hastings algorithm for application on graphical models. Some examples are Structure MCMC (Madigan et al., 1995), Order MCMC (here denoted ORDER_MCMC for consistency) (Friedman and Koller, 2003) and Partition MCMC (Kuipers and Moffa, 2017). They differ in how they explore the space of possible graphs. The idea for ORDER_MCMC is to coarse-grain the DAG space by only considering groups of DAGs as samples. Each group can be described by a specific ordering of nodes as permutation π with the property that parents can only be further down the chain:

$$pa\{\pi(i)\} \subseteq \{\pi(j) | j > i\}. \tag{3.30}$$

The probability of a certain permutation can then be expressed as:

$$P(\pi|\mathbf{D}) = \sum_{G \prec \pi} P(G|\mathbf{D}), \tag{3.31}$$

where $G \prec \pi$ describes all graphs G that are consistent with a given permutation π. Then a new permutation π' is proposed by swapping two of its elements and is accepted with the probability $\min\left\{\frac{P(\pi'|\mathbf{D})}{P(\pi|\mathbf{D})}, 1\right\}$. Only considering permutations instead of all possible graphs significantly reduces computational cost (Friedman and Koller, 2003). The implementation ORDER_MCMC relies on the R package `BiDAG` (Suter et al., 2021). For obtaining edge weights, bagging as discussed in Section 3.1.2 is used. The resulting independent for-loop iterations are parallelized and run with `nIters = 50`, the number of bootstrap iterations, and `cpdag = TRUE`; the PC algorithm is used for initialization. The PC algorithm will be discussed in the next section.

Advantages & Disadvantages

One advantage of Bayesian network methods is that they intrinsically quantify uncertainty by finding probability distributions and not a maximum (Huynh-Thu and Sanguinetti, 2018). Moreover, in contrast to undirected graphical models, Bayesian networks offer a natural way of introducing directed interactions. However, because of Markov equivalence, multiple possible conditional probabilities can explain the data equally well. Furthermore, as the space of network structures grows super-exponentially with the number of vertices, scalability remains a significant challenge (Huynh-Thu and Sanguinetti, 2018). Lastly, Bayesian networks cannot model regulatory cycles in biological networks. A way to circumvent this is to use dynamic Bayesian networks, which usually require time-series data (Huynh-Thu and Sanguinetti, 2018).

3.3.3 Graphical Causal Models

As briefly mentioned in Chapter 1, it is essential to distinguish between associational and causal models. While all previously described methods assign purely associational meaning to undirected and directed edges, this Subsection elucidates how GRNs may be inferred causally. The inference of causal networks from purely observational data is known as causal discovery or causal structure search (Ahmed et al., 2020). The PC algorithm endows Bayesian networks with a causal interpretation and will be discussed as an example.

Causal Discovery Methods

Causal discovery methods for network inference encompass an extensive collection of concepts and are not restricted to graphical models. Categories include continuous, logical, probabilistic, information-theoretic, algebraic, statistical, stochastic, and hybrid models (Ahmed et al., 2020). As a complete description thereof would go beyond the scope of this thesis, here the focus lies on probabilistic models, more specifically directed graphical causal models (DGCM) (Glymour et al., 2019).

Directed Graphical Causal Model

A DGCM consists of three components; a set of vertices representing random variables, their joint probability distribution, and a set of edges where an edge between two vertices encodes "the hypothesis that the two variables would be associated if all other variables were fixed at some values while the tail variable is exogenously varied" (Glymour et al., 2019). For structure learning in DGCMs, constraint-based, score-based, and hybrid algorithms have been considered (White and Vignes, 2018). Constraint-based algorithms usually start with a complete undirected graph, remove edges via conditional independence tests, and then direct edges where adequate information is available. The PC algorithm will be discussed below as an instance thereof. Score-based approaches assign a score to the entire network instead of single edges at a time. After

initializing a network, possible new structures are proposed and the score optimized. Hybrid methods combine ideas of the former types.

PC Algorithm

The constraint-based PC algorithm can estimate a graphical causal structure from observational data (Kalisch and Bühlmann, 2007). It consists of the following steps (Glymour et al., 2019):

1. A complete unweighted and undirected graph $G = (V, E)$ is initialized.
2. Edges are eliminated between variables that are unconditionally independent (independent without conditioning on any other variables). Any statistical conditional independence test can be used here in principle. For the version of the implementation from Table 3.1, a Gaussian conditional independence test with a significance level $\alpha = 0.2$ is employed. α acts as a sparsity parameter, as greater values will lead to fewer pairs of variables being accepted as conditionally independent and thus a denser graph.
3. For each pair of connected (linked via an edge) variables $(A, B) \in E$ the following is done: For growing subsets of the remaining variables $S = \{C\}, \{C, D\}, \{C, D, E\}, ..., E \setminus \{A, B\}$ and while the nodes in the corresponding subset are either all connected to A or B, the edge (A, B) is eliminated if A is conditionally independent of B given S. As soon as the conditioning subset S is not fully connected to either A or B, the algorithm continues with the next pair of connected variables.
4. Edges are directed if possible. For all triplets (A, B, C) where A and B, and B and C are connected, orient the edges as $A \rightarrow B \leftarrow C$, if B was not in the conditioning set on which A and C were determined independent. Such directed triplets are called v-structures.
5. Orient all triplets of variables where $A \rightarrow B - C$, such that $A \rightarrow B \rightarrow C$. This step is known as orientation propagation.

The result is a partially directed acyclic graph (PDAG) representing a Markov equivalence group. By performing bootstrapping with 50 iterations, edge weights can be obtained as described in 3.1.2. The R package `pcalg` was used for the implementation (Kalisch et al., 2012; Hauser and Bühlmann, 2012).

Advantages & Disadvantages

Causal models provide a mathematical language for describing causality and allow the inference of causal networks. This is crucial for achieving the ultimate goal of GRN reconstruction as it engenders a deeper understanding of the regulatory relationships between genes. However, the performance of causal discovery methods for biological data is unsatisfactory (Ahmed et al., 2020). They are usually only suitable for small or moderate-sized networks, so computationally more efficient algorithms are required.

Chapter 4

Evaluation Framework

This chapter begins by describing the technological setup and benchmarking framework used for the implementation and generation of results. Then, it will elucidate various evaluation metrics used to compare methods. The source code can be found on a GitHub repository (`https://github.com/michavol/benchmarking_GRN_inference`, (Vollenweider, 2022)).

4.1 Technological Setup

In order to make the findings of this thesis reproducible, the BEELINE framework by Pratapa et al. (2020) is adopted. Originally, it was designed for benchmarking state-of-the-art algorithms intended for single-cell transcriptomics data. It is modified to accommodate the methods from Chapter 3 and the datasets are replaced by the ones discussed in Chapter 2. Also the evaluation methods are adjusted and extended. It utilizes containerization via Docker, allowing uniform and consistent deployment of software code and all its dependencies independent of the infrastructure (Merkel, 2014). This significantly facilitates the setup process and engenders effortless integration of additional methods and datasets. Benchmarks are performed on an AMD Ryzen 7 3700X (8 cores, 3.60GHz, 32MB) processor. On this specific machine, Docker Desktop on Ubuntu 20.04 is used for containerization management and the resources are configured as follows (parameter definitions are taken from the official website: `https://docs.docker.com/desktop/windows/`):

- **CPUs** = 14

  Number of processors available on the host machine.

- **Memory** = 25 GB

  Amount of runtime memory allocated from the total available memory (RAM) on the machine.

- **Swap File Size** = 3 GB

  Temporary space on a hard drive, used to store information when RAM is fully utilized.

- **Disk Image Size** = 64 GB

  Space that is available for containerization.

4.2 Evaluation Metrics

Multiple methods are available to quantify to what degree an inferred GRN resembles the ground truth. In literature, the receiver operating characteristic (ROC) and precision-recall (PR) curves are standard tools to evaluate the predictive ability of a binary classifier. In addition to these, early precision is also considered. To assess the stability across datasets and the similarity of predictions

among methods, the Jaccard index is introduced. Of course, the computational runtime is also measured. This section outlines the meaning of these evaluation metrics and describes how they are computed.

4.2.1 Directed vs. Undirected Evaluation

As it became evident in Chapter 3, it is important to distinguish between directed and undirected networks as described in Subsection 1.2.1. This especially holds for the evaluation of methods. Accordingly, ROC and PR curves, early precision, and Jaccard indices are implemented in two ways, one version for a directed and the other for an undirected representation of the ground truth GRNs. The ground truth GRN is encoded by an undirected simple graph for the undirected evaluation. Hence, for a fair comparison, results from methods that return weighted partially directed graphs need to be converted into weighted undirected graphs. This can be achieved by considering all edges as unordered pairs of vertices and merging duplicate edges into one using some aggregation operation for the edge weights. Here, the edge weights are averaged. Depending on how edge weights are interpreted, there are also other aggregation operations one can consider. Conversely, weighted undirected graphs first need to be turned into weighted directed ones for directed evaluation. One can simply convert each unordered pair (g_i, g_j) into two ordered pairs $(g_i, g_j), (g_j, g_i)$ and assign the original edge weight to both of them. The idea is that as an undirected edge does not contain any information about orientation, all possible directions should be equally weighted.

4.2.2 ROC & PR Curves

In the context of GRN inference, a true positive TP is an edge that exists in the ground truth network and the predicted network. False positives FP are edges incorrectly predicted to be found in the network. Conversely, false negatives FN are edges in the ground truth that are not identified by the inference method. True negatives TN are correctly excluded edges. The following table defines some quantities that ROC and PR curves rely on.

Quantity	**Formula**
True Positive Rate (TPR)	$\frac{TP}{TP+FN}$
False Positive Rate (FPR)	$\frac{FP}{FP+TN}$
Precision	$\frac{TP}{TP+FP}$
Recall	$\frac{TP}{TP+FN}$

It is important to note that there are conceptually different ways to find ROC and PR curves. While the method-independent computation relies on methods being able to assign edge weights, the method-specific one requires the existence a model parameter which can be varied to control the sparsity of the prediction. As not all methods provide easily tunable sparsity parameters, the evaluation framework uses the method-independent technique. Both will be elaborated on in the following.

Method-Independent Computation of ROC and PR Curves

Consider a groundtruth network and a corresponding predicted GRN. The quantities from the table above are computed once for every new sub-network obtained by thresholding the network sparsity of the GRN at all possible edge weights. That is, first they are calculated for the empty network, then for a network only containing the edge with the greatest edge weight and so on until it is computed once with no threshold at all, that is, with the originally predicted set of edges. This results in about as many TPR, FPR, Precision and Recall values as there are edges in the predicted GRN, which can be used to draw various curves. To obtain a ROC curve, the true positive TPR (x-axis) is plotted against the FPR (y-axis) and the datapoints are then linearly interpolated (connected with straight lines). Analogously the PR curve is plotted with Recall

on the x-axis and Precision on the y-axis and is sometimes preferred over the ROC curve as it delivers more information on highly-skewed datasets (Davis and Goadrich, 2006). To assign a score to the curves, one can approximate the areas under the curves (Keilwagen et al., 2014). These values are called AUROC and AUPR scores for ROC and PR curves, respectively. Values of 1 describe perfect classifiers. It must be noted that when only a few data points are available to plot the curves (when the predicted GRN is sparse), then one needs to be careful about the interpretation the numerical approximation of the area under the curves as linear interpolation is not meaningful in that situation. This is the reason why it was decided that methods ought to predict non-sparse GRNs. For CORR, PPCOR and GENENET, complete networks are predicted by assigning each possible edge a non-zero weight. For other algorithms sparsity parameters are chosen in such way that significantly more edges are contained in the predicted network than in the ground truth. For instance, GLASSO is parametrized with the rather small $\rho_1 = 0.001$ and $\rho_2 = 0.002$. While it is also possible to run GLASSO once with $\rho = 0$ to obtain a complete network, this removes regularization, thereby qualitatively modifies the algorithm, and leads to worse results. The computation of the curves are performed using the R package `PRROC` (Grau et al., 2015).

Method-Specific Computation of ROC and PR Curves

The other way to obtain different data points for TPR, FPR, Precision and Recall relies on model-specific sparsity parameters and does not require the GRN to contain edge weights. Take the computation of ROC and PR curves for GLASSO as an example: The method is run several times with increasing values for parameters ρ_1 and ρ_2 which generally results in a sequence of predicted GRNs with increasing sparsity. For each GRN, TPR, FPR, Precision and Recall are computed, where the number of data points now represents how often the method was run with different parameters. The curves are then drawn the same way as for their method-independent computation.

Method-Independent vs. Method-Specific Computation

Having to pick a single sparsity parameter to run the algorithm with is simultaneously the greatest advantage and disadvantage of the method-independent approach; the algorithm has to be run only once to generate all data points for drawing ROC and PR curves. This makes it a lot faster than the method-specific approach, for which the method needs to be applied once for each data point. Furthermore, it allows the comparison of methods for which the sparsity of the prediction cannot be easily regulated. However, it also requires the user to pick such sparsity parameter and the choice of which may impact the shape of the resulting curves. It ought to be investigated which, if any, choice of sparsity parameters for the method-independent computation leads to similar ROC and PR curves as the ones generated by the method-specific approach.

4.2.3 Early Precision

Here, early precision describes the fraction of true positives in the top-k edges, where k is the number of edges in the ground truth network. The idea is to quantify to what extent the model assigns large edge weights to true edges. The greater the value, the more the ordering by the edge weights can be trusted.

4.2.4 Jaccard Index

The Jaccard index $J(A, B)$ describes the similarity between finite sample sets A and B (Jaccard, 1901). It is defined by:

$$J(A,B) = \frac{|A \cap B|}{|A \cup B|} = \frac{|A \cap B|}{|A| + |B| - |A \cap B|} \tag{4.1}$$

By design, $0 \leq J(A,B) \leq 1$. It is evident that if $|A \cap B| = 0$, i.e. the sets have no elements in common, then $J(A,B) = 0$. The other extreme occurs when A and B are the same; hence, $J(A,B) = 1$. If A and B are empty sets, it is defined to be $J(A,B) = 1$. In the context of GRN inference, this kind of similarity measure may be used two-fold. First, when multiple datasets are generated from the same underlying ground truth, the Jaccard index across the predicted networks of a method can quantify its stability. Second, one can assess how similar predictions of different methods are to each other.

4.2.5 Runtime

The last metric that is measured for benchmarking is the computational runtime, where one can differentiates between elapsed real time (also wall time) and CPU time (`https://www.gnu.org/software/libc/manual/`). The former describes the complete time difference between starting and ending a task. The latter is a measure of the time during which the processor is actively working on a certain task. For tasks using a single processor the wall time is always greater as it includes the CPU time. Their discrepancy arises from computer architecture dependent factors, such as waiting for the system resources to become available. For parallel computations (multiple processors are used simultaneously) the CPU time is the sum of all individual CPU times and can therefore be greater than the elapsed time. For the evaluation, the elapsed time, user time and CPU percentage is recorded. User time is the portion of the CPU time where the CPUs are busy executing application software. It expresses how much actual computation needs to be performed for a method. CPU percentage is the percentage of a CPU being used by a method. If multiple processors are used, its value can be greater than 100%; if a method is run in parallel with 8 processors, but only employs half of each, then the CPU percentage would correspond to 400%. This metric gives an impression of how well a certain implementation of an algorithm is able to use given resources.

Chapter 5

Benchmarking Results

This chapter aims to demonstrate how the different evaluation metrics from Chapter 4.2 can be leveraged to answer diverse questions one may pose about the data and methods. Section 5.5 offers a brief summary of the findings. For the reader's convenience, the methods used and datasets considered are listed below once more:

Type	**Name**	**Concept**	**Graph**	**Parallel**	**Subsection**
Data-Driven	CORR	Correlation	UG	No	3.2.1
	ARACNE	Mutual Information	UG	No	3.2.2
	GENIE3	Regression: Random Forest	PDG	Yes	3.2.3
	GRNBOOST2	Regression: Boosting	PDG	Yes	3.2.3
Probabilistic	GLASSO	Gaussian Graphical Model	UG	No	3.3.1
	PPCOR	Gaussian Graphical Model	UG	No	3.3.1
	GENENET	Gaussian Graphical Model	PDG	No	3.3.1
	ORDER_MCMC	Bayesian Networks: MCMC	PDAG	Yes	3.3.2
	PC	Causal Model	PDAG	Yes	3.3.3

Table 5.1: Overview of methods. All graphs have non-negative edge weights.
(U=Undirected, D=Directed, P=Partially, A=Acyclic, G=Graph)

Source	**Names**	**Type**	**p**	**n**	**k**	**Subsection**
DREAM4	DS-D1:1-5	Sim. Multifactorial Bulk	10	10	15	2.3.1
	DS-D2:1-5	Sim. Multifactorial Bulk	100	100	250	2.3.1
SERGIO	DS-S1:1-15	Sim. Static Single-Cell	100	2700	250	2.3.2
	DS-S2:1-15	Sim. Static Single-Cell	400	2700	1155	2.3.2
	DS-S3:1-15	Sim. Static Single-Cell	1200	2700	2713	2.3.2
	DS-S1-bulk:1-15	Sim. Bulk	100	9	250	2.3.2
	DS-S2-bulk:1-15	Sim. Bulk	400	9	1155	2.3.2
	DS-S3-bulk:1-15	Sim. Bulk	1200	9	2713	2.3.2
TCGA	DS-TCGA-1	Real Cancer Data	15,172	887	?	2.3.3

Table 5.2: Overview of datasets.
(**p** = number of genes, **n** = number of samples, **k** = number of edges, Sim.=Simulated)

In theory, all metrics from Section 4.2 can be evaluated for all datasets in the eight dataset collections from DREAM4 and SERGIO (two for DREAM4, six for SERGIO), as the ground truth networks are available. However, independent of the method, the execution of the evaluation code

takes too long for the large networks (some more than 700000 edges) inferred from DS-S3 and DS-S3-bulk, and their results can not be presented. Even for DS-S2 and DS-S2-bulk, it is pretty slow, so only the datasets DS-S2:1-3 and DS-S2-bulk:1-3 are considered, and the corresponding inferred GRNs needed to be limited to 10000 edges. For the future application of the framework, finding a more efficient implementation of the evaluation metrics is desirable. For the TCGA dataset, only the Jaccard index across methods and runtimes can be computed as there is no ground truth available. The following sections will present how the results for the evaluation metrics can be depicted and analyzed.

