# Bayesian Inference of Biological Networks Using Probabilistic Graphical Models

A Dissertation

Presented in Partial Fulfillment of the Requirements for the

Degree of Doctor of Philosophy

with a

Major in Bioinformatics and Computational Biology

in the

College of Graduate Studies

University of Idaho

by

Evan A Martin

Major Professor: Audrey Qiuyan Fu, Ph.D.

Committee Members: Paul Hohenlohe, Ph.D.; Benjamin Ridenhour, Ph.D.; Terry Soule, Ph.D.

Department Administrator: Dave Tank, Ph.D.

December 2020

# Authorization to Submit Dissertation

This dissertation of Evan A Martin, submitted for the degree of Doctor of Philosophy with a Major in Bioinformatics and Computational Biology and titled "Bayesian Inference of Biological Networks Using Probabilistic Graphical Models," has been reviewed in final form. Permission, as indicated by the signatures and dates below, is now granted to submit final copies to the College of Graduate Studies for approval.

| | | |
|---|---|---|
| Major Professor: | Audrey Qiuyan Fu, Ph.D. | Date |
| Committee Members: | Paul Hohenlohe, Ph.D. | Date |
| | Benjamin Ridenhour, Ph.D. | Date |
| | Terry Soule, Ph.D. | Date |
| Department Administrator: | Dave Tank, Ph.D. | Date |

# Abstract

Gene regulatory networks are a visual representation of genes and their interactions. In this visual representation, nodes correspond to genes, while edges correspond to interactions among genes. Learning the structure of a gene regulatory network from data can provide valuable insights on how genes regulate one another. Understanding the complex regulatory relationships among genes is key to understanding many biological processes. Methods that infer the structure of a gene regulatory network are powerful tools for understanding these processes. The inference from these methods has a wide range of applications such as understanding complex traits or diagnosing and treating disease. Probabilistic graphical models or networks describe the statistical dependence between variables in a system and graphical model based methods are commonly used to infer the structure of a gene regulatory network. We present a Bayesian graphical model based approach to infer the structure of a network and develop a Metropolis-Hastings algorithm for the inference. Our method quantifies uncertainty in the inference, uses edge-level prior probabilities, and incorporates prior or external knowledge of the network structure, and accounts for multiple data types (for example, discrete, continuous, and mixed). We illustrate the accuracy and efficiency of our method through simulation studies and compare our method to existing Bayesian methods. We demonstrate the practical application of our method by applying it to two different biological networks. First, we infer a gene regulatory network from individual level genotype and gene expression data in humans. Second, we infer the combinatorial binding profiles of transcription factors during Drosophila mesoderm development. We extend our method to infer gene regulatory networks by including biological assumptions that regulate the relationships between data types. Specifically, we use the principle of Mendelian randomization to infer causal relationships among genes by incorporating individual level genotype data in the network. We carry out a wide range of simulation studies on gene regulatory networks and demonstrate that our method can accurately infer regulatory relationships among genes.

# Acknowledgements

First, I want to thank my advisor Dr. Audrey Fu for all of her help. Without her this work would not have been possible. I have learned many things from her as we have worked together. She has been instrumental as I transition from just being a student who is trying to get through school to becoming a scientist.

I also want to thank Dr. Erkan Buzbas for his helpful discussions and advice. He has inspired me with the desire to tackle difficult questions that I find interesting.

I want to thank Dr. Terry Soule for his lectures on genetic algorithms. They sparked my interest and helped me form the idea that became the method presented in this dissertation.

# Dedication

To my family:

my parents Steve and Valerie

my wife Emily

my sons Lincoln, Collin, and Stuart

# Table of Contents

Authorization to Submit Dissertation . . . ii
Abstract . . . iii
Acknowledgements . . . iv
Dedication . . . v
Table of Contents . . . vi
List of Tables . . . vii
List of Figures . . . ix
Chapter 1: Introduction . . . 1
1.1 Gene regulatory networks . . . 1
1.2 Gene regulatory network inference . . . 1
Chapter 2: Bayesian inference of directed acyclic graphs with edge-level prior probabilities . . . 5
2.1 Introduction . . . 5
2.2 Methods . . . 7
2.3 Simulation Studies . . . 20
2.4 Applications to biological data . . . 27
2.5 Discussion . . . 39
Chapter 3: A Bayesian graphical model approach to gene regulatory networks with individual level data . . . 57
3.1 Introduction . . . 57
3.2 Model . . . 59
3.3 Simulation study . . . 63
3.4 Discussion . . . 88
Chapter 4: Conclusion and discussion . . . 90
4.1 Conclusion . . . 90
4.2 Discussion . . . 90
4.3 Future work . . . 91
References . . . 92

# List of Tables

2.1 Summary of Bayesian methods for network inference. . . . . 21
2.2 $MSE_1$ of baycn on graphs in simulation studies. . . . . 25
2.3 Edgewise MSE for topology M1. . . . . 41
2.4 Edgewise MSE for topology M2. . . . . 41
2.5 Edgewise MSE for topology GN4. . . . . 42
2.6 Edgewise MSE for topology GN11. . . . . 43
2.7 $MSE_2$ for GN4, GN8, and GN11. . . . . 44
2.8 Precision, power, and $MSE_2$ for GN4. . . . . 45
2.9 Edge state posterior probabilities for GN4: $N = 100$. . . . . 47
2.10 Edge state posterior probabilities for GN4: $N = 200$. . . . . 48
2.11 Edge state posterior probabilities for GN4: $N = 600$. . . . . 49
2.12 Runtime for MCMC methods. . . . . 50
2.13 Posterior probabilities for data sets from GEUVADIS and GTEx. . . . . 51
2.14 Posterior probabilities for eQTL-gene set Q8 with PCs. . . . . 52
2.15 Posterior probabilities for eQTL-gene set Q21 with PCs. . . . . 53
2.16 Posterior probabilities for eQTL-gene set Q23 with PCs. . . . . 53
2.17 Posterior probabilities for eQTL-gene set Q37 with PCs. . . . . 54
2.18 Posterior probabilities for eQTL-gene set Q50 with PCs. . . . . 54
2.19 Posterior probabilities for the Drosophila network. . . . . 55
2.20 Posterior probabilities from baycn and partition MCMC for the Drosophila network. . . . . 56

3.1 The $MSE_1$ for networks in the PMR simulation studies. . . . . 67
3.2 The eMSE for the networks in the confounding variable simulation studies. . . . . 70
3.3 Percent of variation for top two PCs. . . . . 71
3.4 eMSE for M1 and M2 with true edge input. . . . . 72
3.5 eMSE for the layer topology with true edge input. . . . . 73
3.6 eMSE for topology M3 with 2, 5, and 10 confounding variables. . . . . 73
3.7 Number of MCMC iterations and step size. . . . . 74
3.8 The average posterior probability difference for the $W$ edges. . . . . 76
3.9 The edgewise power for BGRN and LASSO. . . . . 85
3.10 The correlation for the feed forward loop and chain topologies. . . . . 85

3.11 The edgewise power for the feed forward loop topologies. . . . . 87
3.12 The eMSE for the feed forward loop topologies. . . . . 88

# List of Figures

2.1 Remove a single directed cycle. . . . . 14
2.2 Remove multiple directed cycles. . . . . 17
2.3 Nested cycles. . . . . 19
2.4 Steps of the cycle remover algorithm. . . . . 20
2.5 Topologies used in Chapter 2 simulation studies. . . . . 22
2.6 Graphs showing the input with false edges. . . . . 26
2.7 Boxplots for $MSE_2$ for all MCMC methods. . . . . 26
2.8 Boxplots of precision, power, and $MSE_1$ for GN4. . . . . 28
2.9 Inferred graph and correlation heat map for eQTL-gene set Q8. . . . . 30
2.10 Inferred graphs for GEUVADIS and GTEx. . . . . 32
2.11 Heat maps for GEUVADIS and GTEx. . . . . 33
2.12 Heat maps with PCs for GEUVADIS. . . . . 34
2.13 Graph inferred by baycn for the Drosophila data set. . . . . 36
2.14 Heat map for Drosophila data. . . . . 38
2.15 Graph used as input to baycn for Drosophila data. . . . . 39

3.1 Topologies used in the PMR simulation studies. . . . . 63
3.2 Topologies used in the confounding variable simulation studies. . . . . 64
3.3 Precision and power for PMR topologies. . . . . 66
3.4 The average posterior probability difference for the $W$ edges. . . . . 77
3.5 Edgewise power for low correlation and few parents. . . . . 79
3.6 Edgewise power for low correlation and many parents. . . . . 80
3.7 Edgewise power for high correlation and few parents. . . . . 81
3.8 Edgewise power for high correlation and many parents. . . . . 82
3.9 Networks used in comparison of BGRN and LASSO. . . . . 84

# Chapter 1: Introduction

## 1.1 Gene regulatory networks

Gene regulatory networks perform two functions. First, gene regulatory networks provide an intuitive way to visualize genes and their interactions. A gene regulatory network is made up of nodes and edges where the nodes represent genes (or gene products) and edges correspond the molecular interaction between them. Therefore, gene regulatory networks provide a map of genes and the relationship between them. The edges (or relationships) in a gene regulatory network can be either directed or undirected. An undirected edge merely describes an association between two genes whereas a directed edge implies a regulatory or causal relationship between genes. An example of a regulatory relationship is that of a gene that produces molecules such as transcription factors or small interfering RNA (siRNA) [16] that then regulate the expression of other genes. How gene expression is regulated is central to many biological systems and processes. Therefore, an important challenge in biology is to understand the structure of complex interactions among genes.

At the same time, gene regulatory networks provide a basis for analyzing data from high-throughput technologies (e.g., microarrays and next-generation sequencing). Recent advancements in these technologies have lead to a drastic increase in the data used to infer gene regulatory networks. Networks inferred from these data provide information about the pathway a gene belongs to in a given tissue. In other words, these networks describe how a particular gene interacts with other genes either directly or indirectly. These interactions describe the biological function of a gene in terms of its relationship to other genes instead of merely concentrating on its individual behavior [6, 22]. Accurately inferring gene regulatory networks has attracted a great deal of scientific interest and has a wide range of applications [21]. A common application is learning the structure of a network which can guide the design of controlled experiments [39, 81], be used to better understand complex traits [87], aid in drug design by identifying target genes or pathways [70, 38, 54], or be used to better understand and diagnose disease [54, 10, 49].

## 1.2 Gene regulatory network inference

Many computational methods have been developed to learn the structure of a gene regulatory network [57, 85, 7, 19]. Many of these methods can be broken into five main groups: information theory, Boolean networks, differential equation models, Bayesian networks, and neural networks. We briefly cover information theory, Boolean networks, differential equation models, and neural networks before covering Bayesian networks in more detail. Information theory methods infer undirected networks and scale well

to large networks and use correlation [34], mutual information [9, 59], or conditional mutual information [93] for the inference. While these methods can handle large networks they are limited because they infer an undirected network. Boolean network methods [73, 65] first take gene expression and reduce it to two expression levels (on or off) by applying a cutoff to the data. These methods can infer directed networks but they lose information by reducing gene expression to either being on or off. Differential equation models [11, 79, 61] can also infer directed networks and use continuous instead of discrete variables. In addition, they can model gene expression over time leading to a more detailed inference. Neural network methods have two main classes: Artificial Neural Networks (ANNs) and Recurrent Neural Networks (RNNs). RNN methods [51, 42] have several advantages over ANN methods [56, 80] as they allow for non-linear relationships among genes and can be applied to time-series data. While Boolean networks, differential equation models, and neural networks provide more detailed inferences than information theory methods, they are computationally much more expensive and do not scale well to larger networks.

Bayesian networks are a widely used class of probabilistic graphical models [47, 64]. Probabilistic graphical models or networks represent the joint probability distribution of the variables in a system. In other words, a graphical model describes how the variables vary in relation to one another. Similar to gene regulatory networks, nodes in a graphical model correspond to variables, while edges in a graphical model represent statistical dependence between variables. The joint distribution is encoded in the structure (which edges are present) of the network. In most cases the structure of a system is unknown beforehand, and methods used for network inference generally focus on learning this structure [41]. Bayesian networks are represented by a directed acyclic graph (DAG). A DAG is a graph with only directed edges and no directed cycles. Under certain circumstances directed edges can be interpreted as causal relationships [18]. Because graphical models can clearly represent complex systems and infer causal relationships between variables, they are powerful tools for modeling and understanding complex systems and are commonly used to infer gene regulatory networks.

Many Bayesian network methods have been developed to learn the structure of a network. These methods can be broken into two classes [40]: constraint-based methods and score-based methods. The constraint-based methods [72, 14, 4] start with a fully connected graph and conduct a series of marginal and conditional independence tests between pairs of nodes. The edge is removed if the test does not reject the null hypothesis which is the two nodes are independent. If an edge is removed additional conditional independence tests are not performed on the pair of nodes. Marginal and conditional tests are both used because two nodes may be marginally dependent but this dependence could be explained away when conditioning on another node or nodes. A major benefit of these algorithms is their efficiency and scalability to large graphs. However, these algorithms do not provide a measure of uncertainty in

the inference and can be sensitive to how nodes (i.e., variables) are ordered [14, 4].

Score-based learning methods move through graph space according to a score computed for the graph given the data. Non-Bayesian methods may take a hill climbing [58, 68] approach to maximize the score, which is typically based on the likelihood or penalized likelihood. Bayesian methods either sample directly from a closed-form posterior distribution [71, 48] or use a Markov Chain Monte Carlo (MCMC) algorithm [27, 24, 30, 66, 33, 63, 29, 44] for the inference. The MCMC methods move through graph space by proposing new graph structures, scoring them, and determining whether to stay at the current structure or move to the proposed structure. The Bayesian score-based methods have an advantage of quantifying uncertainty in the inference. While these methods provide a more detailed inference, by estimating edge probabilities, they are computationally more expensive and do not scale well to larger networks. This is due to the size of graph space [13]. For example, a graph with only 19 nodes has more network structures than the estimated number of atoms in the observable universe. A network of only 19 genes is very small considering the number of genes in the human genome is in the tens of thousands [15]. Despite the many Bayesian network methods currently available, much work still needs to be done to efficiently and accurately infer the structure of a gene regulatory network from data.

We present a Bayesian graphical model based method that addresses some of the issues current methods face. Our method provides a measure of uncertainty in the inference. This is a desirable quality as not all relationships provide the same amount of signal and are, therefore, not all equally likely. In addition, our method can take as input a network obtained either from prior knowledge or from a more efficient, albeit less informative, network inference method. This greatly reduces the search space and compute time of our method. Our method also specifies a prior probability on individual edges in the network. An edge-level prior provides a benchmark to compare with the posterior probability and shows how much the data support an individual edge. Our method also allows for graphs with mixed data types (e.g., discrete and continuous). Specifically, we include these data types to analyze individual level genotype, gene expression, and phenotype data. We also include biological assumptions that restrict the relationships between these data types. For example, we use the principle of Mendelian randomization [4] to infer causal relationships between genes from individual level genotype and gene expression data. In addition, our method can be used to identify trait related genes from individual level genotype, gene expression, and phenotype data.

We demonstrate that our method is widely applicable through simulation studies and real data analyses. We carry out various simulation studies for general networks and demonstrate the accuracy and efficiency of our method. In addition, we perform many simulation studies that are specific to gene regulatory networks. These studies demonstrate that our method can accurately infer regulatory relationships

among genes by including additional variable types and assumptions in the network. We compare our method with existing Bayesian and non-Bayesian methods and find that our method is comparable to or better than existing methods in terms of inference accuracy. We apply our method to real data sets and show how our method can identify gene regulatory relationships from individual level genotype and gene expression data. We also infer the combinatorial binding profiles of transcription factors during Drosophila mesoderm development.

# Chapter 2: Bayesian inference of directed acyclic graphs with edge-level prior probabilities

**Abstract**

Graphical models or networks describe the statistical dependence among multiple variables and are widely used in biology (e.g., gene regulatory networks). Existing Bayesian methods typically assign prior probabilities for the entire directed acyclic graph, but estimate the posterior probabilities for individual edges. In reality, however, it is much easier to formulate prior probabilities for an edge than for the entire graph. Edge-level priors further provide a benchmark for interpreting the posterior edge probabilities. We represent a graph as a vector of edge states; each edge can be in one of three states: the two directions when the edge is present, and absence of this edge in the graph. We can then specify the prior probability of each state for an edge. We develop baycn (BAYesian Causal Network), a Metropolis-Hastings Markov chain Monte Carlo algorithm, for inference under this representation. We apply baycn to infer the probabilities of the regulatory relationships among multiple target genes of the same genetic variant in humans, as well as that of the combinatorial binding of transcription factors in Drosophila mesoderm development. Our method is implemented in the R package baycn, which is available on CRAN (`https://CRAN.R-project.org/package=baycn`).

## 2.1 Introduction

Graphical models (or networks), which may include directed and undirected edges, can be used to represent the statistical dependence among multiple variables and have wide applications in biology, such as gene regulatory networks [24, 66] and protein-protein interaction networks [28, 92]. Under appropriate assumptions, directed edges in a network may represent causal (or regulatory) relationships [74, 32, 18]. Directed acyclic graphs (DAGs) [64], also known as Bayesian networks, are graphs with only directed edges. If undirected edges are viewed as a special case of directed edges, with the two directions being equally likely, then a general graph with directed and undirected edges can also be viewed as a DAG. Here, we take this probabilistic approach to graphs and develop a Bayesian method for not only learning the graph structure but also for quantifying the uncertainty in the inference in the posterior probabilities.

Many Bayesian methods have been developed which explore graph space and draw a sample graph either directly from a closed-form posterior distribution [71, 48, 88, 52, 89, 26, 36] or, more often, use a

Markov Chain Monte Carlo (MCMC) algorithm [55, 25, 27, 30, 33, 63, 29, 44, 45, 43, 67, 83] to sample the posterior distribution. Using the sample drawn from the posterior distribution, they then estimate the posterior probability of edge presence/absence [71, 48, 63], or edge direction and absence [55, 25, 27, 30, 33, 29, 44, 45, 43, 67, 83] for the edges in the network. However, interpreting the posterior probabilities from existing Bayesian methods can be difficult due to two reasons. i) The posterior probabilities of the two possible directions of the same edge may be independent of each other [88, 52, 89, 36]. For example, both probabilities may be close to 1, but it is unclear whether this implies that the two directions are equally likely. ii) It is often unclear what prior should be used when comparing with the posterior. The prior used by existing Bayesian methods is typically for the entire graph [55, 29, 25, 44, 45, 43, 67, 83] and does not translate to the probability of a specific state of an individual edge. For example, suppose that an edge between $X$ and $Y$ is inferred to have a probability of 0.3 in the direction $X \rightarrow Y$ and 0.1 in the opposite direction ($X \leftarrow Y$), what should we conclude? We may reason that the inference does not support the presence of the edge, as the probability of edge presence is only $0.3 + 0.1 = 0.4$. However, this reasoning implicitly assumes that the prior probability of edge presence is 0.5. In another example, $X \rightarrow Y$ is inferred to have a probability of 0.4, and $X \leftarrow Y$ 0.15. Two interpretations are possible here. i) The inference supports $X \rightarrow Y$, as the probability of edge presence is 0.55 and the probability of $X \rightarrow Y$ is much larger than $X \leftarrow Y$. Here, the implicit prior is also 0.5 for *edge presence*. ii) The edge is absent, since there is not enough support for either direction (neither probability is above 0.5). Here, the implicit prior is 0.5 for at least one direction. Both interpretations can be reasonable, but they are based on different priors. The examples above show that an edge-level prior is needed for interpretation of the posterior probabilities, and it is more intuitive and easier to formulate prior probabilities for an edge than for the entire graph.

We address the issue with the prior when inferring networks from static data in this article. To do so, we develop a novel representation of a Bayesian network by specifying the graph in terms of the edges and their states. We specify a prior distribution for the edge states, which provides a benchmark to compare with the posterior probability. Whether the posterior probability for an edge state is large enough depends on the corresponding prior. For example, if the prior probability is 0.05 for an edge state (in other words, one believes that the edge is unlikely to be present a priori), then a posterior probability of 0.3 can already indicate support from the data for this state, even though 0.3 is not typically considered a large posterior. Our method can take as input a graph from another more efficient graph inference method (e.g., from a constraint-based method [40]), which greatly reduces the size of the search space. Our algorithm can also work with multiple data types: continuous, discrete, and mixed. We demonstrate the performance of our method through simulation and in two different biological problems. The first examines the regulatory

relationships among the target genes of genetic variants, while accounting for confounding variables in the inference. This data set has discrete and continuous data. The second problem investigates combinatorial transcription factor binding during Drosophila mesoderm development, using binary binding profiles which have high correlation among them.

## 2.2 Methods

### 2.2.1 The Bayesian graphical model

A graph $\mathcal{G} = (\mathbf{V},\ \mathbf{E})$ is a set of vertices (nodes) $\mathbf{V} = \{1, 2, ..., b\}$ and edges $\mathbf{E} \subseteq \mathbf{V} \times \mathbf{V}$, where $\mathbf{V} \times \mathbf{V}$ is the set of all ordered pairs of nodes, such as $(j, k)$, which denotes an edge pointing from node $j$ to node $k$ where $j, k \in \mathbf{V}$. The structure (or topology) of the graph is typically defined by the (deterministic) adjacency matrix $\mathbf{A}$ of dimension $b \times b$, where $A_{jk} = 1$ and $A_{kj} = 0$ signify an edge from node $j$ to node $k$, and $A_{jk} = 0$ and $A_{kj} = 1$ signify an edge from node $k$ to node $j$. If $A_{jk} = A_{kj} = 0$ there is no edge between nodes $j$ and $k$.

We introduce an alternate representation of the graph to directly describe the states of individual edges with the vector $\mathbf{S} = (S_1, S_2, \ldots, S_m)$, where $m$ is the number of edges. Each edge is in one of three possible states:

$$S_i = \begin{cases} 0, & \text{if } A_{jk} = 1 \text{ and } A_{kj} = 0 \text{ where } j < k; \\ 1, & \text{if } A_{kj} = 1 \text{ and } A_{jk} = 0 \text{ where } j < k; \\ 2, & \text{if } A_{jk} = A_{kj} = 0 \text{ (i.e., the edge is absent),} \end{cases}$$

such that

$$\sum_{k=0}^{2} \Pr(S_i = k) \equiv p_k = 1. \tag{2.1}$$

With a slight abuse of notation, we draw connections to the adjacency matrix and define

$$\Pr(S_{jk} = 0) \equiv \Pr(A_{jk} = 1) \text{ and } \Pr(S_{jk} = 1) \equiv \Pr(A_{kj} = 1), \tag{2.2}$$

where $S_{jk}$ with the double subscript denotes the state of the edge between nodes $j$ and $k$. Under this notation,

$$\Pr(S_{jk} = 2) = 1 - \Pr(A_{jk} = 1) - \Pr(A_{kj} = 1). \tag{2.3}$$

Incidentally, if $\Pr(S_i = 0) = \Pr(S_i = 1) = 0.5$, then the $i$th edge is undirected with the two directions being equally likely.

When data are available at all the nodes (each node represents a variable), the set of nodes $\mathbf{V}$ corresponds to a vector of random variables $\mathbf{T} = (T_1, T_2, ..., T_b)^T$. We aim to infer the posterior edge state probability $\Pr(S_i \mid \mathbf{T})$ for all edges. Similar to Equation (2.1), the posterior probabilities of the three states for an edge also add up to 1. We can also represent these posterior probabilities in a probabilistic adjacency matrix, where each entry is the probability of one of the two possible directions ($\Pr(S_{jk} = 1)$ or $\Pr(S_{kj} = 1)$). Existing Bayesian methods generally produce such a posterior probabilistic adjacency matrix.

The probability of the graph can be written as a product of conditional probabilities where each node is conditioned on its parents

$$\Pr(\mathbf{T} \mid \mathbf{S}, \boldsymbol{\theta}) = \prod_{j=1}^{b} \Pr(T_j \mid pa(T_j), \boldsymbol{\theta}_j) \tag{2.4}$$

where $pa(T_j)$ is the set of nodes that are the parents of $T_j$, $\boldsymbol{\theta}_j$ is the parameter vector for the distribution of $T_j$, and $\boldsymbol{\theta} = \{\boldsymbol{\theta}_1, \ldots, \boldsymbol{\theta}_b\}$. If $pa(T_j) = \emptyset$, where $\emptyset$ is an empty set, the probability is reduced to a marginal probability $\Pr(T_j \mid \boldsymbol{\theta}_j)$.

When we assume normality for the data at each node:

$$T_j \sim \mathrm{N}(\mu_j, \sigma_j^2), \tag{2.5}$$

$$\mu_j = \beta_0 + \sum_{k \in pa(T_j)} \beta_k T_k, \tag{2.6}$$

where $\mu_j$ is the mean and $\sigma_j^2$ the variance. If the node $T_j$ does not have any parents then $\mu_j = \beta_0$. Alternatively, if the data are binary:

$$T_j \sim \text{Bernoulli}(p_j), \tag{2.7}$$

$$\log\left(\frac{p_j}{1 - p_j}\right) = \beta_0 + \sum_{k \in pa(T_j)} \beta_k T_k, \tag{2.8}$$

where $p_j$ is the "success" probability. If node $T_j$ does not have any parents then $\log\big(p_j/(1-p_j)\big) = \beta_0$.

### 2.2.2 The Metropolis-Hastings MCMC algorithm

We have developed baycn (BAYesian Causal Network), a Metropolis-Hastings algorithm, that proposes changes to edge states. The input is the binary adjacency matrix of a candidate graph and the data at

the nodes. The candidate graph may be a fully connected graph, where all pairs of nodes are connected by an edge. A more efficient approach is to run a fast graph inference algorithm to produce a candidate graph and use it as the input, even if this graph contains false edges.

**Algorithm 1:** baycn

**input :** Data matrix, candidate graph, and prior on edge states
**output:** Posterior probability of edge states for each edge considered

1 Randomly generate a starting graph $\mathbf{S}_{(1)}$ from the candidate graph;
2 **for** $i \leftarrow 2$ **to** $M$ **do**
3 Generate a proposal graph $\mathbf{S}'_{(t)}$ from the current graph $\mathbf{S}_{(t-1)}$;
4 Check for and remove directed cycles in $\mathbf{S}'_{(t)}$;
5 Calculate the acceptance probability
$$\alpha_{(t)} = \min\left\{\frac{\Pr(\mathbf{S}'_{(t)})\Pr(\mathbf{T} \mid \mathbf{S}'_{(t)}, \boldsymbol{\theta}_{(t)})\Pr(\mathbf{S}_{(t-1)} \mid \mathbf{S}'_{(t)})}{\Pr(\mathbf{S}_{(t-1)})\Pr(\mathbf{T} \mid \mathbf{S}_{(t-1)}, \boldsymbol{\theta}_{(t-1)})\Pr(\mathbf{S}'_{(t)} \mid \mathbf{S}_{(t-1)})},\ 1\right\};$$
6 Generate $u$ from Uniform$(0, 1)$;
7 **if** $u < \alpha_{(t)}$ **then**
8 $\mathbf{S}_{(t)} = \mathbf{S}'_{(t)}$;
9 **else**
10 $\mathbf{S}_{(t)} = \mathbf{S}_{(t-1)}$;
11 **end**
12 **end**

To generate the proposal graph in Line 3 of Algorithm 1, we first determine the number of edges to change states by sampling from a binomial distribution $\mathrm{B}(m, 1/m)$, where $m$ is the number of edges in the network and $1/m$ is the "success" probability. For each of the selected edges, we then sample from a Bernoulli distribution with probability $p$ to decide which edge state to change to. Since we do not allow for an edge to remain in the same state, $p$ is determined by the prior of the two remaining edge states. For example, if an edge in state 0 is selected to change states, and if the prior probability for the edge states is $(p_0, p_1, p_2) = (0.05, 0.05, 0.9)$, then the probability of switching to state 1 is $p = 0.05/(0.05+0.9)$.

To ensure that the proposal graph does not contain directed cycles, we have further designed two algorithms as part of the MCMC algorithm: the "cycle finder" algorithm (see Section 2.2.5) finds all possible cycles given the input graph; and the "cycle remover" algorithm (see Section 2.2.6) removes all directed cycles from the proposal graph (see Sections 2.2.3 and 2.2.4 for the theorem, proof, and examples).

This MCMC algorithm generates a sample of graphs represented by edge states. For each edge, the relative frequencies of the three states in the MCMC sample provide an estimate of the posterior probabilities of edge states $\Pr(S_i \mid \mathbf{T})$.

Through changes in edge states, our algorithm can sample from multiple Markov equivalent graphs and thus produce posterior probabilities that account for Markov equivalence. With sufficient data,

the posterior probabilities of edge states should be the same asymptotically as expected under Markov equivalence.

### 2.2.3 Identifying and removing directed cycles

In a directed cycle, one can follow the directed edges and return to the starting node (e.g., $T_1 \to T_2 \to T_3 \to T_1$). Directed cycles can have a higher likelihood than the true graph, a graph with directed edges but no cycles, and therefore should be removed during the MCMC iterations when generating a proposal graph.

We have developed a "cycle finder" algorithm (Section 2.2.5) to find all directed cycles (including overlapping cycles, as well as multiple disjoint ones) in a graph, and a "cycle remover" algorithm (Section 2.2.6) to move out of directed cycles such that the proposed graph is free of directed cycles. It is plausible that when proposing a new graph, our MCMC algorithm may propose a graph with one or more directed cycles, try to move out of these directed cycles only to generate a graph with different directed cycles. Therefore, our algorithm may need to repeatedly identify and remove directed cycles in one MCMC iteration. However, since we focus on relatively small graphs in this paper, this scenario is rather unlikely.

Recall that our algorithm next calculates the acceptance probability for the proposed graph relative to the current one. Although the (repeated) removal of directed cycles enters the calculation of the transition probability between the current and proposed graph, the probabilities involving the cycles are in the end canceled in the calculation of the acceptance ratio. Let $\mathbf{D}$ be a vector of indices of the edges that *differ* between the current graph $\mathbf{S}$ and proposed graph $\mathbf{S'}$ and $\mathbf{C}$ be an integer vector where the element $c_j$ indicates the number of edges that *can* change state for the edge denoted by $d_j$ (see examples in Section 2.2.4). These two vectors have the same length, denoted by $h$. The probability of moving from $\mathbf{S}$ to $\mathbf{S'}$, i.e., $\Pr(\mathbf{S} \to \mathbf{S'})$, is the product of the probabilities of changes at individual edges in $\mathbf{D}$, and each of these probabilities further consists of two probabilities: the probability that an edge in the graph is chosen to change states, which is $1/c_j$, and the probability of edge $d_j$ changing from its current state $S_{d_j}$ to the state $S'_{d_j}$, denoted by $\Pr(S_{d_j} \to S'_{d_j})$. Therefore,

$$\Pr(\mathbf{S} \to \mathbf{S'}) = \prod_{j=1}^{h} \frac{1}{c_j} \Pr(S_{d_j} \to S'_{d_j}) = \prod_{j=1}^{h} \frac{1}{c_j} \prod_{j=1}^{h} \Pr(S_{d_j} \to S'_{d_j}). \tag{2.9}$$

We prove that the transition probabilities do not depend on the path taken from the current graph to the proposed graph (the process of introducing and removing directed cycles) but only on the edges that have different states between the two graphs.

**Theorem 1** *When calculating the acceptance probability, $\alpha$, the transition probability between the current and proposed graph, $\Pr(\mathbf{S} \to \mathbf{S'})$ and $\Pr(\mathbf{S'} \to \mathbf{S})$, depends only on the edges whose states are different between the two graphs.*

PROOF Recall that $\mathbf{S}$ is a vector of edge states representing the current graph $\mathcal{G}$ and $\mathbf{S'}$ is a vector of edge states representing the proposal graph $\mathcal{G}'$. Let $\mathbf{D}$ be a vector of indices of the edges that *differ* between the current graph $\mathbf{S}$ and proposed graph $\mathbf{S'}$ and $\mathbf{C}$ be an integer vector where the element $c_j$ represents the number of edges that *can* change state for the edge represented by $d_j$. These two vectors have the same length, denoted by $h$.

We will consider two cases: without and with potential directed cycles in the graph.

*Case 1* Without potential directed cycles the probability of moving from the current graph to the proposed graph is

$$\begin{aligned}\Pr(\mathbf{S} \to \mathbf{S'}) &= \prod_{j=1}^{h} \frac{1}{c_j} \Pr(S_{d_j} \to S'_{d_j}) \\ &= \prod_{j=1}^{h} \frac{1}{c_j} \prod_{j=1}^{h} \Pr(S_{d_j} \to S'_{d_j}). \end{aligned} \tag{2.10}$$

We can use the same procedure of deriving the equation for moving back to the current graph from the proposed graph. Therefore, the probability can be broken down in the same way when moving backwards:

$$\Pr(\mathbf{S'} \to \mathbf{S}) = \prod_{j=1}^{h} \frac{1}{c_j} \prod_{j=1}^{h} \Pr(S'_{d_j} \to S_{d_j}). \tag{2.11}$$

Since there are no potential directed cycles in the network the value $c_j$ will always be $m$ which is the number of edges in the network. Therefore, $\prod_{j=1}^{h} \frac{1}{c_j} = \prod_{j=1}^{h} \frac{1}{m}$ whether going from $\mathbf{S} \to \mathbf{S'}$ or $\mathbf{S'} \to \mathbf{S}$ and will cancel when calculating the acceptance probability, leaving $\prod_{j=1}^{h} \Pr(S_{d_j} \to S'_{d_j})$ and $\prod_{j=1}^{h} \Pr(S'_{d_j} \to S_{d_j})$.

*Case 2* With potential directed cycles there can be multiple paths when moving from the current graph to the proposed graph. Let $\mathbf{C}^k$ be a vector where each $c_j^k$ is the number of edges that can change state in path $k$ when moving from $\mathbf{S}$ to $\mathbf{S'}$ and $\mathbf{C}^{k'}$ be a vector where each $c_j^{k'}$ is the number of edges that can change state in path $k$ when moving from $\mathbf{S'}$ to $\mathbf{S}$. Using equation (2.10) the transition probability of

moving from the current graph to the proposed graph when there are multiple paths becomes

$$\Pr(\mathbf{S} \to \mathbf{S'}) = \sum_{k=1}^{K} \prod_{j=1}^{h} \frac{1}{c_j^k} \prod_{j=1}^{h} \Pr(S_{d_j} \to S'_{d_j}) = \prod_{j=1}^{h} \Pr(S_{d_j} \to S'_{d_j}) \sum_{k=1}^{K} \prod_{j=1}^{h} \frac{1}{c_j^k}. \tag{2.12}$$

Similarly, the transition probability when there are multiple paths of moving back to the current graph from the proposed graph is

$$\Pr(\mathbf{S'} \to \mathbf{S}) = \sum_{k=1}^{K} \prod_{j=1}^{h} \frac{1}{c_j^{k'}} \prod_{j=1}^{h} \Pr(S'_{d_j} \to S_{d_j}) = \prod_{j=1}^{h} \Pr(S'_{d_j} \to S_{d_j}) \sum_{k=1}^{K} \prod_{j=1}^{h} \frac{1}{c_j^{k'}}. \tag{2.13}$$

In equations (2.12) and (2.13) the summation over $K$ represents the different paths (Section 2.2.4) to get from one graph to another and the last equality holds because the edges that are different between $\mathbf{S}$ and $\mathbf{S'}$ do not depend on the path $k$.