5.1 Runtimes

Analyzing runtimes allows investigating how algorithms scale with the problem size and determining whether a particular method is applicable to large real biological datasets such as the TCGA dataset. The following tables present the elapsed time, user time and CPU percentage. Each algorithm is applied once to each dataset and the tables contain the median and standard deviation across the datasets in a collection. For better readability, the best three methods are shaded in green and the worst three in red, where the overall best and worst scores are in bold.

5.1.1 DREAM4

DS-D1:1-5

Table 5.3 below shows that for DS-D1:1-5, all methods finish in less than 20 seconds. This is no surprise as the data only contains 100 gene expression values. As anticipated, especially methods relying on simple metrics such as correlation (CORR), mutual information (ARACNE) and partial correlation (GLASSO, PPCOR, GENENET) are very fast. ORDER_MCMC is by far the slowest with 15.44 seconds. Interestingly, however, the PC algorithm is the slowest when considering user time. This indicates that ORDER_MCMC may require less work than the PC method but cannot fully employ the available CPU resources, which is confirmed by its low CPU percentage of 13%. Also GRNBOOST2 only reaches 53% despite being run in parallel with 14 cores (a maximum CPU percentage of 1400%). As this percentage increases with larger problem sizes, as will be seen later, this may be due to the overhead of its parallelization; the cost of managing multiple processors dominates the benefit of parallelization. An example where parallelization already pays off with $n = 10, p = 10$ is GENIE3, where with a CPU percentage of 218%, the elapsed time is already shorter than the user time. All other methods run with about 99%, which implies that they do not run in parallel.

	Elapsed [s]		**User [s]**		**CPU %**	
	Median	*Std*	*Median*	*Std*	*Median*	*Std*
CORR	0.23	0.07	0.19	0.03	98	1.03
ARACNE	0.24	0.05	0.18	0.03	99	3.83
GENIE3	0.85	0.31	1.12	0.20	**218**	**44.76**
GRNBOOST2	4.16	0.72	1.87	0.64	53	5.32
GLASSO	0.26	0.08	0.20	0.07	98	2.26
PPCOR	**0.20**	**0.07**	**0.15**	**0.05**	99	4.13
GENENET	0.68	0.25	0.60	0.21	99	2.04
ORDER_MCMC	**15.44**	**7.48**	1.69	0.56	**13**	**2.00**
PC	2.47	1.05	**2.30**	**1.02**	99	0.00

Table 5.3: Runtime comparison for DS-D1:1-5.

DS-D2:1-5

With $p = 100$ and $n = 100$ the algorithms already need to process 10000 gene expression values. CORR, ARACNE, PPCOR and GLASSO remain very fast, while some other algorithms, especially GENENET, ORDER_MCMC and PC, experience a significant increase in elapsed time. For GENENET, this is probably due to the multiple testing of the log-ratio of standardized partial variances and for ORDER_MCMC and PC because of the super-exponentially increasing space of possible graphs. GENIE3 is among the slowest when looking at user time but about ten times as fast in terms of elapsed time. It parallelizes well, but the amount of work increases significantly nevertheless. This can be explained with its runtime complexity of $O(pTKn \log n)$ with $K = \sqrt{n}$ (see Subsection 3.2.3). What can also be observed is that GRNBOOST2 is now able to use 133% of the CPU, which indicates that the overhead of parallelization has become less dominant. A surprising result is the low user time and CPU percentage of ORDER_MCMC. As ORDER_MCMC is initialized with the PC algorithm, it should also have a greater user time than PC. Interestingly, the elapsed time difference between the two seems more realistic. Maybe the computation of the user time and CPU percentage metrics are erroneous, which may have to do with using the parallelization packages for R `foreach` and `doParallel` within a Docker container. Lastly, while implementing the PC algorithm from the `pcalg` library offers a parallel computation of the output graph, the CPU percentage of 99% indicates that only one processor is actively used. This is the same for all datasets and requires further investigation.

	Elapsed [s]		**User [s]**		**CPU %**	
	Median	*Std*	*Median*	*Std*	*Median*	*Std*
CORR	0.27	0.03	0.23	0.03	99.00	0.00
ARACNE	0.28	0.02	0.21	0.01	98.00	0.89
GENIE3	3.96	0.03	39.26	0.32	**1024.00**	**11.02**
GRNBOOST2	13.11	1.67	12.47	1.88	133.00	6.45
GLASSO	0.34	0.02	0.27	0.02	99.00	0.63
PPCOR	**0.24**	**0.00**	**0.18**	**0.01**	98.00	0.41
GENENET	90.62	2.04	90.51	2.04	99.00	0.00
ORDER_MCMC	**159.01**	**90.44**	1.78	0.58	**1.00**	**0.41**
PC	96.58	34.96	**96.38**	**34.96**	99.00	0.00

Table 5.4: Runtime comparison for DS-D2:1-5.

5.1.2 SERGIO

DS-S1-bulk:1-15

DS-S1, DS-S2 and DS-S3 are all extremely high-dimensional datasets with only nine samples. One would expect that the runtimes are faster for $n = 9$ and $p = 100$ than for $n = 100$ and $p = 100$ as in Table 5.4. This is the case for PC and GENIE3. The user time for GENIE3 goes from 39.26 down to only 2.27, which makes sense considering its theoretical runtime. In contrast, for GENENET and GLASSO, the user time goes up. This may either be due to implementation specifics or hardware inconsistencies.

	Elapsed [s]		**User [s]**		**CPU %**	
	Median	*Std*	*Median*	*Std*	*Median*	*Std*
CORR	0.26	0.02	0.20	0.01	98	3.30
ARACNE	0.26	0.01	0.20	0.01	98	3.07
GENIE3	1.02	0.02	2.27	0.02	**309**	**5.61**
GRNBOOST2	16.54	5.15	13.77	4.92	128	13.01
GLASSO	3.81	0.75	3.75	0.75	99	0.25
PPCOR	**0.24**	**0.01**	**0.18**	**0.01**	97	0.62

GENENET	93.37	2.07	**93.25**	**2.08**	99	0.00
ORDER_MCMC	**147.62**	**2.04**	1.77	0.02	**1**	**0.00**
PC	7.69	0.77	7.50	0.76	99	0.00

Table 5.5: Runtime comparison for DS-S1-bulk:1-15.

DS-S2-bulk:1-15

For $p = 400$ GENENET and ORDER_MCMC take more than 8 and 20 hours, respectively, and are therefore not assigned any values in Table 5.6. While CORR, ARACNE, PPCOR and GENIE3 still finish under a few seconds, GLASSO now takes about 50 times longer than for $p = 100$. Also PC is about 20 times and GRNBOOST two times slower.

	Elapsed [s]		**User [s]**		**CPU %**	
	Median	*Std*	*Median*	*Std*	*Median*	*Std*
CORR	0.55	0.01	0.43	0.01	93	0.82
ARACNE	**0.46**	**0.01**	**0.38**	**0.01**	97	2.22
GENIE3	2.30	0.10	9.59	0.06	**458**	**17.86**
GRNBOOST2	32.06	0.52	29.52	0.72	139	2.00
GLASSO	**194.64**	**7.41**	194.56	7.38	99	0.00
PPCOR	0.63	0.02	0.51	0.01	93	1.73
GENENET	-	-	-	-	-	-
ORDER_MCMC	-	-	-	-	-	-
PC	144.69	99.56	144.14	99.44	99	0.00

Table 5.6: Runtime comparison for DS-S2-bulk:1-15.

DS-S3-bulk:1-15

When running the algorithms on datasets with $p = 1200$ and $n = 9$, it becomes evident that GLASSO and PC do not scale as nicely with the number of genes as the data-driven algorithms CORR, ARACNE, GENIE3, and GRNBOOST2 do (compare Tables 5.6 and 5.7). This makes clear that PC, ORDER_MCMC, GENENET and GLASSO are unsuitable for datasets such as the one from TCGA.

	Elapsed [s]		**User [s]**		**CPU %**	
	Median	*Std*	*Median*	*Std*	*Median*	*Std*
CORR	**3.17**	**0.06**	**2.69**	**0.05**	**91**	**0.58**
ARACNE	3.55	0.08	3.41	0.07	99	0.45
GENIE3	11.18	0.45	46.95	0.97	**434**	**20.15**
GRNBOOST2	64.78	4.81	67.21	5.41	146	3.32
GLASSO	**4175.00**	**191.92**	**4175.07**	**191.92**	99	0.00
PPCOR	6.50	0.09	6.12	0.09	97	0.50
GENENET	-	-	-	-	-	-
ORDER_MCMC	-	-	-	-	-	-
PC	2909.73	5233.97	2900.74	5234.43	99	0.00

Table 5.7: Runtime comparison for DS-S3-bulk:1-15.

DS-S1:1-15, DS-S2:1-15 and DS-S3:1-15

DS-S1, DS-S2 and DS-S3 are unique in that with $n = 2700$ and $p = 100, 400, 1200$, they are by far not as high-dimensional as the others, which gives methods more information to work with but also more data to handle. Table 5.8 does not show any results for ORDER_MCMC and PC as

these took longer than 24 and 8 hours for $p = 100$, respectively. For $p = 400$, GENENET took more than 8 hours and is therefore not shown in Table 5.9. Tables 5.8 and 5.9 also imply that GENIE3 and GRNBOOST2 could not handle the much larger TCGA dataset in less than a day on the given machine with their current implementation. While it would be possible to run CORR, ARACNE and PPCOR on DS-S3:1-15, evaluating the performance metrics takes more than 10 hours as the predicted networks are vast.

	Elapsed [s]		**User [s]**		**CPU %**	
	Median	*Std*	*Median*	*Std*	*Median*	*Std*
CORR	**0.69**	**0.05**	0.60	0.04	99.00	1.52
ARACNE	0.69	0.06	**0.58**	**0.06**	98.00	0.85
GENIE3	**292.95**	**3.76**	**2199.03**	**18.61**	**761.00**	**6.18**
GRNBOOST2	152.27	3.98	1072.39	31.25	728.00	7.68
GLASSO	0.95	0.08	0.88	0.07	99.00	0.83
PPCOR	1.11	0.04	1.01	0.04	99.00	0.79
GENENET	93.60	1.22	93.44	1.21	99.00	0.00
ORDER_MCMC	-	-	-	-	-	-
PC	-	-	-	-	-	-

Table 5.8: Runtime comparison for DS-S1:1-15.

	Elapsed [s]		**User [s]**		**CPU %**	
	Median	*Std*	*Median*	*Std*	*Median*	*Std*
CORR	3.39	0.00	3.23	0.01	98	0.82
ARACNE	3.71	0.03	3.50	0.03	98	0.50
GENIE3	1425.82	522.72	19370.65	171.04	1363	281.84
GRNBOOST2	1410.68	612.86	18212.12	237.98	1317	323.19
GLASSO	75.09	7.11	74.93	7.10	99	0.00
PPCOR	3.67	0.03	3.47	0.03	98	0.82
GENENET	-	-	-	-	-	-
ORDER_MCMC	-	-	-	-	-	-
PC	-	-	-	-	-	-

Table 5.9: Overall directed score comparison for DS-S2:1-3.

5.1.3 TCGA

Ultimately, methods need to infer networks from real biological data such as the one from TCGA. The runtimes for the other datasets imply that the only ones which could produce results in less than a day on the given machine are CORR, ARACNE, GLASSO and PPCOR. However, the available RAM is insufficient to handle the TCGA dataset with the given setup. Hence, it would be necessary to deploy Docker Desktop on a machine with greater RAM or to distribute the work across many machines. This demonstrates that improving the runtime and memory complexity of GRN inference algorithms is crucial for their application to real biological data.

5.2 Performance Comparison

To analyze how well the methods perform in terms of prediction quality, the AUROC, AUPR and early precision scores for each algorithm are investigated. For each metric, the benchmarking framework produces a table for each dataset collection with scores for individual datasets, their averages and standard deviations. Tables 5.10 and 5.11 show the aggregated results for the datasets DS-D2:1-5 from DREAM4 for directed and undirected evaluation, respectively.

	AUROC		AUPR		EP	
	Mean	*Std*	*Mean*	*Std*	*Mean*	*Std*
CORR	0.729	0.020	0.140	0.028	0.061	0.077
ARACNE	0.713	0.017	0.150	0.014	0.066	0.084
GENIE3	**0.771**	**0.026**	**0.195**	**0.033**	0.074	0.104
GRNBOOST2	0.671	0.036	0.119	0.028	0.058	0.067
GLASSO	0.679	0.038	0.147	0.026	**0.075**	**0.103**
PPCOR	**0.513**	**0.013**	**0.021**	**0.002**	**0.019**	**0.011**
GENENET	0.692	0.033	0.136	0.017	0.061	0.084
ORDER_MCMC	0.708	0.029	0.139	0.029	0.066	0.083
PC	0.674	0.038	0.147	0.036	0.070	0.096

Table 5.10: Overall directed score comparison for DS-D2:1-5.

	AUROC		AUPR		EP	
	Mean	*Std*	*Mean*	*Std*	*Mean*	*Std*
CORR	0.732	0.018	0.261	0.055	0.107	0.098
ARACNE	0.714	0.018	0.270	0.036	0.101	0.109
GENIE3	0.735	0.038	**0.319**	**0.049**	0.107	0.129
GRNBOOST2	**0.741**	**0.022**	0.281	0.051	0.105	0.120
GLASSO	0.680	0.039	0.272	0.056	**0.110**	**0.119**
PPCOR	**0.512**	**0.013**	**0.041**	**0.005**	**0.043**	**0.012**
GENENET	0.693	0.033	0.249	0.041	0.087	0.103
ORDER_MCMC	0.711	0.029	0.282	0.056	0.099	0.104
PC	0.678	0.039	0.258	0.062	0.092	0.111

Table 5.11: Overall undirected score comparison for DS-D2:1-5.

GENIE3 is the clear winner for directed evaluation in terms of AUROC and AUPR scores, whereas GRNBOOST2 is slightly better for undirected evaluation. This indicates that GENIE3 is better at identifying the direction of interactions than GRNBOOST2 for the given dataset. Surprisingly, PPCOR achieves the lowest AUROC score by far, with 0.513 (directed) and 0.512 (undirected), which is almost as bad as a random predictor, resulting in a score of 0.5. However, GLASSO, PPCOR and GENENET rely on the computation of partial correlation and are expected to perform similarly. The same phenomenon can be observed for AUPR values. Interestingly, this differs from DS-S1:1-15 (see Figures 7.5 and 7.7 from the Appendix), where PPCOR is about as good as GENENET and GLASSO. Therefore, PPCOR may have algorithmic issues with high-dimensional datasets. It is worth noting that the results for GENIE3 agree pretty well with its average performance at the DREAM4 competition (directed AUROC: 0.769, directed AUPR: 0.189, undirected AUROC: 0.751, undirected AUPR: 0.314). The discrepancy may arise due to the intrinsic randomness of GENIE3 or slightly different ways of computing the scores. Such performance tables for all dataset collections for which the evaluation is computationally feasible can be found in Section 7.1 in the Appendix. Remember that for the evaluation for DS-S2 and DS-S2-bulk, for all methods, only three datasets and the first 10000 predicted edges are considered, as the evaluation code execution takes too long otherwise. This affects the shape of the curves and the areas under them (see Section 7.2 for further discussion), making it evident that a more efficient implementation for the computation of ROC and PR curves is necessary.

5.2.1 Overall Ranking of Methods

Naturally, tables such as 5.10 and 5.11 can be used to rank methods according to their scores. Here the algorithm with the best score is given a rank of 0 and the others 1,2,3,... accordingly. To deal with methods where no results are available because of scalability issues, the rankings are normalized by dividing each rank by the maximal rank for a given dataset collection. The 'Total' in Table 5.12 is the arithmetic mean of the rankings of the three metrics. This results in a single ranking of all methods, assuming that all metrics are equally important. Table 5.12 shows such normalized ranking for DS-D2:1-5.

	AUROC		**AUPR**		**EP**		**Total**	
	Dir.	*Undir.*	*Dir.*	*Undir.*	*Dir.*	*Undir.*	*Dir.*	*Undir.*
CORR	0.13	0.25	0.50	0.63	0.63	0.13	0.42	0.33
ARACNE	0.25	0.38	0.13	0.50	0.38	0.50	0.25	0.46
GENIE3	**0.00**	0.13	**0.00**	**0.00**	0.13	0.13	**0.04**	**0.08**
GRNBOOST2	0.88	**0.00**	0.88	0.38	0.88	0.38	0.88	0.25
GLASSO	0.63	0.75	0.25	0.25	**0.00**	**0.00**	0.29	0.33
PPCOR	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**
GENENET	0.50	0.63	0.75	0.88	0.75	0.88	0.67	0.79
ORDER_MCMC	0.38	0.50	0.63	0.13	0.38	0.63	0.46	0.42
PC	0.75	0.88	0.38	0.75	0.25	0.75	0.46	0.79

Table 5.12: Normalized rank comparison for DS-D2:1-5.

A value of 0.00 and 1.00 means that the corresponding methods are ranked first and last, respectively. Such rankings can be created for each dataset collection and aggregated to create a ranking of methods across all datasets; see Table 5.13. It is evident that CORR and GENIE3 are overall winners. For undirected evaluation, CORR comes first for every metric, which is surprising as CORR is by far the simplest algorithm. In the total ranking CORR is only surpassed by GENIE3, which was expected as GENIE3 was the winner of the DREAM4 competition. GRNBOOST2 consistently achieves a better ranking for undirected evaluation than for directed one. This indicates that it is good at identifying gene-to-gene associations but not so much at finding directed interactions. Interestingly, all methods relying on computing partial correlations (GLASSO, PPCOR, GENENET) are inferior, where especially PPCOR and GENENET perform very poorly. It is important to note that the algorithms' performance depends on many parameters. Hence, the ranking may look different for more optimized parametrizations of the respective models.

	AUROC		**AUPR**		**EP**		**Total**	
	Dir.	*Undir.*	*Dir.*	*Undir.*	*Dir.*	*Undir.*	*Dir.*	*Undir.*
CORR	**0.15**	**0.10**	**0.20**	**0.17**	0.27	**0.05**	0.21	**0.11**
ARACNE	0.39	0.46	0.29	0.48	0.43	0.42	0.37	0.45
GENIE3	0.19	0.25	0.22	0.21	**0.17**	0.21	**0.19**	0.22
GRNBOOST2	0.64	0.47	0.60	0.47	0.68	0.47	0.64	0.47
GLASSO	0.55	0.54	0.58	0.56	0.58	0.53	0.57	0.54
PPCOR	**0.88**	**0.86**	**0.79**	0.79	**0.73**	0.81	**0.80**	0.82
GENENET	0.63	0.66	0.78	**0.91**	0.71	**0.97**	0.70	**0.84**
ORDER_MCMC	0.46	0.58	0.54	0.46	0.17	0.50	0.39	0.51
PC	0.71	0.78	0.57	0.61	0.61	0.64	0.63	0.68

Table 5.13: Normalized rank comparison for all datasets.

5.2.2 DREAM4 vs. SERGIO-bulk vs SERGIO

to determine whether the ranking of methods is different for different types of data (DREAM4, bulked SERGIO, SERGIO), Table 5.14 may be investigated. It depicts the directed and undirected 'Total' ranking for DREAM4 (DS-D1, DS-D2), bulked SERGIO (DS-S1-bulk, DS-S2-bulk) and full SERGIO (DS-S1, DS-S2) data, where '-' indicates that a method is computationally not feasible for given datasets. GLASSO is an example where the ranking varies greatly among data types. It is the second best algorithm for DREAM4 data and among the worst for full SERGIO data. ORDER_MCMC performs very well for bulked SERGIO data, especially for directed evaluation, but not so much for DREAM4 data. CORR seems to have the most significant relative advantage over other methods for the low-dimensional bulked SERGIO datasets but also performs well on others. GENIE3 is always in the top three and first for DREAM and full SERGIO data for directed evaluation.

	Directed			**Undirected**		
	D	*S-bulk*	*S*	*D*	*S-bulk*	*S*
CORR	0.33	**0.06**	0.22	**0.19**	**0.00**	**0.13**
ARACNE	0.31	0.36	0.44	0.46	0.58	0.32
GENIE3	**0.19**	0.30	**0.09**	0.25	0.25	0.17
GRNBOOST2	0.88	0.73	0.32	0.42	0.51	0.48
GLASSO	0.25	0.62	0.83	0.25	0.54	0.83
PPCOR	**0.90**	0.66	0.84	**0.98**	0.67	0.82
GENENET	0.52	**0.83**	**0.94**	0.75	**0.88**	**1.00**
ORDER_MCMC	0.54	0.08	-	0.60	0.33	-
PC	0.54	0.72	-	0.58	0.77	-

Table 5.14: Normalized total ranking for different data sources.
(D = {DS-D1, DS-D2}, S-bulk = {DS-S1-bulk, DS-S2-bulk}, S = {DS-S1, DS-S2})

5.3 Analysis of ROC and PR Curves

One can look at the ROC and PR curves to gain more detailed insights regarding prediction performance. Figure 5.1 shows all curves obtained for the prediction with DS-D2:1-5. Other examples can be found in the Appendix in Chapter 7.2. Each data point in the curves represents one threshold for edge weights and the corresponding network. The curves nicely show the differences between directed and undirected prediction performance. For the undirected PR curve, the Precision is significantly greater for most recall values than for the directed one. For GENIE3, one can see that the undirected ROC curve ascends rather quickly and then flattens, which is not the case for the directed ROC curve. This indicates that GENIE3 is better at identifying undirected interactions at a low false positive rate than directed ones, which is confirmed by the greater early precision score for undirected evaluation. Such information is lost by only considering the AUROC score of 0.75 for both directed and undirected evaluation. Also the poor performance of PPCOR can be observed. The ROC curves for the PC and GLASSO algorithm demonstrate cases where not all ground truth edges are contained in the predicted list of edges. In theory, FPR = 1 and FPR = 1 can only be obtained when $FN = 0$ and $TN = 0$, respectively, which only holds when all possible edges are contained in the predicted network. Nevertheless, it is always a point on the curve by default, despite not always being computed from the results. As GLASSO is run with $\rho_1 = 0.001$ and $\rho_2 = 0.002$, not all possible edges are in the predicted network. Hence, for the directed ROC curve the point with the greatest possible TPR is (FPR = 0.11, TPR = 0.39), which indicates that only 39% of all true edges are in the predicted set of GLASSO. The linear interpolation between this point and (FPR = 1, TPR = 1) visualizes this. Therefore, it is essential to run methods with sparsity parameters that allow the prediction of many edges. Otherwise, too few data points would be available for drawing the curves, which makes the interpretation of corresponding AUROC and AUPR scores problematic. Furthermore, looking at the point on PR curves with Recall= 1, which

usually corresponds to the data point for the edge weight threshold resulting in the least sparse network, one can read off the number of edges contained in the ground truth network. This is why there the Precision for directed evaluation is half as large as the one for undirected evaluation, as there are double as many possible edges for directed networks as for undirected ones.

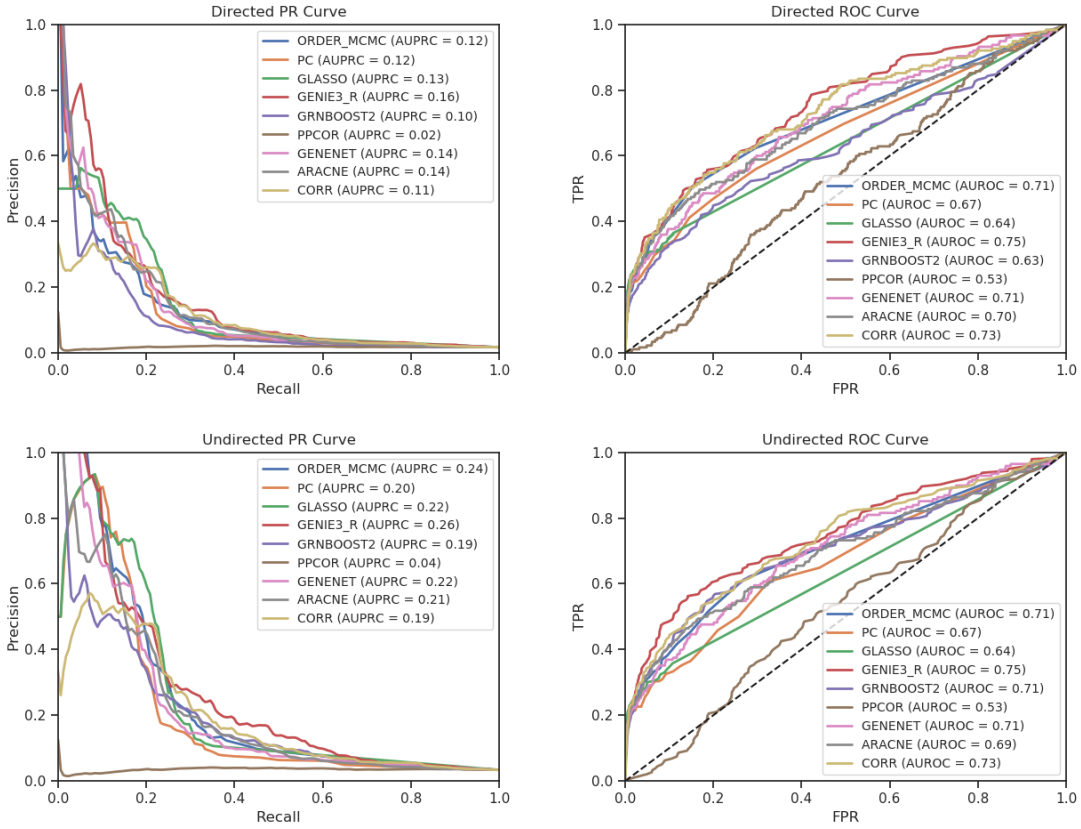

Figure 5.1: Directed and undirected ROC and PR curves for DS-D2:1

As mentioned in the previous section, the evaluation of DS-S2 and DS-S2-bulk is limited to the first 10000 predicted edges; otherwise, it takes several hours for only one dataset. Figure 5.2 shows the different directed ROC curves with the first 10000 predicted edges only (left) and with all edges (right), respectively. In the left plot, one can see how limiting the size of the predicted network also caps the maximal possible TPR as previously pointed out. This visually demonstrates how the sparsity of inferred GRNs affects the shape of the ROC curve and also the area under the curve. GENIE3 only reaches an AUROC score of 0.58, limiting to 10000 edges, whereas it achieves 0.66 without. Similar observations can be made for PR curves.

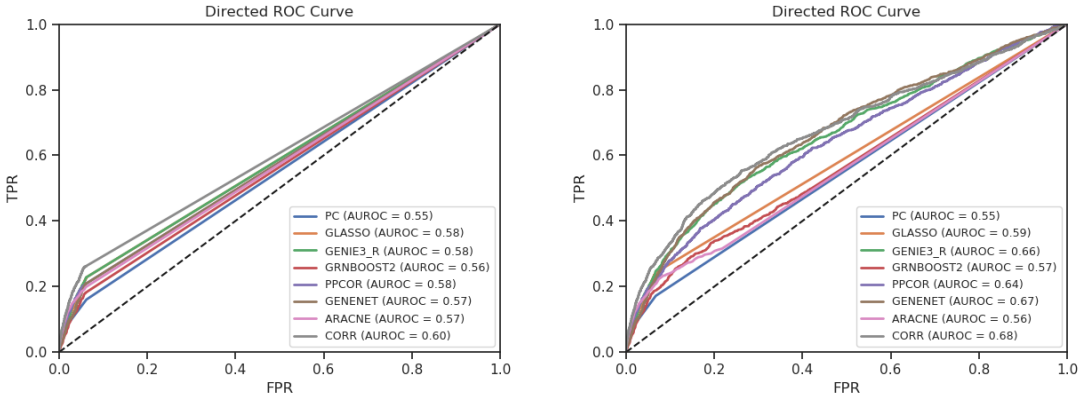

Figure 5.2: Comparison of directed ROC curve for DS-S2-bulk.
(left: length of predicted edge list limited to 10000, right: all predicted edges.)

The analyses from above show that it is vital to investigate ROC and PR curves for better interpretation of the AUROC and AUPR scores and to find additional information about the prediction behavior of the methods. ROC and PR curves can also be used to pick the final network that will be used for its application. For instance, if a scientist considers high Precision more important than Recall, they may choose the edge weight threshold corresponding to a point close to the y-axis on a PR curve.

5.4 Stability and Similarity

The Jaccard indices can be used to assess the stability and similarity of methods. The following subsections demonstrate how the results from the evaluation framework can be analyzed to gain insights.

5.4.1 Jaccard Indices Across Datasets

The Jaccard indices across datasets generated from the same ground truth help evaluate how consistently a method predicts and also how variable simulators are. Table 7.15 shows the median pairwise Jaccard indices for DS-S1:1-15. Each of the 15 datasets in DS-S1 is generated by SERGIO based on the same ground truth network. As the simulator is stochastic in nature, the produced gene expression datasets differ slighly but should contain the same information about the underlying network. The pairwise Jaccard index measures how similar the predictions of a certain method are for any two of the datasets. Low values can indicate that method predictions vary greatly despite similar underlying networks, that the simulator produces very different gene expression values or a combination of both. As the variation in simulation is method-independent, one can use a relative comparison of the indices for evaluating the ranking of methods in terms of their prediction stability. The other tables can be found in the Appendix in Chapter 7.4. The table shows that ARACNE is the most stable, whereas PPCOR and GENENET are inconsistent.

	Jaccard Index	
	Dir.	*Undir.*
CORR	0.183	0.255
ARACNE	**0.200**	**0.272**
GENIE3	0.142	0.179
GRNBOOST2	0.103	0.141
GLASSO	0.142	0.178
PPCOR	0.040	0.072

GENENET	**0.040**	**0.070**
ORDER_MCMC	-	-
PC	-	-

Table 5.15: Median pairwise Jaccard indices for DS-S1:1-15.

5.4.2 Jaccard Indices Across Methods

Jaccard indices across methods give an impression of how alike certain methods are. The corresponding average Jaccard indices can be visualized with a heatmap as in Figure 5.3 for DS-S1:1-15 and undirected evaluation. Undirected and directed heatmaps for the other dataset collections can be found in the Appendix in Chapter 7.3. High values indicate that methods predict similar sets of edges among the first k edges, where k is the size of the ground truth network. For instance, one can see that the pair with GENIE3 and GRNBOOST2 has a value of 0.7, which makes sense as they both rely on non-linear regression using decision trees. Also 0.7 for ARACNE and CORR is not surprising as they both work with simple metrics for relatedness. GENENET and PPCOR have a relatively high value of 0.48, making sense as they both compute partial correlations between random variables.

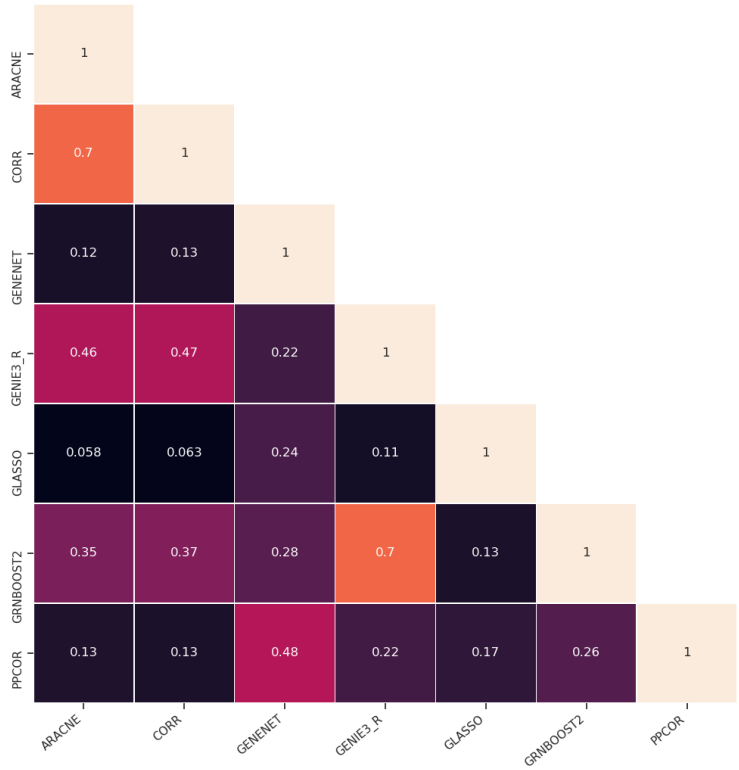

Figure 5.3: Undirected Jaccard indices across methods for DS-S1:1-15.