For each path $k$

$$\mathbf{C}^k = (c_1^k, c_2^k, c_3^k, \ldots, c_h^k) \tag{2.14}$$

$$= (\underbrace{c_1^k, c_2^k, \ldots, c_j^k}_{\text{create cycle(s)}}, \underbrace{c_{j+1}^k, \ldots, c_h^k}_{\text{remove cycle(s)}}) \tag{2.15}$$

$$= (\underbrace{m, \ldots, m}_{j}, c_{j+1}^k, \ldots, c_h^k). \tag{2.16}$$

The first $j$ elements can create one or more directed cycles. The remaining $h - j$ elements then remove the cycle(s) that were introduced in the network and their values are equal to the number of edges that make up the directed cycle that is being removed. The cycles that are created and removed in any path $k$ from $\mathbf{S}$ to $\mathbf{S'}$ can also be created and removed when moving from $\mathbf{S'}$ to $\mathbf{S}$. Therefore, the equations for moving from the proposed graph to the current graph will be the same as Equations 2.14 - 2.16 except for the $'$ symbol indicating that we are moving backwards:

$$\mathbf{C}^{k'} = (c_1^{k'}, c_2^{k'}, c_3^{k'}, \ldots, c_h^{k'}) \tag{2.17}$$

$$= (\underbrace{c_1^{k'}, c_2^{k'}, \ldots, c_j^{k'}}_{\text{create cycle(s)}}, \underbrace{c_{j+1}^{k'}, \ldots, c_h^{k'}}_{\text{remove cycle(s)}}) \tag{2.18}$$

$$= (\underbrace{m, \ldots, m}_{j}, c_{j+1}^{k'}, \ldots, c_h^{k'}), \tag{2.19}$$

and

$$\sum_{k=1}^{K}\prod_{j=1}^{h}\frac{1}{c_j^k}=\sum_{k=1}^{K}\prod_{j=1}^{h}\frac{1}{c_j^{k'}}. \tag{2.20}$$

The terms in equation 2.20 will cancel when calculating the acceptance probability and we will be left with $\prod_{j=1}^{h}\Pr(S_{d_j}\rightarrow S'_{d_j})$ and $\prod_{j=1}^{h}\Pr(S'_{d_j}\rightarrow S_{d_j})$. ■

### 2.2.4 Examples of removing directed cycles

#### 2.2.4.1 *Example 1 – one directed cycle*

The candidate graph for this example (Figure 2.1a) consists of five edges. Consider the current graph in Figure 2.1b with states $\mathbf{S}=(0,0,0,1,1)$ and the proposed graph in Figure 2.1d with states $\mathbf{S'}=(0,1,2,1,1)$. Edges #2 and #3 have different states between $\mathbf{S}$ and $\mathbf{S'}$ therefore $\mathbf{D}=2,3$. There are two different paths to move from $\mathbf{S}$ to $\mathbf{S'}$. In path 1 there are two steps: i) edge #2 changes direction which creates a directed cycle between nodes $T_1$, $T_2$, and $T_3$ (Figure 2.1c) and ii) the directed cycle is removed by edge #3 changing from state 0 to 2. Therefore, the first element of $\mathbf{C}^1$ is five because in the first step all five of the edges could change state. The second element of $\mathbf{C}^1$ is three because the directed cycle, consisting of edges #1, #2, and #3, must be removed and only these three edges can change state to remove this cycle. In path 2 edges #2 and #3 change states in one step and $\mathbf{C}^2=5,5$ because for both edges that are different between $\mathbf{S}$ and $\mathbf{S'}$ all five edges could change state. In other words, no directed cycles were introduced and then removed. If the prior on edge states is $p_0=0.05$, $p_1=0.05$ and $p_2=0.9$ then the probabilities for the two paths are

$$\text{path 1: }\Pr(S_{d_2}\rightarrow S'_{d_2})=\frac{0.05}{0.95},\ \Pr(S_{d_3}\rightarrow S'_{d_3})=\frac{0.9}{0.95},\ \mathbf{C}^1=5,3$$

and

$$\text{path 2: }\Pr(S_{d_2}\rightarrow S'_{d_2})=\frac{0.05}{0.95},\ \Pr(S_{d_3}\rightarrow S'_{d_3})=\frac{0.9}{0.95},\ \mathbf{C}^2=5,5.$$

Combining the probabilities from each path we obtain the transition probability of moving to the proposed graph:

$$\Pr(\mathbf{S}\rightarrow\mathbf{S'})=\frac{1}{5}\frac{1}{3}\frac{0.05}{0.95}\frac{0.9}{0.95}+\frac{1}{5}\frac{1}{5}\frac{0.05}{0.95}\frac{0.9}{0.95}=\left(\frac{1}{5}\frac{1}{3}+\left(\frac{1}{5}\right)^2\right)\frac{0.05}{0.95}\frac{0.9}{0.95}. \tag{2.21}$$

Any directed cycle created when moving from $\mathbf{S}$ to $\mathbf{S'}$ needs to be created when moving from $\mathbf{S'}$ to $\mathbf{S}$. Therefore, when moving from $\mathbf{S'}$ to $\mathbf{S}$ there are also two paths. Path 1 is made up of two steps: i) edge #3 changes from state 2 to 0 creating a cycle between nodes $T_1$, $T_2$, and $T_3$ and ii) the cycle is removed

by changing edge #2 from state 1 to 0. For path 2 edges #2 and #3 both change states in one step. The probabilities for the paths are

$$\text{path 1: } \Pr(S'_{d_2} \to S_{d_2}) = \frac{0.05}{0.95},\ \Pr(S'_{d_3} \to S_{d_3}) = \frac{0.05}{0.1},\ \mathbf{C}^{1'} = 5, 3$$

and

$$\text{path 2: } \Pr(S'_{d_2} \to S_{d_2}) = \frac{0.05}{0.95},\ \Pr(S'_{d_3} \to S_{d_3}) = \frac{0.05}{0.1},\ \mathbf{C}^{2'} = 5, 5.$$

The transition probability of moving back to the current graph is

$$\Pr(\mathbf{S'} \to \mathbf{S}) = \frac{1}{5}\frac{1}{3}\frac{0.05}{0.95}\frac{0.05}{0.1} + \frac{1}{5}\frac{1}{5}\frac{0.05}{0.95}\frac{0.05}{0.1} = \left(\frac{1}{5}\frac{1}{3} + \left(\frac{1}{5}\right)^2\right)\frac{0.05}{0.95}\frac{0.05}{0.1}. \tag{2.22}$$

The term $\frac{1}{5}\frac{1}{3}+\left(\frac{1}{5}\right)^2$ in equations 2.21 and 2.22 cancels out when calculating the acceptance probability, $\alpha$, and we are left with the probability of moving between the states that differ between the current graph (Figure 2.1b) and the proposed graph (Figure 2.1d). More generally, we can apply the same procedure to traverse the paths between any two graphs.

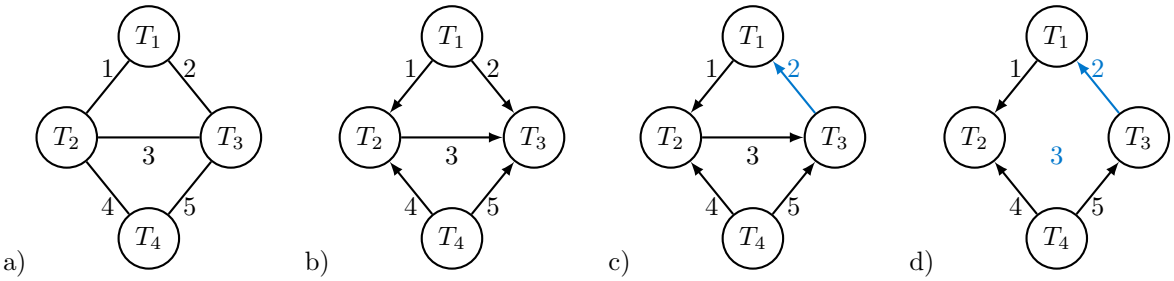



Figure 2.1: **Graphs used for illustrating one directed cycle being introduced and then removed.** a) The candidate graph showing which edges are considered in the inference. b) The current graph. c) An intermediate graph between the current graph and the proposed graph where a directed cycle has been introduced into the network. d) The proposed graph after the directed cycle has been removed.

#### 2.2.4.2 *Example 2 – multiple directed cycles*

We show a second more complex example below using the same candidate graph from Figure 2.1a. If we start with the graph in Figure 2.2a with states $\mathbf{S} = (0, 0, 0, 1, 1)$ and the proposed graph in Figure 2.2d with states $\mathbf{S'} = (2, 1, 1, 1, 0)$ there are four edges with different states between the two graphs and $\mathbf{D} = 1, 2, 3, 5$. There are three different paths to move from $\mathbf{S}$ to $\mathbf{S'}$. The steps in path 1 are: i) edges #2 and #5 change directions creating two directed cycles, the first cycle is between nodes $T_1$, $T_2$, and $T_3$ and the second cycle is between nodes $T_2$, $T_3$, and $T_4$ (Figure 2.2b), ii) edge #1 changes from state 0 to 2

removing the first cycle (Figure 2.2c), and iii) edge #3 changes direction which removes the second cycle (Figure 2.2d). The first two elements in the vector $\mathbf{C}^1$ are five because all five edges could change state in the first step which involves two edges changing states. In this step two directed cycles are created (one involving edges #1, #2, and #3 and the other involving edges #3, #4, and #5). The third element of $\mathbf{C}^1$ is three because only edges #1, #2, and #3 can change state to remove this cycle. The fourth element of $\mathbf{C}^1$ is also three because only edges #3, #4, and #5 can change state to remove the second cycle. The steps in path 2 are: i) edges #1, #2, and #5 all change states creating one directed cycle between nodes $T_2$, $T_3$, and $T_4$ (Figure 2.2c) and ii) edge #3 changes direction removing the cycle. Here, the first three elements of $\mathbf{C}^2$ are five because all five of the edges can change state in the first step. In the second step the cycle made up of edges #3, #4, and #5 must be removed. The fourth element of $\mathbf{C}^2$ is three because one of these edges must change state to remove this directed cycle. In path 3 edges #1, #2, #3, and #5 all change state in one step. Therefore, $\mathbf{C}^3$ is five for all four elements of this vector because all five edges could change state in this step. If the prior on edge states is $p_0 = 0.05$, $p_1 = 0.05$, and $p_2 = 0.9$ then the probabilities for the three paths are

$$\text{path 1: } \Pr(S_{d_1} \to S'_{d_1}) = \frac{0.9}{0.95}, \Pr(S_{d_2} \to S'_{d_2}) = \frac{0.05}{0.95}, \Pr(S_{d_3} \to S'_{d_3}) = \frac{0.05}{0.95}, \Pr(S_{d_5} \to S'_{d_5}) = \frac{0.05}{0.95},$$
$$\mathbf{C}^1 = 5, 5, 3, 3,$$

$$\text{path 2: } \Pr(S_{d_1} \to S'_{d_1}) = \frac{0.9}{0.95}, \Pr(S_{d_2} \to S'_{d_2}) = \frac{0.05}{0.95}, \Pr(S_{d_3} \to S'_{d_3}) = \frac{0.05}{0.95}, \Pr(S_{d_5} \to S'_{d_5}) = \frac{0.05}{0.95},$$
$$\mathbf{C}^2 = 5, 5, 5, 3,$$

and

$$\text{path 3: } \Pr(S_{d_1} \to S'_{d_1}) = \frac{0.9}{0.95}, \Pr(S_{d_2} \to S'_{d_2}) = \frac{0.05}{0.95}, \Pr(S_{d_3} \to S'_{d_3}) = \frac{0.05}{0.95}, \Pr(S_{d_5} \to S'_{d_5}) = \frac{0.05}{0.95},$$
$$\mathbf{C}^3 = 5, 5, 5, 5.$$

Therefore, the transition probability of moving to the proposed graph is

$$\begin{aligned}\Pr(\mathbf{S} \to \mathbf{S'}) &= \frac{1}{5}\frac{1}{5}\frac{1}{3}\frac{1}{3}\frac{0.9}{0.95}\frac{0.05}{0.95}\frac{0.05}{0.95}\frac{0.05}{0.95} + \frac{1}{5}\frac{1}{5}\frac{1}{5}\frac{1}{3}\frac{0.9}{0.95}\frac{0.05}{0.95}\frac{0.05}{0.95}\frac{0.05}{0.95} + \frac{1}{5}\frac{1}{5}\frac{1}{5}\frac{1}{5}\frac{0.9}{0.95}\frac{0.05}{0.95}\frac{0.05}{0.95}\frac{0.05}{0.95} \\ &= \left(\left(\frac{1}{5}\right)^2\left(\frac{1}{3}\right)^2 + \left(\frac{1}{5}\right)^3\frac{1}{3} + \left(\frac{1}{5}\right)^4\right)\frac{0.9}{0.95}\frac{0.05}{0.95}\frac{0.05}{0.95}\frac{0.05}{0.95}. \end{aligned} \tag{2.23}$$

As in example 1 any directed cycle that is created when moving from $\mathbf{S}$ to $\mathbf{S'}$ needs to also be created when moving from $\mathbf{S'}$ to $\mathbf{S}$. There are also three different paths to move back to $\mathbf{S}$ from $\mathbf{S'}$. In path 1 the steps are: i) edges #1 and #3 change states creating two directed cycles the first cycle is between

nodes $T_1$, $T_2$, and $T_3$ and the second cycle is between nodes $T_2$, $T_3$, and $T_4$, ii) edge #2 changes direction removing the first cycle, and iii) edge #5 changes direction removing the second directed cycle. Path 2 has two steps i) edges #1, #2 and #3 all change state creating a directed cycle between nodes $T_2$, $T_3$, and $T_4$ and ii) edge #5 changes direction removing the cycle. In path 3 there is only one step where edges #1, #2, #3, and #5 all change states. The probabilities for these paths are

$$\text{path 1: } \Pr(S'_{d_1} \to S_{d_1}) = \frac{0.05}{0.1}, \Pr(S'_{d_2} \to S_{d_2}) = \frac{0.05}{0.95}, \Pr(S'_{d_3} \to S_{d_3}) = \frac{0.05}{0.95}, \Pr(S'_{d_5} \to S_{d_5}) = \frac{0.05}{0.95},$$
$$\mathbf{C}^{1'} = 5, 5, 3, 3,$$

$$\text{path 2: } \Pr(S'_{d_1} \to S_{d_1}) = \frac{0.05}{0.1}, \Pr(S'_{d_2} \to S_{d_2}) = \frac{0.05}{0.95}, \Pr(S'_{d_3} \to S_{d_3}) = \frac{0.05}{0.95}, \Pr(S'_{d_5} \to S_{d_5}) = \frac{0.05}{0.95},$$
$$\mathbf{C}^{2'} = 5, 5, 5, 3,$$

and

$$\text{path 3: } \Pr(S'_{d_1} \to S_{d_1}) = \frac{0.05}{0.1}, \Pr(S'_{d_2} \to S_{d_2}) = \frac{0.05}{0.95}, \Pr(S'_{d_3} \to S_{d_3}) = \frac{0.05}{0.95}, \Pr(S'_{d_5} \to S_{d_5}) = \frac{0.05}{0.95},$$
$$\mathbf{C}^{3'} = 5, 5, 5, 5.$$

Therefore, the transition probability of moving back to the current graph is

$$\begin{aligned}\Pr(\mathbf{S'} \to \mathbf{S}) &= \frac{1}{5}\frac{1}{5}\frac{1}{3}\frac{1}{3}\frac{0.05}{0.1}\frac{0.05}{0.95}\frac{0.05}{0.95}\frac{0.05}{0.95} + \frac{1}{5}\frac{1}{5}\frac{1}{5}\frac{1}{3}\frac{0.05}{0.1}\frac{0.05}{0.95}\frac{0.05}{0.95}\frac{0.05}{0.95} + \frac{1}{5}\frac{1}{5}\frac{1}{5}\frac{1}{5}\frac{0.05}{0.1}\frac{0.05}{0.95}\frac{0.05}{0.95}\frac{0.05}{0.95} \\ &= \left(\left(\frac{1}{5}\right)^2\left(\frac{1}{3}\right)^2 + \left(\frac{1}{5}\right)^3\frac{1}{3} + \left(\frac{1}{5}\right)^4\right)\frac{0.05}{0.1}\frac{0.05}{0.95}\frac{0.05}{0.95}\frac{0.05}{0.95}. \end{aligned} \tag{2.24}$$

The term $\left(\frac{1}{5}\right)^2\left(\frac{1}{3}\right)^2 + \left(\frac{1}{5}\right)^3\frac{1}{3} + \left(\frac{1}{5}\right)^4$ in equations 2.23 and 2.24 cancels out when calculating the acceptance probability, $\alpha$, and we are left with the probability of moving between the states that differ between the current graph (Figure 2.2a) and the proposed graph (Figure 2.2d). More generally, we can apply the same procedure to traverse the paths between any two graphs.

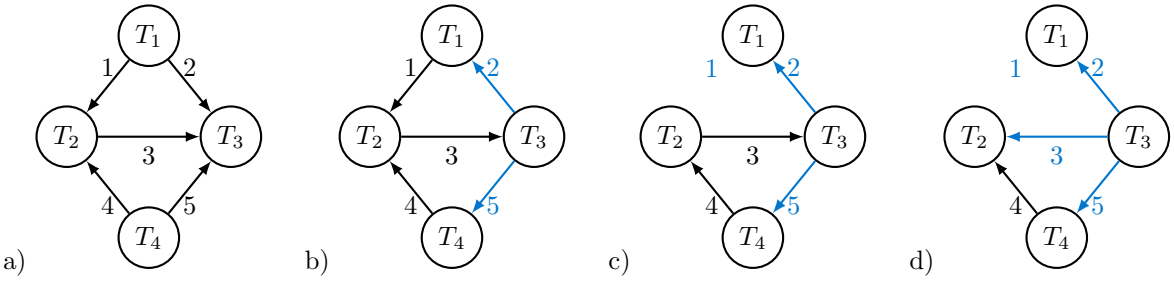



Figure 2.2: **Four graphs used for illustrating multiple directed cycles being introduced and then removed.** a) The current graph. b) An intermediate graph between the current graph and the proposed graph where two directed cycles have been introduced into the graph. c) An intermediate graph where one of the directed cycles has been removed. d) The proposed graph.

### 2.2.5 Cycle finder algorithm

1. Find the nodes that are connected to two or more nodes as a cycle contains at least three nodes and each node in a cycle has at least one incoming edge and one outgoing edge. To do so, we use the following steps:

   i. Add the adjacency matrix to its transpose.

   ii. Sum each row and delete rows (the row indices are preserved) with a sum less than 2.

   iii. Apply the following rule to the adjacency matrix

   $$\forall A_{j,k} = 1, \begin{cases} A_{j,k} = 0 \text{ if } k \notin J \\ A_{j,k} = 1 \text{ if } k \in J, \end{cases}$$

   where $J$ is the set of remaining row indices in the adjacency matrix.

   iv. Repeat steps ii and iii until $\forall A_{j,k} = 1$, $k \in J$ or the reduced adjacency matrix has two or fewer rows.

2. Create a tree as deep as possible starting with the node (i.e., root node) whose index is in the first row of the reduced adjacency matrix. To do this we create a branch, which is a vector of node indices, for each of the nodes (i.e., child nodes) connected to the root. For each branch we add the index of the child node to the end of the vector, repeating the process of a child node becoming the parent node, until we add an index that has been added to the branch previously. If a parent node has two or more children a new branch is created for each child node.

3. Remove the nodes from the branches that do not belong to the cycle. In addition to the nodes that create a cycle a branch may also contain nodes outside the cycle. To remove these nodes, we start

at the last node (i.e., the leaf) of the branches created in the previous step and work up the branch until we come to a node index that matches the leaf. Nodes above this node are then removed. For example, a branch may be $(3, 4, 1, 6, 5, 2, 1)$ and the trimmed branch would be $(1, 6, 5, 2, 1)$.

4. Convert the trimmed branches of a cycle into a vector of adjacency matrix coordinates by pairing up the adjacent nodes in each trimmed branch to produce a vector of edge coordinates. For example, if a trimmed branch is $(1, 6, 5, 2, 1)$ then the adjacency matrix coordinates for the edges between the nodes in the cycle are $\big((1, 6), (6, 5), (5, 2), (2, 1)\big)$.

5. Generate a vector of edge states from the vector of edge coordinates by considering each pair of coordinates and comparing the first number to the second number. If the first number is smaller than the second number the edge state is 0, indicating that the edge points from the node with a smaller index to the node with the larger index. If the first number is larger than the second number the edge state is 1, indicating that the edge points from the node with a larger index to the node with a smaller index. The coordinates vector from the example in Step 4 is $\big((1, 6), (6, 5), (5, 2), (2, 1)\big)$ and the edge states that form a directed cycle are $(0, 1, 1, 1)$.

6. Use the vector of edge coordinates from step 4 to create a vector of edge indices. For example, the edge indices for the coordinates $\big((1, 6), (6, 5), (5, 2), (2, 1)\big)$ are $(2, 6, 5, 1)$.

7. Define each cycle by the edges in the cycle and the states that form a cycle Using the vector of edge states (step 5) and the vector of edge indices (step 6). The cycle vector for the $j$th cycle is

$$(S_1, S_2, \ldots, S_m) \tag{2.25}$$

where $m$ is the number of edges in the network and each $S_k$ can take one of three values:

$$S_k = \begin{cases} 9 & \text{if this edge is not involved in the } j\text{th cycle} \\ 1 & \text{if edge } S_k \text{ is in state 1} \\ 0 & \text{if edge } S_k \text{ is in state 0.} \end{cases}$$

For example, Figure 2.3 shows a graph with two cycles nested within a larger cycle. If the edges are oriented as shown in the figure the cycle vector for the cycle involving edges $(2, 3, 5, 6, 7)$ with states $\mathbf{S} = (0, 1, 0, 0, 0)$ is

$$(9, 0, 1, 9, 0, 0, 0),$$

and the cycle vector for the cycle involving edges $(1, 3, 4, 5, 6, 7)$ with states $\mathbf{S} = (0, 1, 0, 0, 0, 0)$ is

$$(0, 9, 1, 0, 0, 0, 0).$$

Even though the cycles in the example above have four edges in common with the same edge direction for each of these edges, we are able to distinguish between the two cycles by comparing the cycle vectors elementwise.

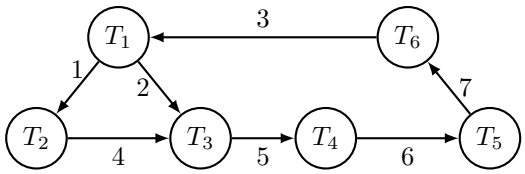



Figure 2.3: **A graph with two nested directed cycles.** One directed cycle is made up of edges $\{2, 5, 6, 7, 3\}$ and the other with edges $\{1, 4, 5, 6, 7, 3\}$. The two cycles have four edges in common.

### 2.2.6 Cycle remover algorithm

1. For the proposed graph create a cycle vector, from the proposed edge states, for each set of edges that could form a directed cycle. For example, if the graph shown in Figure 2.4b is the proposed graph then the cycle vector would be $(0, 2, 1, 1)$.

2. For each set of edges that could form a directed cycle, compare the cycle vector of the proposed graph with the cycle vector of the directed cycles elementwise. For example, if the graph shown in Figure 2.4a is the true graph then the cycle vectors for the directed cycles would be $(0, 1, 0, 1)$ and $(1, 0, 1, 0)$. It is important to note that for each set of edges that could form a directed cycle there are two cycles: one with the edges oriented clockwise and another with the edges oriented counterclockwise. The cycle vector (created in the previous step) for the proposed graph is $(0, 2, 1, 1)$ which does not match either of the directed cycle vectors. Therefore, the proposed graph does not have any directed cycles.

3. For each directed cycle, randomly select an edge and change it from its current state to a different state according to the prior probability of edge states.

4. Repeat steps 1 - 3 until there are no directed cycles in the graph.


Figure 2.4: **Two graphs used for illustrating the steps of the cycle remover algorithm.** a) The true graph and b) the proposed graph.

### 2.2.7 Relationship to existing Bayesian methods

We compare baycn to five methods (Table 2.1), four of which are also MCMC methods. Two methods are based on the graph structure: Gibbs [29] and MC$^3$ [55], and two methods are based on node orderings or node partitions (i.e., subsets of nodes): order MCMC [25, 45] and partition MCMC [44]. These methods assign a prior for the entire graph or for node orderings/partitions.

We also compare with scanBMA [89], one of the more recent Bayesian model averaging methods that specifies a prior probability for individual edges [88, 52, 89]. However, scanBMA searches the space of parents for each node in the network, for example, the possible parent nodes for node $X$, and independently the possible parent nodes for node $Y$. If two nodes $X$ and $Y$ are connected, the posterior probability of $X \leftarrow Y$ and that of $X \rightarrow Y$ are independent of each other, and can each take the value 1. As a result, scanBMA is unable to detect the v structure of $X \rightarrow Y \leftarrow Z$ and tends to include an edge between $X$ and $Z$: although $X$ and $Z$ are marginally independent, $Z$ becomes dependent on $X$ in the presence of $Y$, and scanBMA thus infers both $Z$ and $Y$ to be parents of $X$, and similarly, both $X$ and $Y$ to be parents of $Z$.

## 2.3 Simulation Studies

### 2.3.1 Data simulation

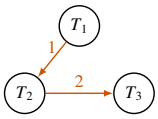
We simulated data under seven different topologies (Figure 2.5). Each node follows a normal distribution where the mean is a linear model (Equation 2.6). For all nodes the variance is set to one and the intercept of the linear model, $\beta_0$, is set to zero. In addition, all other $\beta$s take the same value and are referred to as the signal strength. We used three values for the signal strength (similar to Badsha and Fu [4]): 0.2 (weak), 0.5 (moderate), and 1 (strong), and three values for the sample size: 100, 200 and 600. For each topology we simulated 25 data sets under each of the nine combinations of signal strength and sample size. When summarizing the results, if the results are grouped by, for example, topology, we

Table 2.1: **Summary of our method and other Bayesian methods for network inference.**

| | baycn | Gibbs [29] | $MC^3$ [55] | order MCMC [25, 45] | partition MCMC [44] | scanBMA [89] |
|---|---|---|---|---|---|---|
| Input | Data matrix, adjacency matrix, prior on edge states | Data matrix, a list of edges to be considered can be specified, prior on graph space | Data matrix, a list of edges to be considered can be specified, prior on graph space | Data matrix, edges to be considered can be specified, parameters to calculate the graph score | Data matrix, edges to be considered can be specified, parameters to calculate the graph score | Data matrix, prior for edge presence |
| Output | Posterior probabilities of three edge states | Posterior probabilistic adjacency matrix | Posterior probabilistic adjacency matrix | Posterior probabilistic adjacency matrix | Posterior probabilistic adjacency matrix | Posterior probabilities of parent nodes |
| Sampling space | Edges and edge directions | Edges and edge directions | Edges and edge directions | Node orderings | Node partitions | Parent nodes |
| Type of Sampling | Edges and edge directions | Edges and edge directions | Edges and edge directions | Node orderings | Node partitions | Linear models |
| MCMC Move | Select a subset of edges and change their states | Select a subset of nodes and change their parent nodes | Add or remove an edge | Move a node to a new position | Split an existing partition or merge two partitions | NA |
| Type of prior | Edge states | Parameters and graph structure | Parameters and graph structure | Parameters and node orderings | Parameters and node partitions | Parameters and edges |
| Type of data | Discrete, continuous, or mixed | Discrete or continuous | Discrete or continuous | Discrete or continuous | Discrete or continuous | Continuous |

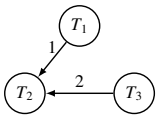
use the output from all the data sets with different sample size and signal strength values.

(a) M1



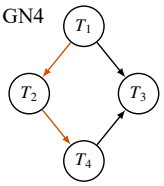

| | Expected Probability | | |
|---|---|---|---|
| Edge | 0 | 1 | 2 |
| 1 | 0.33 | 0.67 | 0 |
| 2 | 0.67 | 0.33 | 0 |

(b) M2

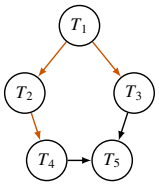



| | Expected Probability | | |
|---|---|---|---|
| Edge | 0 | 1 | 2 |
| 1 | 1 | 0 | 0 |
| 2 | 0 | 1 | 0 |

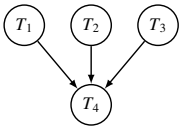
(c) GN4



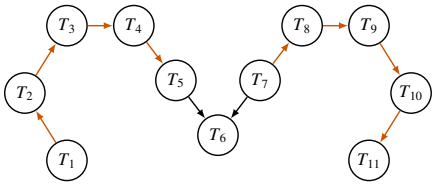

(d) GN5




(e) multi-parent




(f) GN11




(g) GN8

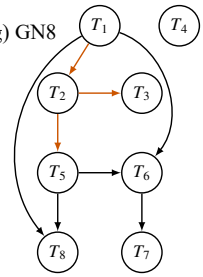



Figure 2.5: **Seven topologies used in simulation studies.** Orange edges have Markov equivalent edges and cannot be deterministically inferred. Two graphs are Markov equivalent if they have the same likelihood and represent the same conditional independence [82]. A set of Markov equivalent graphs form a Markov equivalence class. Here, the graph $T_1 \to T_2 \to T_3$ (M1 in **a**) is Markov equivalent to $T_1 \leftarrow T_2 \to T_3$, and $T_1 \leftarrow T_2 \leftarrow T_3$; all depict marginal dependence between $T_1$ and $T_3$ (i.e., $T_1 \not\perp\!\!\!\perp T_3$) and conditional independence given $T_2$ (i.e., $T_1 \perp\!\!\!\perp T_3 \mid T_2$). In contrast, the graph $T_1 \to T_2 \leftarrow T_3$ (M2 in **b**), also known as a v structure, has no Markov equivalent graphs. In this graph $T_1$ and $T_3$ are marginally independent (i.e., $T_1 \perp\!\!\!\perp T_3$) and conditionally dependent given $T_2$ (i.e., $T_1 \not\perp\!\!\!\perp T_3 \mid T_2$). The probabilities of the edge states for each edge in the inferred graph need to account for Markov equivalence (see the expected probabilities for the states of each edge in **a** and **b**). (**c**)-(**g**) Larger networks that contain M1 and M2 as subgraphs. Graphs GN4 (**c**), GN5 (**d**), and GN8 (**g**) all have edges that can potentially form one or more directed cycles. The multi-parent graph (**e**) consists of three v structures, and GN11 (**f**) consists of two subgraphs separated by a v structure and has more Markov equivalent graphs than the other networks.

### 2.3.2 Metrics for assessing method performance

We assess the performance and compare methods using the following metrics:

1. The edgewise Mean Squared Error (eMSE): This is the MSE between the expected and posterior probabilities of the three states for edge $j$:

$$\text{eMSE}_j = \frac{1}{3}\sum_{k=0}^{2}\left[\Pr(S_j = k) - \Pr(S_j = k \mid \mathbf{T})\right]^2, \tag{2.26}$$

   where $\Pr(S_j = k)$ is the expected probability under Markov equivalence. This metric informs us which edges are more accurately inferred and which ones are not.

2. $\text{MSE}_1$: This is the MSE for the whole graph based on three possible edge states. It is the eMSE

averaged over all $m$ edges in the graph:

$$\mathrm{MSE}_1 = \frac{1}{m}\sum_{j=1}^{m}\mathrm{eMSE}_j = \frac{1}{3m}\sum_{j=1}^{m}\sum_{k=0}^{2}\big[\Pr(S_j = k) - \Pr(S_j = k \mid \mathbf{T})\big]^2. \tag{2.27}$$

3. $\mathrm{MSE}_2$: This is the MSE between the expected and posterior probabilistic adjacency matrices on all $m$ edges. This metric is essentially the same as $\mathrm{MSE}_1$, but makes it easy to compare with other Bayesian methods, which generally produce a probabilistic adjacency matrix.

$$\mathrm{MSE}_2 = \frac{1}{2m}\sum_{\substack{(j,k) \text{ or} \\ (k,j)\in\mathbf{E}}}\Big\{\big[\Pr(A_{jk}=1) - \Pr(A_{jk}=1 \mid \mathbf{T})\big]^2 + \big[\Pr(A_{kj}=1) - \Pr(A_{kj}=1 \mid \mathbf{T})\big]^2\Big\}. \tag{2.28}$$

   For baycn, we use the posterior probability for state 0 and state 1 in place of $\Pr(A_{jk} = 1 \mid \mathbf{T})$ and $\Pr(A_{kj} = 1 \mid \mathbf{T})$ respectively.

4. Precision and power for the whole graph. Precision or $1 -$ False Discovery Rate (FDR) measures how many of the inferred edges are in the true graph, and power measures how many of the inferred edges are true edges.

$$\text{Precision} = 1 - \text{FDR} = \frac{\#\text{ of true edges inferred}}{\#\text{ of inferred edges}} \tag{2.29}$$

$$\text{Power} = \frac{\#\text{ of true edges inferred}}{\#\text{ of edges in true graph}} \tag{2.30}$$

   To calculate these metrics for simulated data, we apply a cutoff of 0.5 to the posterior probability of edge presence (i.e., the sum of the probability of both directions). These metrics ignore the nuances in the probabilities, but are easy to interpret as percentages and provide a quick indication of the inference accuracy.

### 2.3.3 Estimating posterior probabilities of edge states using true edges as the input

For each topology, we ran baycn once per simulated data set, used a burn-in of 20%, and used the prior, $(p_0, p_1, p_2) = (0.05, 0.05, 0.9)$, on edge states. For M1, M2, GN4, GN5, and multi-parent topologies (Figure 2.5a-e), we ran baycn for 30,000 iterations with a step size of 120. For GN8 and GN11 (Figure 2.5f-g), we ran baycn for 50,000 iterations with a step size of 200.

For this simulation we calculate $\text{MSE}_1$ (see Section 2.3.2), which measures the discrepancy between the expected and posterior probabilities across all edges in the network. In general, the performance of baycn depends on the signal strength, $\beta$, and sample size, $N$. As expected, for each topology $\text{MSE}_1$ decreases as both $N$ and $\beta$ increase (Table 2.2). The signal strength has a larger effect on $\text{MSE}_1$, because for all seven topologies $\text{MSE}_1$ is much lower for a strong signal and small sample size than for a weak signal and large sample size. A close examination of the graphs inferred for different values of $\text{MSE}_1$ shows that an $\text{MSE}_1$ of below 0.1 typically corresponds to accurate inference: the direction of each edge is correctly inferred, and the posterior probabilities of each edge state is similar to that of the expected probabilities. Using this guideline, we observed that baycn performs well on GN4, GN5, and GN8 for $\beta = 0.5$ even with a small sample size, indicating that baycn can find and remove directed cycles from the graph. For GN11 baycn can correctly identify the edge directions in smaller subgraphs separated by a v structure. For both M2 and multi-parent topologies that only contain v structures, $\text{MSE}_1$ is nearly perfect at $\beta = 0.5$ and 1 but is much larger at $\beta = 0.2$. This is consistent with our observation that it is generally difficult for existing graph inference algorithms to correctly identify v structures with a weak signal [4, 5].

### 2.3.4 Identification of false positive edges and the choice of priors

For this assessment we include a false edge in M1, M2 and GN4, and two false edges in GN11 (Figure 2.6). We used the same data generated in the previous section for these topologies (without false edges) and ran baycn with the input being the true edges plus the false edges. We explored the impact of three edge-state priors on the inference, with an increasing probability of the edge being absent: prior 1: $(p_0, p_1, p_2) = (1/3, 1/3, 1/3)$; prior 2: $(p_0, p_1, p_2) = (0.25, 0.25, 0.5)$, and prior 3: $(p_0, p_1, p_2) = (0.05, 0.05, 0.9)$.

We calculated the edgewise MSE or eMSE (see Section 2.3.2) for each edge in the network, which measures the discrepancy between the expected and posterior probabilities for the three edge states. Similar to the previous section, we use eMSE $< 0.1$ as the criterion for correct inference. We observed that baycn can identify false positive edges under prior 3, which assigns a large prior probability to edge absence (Tables 2.3 - 2.6). In all four graphs the eMSE for the false edges decreases as $p_2$ increases. On the other hand, the edge probabilities of the true edges are generally estimated correctly under all three priors, even when the false edges are not properly identified and have a large eMSE (Tables 2.3 - 2.6). This investigation confirms that prior 3 is the prior of choice, as it balances the need to detect false positive edges and to correctly infer true edges.

Table 2.2: **Performance of baycn on all the graphs used in simulation studies.** Features of the graphs, such as the number of edges and v structures, are listed (also see Figure 2.5). The mean and standard deviation of $MSE_1$ (on three states of each edge; see Section 2.3.2), sample size $N$, and signal strength $\beta$ are also listed. For each simulation scenario we generated 25 independent data sets and ran baycn once on each data set.

| | | | | MSE | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| | | | | $\beta = 0.2$ | | $\beta = 0.5$ | | $\beta = 1$ | |
| Topology | # edges | # v structures | $N$ | mean | sd | mean | sd | mean | sd |
| M1 | 2 | 0 | 100 | 0.1796 | 0.0949 | 0.0127 | 0.0353 | 0.0011 | 0.0016 |
| | | | 200 | 0.0734 | 0.074 | 0.0014 | 0.0015 | 0.0012 | 0.0013 |
| | | | 600 | 0.0237 | 0.0326 | 0.001 | 0.0009 | 0.0008 | 0.0007 |
| M2 | 2 | 1 | 100 | 0.3384 | 0.1081 | 0.0909 | 0.0919 | 0 | 0 |
| | | | 200 | 0.2688 | 0.0881 | 0.0597 | 0.082 | 0 | 0 |
| | | | 600 | 0.1323 | 0.0735 | 0 | 0 | 0 | 0 |
| GN4 | 4 | 1 | 100 | 0.2731 | 0.0711 | 0.0674 | 0.0417 | 0.01 | 0.0276 |
| | | | 200 | 0.167 | 0.0554 | 0.0755 | 0.0639 | 0.0046 | 0.0142 |
| | | | 600 | 0.0839 | 0.0259 | 0.0503 | 0.0602 | 0.0069 | 0.0182 |
| GN5 | 5 | 1 | 100 | 0.2687 | 0.0785 | 0.0396 | 0.0465 | 0.0022 | 0.0023 |
| | | | 200 | 0.1562 | 0.049 | 0.0171 | 0.0235 | 0.002 | 0.0031 |
| | | | 600 | 0.0684 | 0.0338 | 0.0114 | 0.0483 | 0.002 | 0.0024 |
| Mulit-parent | 3 | 3 | 100 | 0.3361 | 0.0961 | 0.0486 | 0.0698 | 0 | 0 |
| | | | 200 | 0.237 | 0.0812 | 0.0032 | 0.0135 | 0 | 0 |
| | | | 600 | 0.1418 | 0.0625 | 0 | 0.0002 | 0 | 0 |
| GN11 | 10 | 1 | 100 | 0.2266 | 0.0513 | 0.0353 | 0.0225 | 0.0042 | 0.0031 |
| | | | 200 | 0.139 | 0.0431 | 0.0121 | 0.0162 | 0.0066 | 0.0041 |
| | | | 600 | 0.0613 | 0.0197 | 0.0047 | 0.0043 | 0.0046 | 0.0039 |
| GN8 | 8 | 2 | 100 | 0.2959 | 0.1517 | 0.0667 | 0.0666 | 0.0247 | 0.0526 |
| | | | 200 | 0.197 | 0.144 | 0.0692 | 0.0962 | 0.0186 | 0.0486 |
| | | | 600 | 0.1147 | 0.1073 | 0.0556 | 0.0858 | 0.0109 | 0.0467 |

### 2.3.5 Comparison with existing Bayesian methods

For the comparison with the MCMC methods we focus on GN4, GN8 and GN11, which have different levels of complexity (Figure 2.5; Table 2.7). Using the true edges as the input to all the methods, we ran each method for 30,000 iterations on GN4 and for 50,000 iterations on GN8 and GN11 and used a burn-in of 20%. We set the step size in baycn to 120 for GN4 and 200 for GN8 and GN11. We used the default step size for order and partition MCMC, which results in the step size being 30 for GN4 and 50 for GN8 and GN11. Gibbs and $MC^3$ use all of the iterations after the burn-in.