5.5 Summary

The previous sections show how the evaluation framework can be employed to answer diverse questions about the methods and data. The framework can also be easily extended with additional metrics when new types of questions arise. Some questions which may be answered:

1. How well do the algorithms scale with the number of genes and samples?

Runtime tables from Section 5.1 show that CORR, ARACNE, GLASSO and PPCOR scale very well with the number of genes and samples and can be applied to real large datasets such as cancer gene expression data from TCGA. GENIE3 and GRNBOOST2 take a long time for larger datasets and GENENET, ORDER_MCMC and PC only work on small datasets. One can consider to not only measure user time, elapsed time and CPU percentage but also memory usage. This would provide a more complete picture of how computational resources are used by methods.

2. How do algorithms compare in terms of prediction performance overall, for undirected vs directed evaluation, and for different types of data?

Tables 5.13 and 5.14 from Section 5.2 offer one way to answer this question. They summarize all evaluation tables by ranking the methods in a normalized fashion. Of course, this is just one out of many ways to do so. It appears that GENIE3 is the winner for directed inference and that data-driven methods tend to perform better than probabilistic ones.

3. How does one pick the final edge weight threshold for the predicted GRN of a method?

By investigating ROC and PR curves like those discussed in Section 5.3, the properties of differently sparse networks can be better understood and a threshold chosen accordingly. Naturally, they provide more information than just the area under the curve and show how AUROC and AUPR scores may be interpreted.

4. Given multiple datasets generated from the same ground truth, how stable are the predictions of specific methods?

For instance, Table 7.15 from Subsection 5.4.1 shows that ARACNE is the most stable method for DS-S1 in terms of median pairwise Jaccard indices. Similarly, the stability can be investigated for any other dataset and also incorporated into the performance ranking.

5. How similar are the predicted networks of different methods?

Heatmaps like the one from Subsection 5.4.2 show how similar the first k (number of edges found in ground truth network) predicted edges are across different methods. They can be used to spot abnormalities and to assess the viability of ensemble methods where predictions from many methods are aggregated in some way.

Chapter 6

Conclusion

This thesis looked at several aspects of GRN inference, a mature field of research with critical applications in biomedicine. Naturally, only a broad brush introduction to the field can be given, as it is such a vast and rich research area. After describing biological, mathematical, and computational considerations, available data and methods were conceptually structured. The discussed methods rely on ideas ranging from simple associational concepts such as Pearson correlation to the more complex probabilistic Graphical Causal Models. To quantitatively evaluate how well a method works, knowing the ground truth network underlying given data is essential. As these are generally not known for real biological datasets, it is necessary to have simulators such as the GeneNetWeaver or SERGIO to generate different gene expression data. This thesis provides a reproducible and easily extensible benchmarking framework for GRN inference methods, which allows a systematic comparison of predictions of algorithms in terms of runtime, quality, stability, and similarity. An exemplary selection of methods was picked and applied to simulated and real datasets to demonstrate this. The evaluation metrics produced by the framework could be successfully employed to answer diverse questions regarding the results.

Several improvements can be made for future use. First, a more systematic generation of datasets would engender a better investigation of the impact of simulation parameters on method runtime and performance. SERGIO and GeneNetWeaver require the user to provide a ground truth network inspired by real organisms, making it challenging to generate many arbitrarily large datasets. Second, the framework may also be extended to include signed directed evaluation, where the type of interaction (activation vs. repression) is also assessed. Third, the evaluation code for large networks needs to be made more efficient and methods parallelized. Lastly, it would be desirable to deploy the benchmarking framework on a high-performance cluster as it became evident that the efficiency of algorithms and extensive computational resources are essential for handling real biological datasets. Furthermore, it ought to be further investigated how the method-dependent and method-independent approaches to the computation of ROC and PR curves discussed in Section 4.2 differ.

While a lot of progress has been made in the field of GRN inference, there are still myriads of potential methods and types of data which ought to be investigated further. To make this research as effective as possible, reproducible benchmarking frameworks are crucial. It significantly facilitates comparisons with state-of-the-art algorithms and engenders a more systematic analysis of the impact of simulation and model parameters.

Chapter 7

Appendix

7.1 Overall Performance Comparisons

7.1.1 DS-D1:1-5

	AUROC		AUPR		EP	
	Mean	*Std*	*Mean*	*Std*	*Mean*	*Std*
CORR	0.649	0.101	**0.329**	**0.127**	0.147	0.066
ARACNE	0.651	0.095	0.299	0.103	0.120	0.068
GENIE3	0.616	0.126	0.284	0.128	**0.198**	**0.059**
GRNBOOST2	0.558	0.086	0.191	0.069	**0.118**	**0.048**
GLASSO	**0.659**	**0.091**	0.289	0.106	0.165	0.087
PPCOR	**0.535**	**0.048**	**0.187**	**0.047**	0.147	0.066
GENENET	0.638	0.090	0.265	0.076	0.168	0.088
ORDER_MCMC	0.596	0.047	0.228	0.074	0.181	0.100
PC	0.618	0.097	0.258	0.090	0.134	0.075

Table 7.1: Overall directed score comparison for DS-D1:1-5.

	AUROC		AUPR		EP	
	Mean	*Std*	*Mean*	*Std*	*Mean*	*Std*
CORR	0.667	0.119	**0.525**	**0.143**	**0.373**	**0.098**
ARACNE	0.664	0.115	0.499	0.158	0.292	0.064
GENIE3	0.638	0.153	0.495	0.164	0.369	0.070
GRNBOOST2	0.623	0.123	0.478	0.154	0.336	0.095
GLASSO	**0.685**	**0.109**	0.508	0.160	0.333	0.158
PPCOR	**0.537**	**0.056**	**0.334**	**0.081**	0.255	0.090
GENENET	0.658	0.107	0.459	0.133	**0.252**	**0.107**
ORDER_MCMC	0.589	0.062	0.402	0.080	0.306	0.113
PC	0.650	0.135	0.519	0.128	0.316	0.161

Table 7.2: Overall undirected score comparison for DS-D1:1-5.

7.1.2 DS-D2:1-5

	AUROC		**AUPR**		**EP**	
	Mean	*Std*	*Mean*	*Std*	*Mean*	*Std*
CORR	0.729	0.020	0.140	0.028	0.061	0.077
ARACNE	0.713	0.017	0.150	0.014	0.066	0.084
GENIE3	**0.771**	**0.026**	**0.195**	**0.033**	0.074	0.104
GRNBOOST2	0.671	0.036	0.119	0.028	0.058	0.067
GLASSO	0.679	0.038	0.147	0.026	**0.075**	**0.103**
PPCOR	**0.513**	**0.013**	**0.021**	**0.002**	**0.019**	**0.011**
GENENET	0.692	0.033	0.136	0.017	0.061	0.084
ORDER_MCMC	0.708	0.029	0.139	0.029	0.066	0.083
PC	0.674	0.038	0.147	0.036	0.070	0.096

Table 7.3: Overall directed score comparison for DS-D2:1-5.

	AUROC		**AUPR**		**EP**	
	Mean	*Std*	*Mean*	*Std*	*Mean*	*Std*
CORR	0.732	0.018	0.261	0.055	0.107	0.098
ARACNE	0.714	0.018	0.270	0.036	0.101	0.109
GENIE3	0.735	0.038	**0.319**	**0.049**	0.107	0.129
GRNBOOST2	**0.741**	**0.022**	0.281	0.051	0.105	0.120
GLASSO	0.680	0.039	0.272	0.056	**0.110**	**0.119**
PPCOR	**0.512**	**0.013**	**0.041**	**0.005**	**0.043**	**0.012**
GENENET	0.693	0.033	0.249	0.041	0.087	0.103
ORDER_MCMC	0.711	0.029	0.282	0.056	0.099	0.104
PC	0.678	0.039	0.258	0.062	0.092	0.111

Table 7.4: Overall undirected score comparison for DS-D2:1-5.

7.1.3 DS-S1:1-15

	AUROC		**AUPR**		**EP**	
	Mean	*Std*	*Mean*	*Std*	*Mean*	*Std*
CORR	**0.721**	**0.012**	0.109	0.013	0.185	0.019
ARACNE	0.718	0.014	0.107	0.013	0.184	0.016
GENIE3	0.710	0.022	**0.111**	**0.014**	**0.187**	**0.019**
GRNBOOST2	0.695	0.021	0.094	0.009	0.155	0.016
GLASSO	0.583	0.012	0.048	0.004	0.091	0.012
PPCOR	0.572	0.015	0.047	0.007	**0.074**	**0.013**
GENENET	**0.569**	**0.012**	**0.045**	**0.005**	0.076	0.015
ORDER_MCMC	-	-	-	-	-	-
PC	-	-	-	-	-	-

Table 7.5: Overall directed score comparison for DS-S1:1-15.

	AUROC		**AUPR**		**EP**	
	Mean	*Std*	*Mean*	*Std*	*Mean*	*Std*
CORR	**0.728**	**0.012**	**0.218**	**0.026**	**0.268**	**0.025**
ARACNE	0.724	0.015	0.214	0.025	0.265	0.021
GENIE3	0.656	0.021	0.169	0.012	0.223	0.015

GRNBOOST2	0.634	0.020	0.136	0.011	0.189	0.014
GLASSO	0.585	0.012	0.097	0.008	0.145	0.017
PPCOR	0.574	0.015	0.095	0.013	0.119	0.016
GENENET	**0.570**	**0.012**	**0.090**	**0.011**	**0.115**	**0.013**
ORDER_MCMC	-	-	-	-	-	-
PC	-	-	-	-	-	-

Table 7.6: Overall undirected score comparison for DS-S1:1-15.

7.1.4 DS-S2:1-3

	AUROC		**AUPR**		**EP**	
	Mean	*Std*	*Mean*	*Std*	*Mean*	*Std*
CORR	0.663	0.002	0.072	0.003	0.164	0.005
ARACNE	0.660	0.003	0.065	0.001	0.155	0.005
GENIE3	**0.701**	**0.004**	0.136	0.005	**0.238**	**0.002**
GRNBOOST2	0.700	0.006	**0.155**	**0.007**	0.164	0.116
GLASSO	**0.517**	**0.002**	**0.009**	**0.000**	**0.022**	**0.004**
PPCOR	0.535	0.002	0.014	0.001	0.036	0.002
GENENET	-	-	-	-	-	-
ORDER_MCMC	-	-	-	-	-	-
PC	-	-	-	-	-	-

Table 7.7: Overall directed score comparison for DS-S2:1-3.

	AUROC		**AUPR**		**EP**	
	Mean	*Std*	*Mean*	*Std*	*Mean*	*Std*
CORR	0.708	0.001	0.156	0.006	0.241	0.004
ARACNE	0.706	0.003	0.141	0.003	0.236	0.000
GENIE3	**0.717**	**0.004**	**0.195**	**0.004**	**0.275**	**0.007**
GRNBOOST2	0.704	0.001	0.183	0.003	0.175	0.124
GLASSO	**0.523**	**0.003**	**0.018**	**0.001**	**0.036**	**0.002**
PPCOR	0.549	0.001	0.029	0.001	0.060	0.006
GENENET	-	-	-	-	-	-
ORDER_MCMC	-	-	-	-	-	-
PC	-	-	-	-	-	-

Table 7.8: Overall undirected score comparison for DS-S2:1-3.

7.1.5 DS-S1-bulk:1-15

	AUROC		**AUPR**		**EP**	
	Mean	*Std*	*Mean*	*Std*	*Mean*	*Std*
CORR	0.648	0.017	0.052	0.006	0.084	0.013
ARACNE	0.536	0.015	0.037	0.005	0.068	0.015
GENIE3	**0.653**	**0.018**	0.047	0.004	0.063	0.011
GRNBOOST2	0.564	0.020	0.036	0.004	0.054	0.015
GLASSO	0.549	0.013	0.033	0.001	0.040	0.010
PPCOR	**0.530**	**0.024**	**0.030**	**0.003**	0.040	0.010

GENENET	0.544	0.019	0.030	0.002	**0.037**	**0.014**
ORDER_MCMC	0.596	0.017	**0.053**	**0.009**	**0.099**	**0.019**
PC	0.537	0.012	0.033	0.003	0.048	0.009

Table 7.9: Overall directed score comparison for DS-S1-bulk:1-15.

	AUROC		**AUPR**		**EP**	
	Mean	*Std*	*Mean*	*Std*	*Mean*	*Std*
CORR	**0.652**	**0.018**	**0.652**	**0.018**	**0.150**	**0.022**
ARACNE	0.537	0.015	0.537	0.015	0.115	0.017
GENIE3	0.634	0.020	0.634	0.020	0.106	0.012
GRNBOOST2	0.639	0.044	0.639	0.044	0.143	0.049
GLASSO	0.551	0.013	0.551	0.013	0.081	0.011
PPCOR	**0.530**	**0.025**	**0.530**	**0.025**	0.073	0.015
GENENET	0.545	0.020	0.545	0.020	**0.068**	**0.021**
ORDER_MCMC	0.584	0.017	0.584	0.017	0.128	0.015
PC	0.541	0.014	0.541	0.014	0.084	0.011

Table 7.10: Overall undirected score comparison for DS-S1-bulk:1-15.

7.1.6 DS-S2-bulk:1-3

	AUROC		**AUPR**		**EP**	
	Mean	*Std*	*Mean*	*Std*	*Mean*	*Std*
CORR	**0.605**	**0.001**	**0.050**	**0.001**	**0.119**	**0.007**
ARACNE	0.567	0.004	0.036	0.002	0.089	0.003
GENIE3	0.592	0.000	0.037	0.002	0.091	0.007
GRNBOOST2	0.563	0.007	**0.024**	**0.002**	**0.045**	**0.005**
GLASSO	0.585	0.003	0.033	0.000	0.078	0.001
PPCOR	0.572	0.003	0.034	0.004	0.082	0.006
GENENET	-	-	-	-	-	-
ORDER_MCMC	-	-	-	-	-	-
PC	**0.553**	**0.001**	0.031	0.003	0.073	0.002

Table 7.11: Overall undirected score comparison for DS-S2-bulk:1-3.

	AUROC		**AUPR**		**EP**	
	Mean	*Std*	*Mean*	*Std*	*Mean*	*Std*
CORR	**0.578**	**0.003**	**0.023**	**0.001**	**0.073**	**0.002**
ARACNE	0.557	0.003	0.018	0.001	0.058	0.002
GENIE3	0.560	0.002	0.015	0.001	0.047	0.002
GRNBOOST2	**0.538**	**0.002**	**0.011**	**0.001**	**0.028**	**0.001**
GLASSO	0.553	0.003	0.014	0.000	0.046	0.004
PPCOR	0.552	0.003	0.016	0.002	0.053	0.004
GENENET	-	-	-	-	-	-
ORDER_MCMC	-	-	-	-	-	-
PC	0.540	0.003	0.014	0.002	0.042	0.002

Table 7.12: Overall directed score comparison for DS-S2-bulk:1-3.