We compare the performance of each MCMC method with $MSE_2$ (see Section 2.3.2), which is calculated between the expected and posterior probability for the two directions each edge can take across all edges in the network. Again, we use $MSE_2 < 0.1$ as the criterion for accurate inference. We observed

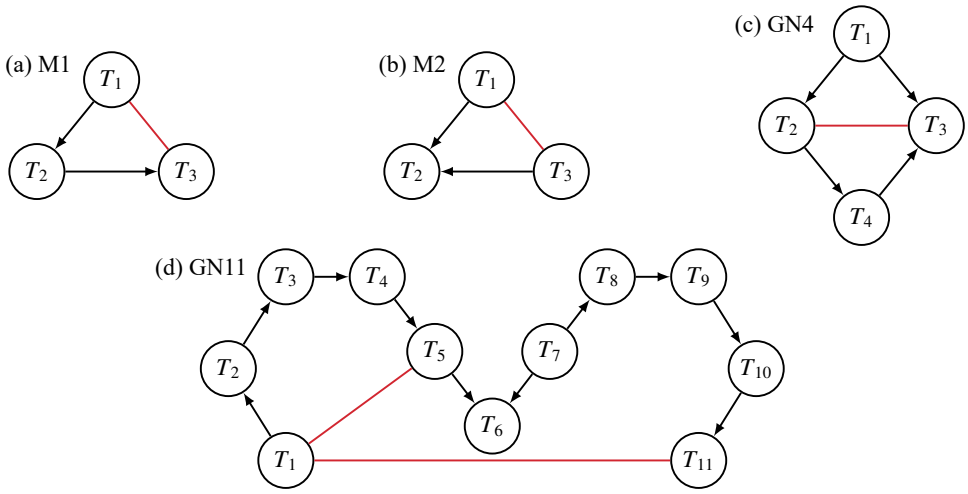



Figure 2.6: **The black edges (true edges) were used to simulate the data and the red edges (false edges) were added to the true adjacency matrix as input to baycn.**

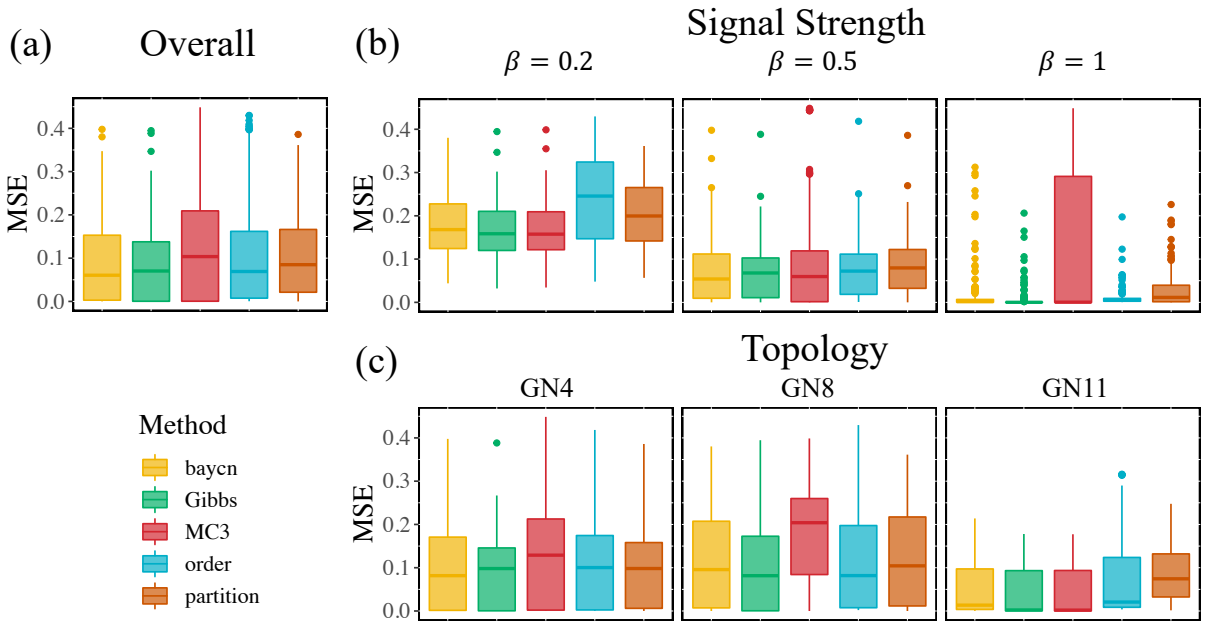



Figure 2.7: **Boxplots of $MSE_2$ (on the posterior probabilistic adjacency matrix) for baycn and other Bayesian methods.** We simulated data from three topologies: GN4, GN8, and GN11 in Figure 2.5 with three values of signal strength and sample size. The true edges were used as the input to each method. We grouped $MSE_2$ (see Section 2.3.2) in different ways to highlight the performance of each method in different simulation scenarios. (**a**) The overall $MSE_2$ grouped by method. This plot combines the $MSE_2$ for all topologies, signal strengths, and sample sizes. (**b**) $MSE_2$ grouped by method and signal strength $\beta$. Here the $MSE_2$ for all topologies and sample sizes are combined. (**c**) $MSE_2$ grouped by method and topology. In these plots the $MSE_2$ for the signal strengths and sample sizes are combined. Also see Table 2.7.

that all five methods have similar $MSE_2$, with $MC^3$ having larger mean $MSE_2$ and larger variation (Figure 2.7a). Since the signal strength is shown in the previous sections to be crucial to the performance, we grouped the results by signal strength. In general $MSE_2$ decreases as the signal strength increases, except for $MC^3$ which performs poorly at $\beta = 1$ (Figure 2.7b). Grouping the results by topology, we observed that $MSE_2$ is generally small for GN11, which is not as complex despite having the largest number of nodes. In summary, across different simulation scenarios, baycn and Gibbs show similar and better performance than other methods. Partition and order MCMC are slightly worse in some cases, whereas $MC^3$ is the least accurate and least stable among the MCMC methods.

In addition, we compare with the MCMC methods and scanBMA on GN4. A fully connected graph was used as the input for the MCMC methods. We calculated precision and power (see Section 2.3.2) for all methods and $MSE_2$ for the MCMC methods, since the posterior probabilities from scanBMA do not have the same interpretation as the MCMC methods. We use the cutoff of 0.5 for edge presence for all methods. Precision is nearly perfect for the MCMC methods in most of the simulation scenarios (Figure 2.8; Table 2.8). However, for scanBMA precision is lower than all other methods for a strong and moderate signal, mainly due to its inability to infer the v structure. Power is nearly 100%, for all methods, for both $\beta = 0.5$ and 1 (Figure 2.8; Table 2.8). However, power is low for data sets with $\beta = 0.2$ and the variation is larger for order and partition MCMC, as they tend to infer a higher posterior probability for edge absence (Tables 2.9 - 2.11). Similar to the results when the true edges were used as input $MSE_2$ decreases as the signal strength increases. Similar to their performance on power, order and partition MCMC have a higher mean and variation at $\beta = 0.2$. We also note that Gibbs and $MC^3$ have slightly higher mean $MSE_2$ when $\beta = 1$.

To compare the runtime of the MCMC methods, we ran each algorithm on an Intel Xeon D-1540 2.00 GHz processor with 128 GB of memory on data from GN4, GN8, and GN11 with $\beta = 1$ and $N = 600$ (Table 2.12). For all three topologies, order MCMC is the fastest, followed closely by baycn and partition MCMC. baycn is approximately three times as fast as $MC^3$ and can be 50 times as fast as Gibbs.

## 2.4 Applications to biological data

### 2.4.1 Direct and indirect targets of genetic variants

Genetic variants play an important role in regulating the expression of genes [12]. Several large genomic consortia have identified widespread genetic variants associated with gene expression [46, 78]. Among them, the GEUVADIS (Genetic European Variation in Disease) project measured gene expression in Lymphoblastoid Cell Lines (LCLs) from a subset of individuals from the 1000 Genomes Project [2],

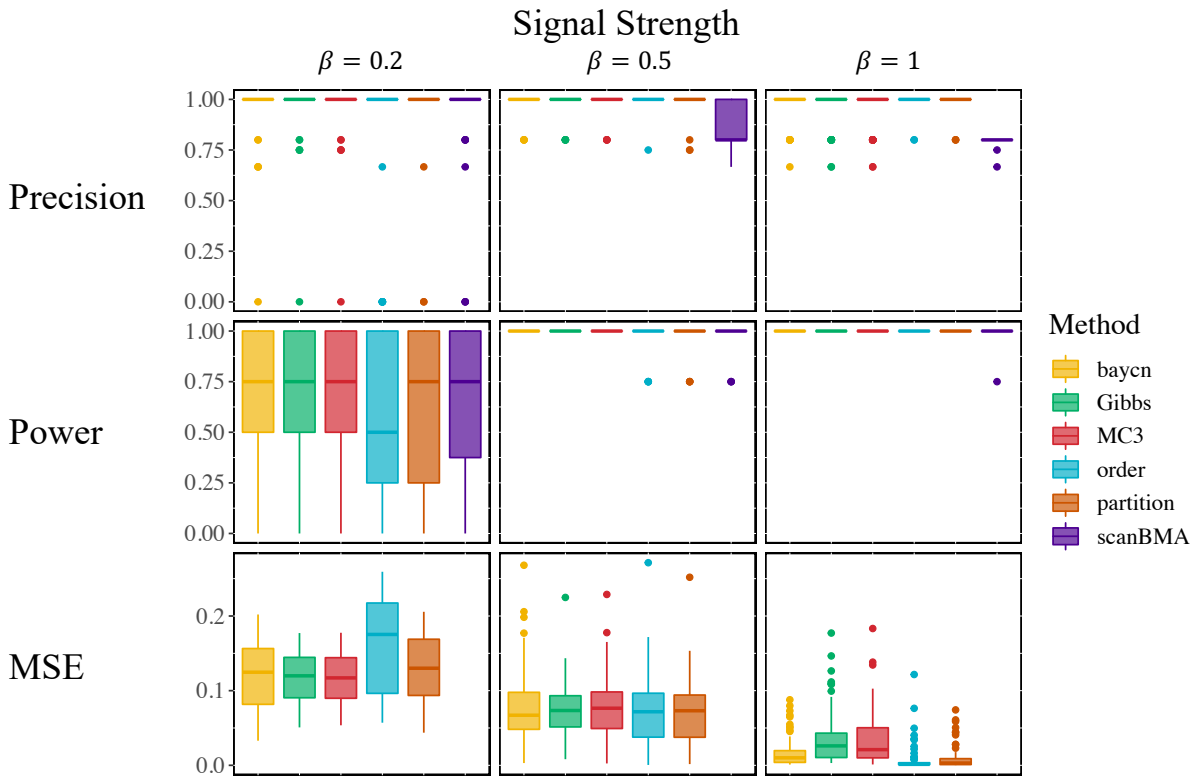



Figure 2.8: **Boxplots of precision, power, and $MSE_2$ for baycn and other Bayesian methods.** We simulated data from GN4 (Figure 2.5c) with three values of signal strength and sample size. A fully connected graph was used as the input to each method. We considered an edge present if the posterior probability for edge presence (the sum of the posterior for the two directions) was greater than 0.5. Precision, power, and $MSE_2$ (see Section 2.3.2) are grouped by method and signal strength $\beta$. For these plots each metric is combined across all three sample sizes. Also see Table 2.8. The last row only displays $MSE_2$ for the MCMC methods because scanBMA is unable to determine edge direction from static data.

which measured the genotypes of individuals of multiple ethnicities. GEUVADIS identified a large number of genetic variants that are associated with the expression of one or more genes: these variants are termed expression quantitative trait loci (eQTLs), and the associated genes are potential targets of the eQTL. In particular, 62 eQTLs are associated with more than one gene. However, since the association analysis examined one eQTL-gene pair at a time, it is unclear which associated genes are more likely to be the direct targets, and which ones the indirect targets. To address this question, we can infer a network for each eQTL and its associated genes: genes are more likely to be direct targets if they are directly connected to the eQTL.

We focused on the European sample, which has a decent sample size of 373, and applied baycn to each eQTL-gene set. Since the number of nodes in each graph is small, we used a fully connected graph as the input to baycn with a prior on edge states of $(p_0, p_1, p_2) = (0.05, 0.05, 0.9)$ . We also included the

constraint that a gene cannot be the parent of an eQTL. We ran baycn once per eQTL-gene set for 30,000 iterations with a burn-in of 20% and a step size of 120. When determining which edges to include in the inferred graph, we required a posterior probability of $> 0.4$ for edge presence. In addition, we considered an edge directed if the difference between the posterior probabilities for the two directions is greater than 0.2.

To validate the graphs inferred from GEUVADIS, we also used data from the GTEx (Genotype-Tissue Expression) consortium, which collects genotype and gene expression data from over 50 tissues from approximately 900 individuals [78]. We found that data are available for 46 eQTL-gene sets in LCLs from 115 individuals in GTEx. We applied baycn to these data with the same parameter settings as above, and present the results of seven eQTL-gene sets: four sets each involve two genes and three sets each involve three genes.

We further examined the effect of confounding variables on the inference. On one hand, the gene expression data in GEUVADIS and GTEx had been normalized using the PEER method [75] to remove potential impact of demographic variables, batch effect, and other covariates. On the other hand, gene regulation is a complex process, and genes not included in an eQTL-gene set may also have an impact on the set. To account for this type of confounding, we included Principal Components (PCs) associated with each eQTL-gene set in the network inference. We performed a principal component analysis on the gene expression data for all genes from GEUVADIS. We kept the top ten PCs and calculated the Pearson correlation between each of these PCs and each eQTL and gene from all the eQTL-gene sets. Then we tested all pairwise correlations for statistical significance using the q value method [77] with the false discovery rate (FDR) = 0.05. The PCs that were significantly associated with an eQTL-gene set were then included in the network as confounding variables. We used a fully connected graph as the input to baycn for each eQTL-gene set that had at least one PC associated with it (excluding the edges between any two PC nodes). On each eQTL-gene-PC set we ran baycn for 50,000 iterations with a burn-in of 20% and a step size of 200. We used the same priors and criteria for edge presence and direction as above.

These analyses show that baycn can identify the regulatory relationship among multiple genes associated with the same eQTL, while accounting for the effect of confounding variables (Figure 2.9, Figure 2.10). The posterior probabilities estimated by baycn are largely consistent with the sample correlation and give a detailed description of the relationship among the variables (Figure 2.10). For example, in the eQTL-gene set Q8, baycn infers the gene PNP to be a direct target of the eQTL rs11305802 for both GEUVADIS and GTEx (Figure 2.9a, b), consistent with the correlations (0.56 in GEUVADIS and 0.4 in GTEx). In comparison, baycn infers an edge between rs11306802 and the gene RP11-203M5.8 for GEUVADIS but not for GTEx, also consistent with the correlations (0.41 in GEUVADIS and 0.23

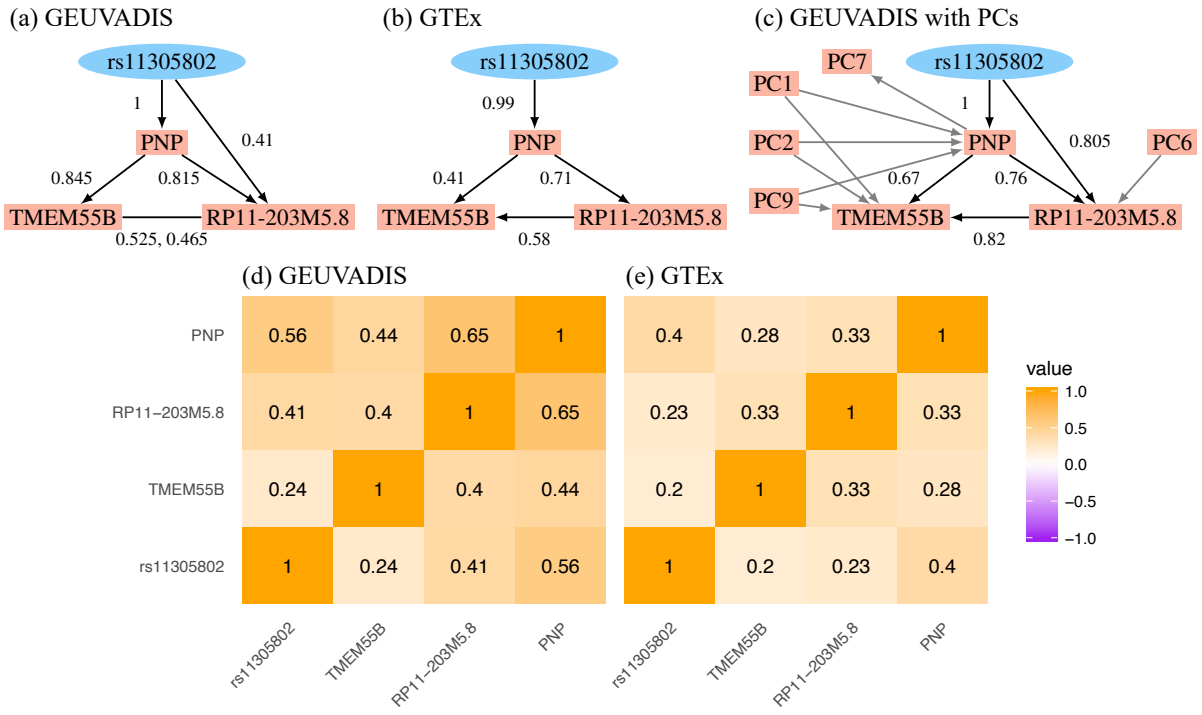



Figure 2.9: **The graph inferred by baycn and correlation heat maps for eQTL-gene set Q8 from GEUVADIS and GTEx.** The numbers in the graphs indicate the posterior probability of the edge for the direction shown. For undirected edges the posterior probability for each direction is shown. We consider and edge present if the posterior probability of edge presence is greater than 0.4 and an edge is considered directed if the difference between the posterior probabilities for the two directions is greater than 0.2. (**a**) Inferred graph for data from GEUVADIS. A fully connected graph was used as the input to baycn. (**b**) Inferred graph for data from GTEx. A fully connected graph was used as the input to baycn. (**c**) Inferred graph for data from GEUVADIS with five PCs included in the network as confounding variables. A fully connected graph (excluding the edges between any two PCs) was used as the input to baycn. The edges involving PC nodes are shown in gray; the posterior probability for these edges is not included in the graph but can be found in Table 2.14. (**d**) and (**e**) Correlation heat maps for the nodes of interest.

in GTEx). Similarly, the edge between TMEM55B and RP11-203M5.8 is inferred to be present in both consortia, also consistent with the correlations, although the direction is ambiguous. Including PCs does not affect the DAG inference for Q8 and has no or small impact on other eQTL-gene sets (compare Table 2.13 to Tables 2.14 - 2.18).

The graphs inferred by baycn are the same for GEUVADIS and GTEx on the eQTL-gene sets Q20 and Q50 (Figure 2.10) while for Q21, Q23, Q37, and Q62 the inference is different between the two consortia (Figure 2.10). The eQTL-gene sets Q20 and Q50 (Figure 2.10) have similar correlation structures (Figure 2.11) between the two consortia which leads to similar posterior probabilities for the inferred edges (Table 2.13). For both Q23 and Q37 (Figure 2.10) no edges are inferred for GTEx. This inference also agrees

with the correlation structure of the two data sets (Figure 2.11) as the correlation, for both eQTL-gene sets in GTEx, is close to zero between most variables. Of the five edges inferred in GEUVADIS for Q21 (Figure 2.10) only two edges (rs147156488 – FAM27D1 and FAM27D1 – FAM27A) are inferred for GTEx. Again if we examine the correlation (Figure 2.11) we can see that the absence of these edges is expected. For Q62 (Figure 2.10) the only edge inferred for GTEx is between the two genes which is consistent with the near-zero correlation between the eQTL and the two genes.

When including PCs in the network the inferred graphs remain largely the same. For both Q37 and Q50 the same edges are inferred with and without PCs and the posterior probability for these edges are nearly identical (compare Table 2.13 to Tables 2.15 - 2.18). The eQTL-gene set Q21 has one edge less (between rs147156488 and FAM27C) when including the PCs in the network, but the posterior probabilities for the other edges are similar (compare Table 2.13 to 2.15). The eQTL-gene set Q23 has the same edges inferred with and without PCs, but the direction of the edge between AGAP9 and AGAP10 is not the same (Figure 2.10).

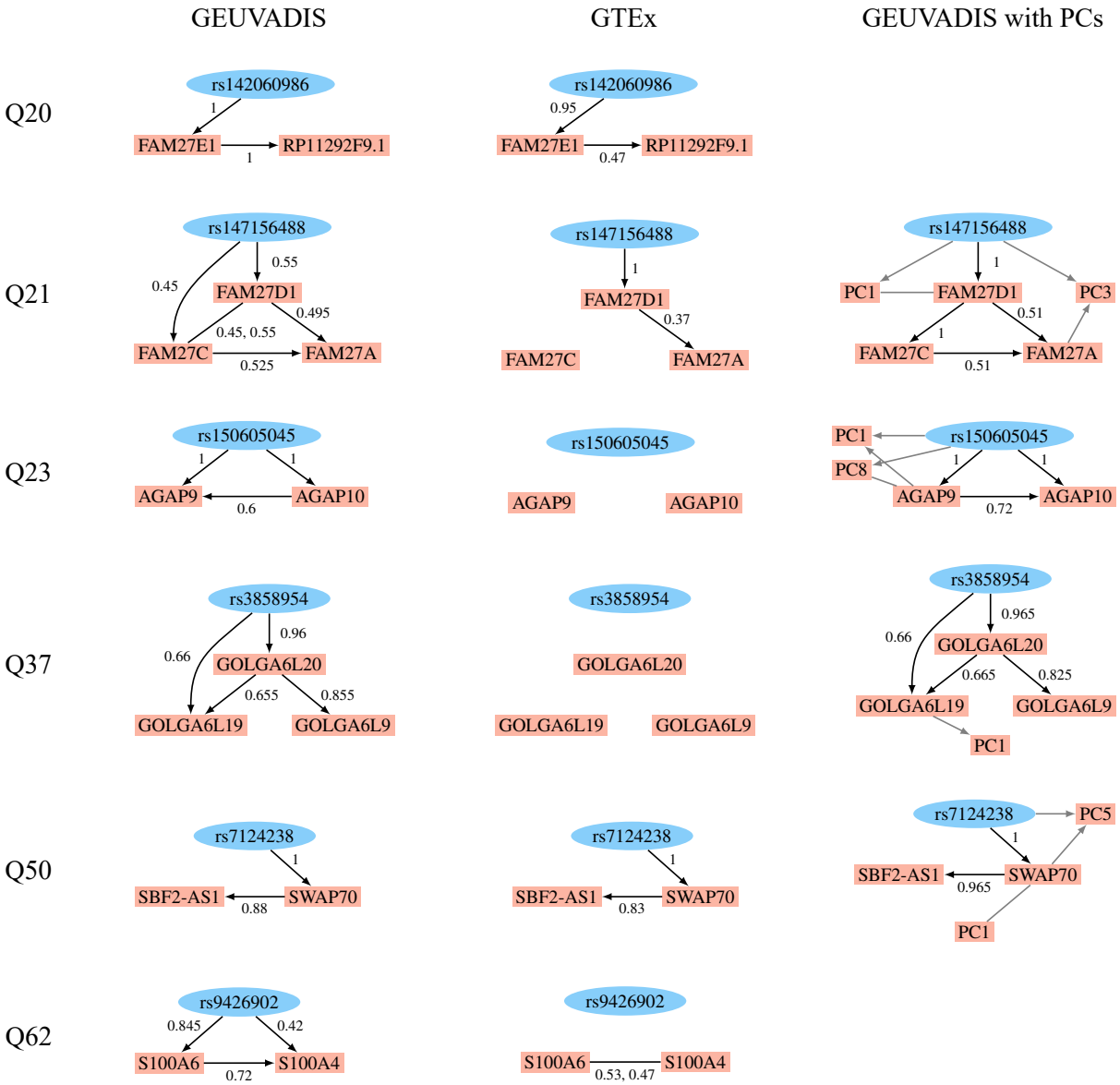



Figure 2.10: **The graphs inferred by baycn for eQTL-gene sets from GEUVADIS and GTEx.** The numbers next to an edge indicate the posterior probability of the edge for the direction shown. For undirected edges the posterior probability for each direction is shown. The third column shows the inference from baycn with PCs included in the eQTL-gene sets from GEUVADIS as confounding variables. The edges involving PC nodes are shown in gray; the posterior probability for these edges is not included in the graph but can be found in Tables 2.15 - 2.18. Neither eQTL-gene set Q20 or Q62 have any PCs significantly correlated with them.

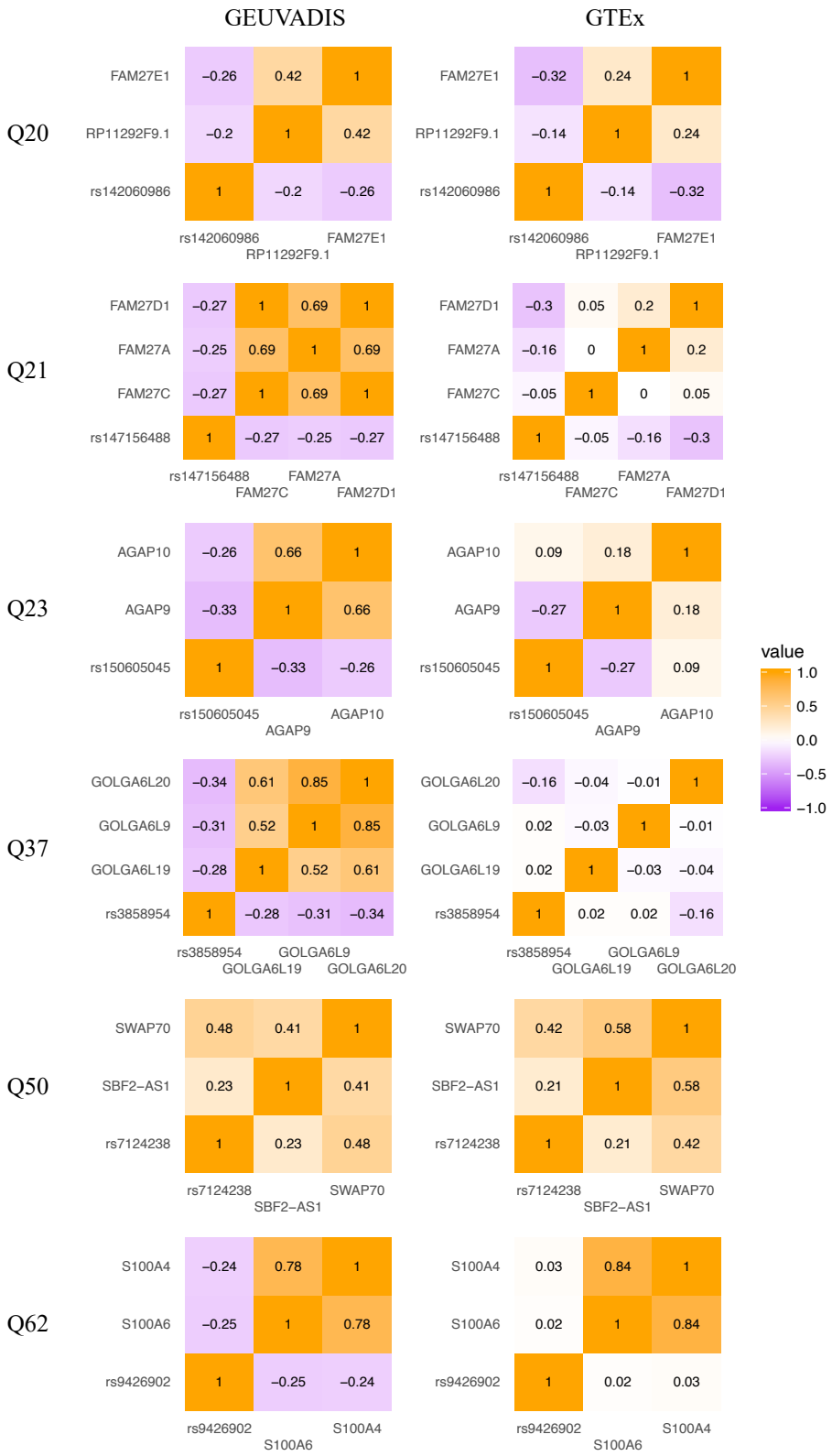



Figure 2.11: **Correlation heat maps for eQTL-gene sets from GEUVADIS and GTEx.**

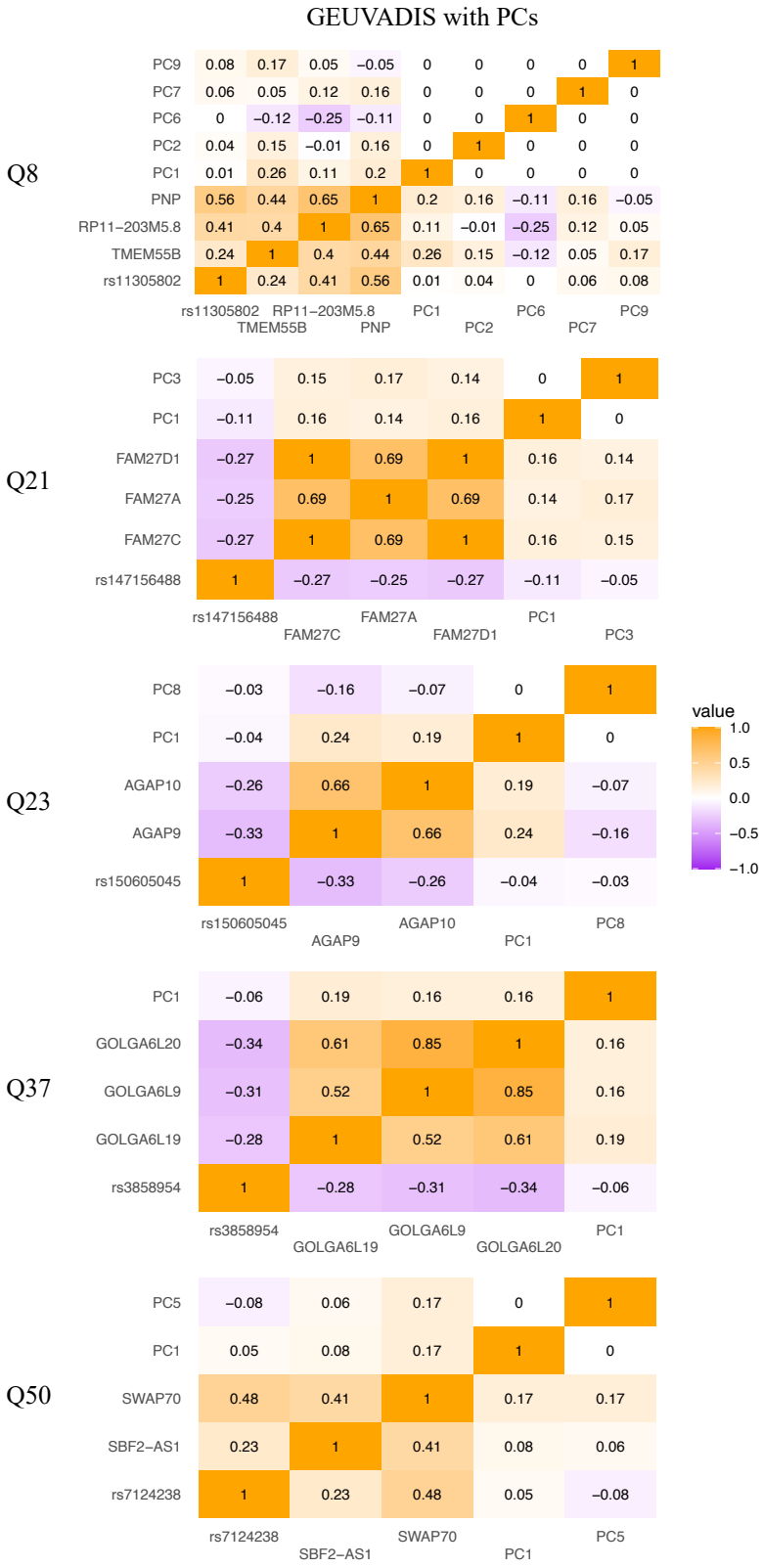



Figure 2.12: **Correlation heat maps for eQTL-gene-PC sets from GEUVADIS.**

### 2.4.2 Combinatorial binding of transcription factors

Transcription factors (TFs) regulate the expression of target genes by binding to regulatory sequences in the genome [17]. Often multiple TFs bind to the same regions and jointly influence gene expression – this combinatorial binding of TFs has been under intensive research for over two decades [84]. During early development in Drosophila, Zinzen et al. [94] showed that several TFs bind to *cis*-regulatory regions known as *cis*-regulatory modules (CRMs) during mesoderm differentiation. They measured in vivo binding for five key TFs using ChIP-chip assays at two hour intervals during mesoderm development in six tissue types in the embryos of *Drosophila melanogaster*: Twist (Twi) and Tinman (Tin) were both assayed from 2-8h, Myocyte enhancing factor 2 (Mef2) from 2-12h, Bagpipe (Bap) from 6-8h, and Biniou (Bin) from 6-12h. The six tissue types are mesoderm (Meso), somatic muscle (SM), visceral muscle (VM), mesoderm and somatic muscle (Meso&SM), visceral muscle and somatic muscle (VM&SM), and cardiac muscle (CM). In particular, the binary binding (binding versus no binding) profiles of TFs at 310 CRMs for tissue-specific genes was key to understand their combinatorial binding. Because of repeated measurements over time and combinatorial binding, there exist strong correlations among these binding profiles (Figure 2.14).

Previously, Stojnic et al. [76] analyzed this data set and constructed six separate graphs, one for each tissue type. Due to the strong correlation among TF binding profiles (e.g., among the five Mef2 binding profiles), most graph inference methods could not tell them apart and tended to infer a dense network. This earlier study therefore developed a unique vocabulary to define the dependence structure, as well as a novel algorithm for inference. However, this vocabulary differed substantially from that of the standard DAG and therefore was not straightforward to understand. Additionally, the graph inference method developed in this earlier study was not Bayesian, and it was unclear how reliable an inferred edge was in the presence of strong correlations.

We applied baycn to the Drosophila data and infer a network for the six tissues and five TFs. The number of nodes in this graph is fairly large, so we first used MRPC [5, 4] to generate a candidate graph. Among multiple correlated TF binding profiles that were associated with a tissue type (e.g., between multiple Mef2 profiles and Meso&SM), MRPC, being a conservative graph inference method, typically retained only one association. To better understand which binding profile was truly key to the tissue type, we modified the candidate graph to include additional edges (e.g., we included edges between Meso&SM and all five Mef2 profiles and all three Tin profiles; see Figure 2.15). We ran baycn for 500,000 iterations with a burn-in of 0.2 and a step size of 800. We used a prior of $(p_0, p_1, p_2) = (0.05, 0.05, 0.9)$ and required a posterior probability $> 0.4$ for edge presence. Again, we considered an edge directed if the difference

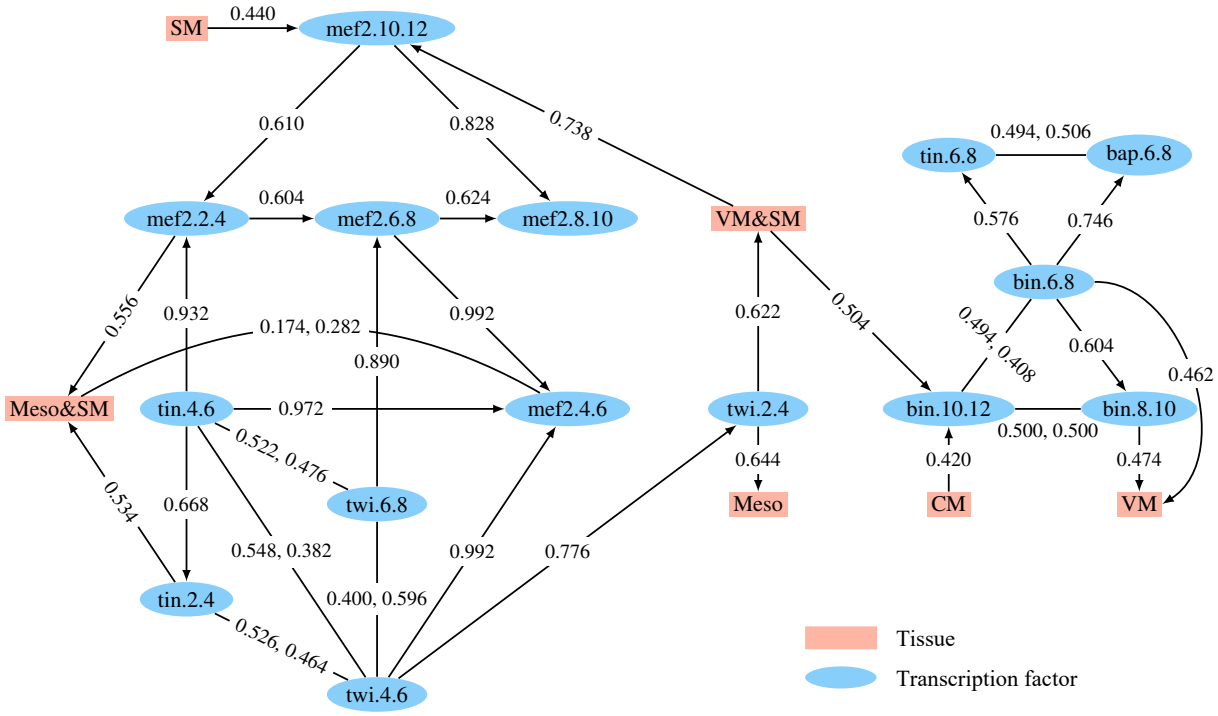



Figure 2.13: **The graph inferred by baycn for the transcription factor binding and tissue expression data [94] of the Drosophila embryo.** The five transcription factors are: Twist (Twi), Tinman (Tin), Myocyte enhancing factor 2 (Mef2), Bagpipe (Bap), and Biniou (Bin) and were assayed at two hour intervals during the first 12 hours of development. The six tissue types are: mesoderm (Meso), somatic muscle (SM), visceral muscle (VM), cardiac muscle (CM), mesoderm and somatic muscle (Meso&SM), visceral muscle and somatic muscle (VM&SM). An edge is considered present if the posterior probability of edge presence is greater than 0.4 and an edge is considered directed if the difference between the posterior probabilities for the two directions is greater than 0.2.

between the posterior probabilities for the two directions is greater than 0.2.

Our analysis shows that baycn can identify TFs known to drive tissue differentiation even when the TFs are highly correlated with one another and while considering the entire network: i) We confirmed the edge connecting a Twi node with Meso, when Twi is known to be a primary TF required for mesoderm formation [91]; ii) Twi is also known to directly regulate the expression of both Tin and Mef2 [69]; our inferred edges are consistent with these results; iii) Tin and Mef2 are essential for the specification of the dorsal mesoderm [3, 53] and muscle tissue differentiation [50, 8, 31]; in our inferred graph, there are edges between Meso&SM and these two TFs; and iv) Bap and Bin are both involved in the formation of visceral muscle [37, 90], and in our inference, both are also in the subgraph for VM.

Furthermore, while it is still difficult to identify which TF in the correlated binding profiles is essential to a tissue type, baycn is able to remove unlikely edges. For example, although the input graph contained all the edges between Meso&SM and Mef2 (five binding profiles) and Tin (three binding profiles), baycn

only inferred an edge between Meso&SM and Mef2 at 2-4h and 4-6h, as well as between Meso&SM and Tin at 2-4h (Figure 2.13). Additionally, baycn infers an edge between VM and two of the three time points for Bin (6-8h and 8-10h). It is particularly interesting to note that baycn inferred the edge between Meso&SM and Tin at 4-6h to be absent with a high posterior probability of 0.812 (Table 2.19), and that Tin at 4-6h impacts the tissue through Tin at 2-4h and Mef2 at 2-4h. In Stojnic et al., however, we could only deduce that all three nodes jointly influence this tissue, but were unable to disentangle the relationships among the TFs. The result from baycn suggests that the binding of Tin and Mef2 at earlier hours are more influential for tissue formation than the later binding of Tin.

We also ran partition MCMC on this data set and obtained similar results to that described above. We used the same graph for the input to partition MCMC and ran it for the same number of iterations and used the same burn-in of 20%. We used 0.5 as the probability cutoff for edge presence because partition MCMC lacks an edge-level prior. Additionally, even though the data are binary, we used the option for continuous data when running partition MCMC, the option for discrete data failed. Despite these differences, both algorithms agree on which edges are present and which are absent (Table 2.20). On the other hand, the two methods disagree on the direction of 13 inferred edges: for six of these edges, baycn infers them as directed while partition MCMC infers them as undirected; for five of them, partition MCMC infers them as directed and baycn infers them as undirected; and for two of the edges, baycn and partition MCMC infer the opposite direction. We discuss the interpretation of the inferred edge direction in Discussion.

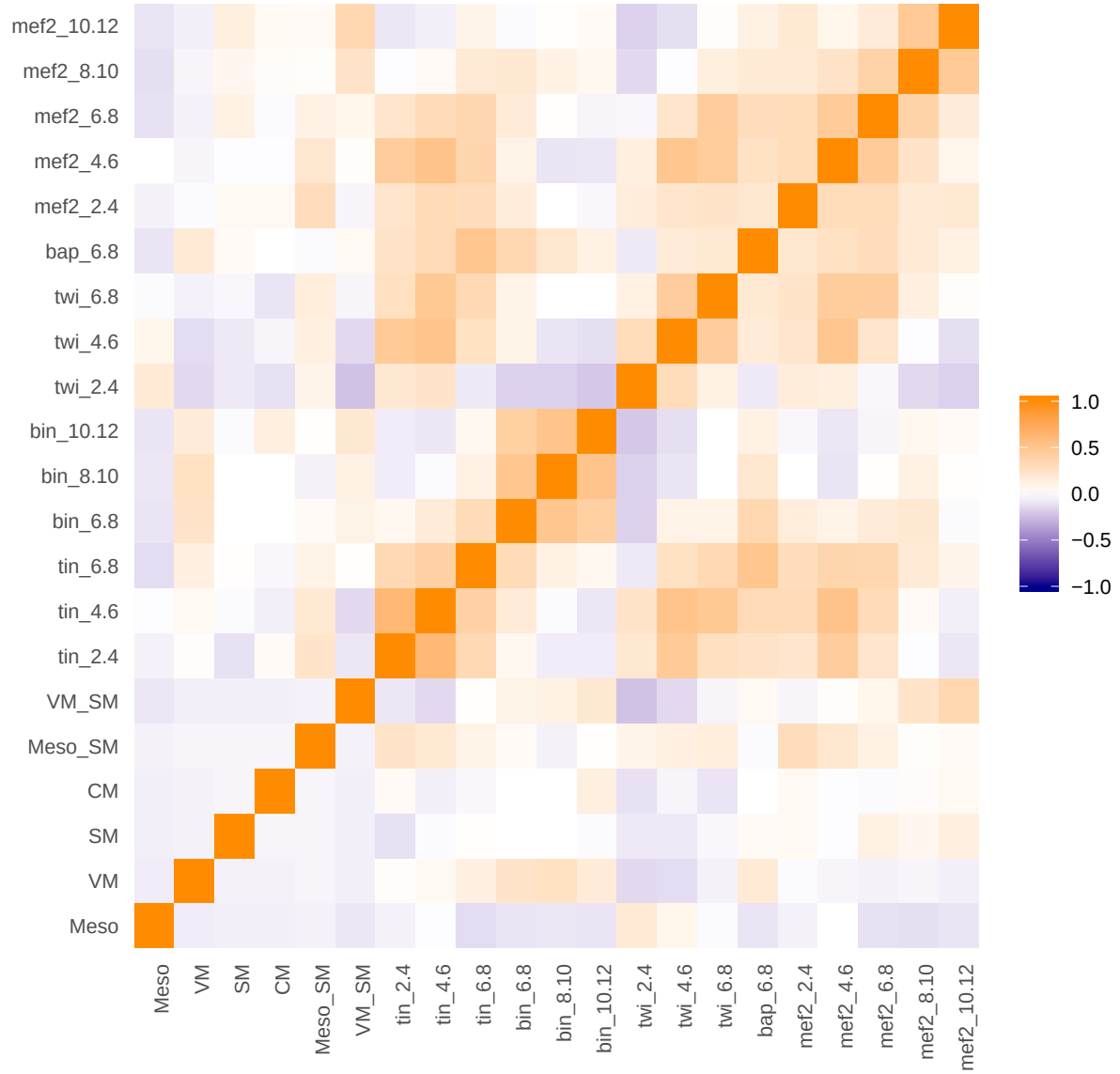



Figure 2.14: **Heat map of the Pearson correlation for each tissue and transcription factor in the Drosophila data set.**

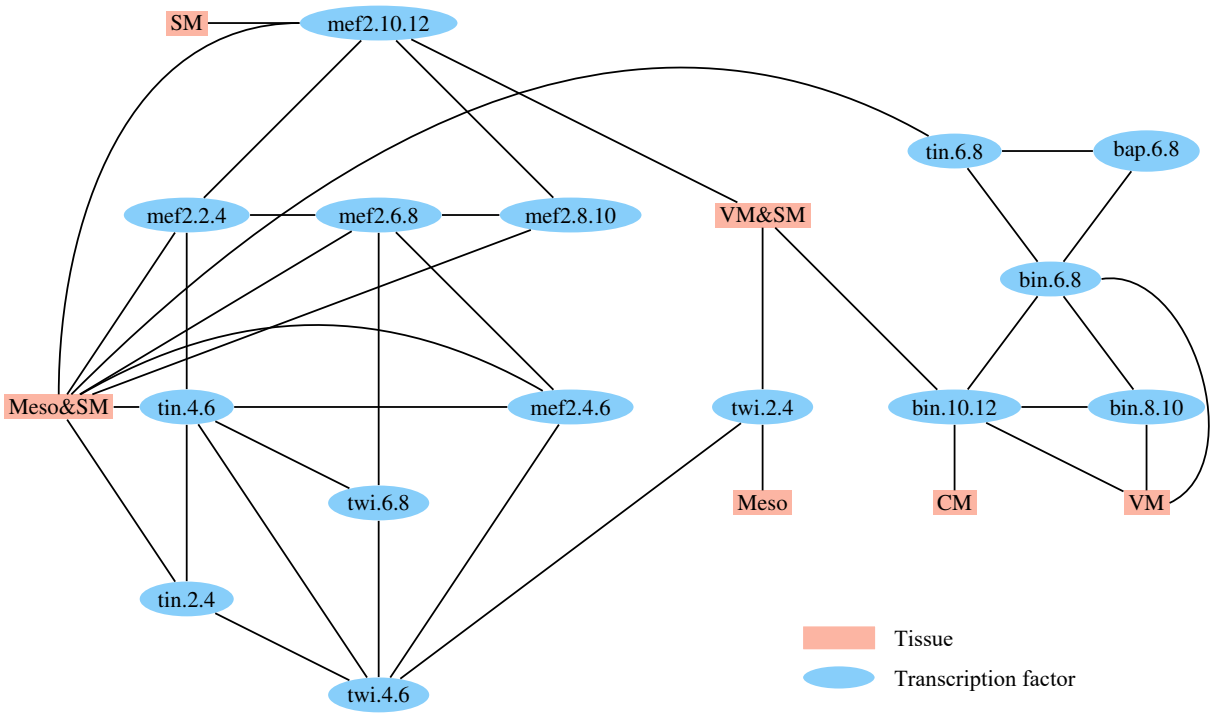



Figure 2.15: **Graph used as the input to baycn for the Drosophila data analysis.** We used the graph output by MRPC then added additional edges between Meso&SM and the transcription factors Mef2 and Tin and between VM and the transcription factor Bin.

## 2.5 Discussion

Here we present an alternative Bayesian approach to DAG inference. We have developed a new and coherent representation of a graph in terms of edge states. A prior distribution can then be assigned to an edge, and the posterior probabilities of edge states can be compared to the corresponding prior and interpreted accordingly. We have developed an MCMC algorithm for sampling under this representation. Our algorithm deals with directed cycles, accounts for Markov equivalence, and is applicable to diverse data types: continuous, discrete, and mixed. We have demonstrated through simulation studies that baycn is fast and can accurately estimate the edge probabilities in general, and that it performs as well as or better than current Bayesian methods in terms of precision, power, and MSE. This model makes it easy to experiment with different beliefs about the edge states: we can increase or decrease the prior probability for a certain edge state. With a suitable prior for edge states, baycn can correctly estimate edge probabilities both when the true edges are the input and when false edges are included in the input. However, the graphs we have analyzed here are fairly small and additional work is needed to handle much larger graphs.

We have used a generally conservative cutoff to interpret the posterior probabilities. In simulation studies, we used 0.5 as the cutoff value for the posterior probability of edge presence when calculating precision and power. This value is a natural choice for other Bayesian methods under comparison, since edge-level prior probabilities are unavailable. With baycn, 0.5 can be over conservative, especially when prior 3 of $(p_0, p_1, p_2) = (0.05, 0.05, 0.9)$ is used. Under this prior, a posterior probability of say, 0.2, for each of the two directions can already indicate strong evidence for edge presence, even though the edge presence probability is 0.4 in this case. However, precision and power for baycn with such a conservative cutoff are still nearly perfect in most cases and comparable to or better than other methods. When analyzing real data, we use a slightly lower cutoff of 0.4, as the edges inferred under this cutoff are more consistent with our interpretation of the correlations. In addition, to determine edge direction we use the cutoff of 0.2 for the difference between the posterior probabilities for edge direction. We demonstrate that while 0.2 is not a large difference it is sufficient to consider an edge directed. If we take multiple MCMC samples and just consider edge presence, then for any given edge we can approximate the proportion for one direction with a normal distribution. By the central limit theorem, the mean for this distribution is $p$ and the standard error is $\sqrt{p(1-p)/n}$, where $p$ is the proportion of the specified direction and $n$ is the size of the MCMC sample. For example, if the posterior probabilities for the two directions are 0.4 and 0.6 and we have a sample of 200, then the standard error would be $\sqrt{0.4 \times 0.6/200} = 0.0346$. Therefore, a difference of 0.2 between the two directions is already well into the tail of this distribution.

Another caveat when interpreting the inferred network is that the direction of an edge indicates statistical dependence. With additional assumptions, the direction may also indicate the actual, causal mechanism. In the application of the multiple target genes, for example, we constrained the edges between a genetic variant and a gene expression variable to always point to the gene expression variable. This is consistent with the biological principle that DNA regulates RNA, but not the other way around. With this constraint, other directed edges may suggest regulatory relationships. For example, the subgraph rs11305802 $\rightarrow$ PNP $\rightarrow$ TMEM55B in Figure 2.9, which is unaffected by the confounding variables we considered here, likely suggests that the eQTL rs11305803 regulates gene TMEM55B through gene PNP. In the application to the transcription factor binding data, however, no suitable constraint was applied. As a result, the direction in Figure 2.13 should be interpreted carefully. For example, the subgraph tin.4.6 $\rightarrow$ mef2.2.4 $\rightarrow$ Meso&SM indicates that the association between Tin at 4-6h and the tissue can be explained away by Mef2 at 2-4h; in other words, the formation of Meso&SM is more directly driven by Mef2 binding at 2-4h than by Tin binding at 4-6h. It cannot be interpreted as Tin binding at 4-6h regulates Mef2 binding at 2-4h. Similarly, the subgraph VM&SM $\rightarrow$ mef2.10.12 $\rightarrow$ mef2.8.10 also indicates that the association between Mef2 at 8-10h and VM&SM can be explained away by the later

binding of Mef2 at 10-12h, meaning that the formation of VM&SM is more directly associated with Mef2 binding at a later time than at an earlier time. Again, it cannot be interpreted as VM&SM regulates TF binding.

Table 2.3: **The mean and standard deviation of the edgewise MSE for each edge in topology M1.** We used the data sets previously simulated for M1 and included one false edge with the true edges in the input to baycn. We ran baycn with three different priors on edge states for each data set. The rows in gray represent false edges.

| | | eMSE: Topology M1 | | | | | |
|---|---|---|---|---|---|---|---|
| | | Prior 1 | | Prior 2 | | Prior 3 | |
| $N$ | Edge | mean | sd | mean | sd | mean | sd |
| 100 | 1 | 0.0109 | 0.006 | 0.0078 | 0.0047 | 0.002 | 0.0026 |
| | 2 | 0.2853 | 0.0568 | 0.1859 | 0.0618 | 0.0131 | 0.0139 |
| | 3 | 0.0107 | 0.0061 | 0.0091 | 0.0059 | 0.0014 | 0.0015 |
| 200 | 1 | 0.0116 | 0.0067 | 0.0075 | 0.005 | 0.002 | 0.0027 |
| | 2 | 0.2861 | 0.0816 | 0.1953 | 0.0992 | 0.03 | 0.0614 |
| | 3 | 0.0109 | 0.0063 | 0.0094 | 0.0046 | 0.0019 | 0.0025 |
| 600 | 1 | 0.0127 | 0.0062 | 0.008 | 0.0056 | 0.0008 | 0.001 |
| | 2 | 0.2665 | 0.038 | 0.1593 | 0.052 | 0.0095 | 0.0068 |
| | 3 | 0.0139 | 0.0085 | 0.0062 | 0.004 | 0.0016 | 0.0021 |

Table 2.4: **The mean and standard deviation of the edgewise MSE for each edge in topology M2.** We used the data sets previously simulated for M2 and included one false edge with the true edges in the input to baycn. We ran baycn with three different priors on edge states for each data set. The rows in gray represent false edges.

| | | eMSE: Topology M2 | | | | | |
|---|---|---|---|---|---|---|---|
| | | Prior 1 | | Prior 2 | | Prior 3 | |
| $N$ | Edge | mean | sd | mean | sd | mean | sd |
| 100 | 1 | 0.1366 | 0.0269 | 0.1197 | 0.0199 | 0.0224 | 0.026 |
| | 2 | 0.4035 | 0.0368 | 0.3285 | 0.0548 | 0.0656 | 0.0666 |
| | 3 | 0.1461 | 0.025 | 0.1241 | 0.0226 | 0.0232 | 0.028 |
| 200 | 1 | 0.1329 | 0.0262 | 0.1215 | 0.0222 | 0.0297 | 0.0354 |
| | 2 | 0.4111 | 0.0373 | 0.3474 | 0.0612 | 0.0863 | 0.0975 |
| | 3 | 0.1391 | 0.0227 | 0.1209 | 0.0245 | 0.0317 | 0.0396 |
| 600 | 1 | 0.1369 | 0.0207 | 0.1183 | 0.0221 | 0.0236 | 0.0164 |
| | 2 | 0.4062 | 0.0347 | 0.3362 | 0.0459 | 0.0647 | 0.0377 |
| | 3 | 0.1378 | 0.028 | 0.1186 | 0.0223 | 0.0224 | 0.0149 |

Table 2.5: **The mean and standard deviation of the edgewise MSE for each edge in topology GN4.** We used the data sets previously simulated for GN4 and included one false edge with the true edges in the input to baycn. We ran baycn with three different priors on edge states for each data set. The rows in gray represent false edges.

| | | eMSE: Topology GN4 | | | | | |
|---|---|---|---|---|---|---|---|
| | | Prior 1 | | Prior 2 | | Prior 3 | |
| $N$ | Edge | mean | sd | mean | sd | mean | sd |
| 100 | 1 | 0.0101 | 0.0182 | 0.011 | 0.0165 | 0.0418 | 0.0797 |
| | 2 | 0.011 | 0.0204 | 0.0098 | 0.0192 | 0.0431 | 0.0821 |
| | 3 | 0.2235 | 0.1185 | 0.1368 | 0.1257 | 0.0301 | 0.1031 |
| | 4 | 0.0085 | 0.0148 | 0.0078 | 0.0132 | 0.0462 | 0.0817 |
| | 5 | 0.0087 | 0.0162 | 0.0107 | 0.0203 | 0.0459 | 0.0888 |
| 200 | 1 | 0.0012 | 0.0023 | 0.0019 | 0.0036 | 0.0152 | 0.0414 |
| | 2 | 0.0004 | 0.0013 | 0.0012 | 0.004 | 0.0156 | 0.0462 |
| | 3 | 0.2154 | 0.079 | 0.1139 | 0.0669 | 0.005 | 0.0083 |
| | 4 | 0.0013 | 0.002 | 0.0023 | 0.0045 | 0.0181 | 0.0432 |
| | 5 | 0.0003 | 0.001 | 0.0013 | 0.0043 | 0.012 | 0.0317 |
| 600 | 1 | 0.0008 | 0.0015 | 0.0009 | 0.001 | 0.0103 | 0.0387 |
| | 2 | 0 | 0 | 0 | 0 | 0.0102 | 0.0402 |
| | 3 | 0.2534 | 0.1019 | 0.1544 | 0.1028 | 0.0092 | 0.0201 |
| | 4 | 0.0007 | 0.0012 | 0.0012 | 0.0015 | 0.011 | 0.038 |
| | 5 | 0 | 0 | 0 | 0 | 0.0112 | 0.0503 |

Table 2.6: **The mean and standard deviation of the edgewise MSE for each edge in topology GN11.** We used the data sets previously simulated for GN11 and included two false edges with the true edges in the input to baycn. We ran baycn with three different priors on edge states for each data set. The rows in gray represent false edges.

| | | eMSE: Topology GN11 | | | | | |
|---|---|---|---|---|---|---|---|
| | | Prior 1 | | Prior 2 | | Prior 3 | |
| $N$ | Edge | mean | sd | mean | sd | mean | sd |
| 100 | 1 | 0.0023 | 0.0035 | 0.003 | 0.0038 | 0.0046 | 0.006 |
| | 2 | 0.2976 | 0.1311 | 0.2084 | 0.1464 | 0.0386 | 0.1027 |
| | 3 | 0.2931 | 0.0658 | 0.2039 | 0.0851 | 0.0168 | 0.0239 |
| | 4 | 0.0018 | 0.0028 | 0.0035 | 0.0033 | 0.009 | 0.0104 |
| | 5 | 0.0029 | 0.0046 | 0.0037 | 0.0048 | 0.0087 | 0.0116 |
| | 6 | 0.0045 | 0.007 | 0.0043 | 0.0064 | 0.0038 | 0.0044 |
| | 7 | 0 | 0 | 0 | 0 | 0 | 0 |
| | 8 | 0 | 0 | 0 | 0 | 0 | 0 |
| | 9 | 0.0009 | 0.0011 | 0.0014 | 0.0023 | 0.0037 | 0.005 |
| | 10 | 0.0018 | 0.002 | 0.002 | 0.0025 | 0.0107 | 0.0153 |
| | 11 | 0.002 | 0.0019 | 0.0024 | 0.0029 | 0.0097 | 0.0132 |
| | 12 | 0.0016 | 0.0018 | 0.0016 | 0.0026 | 0.0044 | 0.0057 |
| 200 | 1 | 0.0048 | 0.0046 | 0.0033 | 0.0042 | 0.0029 | 0.0036 |
| | 2 | 0.286 | 0.1233 | 0.1938 | 0.1334 | 0.0338 | 0.0753 |
| | 3 | 0.3161 | 0.0676 | 0.2138 | 0.0676 | 0.0188 | 0.0152 |
| | 4 | 0.0032 | 0.0052 | 0.0028 | 0.0038 | 0.0076 | 0.0092 |
| | 5 | 0.0026 | 0.0043 | 0.0029 | 0.006 | 0.0069 | 0.0085 |
| | 6 | 0.0041 | 0.0056 | 0.0032 | 0.0056 | 0.0039 | 0.0052 |
| | 7 | 0 | 0 | 0 | 0 | 0 | 0 |
| | 8 | 0 | 0 | 0 | 0 | 0 | 0 |
| | 9 | 0.0006 | 0.0009 | 0.0004 | 0.0005 | 0.008 | 0.0137 |
| | 10 | 0.0013 | 0.0016 | 0.0007 | 0.0013 | 0.0115 | 0.0148 |
| | 11 | 0.0018 | 0.0021 | 0.0014 | 0.002 | 0.0093 | 0.0112 |
| | 12 | 0.0021 | 0.0025 | 0.0012 | 0.0018 | 0.0038 | 0.0041 |
| 600 | 1 | 0.0069 | 0.0069 | 0.0052 | 0.0058 | 0.0048 | 0.0056 |
| | 2 | 0.2735 | 0.0938 | 0.1783 | 0.0949 | 0.0147 | 0.0231 |
| | 3 | 0.3436 | 0.1118 | 0.2446 | 0.129 | 0.0557 | 0.1172 |
| | 4 | 0.0049 | 0.0062 | 0.0028 | 0.003 | 0.0108 | 0.0147 |
| | 5 | 0.0024 | 0.0029 | 0.0017 | 0.0021 | 0.0077 | 0.0112 |
| | 6 | 0.0027 | 0.003 | 0.0015 | 0.0015 | 0.0024 | 0.0023 |
| | 7 | 0 | 0 | 0 | 0 | 0 | 0 |
| | 8 | 0 | 0 | 0 | 0 | 0 | 0 |
| | 9 | 0.0006 | 0.0008 | 0.0009 | 0.0013 | 0.0031 | 0.0036 |
| | 10 | 0.0015 | 0.0018 | 0.0023 | 0.0033 | 0.0058 | 0.0069 |
| | 11 | 0.0017 | 0.0021 | 0.0029 | 0.0047 | 0.0066 | 0.0087 |
| | 12 | 0.0026 | 0.005 | 0.0022 | 0.0041 | 0.0046 | 0.0047 |

Table 2.7: **Mean and standard deviation of $MSE_2$ when the true skeleton was used as input.**

| | | | $MSE_2$ | | | | | |
|---|---|---|---|---|---|---|---|---|
| | | | GN4 | | GN8 | | GN11 | |
| Method | $N$ | $\beta$ | mean | sd | mean | sd | mean | sd |
| baycn | 100 | 0.2 | 0.2305 | 0.0351 | 0.2707 | 0.0294 | 0.1743 | 0.0259 |
| | | 0.5 | 0.0987 | 0.0607 | 0.0951 | 0.0401 | 0.0514 | 0.0322 |
| | | 1 | 0.015 | 0.0414 | 0.0286 | 0.0549 | 0.0064 | 0.0047 |
| Gibbs | 100 | 0.2 | 0.2034 | 0.0292 | 0.2483 | 0.0276 | 0.1504 | 0.0215 |
| | | 0.5 | 0.1009 | 0.0286 | 0.0873 | 0.0242 | 0.0439 | 0.03 |
| | | 1 | 0.0221 | 0.0357 | 0.0313 | 0.0518 | 0.0005 | 0.0003 |
| $MC^3$ | 100 | 0.2 | 0.2039 | 0.0279 | 0.2488 | 0.0283 | 0.1508 | 0.0202 |
| | | 0.5 | 0.1022 | 0.0325 | 0.0888 | 0.027 | 0.0438 | 0.0299 |
| | | 1 | 0.1913 | 0.2189 | 0.1573 | 0.124 | 0.0003 | 0.0002 |
| order | 100 | 0.2 | 0.3144 | 0.0563 | 0.368 | 0.0375 | 0.271 | 0.0336 |
| | | 0.5 | 0.1187 | 0.0436 | 0.0997 | 0.0384 | 0.0657 | 0.0374 |
| | | 1 | 0.0146 | 0.0195 | 0.0309 | 0.0432 | 0.0069 | 0.0023 |
| partition | 100 | 0.2 | 0.2505 | 0.0423 | 0.316 | 0.025 | 0.2067 | 0.0247 |
| | | 0.5 | 0.1125 | 0.0388 | 0.1202 | 0.0478 | 0.0656 | 0.0329 |
| | | 1 | 0.0213 | 0.023 | 0.0393 | 0.05 | 0.0291 | 0.0223 |
| baycn | 200 | 0.2 | 0.1796 | 0.0484 | 0.2129 | 0.0372 | 0.1308 | 0.0269 |
| | | 0.5 | 0.1132 | 0.0958 | 0.1302 | 0.0792 | 0.0182 | 0.0243 |
| | | 1 | 0.0069 | 0.0214 | 0.0649 | 0.1041 | 0.0098 | 0.0062 |
| Gibbs | 200 | 0.2 | 0.1716 | 0.0487 | 0.2071 | 0.0345 | 0.1239 | 0.0248 |
| | | 0.5 | 0.106 | 0.0683 | 0.0871 | 0.031 | 0.0134 | 0.0269 |
| | | 1 | 0.0015 | 0.0052 | 0.0005 | 0.0018 | 0.0004 | 0.0004 |
| $MC^3$ | 200 | 0.2 | 0.1725 | 0.0472 | 0.2071 | 0.035 | 0.1237 | 0.0252 |
| | | 0.5 | 0.1152 | 0.0966 | 0.1083 | 0.0626 | 0.0133 | 0.0268 |
| | | 1 | 0.2132 | 0.2264 | 0.2105 | 0.1217 | 0.0003 | 0.0002 |
| order | 200 | 0.2 | 0.2743 | 0.0793 | 0.2993 | 0.053 | 0.2052 | 0.0459 |
| | | 0.5 | 0.1089 | 0.072 | 0.0857 | 0.032 | 0.0322 | 0.0327 |
| | | 1 | 0.0067 | 0.0182 | 0.0051 | 0.0024 | 0.0087 | 0.004 |
| partition | 200 | 0.2 | 0.2035 | 0.045 | 0.2605 | 0.0325 | 0.1653 | 0.0327 |
| | | 0.5 | 0.1034 | 0.0628 | 0.1136 | 0.0561 | 0.0426 | 0.0259 |
| | | 1 | 0.0075 | 0.0133 | 0.013 | 0.0171 | 0.0327 | 0.0216 |
| baycn | 600 | 0.2 | 0.1235 | 0.0372 | 0.1721 | 0.0813 | 0.0908 | 0.0288 |
| | | 0.5 | 0.0755 | 0.0903 | 0.1005 | 0.0992 | 0.0071 | 0.0064 |
| | | 1 | 0.0104 | 0.0273 | 0.0301 | 0.0686 | 0.0068 | 0.0059 |
| Gibbs | 600 | 0.2 | 0.125 | 0.0272 | 0.1674 | 0.0724 | 0.0864 | 0.0276 |
| | | 0.5 | 0.0893 | 0.0835 | 0.0651 | 0.0431 | 0.0006 | 0.0012 |
| | | 1 | 0 | 0 | 0 | 0 | 0.0005 | 0.0004 |
| $MC^3$ | 600 | 0.2 | 0.1241 | 0.0282 | 0.1683 | 0.0741 | 0.0868 | 0.0274 |
| | | 0.5 | 0.229 | 0.2196 | 0.1316 | 0.1398 | 0.0005 | 0.0013 |
| | | 1 | 0.2311 | 0.2266 | 0.2232 | 0.1151 | 0.0003 | 0.0002 |
| order | 600 | 0.2 | 0.1351 | 0.0397 | 0.1648 | 0.069 | 0.1055 | 0.0382 |
| | | 0.5 | 0.0891 | 0.0898 | 0.0518 | 0.0433 | 0.0146 | 0.0158 |
| | | 1 | 0.0018 | 0.0006 | 0.0038 | 0.0005 | 0.0109 | 0.0052 |
| partition | 600 | 0.2 | 0.1257 | 0.028 | 0.1665 | 0.0615 | 0.1 | 0.0305 |
| | | 0.5 | 0.091 | 0.0803 | 0.0736 | 0.0792 | 0.0394 | 0.02 |
| | | 1 | 0.001 | 0.0015 | 0.0059 | 0.0146 | 0.0477 | 0.0389 |

Table 2.8: **The mean and standard deviation of precision, power, and $MSE_2$ for topology GN4.** A fully connected graph was the input to each algorithm. When calculating precision and power we considered an edge present if the sum of the posterior probabilities of the two directions was greater than 0.5.

| | | | Precision | | Power | | $MSE_2$ | |
|---|---|---|---|---|---|---|---|---|
| Method | $N$ | $\beta$ | mean | sd | mean | sd | mean | sd |
| baycn | 100 | 0.2 | 0.94 | 0.2077 | 0.43 | 0.2654 | 0.2398 | 0.0326 |
| | | 0.5 | 0.982 | 0.0627 | 0.98 | 0.0692 | 0.1235 | 0.0478 |
| | | 1 | 0.9707 | 0.0841 | 1 | 0 | 0.0439 | 0.0387 |
| Gibbs | 100 | 0.2 | 0.942 | 0.206 | 0.5 | 0.2602 | 0.2163 | 0.0303 |
| | | 0.5 | 0.984 | 0.0554 | 1 | 0 | 0.1212 | 0.0299 |
| | | 1 | 0.9093 | 0.1184 | 1 | 0 | 0.0938 | 0.0592 |
| $MC^3$ | 100 | 0.2 | 0.942 | 0.206 | 0.49 | 0.265 | 0.2155 | 0.0312 |
| | | 0.5 | 0.976 | 0.0663 | 1 | 0 | 0.1215 | 0.0329 |
| | | 1 | 0.9093 | 0.1184 | 1 | 0 | 0.0945 | 0.0683 |
| order | 100 | 0.2 | 0.80 | 0.4082 | 0.29 | 0.2126 | 0.315 | 0.0562 |
| | | 0.5 | 0.99 | 0.05 | 0.96 | 0.0935 | 0.1197 | 0.0502 |
| | | 1 | 0.992 | 0.04 | 1 | 0 | 0.0193 | 0.0287 |
| partition | 100 | 0.2 | 0.96 | 0.2 | 0.36 | 0.2051 | 0.2546 | 0.04 |
| | | 0.5 | 0.982 | 0.0627 | 0.98 | 0.0692 | 0.1118 | 0.0357 |
| | | 1 | 0.976 | 0.0663 | 1 | 0 | 0.0287 | 0.0311 |
| scanBMA | 100 | 0.2 | 0.88 | 0.3317 | 0.35 | 0.25 | - | - |
| | | 0.5 | 0.948 | 0.0952 | 0.97 | 0.0829 | - | - |
| | | 1 | 0.7927 | 0.0281 | 0.99 | 0.05 | - | - |
| baycn | 200 | 0.2 | 0.99 | 0.05 | 0.74 | 0.2222 | 0.1812 | 0.0427 |
| | | 0.5 | 0.992 | 0.04 | 1 | 0 | 0.101 | 0.0769 |
| | | 1 | 0.984 | 0.0554 | 1 | 0 | 0.0307 | 0.0427 |
| Gibbs | 200 | 0.2 | 0.98 | 0.0692 | 0.77 | 0.2155 | 0.1763 | 0.0383 |
| | | 0.5 | 0.992 | 0.04 | 1 | 0 | 0.1134 | 0.0555 |
| | | 1 | 0.952 | 0.0872 | 1 | 0 | 0.0521 | 0.0476 |
| $MC^3$ | 200 | 0.2 | 0.98 | 0.0692 | 0.77 | 0.2155 | 0.1765 | 0.0376 |
| | | 0.5 | 0.992 | 0.04 | 1 | 0 | 0.1189 | 0.0619 |
| | | 1 | 0.96 | 0.0816 | 1 | 0 | 0.0525 | 0.0452 |
| order | 200 | 0.2 | 0.9067 | 0.2809 | 0.48 | 0.2385 | 0.2768 | 0.0782 |
| | | 0.5 | 1 | 0 | 1 | 0 | 0.1072 | 0.0721 |
| | | 1 | 0.992 | 0.04 | 1 | 0 | 0.011 | 0.0349 |
| partition | 200 | 0.2 | 0.9867 | 0.0667 | 0.59 | 0.2026 | 0.2087 | 0.0448 |
| | | 0.5 | 0.992 | 0.04 | 1 | 0 | 0.1043 | 0.0646 |
| | | 1 | 0.992 | 0.04 | 1 | 0 | 0.012 | 0.0261 |
| scanBMA | 200 | 0.2 | 0.9787 | 0.0763 | 0.67 | 0.225 | - | - |
| | | 0.5 | 0.8747 | 0.1077 | 1 | 0 | - | - |
| | | 1 | 0.8 | 0 | 1 | 0 | - | - |
| baycn | 600 | 0.2 | 0.992 | 0.04 | 1 | 0 | 0.133 | 0.0358 |
| | | 0.5 | 0.992 | 0.04 | 1 | 0 | 0.0954 | 0.096 |
| | | 1 | 0.992 | 0.04 | 1 | 0 | 0.0125 | 0.0102 |

| | | | Precision | | Power | | $MSE_2$ | |
|---|---|---|---|---|---|---|---|---|
| Method | $N$ | $\beta$ | mean | sd | mean | sd | mean | sd |
| Gibbs | 600 | 0.2 | 0.992 | 0.04 | 0.99 | 0.05 | 0.1274 | 0.0219 |
| | | 0.5 | 0.992 | 0.04 | 1 | 0 | 0.0982 | 0.0741 |
| | | 1 | 0.992 | 0.04 | 1 | 0 | 0.0163 | 0.0156 |
| $MC^3$ | 600 | 0.2 | 0.992 | 0.04 | 0.99 | 0.05 | 0.131 | 0.0225 |
| | | 0.5 | 0.992 | 0.04 | 1 | 0 | 0.1098 | 0.093 |
| | | 1 | 1 | 0 | 1 | 0 | 0.0153 | 0.0124 |
| order | 600 | 0.2 | 1 | 0 | 0.96 | 0.1181 | 0.1395 | 0.0391 |
| | | 0.5 | 1 | 0 | 1 | 0 | 0.0909 | 0.0903 |
| | | 1 | 1 | 0 | 1 | 0 | 0.002 | 0.0008 |
| partition | 600 | 0.2 | 1 | 0 | 0.97 | 0.1099 | 0.1288 | 0.0272 |
| | | 0.5 | 1 | 0 | 1 | 0 | 0.0869 | 0.077 |
| | | 1 | 1 | 0 | 1 | 0 | 0.0016 | 0.0018 |
| scanBMA | 600 | 0.2 | 0.976 | 0.0663 | 0.99 | 0.05 | - | - |
| | | 0.5 | 0.8 | 0 | 1 | 0 | - | - |
| | | 1 | 0.8 | 0 | 1 | 0 | - | - |

Table 2.9: **Posterior probability from the first 7 data sets of GN4 with $N = 100$ and $\beta = 0.2$.** Rows in gray are the false edges, edge numbers with -v are the edges that make a v structure, and edge numbers in bold are the edges where baycn inferred a true edge as present but partition MCMC did not.

| | | Posterior Probability | | | | | |
|---|---|---|---|---|---|---|---|
| | | baycn | | | partition | | |
| Data set | Edge | zero | one | two | zero | one | two |
| 1 | 1 | 0.08 | 0.045 | 0.875 | 0.032 | 0.04 | 0.928 |
| | 2-v | 0.12 | 0.085 | 0.795 | 0.045 | 0.047 | 0.908 |
| | 3 | 0.025 | 0.055 | 0.92 | 0.03 | 0.026 | 0.944 |
| | 4 | 0.08 | 0.045 | 0.875 | 0.044 | 0.029 | 0.928 |
| | 5 | 0.265 | 0.21 | 0.525 | 0.173 | 0.117 | 0.709 |
| | 6-v | 0.48 | 0.515 | 0.005 | 0.536 | 0.463 | 0.001 |
| 2 | 1 | 0.435 | 0.405 | 0.16 | 0.405 | 0.358 | 0.237 |
| | 2-v | 0.045 | 0.075 | 0.88 | 0.025 | 0.027 | 0.948 |
| | 3 | 0.035 | 0.1 | 0.865 | 0.025 | 0.04 | 0.935 |
| | 4 | 0.245 | 0.32 | 0.435 | 0.209 | 0.289 | 0.501 |
| | **5** | 0.2 | 0.36 | 0.44 | 0.145 | 0.244 | 0.611 |
| | 6-v | 0.505 | 0.44 | 0.055 | 0.516 | 0.434 | 0.05 |
| 3 | 1 | 0.515 | 0.385 | 0.1 | 0.473 | 0.349 | 0.178 |
| | **2-v** | 0.285 | 0.285 | 0.43 | 0.177 | 0.226 | 0.597 |
| | 3 | 0.19 | 0.34 | 0.47 | 0.161 | 0.213 | 0.626 |
| | 4 | 0.085 | 0.045 | 0.87 | 0.039 | 0.037 | 0.924 |
| | 5 | 0.25 | 0.435 | 0.315 | 0.204 | 0.363 | 0.433 |
| | 6-v | 0.15 | 0.25 | 0.6 | 0.126 | 0.133 | 0.741 |
| 4 | 1 | 0.14 | 0.105 | 0.755 | 0.101 | 0.097 | 0.802 |
| | 2-v | 0.185 | 0.19 | 0.625 | 0.126 | 0.132 | 0.742 |
| | 3 | 0.055 | 0.095 | 0.85 | 0.051 | 0.081 | 0.868 |
| | 4 | 0.085 | 0.06 | 0.855 | 0.056 | 0.047 | 0.897 |
| | **5** | 0.455 | 0.54 | 0.005 | 0.511 | 0.488 | 0.001 |
| | 6-v | 0.185 | 0.18 | 0.635 | 0.105 | 0.121 | 0.774 |
| 5 | 1 | 0.16 | 0.175 | 0.665 | 0.142 | 0.152 | 0.706 |
| | 2-v | 0.175 | 0.115 | 0.71 | 0.076 | 0.06 | 0.864 |
| | 3 | 0.09 | 0.09 | 0.82 | 0.047 | 0.031 | 0.921 |
| | 4 | 0.06 | 0.095 | 0.845 | 0.025 | 0.026 | 0.949 |
| | 5 | 0.075 | 0.06 | 0.865 | 0.036 | 0.052 | 0.911 |
| | 6-v | 0.485 | 0.46 | 0.055 | 0.499 | 0.464 | 0.037 |
| 6 | 1 | 0.165 | 0.25 | 0.585 | 0.153 | 0.132 | 0.714 |
| | 2-v | 0.52 | 0.475 | 0.005 | 0.469 | 0.491 | 0.04 |
| | 3 | 0.045 | 0.08 | 0.875 | 0.032 | 0.04 | 0.928 |
| | 4 | 0.03 | 0.025 | 0.945 | 0.03 | 0.029 | 0.941 |
| | 5 | 0.1 | 0.125 | 0.775 | 0.066 | 0.057 | 0.877 |
| | 6-v | 0.06 | 0.035 | 0.905 | 0.036 | 0.022 | 0.941 |
| 7 | 1 | 0.48 | 0.43 | 0.09 | 0.416 | 0.383 | 0.201 |
| | 2-v | 0.065 | 0.085 | 0.85 | 0.05 | 0.045 | 0.905 |
| | 3 | 0.05 | 0.05 | 0.9 | 0.039 | 0.06 | 0.901 |
| | 4 | 0.11 | 0.04 | 0.85 | 0.059 | 0.024 | 0.918 |
| | 5 | 0.145 | 0.18 | 0.675 | 0.097 | 0.117 | 0.786 |
| | 6-v | 0.16 | 0.2 | 0.64 | 0.13 | 0.116 | 0.754 |