7.2 ROC and PR Curves

7.2.1 DS-D1:5

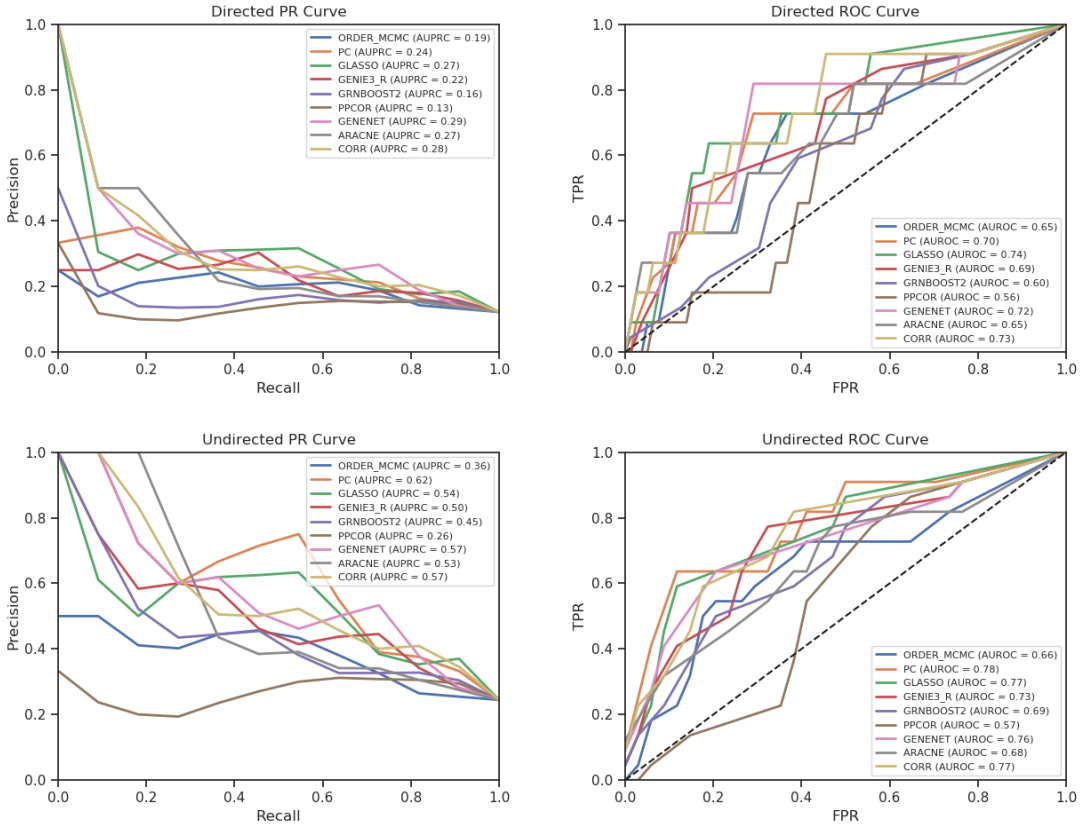

Figure 7.1: Directed and undirected ROC and PR curves for DS-D1:5

7.2.2 DS-D2:1

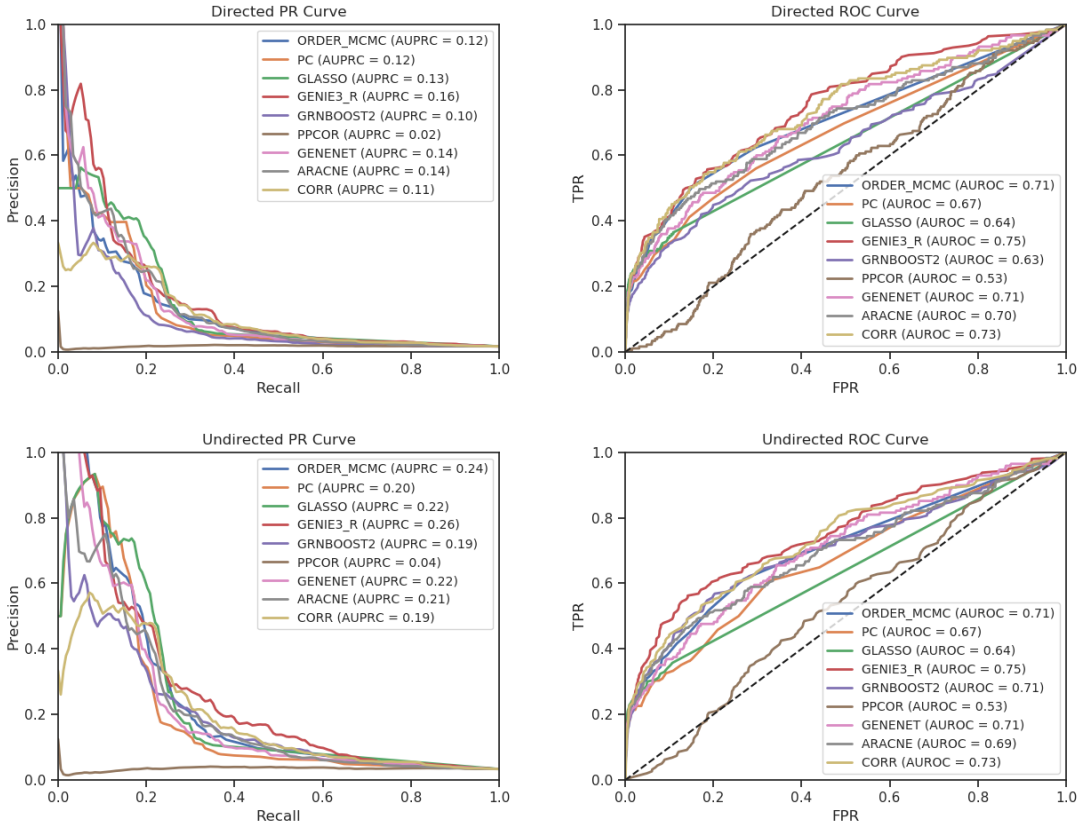

Figure 7.2: Directed and undirected ROC and PR curves for DS-D2:1

7.2.3 DS-S1-bulk:1

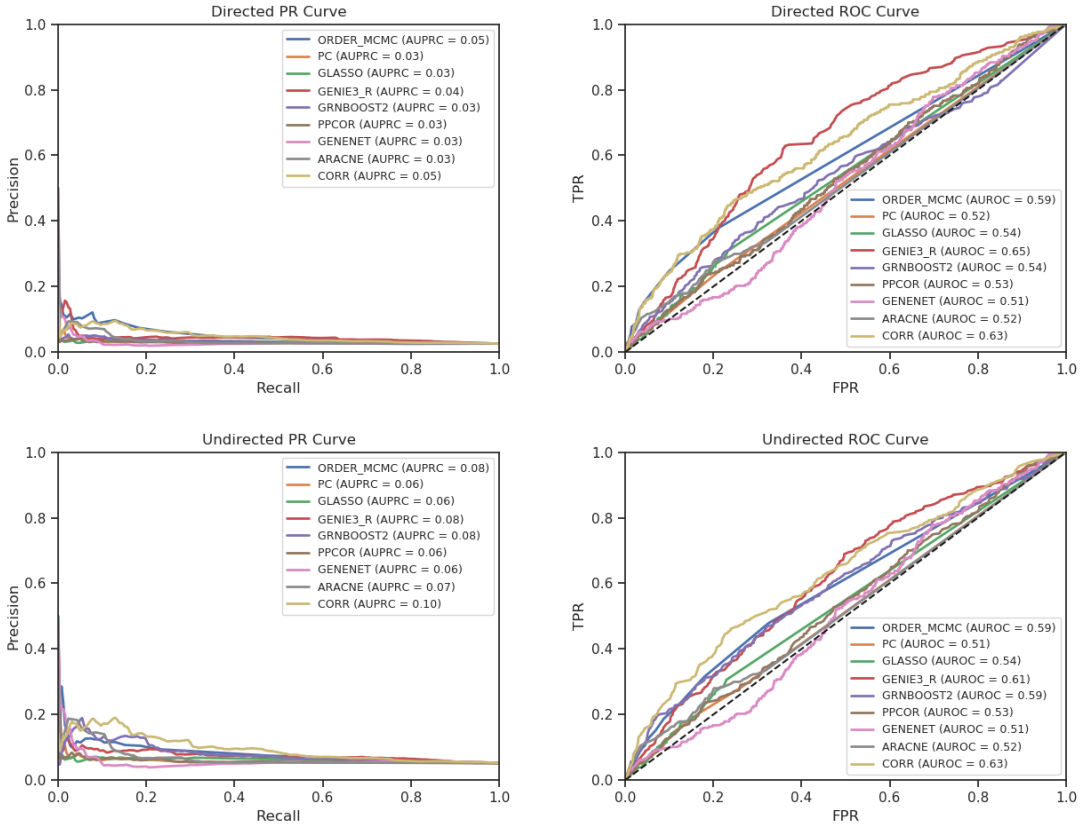

Figure 7.3: Directed and undirected ROC and PR curves for DS-S1-bulk:1

7.2.4 DS-S2-bulk:1

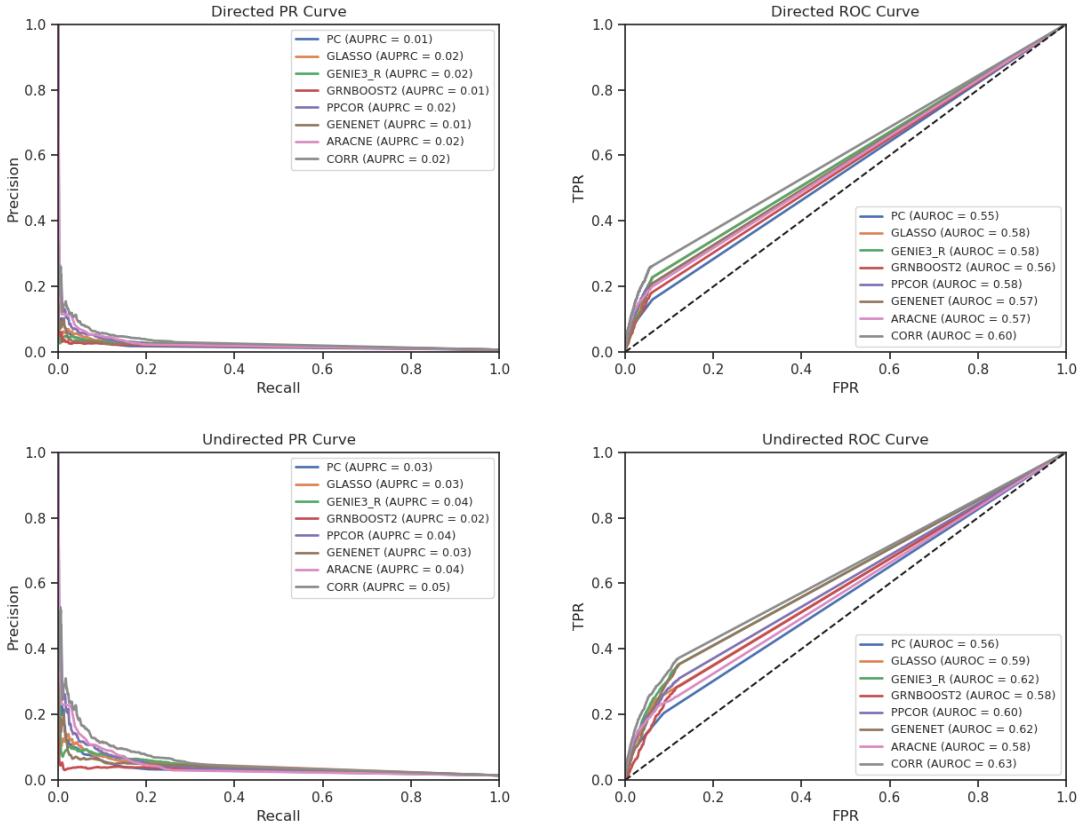

Figure 7.4: Directed and undirected ROC and PR curves for DS-S2-bulk:1

7.2.5 DS-S1:1

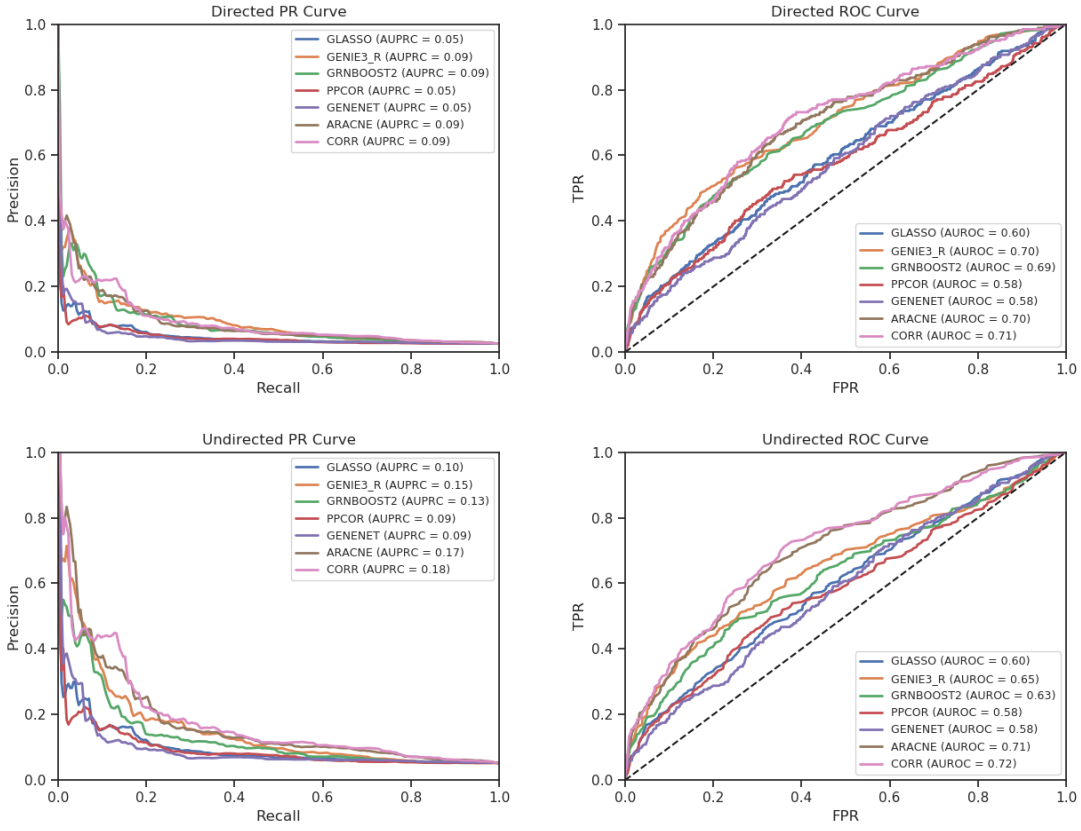

Figure 7.5: Directed and undirected ROC and PR curves for DS-S1:1

7.3 Jaccard Indices Across Methods

7.3.1 DS-D1:1-5

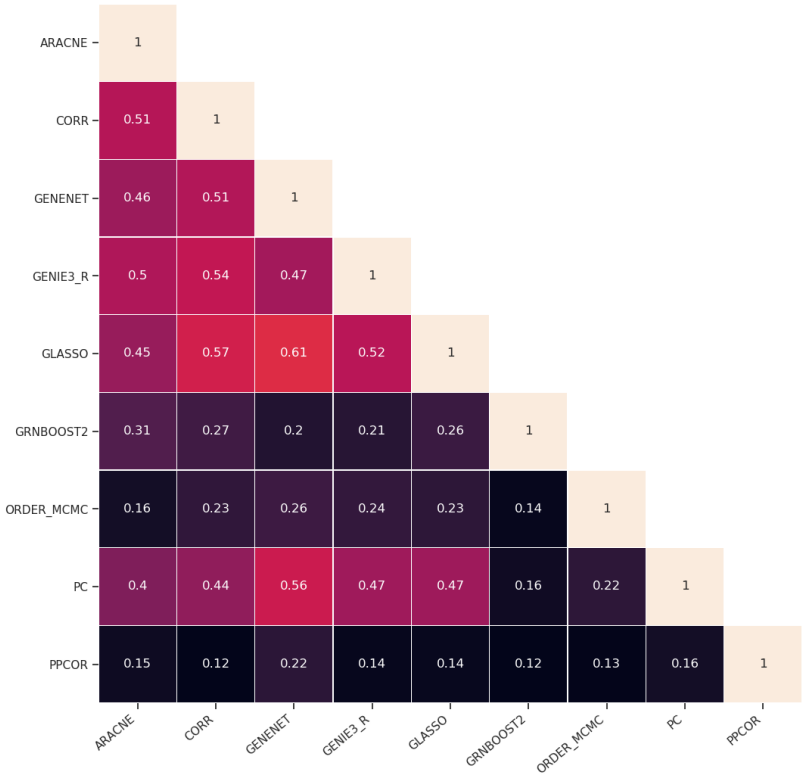

Figure 7.6: Directed Jaccard indices across methods for DS-D1:1-5.

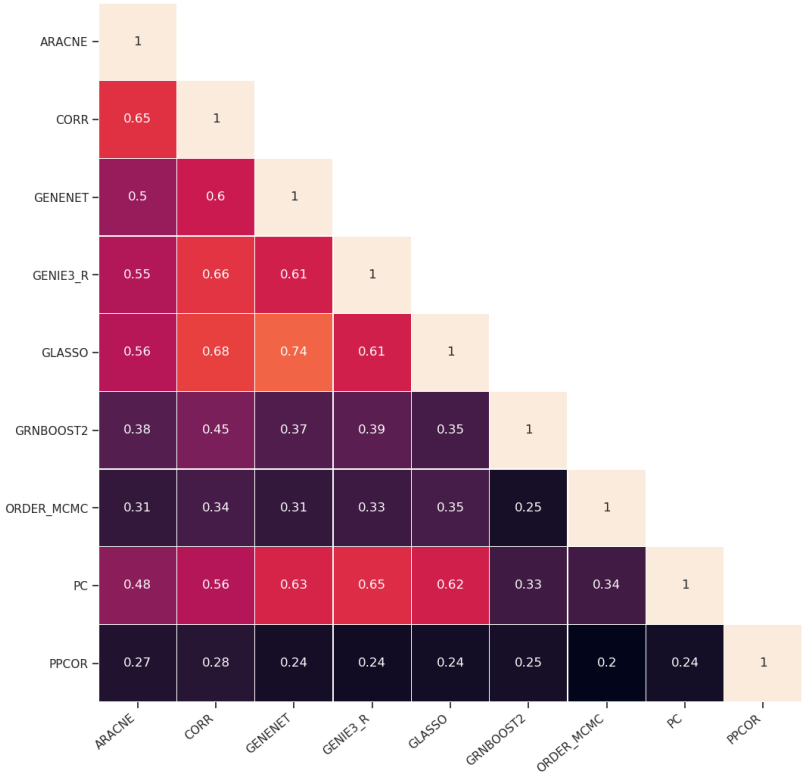

Figure 7.7: Undirected Jaccard indices across methods for DS-D1:1-5.