Table 2.10: **Posterior probability from the first 7 data sets of GN4 with $N = 200$ and $\beta = 0.2$.** Rows in gray are the false edges, edge numbers with -v are the edges that make a v structure, and edge numbers in bold are the edges where baycn inferred a true edge as present but partition MCMC did not.

| | | Posterior Probability | | | | | |
|---|---|---|---|---|---|---|---|
| | | baycn | | | partition | | |
| Data set | Edge | zero | one | two | zero | one | two |
| 1 | 1 | 0.415 | 0.38 | 0.205 | 0.389 | 0.257 | 0.354 |
| | 2-v | 0.425 | 0.515 | 0.06 | 0.419 | 0.459 | 0.122 |
| | 3 | 0.095 | 0.045 | 0.86 | 0.037 | 0.032 | 0.93 |
| | 4 | 0.075 | 0.09 | 0.835 | 0.037 | 0.054 | 0.909 |
| | 5 | 0.14 | 0.115 | 0.745 | 0.056 | 0.091 | 0.853 |
| | 6-v | 0.605 | 0.395 | 0 | 0.599 | 0.392 | 0.01 |
| 2 | 1 | 0.33 | 0.67 | 0 | 0.272 | 0.728 | 0 |
| | 2-v | 0.335 | 0.645 | 0.02 | 0.231 | 0.692 | 0.077 |
| | 3 | 0.03 | 0.075 | 0.895 | 0.015 | 0.031 | 0.954 |
| | 4 | 0.23 | 0.16 | 0.61 | 0.121 | 0.079 | 0.8 |
| | **5** | 0.37 | 0.34 | 0.29 | 0.204 | 0.172 | 0.623 |
| | 6-v | 0.525 | 0.475 | 0 | 0.475 | 0.509 | 0.016 |
| 3 | 1 | 0.37 | 0.385 | 0.245 | 0.298 | 0.294 | 0.408 |
| | **2-v** | 0.32 | 0.285 | 0.395 | 0.218 | 0.19 | 0.592 |
| | 3 | 0.025 | 0.07 | 0.905 | 0.02 | 0.016 | 0.964 |
| | 4 | 0.13 | 0.085 | 0.785 | 0.072 | 0.065 | 0.863 |
| | 5 | 0.335 | 0.535 | 0.13 | 0.349 | 0.43 | 0.221 |
| | 6-v | 0.32 | 0.44 | 0.24 | 0.253 | 0.286 | 0.461 |
| 4 | 1 | 0.11 | 0.13 | 0.76 | 0.07 | 0.051 | 0.879 |
| | **2-v** | 0.415 | 0.29 | 0.295 | 0.3 | 0.181 | 0.519 |
| | 3 | 0.135 | 0.07 | 0.795 | 0.052 | 0.046 | 0.901 |
| | 4 | 0.115 | 0.19 | 0.695 | 0.076 | 0.125 | 0.799 |
| | 5 | 0.235 | 0.25 | 0.515 | 0.141 | 0.188 | 0.671 |
| | 6-v | 0.385 | 0.525 | 0.09 | 0.378 | 0.461 | 0.161 |
| 5 | 1 | 0.51 | 0.48 | 0.01 | 0.498 | 0.47 | 0.032 |
| | 2-v | 0.41 | 0.4 | 0.19 | 0.383 | 0.312 | 0.305 |
| | 3 | 0.035 | 0.045 | 0.92 | 0.019 | 0.021 | 0.96 |
| | 4 | 0.045 | 0.035 | 0.92 | 0.016 | 0.021 | 0.963 |
| | 5 | 0.505 | 0.485 | 0.01 | 0.484 | 0.494 | 0.022 |
| | 6-v | 0.335 | 0.385 | 0.28 | 0.231 | 0.304 | 0.465 |
| 6 | 1 | 0.225 | 0.17 | 0.605 | 0.146 | 0.103 | 0.751 |
| | 2-v | 0.56 | 0.44 | 0 | 0.591 | 0.403 | 0.006 |
| | 3 | 0.4 | 0.48 | 0.12 | 0.47 | 0.367 | 0.163 |
| | 4 | 0.03 | 0.05 | 0.92 | 0.021 | 0.02 | 0.959 |
| | 5 | 0.34 | 0.605 | 0.055 | 0.31 | 0.562 | 0.127 |
| | **6-v** | 0.285 | 0.295 | 0.42 | 0.17 | 0.19 | 0.641 |
| 7 | **1** | 0.335 | 0.36 | 0.305 | 0.248 | 0.233 | 0.519 |
| | **2-v** | 0.27 | 0.235 | 0.495 | 0.166 | 0.157 | 0.677 |
| | 3 | 0.075 | 0.035 | 0.89 | 0.02 | 0.019 | 0.961 |
| | 4 | 0.105 | 0.12 | 0.775 | 0.059 | 0.082 | 0.859 |
| | 5 | 0.325 | 0.46 | 0.215 | 0.234 | 0.406 | 0.359 |
| | 6-v | 0.425 | 0.575 | 0 | 0.439 | 0.546 | 0.015 |

Table 2.11: **Posterior probability from the first 7 data sets of GN4 with $N = 600$ and $\beta = 0.2$.** Rows in gray are the false edges, edge numbers with -v are the edges that make a v structure, and edge numbers in bold are the edges where baycn inferred a true edge as present but partition MCMC did not.

| | | Posterior Probability | | | | | |
|---|---|---|---|---|---|---|---|
| | | baycn | | | partition | | |
| Data set | Edge | zero | one | two | zero | one | two |
| 1 | 1 | 0.61 | 0.39 | 0 | 0.511 | 0.489 | 0 |
| | **2-v** | 0.345 | 0.325 | 0.33 | 0.224 | 0.148 | 0.627 |
| | 3 | 0.065 | 0.03 | 0.905 | 0.02 | 0.012 | 0.968 |
| | 4 | 0.065 | 0.045 | 0.89 | 0.016 | 0.016 | 0.968 |
| | 5 | 0.405 | 0.595 | 0 | 0.491 | 0.509 | 0 |
| | **6-v** | 0.29 | 0.26 | 0.45 | 0.091 | 0.142 | 0.767 |
| 2 | 1 | 0.61 | 0.39 | 0 | 0.632 | 0.368 | 0 |
| | 2-v | 0.585 | 0.415 | 0 | 0.641 | 0.359 | 0 |
| | 3 | 0.025 | 0.055 | 0.92 | 0.009 | 0.014 | 0.978 |
| | 4 | 0.05 | 0.05 | 0.9 | 0.017 | 0.014 | 0.969 |
| | 5 | 0.36 | 0.64 | 0 | 0.424 | 0.576 | 0 |
| | 6-v | 0.445 | 0.555 | 0 | 0.401 | 0.589 | 0.01 |
| 3 | 1 | 0.63 | 0.37 | 0 | 0.615 | 0.385 | 0 |
| | 2-v | 0.465 | 0.465 | 0.07 | 0.379 | 0.384 | 0.237 |
| | 3 | 0.13 | 0.095 | 0.775 | 0.037 | 0.034 | 0.929 |
| | 4 | 0.095 | 0.15 | 0.755 | 0.019 | 0.03 | 0.951 |
| | 5 | 0.44 | 0.56 | 0 | 0.426 | 0.574 | 0 |
| | 6-v | 0.645 | 0.355 | 0 | 0.54 | 0.46 | 0 |
| 4 | 1 | 0.57 | 0.43 | 0 | 0.469 | 0.53 | 0.001 |
| | 2-v | 0.555 | 0.39 | 0.055 | 0.368 | 0.431 | 0.201 |
| | 3 | 0.065 | 0.065 | 0.87 | 0.007 | 0.01 | 0.983 |
| | 4 | 0.085 | 0.055 | 0.86 | 0.007 | 0.01 | 0.983 |
| | 5 | 0.63 | 0.37 | 0 | 0.505 | 0.495 | 0 |
| | 6-v | 0.45 | 0.55 | 0 | 0.495 | 0.505 | 0 |
| 5 | 1 | 0.545 | 0.455 | 0 | 0.586 | 0.414 | 0 |
| | 2-v | 0.515 | 0.485 | 0 | 0.602 | 0.392 | 0.006 |
| | 3 | 0.025 | 0.045 | 0.93 | 0.007 | 0.006 | 0.986 |
| | 4 | 0.065 | 0.04 | 0.895 | 0.012 | 0.01 | 0.978 |
| | 5 | 0.525 | 0.475 | 0 | 0.439 | 0.559 | 0.002 |
| | 6-v | 0.355 | 0.645 | 0 | 0.425 | 0.575 | 0 |
| 6 | 1 | 0.765 | 0.23 | 0.005 | 0.758 | 0.233 | 0.009 |
| | 2-v | 0.645 | 0.355 | 0 | 0.682 | 0.314 | 0.004 |
| | 3 | 0.15 | 0.07 | 0.78 | 0.02 | 0.015 | 0.965 |
| | 4 | 0.06 | 0.05 | 0.89 | 0.007 | 0.006 | 0.986 |
| | 5 | 0.33 | 0.67 | 0 | 0.232 | 0.768 | 0 |
| | 6-v | 0.375 | 0.625 | 0 | 0.283 | 0.711 | 0.006 |
| 7 | 1 | 0.425 | 0.575 | 0 | 0.46 | 0.54 | 0 |
| | 2-v | 0.4 | 0.6 | 0 | 0.5 | 0.5 | 0 |
| | 3 | 0.055 | 0.06 | 0.885 | 0.006 | 0.012 | 0.981 |
| | 4 | 0.06 | 0.06 | 0.88 | 0.012 | 0.017 | 0.97 |
| | 5 | 0.5 | 0.5 | 0 | 0.569 | 0.431 | 0 |
| | 6-v | 0.45 | 0.55 | 0 | 0.534 | 0.46 | 0.006 |

Table 2.12: **The mean runtime in seconds across 25 data sets.** For each topology 25 data sets were generated with $\beta = 1$ and $N = 600$ and each algorithm was run once per data set and the runtime in seconds was recorded. All algorithms were run on an Intel Xeon D-1540 2.00 GHz processor with 128 GB of memory.

| | Runtime | | | | |
|---|---|---|---|---|---|
| Topology | baycn | Gibbs | $MC^3$ | order | partition |
| GN4 | 4.49 | 235.00 | 11.94 | 1.78 | 3.93 |
| GN8 | 8.11 | 363.91 | 22.49 | 2.88 | 7.97 |
| GN11 | 7.07 | 380.97 | 23.52 | 3.10 | 9.39 |

Table 2.13: **Posterior probabilities from baycn on eQTL-gene sets from GEUVADIS and GTEx.** A fully connected graph was used as the input to baycn.

| | | Posterior Probability | | | | | |
|---|---|---|---|---|---|---|---|
| | | GEUVADIS | | | GTEx | | |
| Set | Edge | zero | one | two | zero | one | two |
| Q8 | rs11305802-TMEM55B | 0.210 | 0.000 | 0.79 | 0.250 | 0.00 | 0.750 |
| | rs11305802-RP11-203M5.8 | 0.410 | 0.000 | 0.59 | 0.360 | 0.00 | 0.640 |
| | rs11305802-PNP | 1.000 | 0.000 | 0.00 | 0.990 | 0.00 | 0.010 |
| | TMEM55B-RP11-203M5.8 | 0.525 | 0.465 | 0.01 | 0.330 | 0.58 | 0.090 |
| | TMEM55B-PNP | 0.155 | 0.845 | 0.00 | 0.145 | 0.41 | 0.445 |
| | RP11-203M5.8-PNP | 0.185 | 0.815 | 0.00 | 0.180 | 0.71 | 0.110 |
| Q20 | rs142060986-RP11292F9.1 | 0 | 0 | 1 | 0.00 | 0.00 | 1.00 |
| | rs142060986-FAM27E1 | 1 | 0 | 0 | 0.95 | 0.00 | 0.05 |
| | RP11292F9.1-FAM27E1 | 0 | 1 | 0 | 0.17 | 0.47 | 0.36 |
| Q21 | rs147156488-FAM27C | 0.450 | 0.000 | 0.550 | 0.000 | 0.000 | 1.000 |
| | rs147156488-FAM27A | 0.000 | 0.000 | 1.000 | 0.000 | 0.000 | 1.000 |
| | rs147156488-FAM27D1 | 0.550 | 0.000 | 0.450 | 1.000 | 0.000 | 0.000 |
| | FAM27C-FAM27A | 0.525 | 0.010 | 0.465 | 0.055 | 0.040 | 0.905 |
| | FAM27C-FAM27D1 | 0.450 | 0.550 | 0.000 | 0.055 | 0.035 | 0.910 |
| | FAM27A-FAM27D1 | 0.015 | 0.495 | 0.490 | 0.100 | 0.370 | 0.530 |
| Q23 | rs150605045-AGAP9 | 1.0 | 0.0 | 0 | 0.015 | 0.00 | 0.985 |
| | rs150605045-AGAP10 | 1.0 | 0.0 | 0 | 0.000 | 0.00 | 1.000 |
| | AGAP9-AGAP10 | 0.4 | 0.6 | 0 | 0.180 | 0.16 | 0.660 |
| Q37 | rs3858954-GOLGA6L19 | 0.660 | 0.000 | 0.34 | 0.020 | 0.00 | 0.980 |
| | rs3858954-GOLGA6L9 | 0.310 | 0.000 | 0.69 | 0.095 | 0.00 | 0.905 |
| | rs3858954-GOLGA6L20 | 0.960 | 0.000 | 0.04 | 0.205 | 0.00 | 0.795 |
| | GOLGA6L19-GOLGA6L9 | 0.030 | 0.040 | 0.93 | 0.045 | 0.07 | 0.885 |
| | GOLGA6L19-GOLGA6L20 | 0.345 | 0.655 | 0.00 | 0.055 | 0.05 | 0.895 |
| | GOLGA6L9-GOLGA6L20 | 0.145 | 0.855 | 0.00 | 0.065 | 0.07 | 0.865 |
| Q50 | rs7124238-SBF2-AS1 | 0.22 | 0.00 | 0.78 | 0.18 | 0.00 | 0.82 |
| | rs7124238-SWAP70 | 1.00 | 0.00 | 0.00 | 1.00 | 0.00 | 0.00 |
| | SBF2-AS1-SWAP70 | 0.12 | 0.88 | 0.00 | 0.17 | 0.83 | 0.00 |
| Q62 | rs9426902-S100A6 | 0.845 | 0.00 | 0.155 | 0.095 | 0.00 | 0.905 |
| | rs9426902-S100A4 | 0.420 | 0.00 | 0.580 | 0.090 | 0.00 | 0.910 |
| | S100A6-S100A4 | 0.720 | 0.28 | 0.000 | 0.530 | 0.47 | 0.000 |

Table 2.14: **Posterior probabilities from baycn for eQTL-gene set Q8 from GEUVADIS with five PCs included in the network as confounding variables.** A fully connected graph (excluding the edges between PC nodes) was used as the input to baycn. The rows highlighted in gray indicate the edges between the nodes of interest.

| | Posterior Probability | | |
|---|---|---|---|
| Edge | zero | one | two |
| rs11305802-TMEM55B | 0.170 | 0.000 | 0.830 |
| rs11305802-RP11-203M5.8 | 0.805 | 0.000 | 0.195 |
| rs11305802-PNP | 1.000 | 0.000 | 0.000 |
| rs11305802-PC1 | 0.240 | 0.000 | 0.760 |
| rs11305802-PC2 | 0.195 | 0.000 | 0.805 |
| rs11305802-PC6 | 0.255 | 0.000 | 0.745 |
| rs11305802-PC7 | 0.125 | 0.000 | 0.875 |
| rs11305802-PC9 | 0.370 | 0.000 | 0.630 |
| TMEM55B-RP11-203M5.8 | 0.150 | 0.820 | 0.030 |
| TMEM55B-PNP | 0.330 | 0.670 | 0.000 |
| TMEM55B-PC1 | 0.385 | 0.615 | 0.000 |
| TMEM55B-PC2 | 0.085 | 0.550 | 0.365 |
| TMEM55B-PC6 | 0.015 | 0.055 | 0.930 |
| TMEM55B-PC7 | 0.050 | 0.030 | 0.920 |
| TMEM55B-PC9 | 0.260 | 0.730 | 0.010 |
| RP11-203M5.8-PNP | 0.760 | 0.240 | 0.000 |
| RP11-203M5.8-PC1 | 0.075 | 0.145 | 0.780 |
| RP11-203M5.8-PC2 | 0.085 | 0.240 | 0.675 |
| RP11-203M5.8-PC6 | 0.315 | 0.685 | 0.000 |
| RP11-203M5.8-PC7 | 0.175 | 0.115 | 0.710 |
| RP11-203M5.8-PC9 | 0.035 | 0.060 | 0.905 |
| PNP-PC1 | 0.190 | 0.715 | 0.095 |
| PNP-PC2 | 0.135 | 0.845 | 0.020 |
| PNP-PC6 | 0.040 | 0.065 | 0.895 |
| PNP-PC7 | 0.565 | 0.180 | 0.255 |
| PNP-PC9 | 0.165 | 0.695 | 0.140 |

Table 2.15: **Posterior probabilities from baycn for eQTL-gene set Q21 from GEUVADIS with two PCs included in the network as confounding variables.** A fully connected graph was used as the input to baycn. The rows highlighted in gray indicate the edges between the nodes of interest.

| | Posterior Probability | | |
|---|---|---|---|
| Edge | zero | one | two |
| rs147156488-FAM27C | 0.000 | 0.000 | 1.000 |
| rs147156488-FAM27A | 0.000 | 0.000 | 1.000 |
| rs147156488-FAM27D1 | 1.000 | 0.000 | 0.000 |
| rs147156488-PC1 | 1.000 | 0.000 | 0.000 |
| rs147156488-PC3 | 1.000 | 0.000 | 0.000 |
| FAM27C-FAM27A | 0.510 | 0.030 | 0.460 |
| FAM27C-FAM27D1 | 0.000 | 1.000 | 0.000 |
| FAM27C-PC1 | 0.250 | 0.030 | 0.720 |
| FAM27C-PC3 | 0.180 | 0.140 | 0.680 |
| FAM27A-FAM27D1 | 0.000 | 0.510 | 0.490 |
| FAM27A-PC1 | 0.145 | 0.050 | 0.805 |
| FAM27A-PC3 | 0.555 | 0.115 | 0.330 |
| FAM27D1-PC1 | 0.245 | 0.220 | 0.535 |
| FAM27D1-PC3 | 0.160 | 0.125 | 0.715 |

Table 2.16: **Posterior probabilities from baycn for eQTL-gene set Q23 from GEUVADIS with two PCs included in the network as confounding variables.** A fully connected graph (excluding the edges between PC nodes) was used as the input to baycn. The rows highlighted in gray indicate the edges between the nodes of interest.

| | Posterior Probability | | |
|---|---|---|---|
| Edge | zero | one | two |
| rs150605045-AGAP9 | 1.000 | 0.000 | 0.000 |
| rs150605045-AGAP10 | 1.000 | 0.000 | 0.000 |
| rs150605045-PC1 | 1.000 | 0.000 | 0.000 |
| rs150605045-PC8 | 1.000 | 0.000 | 0.000 |
| AGAP9-AGAP10 | 0.720 | 0.280 | 0.000 |
| AGAP9-PC1 | 0.595 | 0.385 | 0.020 |
| AGAP9-PC8 | 0.495 | 0.455 | 0.050 |
| AGAP10-PC1 | 0.065 | 0.065 | 0.870 |
| AGAP10-PC8 | 0.035 | 0.080 | 0.885 |

Table 2.17: **Posterior probabilities from baycn for eQTL-gene set Q37 from GEUVADIS with one PC included in the network as a confounding variable.** A fully connected graph was used as the input to baycn. The rows highlighted in gray indicate the edges between the nodes of interest.

| | Posterior Probability | | |
|---|---|---|---|
| Edge | zero | one | two |
| rs3858954-GOLGA6L19 | 0.660 | 0.000 | 0.340 |
| rs3858954-GOLGA6L9 | 0.335 | 0.000 | 0.665 |
| rs3858954-GOLGA6L20 | 0.965 | 0.000 | 0.035 |
| rs3858954-PC1 | 0.100 | 0.000 | 0.900 |
| GOLGA6L19-GOLGA6L9 | 0.035 | 0.055 | 0.910 |
| GOLGA6L19-GOLGA6L20 | 0.335 | 0.665 | 0.000 |
| GOLGA6L19-PC1 | 0.665 | 0.210 | 0.125 |
| GOLGA6L9-GOLGA6L20 | 0.175 | 0.825 | 0.000 |
| GOLGA6L9-PC1 | 0.110 | 0.030 | 0.860 |
| GOLGA6L20-PC1 | 0.135 | 0.050 | 0.815 |

Table 2.18: **Posterior probabilities from baycn for eQTL-gene set Q50 from GEUVADIS with two PCs included in the network as confounding variables.** A fully connected graph (excluding the edges between PC nodes) was used as the input to baycn. The rows highlighted in gray indicate the edges between the nodes of interest.

| | Posterior Probability | | |
|---|---|---|---|
| Edge | zero | one | two |
| rs7124238-SBF2-AS1 | 0.195 | 0.000 | 0.805 |
| rs7124238-SWAP70 | 1.000 | 0.000 | 0.000 |
| rs7124238-PC1 | 0.135 | 0.000 | 0.865 |
| rs7124238-PC5 | 0.465 | 0.000 | 0.535 |
| SBF2-AS1-SWAP70 | 0.035 | 0.965 | 0.000 |
| SBF2-AS1-PC1 | 0.035 | 0.020 | 0.945 |
| SBF2-AS1-PC5 | 0.015 | 0.040 | 0.945 |
| SWAP70-PC1 | 0.550 | 0.415 | 0.035 |
| SWAP70-PC5 | 0.280 | 0.720 | 0.000 |

Table 2.19: **Posterior probabilities inferred by baycn for the Drosophila data set.** The graph output from MRPC was used as the input to baycn with the addition of the edges: VM-bin_6.8, VM-bin_10.12, Meso_SM-tin_4.6, Meso_SM-tin_6.8, Meso_SM-mef2_4.6, Meso_SM-mef2_6.8, Meso_SM-mef2_8.10, and Meso_SM-mef2_10.12.

| | Posterior Probability | | |
|---|---|---|---|
| Edge | zero | one | two |
| Meso-twi_2.4 | 0.282 | 0.644 | 0.074 |
| VM-bin_6.8 | 0.126 | 0.462 | 0.412 |
| VM-bin_8.10 | 0.172 | 0.474 | 0.354 |
| VM-bin_10.12 | 0.076 | 0.054 | 0.870 |
| SM-mef2_10.12 | 0.440 | 0.208 | 0.352 |
| CM-bin_10.12 | 0.420 | 0.196 | 0.384 |
| Meso_SM-tin_2.4 | 0.102 | 0.534 | 0.364 |
| Meso_SM-tin_4.6 | 0.024 | 0.164 | 0.812 |
| Meso_SM-tin_6.8 | 0.180 | 0.040 | 0.780 |
| Meso_SM-mef2_2.4 | 0.226 | 0.556 | 0.218 |
| Meso_SM-mef2_4.6 | 0.174 | 0.282 | 0.544 |
| Meso_SM-mef2_6.8 | 0.024 | 0.050 | 0.926 |
| Meso_SM-mef2_8.10 | 0.052 | 0.056 | 0.892 |
| Meso_SM-mef2_10.12 | 0.036 | 0.048 | 0.916 |
| VM_SM-bin_10.12 | 0.504 | 0.282 | 0.214 |
| VM_SM-twi_2.4 | 0.378 | 0.622 | 0.000 |
| VM_SM-mef2_10.12 | 0.738 | 0.262 | 0.000 |
| tin_2.4-tin_4.6 | 0.332 | 0.668 | 0.000 |
| tin_2.4-twi_4.6 | 0.526 | 0.464 | 0.010 |
| tin_4.6-twi_4.6 | 0.548 | 0.382 | 0.070 |
| tin_4.6-twi_6.8 | 0.522 | 0.476 | 0.002 |
| tin_4.6-mef2_2.4 | 0.932 | 0.066 | 0.002 |
| tin_4.6-mef2_4.6 | 0.972 | 0.028 | 0.000 |
| tin_6.8-bin_6.8 | 0.300 | 0.576 | 0.124 |
| tin_6.8-bap_6.8 | 0.494 | 0.506 | 0.000 |
| bin_6.8-bin_8.10 | 0.604 | 0.396 | 0.000 |
| bin_6.8-bin_10.12 | 0.494 | 0.408 | 0.098 |
| bin_6.8-bap_6.8 | 0.746 | 0.216 | 0.038 |
| bin_8.10-bin_10.12 | 0.500 | 0.500 | 0.000 |
| twi_2.4-twi_4.6 | 0.224 | 0.776 | 0.000 |
| twi_4.6-twi_6.8 | 0.400 | 0.596 | 0.004 |
| twi_4.6-mef2_4.6 | 0.992 | 0.008 | 0.000 |
| twi_6.8-mef2_6.8 | 0.890 | 0.110 | 0.000 |
| mef2_2.4-mef2_6.8 | 0.604 | 0.284 | 0.112 |
| mef2_2.4-mef2_10.12 | 0.310 | 0.610 | 0.080 |
| mef2_4.6-mef2_6.8 | 0.008 | 0.992 | 0.000 |
| mef2_6.8-mef2_8.10 | 0.624 | 0.376 | 0.000 |
| mef2_8.10-mef2_10.12 | 0.172 | 0.828 | 0.000 |

Table 2.20: **Posterior probabilities inferred by baycn and partition MCMC for the Drosophila data set.** The same graph was used as the input for both algorithms. Both algorithms agree on which edges are present and which are absent. The rows highlighted in gray are the edges where the posterior probabilities for edge presence are different (e.g., different directions may be inferred).

| | Posterior Probability | | | | | |
|---|---|---|---|---|---|---|
| | baycn | | | partition | | |
| Edge | zero | one | two | zero | one | two |
| Meso-twi_2.4 | 0.282 | 0.644 | 0.074 | 0.22 | 0.70 | 0.08 |
| VM-bin_6.8 | 0.126 | 0.462 | 0.412 | 0.11 | 0.68 | 0.21 |
| VM-bin_8.10 | 0.172 | 0.474 | 0.354 | 0.45 | 0.50 | 0.05 |
| VM-bin_10.12 | 0.076 | 0.054 | 0.870 | 0.16 | 0.13 | 0.71 |
| SM-mef2_10.12 | 0.440 | 0.208 | 0.352 | 0.55 | 0.33 | 0.12 |
| CM-bin_10.12 | 0.420 | 0.196 | 0.384 | 0.61 | 0.34 | 0.05 |
| Meso_SM-tin_2.4 | 0.102 | 0.534 | 0.364 | 0.21 | 0.73 | 0.06 |
| Meso_SM-tin_4.6 | 0.024 | 0.164 | 0.812 | 0.04 | 0.28 | 0.68 |
| Meso_SM-tin_6.8 | 0.180 | 0.040 | 0.780 | 0.03 | 0.17 | 0.80 |
| Meso_SM-mef2_2.4 | 0.226 | 0.556 | 0.218 | 0.34 | 0.66 | 0.00 |
| Meso_SM-mef2_4.6 | 0.174 | 0.282 | 0.544 | 0.44 | 0.20 | 0.36 |
| Meso_SM-mef2_6.8 | 0.024 | 0.050 | 0.926 | 0.07 | 0.13 | 0.80 |
| Meso_SM-mef2_8.10 | 0.052 | 0.056 | 0.892 | 0.14 | 0.14 | 0.72 |
| Meso_SM-mef2_10.12 | 0.036 | 0.048 | 0.916 | 0.11 | 0.14 | 0.75 |
| VM_SM-bin_10.12 | 0.504 | 0.282 | 0.214 | 0.45 | 0.51 | 0.04 |
| VM_SM-twi_2.4 | 0.378 | 0.622 | 0.000 | 0.32 | 0.65 | 0.03 |
| VM_SM-mef2_10.12 | 0.738 | 0.262 | 0.000 | 0.74 | 0.26 | 0.00 |
| tin_2.4-tin_4.6 | 0.332 | 0.668 | 0.000 | 0.20 | 0.80 | 0.00 |
| tin_2.4-twi_4.6 | 0.526 | 0.464 | 0.010 | 0.52 | 0.48 | 0.00 |
| tin_4.6-twi_4.6 | 0.548 | 0.382 | 0.070 | 0.82 | 0.18 | 0.00 |
| tin_4.6-twi_6.8 | 0.522 | 0.476 | 0.002 | 0.74 | 0.26 | 0.00 |
| tin_4.6-mef2_2.4 | 0.932 | 0.066 | 0.002 | 0.87 | 0.12 | 0.01 |
| tin_4.6-mef2_4.6 | 0.972 | 0.028 | 0.000 | 0.96 | 0.04 | 0.00 |
| tin_6.8-bin_6.8 | 0.300 | 0.576 | 0.124 | 0.68 | 0.29 | 0.03 |
| tin_6.8-bap_6.8 | 0.494 | 0.506 | 0.000 | 0.76 | 0.24 | 0.00 |
| bin_6.8-bin_8.10 | 0.604 | 0.396 | 0.000 | 0.89 | 0.11 | 0.00 |
| bin_6.8-bin_10.12 | 0.494 | 0.408 | 0.098 | 0.85 | 0.15 | 0.00 |
| bin_6.8-bap_6.8 | 0.746 | 0.216 | 0.038 | 0.54 | 0.45 | 0.01 |
| bin_8.10-bin_10.12 | 0.500 | 0.500 | 0.000 | 0.54 | 0.46 | 0.00 |
| twi_2.4-twi_4.6 | 0.224 | 0.776 | 0.000 | 0.11 | 0.89 | 0.00 |
| twi_4.6-twi_6.8 | 0.400 | 0.596 | 0.004 | 0.53 | 0.47 | 0.00 |
| twi_4.6-mef2_4.6 | 0.992 | 0.008 | 0.000 | 0.94 | 0.06 | 0.00 |
| twi_6.8-mef2_6.8 | 0.890 | 0.110 | 0.000 | 0.50 | 0.50 | 0.00 |
| mef2_2.4-mef2_6.8 | 0.604 | 0.284 | 0.112 | 0.34 | 0.59 | 0.07 |
| mef2_2.4-mef2_10.12 | 0.310 | 0.610 | 0.080 | 0.54 | 0.44 | 0.02 |
| mef2_4.6-mef2_6.8 | 0.008 | 0.992 | 0.000 | 0.35 | 0.65 | 0.00 |
| mef2_6.8-mef2_8.10 | 0.624 | 0.376 | 0.000 | 0.52 | 0.48 | 0.00 |
| mef2_8.10-mef2_10.12 | 0.172 | 0.828 | 0.000 | 0.13 | 0.87 | 0.00 |

# Chapter 3: A Bayesian graphical model approach to gene regulatory networks with individual level data

**Abstract**

Gene regulatory networks are biological networks that describe the interaction between genes. By measuring gene expression at the individual level, it is possible to infer a network directly from data. However, without additional information, such as an individual's genotype, it is not possible to infer a regulatory relationship from observational data. We modify the general-purpose network inference method (baycn) which we developed previously to infer a gene regulatory network. Here, we propose baycn for Gene Regulatory Networks (BGRN) which incorporates individual level genotype data to infer causal relationships between genes. Our method not only infers the structure of the network but also provides a measure of uncertainty for the inference. We explore the performance of BGRN through extensive simulations and show that it is able to accurately infer the regulatory relationships between genes. We demonstrate that BGRN can accurately infer a network while accounting for some types of confounding variables and that for other types of confounding BGRN does not perform as well. We also show the impact of the number of confounding variables on inference accuracy. We demonstrate that BGRN can correctly identify trait related genes when the correlation between them is low, but when the correlation is high inference accuracy depends on the number of genes that regulate the trait.

## 3.1 Introduction

A gene regulatory network is a graph that illustrates the interactions among genes and describes how gene expression is regulated at a given point in time. The genes in the graph are represented by nodes and an edge represents a relationship between genes. Gene regulatory networks can be inferred directly from individual level gene expression data. With recent advancements in high-throughput technology (e.g., microarrays and next-generation sequencing) the availability of gene expression data has grown tremendously. Along with the growth in data, the research performed on inferring the regulatory relationship among genes has also greatly increased. For example, gene regulatory networks have been used extensively to better understand how genes contribute to complex traits and diseases [87]. Learning the

structure (the relationships between genes) of a gene regulatory network can shed light on what genes are potentially driving disease [49, 54]. This information can be used to more effectively diagnose and treat [70, 54] disease. Therefore, understanding the structure of a gene regulatory network is of great scientific interest.

Many methods have been developed to infer a gene regulatory network [57, 85, 7, 19]. Here, we focus a class of methods based on probabilistic graphical models or networks. In general, these methods use observational gene expression data to learn the structure of the network. The structure inferred by these networks represents the statistical dependence among genes and additional information, such as an individual's genotype, is needed to also infer a causal relationship between genes. The use of genetic variants as instrumental variables [20] is a technique for inferring causal relationships from observational data and is known as the principle of Mendelian randomization (PMR) [4]. A few recent network inference methods [4, 35] employ this technique and include individual level genotype data in the network to infer a variety of causal relationships among genes. However, using genetic variants as instrumental variables is not commonly used by current network inference methods.

In general, current network inference methods use data normalization techniques, such as probabilistic estimation of expression residuals (PEER) normalization [75], to account for the effect of potential confounding from environmental, technical, or demographic variables. They do not explicitly account for the affect of confounding variables on the inferred network by including them as variables in the network. Gene regulation is a complex process and genes not included in the network could play a role in the regulatory relationships among genes in the network. A method, Genomic Mediation analysis with Adaptive Confounding adjustment (GMAC), recently developed by Yang et al. [86] explicitly accounts for the effect of genes outside the network by performing a principal component analysis on the genome-wide expression data and including the top principal components (PCs) in the network. The top PCs capture a large portion of the signal from the genome-wide expression data and serve as a proxy for unknown confounding variables. However, GMAC focuses on inferring one type of relationship ($X \rightarrow Y \rightarrow Z$) and is not applied to larger networks. Furthermore, they consider three types of confounding variables but when they perform the inference two of these types of confounding variables are filtered out and only the third type is accounted for.

We build on our previous method BAYesian Causal Network (baycn) [60] which is a general-purpose network inference method for inferring directed acyclic graphs (DAGs). Our method, baycn, quantifies the uncertainty of the inference by estimating the posterior probability for edge direction and absence for each edge in the network, specifies edge-level prior probabilities, and accounts for prior or outside knowledge of the network structure. We develop baycn for Gene Regulatory Networks (BGRN) which is

designed specifically for inferring gene regulatory networks. Our contributions include adding different variable types which allows for additional information, such as individual level genotype and phenotype data, to be included in the network. We also include assumptions (such as the PMR) that restrict the edge direction between variable types. When incorporating the PMR genetic variants are used to help infer causal relationships between genes. In addition to including genotype and phenotype data we also explicitly account for confounding variables by including them as nodes in the network and explore the effect of confounding variables on the inference. We show through multiple simulation studies that BGRN can effectively use genotype data to accurately infer a variety of regulatory relationships among genes. We demonstrate when BGRN can account for the effect of confounding variables and when the type of confounding variable or the number of confounders adversely affect inference accuracy. Finally, we explore the effect of network structure on identifying trait related genes. We illustrate that when the correlation among these genes is low BGRN can accurately identify the trait related genes. However, when the correlation is high among the trait related genes inference accuracy depends on the number of genes that regulate the trait.

## 3.2 Model

### 3.2.1 Basics of graphical models and notation

In general, a graph $\mathcal{G} = (\mathbf{V},\ \mathbf{E})$ is a set of vertices (or nodes) $\mathbf{V} = \{1, 2, ..., b\}$ and edges $\mathbf{E} \subseteq \mathbf{V} \times \mathbf{V}$, where $\mathbf{V} \times \mathbf{V}$ is the set of all ordered pairs of nodes, such as $(j, k)$, which denotes an edge pointing from node $j$ to node $k$ where $j, k \in \mathbf{V}$. In this example node $j$ is referred to as the parent and node $k$ as the child. Here we use the representation of a graph we previously developed in baycn [60] which defines a graph in terms of the edges and the states an edge can take. Under this representation a graph is denoted by a vector of edge states $\mathbf{S} = (S_1, S_2, \ldots, S_m)$, where $m$ is the number of edges considered and each edge can be in one of three possible states (see [60] for additional details).

We review the distribution of the original variable from baycn as well as describe the distribution of the additional variables added to BGRN. The set of nodes $\mathbf{V}$ corresponds to a random vector

$$\mathbf{X} = (U_1, U_2, \ldots, U_j, T_{j+1}, \ldots, T_k, W_{k+1}, \ldots, W_b)^T,$$

where each $U_j$ and $W_j$ represent discrete random variables and each $T_j$ represents a continuous random variable. In this paper we consider the $U$ nodes as genetic variants (e.g., copy number variation or expression quantitative trait loci), the $W$ nodes as clinical phenotypes (e.g., disease or medical condition),

and the $T$ nodes represent genes (i.e., gene expression). It is important to note that while the model is presented with these node types in mind BGRN is not limited to only analyzing genomic data and can be applied to any data that follow these distributions. We assume the $U_j$ variables to have a multinomial distribution

$$U_j \sim \text{Multinomial}(p_{j_1}, p_{j_2}, \ldots, p_{j_{o-1}}), \tag{3.1}$$

where there are $o$ levels. The $T_j$ variables are assumed to follow a normal distribution

$$T_j \sim \text{N}(\mu_j, \sigma_j^2), \tag{3.2}$$

$$\mu_j = \beta_0 + \sum_{k \in pa(T_j)} \beta_k U_k + \sum_{l \in pa(T_j)} \beta_l T_l, \tag{3.3}$$

where $pa(T_j)$ is the set of parents of node $T_j$ and $\sigma_j^2$ the variance. If the node $T_j$ does not have any parents then $\mu_j = \beta_0$. We assume the $W_j$ variables to be binary where:

$$W_j \sim \text{Bernoulli}(p_j), \tag{3.4}$$

$$\log\left(\frac{p_j}{1-p_j}\right) = \beta_0 + \sum_{k \in pa(W_j)} \beta_k U_k + \sum_{l \in pa(W_j)} \beta_l T_l, \tag{3.5}$$

where $p_j$ is the "success" probability. If node $W_j$ does not have any parents then $\log\big(p_j/(1-p_j)\big) = \beta_0$.

### 3.2.2 Graph assumptions

We include several biological assumptions or restrictions between the different variable types. The first assumption we use is the principle of Mendelian randomization (PMR) which is necessary for inferring causal relationships from observational data. The gold standard for determining a causal relationship is the randomized controlled trial (RCT). However, it is often not possible or ethical to carry out an RCT when dealing with human subjects. The PMR is an alternative approach to an RCT in that it assumes alleles are randomly assigned to individuals in a population. Because alleles are assumed to be randomly assigned to an individual and are independent of any other factors they can be used as an instrumental variable. Two more assumptions are necessary when using genetic variants as instrumental variables: i) the genotype causes the phenotype – not the other way around and ii) the genotype is independent of any confounding variables that are associated with the genes. Incorporating the PMR means the following constraint is placed on the network: genetic variants are always the parents of genes. By including the PMR we are able to uniquely infer the direction of more edges in the network because the PMR breaks

up the Markov equivalence classes of a graph. When using the PMR directed edges represent a statistical relationship between genes. These edges can also imply the causal or regulatory relationship among genes, but a directed edge does not guarantee a causal relationship and care should be taken when interpreting directed edges as causal effects [18].

Two graphs are Markov equivalent [82] if they represent the same conditional independence structure. Multiple Markov equivalent graphs form a Markov equivalence class. For example, the graph $X \rightarrow Y \rightarrow Z$ has two other Markov equivalent graphs $X \leftarrow Y \rightarrow Z$ and $X \leftarrow Y \leftarrow Z$. Due to Markov equivalence network inference methods are unable to distinguish between these three graphs and are, therefore, unable to determine the true direction of the edges. If, however, $X$ was restricted to be the parent of $Y$ the graph $X \rightarrow Y \rightarrow Z$ has no other Markov equivalent graphs and, with this restriction in place, network inference methods can determine the true direction of the $Y - Z$ edge.

We include an additional restriction in the network that involves clinical phenotype variables. We use the assumption that a clinical phenotype is influenced by genetic variants and genes. In other words, the clinical phenotypes are restricted to always being the children of genetic variants and genes. Both the PMR and clinical phenotype assumptions reduce the space of DAGs searched by BGRN.

When including edges between two genetic variant nodes we assume there is no regulatory relationship among them. However, in the Metropolis-Hastings algorithm a directed edge will always be proposed between the two nodes, but the inference should reflect the Markov equivalence between the nodes with the two directions being equally likely.

### 3.2.3 Metropolis-Hastings algorithm

Here we demonstrate how the additional node types and assumptions affect the Metropolis-Hastings algorithm we developed previously [60]. The key steps from the original algorithm remain the same. However, while the input and main steps of the algorithm are the same, there are several differences which allow for the PMR and clinical phenotype assumptions to be incorporated into BGRN. First, we include additional steps in the Metropolis-Hastings algorithm when using the biological assumptions to ensure the edges between the different variable (or node) types are oriented in the correct direction. Line 1 of Algorithm 2 ensures that all edges from the genetic variant nodes are oriented so the genetic variant is the parent of all other variable types. This step also checks that the clinical phenotype variables are not the parents of any other variable types. If any edges violate these restrictions the direction of these edges is reversed. The same procedure of checking for and correcting edge directions occurs again on Line 4 of Algorithm 2. After this point we do not need to check for violations of the PMR or clinical phenotype assumptions because the edges involving the genetic variant or clinical phenotype nodes are

not changed again until the next iteration of the algorithm. For example, the graph proposed at Line 6 in Algorithm 2, after directed cycles have been removed, cannot have a genetic variant or clinical phenotype edge oriented in the wrong direction because cycles can only occur among variables of the same type (e.g., gene expression variables).

**Algorithm 2:** BGRN

**input :** Data matrix, candidate graph, and prior on edge states
**output:** Posterior probability of edge states for each edge considered

1 Randomly generate a starting graph $\mathbf{S}_{(1)}$ from the candidate graph;
2 Check for and correct any violations of the PMR and clinical phenotype assumptions in $\mathbf{S}_{(1)}$;
3 **for** $i \leftarrow 2$ **to** $M$ **do**
4 Generate a proposal graph $\mathbf{S}'_{(t)}$ from the current graph $\mathbf{S}_{(t-1)}$;
5 Check for and correct any violations of the PMR and clinical phenotype assumptions in $\mathbf{S}'_{(t)}$;
6 Check for and remove directed cycles in $\mathbf{S}'_{(t)}$;
7 Calculate the acceptance probability
$$\alpha_{(t)} = \min\left\{\frac{\Pr(\mathbf{S}'_{(t)})\Pr(\mathbf{T} \mid \mathbf{S}'_{(t)}, \boldsymbol{\theta}_{(t)})\Pr(\mathbf{S}_{(t-1)} \mid \mathbf{S}'_{(t)})}{\Pr(\mathbf{S}_{(t-1)})\Pr(\mathbf{T} \mid \mathbf{S}_{(t-1)}, \boldsymbol{\theta}_{(t-1)})\Pr(\mathbf{S}'_{(t)} \mid \mathbf{S}_{(t-1)})},\ 1\right\};$$
8 Generate $u$ from Uniform$(0, 1)$;
9 **if** $u < \alpha_{(t)}$ **then**
10 $\mathbf{S}_{(t)} = \mathbf{S}'_{(t)}$;
11 **else**
12 $\mathbf{S}_{(t)} = \mathbf{S}_{(t-1)}$;
13 **end**
14 **end**

The steps added to Algorithm 2 which carry out the PMR and clinical phenotype assumptions do not affect the transition probability and acceptance ratio calculations. These calculations are not affected because they only depend on the current and proposed graphs and the additional steps that enforce these assumptions occur as a part of generating a starting (Line 1) and proposal (Line 4) graph. Therefore, a graph is never proposed, or accepted, with an edge oriented in a way that violates the PMR and clinical phenotype assumptions.

We also change how the log likelihood of the network is calculated to account for the distributions of the two discrete variable types. We assume the genetic variant nodes follow a multinomial distribution (Equation 3.1) and we use ordinal logistic regression [62] when calculating the likelihood of a genetic variant that has one or more parents. We assume the clinical phenotype nodes follow a binomial distribution (Equation 3.5) and we use logistic regression to calculate the log likelihood when a clinical phenotype has one or more parents.

## 3.3 Simulation study

### 3.3.1 Data simulation

We simulated data under two main scenarios: graphs without confounding variables and graphs with confounding variables. The simulations without confounding variables can be further broken down into two different studies. We refer to these two simulation studies as PMR and clinical phenotype.

For the PMR simulations we consider seven different topologies (Figure 3.1) which have different numbers of nodes, edges, and v structures. Topologies M1, M2 (a v structure), M3, and M4 are the building blocks of larger more complex graphs. We use these graphs to test the ability of BGRN to correctly identify edge directions in simple graphs. The multi-parent graph is made up of multiple v structures which other network inference algorithms struggle in correctly identifying the direction of all the edges [4]. The star and layer topologies are made up of multiple M1 models and represent common structures found in biological systems [1]. We simulated data with a sample size of 50, 200, and 500 with signal strength values of 0.2, 0.5, and 1. These simulations demonstrate how BGRN can use the PMR to uniquely infer the direction of more edges in the network.

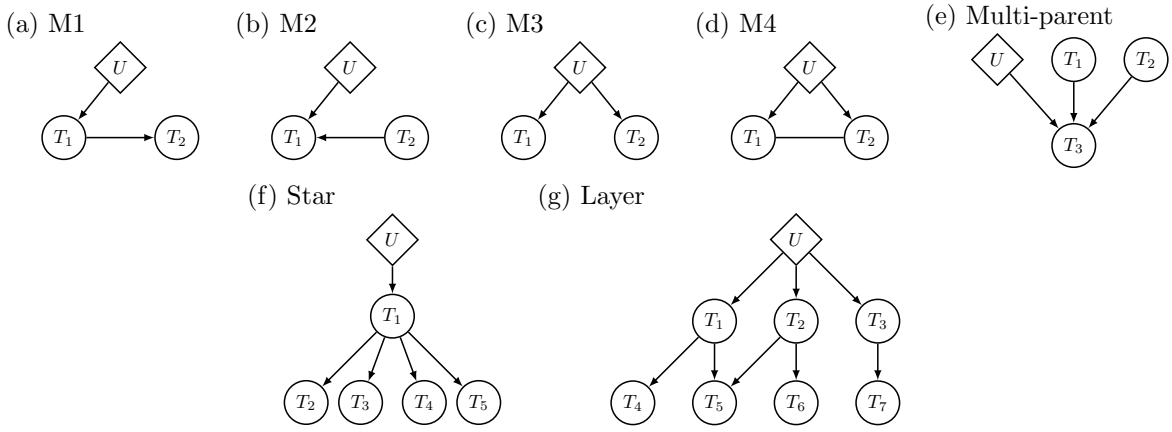



Figure 3.1: **Seven topologies used in the PMR simulation studies.** In these topologies the $U$ nodes represent genetic variants and the $T$ nodes represent gene expression. (**a**)-(**d**) Small graphs that are the building blocks of larger networks. M2 is also known as a v structure. (**e**) This topology is made up of multiple v structures. (**f**)-(**g**) Larger topologies that are made up of multiple M1 subgraphs.

For the confounding variable simulations we consider four topologies (Figure 3.2) with three different types of confounding variables: common parent, common child, and intermediate (Figure 3.2). We simulated data with a sample size of 200, 500, and 1500. We also used the signal strength values of 0.2, 0.5, and 1 between the non-confounding variables. We used a signal strength of 0.2 for all edges between a confounding and non-confounding variable, even when the signal strength between the non-confounding

variables is 0.5 or 1. For topologies M1, M2, and M3 we simulated data for graphs with 2, 5, and 10 confounding variables between nodes $T_1$ and $T_2$. For the layer topology we simulated data for graphs with two confounding variables one between nodes $T_1$ and $T_5$ and another between nodes $T_2$ and $T_6$. We chose these two edges because from simulations without confounding variables these two edges had the lowest and highest eMSE (Equation 2.26) respectively. We only simulated data with the common parent and intermediate confounding variables for the layer topology.

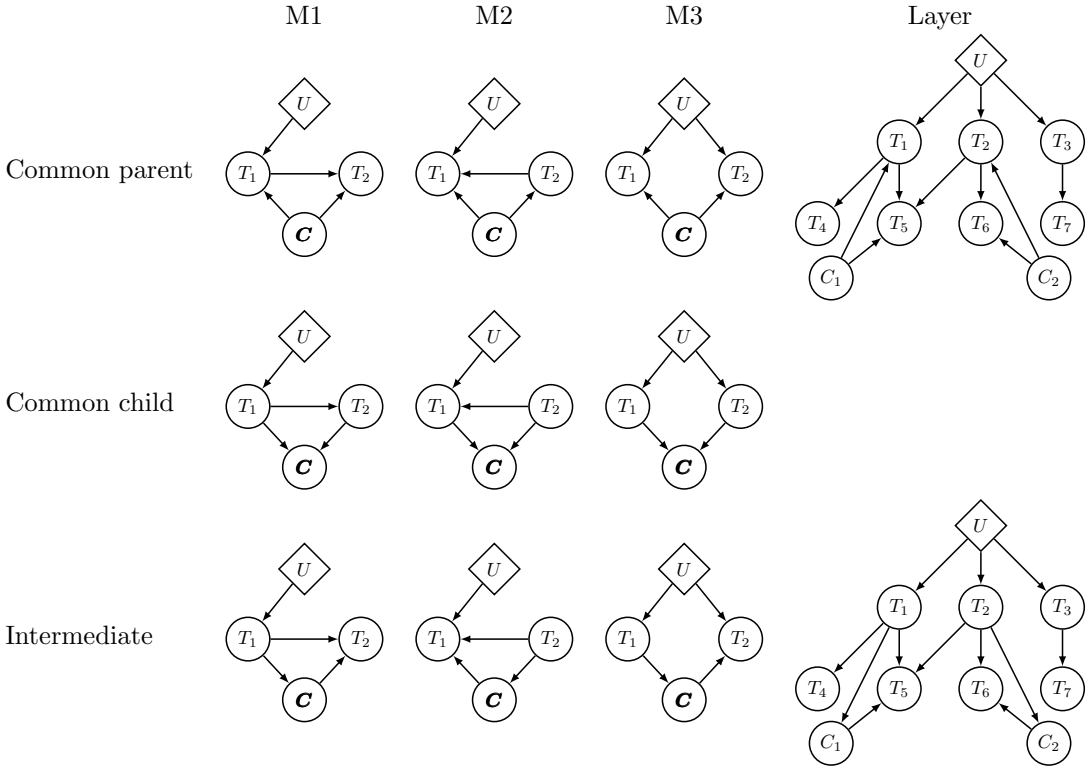



Figure 3.2: **Four topologies and three confounding variable types used in simulation studies.** In these topologies the $U$ nodes represent genetic variants, the $T$ nodes represent gene expression, and the $C$ nodes represent confounding variables. In graphs M1-M3 $\boldsymbol{C}$ represents a vector of confounding variables.

For the clinical phenotype simulations we simulated data for a wide range of topologies. We generated data with sample sizes of 500, 1,000, and 2,000 and signal strength values of 0.2, 0.5, 1, 1.2, and 1.5. For the same topology we also varied the number of true parents of the clinical phenotype and varied which gene expression variables were the true parents. The signal strength for the edges to the clinical phenotype is the same as the signal strength between the genetic variant and gene expression variables for the values 0.2, 0.5, and 1. However, when the signal strength between the genetic variant and gene expression variables is either 1.2 or 1.5 the signal strength for the edges to the clinical phenotype is fixed

at 1.

For all simulation studies we generated data for genetic variants under a multinomial distribution (Equation 3.1) with three levels where $(p_1, p_2, p_3) = (0.3025, 0.4950, 0.2025)$. We generated data under a normal distribution with the mean following a linear model (Equation 3.3) for the gene expression and confounding variables. We generated data under a Bernoulli distribution with the log odds following a linear model (Equation 3.5) for the clinical phenotype. For both the normal and Bernoulli distributed variables if the node does not have any parents, then the linear model is reduced to the intercept $\beta_0$. For simplicity, we set the variance to one for all normally distributed variables. The intercepts are set to zero for all linear models, and all other $\beta$s take the same value. We refer to the $\beta$ coefficients of the parents as the signal strength. For all topologies in the three simulation scenarios we simulated 25 independent data sets for each combination of topology, signal strength, and sample size.

### 3.3.2 Identifying causal relationships using the principle of Mendelian randomization

For each topology, we ran BGRN once per simulated data set, used a burn-in of 20%, and used the prior $(p_0, p_1, p_2) = (0.05, 0.05, 0.9)$ on edge states. For the M1, M2, M3, M4, and multi-parent topologies (Figure 3.1a-e) we ran BGRN for 30,000 iterations and a step size of 120. For the layer and star topologies (Figure 3.1f-g) we ran BGRN for 75,000 iterations with a step size of 300.

For this simulation study we calculate precision, power, and $\text{MSE}_1$ (see Section 2.3.2) to measure inference accuracy. We use the cutoff of 0.4 for edge presence when calculating precision and power. Precision is one nearly for the M1, M2, M3, and multi-parent topologies for all three sample sizes when the signal strength is 1. For the star and layer topologies, which are made up of multiple M1 models, precision is close to one for all three sample sizes when the signal strength is 1 and when the signal strength is lower (0.2 or 0.5) precision is also close to one for a large sample size but decreases as the sample size decreases. Power is near one for all topologies and all sample sizes when the signal strength is 0.5 and 1. These results show that BGRN correctly identifies the true edges the majority of the time when the signal strength is high or when the sample size is large but it tends to infer some false edges as present, for larger topologies, under these circumstances.

We also assess the performance of BGRN with $\text{MSE}_1$. We use the cutoff of $\text{MSE}_1 < 0.1$ to indicate accurate inference (see [60] for more details). Overall, both the sample size $N$ and signal strength $\beta$ play a role in how well BGRN performs as $\text{MSE}_1$ decreases as both $N$ and $\beta$ increase (Table 3.1). Even for a sample size of only 50 $\text{MSE}_1 < 0.1$ for a strong signal for all seven topologies. For a moderate signal ($\beta = 0.5$), $\text{MSE}_1$ falls below the 0.1 cutoff for the larger sample sizes. However, for a moderate signal

strength and a sample size of 50, $MSE_1$ is above the cutoff or just below it. Therefore, a strong signal is needed for very small sample sizes for the inference to be accurate.

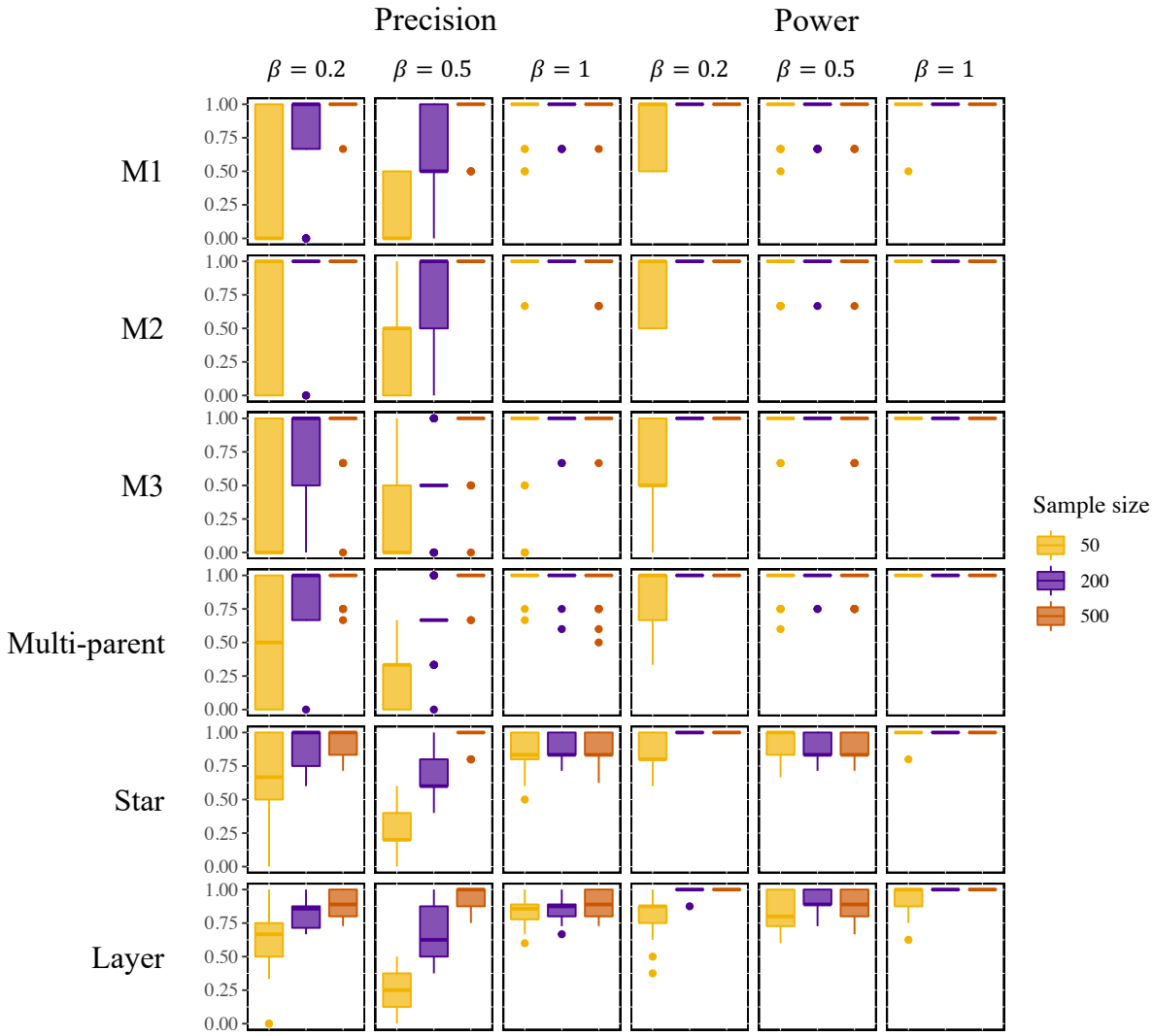



Figure 3.3: **Boxplots of precision and power for BGRN.** For each topology we simulated 25 independent data sets for each combination of signal strength and sample size. A fully connected graph was used as the input to BGRN. We considered an edge present if the posterior probability for edge presence (the sum of the two directions) was greater than 0.4.

Table 3.1: **The mean and standard deviation of $MSE_1$ (Equation 2.27) for all topologies in the PMR simulation study.** For each combination of topology, sample size, and signal strength we simulated 25 data sets and ran BGRN once per data set. We used a fully connected graph as the input to BGRN.

| | | $MSE_1$ | | | | | |
|---|---|---|---|---|---|---|---|
| | | $\beta = 0.2$ | | $\beta = 0.5$ | | $\beta = 1$ | |
| Topology | $N$ | mean | sd | mean | sd | mean | sd |
| M1 | 50 | 0.2773 | 0.2162 | 0.1248 | 0.1441 | 0.0687 | 0.12 |
| | 200 | 0.1869 | 0.1748 | 0.0401 | 0.0716 | 0.0296 | 0.0466 |
| | 500 | 0.0911 | 0.109 | 0.0323 | 0.0706 | 0.029 | 0.0634 |
| M2 | 50 | 0.2608 | 0.2133 | 0.1282 | 0.1418 | 0.0471 | 0.0657 |
| | 200 | 0.1398 | 0.1429 | 0.0364 | 0.0587 | 0.0202 | 0.0433 |
| | 500 | 0.0579 | 0.0755 | 0.0338 | 0.0781 | 0.0153 | 0.0238 |
| M3 | 50 | 0.2526 | 0.2212 | 0.1463 | 0.1752 | 0.0157 | 0.0452 |
| | 200 | 0.1842 | 0.2034 | 0.0096 | 0.0339 | 0.0035 | 0.0071 |
| | 500 | 0.047 | 0.1076 | 0.0055 | 0.0151 | 0.0122 | 0.048 |
| M4 | 50 | 0.3241 | 0.1708 | 0.1333 | 0.1527 | 0.0597 | 0.103 |
| | 200 | 0.1874 | 0.1823 | 0.0117 | 0.0269 | 4e-04 | 9e-04 |
| | 500 | 0.0421 | 0.0966 | 2e-04 | 7e-04 | 3e-04 | 9e-04 |
| Multi-parent | 50 | 0.2355 | 0.2254 | 0.0936 | 0.1257 | 0.0319 | 0.0738 |
| | 200 | 0.1484 | 0.1596 | 0.0265 | 0.0488 | 0.0084 | 0.0296 |
| | 500 | 0.0715 | 0.0862 | 0.0249 | 0.0563 | 0.0175 | 0.0518 |
| Star | 50 | 0.1568 | 0.2082 | 0.0858 | 0.1324 | 0.0291 | 0.0612 |
| | 200 | 0.0992 | 0.1489 | 0.0246 | 0.0431 | 0.0215 | 0.0391 |
| | 500 | 0.0569 | 0.0901 | 0.0245 | 0.0524 | 0.0251 | 0.0582 |
| Layer | 50 | 0.134 | 0.1981 | 0.0717 | 0.1254 | 0.0379 | 0.0751 |
| | 200 | 0.0898 | 0.1389 | 0.0319 | 0.0676 | 0.0213 | 0.0445 |
| | 500 | 0.0478 | 0.08 | 0.0218 | 0.0503 | 0.0217 | 0.0515 |

### 3.3.3 Estimating posterior probabilities in the presence of confounding variables

For each topology, we ran BGRN once per simulated data set, used a burn-in of 20%, and used the prior $(p_0, p_1, p_2) = (0.05, 0.05, 0.9)$ on edge states. For the M1, M2, and M3 topologies (Figure 3.2 columns 1-3) we ran BGRN for 30,000 iterations and a step size of 120. For the layer topology (Figure 3.1, 4th column) we ran BGRN for 50,000 iterations with a step size of 200. For topologies M1 and M2 we used the true edges and a fully connected graph as the input. For the layer topology we also used two different inputs to BGRN: the true edges and a fully connected graph between the non-confounding variables in addition to the true edges involving the confounding variables. For topology M3 we only used the fully connected graph as the input as we wanted to test how confounding variables affected the

inference when an edge is not present between two nodes. When there are either 5 or 10 confounding variables in topologies M1, M2, and M3 we performed a principal component analysis (PCA) on the confounding variables. After performing the PCA we kept the top two principal components (PCs) and included them in the network as confounding variables. The amount of variation for the top two PCs for each type of confounding variable is shown in Table 3.3. We used the same inputs for topologies M1-M3 with 5 and 10 confounding variables as with two confounding variables.

We assess the performance of BGRN with the eMSE (see Section 2.3.2 for details) on the edges between the nodes where confounding variables are present (e.g., the $T_1 - T_2$ edge in topologies M1-M3). We do not use the whole graph MSE as a measure of inference accuracy as we are most interested in how the presence of confounding variables affect the inference of the edges where confounding variables are present. In addition, previous simulation studies (for example, the previous section using the PMR) have shown that inference accuracy is generally low when the signal strength is 0.2 which is the signal strength used for confounding variables. Therefore, we expect the eMSE for the confounding variable edges to be high. For all topologies, sample sizes, and number of confounders the eMSE for the edges where confounding variables are present (e.g., the $T_1 - T_2$ edge in topologies M1-M3) is zero or near zero when the true edges are the input and the signal strength is 1 (Tables 3.4 and 3.5). The majority of the time the eMSE falls below the 0.1 cutoff when the signal strength is 0.5. However, there are a few cases, usually when the number of confounders is 10, that the eMSE is much higher than this cutoff (Table 3.4).

When a fully connected graph is used as input the type of confounding variable greatly affects inference accuracy (Table 3.2). For example, when common parent confounding variables are present in topologies M1 and M2 the posterior for edge presence for the $U - T_2$ edge increases as the sample size increases. When this edge is inferred present, the $T_1 - T_2$ edge is inferred in the opposite direction of the true direction with a high posterior probability. The $T_2 - T_6$ edge in the layer topology also suffers from this effect at larger sample sizes in the presence of common parent and intermediate confounding variables (Table 3.2). In this topology the $U - T_6$ edge is inferred present and the direction of the $T_2 - T_6$ edge is inferred in the opposite direction from the true direction.

Overall the $T_1 - T_2$ edge is largely unaffected by confounding variables for topology M3 when 2 confounding variables are present (Table 3.2). However, in the presence of common parent confounding variables this edge is often inferred as present when there are 5 or 10 confounding variables in the network (Table 3.6). These results show that unless the signal strength is lower or the number of confounding variables is high the presence of confounding variables between two nodes does not cause an edge to be inferred as present between them. In summary, when a fully connected graph is used as the input inference accuracy depends on the type of confounding variable present (Table 3.2). On the other hand,

the type of confounding variable does not affect inference accuracy, for a strong signal, when the true edges are used as the input. This is the case even when there are a large number of confounding variables in the network (Table 3.4).

Table 3.2: **The mean and standard deviation of the eMSE for the $T_1 - T_2$, $T_1 - T_5$, and $T_2 - T_6$ edges with 2 confounding variables.** The input to BGRN was a fully connected graph.

| | | | | eMSE | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| | | | | $\beta = 0.2$ | | $\beta = 0.5$ | | $\beta = 1$ | |
| Topology | Type CV | N | edge | mean | sd | mean | sd | mean | sd |
| M1 | Common parent | 200 | T1-T2 | 0.2547 | 0.1157 | 0.1409 | 0.1391 | 0.1157 | 0.1157 |
| | | 500 | T1-T2 | 0.2149 | 0.096 | 0.105 | 0.1396 | 0.4045 | 0.2161 |
| | | 1500 | T1-T2 | 0.0944 | 0.1222 | 0.0434 | 0.0615 | 0.5918 | 0.1342 |
| | Common child | 200 | T1-T2 | 0.3781 | 0.144 | 0.0628 | 0.0899 | 0.022 | 0.0312 |
| | | 500 | T1-T2 | 0.2313 | 0.1351 | 0.03 | 0.0564 | 0.0161 | 0.0287 |
| | | 1500 | T1-T2 | 0.1174 | 0.0842 | 0.0178 | 0.0296 | 0.0197 | 0.0445 |
| | Intermediate | 200 | T1-T2 | 0.2761 | 0.1203 | 0.1751 | 0.1737 | 0.0425 | 0.0696 |
| | | 500 | T1-T2 | 0.2396 | 0.1145 | 0.0727 | 0.1005 | 0.023 | 0.034 |
| | | 1500 | T1-T2 | 0.1035 | 0.1463 | 0.0251 | 0.0356 | 0.0378 | 0.0499 |
| M2 | Common parent | 200 | T1-T2 | 0.2543 | 0.1177 | 0.1419 | 0.1569 | 0.1417 | 0.1878 |
| | | 500 | T1-T2 | 0.2125 | 0.1306 | 0.0845 | 0.1206 | 0.2302 | 0.1618 |
| | | 1500 | T1-T2 | 0.1173 | 0.0896 | 0.027 | 0.0532 | 0.5116 | 0.1745 |
| | Common child | 200 | T1-T2 | 0.4182 | 0.1588 | 0.1268 | 0.1309 | 0.0303 | 0.0446 |
| | | 500 | T1-T2 | 0.2756 | 0.0993 | 0.0368 | 0.0535 | 0.0557 | 0.0532 |
| | | 1500 | T1-T2 | 0.1709 | 0.1095 | 0.0589 | 0.0586 | 0.0497 | 0.0596 |
| | Intermediate | 200 | T1-T2 | 0.2385 | 0.1091 | 0.1782 | 0.1805 | 0.0873 | 0.1389 |
| | | 500 | T1-T2 | 0.1802 | 0.1196 | 0.1202 | 0.1257 | 0.2752 | 0.187 |
| | | 1500 | T1-T2 | 0.1167 | 0.0994 | 0.0131 | 0.0446 | 0.2927 | 0.2053 |
| M3 | Common parent | 200 | T1-T2 | 0.0248 | 0.0459 | 0.0133 | 0.017 | 0.0121 | 0.0179 |
| | | 500 | T1-T2 | 0.0091 | 0.019 | 0.0101 | 0.0217 | 0.0356 | 0.0872 |
| | | 1500 | T1-T2 | 0.0219 | 0.0405 | 0.0077 | 0.008 | 0.012 | 0.0111 |
| | Common child | 200 | T1-T2 | 0.0171 | 0.0233 | 0.0112 | 0.0131 | 0.0079 | 0.006 |
| | | 500 | T1-T2 | 0.0224 | 0.0437 | 0.0226 | 0.0619 | 0.0155 | 0.0248 |
| | | 1500 | T1-T2 | 0.0547 | 0.0932 | 0.0179 | 0.0244 | 0.0252 | 0.0432 |
| | Intermediate | 200 | T1-T2 | 0.0153 | 0.021 | 0.034 | 0.0865 | 0.011 | 0.0135 |
| | | 500 | T1-T2 | 0.0352 | 0.0748 | 0.0104 | 0.0288 | 0.0316 | 0.0893 |
| | | 1500 | T1-T2 | 0.0119 | 0.0156 | 0.0068 | 0.0075 | 0.0221 | 0.0701 |
| Layer | Common parent | 200 | T1-T5 | 0.2693 | 0.1392 | 0.0257 | 0.0493 | 3e-04 | 8e-04 |
| | | | T2-T6 | 0.2362 | 0.0872 | 0.0962 | 0.1675 | 0.0837 | 0.1822 |
| | | 500 | T1-T5 | 0.1377 | 0.1424 | 0.0051 | 0.019 | 0.0332 | 0.0747 |
| | | | T2-T6 | 0.1545 | 0.2351 | 0.0705 | 0.186 | 0.1862 | 0.2724 |
| | | 1500 | T1-T5 | 0.113 | 0.1289 | 3e-04 | 0.0012 | 0.015 | 0.0365 |
| | | | T2-T6 | 0.2933 | 0.3377 | 0.2933 | 0.3377 | 0.2588 | 0.3196 |
| | Intermediate | 200 | T1-T5 | 0.2436 | 0.1458 | 0.0265 | 0.0588 | 0.0047 | 0.0173 |
| | | | T2-T6 | 0.2506 | 0.1279 | 0.0816 | 0.1028 | 0.1119 | 0.2312 |
| | | 500 | T1-T5 | 0.1669 | 0.1365 | 0.0075 | 0.0181 | 9e-04 | 0.0035 |
| | | | T2-T6 | 0.23 | 0.2463 | 0.2415 | 0.3101 | 0.1203 | 0.2489 |
| | | 1500 | T1-T5 | 0.0526 | 0.0598 | 0 | 1e-04 | 0 | 0 |
| | | | T2-T6 | 0.32 | 0.3399 | 0.2667 | 0.3333 | 0.2133 | 0.3174 |

Table 3.3: **The average percent of variation explained by the top two PCs for networks with 5 or 10 confounding variables.**

| | | | | Percent variation of top 2 PCs | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| | | | | $\beta = 0.2$ | | $\beta = 0.5$ | | $\beta = 1$ | |
| Topology | Type CV | $N$ | # CV | mean | sd | mean | sd | mean | sd |
| M1 | Common parent | 200 | 5 | 0.4687 | 0.0133 | 0.4689 | 0.0137 | 0.467 | 0.0145 |
| | | | 10 | 0.2653 | 0.0095 | 0.2693 | 0.0095 | 0.2677 | 0.0074 |
| | Common child | | 5 | 0.4914 | 0.0163 | 0.5184 | 0.0161 | 0.5693 | 0.025 |
| | | | 10 | 0.3169 | 0.016 | 0.3372 | 0.0194 | 0.4062 | 0.0188 |
| | Intermediate | | 5 | 0.4803 | 0.0179 | 0.4792 | 0.014 | 0.4779 | 0.0133 |
| | | | 10 | 0.2802 | 0.0104 | 0.2848 | 0.0086 | 0.2821 | 0.0103 |
| | Common parent | 500 | 5 | 0.4478 | 0.0091 | 0.4438 | 0.01 | 0.4441 | 0.0099 |
| | | | 10 | 0.2414 | 0.0053 | 0.2419 | 0.005 | 0.242 | 0.0056 |
| | Common child | | 5 | 0.4795 | 0.0081 | 0.4966 | 0.0116 | 0.5499 | 0.0151 |
| | | | 10 | 0.2945 | 0.0105 | 0.3225 | 0.0086 | 0.3978 | 0.0121 |
| | Intermediate | | 5 | 0.4527 | 0.0113 | 0.4523 | 0.0074 | 0.4617 | 0.0115 |
| | | | 10 | 0.2568 | 0.0082 | 0.2632 | 0.0067 | 0.2702 | 0.0072 |
| M2 | Common parent | 200 | 5 | 0.4666 | 0.0168 | 0.4689 | 0.0192 | 0.4713 | 0.0127 |
| | | | 10 | 0.2711 | 0.007 | 0.2656 | 0.0072 | 0.2672 | 0.0102 |
| | Common child | | 5 | 0.4956 | 0.0181 | 0.5127 | 0.0194 | 0.5459 | 0.0201 |
| | | | 10 | 0.3102 | 0.0126 | 0.3344 | 0.0166 | 0.3757 | 0.0149 |
| | Intermediate | | 5 | 0.4757 | 0.0169 | 0.478 | 0.0217 | 0.4787 | 0.0113 |
| | | | 10 | 0.2763 | 0.0109 | 0.2795 | 0.011 | 0.2813 | 0.0134 |
| | Common parent | 500 | 5 | 0.4441 | 0.0097 | 0.4389 | 0.0077 | 0.445 | 0.0088 |
| | | | 10 | 0.2412 | 0.006 | 0.2423 | 0.0046 | 0.2438 | 0.0069 |
| | Common child | | 5 | 0.4783 | 0.0109 | 0.4976 | 0.0093 | 0.526 | 0.0145 |
| | | | 10 | 0.2968 | 0.01 | 0.3195 | 0.0122 | 0.3643 | 0.0114 |
| | Intermediate | | 5 | 0.4554 | 0.0088 | 0.4518 | 0.0091 | 0.4522 | 0.0118 |
| | | | 10 | 0.2576 | 0.0066 | 0.2576 | 0.0075 | 0.2595 | 0.0093 |
| M3 | Common parent | 200 | 5 | 0.4682 | 0.0145 | 0.4653 | 0.0113 | 0.463 | 0.0126 |
| | | | 10 | 0.2704 | 0.0083 | 0.2665 | 0.01 | 0.2724 | 0.0087 |
| | Common child | | 5 | 0.4905 | 0.0176 | 0.5017 | 0.0224 | 0.5168 | 0.0213 |
| | | | 10 | 0.3028 | 0.0128 | 0.3129 | 0.014 | 0.3515 | 0.0158 |
| | Intermediate | | 5 | 0.4737 | 0.0146 | 0.4832 | 0.0156 | 0.4798 | 0.0155 |
| | | | 10 | 0.2806 | 0.0114 | 0.2799 | 0.0115 | 0.2878 | 0.0118 |
| | Common parent | 500 | 5 | 0.4402 | 0.009 | 0.438 | 0.0094 | 0.4424 | 0.0097 |
| | | | 10 | 0.2425 | 0.005 | 0.2414 | 0.0042 | 0.2424 | 0.0059 |
| | Common child | | 5 | 0.4714 | 0.0094 | 0.4784 | 0.0111 | 0.5056 | 0.0112 |
| | | | 10 | 0.2875 | 0.0084 | 0.2968 | 0.0074 | 0.3328 | 0.014 |
| | Intermediate | | 5 | 0.4526 | 0.0092 | 0.4548 | 0.0124 | 0.4604 | 0.0097 |
| | | | 10 | 0.262 | 0.0076 | 0.2598 | 0.0057 | 0.2694 | 0.0079 |