7.3.2 DS-D2:1-5

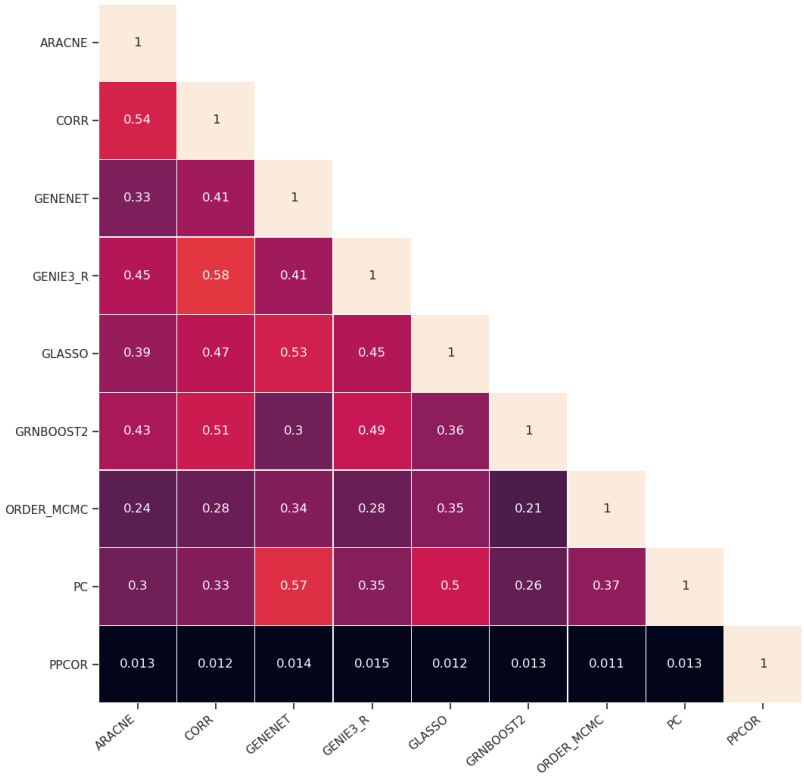

Figure 7.8: Directed Jaccard indices across methods for DS-D2

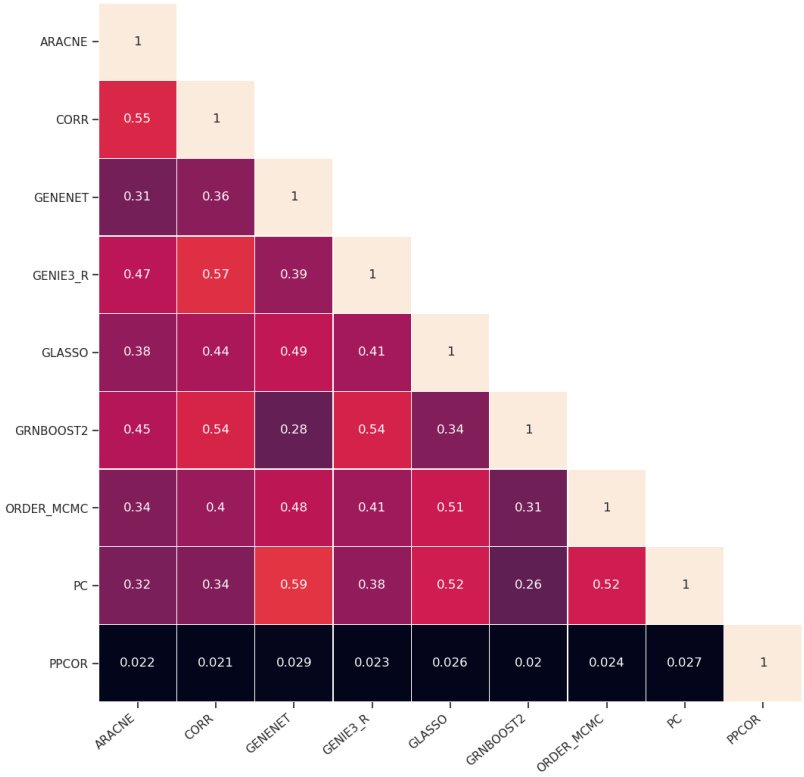

Figure 7.9: Undirected Jaccard indices across methods for DS-D2

7.3.3 DS-S1-bulk:1-15

Jaccard Indices Across Methods

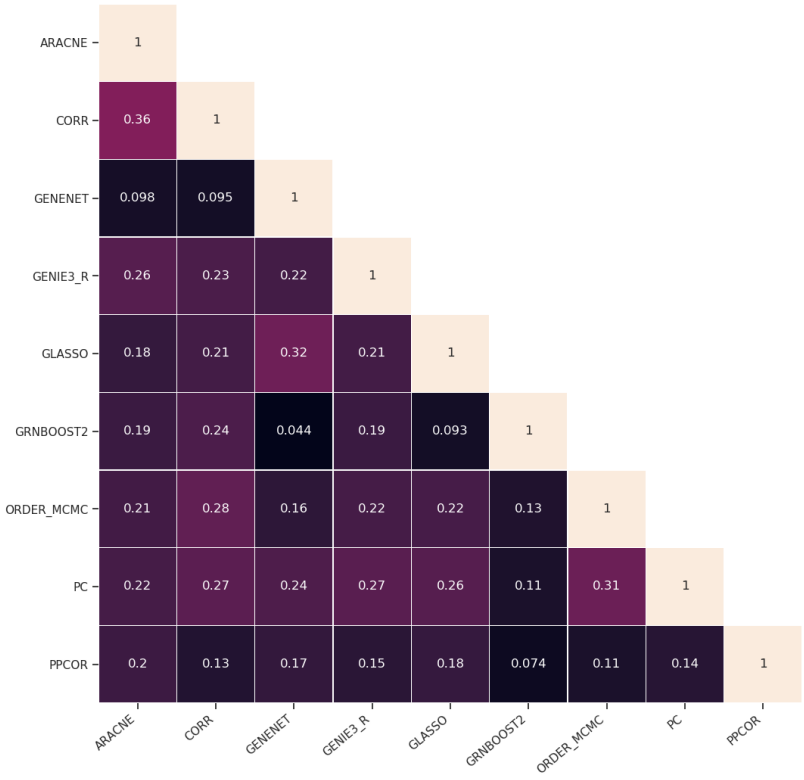

Figure 7.10: Undirected Jaccard indices across methods for DS-S1-bulk:1-15.

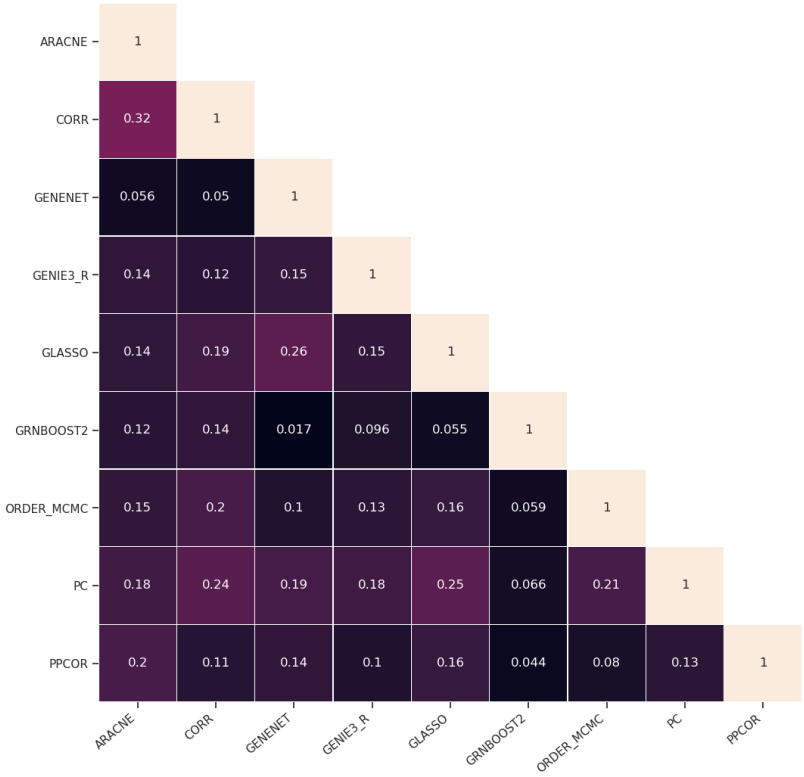

Figure 7.11: Directed Jaccard indices across methods for DS-S1-bulk:1-15.

7.3.4 DS-S2-bulk:1-3

Jaccard Indices Across Methods

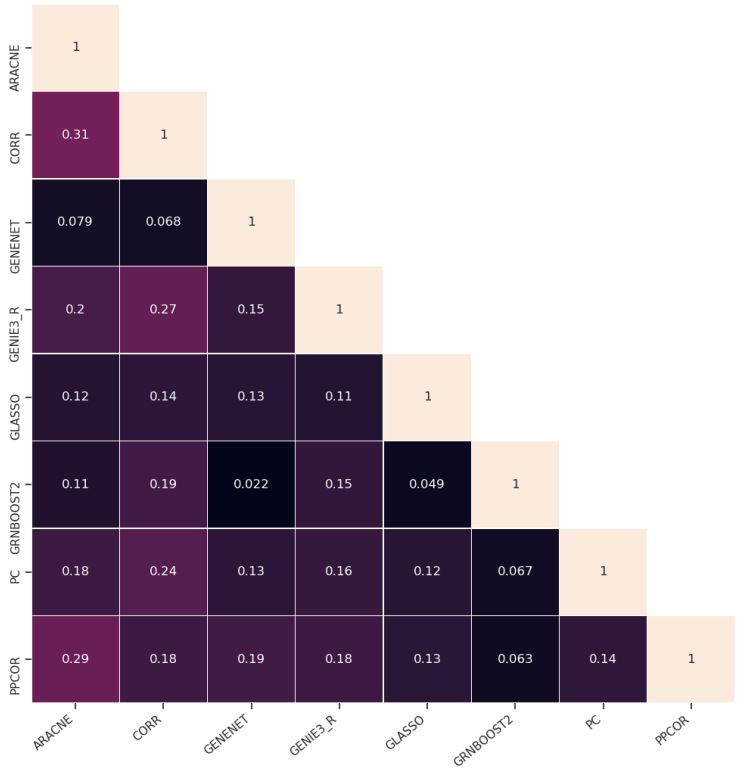

Figure 7.12: Undirected Jaccard indices across methods for DS-S2-bulk:1-3.

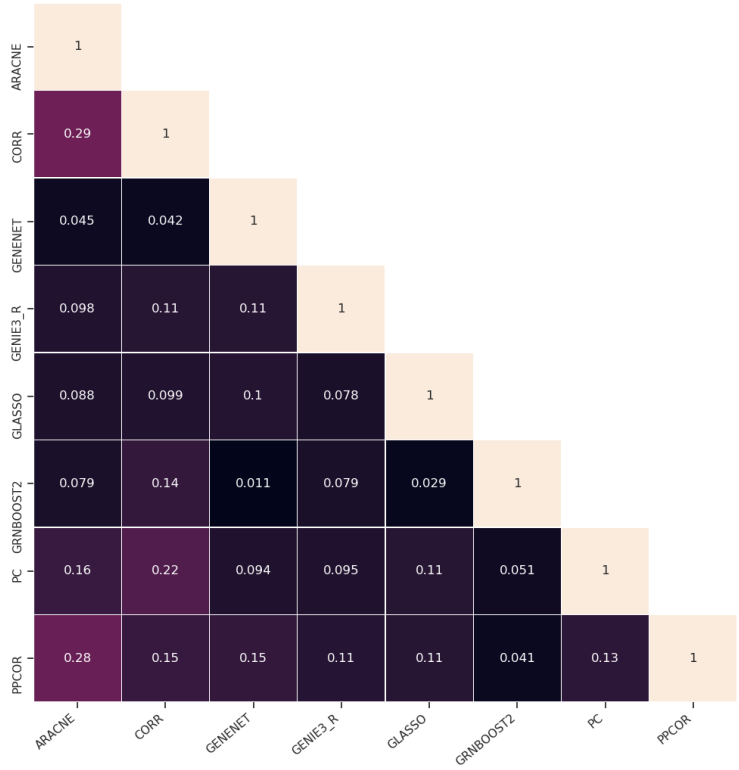

Figure 7.13: Directed Jaccard indices across methods for DS-S2-bulk:1-3.

7.3.5 DS-S1:1-15

Figure 7.14: Undirected Jaccard indices across methods for DS-S1:1-15.

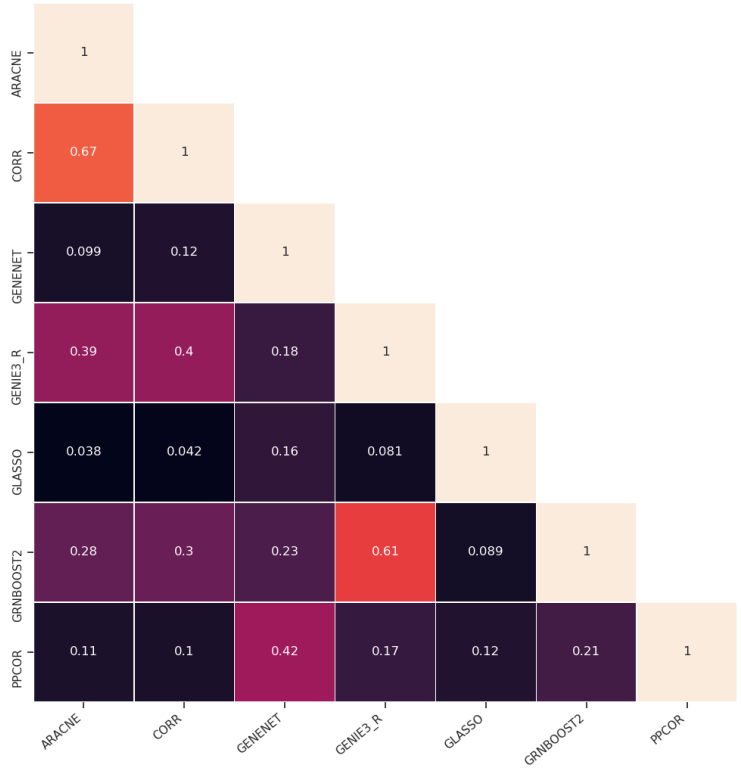

Figure 7.15: Directed Jaccard indices across methods for DS-S1:1-15.

7.3.6 DS-S2:1-3

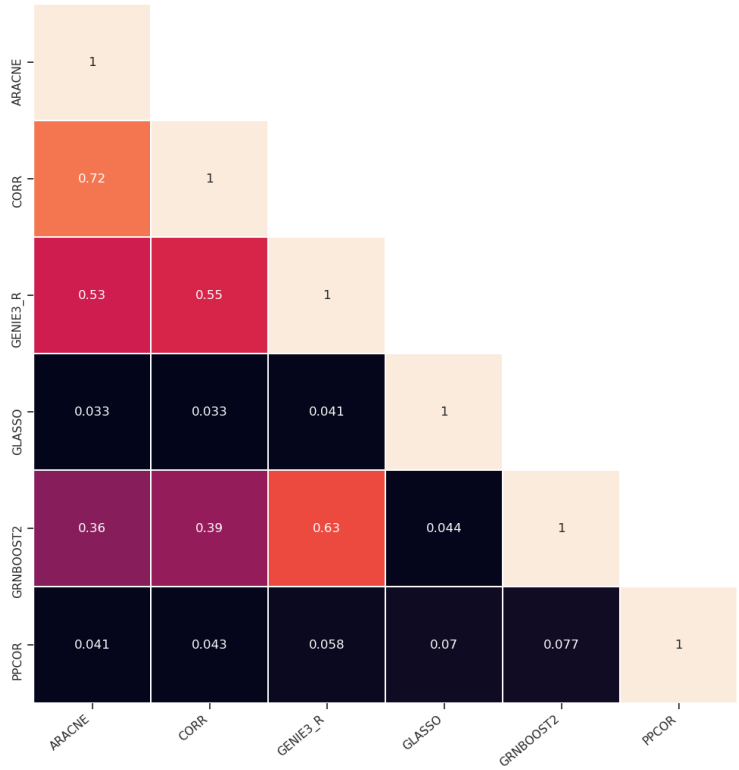

Figure 7.16: Undirected Jaccard indices across methods for DS-S1:1-3.