Table 3.4: **The mean and standard deviation of the eMSE for the $T_1 - T_2$ edge in topologies M1 and M2.** The true edges were used as the input to BGRN.

| | | | | eMSE for the $T_1 - T_2$ edge | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| | | | | $\beta = 0.2$ | | $\beta = 0.5$ | | $\beta = 1$ | |
| Topology | Type CV | $N$ | # CV | mean | sd | mean | sd | mean | sd |
| M1 | Common parent | 200 | 2 | 0.2299 | 0.1208 | 0.1195 | 0.1771 | 0 | 0 |
| | | | 5 | 0.1809 | 0.072 | 0.0793 | 0.137 | 0 | 0 |
| | | | 10 | 0.2359 | 0.1124 | 0.194 | 0.2279 | 0.0051 | 0.0241 |
| | | 500 | 2 | 0.2008 | 0.0878 | 0.0433 | 0.1407 | 0 | 0 |
| | | | 5 | 0.1575 | 0.0864 | 0.1218 | 0.1948 | 0 | 0 |
| | | | 10 | 0.2291 | 0.147 | 0.0308 | 0.0638 | 0 | 0 |
| | Common child | 200 | 2 | 0.3883 | 0.1499 | 0.037 | 0.1022 | 0 | 0 |
| | | | 5 | 0.5177 | 0.0728 | 0.0833 | 0.1241 | 0 | 0 |
| | | | 10 | 0.4287 | 0.1583 | 0.4027 | 0.1573 | 0 | 0 |
| | | 500 | 2 | 0.2306 | 0.158 | 0.0063 | 0.0259 | 0 | 0 |
| | | | 5 | 0.4966 | 0.0677 | 0.0235 | 0.0825 | 0 | 0 |
| | | | 10 | 0.1827 | 0.1512 | 0.2805 | 0.228 | 0 | 0 |
| | intermediate | 200 | 2 | 0.258 | 0.1208 | 0.1563 | 0.2133 | 0 | 0 |
| | | | 5 | 0.1895 | 0.1441 | 0.0766 | 0.1377 | 0 | 0 |
| | | | 10 | 0.2305 | 0.1157 | 0.0321 | 0.0891 | 0.0012 | 0.0062 |
| | | 500 | 2 | 0.2091 | 0.1278 | 0.0367 | 0.1044 | 0 | 0 |
| | | | 5 | 0.1218 | 0.1326 | 0.0011 | 0.0053 | 0 | 0 |
| | | | 10 | 0.1128 | 0.1365 | 1e-04 | 5e-04 | 0 | 0 |
| M2 | Common parent | 200 | 2 | 0.2571 | 0.1362 | 0.0987 | 0.1786 | 0.0019 | 0.0094 |
| | | | 5 | 0.1699 | 0.0991 | 0.0506 | 0.0978 | 0 | 0 |
| | | | 10 | 0.1509 | 0.0602 | 0.0751 | 0.1523 | 0 | 0 |
| | | 500 | 2 | 0.2062 | 0.1579 | 0.0394 | 0.124 | 0 | 0 |
| | | | 5 | 0.1611 | 0.1035 | 0.0374 | 0.1275 | 0 | 0 |
| | | | 10 | 0.1392 | 0.1061 | 0.0058 | 0.0288 | 0 | 0 |
| | Common child | 200 | 2 | 0.4158 | 0.1586 | 0.1009 | 0.1534 | 0 | 0 |
| | | | 5 | 0.5474 | 0.0457 | 0.1174 | 0.1786 | 0 | 0 |
| | | | 10 | 0.4077 | 0.1117 | 0.3324 | 0.2017 | 0 | 0 |
| | | 500 | 2 | 0.2434 | 0.1206 | 0.0118 | 0.0378 | 0 | 0 |
| | | | 5 | 0.5287 | 0.0765 | 0.035 | 0.0716 | 0 | 0 |
| | | | 10 | 0.2341 | 0.1118 | 0.0758 | 0.1386 | 0 | 0 |
| | Intermediate | 200 | 2 | 0.2386 | 0.1091 | 0.1702 | 0.2147 | 0 | 0 |
| | | | 5 | 0.2647 | 0.1371 | 0.1343 | 0.212 | 1e-04 | 4e-04 |
| | | | 10 | 0.157 | 0.0911 | 0.0965 | 0.1673 | 0 | 1e-04 |
| | | 500 | 2 | 0.1889 | 0.1297 | 0.0286 | 0.0665 | 0 | 0 |
| | | | 5 | 0.1237 | 0.1045 | 0.0409 | 0.1368 | 0 | 0 |
| | | | 10 | 0.1292 | 0.0845 | 0.0153 | 0.0414 | 0 | 0 |

Table 3.5: **The mean and standard deviation of the eMSE for the $T_1 - T_5$ and $T_2 - T_6$ edges from the layer topology.** The true edges were used as input to BGRN.

| | | | | eMSE | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| | | | | $\beta = 0.2$ | | $\beta = 0.5$ | | $\beta = 1$ | |
| Topology | Type CV | N | edge | mean | sd | mean | sd | mean | sd |
| Layer | Common parent | 200 | T1-T5 | 0.2632 | 0.1408 | 0.0128 | 0.0437 | 0 | 0 |
| | | | T2-T6 | 0.1757 | 0.0723 | 0.0574 | 0.101 | 0 | 0 |
| | | 500 | T1-T5 | 0.1191 | 0.1379 | 0 | 0 | 0 | 0 |
| | | | T2-T6 | 0.2403 | 0.2633 | 0.0096 | 0.041 | 0 | 0 |
| | Intermediate | 200 | T1-T5 | 0.2435 | 0.149 | 0.0264 | 0.0874 | 0 | 0 |
| | | | T2-T6 | 0.2096 | 0.1001 | 0.0846 | 0.1113 | 0 | 0 |
| | | 500 | T1-T5 | 0.1323 | 0.1201 | 0.0027 | 0.0124 | 0 | 0 |
| | | | T2-T6 | 0.2226 | 0.2229 | 0.1302 | 0.2662 | 0 | 0 |

Table 3.6: **The mean and standard deviation of the eMSE for the $T_1 - T_2$ edge in topology M3.** The input to BGRN was a fully connected graph.

| | | | | eMSE | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| | | | | $\beta = 0.2$ | | $\beta = 0.5$ | | $\beta = 1$ | |
| Topology | Type CV | $N$ | # CV | mean | sd | mean | sd | mean | sd |
| M3 | Common parent | 200 | 2 | 0.0248 | 0.0459 | 0.0133 | 0.017 | 0.0121 | 0.0179 |
| | | | 5 | 0.1102 | 0.1236 | 0.1154 | 0.1398 | 0.1397 | 0.1697 |
| | | | 10 | 0.4177 | 0.1684 | 0.3978 | 0.1691 | 0.2591 | 0.1805 |
| | | 500 | 2 | 0.0091 | 0.019 | 0.0101 | 0.0217 | 0.0356 | 0.0872 |
| | | | 5 | 0.2607 | 0.2106 | 0.2342 | 0.2098 | 0.2774 | 0.2344 |
| | | | 10 | 0.4858 | 0.0984 | 0.5041 | 0.0266 | 0.4876 | 0.06 |
| | Common child | 200 | 2 | 0.0171 | 0.0233 | 0.0112 | 0.0131 | 0.0079 | 0.006 |
| | | | 5 | 0.0402 | 0.0463 | 0.0179 | 0.0202 | 0.034 | 0.0462 |
| | | | 10 | 0.0914 | 0.1455 | 0.0662 | 0.0705 | 0.0589 | 0.0999 |
| | | 500 | 2 | 0.0224 | 0.0437 | 0.0226 | 0.0619 | 0.0155 | 0.0248 |
| | | | 5 | 0.0586 | 0.0425 | 0.015 | 0.0125 | 0.0372 | 0.0636 |
| | | | 10 | 0.0779 | 0.1201 | 0.0211 | 0.0297 | 0.0211 | 0.0233 |
| | Intermediate | 200 | 2 | 0.0153 | 0.021 | 0.034 | 0.0865 | 0.011 | 0.0135 |
| | | | 5 | 0.0968 | 0.146 | 0.0303 | 0.0507 | 0.0233 | 0.0733 |
| | | | 10 | 0.1223 | 0.1826 | 0.1279 | 0.1721 | 0.0202 | 0.0338 |
| | | 500 | 2 | 0.0352 | 0.0748 | 0.0104 | 0.0288 | 0.0316 | 0.0893 |
| | | | 5 | 0.043 | 0.0804 | 0.0135 | 0.0209 | 0.025 | 0.053 |
| | | | 10 | 0.0737 | 0.13 | 0.0441 | 0.0894 | 0.0334 | 0.099 |

### 3.3.4 Genes causal to a single clinical phenotype

For the clinical phenotype simulations we ran BGRN with two different inputs for each topology. For the first input, we used the true edges for the inner graph (the edges of the genetic variants and genes) and all of the edges to the clinical phenotype. For the second input, we only considered all possible edges to the clinical phenotype. We refer to the second input as the multi-parent input. When we used the multi-parent input to BGRN we ran it for 20,000 iterations with a step size of 32 (unless stated otherwise). Table 3.7 shows the number of iterations and the step size for each size of topology considered when the true edges plus all the edges to the clinical phenotype were used as the input.

Table 3.7: **The number of MCMC iterations and step size according to the number of nodes in the network.**

| # nodes | Iterations | Step size |
|---|---|---|
| 5-6 | 20,000 | 32 |
| 7-8 | 50,000 | 80 |
| 9-10 | 75,000 | 120 |
| 13 | 100,000 | 160 |

In the following simulation studies, we focus on networks with one clinical phenotype node. We explore the effect of topology on identifying trait related genes. We examine how the number of true edges and the correlation among genes affect the inference of trait related genes. We demonstrate the difficulty in choosing an appropriate prior when dealing with densely connected networks. We also compare BGRN with the widely used variable selection method Least Absolute Shrinkage and Selection Operator (LASSO) on networks with a high correlation among genes.

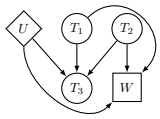
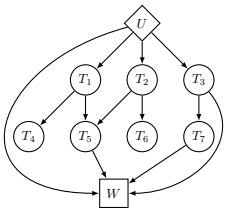
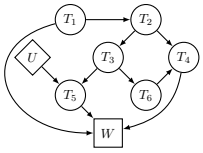
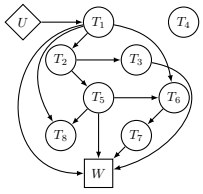
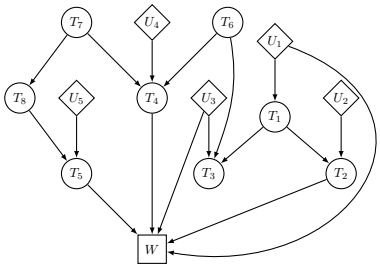
#### 3.3.4.1 Effect of topology on identifying trait related genes

The inference of the clinical phenotype (i.e., $W$) edges is unaffected by other edges in the network. In other words, if all possible edges for a given topology are considered in the inference or if only the edges to $W$ are considered, the posterior probabilities for the $W$ edges will be nearly identical between the two analyses. This is due to two factors. First, we place a constraint on the network such that the clinical phenotype is always the child of other variable types. Second, the likelihood of a network can be written as a product of conditional probabilities [74, 40] where each node is conditioned on its parents:

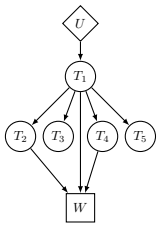
$$\Pr(\mathbf{X} \mid \mathbf{S}, \boldsymbol{\theta}) = \prod_{j=1}^{b} \Pr(X_j \mid pa(X_j), \boldsymbol{\theta}_j), \tag{3.6}$$

where $pa(X_j)$ are the parents of $X_j$. Therefore, the likelihood for the clinical phenotype is the highest, given a sufficient signal in the data, when all of the true edges to $W$ are present and the clinical phenotype will contribute the same amount to the likelihood of the entire graph no matter which input is used. This is the case regardless of which other edges are included in the inference. Table 3.8 shows the average difference in the posterior probability for each edge between the two inputs when the sample size is 500. The average difference is calculated across the three edge states for each clinical phenotype edge. The largest difference between posterior probabilities for the two inputs is near 0.05 and many of the differences are near zero (Table 3.8 and Figure 3.8) across all three signal strength values. Such a small difference between the posterior probabilities of the two inputs shows that, because of the constraint placed on the clinical phenotype node, the addition of other edges in the input does not affect the inference of the clinical phenotype edges. Therefore, when the clinical phenotype edges are the focus of the inference the input does not need to include the edges in the inner graph (i.e., edges between other variable types) which greatly reduces the search space and runtime.

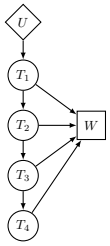
Table 3.8: **The average posterior probability difference for the $W$ edges.** The difference is calculated from the posterior probability for each $W$ edge between the two inputs to BGRN. The rows in gray indicate true edges.

| | | Posterior Probability Difference | | | | | |
|---|---|---|---|---|---|---|---|
| | | $\beta = 0.2$ | | $\beta = 0.5$ | | $\beta = 1$ | |
| Topology | Edge | mean | sd | mean | sd | mean | sd |
|  | U-W | 0.0135 | 0.0134 | 0.0075 | 0.0128 | 6e-04 | 0.0022 |
| | T1-W | 0.0144 | 0.0152 | 0.0011 | 0.0018 | 0 | 0 |
| | T2-W | 0.0095 | 0.0085 | 0.0015 | 0.0041 | 0 | 0 |
| | T3-W | 0.0101 | 0.0069 | 0.0109 | 0.0082 | 0.0089 | 0.008 |
|  | U-W | 0.0142 | 0.0117 | 0.0079 | 0.0087 | 0.0101 | 0.0086 |
| | T1-W | 0.0086 | 0.0063 | 0.0121 | 0.0106 | 0.0115 | 0.0082 |
| | T2-W | 0.0136 | 0.0107 | 0.0126 | 0.0102 | 0.0111 | 0.0065 |
| | T3-W | 0.0122 | 0.0115 | 0.0043 | 0.0061 | 0.0052 | 0.007 |
| | T4-W | 0.0141 | 0.0096 | 0.0141 | 0.0089 | 0.0142 | 0.0094 |
| | T5-W | 0.0094 | 0.0106 | 7e-04 | 0.0015 | 6e-04 | 0.002 |
| | T6-W | 0.0114 | 0.0065 | 0.0097 | 0.0082 | 0.0126 | 0.0104 |
| | T7-W | 0.0092 | 0.0086 | 0.0033 | 0.005 | 5e-04 | 0.0015 |
|  | U-W | 0.0111 | 0.0103 | 0.0105 | 0.0106 | 0.0125 | 0.0107 |
| | T1-W | 0.0123 | 0.0082 | 0.0024 | 0.0059 | 0.002 | 0.0039 |
| | T2-W | 0.0126 | 0.0085 | 0.0089 | 0.0059 | 0.013 | 0.0106 |
| | T3-W | 0.011 | 0.0081 | 0.0106 | 0.0076 | 0.0096 | 0.0085 |
| | T4-W | 0.0117 | 0.0091 | 1e-04 | 3e-04 | 0 | 0 |
| | T5-W | 0.009 | 0.0083 | 6e-04 | 0.002 | 1e-04 | 3e-04 |
| | T6-W | 0.0086 | 0.006 | 0.0118 | 0.0078 | 0.0127 | 0.0112 |
|  | U-W | 0.0154 | 0.011 | 0.0159 | 0.0102 | 0.0106 | 0.0075 |
| | T1-W | 0.013 | 0.0119 | 0.0058 | 0.0076 | 0.0089 | 0.009 |
| | T2-W | 0.0113 | 0.0075 | 0.0175 | 0.0133 | 0.0142 | 0.0094 |
| | T3-W | 0.0157 | 0.016 | 0.0014 | 0.0034 | 1e-04 | 4e-04 |
| | T4-W | 0.0124 | 0.0106 | 0.0109 | 0.0067 | 0.0101 | 0.0074 |
| | T5-W | 0.0146 | 0.0097 | 0.0029 | 0.0047 | 0.0091 | 0.008 |
| | T6-W | 0.0124 | 0.0104 | 0.0116 | 0.0095 | 0.0115 | 0.0086 |
| | T7-W | 0.0104 | 0.0083 | 0.0011 | 0.0032 | 4e-04 | 9e-04 |
| | T8-W | 0.0139 | 0.0106 | 0.0119 | 0.0097 | 0.0119 | 0.0098 |
|  | U1-W | 0.0129 | 0.0093 | 0.0116 | 0.0104 | 0.0135 | 0.009 |
| | U2-W | 0.0108 | 0.0085 | 0.0115 | 0.0094 | 0.0143 | 0.0128 |
| | U3-W | 0.0145 | 0.0136 | 0.0117 | 0.0132 | 0.0108 | 0.0097 |
| | U4-W | 0.0084 | 0.0069 | 0.0086 | 0.0085 | 0.0094 | 0.0064 |
| | U5-W | 0.0158 | 0.0097 | 0.0105 | 0.0068 | 0.0099 | 0.0069 |
| | T1-W | 0.0121 | 0.0086 | 0.0119 | 0.0065 | 0.0156 | 0.0114 |
| | T2-W | 0.0149 | 0.0145 | 0.0012 | 0.0018 | 6e-04 | 0.0022 |
| | T3-W | 0.0154 | 0.0117 | 0.0089 | 0.0087 | 0.0113 | 0.0105 |
| | T4-W | 0.0128 | 0.0114 | 0.0025 | 0.0056 | 0.0014 | 0.0041 |
| | T5-W | 0.0151 | 0.0145 | 0.0012 | 0.0028 | 0.0012 | 0.0026 |
| | T6-W | 0.0114 | 0.0112 | 0.0125 | 0.0094 | 0.0099 | 0.0078 |
| | T7-W | 0.0149 | 0.0093 | 0.0113 | 0.008 | 0.0121 | 0.0092 |
| | T8-W | 0.014 | 0.0087 | 0.0155 | 0.0114 | 0.0135 | 0.0107 |

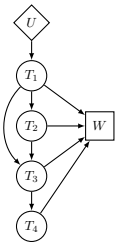
Topology




| | Posterior Probability Difference | | | | | |
|---|---|---|---|---|---|---|
| | $\beta = 0.2$ | | $\beta = 0.5$ | | $\beta = 1$ | |
| Edge | mean | sd | mean | sd | mean | sd |
| U-W | 0.0121 | 0.0101 | 0.0091 | 0.0081 | 0.0118 | 0.0088 |
| T1-W | 0.0178 | 0.0156 | 0.0025 | 0.0029 | 0.0084 | 0.0108 |
| T2-W | 0.0141 | 0.0135 | 0.002 | 0.0036 | 1e-04 | 4e-04 |
| T3-W | 0.0104 | 0.0076 | 0.0105 | 0.008 | 0.013 | 0.0076 |
| T4-W | 0.0112 | 0.0096 | 0.0013 | 0.0032 | 0 | 0 |
| T5-W | 0.0122 | 0.0094 | 0.0101 | 0.0074 | 0.0119 | 0.0106 |




| | Posterior Probability Difference | | | | | |
|---|---|---|---|---|---|---|
| | $\beta = 0.2$ | | $\beta = 0.5$ | | $\beta = 1$ | |
| Edge | mean | sd | mean | sd | mean | sd |
| U-W | 0.0125 | 0.0083 | 0.0117 | 0.0099 | 0.0129 | 0.0097 |
| T1-W | 0.0121 | 0.0106 | 0.0022 | 0.003 | 0.0067 | 0.0084 |
| T2-W | 0.0091 | 0.0073 | 0.0025 | 0.0032 | 0.0054 | 0.0076 |
| T3-W | 0.0074 | 0.0076 | 0.0083 | 0.0144 | 0.0045 | 0.0079 |
| T4-W | 0.012 | 0.0099 | 0.0019 | 0.0049 | 5e-04 | 9e-04 |




| | Posterior Probability Difference | | | | | |
|---|---|---|---|---|---|---|
| | $\beta = 0.2$ | | $\beta = 0.5$ | | $\beta = 1$ | |
| Edge | mean | sd | mean | sd | mean | sd |
| U-W | 0.0116 | 0.0116 | 0.0119 | 0.0098 | 0.0157 | 0.0139 |
| T1-W | 0.0108 | 0.0114 | 0.0095 | 0.0129 | 0.0099 | 0.0118 |
| T2-W | 0.0103 | 0.0117 | 0.0058 | 0.008 | 0.0082 | 0.0091 |
| T3-W | 0.0131 | 0.0115 | 0.0036 | 0.0086 | 0.0098 | 0.0117 |
| T4-W | 0.0147 | 0.0102 | 0.003 | 0.0074 | 0.0033 | 0.0052 |

Figure 3.4: **The average posterior probability difference for the $W$ edges.** The difference is calculated between the posterior probability of the three edge states for each $W$ edge when the input contains the edges in the inner graph in addition to all the $W$ edges and when the input contains just the edges to $W$. The difference between the posterior for each edge is then averaged across the three edge states. The average difference for each edge is then averaged across the 25 data sets simulated for each combination of topology, signal strength, and a sample size of 500. The topologies shown on the left are the graphs used to generate the data. The rows in gray indicate the true edges.

#### 3.3.4.2 Effect of correlation and number of parents on identifying trait related genes

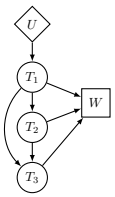
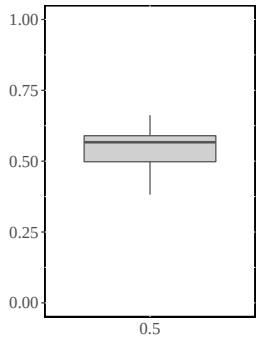
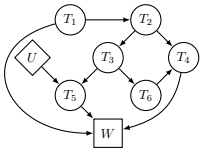
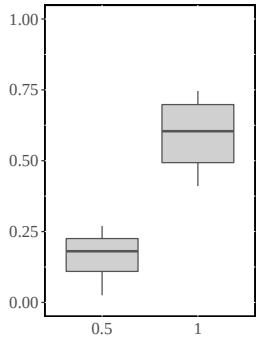
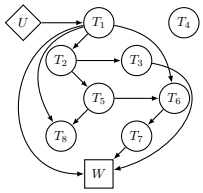
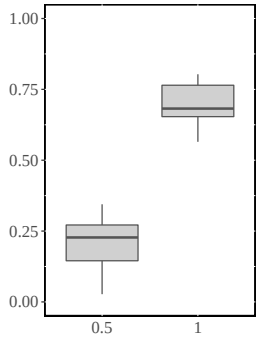
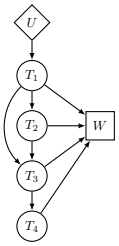
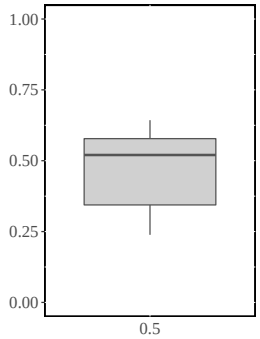
Here we use the edgewise power to measure the inference accuracy for specific edges in the network. The edgewise power is calculated across the inferred network from multiple independent data sets from

the same topology and is defined by:

$$\text{Edgewise power} = \frac{\#\text{ of times edge is inferred}}{\#\text{ of independent data sets}}. \tag{3.7}$$

When calculating the edgewise power, we use 0.4 as the cutoff for edge presence (unless otherwise stated). We consider the correlation to be low if all the correlations are in the 0.70s or lower and we consider the correlation to be high if all the correlations are in the 0.70s or higher. In this section when we refer to the correlation we only consider the correlation among the true parents of the clinical phenotype. For this simulation study we also take into account the number of parents the clinical phenotype has. For example, we consider the clinical phenotype to have few parents if it has less than four true parents. We consider the clinical phenotype to have many parents if it has more than four true parents.

BGRN can accurately identify trait related genes for both low and high correlation values depending on the number of true parents the trait has. The edgewise power for the true edges is near one when the correlation is low among the true parents of the trait (Figures 3.5 - 3.6). This is true whether or not the trait has few or many true parents. In addition, the edgewise power for most of the false edges is near zero (Figures 3.5 - 3.6). On the other hand, when the correlation is high inference accuracy depends on the number of true parents the trait has. For example, when the trait only has a few true parents the edgewise power is near one for the true edges and near zero for the false edges (Figure 3.7). However, when the signal strength is 1.5 for the chain 3 topology the edgewise power for some true edges is much lower (Figure 3.7). As the number of true parents increases the edgewise power for the true edges decreases (Figure 3.8). In summary, BGRN can correctly identify trait related genes when the correlation among them is low, but when the correlation among the trait related genes is high the performance of BGRN depends on the number of true parents the trait has.

| | Topology | Edgewise power | Correlation of true parents |
|---|---|---|---|
| Feed forward loop 4 |  | |  |
| GN6 |  | |  |
| GN8 |  | |  |
| Feed forward loop 4 |  | |  |

| Edge | $\beta = 0.5$ |
|---|---|
| U-W | 0.04 |
| T1-W | 1 |
| T2-W | 1 |
| T3-W | 1 |

| Edge | $\beta = 0.5$ | $\beta = 1$ |
|---|---|---|
| U-W | 0.08 | 0 |
| T1-W | 1 | 1 |
| T2-W | 0 | 0.04 |
| T3-W | 0.04 | 0.08 |
| T4-W | 1 | 1 |
| T5-W | 1 | 1 |
| T6-W | 0.12 | 0.08 |

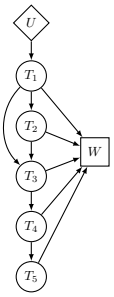
| Edge | $\beta = 0.5$ | $\beta = 1$ |
|---|---|---|
| U-W | 0 | 0.04 |
| T1-W | 1 | 0.96 |
| T2-W | 0.08 | 0 |
| T3-W | 1 | 1 |
| T4-W | 0 | 0.08 |
| T5-W | 0.04 | 0.04 |
| T6-W | 0.08 | 0.04 |
| T7-W | 1 | 1 |
| T8-W | 0.04 | 0.12 |

| Edge | $\beta = 0.5$ |
|---|---|
| U-W | 0.12 |
| T1-W | 1 |
| T2-W | 0.96 |
| T3-W | 1 |
| T4-W | 1 |

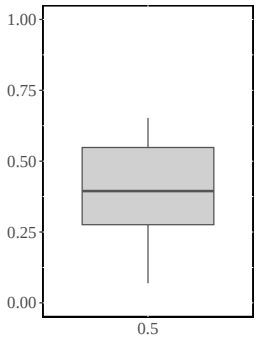
Figure 3.5: **Edgewise power tables and box plots of the correlation between continuous variables.** The topologies in the left column depict the true graphs that generated the data. The middle column shows the edgewise power for all $W$ edges in the network. Rows for the true edges are highlighted in gray. The right column shows the box plots for the correlation between the true parents of $W$.

| | Topology | Edgewise power | Correlation of true parents |
|---|---|---|---|

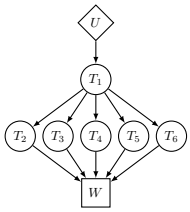
Feed forward loop 5



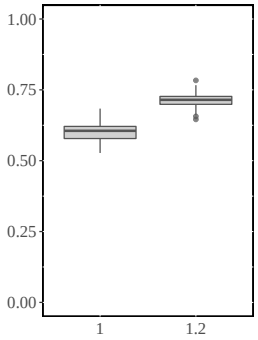

| Edge | $\beta = 0.5$ |
|---|---|
| U-W | 0.04 |
| T1-W | 0.96 |
| T2-W | 0.92 |
| T3-W | 0.96 |
| T4-W | 1 |
| T5-W | 1 |



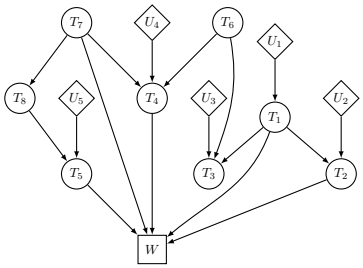

Star




| Edge | $\beta = 1$ | $\beta = 1.2$ |
|---|---|---|
| U-W | 0.12 | 0.12 |
| T1-W | 0.04 | 0.16 |
| T2-W | 1 | 1 |
| T3-W | 1 | 1 |
| T4-W | 1 | 0.96 |
| T5-W | 1 | 1 |
| T6-W | 1 | 1 |




GN13




| Edge | $\beta = 0.5$ | $\beta = 1$ |
|---|---|---|
| U1-W | 0 | 0 |
| U2-W | 0.08 | 0.04 |
| U3-W | 0.04 | 0.08 |
| U4-W | 0.12 | 0.2 |
| U5-W | 0.08 | 0.08 |
| T1-W | 0.92 | 0.96 |
| T2-W | 1 | 1 |
| T3-W | 0.12 | 0.04 |
| T4-W | 1 | 1 |
| T5-W | 1 | 1 |
| T6-W | 0.08 | 0.12 |
| T7-W | 0.96 | 0.88 |
| T8-W | 0.08 | 0.16 |

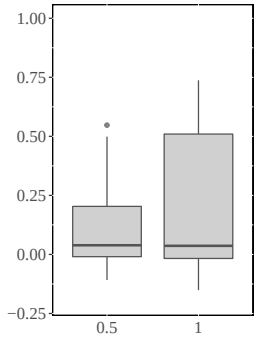
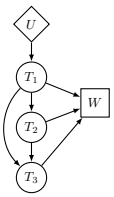
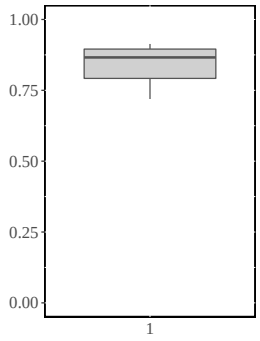
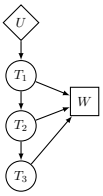
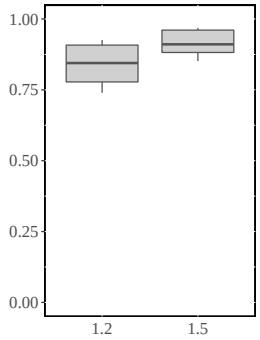


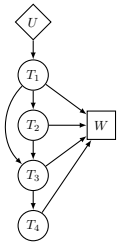
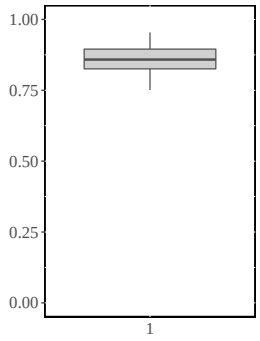
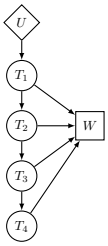
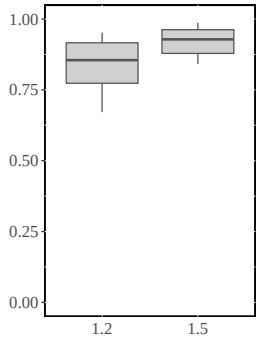
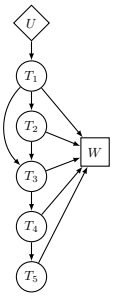
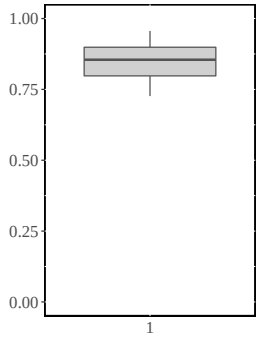
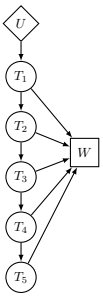
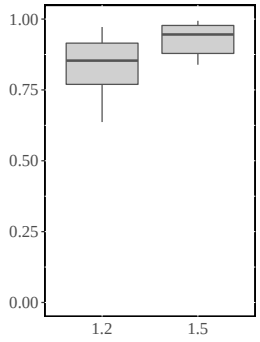

Figure 3.6: **Edgewise power tables and box plots of the correlation between continuous variables.** The topologies in the left column depict the true graphs that generated the data. The middle column shows the edgewise power for all $W$ edges in the network. Rows for the true edges are highlighted in gray. The right column shows the box plots for the correlation between the true parents of $W$.

| | Topology | Edgewise power | Correlation of true parents |
|---|---|---|---|
| Feed forward loop 3 |  | |  |
| Chain 3 |  | |  |

| Edge | $\beta = 1$ |
|---|---|
| U-W | 0 |
| T1-W | 0.96 |
| T2-W | 0.96 |
| T3-W | 1 |

| Edge | $\beta = 1.2$ | $\beta = 1.5$ |
|---|---|---|
| U-W | 0 | 0.08 |
| T1-W | 0.96 | 0.84 |
| T2-W | 0.88 | 0.64 |
| T3-W | 1 | 1 |

Figure 3.7: **Edgewise power tables and box plots of the correlation between continuous variables.** The topologies in the left column depict the true graphs that generated the data. The middle column shows the edgewise power for all $W$ edges in the network. Rows for the true edges are highlighted in gray. The right column shows the box plots for the correlation between the true parents of $W$.

| | Topology | Edgewise power | Correlation of true parents |
|---|---|---|---|
| Feed forward loop 4 |  | |  |
| Chain 4 |  | |  |
| Feed forward loop 5 |  | |  |
| Chain 5 |  | |  |

| Edge | $\beta = 1$ |
|---|---|
| U-W | 0.04 |
| T1-W | 0.84 |
| T2-W | 0.76 |
| T3-W | 0.92 |
| T4-W | 1 |

| Edge | $\beta = 1.2$ | $\beta = 1.5$ |
|---|---|---|
| U-W | 0.04 | 0.12 |
| T1-W | 0.92 | 0.6 |
| T2-W | 0.72 | 0.72 |
| T3-W | 0.88 | 0.44 |
| T4-W | 1 | 0.96 |

| Edge | $\beta = 1$ |
|---|---|
| U-W | 0.04 |
| T1-W | 0.68 |
| T2-W | 0.68 |
| T3-W | 0.84 |
| T4-W | 72 |
| T5-W | 0.96 |

| Edge | $\beta = 1.2$ | $\beta = 1.5$ |
|---|---|---|
| U-W | 0.08 | 0.08 |
| T1-W | 0.64 | 0.4 |
| T2-W | 0.76 | 0.64 |
| T3-W | 0.76 | 0.64 |
| T4-W | 0.68 | 0.56 |
| T5-W | 0.88 | 0.8 |

Figure 3.8: **Edgewise power tables and box plots of the correlation between continuous variables.** The topologies in the left column depict the true graphs that generated the data. The middle column shows the edgewise power for all $W$ edges in the network. Rows for the true edges are highlighted in gray. The right column shows the box plots for the correlation between the true parents of $W$.

#### 3.3.4.3 Method comparison for identifying trait related genes with high correlation

We compare BGRN with LASSO on networks with few true trait related genes and high correlation among all genes in the network. LASSO is widely used as a variable selection tool. However, it is unclear how it performs in scenarios with high correlation. For this simulation study we generated data under two main topologies (Figure 3.9) with high correlation among all the genes. In these topologies only two genes in the network are the true parents of the clinical phenotype. For this analysis we used the version of LASSO implemented in the glmnet [23] R package.

We focus on the feed forward loop and chain topologies (Table 3.10) which have a high correlation between the gene expression variables in the inner graph (the non-clinical phenotype variables). To create a high correlation among the gene expression variables for the chain topologies we increased the signal strength to 1.2 and 1.5 between the variables in the inner graph. Even though the signal strength is above one for the inner graph, we keep the signal strength for the parents of the clinical phenotype at one. The structure of the feed forward loop topologies creates a high correlation among the gene expression nodes without increasing the signal strength above one, therefore, we use a value of one for the signal strength for all feed forward loop topologies in this section. For each topology we simulated 25 independent data sets for each combination of topology and signal strength and used a sample size of 500.

We ran BGRN for 10,000 iterations on each topology with a burn-in of 20% and a step size of 16. We used the prior of $(p_1, p_2, p_3) = (0.05, 0.05, 0.9)$ and we consider an edge present if the posterior probability for edge presence is $> 0.4$. When analyzing the data using LASSO we selected values for the shrinkage penalty parameter $\lambda$ using the cross validation function [23]. This function returns two $\lambda$ values. The first, $\lambda.min$, is the $\lambda$ value that has the lowest MSE across the cross validation sets. The second, $\lambda.1se$, is the largest $\lambda$ value that has an MSE within one standard deviation of $\lambda.min$ and creates a model with the fewest variables. For models inferred by LASSO we consider an edge present if the $\beta$ coefficient of the variable is not zero. For BGRN we only consider the edges from the genetic variant and gene expression variables to the clinical phenotype node and for LASSO the clinical phenotype is the response variable with all other variables considered as predictor variables.