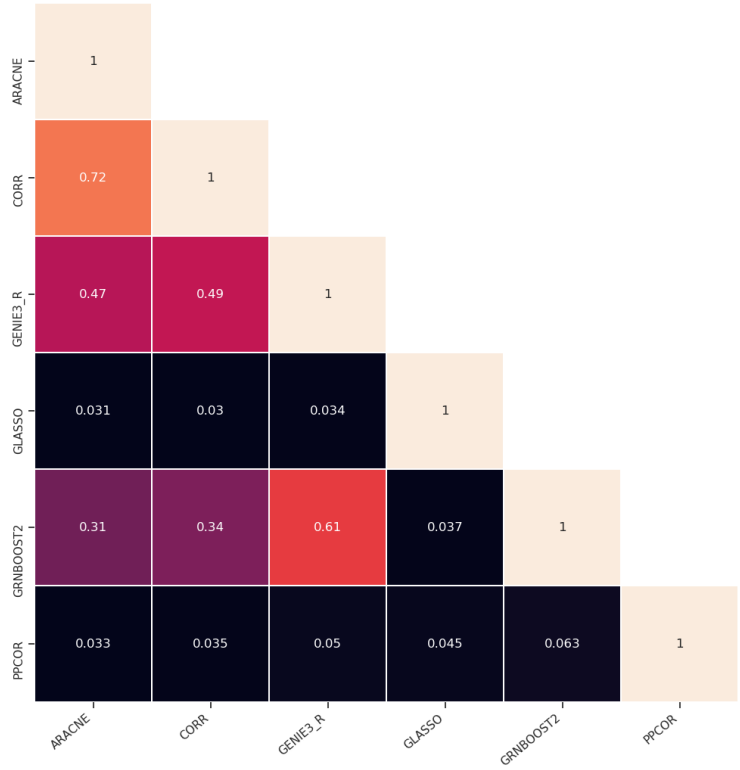

Figure 7.17: Directed Jaccard indices across methods for DS-S2:1-3.

7.4 Jaccard Indices Across Datasets

7.4.1 DS-S1-bulk:1-15

	Jaccard Index	
	Dir.	*Undir.*
CORR	0.045	0.138
ARACNE	0.036	0.098
GENIE3	0.069	0.114
GRNBOOST2	0.028	0.096
GLASSO	0.028	0.075
PPCOR	**0.016**	**0.060**
GENENET	0.036	0.095
ORDER_MCMC	**0.079**	**0.152**
PC	0.034	0.091

Table 7.13: Median pairwise Jaccard indices for DS-S1-bulk:1-15.

7.4.2 DS-S2-bulk:1-3

	Jaccard Index	
	Dir.	*Undir.*
CORR	**0.184**	**0.219**
ARACNE	0.061	0.071
GENIE3	0.060	0.136
GRNBOOST2	0.042	0.097
GLASSO	0.042	0.080
PPCOR	**0.039**	**0.045**
GENENET	-	-
ORDER_MCMC	**-**	**-**
PC	0.039	0.060

Table 7.14: Median pairwise Jaccard indices for DS-S2-bulk:1-3.

7.4.3 DS-S1:1-15

	Jaccard Index	
	Dir.	*Undir.*
CORR	0.183	0.255
ARACNE	**0.200**	**0.272**
GENIE3	0.142	0.179
GRNBOOST2	0.103	0.141
GLASSO	0.142	0.178
PPCOR	0.040	0.072
GENENET	**0.040**	**0.070**
ORDER_MCMC	-	-
PC	-	-

Table 7.15: Median pairwise Jaccard indices for DS-S1:1-15.

7.4.4 DS-S2:1-3

	Jaccard Index	
	Dir.	*Undir.*
CORR	0.377	0.332
ARACNE	**0.423**	**0.365**
GENIE3	0.326	0.275
GRNBOOST2	0.206	0.176
GLASSO	0.048	0.085
PPCOR	**0.011**	**0.023**
GENENET	-	-
ORDER_MCMC	-	-
PC	-	-

Table 7.16: Median pairwise Jaccard indices for DS-S2:1-3.

Bibliography

Method of the year 2013, 2014. ISSN 1548-7105. URL https://doi.org/10.1038/nmeth.2801.

Syed Sazzad Ahmed, Swarup Roy, and Jugal Kalita. Assessing the effectiveness of causality inference methods for gene regulatory networks. *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, 17(1):56–70, 2020. doi: 10.1109/TCBB.2018.2853728.

Stig Andersen. Judea pearl, probabilistic reasoning in intelligent systems: Networks of plausible inference. *Artif. Intell.*, 48:117–124, 02 1991. doi: 10.1016/0004-3702(91)90084-W.

Oscar Aparicio, Joseph V. Geisberg, and Kevin Struhl. Chromatin immunoprecipitation for determining the association of proteins with specific genomic sequences in vivo. *Current Protocols in Cell Biology*, 23:17.7.1–17.7.23, 6 2004. ISSN 1934-2616. doi: 10.1002/0471143030.CB1707S23. URL https://onlinelibrary.wiley.com/doi/full/10.1002/0471143030.cb1707s23https://onlinelibrary.wiley.com/doi/abs/10.1002/0471143030.cb1707s23https://currentprotocols.onlinelibrary.wiley.com/doi/10.1002/0471143030.cb1707s23.

Janusz Awek and Tom Arodz. Ennet: inferring large gene regulatory networks from expression data using gradient boosting. *BMC systems biology*, 7:106, 10 2013. doi: 10.1186/1752-0509-7-106.

David Barber. *Bayesian Reasoning and Machine Learning*. Cambridge University Press, 2012. doi: 10.1017/CBO9780511804779.

Attila Becskei and Luis Serrano. Engineering stability in gene networks by autoregulation. *Nature*, 405:590–593, 2000. ISSN 1476-4687. doi: 10.1038/35014651. URL https://doi.org/10.1038/35014651.

Niko Beerenwinkel and Juliane Siebourg. *Probability, Statistics, and Computational Science*, pages 77–110. Humana Press, 2012. ISBN 978-1-61779-582-4. doi: 10.1007/978-1-61779-582-4_3. URL https://doi.org/10.1007/978-1-61779-582-4_3.

Jan Beirlant, Edward J. Dudewicz, László Györfi, and Istvan Denes. Nonparametric entropy estimation. an overview. In *Nonparametric entropy estimation. An overview*, 1997.

Archana Belle, Amos Tanay, Ledion Bitincka, Ron Shamir, and Erin K. O'Shea. Quantification of protein half-lives in the budding yeast proteome. *Proceedings of the National Academy of Sciences of the United States of America*, 103:13004–13009, 8 2006. ISSN 00278424. doi: 10.1073/pnas.0605420103.

Pau Bellot, Philippe Salembier, Ngoc C Pham, and Patrick E Meyer. Chapter 12 unsupervised grn ensemble. *Methods in Molecular Biology*, 2019. doi: 10.1007/978-1-4939-8882-2_12. URL https://doi.org/10.1007/978-1-4939-8882-2_12,.

Garry W. Blakely. A genetic switch, third edition, phage lambda revisited. m. ptashne. cold spring harbor laboratory press. 2004. 154 pages. isbn 0 87969 716 4. price $39 (paperback). *Genetical Research*, 84(3):193–194, 2004. doi: 10.1017/S0016672304227276.

Kevin Bleakley, Gérard Biau, and Jean Philippe Vert. Supervised reconstruction of biological networks with local models. *Bioinformatics*, 23, 7 2007. ISSN 13674803. doi: 10.1093/bioinformatics/btm204.

Leo Breiman. Random forests. *Machine Learning*, 45:5–32, 2001. ISSN 1573-0565. doi: 10.1023/A:1010933404324. URL https://doi.org/10.1023/A:1010933404324.

Atul Janardhan Butte and Isaac S. Kohane. Mutual information relevance networks: functional genomic clustering using pairwise entropy measurements. *Pacific Symposium on Biocomputing. Pacific Symposium on Biocomputing*, pages 418–29, 2000.

Peter Bühlmann and Sara A. van de Geer. Statistics for high-dimensional data: Methods, theory and applications. In *Statistics for High-Dimensional Data: Methods, Theory and Applications*, 2011.

Emmanuel Candes and Terence Tao. The Dantzig selector: Statistical estimation when p is much larger than n. *The Annals of Statistics*, 35(6):2313 – 2351, 2007. doi: 10.1214/009053606000001523. URL https://doi.org/10.1214/009053606000001523.

João Carneiro. *The Role of Non-coding DNA Structural Information in Phylogeny, Evolution and Disease*. PhD thesis, University of Porto, 07 2013.

Thalia E. Chan, Michael P.H. Stumpf, and Ann C. Babtie. Gene regulatory network inference from single-cell data using multivariate information measures. *Cell Systems*, 5(3):251–267.e3, 2017. ISSN 2405-4712. doi: https://doi.org/10.1016/j.cels.2017.08.014. URL https://www.sciencedirect.com/science/article/pii/S2405471217303861.

Julien Chiquet, Yves Grandvalet, Christophe Ambroise, J Chiquet, C Ambroise, and Y Grandvalet. Inferring multiple graphical structures. *Statistics and Computing 2010 21:4*, 21:537–553, 6 2010. ISSN 1573-1375. doi: 10.1007/S11222-010-9191-2. URL https://link.springer.com/article/10.1007/s11222-010-9191-2.

Francis Crick. Central dogma of molecular biology. *Nature*, 227:561–563, 1970. ISSN 1476-4687. doi: 10.1038/227561a0. URL https://doi.org/10.1038/227561a0.

Jesse Davis and Mark Goadrich. The relationship between precision-recall and roc curves. In *Proceedings of the 23rd International Conference on Machine Learning*, ICML '06, page 233–240, New York, NY, USA, 2006. Association for Computing Machinery. ISBN 1595933832. doi: 10.1145/1143844.1143874. URL https://doi.org/10.1145/1143844.1143874.

Alberto de la Fuente. Gene network inference. In *Springer Berlin Heidelberg*, 2013.

Payam Dibaeinia and Saurabh Sinha. Sergio: A single-cell expression simulator guided by gene regulatory networks. *Cell Systems*, 11:252–271.e11, 9 2020. ISSN 2405-4712. doi: 10.1016/J.CELS.2020.08.003.

Bradley Efron. Nonparametric standard errors and confidence intervals. *Canadian Journal of Statistics-revue Canadienne De Statistique*, 9:139–158, 1981.

Frank Emmert-Streib, Matthias Dehmer, and Benjamin Haibe-Kains. Gene regulatory networks and their applications: understanding biological and medical problems in terms of networks. *Frontiers in Cell and Developmental Biology*, 2, 8 2014. ISSN 2296634X. doi: 10.3389/FCELL.2014.00038. URL /pmc/articles/PMC4207011//pmc/articles/PMC4207011/?report=abstracthttps://www.ncbi.nlm.nih.gov/pmc/articles/PMC4207011/.

Aravind Subramanian et al. Gene set enrichment analysis: a knowledge-based approach for interpreting genome-wide expression profiles. *Proceedings of the National Academy of Sciences of the United States of America*, 102:15545–15550, 10 2005. ISSN 0027-8424. doi: 10.1073/PNAS.0506580102. URL https://pubmed.ncbi.nlm.nih.gov/16199517/.

Sara Aibar et al. Scenic: Single-cell regulatory network inference and clustering. *Nature methods*, 14:1083, 10 2017. ISSN 15487105. doi: 10.1038/NMETH.4463. URL /pmc/articles/PMC5937676//pmc/articles/PMC5937676/?report=abstracthttps://www.ncbi.nlm.nih.gov/pmc/articles/PMC5937676/.

J. Faith, Jeremiah, Boris Hayete, T. Thaden, Joshua, Ilaria Mogno, Jamey Wierzbowski, Guillaume Cottarel, Simon Kasif, J. Collins, James, and S. Gardner, Timothy. Large-scale mapping and validation of escherichia coli transcriptional regulation from a compendium of expression profiles, 2007. URL https://open.bu.edu/handle/2144/2958.

Yue Fan and Xiuli Ma. Gene regulatory network inference using 3d convolutional neural network. In *Thirty-Fifth AAAI Conference on Artificial Intelligence, AAAI 2021, Thirty-Third Conference on Innovative Applications of Artificial Intelligence, IAAI 2021, The Eleventh Symposium on Educational Advances in Artificial Intelligence, EAAI 2021, Virtual Event, February 2-9, 2021*, pages 99–106. AAAI Press, 2021. ISBN 978-1-57735-866-4. URL https://ojs.aaai.org/index.php/AAAI/article/view/16082.

Ronald Aylmer Sir Fisher. 035: The distribution of the partial correlation coefficient. *Metron*, 1924.

Jerome Friedman, Trevor Hastie, and Robert Tibshirani. Sparse inverse covariance estimation with the graphical lasso. *Biostatistics*, 9:432–441, 7 2008. ISSN 1465-4644. doi: 10.1093/biostatistics/kxm045. URL https://doi.org/10.1093/biostatistics/kxm045.

Jerome Friedman, J. Hastie, and R. Tibshirani. *The elements of statistical learning*. Springer New York, NY, 01 2009.

Nir Friedman and Daphne Koller. Being bayesian about network structure. a bayesian approach to structure discovery in bayesian networks. *Machine Learning*, 50:95–125, 2003. ISSN 1573-0565. doi: 10.1023/A:1020249912095. URL https://doi.org/10.1023/A:1020249912095.

Fabian Fröhlich, Carolin Loos, and Jan Hasenauer. Scalable inference of ordinary differential equation models of biochemical processes, 2017. URL https://arxiv.org/abs/1711.08079.

Hernan G Garcia, Jane Kondev, Nigel Orme, Julie A Theriot, and Rob Phillips. Thermodynamics of biological processes. *Methods in enzymology*, 492:27–59, 2011. ISSN 1557-7988. doi: 10.1016/B978-0-12-381268-1.00014-8. URL https://pubmed.ncbi.nlm.nih.gov/21333788https://www.ncbi.nlm.nih.gov/pmc/articles/PMC3264492/.

Daniel T. Gillespie. The chemical langevin equation. *Journal of Chemical Physics*, 113:297–306, 2000.

Clark Glymour, Kun Zhang, and Peter Spirtes. Review of causal discovery methods based on graphical models. *Frontiers in Genetics*, 10:524, 2019. ISSN 16648021. doi: 10.3389/FGENE.2019.00524. URL /pmc/articles/PMC6558187//pmc/articles/PMC6558187/?report=abstracthttps://www.ncbi.nlm.nih.gov/pmc/articles/PMC6558187/.

Jan Grau, Ivo Grosse, and Jens Keilwagen. Prroc: computing and visualizing precision-recall and receiver operating characteristic curves in r. *Bioinformatics*, 31(15):2595–2597, 2015.

Bradford Hall, Advait Limaye, and Ashok B. Kulkarni. Overview: Generation of gene knockout mice. *Current Protocols in Cell Biology*, 44:19.12.1–19.12.17, 9 2009. ISSN 1934-2616. doi: 10.1002/0471143030.CB1912S44. URL https://onlinelibrary.wiley.com/doi/full/10.1002/0471143030.cb1912s44https://onlinelibrary.wiley.com/doi/abs/10.1002/0471143030.cb1912s44https://currentprotocols.onlinelibrary.wiley.com/doi/10.1002/0471143030.cb1912s44.

Trevor Hastie, Robert Tibshirani, and Jerome Friedman. The elements of statistical learning – data mining, inference and prediction, 2013.

Alain Hauser and Peter Bühlmann. Characterization and greedy learning of interventional Markov equivalence classes of directed acyclic graphs. *Journal of Machine Learning Research*, 13:2409–2464, 2012. URL https://jmlr.org/papers/v13/hauser12a.html.

Arthur E Hoerl and Robert W Kennard. Ridge regression: Biased estimation for nonorthogonal problems. *Technometrics*, 12:55–67, 1970. doi: 10.1080/00401706.1970.10488634. URL https://www.tandfonline.com/doi/abs/10.1080/00401706.1970.10488634.

Daniel J. Hogan, Daniel P. Riordan, André P. Gerber, Daniel Herschlag, and Patrick O. Brown. Diverse rna-binding proteins interact with functionally related sets of rnas, suggesting an extensive regulatory system. *PLoS biology*, 6:2297–2313, 10 2008. ISSN 1545-7885. doi: 10.1371/JOURNAL.PBIO.0060255. URL https://pubmed.ncbi.nlm.nih.gov/18959479/.

Sture Holm. Board of the foundation of the scandinavian journal of statistics a simple sequentially rejective multiple test procedure a simple sequentially rejective multiple test procedure. *Source: Scandinavian Journal of Statistics*, 6:65–70, 1979.

Vân Anh Huynh-Thu and Guido Sanguinetti. Gene regulatory network inference: an introductory survey, 2018. URL https://arxiv.org/abs/1801.04087.

Vân Anh Huynh-Thu, Alexandre Irrthum, Louis Wehenkel, and Pierre Geurts. Inferring regulatory networks from expression data using tree-based methods. *PLOS ONE*, 5:e12776, 2010. ISSN 1932-6203. doi: 10.1371/JOURNAL.PONE.0012776. URL https://journals.plos.org/plosone/article?id=10.1371/journal.pone.0012776.

Paul Jaccard. Etude de la distribution florale dans une portion des alpes et du jura. *Bulletin de la Societe Vaudoise des Sciences Naturelles*, 37:547–579, 01 1901. doi: 10.5169/seals-266450.

Michael I. Jordan. Learning in graphical models. *Kluwer Academic Publishers*, 1998. doi: 10.1007/978-94-011-5014-9.

Michael I. Jordan. Graphical models. *Statistical Science*, 19(1):140 – 155, 2004. doi: 10.1214/088342304000000026. URL https://doi.org/10.1214/088342304000000026.

Markus Kalisch and Peter Bühlmann. Estimating high-dimensional directed acyclic graphs with the pc-algorithm peter b ¨ uhlmann. *Journal of Machine Learning Research*, 8:613–636, 2007.

Markus Kalisch, Martin Mächler, Diego Colombo, Marloes H. Maathuis, and Peter Bühlmann. Causal inference using graphical models with the R package pcalg. *Journal of Statistical Software*, 47(11):1–26, 2012. doi: 10.18637/jss.v047.i11.

Minoru Kanehisa, Miho Furumichi, Mao Tanabe, Yoko Sato, and Kanae Morishima. Kegg: new perspectives on genomes, pathways, diseases and drugs. *Nucleic Acids Research*, 45:D353, 1 2017. ISSN 13624962. doi: 10.1093/NAR/GKW1092. URL /pmc/articles/PMC5210567//pmc/articles/PMC5210567/?report=abstracthttps://www.ncbi.nlm.nih.gov/pmc/articles/PMC5210567/.

Jens Keilwagen, Ivo Grosse, and Jan Grau. Area under precision-recall curves for weighted and unweighted data. *PLOS ONE*, 9(3), 2014.

Seongho Kim. ppcor: An r package for a fast calculation to semi-partial correlation coefficients. *Communications for statistical applications and methods*, 22:665–674, 11 2015. ISSN 2287-7843. doi: 10.5351/CSAM.2015.22.6.665. URL https://pubmed.ncbi.nlm.nih.gov/26688802https://www.ncbi.nlm.nih.gov/pmc/articles/PMC4681537/.

Wilhelm Kirch, editor. *Pearson's Correlation Coefficient*, pages 1090–1091. Springer Netherlands, 2008. ISBN 978-1-4020-5614-7. doi: 10.1007/978-1-4020-5614-7_2569. URL https://doi.org/10.1007/978-1-4020-5614-7_2569.

Jack Kuipers and Giusi Moffa. Partition mcmc for inference on acyclic digraphs. *Journal of the American Statistical Association*, 112(517):282–299, 2017. doi: 10.1080/01621459.2015.1133426. URL https://doi.org/10.1080/01621459.2015.1133426.

Steffen L. Lauritzen. *Graphical models*. Number 17 in Oxford Statistical Science Series. Clarendon Press, 1996. ISBN 0198522193.

Robert D. Leclerc. Survival of the sparsest: robust gene networks are parsimonious. *Molecular Systems Biology*, 4:213, 2008. ISSN 17444292. doi: 10.1038/MSB.2008.52. URL /pmc/articles/PMC2538912//pmc/articles/PMC2538912/?report=abstracthttps://www.ncbi.nlm.nih.gov/pmc/articles/PMC2538912/.

Yann LeCun, Y. Bengio, and Geoffrey Hinton. Deep learning. *Nature*, 521:436–44, 05 2015. doi: 10.1038/nature14539.

Jörg Linde, Sylvie Schulze, Sebastian G. Henkel, and Reinhard Guthke. Data- and knowledge-based modeling of gene regulatory networks: an update. *EXCLI Journal*, 14:346, 3 2015. ISSN 16112156. doi: 10.17179/EXCLI2015-168. URL /pmc/articles/PMC4817425//pmc/articles/PMC4817425/?report=abstracthttps://www.ncbi.nlm.nih.gov/pmc/articles/PMC4817425/.

Rohan Lowe, Neil Shirley, Mark Bleackley, Stephen Dolan, and Thomas Shafee. Transcriptomics technologies. *PLoS Computational Biology*, 13, 5 2017. ISSN 15537358. doi: 10.1371/JOURNAL.PCBI.1005457. URL /pmc/articles/PMC5436640//pmc/articles/PMC5436640/?report=abstracthttps://www.ncbi.nlm.nih.gov/pmc/articles/PMC5436640/.

David Madigan, Jeremy York, and Denis Allard. Bayesian graphical models for discrete data bayesian graphical models f or discrete data 2battelle pacific northwest laboratories. *International Statistical Review*, 63:215–232, 1995.

Daniel Marbach, Thomas Schaffter, Dario Floreano, Robert J Prill, and Gustavo Stolovitzky. The dream4 in-silico network challenge training data, gold standards, and supplementary information, 2009. URL http://www.gnu.org/.

Daniel Marbach, Robert J. Prill, Thomas Schaffter, Claudio Mattiussi, Dario Floreano, and Gustavo Stolovitzky. Revealing strengths and weaknesses of methods for gene network inference. *Proceedings of the National Academy of Sciences*, 107(14):6286–6291, 2010. doi: 10.1073/pnas.0913357107. URL https://www.pnas.org/doi/abs/10.1073/pnas.0913357107.

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. *BMC Bioinformatics*, 7:S7, 3 2006. ISSN 14712105. doi: 10.1186/1471-2105-7-S1-S7. URL /pmc/articles/PMC1810318//pmc/articles/PMC1810318/?report=abstracthttps://www.ncbi.nlm.nih.gov/pmc/articles/PMC1810318/.

Dirk Merkel. Docker: lightweight linux containers for consistent development and deployment. *Linux journal*, page 2, 2014.

Patrick E. Meyer, Kevin Kontos, Frederic Lafitte, and Gianluca Bontempi. Information-theoretic inference of large transcriptional regulatory networks. *EURASIP J. Bioinformatics Syst. Biol.*, 2007:8, jan 2007. ISSN 1687-4145. doi: 10.1155/2007/79879. URL https://doi.org/10.1155/2007/79879.

Patrick E Meyer, Frédéric Lafitte, and Gianluca Bontempi. minet: A r/bioconductor package for inferring large transcriptional networks using mutual information. *BMC bioinformatics*, 9:461, 10 2008. ISSN 1471-2105. doi: 10.1186/1471-2105-9-461. URL https://pubmed.ncbi.nlm.nih.gov/18959772https://www.ncbi.nlm.nih.gov/pmc/articles/PMC2630331/.

George Michailidis. Statistical challenges in biological networks. *Journal of Computational and Graphical Statistics*, 21:840–855, 2012. doi: 10.1080/10618600.2012.738614.

Thomas Moerman, Sara Aibar Santos, Carmen Bravo González-Blas, Jaak Simm, Yves Moreau, Jan Aerts, and Stein Aerts. Grnboost2 and arboreto: efficient and scalable inference of gene regulatory networks. *Bioinformatics (Oxford, England)*, 35:2159–2161, 6 2019. ISSN 1367-4811. doi: 10.1093/BIOINFORMATICS/BTY916. URL https://pubmed.ncbi.nlm.nih.gov/30445495/.

Donna L. Mohr, William J. Wilson, and Rudolf J. Freund. Nonparametric methods. *Statistical Methods*, pages 651–683, 2022. doi: 10.1016/B978-0-12-823043-5.00014-X.

Márcio A. Mourão, Jeyaraman Srividhya, Patrick E. McSharry, Edmund J. Crampin, and Santiago Schnell. A graphical user interface for a method to infer kinetics and network architecture (mikana). *PLoS ONE*, 6, 2011.

Roman F Nalewajski. *Elements of Information Theory*, pages 371–395. Springer Berlin Heidelberg, 2012. ISBN 978-3-642-20180-6. doi: 10.1007/978-3-642-20180-6_8. URL https://doi.org/10.1007/978-3-642-20180-6_8.

José Fernández Navarro. Single-cell vs. bulk sequencing: which one to use when? - biolizard. *BioLizard*, 2021. URL https://lizard.bio/blog/single-cell-vs-bulk/.

Alexandru Niculescu-Mizil and Rich Caruana. Inductive transfer for bayesian network structure learning. *Journal of Machine Learning Research - Proceedings Track*, 2:339–346, 01 2007.

Nooshin Omranian, Jeanne M.O. Eloundou-Mbebi, Bernd Mueller-Roeber, and Zoran Nikoloski. Gene regulatory network inference using fused lasso on multiple data sets. *Scientific Reports 2016 6:1*, 6:1–14, 2 2016. ISSN 2045-2322. doi: 10.1038/srep20533. URL https://www.nature.com/articles/srep20533.

Rainer Opgen-Rhein and Korbinian Strimmer. From correlation to causation networks: a simple approximate learning algorithm and its application to high-dimensional plant gene expression data. *BMC Systems Biology*, 1:37 – 37, 2007.

Judea Pearl. An introduction to causal inference. *The International Journal of Biostatistics*, 6, 1 2010. ISSN 15574679. doi: 10.2202/1557-4679.1203. URL /pmc/articles/PMC2836213//pmc/articles/PMC2836213/?report=abstracthttps://www.ncbi.nlm.nih.gov/pmc/articles/PMC2836213/.

Christopher A. Penfold, Vicky Buchanan-Wollaston, Katherine J. Denby, and David L. Wild. Nonparametric bayesian inference for perturbed and orthologous gene regulatory networks. *Bioinformatics*, 28:i233, 6 2012. ISSN 13674803. doi: 10.1093/BIOINFORMATICS/BTS222. URL /pmc/articles/PMC3371854//pmc/articles/PMC3371854/?report=abstracthttps://www.ncbi.nlm.nih.gov/pmc/articles/PMC3371854/.

R. Penrose. A generalized inverse for matrices. *Mathematical Proceedings of the Cambridge Philosophical Society*, 51(3):406–413, 1955. doi: 10.1017/S0305004100030401.

Aditya Pratapa, Amogh P. Jalihal, Jeffrey N. Law, Aditya Bharadwaj, and T. M. Murali. Benchmarking algorithms for gene regulatory network inference from single-cell transcriptomic data. *Nature Methods 2020 17:2*, 17:147–154, 1 2020. ISSN 1548-7105. doi: 10.1038/s41592-019-0690-6. URL https://www.nature.com/articles/s41592-019-0690-6.

William Revelle. *psych: Procedures for Psychological, Psychometric, and Personality Research*. Northwestern University, Evanston, Illinois, 2022. URL https://CRAN.R-project.org/package=psych. R package version 2.2.5.

Leon E. Rosenberg and Diane Drobnis Rosenberg. Birth defects. *Human Genes and Genomes*, pages 241–258, 2012. doi: 10.1016/B978-0-12-385212-0.00015-9.

Thomas Schaffter, Daniel Marbach, and Dario Floreano. Genenetweaver: in silico benchmark generation and performance profiling of network inference methods. *Bioinformatics (Oxford, England)*, 27:2263–2270, 8 2011. ISSN 1367-4811. doi: 10.1093/BIOINFORMATICS/BTR373. URL https://pubmed.ncbi.nlm.nih.gov/21697125/.

Harsh Shrivastava, Xiuwei Zhang, Le Song, and Srinivas Aluru. Grnular: A deep learning framework for recovering single-cell gene regulatory networks. *Journal of Computational Biology*, 29: 27–44, 1 2022. ISSN 10665277. doi: 10.1089/CMB.2021.0437/ASSET/IMAGES/LARGE/CMB.2021.0437_FIGURE9.JPEG. URL https://www.liebertpub.com/doi/10.1089/cmb.2021.0437.

Alexei Stepanov. On the kendall correlation coefficient, 2015. URL https://arxiv.org/abs/1507.01427.

Polina Suter, Jack Kuipers, Giusi Moffa, and Niko Beerenwinkel. Bayesian structure learning and sampling of bayesian networks with the r package bidag, 2021. URL https://arxiv.org/abs/2105.00488.

Ryan J. Tibshirani. The lasso problem and uniqueness. *Electronic Journal of Statistics*, 7:1456 – 1490, 2013. doi: 10.1214/13-EJS815. URL https://doi.org/10.1214/13-EJS815.

Richard J. Trudeau. *Introduction to Graph Theory (Dover Books on Advanced Mathematics)*. Dover Publications Inc., 1994. ISBN 0486678709. URL http://www.amazon.co.uk/Introduction-Graph-Theory-Advanced-Mathematics/dp/0486678709.

Akif Uzman. Genes and signals: Ptashne, m., gann, a. *Biochemistry and Molecular Biology Education*, 30:340–341, 9 2002. ISSN 1539-3429. doi: 10.1002/BMB.2002.494030059994. URL https://onlinelibrary.wiley.com/doi/full/10.1002/bmb.2002.494030059994https://onlinelibrary.wiley.com/doi/abs/10.1002/bmb.2002.494030059994https://iubmb.onlinelibrary.wiley.com/doi/10.1002/bmb.2002.494030059994.

Sipko van Dam, Urmo Võsa, Adriaan van der Graaf, Lude Franke, and João Pedro de Magalhães. Gene co-expression analysis for functional classification and gene–disease predictions. *Briefings in Bioinformatics*, 19:575–592, 7 2018. ISSN 1477-4054. doi: 10.1093/bib/bbw139. URL https://doi.org/10.1093/bib/bbw139.

Tim Van den Bulcke, Koenraad Van Leemput, Bart Naudts, Piet van Remortel, Hongwu Ma, Alain Verschoren, Bart De Moor, and Kathleen Marchal. Syntren : a generator of synthetic gene expression data for design and analysis of structure learning algorithms. *BMC BIOINFORMATICS*, 7:12, 2006. ISSN 1471-2105. URL http://dx.doi.org/10.1186/1471-2105-7-43.

Michael Vollenweider. benchmarking_GRN_inference, 6 2022.

Lourens Waldorp and Maarten Marsman. Relations between networks, regression, partial correlation, and latent variable model, 2020. URL https://arxiv.org/abs/2007.10656.

John N. Weinstein. The cancer genome atlas pan-cancer analysis project. *Nature genetics*, 45: 1113–1120, 10 2013. ISSN 1546-1718. doi: 10.1038/NG.2764. URL https://pubmed.ncbi.nlm.nih.gov/24071849/.

Adriano V. Werhli and Dirk Husmeier. Reconstructing gene regulatory networks with bayesian networks by combining expression data with multiple sources of prior knowledge. *Statistical applications in genetics and molecular biology*, 6, 5 2007. ISSN 1544-6115. doi: 10.2202/1544-6115.1282. URL https://pubmed.ncbi.nlm.nih.gov/17542777/.

Alex White and Matthieu Vignes. Causal queries from observational data in biological systems via bayesian networks: An empirical study in small networks. *Methods in Molecular Biology*, 1883: 111–142, 5 2018. ISSN 10643745. doi: 10.48550/arxiv.1805.01608. URL https://arxiv.org/abs/1805.01608v1.

Steven Woodhouse, Nir Piterman, Christoph M. Wintersteiger, Berthold Göttgens, and Jasmin Fisher. Scns: a graphical tool for reconstructing executable regulatory networks from single-cell genomic data. *BMC systems biology*, 12, 5 2018. ISSN 1752-0509. doi: 10.1186/S12918-018-0581-Y. URL https://pubmed.ncbi.nlm.nih.gov/29801503/.

Luke Zappia, Belinda Phipson, and Alicia Oshlack. Splatter: Simulation of single-cell rna sequencing data. *Genome Biology*, 18:1–15, 9 2017. ISSN 1474760X. doi: 10.1186/S13059-017-1305-0/FIGURES/6. URL https://genomebiology.biomedcentral.com/articles/10.1186/s13059-017-1305-0.

Hui Zou and T Hastie. Regularization and variable selection via the elastic nets. *J. Royal Stat. Soc. B*, 67:301–320, 6 2015.

Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zurich

Eigenständigkeitserklärung

Die unterzeichnete Eigenständigkeitserklärung ist Bestandteil jeder während des Studiums verfassten Semester-, Bachelor- und Master-Arbeit oder anderen Abschlussarbeit (auch der jeweils elektronischen Version).

Die Dozentinnen und Dozenten können auch für andere bei ihnen verfasste schriftliche Arbeiten eine Eigenständigkeitserklärung verlangen.

Ich bestätige, die vorliegende Arbeit selbständig und in eigenen Worten verfasst zu haben. Davon ausgenommen sind sprachliche und inhaltliche Korrekturvorschläge durch die Betreuer und Betreuerinnen der Arbeit.

Titel der Arbeit (in Druckschrift):

BENCHMARKING FRAMEWORK FOR GENE REGULATORY NETWORK INFERENCE METHODS

Verfasst von (in Druckschrift):
Bei Gruppenarbeiten sind die Namen aller Verfasserinnen und Verfasser erforderlich.

Name(n):	**Vorname(n):**
VOLLENWEIDER	MICHAEL

Ich bestätige mit meiner Unterschrift:
- Ich habe keine im Merkblatt „Zitier-Knigge“ beschriebene Form des Plagiats begangen.
- Ich habe alle Methoden, Daten und Arbeitsabläufe wahrheitsgetreu dokumentiert.
- Ich habe keine Daten manipuliert.
- Ich habe alle Personen erwähnt, welche die Arbeit wesentlich unterstützt haben.

Ich nehme zur Kenntnis, dass die Arbeit mit elektronischen Hilfsmitteln auf Plagiate überprüft werden kann.

Ort, Datum	**Unterschrift(en)**
28.07.2022	

Bei Gruppenarbeiten sind die Namen aller Verfasserinnen und Verfasser erforderlich. Durch die Unterschriften bürgen sie gemeinsam für den gesamten Inhalt dieser schriftlichen Arbeit.