We compare the performance of BGRN and LASSO with the edgewise power (Equation 3.7) for all clinical phenotype edges. For BGRN the edgewise power for the true edges is nearly one in all scenarios (Table 3.9), one exception is the $T_2 - W$ edge in the chain topology when the signal strength is 1.5 (Table 3.9). The edgewise power of the true edges for LASSO is one in all scenarios (Table 3.9). On the other hand, the edgewise power for the false edges is near zero for BGRN. Again, the exception is for the chain

topology with a signal strength of 1.5 where the edgewise power of the $T_5 - W$ edge is 0.32. When using $\lambda.min$ from LASSO for variable selection the edgewise power for all the false edges is high (Table 3.9). In other words, LASSO infers false edges as present most of the time. The edgewise power decreases for the false edges when the model is inferred with $\lambda.1se$, but it can still be high ($> 0.5$) for some edges. Overall, BGRN outperforms LASSO in identifying trait related genes when the correlation is high between genes as LASSO tends to infer far more false edges than BGRN.

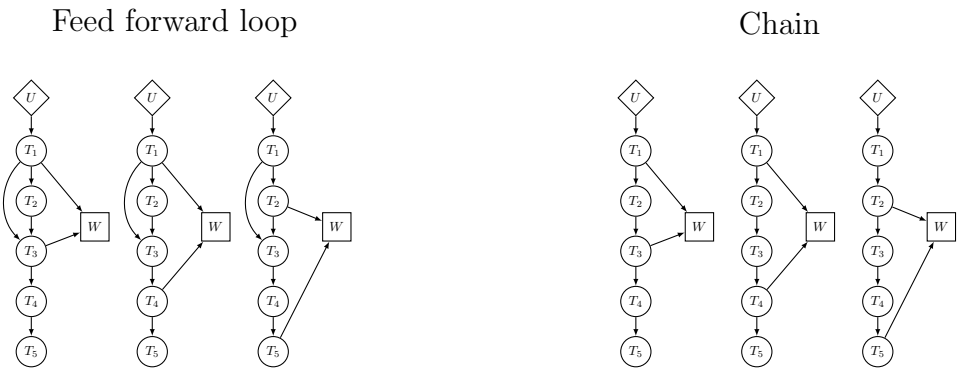



Figure 3.9: **Six networks used in method comparison for identifying trait related genes.** Both the feed forward loop and chain topologies have high correlation among the genes.

Table 3.9: **The edgewise power for BGRN and LASSO for the feed forward loop and chain topologies.** The rows highlighted in gray are the true edges.

| | | Edgewise power | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| | | $T_1, T_3$ | | | $T_1, T_4$ | | | $T_2, T_5$ | | |
| | | | LASSO | | | LASSO | | | LASSO | |
| Topology | Edge | BGRN | $\lambda$.min | $\lambda$.1se | BGRN | $\lambda$.min | $\lambda$.1se | BGRN | $\lambda$.min | $\lambda$.1se |
| Feed forward loop | U-W | 0 | 0.52 | 0 | 0.04 | 0.68 | 0.12 | 0.08 | 0.64 | 0.16 |
| | T1-W | 0.96 | 1 | 1 | 1 | 1 | 1 | 0.12 | 0.36 | 0.12 |
| | T2-W | 0 | 0.6 | 0.28 | 0.08 | 0.56 | 0.12 | 1 | 1 | 1 |
| | T3-W | 1 | 1 | 1 | 0.04 | 0.52 | 0.6 | 0.12 | 0.44 | 0.4 |
| | T4-W | 0.08 | 0.36 | 0.16 | 1 | 1 | 1 | 0.08 | 0.36 | 0.4 |
| | T5-W | 0.04 | 0.48 | 0.16 | 0.04 | 0.52 | 0.44 | 1 | 1 | 1 |
| Chain $\beta = 1.2$ | U-W | 0.16 | 0.68 | 0.16 | 0.08 | 0.72 | 0.16 | 0 | 0.72 | 0.08 |
| | T1-W | 1 | 1 | 1 | 1 | 1 | 1 | 0.12 | 0.52 | 0.32 |
| | T2-W | 0.04 | 0.6 | 0.4 | 0.12 | 0.48 | 0.28 | 0.92 | 1 | 1 |
| | T3-W | 1 | 1 | 1 | 0 | 0.64 | 0.52 | 0.12 | 0.6 | 0.6 |
| | T4-W | 0 | 0.32 | 0.08 | 1 | 1 | 1 | 0.04 | 0.28 | 0.24 |
| | T5-W | 0.08 | 0.36 | 0.2 | 0.04 | 0.6 | 0.44 | 1 | 1 | 1 |
| Chain $\beta = 1.5$ | U-W | 0.04 | 0.56 | 0.16 | 0.04 | 0.6 | 0.2 | 0.04 | 0.8 | 0.16 |
| | T1-W | 0.92 | 1 | 1 | 0.84 | 1 | 1 | 0.08 | 0.56 | 0.44 |
| | T2-W | 0.12 | 0.64 | 0.52 | 0.16 | 0.6 | 0.56 | 0.68 | 1 | 1 |
| | T3-W | 0.96 | 1 | 1 | 0.08 | 0.56 | 0.48 | 0.08 | 0.2 | 0.24 |
| | T4-W | 0.08 | 0.44 | 0.32 | 0.92 | 1 | 1 | 0.04 | 0.48 | 0.48 |
| | T5-W | 0.16 | 0.48 | 0.4 | 0.32 | 0.52 | 0.56 | 1 | 1 | 1 |

Table 3.10: **The Pearson correlation between the continuous variables for the feed forward loop and chain topologies.**

| | Correlation | | | | | |
|---|---|---|---|---|---|---|
| Topology | Min. | 1st Qu. | Median | Mean | 3rd Qu. | Max. |
| Feed forward loop | 0.7200 | 0.8000 | 0.8500 | 0.8529 | 0.9000 | 0.9600 |
| Chain $\beta = 1.2$ | 0.6500 | 0.7700 | 0.8500 | 0.8461 | 0.9200 | 0.9700 |
| Chain $\beta = 1.5$ | 0.8400 | 0.8800 | 0.9400 | 0.9307 | 0.9800 | 0.9900 |

#### 3.3.4.4 Comparison of priors for traits with many parents

In this simulation study we compare the effect of different prior probabilities on inference accuracy for densely connected networks. We focus on the feed forward loop (FFL) topology where all genes are the parents of the clinical phenotype. Again, we chose this topology because it produces high correlation among the genes. We compare the inference from four different priors: prior 1 $(p_1, p_2, p_3) = (1/3, 1/3, 1/3)$, prior 2 $(p_1, p_2, p_3) = (0.25, 0.25, 0.5)$, prior 3 $(p_1, p_2, p_3) = (0.2, 0.2, 0.6)$, and the default

prior $(p_1, p_2, p_3) = (0.05, 0.05, 0.9)$. For each prior, we ran BGRN once per data set for 10,000 iterations, used a burn-in of 20%, a step size of 16, and used the multi-parent input which consists of just the clinical phenotype edges. To assess the performance of the different priors we calculate the edgewise power (Equation 3.7) and eMSE (see Section 2.3.2). For priors 1-3 we consider an edge present if the posterior probability of edge presence is $> p_1 + p_2 + 0.2$. For example, if we are using prior 2 then we would consider an edge present if the posterior for edge presence is $> 0.25 + 0.25 + 0.2 = 0.7$. When using the default prior we consider an edge present if the posterior for edge presence is $> 0.4$.

There is no clear choice of which prior to use when dealing with densely connected graphs that have a high correlation among genes. This arises from the need to balance identifying true and false edges and the fact that when the correlation is high there are multiple graph structures that produce a similarly high likelihood. For example, when the signal strength is one, the edgewise power for the FFL 5 topology is similar for the true edges when either prior 1 or the default prior are used (Table 3.11). This is counter intuitive because the default prior heavily favors edge absence while prior 1 favors edge presence. However, this occurs because graphs where all true edges are present have a similar likelihood to graphs with one or two true edges missing. Therefore, when the Markov chain is at the true graph and a new graph is proposed with one or two true parents missing the proposed graph will be accepted often. In other words, even if the prior probability favors edge presence the acceptance ratio (Algorithm 2) is small enough that the proposed graph will be accepted often, reducing the edgewise power for the true edges.

This is less of a problem when there are fewer genes in the network or when the correlation among genes is lower. For example, the edgewise power for the true edges for prior 1 is higher than for the default prior for topology FFL 4 with a signal strength of 1 (Table 3.11). In addition, when the correlation between genes is lower (e.g., when the signal strength is 0.5) the default prior outperforms other priors (Table 3.11) as the edgewise power is near one for the true edges and the edgewise power for the false edge is lower than for the other priors.

It is also unclear what prior is best when using the eMSE to compare performance. The eMSE is the lowest, for the true edges, when using prior 1 (Table 3.12). This is the case because the posterior probability for edge presence will be closer to one, for all edges, which is the expected probability for the true edges. On the other hand, the eMSE is the highest for the false edges when using prior 1. Again, this occurs because the posterior probability for edge presence is high for the false edges when using prior 1 and the expected probability for edge presence for these edges is zero. Furthermore, the eMSE for the true edges is higher when using the default prior (Table 3.12) as the posterior probability for edge presence will be smaller (closer to zero). For this same reason the eMSE for the false edges is the smallest when using the default prior. Priors 2 and 3 have a low eMSE for true edges, but not as low as prior 1

(Table 3.12). Likewise, the eMSE is lower for the false edges when using priors 2 and 3, but not as low as the default prior. Therefore, priors 2 and 3 do not offer a favorable compromise between prior 1 and the default prior.

In summary, both the edgewise power and eMSE show that there is not one prior that performs best when a trait has many parents and the correlation among them is high. The default prior performs well at not inferring the false edges as present. However, it does not perform as well for either metric in identifying the true edges. At the opposite extreme is prior 1 which highly favors edge presence. This prior performs well at finding the true edges when using the eMSE to measure inference accuracy. However, when using the edgewise power as the metric, the performance for prior 1 is similar to that of the default prior. Prior 1 also performs poorly at identifying false edges when using both metrics to measure inference accuracy. Priors 2 and 3 show similar trends to prior 1 in that they perform well at identifying the true edges but their performance is poor when it comes to identifying false edges.

Table 3.11: **The edgewise power for the feed forward loop 4 and 5 topologies for four different priors on edge states.** The true edges are highlighted in gray.

| | | Edgewise power | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | prior 1 | | | prior 2 | | | prior 3 | | | default | | |
| | Edge | 0.2 | 0.5 | 1 | 0.2 | 0.5 | 1 | 0.2 | 0.5 | 1 | 0.2 | 0.5 | 1 |
| FFL 4 | U-W | 0.16 | 0.2 | 0.16 | 0.24 | 0.36 | 0.28 | 0.24 | 0.36 | 0.32 | 0.04 | 0.12 | 0.04 |
| | T1-W | 0.84 | 1 | 1 | 0.84 | 1 | 1 | 0.84 | 1 | 0.96 | 0.84 | 1 | 0.84 |
| | T2-W | 0.88 | 0.96 | 0.84 | 0.92 | 1 | 0.92 | 0.92 | 1 | 0.96 | 0.88 | 0.96 | 0.76 |
| | T3-W | 0.76 | 1 | 0.96 | 0.84 | 1 | 0.96 | 0.84 | 1 | 0.96 | 0.68 | 1 | 0.92 |
| | T4-W | 0.68 | 1 | 1 | 0.72 | 1 | 1 | 0.76 | 1 | 1 | 0.48 | 1 | 1 |
| FFL 5 | U-W | 0.08 | 0.08 | 0.08 | 0.24 | 0.08 | 0.12 | 0.2 | 0.08 | 0.2 | 0.08 | 0.04 | 0.04 |
| | T1-W | 0.72 | 0.96 | 0.76 | 0.76 | 0.96 | 0.8 | 0.76 | 1 | 0.84 | 0.64 | 0.96 | 0.68 |
| | T2-W | 0.8 | 0.92 | 0.76 | 0.84 | 0.96 | 0.88 | 0.84 | 0.96 | 0.88 | 0.68 | 0.92 | 0.68 |
| | T3-W | 0.84 | 1 | 0.88 | 0.92 | 1 | 0.88 | 0.92 | 1 | 0.88 | 0.72 | 0.96 | 0.84 |
| | T4-W | 0.88 | 1 | 0.8 | 0.92 | 1 | 0.84 | 0.96 | 1 | 0.88 | 0.8 | 1 | 0.72 |
| | T5-W | 0.8 | 1 | 0.96 | 0.92 | 1 | 0.96 | 0.88 | 1 | 0.96 | 0.64 | 1 | 0.96 |

Table 3.12: **The edgewise MSE for the feed forward loop 4 and 5 topologies for four different priors on edge states.** The true edges are highlighted in gray.

| | | Edgewise MSE | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| | | prior 1 | | prior 2 | | prior 3 | | default | |
| | Edge | mean | sd | mean | sd | mean | sd | mean | sd |
| FFL 4 $\beta = 0.2$ | U-W | 0.3833 | 0.0849 | 0.2678 | 0.1082 | 0.1908 | 0.1077 | 0.0231 | 0.0268 |
| | T1-W | 0.011 | 0.0228 | 0.0253 | 0.0498 | 0.0404 | 0.076 | 0.1404 | 0.174 |
| | T2-W | 0.0071 | 0.0167 | 0.0187 | 0.0427 | 0.0311 | 0.0631 | 0.121 | 0.1577 |
| | T3-W | 0.0128 | 0.0214 | 0.0318 | 0.0485 | 0.0489 | 0.0665 | 0.1798 | 0.1642 |
| | T4-W | 0.0136 | 0.021 | 0.0315 | 0.0403 | 0.0531 | 0.0583 | 0.2219 | 0.1901 |
| FFL 4 $\beta = 0.5$ | U-W | 0.4155 | 0.1119 | 0.2994 | 0.1495 | 0.2292 | 0.152 | 0.0617 | 0.1114 |
| | T1-W | 0.001 | 0.0028 | 0.0029 | 0.0075 | 0.0049 | 0.0131 | 0.0391 | 0.0699 |
| | T2-W | 9e-04 | 0.003 | 0.0019 | 0.0061 | 0.0043 | 0.0133 | 0.0337 | 0.0861 |
| | T3-W | 0 | 1e-04 | 3e-04 | 0.0016 | 6e-04 | 0.0028 | 0.009 | 0.0439 |
| | T4-W | 0 | 0 | 1e-04 | 2e-04 | 1e-04 | 3e-04 | 0.0025 | 0.0064 |
| FFL 4 $\beta = 1$ | U-W | 0.3833 | 0.0946 | 0.2539 | 0.109 | 0.1908 | 0.1108 | 0.0281 | 0.0486 |
| | T1-W | 0.0019 | 0.0035 | 0.0074 | 0.014 | 0.0152 | 0.0297 | 0.0937 | 0.1446 |
| | T2-W | 0.0056 | 0.0116 | 0.0138 | 0.0257 | 0.0212 | 0.0383 | 0.1016 | 0.1557 |
| | T3-W | 0.0024 | 0.0079 | 0.0061 | 0.0176 | 0.0103 | 0.0313 | 0.0518 | 0.1196 |
| | T4-W | 0 | 1e-04 | 0 | 1e-04 | 1e-04 | 2e-04 | 0.002 | 0.0058 |
| FFL 5 $\beta = 0.2$ | U-W | 0.3626 | 0.0923 | 0.2415 | 0.1147 | 0.1751 | 0.1118 | 0.0269 | 0.044 |
| | T1-W | 0.016 | 0.0262 | 0.0364 | 0.058 | 0.053 | 0.0791 | 0.1777 | 0.2027 |
| | T2-W | 0.009 | 0.0166 | 0.0242 | 0.0413 | 0.0377 | 0.0562 | 0.156 | 0.1741 |
| | T3-W | 0.0092 | 0.0181 | 0.0219 | 0.0387 | 0.039 | 0.0647 | 0.1519 | 0.1553 |
| | T4-W | 0.005 | 0.0108 | 0.0146 | 0.0279 | 0.0251 | 0.0446 | 0.1123 | 0.1402 |
| | T5-W | 0.0074 | 0.0127 | 0.0215 | 0.0326 | 0.0369 | 0.0557 | 0.1762 | 0.1823 |
| FFL 5 $\beta = 0.5$ | U-W | 0.3599 | 0.0804 | 0.2276 | 0.0957 | 0.1658 | 0.0973 | 0.0209 | 0.0429 |
| | T1-W | 0.0011 | 0.0043 | 0.0039 | 0.0148 | 0.0056 | 0.021 | 0.0343 | 0.0956 |
| | T2-W | 0.002 | 0.007 | 0.0068 | 0.0243 | 0.0105 | 0.0377 | 0.0437 | 0.1224 |
| | T3-W | 3e-04 | 0.0012 | 0.0011 | 0.0052 | 0.0016 | 0.0077 | 0.0143 | 0.0616 |
| | T4-W | 3e-04 | 0.0011 | 7e-04 | 0.0019 | 0.0013 | 0.0034 | 0.0243 | 0.0544 |
| | T5-W | 1e-04 | 2e-04 | 1e-04 | 3e-04 | 2e-04 | 7e-04 | 0.0048 | 0.0156 |
| FFL 5 $\beta = 1$ | U-W | 0.3726 | 0.0925 | 0.2512 | 0.1119 | 0.183 | 0.1236 | 0.0419 | 0.1181 |
| | T1-W | 0.0121 | 0.0223 | 0.0272 | 0.0449 | 0.0451 | 0.0715 | 0.1617 | 0.1911 |
| | T2-W | 0.0076 | 0.0142 | 0.0198 | 0.031 | 0.0346 | 0.051 | 0.1553 | 0.1905 |
| | T3-W | 0.0061 | 0.0144 | 0.0151 | 0.0343 | 0.0233 | 0.0506 | 0.0838 | 0.1467 |
| | T4-W | 0.006 | 0.013 | 0.0169 | 0.0296 | 0.03 | 0.0536 | 0.1339 | 0.1852 |
| | T5-W | 0.0013 | 0.0059 | 0.0034 | 0.0135 | 0.0072 | 0.0312 | 0.0349 | 0.107 |

## 3.4 Discussion

We have presented BGRN which is a Bayesian graphical model based method for inferring gene regulatory networks from individual level data. Our method incorporates genotype data to infer causal relationships among genes from observational data, allows for phenotype data to be included in the

network, and employs assumptions to regulate the relationship between data types. This method is an extension of our previous work which efficiently learns the structure of a network, provides a measure of uncertainty in the inference, and allows for edge-level prior probabilities.

We carried out extensive simulations under a wide range of scenarios to show how BGRN performs when including genotype data in the network and applying the PMR assumption. We show that when using the principle of Mendelian randomization, BGRN has high precision and power for a variety of networks commonly found in biological systems. We demonstrated that BGRN can account for the effect of confounding variables by treating them as nodes in the network. We also identified types of confounding variables that can greatly affect inference accuracy and showed that the number of confounders can also adversely impact the inference. We also investigated how the structure of a gene regulatory network affects the inference of clinical phenotype (or trait) related genes. We demonstrated that the presence of additional edges in the network (e.g., edges between non-clinical phenotype variables) does not affect the inference of the clinical phenotype edges. Therefore, the inference of a network with clinical phenotype variables can be broken into two subgraphs. The first subgraph only considers the relationships in the inner graph (edges between non-clinical phenotype variables). The second subgraph focuses only on the edges from the genetic variant and gene expression variables to the clinical phenotype variable. Considering these two subgraphs separately greatly reduces the runtime of BGRN without sacrificing inference accuracy. We showed that the correlation and how densely connected the true network is both play a role in how well BGRN performs on identifying trait related genes. We also compared BGRN with the common variable selection method LASSO. We found that BGRN outperforms LASSO in inference accuracy for networks when the correlation is high among genes and the clinical phenotype has few true parents. We explored the effect of prior probabilities on densely connected graphs and demonstrate the difficulty in selecting an appropriate prior when the correlation is high for densely connected networks.

The simulations carried out here are on relatively small graphs and additional work is needed in order for BGRN to scale up to larger networks. More work also needs to be done to better understand and find a solution for the affect common parent confounding variables have on the inference. Future directions also include carrying out more method comparison simulations with additional existing methods to show how BGRN performs in comparison.

# Chapter 4: Conclusion and discussion

## 4.1 Conclusion

We have developed baycn as a general-purpose graph inference method that provides a measure of uncertainty in the inference, specifies an edge-level prior, and can take a candidate graph as the input. We represent a graph in terms of the edges and the states they can take. This representation allows for edge-level prior probabilities to be specified. An edge-level prior provides a direct comparison with the posterior and shows how much the data support an individual edge. This benchmark allows for more sensible cutoffs to be set for determining edge presence and direction. An edge-level prior also allows flexibility in representing prior beliefs on how densely connected the network is. In addition, baycn allows for a candidate graph to be used for the input. The edges included in the inference can be selected from prior knowledge (e.g., results from previous work or experiments) or from the output of another more efficient network inference method. Using an input graph reduces the space searched by baycn which greatly decreases the runtime.

We extended baycn and developed BGRN which has the same advantages mentioned previously for baycn but is specifically designed to infer gene regulatory networks from individual level data. To take advantage of the genetic data available we included additional variable types in BGRN. Including genetic variants in the network and using the principle of Mendelian Randomization allows for more accurate inference and for causal relationships to be inferred among genes. The clinical phenotype assumption allows networks with clinical phenotype data to be split into two different subgraphs. The first subgraph consists of the edges between the genetic variant and gene expression nodes. The second subgraph only involves the edges from the genetic variant and gene expression nodes to the clinical phenotype node. Inferring these two subgraphs separately reduces runtime without sacrificing the inference accuracy of either subgraph. These features make both baycn and BGRN highly adjustable and applicable to a wide range of problems.

## 4.2 Discussion

We make several assumptions that need to be considered when applying our methods to real data. i) When using the principle of Mendelian randomization BGRN can infer causal relationships among genes. However, care needs to be taken in how directed edges are interpreted when this principle is not being used. Both baycn and BGRN learn the statistical dependence among variables and additional information is needed in order for causal relationships to be inferred. Not all directed edges will represent a causal

relationship [18]. ii) We showed that when using the clinical phenotype assumption a network can be broken into two subgraphs when performing the inference. However, if this assumption is violated then the network can no longer be inferred as two separate subgraphs. iii) We assume that the relationship among variables is linear. It is not clear how baycn and BGRN will perform in scenarios when there is a non-linear relationship between variables. Therefore, the results from our methods should be treated carefully if applied to networks with variables that could have a non-linear relationship.

## 4.3 Future work

We have applied our baycn and BGRN to relatively small graphs and more work is needed in order for them to scale up to larger more complex networks. Possible areas of improvement include how directed cycles are found and removed as the number of potential directed cycles can increase dramatically for densely connected networks. Improvements can also be made to how a graph is represented in code. Currently we employ two representations of a network: edge state vectors and adjacency matrices both of which represent the edges in the network and their direction. These two representations are used to perform different calculations and a conversion between them takes place at each iteration. In addition, all code is written in the R language and the speed of our method would improve if portions of the algorithm were written in C++.

Additional areas for future work include analyzing other real data sets. Our method can analyze networks with three different variable types. So far we have analyzed real data sets that involve genotype and gene expression data. However, we have not performed any analyses that include all three types of data. For example, we have not applied BGRN to real data sets that include an individuals phenotype, such as disease state, in addition to the genotype and gene expression data. An analysis using these data types would provide a measure of uncertainty on the inference of trait related variants and genes. Other applications of our method include incorporating epigenetic data in the network such as DNA methylation to determine whether gene expression regulates methylation or the other way around. Additional comparisons with existing methods, both in simulated and real data, need to be done to show how our method performs in comparison and where additional improvements are needed.

# References


[1] Uri Alon. Network motifs: theory and experimental approaches. *Nature Reviews Genetics*, 8(6):450–461, 2007.

[2] A Auton, G Abecasis, D Altshuler, et al. A global reference for human genetic variation. *Nature*, 526(7571):68–74, 2015.

[3] Natalia Azpiazu and Manfred Frasch. tinman and bagpipe: two homeo box genes that determine cell fates in the dorsal mesoderm of Drosophila. *Genes & development*, 7(7b):1325–1340, 1993.

[4] Md Bahadur Badsha and Audrey Qiuyan Fu. Learning causal biological networks with the principle of Mendelian randomization. *Frontiers in Genetics*, 10:460, 2019.

[5] Md Bahadur Badsha, Evan A Martin, and Audrey Qiuyan Fu. MRPC: An R package for accurate inference of causal graphs. *arXiv preprint arXiv:1806.01899*, 2018.

[6] Albert-Laszlo Barabasi and Zoltan N Oltvai. Network biology: understanding the cell's functional organization. *Nature reviews genetics*, 5(2):101–113, 2004.

[7] Sara Barbosa, Bastian Niebel, Sebastian Wolf, Klaus Mauch, and Ralf Takors. A guide to gene regulatory network inference for obtaining predictive solutions: Underlying assumptions and fundamental biological and data constraints. *Biosystems*, 174:37–48, 2018.

[8] Barbara A Bour, Martha A O'Brien, Wendy L Lockwood, Elliott S Goldstein, Rolf Bodmer, Paul H Taghert, Susan M Abmayr, and Hanh T Nguyen. Drosophila MEF2, a transcription factor that is essential for myogenesis. *Genes & development*, 9(6):730–741, 1995.

[9] Atul J Butte and Isaac S Kohane. Mutual information relevance networks: functional genomic clustering using pairwise entropy measurements. In *Biocomputing 2000*, pages 418–429. World Scientific, 1999.

[10] Thierry Chekouo, Francesco C Stingo, James D Doecke, and Kim-Anh Do. miRNA–target gene regulatory networks: A Bayesian integrative approach to biomarker selection with application to kidney cancer. *Biometrics*, 71(2):428–438, 2015.

[11] Kuang-Chi Chen, Tse-Yi Wang, Huei-Hun Tseng, Chi-Ying F Huang, and Cheng-Yan Kao. A stochastic differential equation model for quantifying transcriptional regulatory network in saccharomyces cerevisiae. *Bioinformatics*, 21(12):2883–2890, 2005.

[12] Vivian G Cheung and Richard S Spielman. Genetics of human gene expression: mapping DNA variants that influence gene expression. *Nature Reviews Genetics*, 10(9):595–604, 2009.

[13] David Maxwell Chickering, David Heckerman, and Christopher Meek. Large-sample learning of Bayesian networks is NP-hard. *Journal of Machine Learning Research*, 5(Oct):1287–1330, 2004.

[14] Diego Colombo and Marloes H Maathuis. Order-independent constraint-based causal structure learning. *The Journal of Machine Learning Research*, 15(1):3741–3782, 2014.

[15] International Human Genome Sequencing Consortium et al. Finishing the euchromatic sequence of the human genome. *Nature*, 431(7011):931, 2004.

[16] Hassan Dana, Ghanbar Mahmoodi Chalbatani, Habibollah Mahmoodzadeh, Rezvan Karimloo, Omid Rezaiean, Amirreza Moradzadeh, Narges Mehmandoost, Fateme Moazzen, Ali Mazraeh, Vahid Marmari, et al. Molecular mechanisms and biological functions of siRNA. *International journal of biomedical science: IJBS*, 13(2):48, 2017.

[17] Eric H Davidson. *The regulatory genome: gene regulatory networks in development and evolution*. Elsevier, 2010.

[18] A Philip Dawid. Beware of the DAG! In *JMLR Workshop and Conference Proceedings*, pages 59–86, 2010.

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

[20] Vanessa Didelez and Nuala Sheehan. Mendelian randomization as an instrumental variable approach to causal inference. *Statistical methods in medical research*, 16(4):309–330, 2007.

[21] 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*, 2:38, 2014.

[22] Frank Emmert-Streib and Galina V Glazko. Network biology: a direct approach to study biological function. *Wiley Interdisciplinary Reviews: Systems Biology and Medicine*, 3(4):379–391, 2011.

[23] Jerome Friedman, Trevor Hastie, and Rob Tibshirani. Regularization paths for generalized linear models via coordinate descent. *Journal of statistical software*, 33(1):1, 2010.

[24] Nir Friedman. Inferring cellular networks using probabilistic graphical models. *Science*, 303(5659):799–805, 2004.

[25] Nir Friedman and Daphne Koller. Being Bayesian about network structure. A Bayesian approach to structure discovery in Bayesian networks. *Machine Learning*, 50(1-2):95–125, 2003.

[26] Maciej Fronczuk, Adrian E Raftery, and Ka Yee Yeung. CyNetworkBMA: a Cytoscape app for inferring gene regulatory networks. *Source code for biology and medicine*, 10(1):1–7, 2015.

[27] Paolo Giudici and Robert Castelo. Improving Markov chain Monte Carlo model search for data mining. *Machine Learning*, 50(1-2):127–158, 2003.

[28] Shawn M Gomez, William Stafford Noble, and Andrey Rzhetsky. Learning to predict protein–protein interactions from protein sequences. *Bioinformatics*, 19(15):1875–1881, 2003.

[29] Robert JB Goudie and Sach Mukherjee. A Gibbs sampler for learning DAGs. *The Journal of Machine Learning Research*, 17(1):1032–1070, 2016.

[30] Marco Grzegorczyk and Dirk Husmeier. Improving the structure MCMC sampler for Bayesian networks by introducing a new edge reversal move. *Machine Learning*, 71(2-3):265, 2008.

[31] Deborah Gunthorpe, Kathryn E Beatty, and Michael V Taylor. Different levels, but not different isoforms, of the Drosophila transcription factor DMEF2 affect distinct aspects of muscle differentiation. *Developmental biology*, 215(1):130–145, 1999.

[32] Isabelle Guyon, Dominik Janzing, and Bernhard Schölkopf. Causality: Objectives and Assessment. In *JMLR Workshop and Conference Proceedings*, pages 1–38, 2010.

[33] Yangbo He, Jinzhu Jia, Bin Yu, et al. Reversible MCMC on Markov equivalence classes of sparse directed acyclic graphs. *The Annals of Statistics*, 41(4):1742–1779, 2013.

[34] Steve Horvath and Jun Dong. Geometric interpretation of gene coexpression network analysis. *PLoS comput biol*, 4(8):e1000117, 2008.

[35] Richard Howey, So-Youn Shin, Caroline Relton, George Davey Smith, and Heather J Cordell. Bayesian network analysis incorporating genetic anchors complements conventional Mendelian randomization approaches for exploratory analysis of causal relationships in complex data. *PLoS Genetics*, 16(3):e1008198, 2020.

[36] Ling-Hong Hung, Kaiyuan Shi, Migao Wu, William Chad Young, Adrian E Raftery, and Ka Yee Yeung. fastBMA: scalable network inference and transitive reduction. *GigaScience*, 6(10):gix078, 2017.

[37] Janus S Jakobsen, Martina Braun, Jeanette Astorga, E Hilary Gustafson, Thomas Sandmann, Michal Karzynski, Peter Carlsson, and Eileen EM Furlong. Temporal ChIP-on-chip reveals Biniou as a universal regulator of the visceral muscle transcriptional network. *Genes & development*, 21(19):2448–2460, 2007.

[38] Zhenran Jiang and Yanhong Zhou. Using gene networks to drug target identification. *Journal of Integrative Bioinformatics*, 2(1):48–57, 2005.

[39] Guy Karlebach and Ron Shamir. Modelling and analysis of gene regulatory networks. *Nature Reviews Molecular Cell Biology*, 9(10):770–780, 2008.

[40] Daphne Koller and Nir Friedman. *Probabilistic Graphical Models: Principles and Techniques*. MIT Press, 2009.

[41] Daphne Koller, Nir Friedman, Lise Getoor, and Ben Taskar. Graphical models in a nutshell. *Introduction to statistical relational learning*, 43, 2007.

[42] Mina Moradi Kordmahalleh, Mohammad Gorji Sefidmazgi, Scott H Harrison, and Abdollah Homaifar. Identifying time-delayed gene regulatory networks via an evolvable hierarchical recurrent neural network. *BioData mining*, 10(1):29, 2017.

[43] Gilles Kratzer and Reinhard Furrer. Is a single unique bayesian network enough to accurately represent your data? *arXiv preprint arXiv:1902.06641*, 2019.

[44] Jack Kuipers and Giusi Moffa. Partition MCMC for inference on acyclic digraphs. *Journal of the American Statistical Association*, 112(517):282–299, 2017.

[45] Jack Kuipers, Polina Suter, and Giusi Moffa. Efficient structure learning and sampling of Bayesian networks. *arXiv preprint arXiv:1803.07859*, 2018.

[46] Tuuli Lappalainen, Michael Sammeth, Marc R Friedländer, et al. Transcriptome and genome sequencing uncovers functional variation in humans. *Nature*, 501(7468):506–511, 2013.

[47] Steffen L Lauritzen. *Graphical models*, volume 17. Clarendon Press, 1996.

[48] Gwenaël GR Leday and Sylvia Richardson. Fast Bayesian inference in large Gaussian graphical models. *arXiv preprint arXiv:1803.08155*, 2018.

[49] Liang Liang, Li Gao, Xiao-Ping Zou, Meng-Lan Huang, Gang Chen, Jian-Jun Li, and Xiao-Yong Cai. Diagnostic significance and potential function of miR-338-5p in hepatocellular carcinoma: A bioinformatics study with microarray and RNA sequencing data. *Molecular medicine reports*, 17(2):2297–2312, 2018.

[50] Brenda Lilly, Samuel Galewsky, Anthony B Firulli, Robert A Schulz, and Eric N Olson. D-MEF2: a MADS box transcription factor expressed in differentiating mesoderm and muscle cell lineages during Drosophila embryogenesis. *Proceedings of the national academy of sciences*, 91(12):5662–5666, 1994.

[51] Hong Ling, Sandhya Samarasinghe, and Don Kulasiri. Novel recurrent neural network for modelling biological networks: oscillatory p53 interaction dynamics. *Biosystems*, 114(3):191–205, 2013.

[52] Kenneth Lo, Adrian E Raftery, Kenneth M Dombek, Jun Zhu, Eric E Schadt, Roger E Bumgarner, and Ka Yee Yeung. Integrating external biological knowledge in the construction of regulatory networks from time-series expression data. *BMC systems biology*, 6(1):101, 2012.

[53] Wendy K Lockwood and Rolf Bodmer. The patterns of wingless, decapentaplegic, and tinman position the Drosophila heart. *Mechanisms of development*, 114(1-2):13–26, 2002.

[54] Piyush B Madhamshettiwar, Stefan R Maetschke, Melissa J Davis, Antonio Reverter, and Mark A Ragan. Gene regulatory network inference: evaluation and application to ovarian cancer allows the prioritization of drug targets. *Genome medicine*, 4(5):41, 2012.

[55] David Madigan, Jeremy York, and Denis Allard. Bayesian graphical models for discrete data. *International Statistical Review/Revue Internationale de Statistique*, 63(2):215–232, 1995.

[56] Maria E Manioudaki and Panayiota Poirazi. Modeling regulatory cascades using Artificial Neural Networks: the case of transcriptional regulatory networks shaped during the yeast stress response. *Frontiers in genetics*, 4:110, 2013.

[57] Daniel Marbach, James C Costello, Robert Küffner, Nicole M Vega, Robert J Prill, Diogo M Camacho, Kyle R Allison, Manolis Kellis, James J Collins, and Gustavo Stolovitzky. Wisdom of crowds for robust gene network inference. *Nature methods*, 9(8):796–804, 2012.

[58] D. Margaritis. *Learning Bayesian Network Model Structure from Data*. PhD thesis, School of Computer Science, Carnegie-Mellon University, Pittsburgh, PA, May 2003. Available as Technical Report CMU-CS-03-153.

[59] Adam A Margolin, Ilya Nemenman, Katia Basso, Chris Wiggins, Gustavo Stolovitzky, Riccardo Dalla Favera, and Andrea Califano. Aracne: an algorithm for the reconstruction of gene regulatory networks in a mammalian cellular context. In *BMC bioinformatics*, volume 7, page S7. Springer, 2006.

[60] Evan A Martin and Audrey Qiuyan Fu. Bayesian inference of directed acyclic graphs with edge-level prior probabilities. *arXiv preprint arXiv:1909.10678*, 2019.

[61] Hirotaka Matsumoto, Hisanori Kiryu, Chikara Furusawa, Minoru SH Ko, Shigeru BH Ko, Norio Gouda, Tetsutaro Hayashi, and Itoshi Nikaido. SCODE: an efficient regulatory network inference algorithm from single-cell RNA-Seq during differentiation. *Bioinformatics*, 33(15):2314–2321, 2017.

[62] Peter McCullagh. Regression models for ordinal data. *Journal of the Royal Statistical Society: Series B (Methodological)*, 42(2):109–127, 1980.

[63] Abdolreza Mohammadi, Ernst C Wit, et al. Bayesian structure learning in sparse Gaussian graphical models. *Bayesian Analysis*, 10(1):109–138, 2015.

[64] Judea Pearl. *Probabilistic reasoning in intelligent systems: networks of plausible inference*. Elsevier, 2014.

[65] Marta E Polak, Chuin Ying Ung, Joanna Masapust, Tom C Freeman, and Michael R Ardern-Jones. Petri net computational modelling of langerhans cell interferon regulatory factor network predicts their role in t cell activation. *Scientific reports*, 7(1):1–13, 2017.

[66] Andrea Rau, Florence Jaffrézic, Jean-Louis Foulley, and Rebecca W Doerge. Reverse engineering gene regulatory networks using approximate Bayesian computation. *Statistics and Computing*, 22(6):1257–1271, 2012.

[67] Vahid Rezaei Tabar, Hamid Zareifard, Selva Salimi, and Dariusz Plewczynski. An empirical Bayes approach for learning directed acyclic graph using MCMC algorithm. *Statistical Analysis and Data Mining: The ASA Data Science Journal*, 12(5):394–403, 2019.

[68] Stuart J Russell and Peter Norvig. *Artificial Intelligence: A Modern Approach*. Prentice Hall, 2009.

[69] Thomas Sandmann, Charles Girardot, Marc Brehme, Waraporn Tongprasit, Viktor Stolc, and Eileen EM Furlong. A core transcriptional network for early mesoderm development in Drosophila melanogaster. *Genes & development*, 21(4):436–449, 2007.

[70] Christopher J Savoie, Sachiyo Aburatani, Shouji Watanabe, Yoshihiro Eguchi, Shigeru Muta, Seiya Imoto, Satoru Miyano, Satoru Kuhara, and Kosuke Tashiro. Use of gene networks from full genome microarray libraries to identify functionally relevant drug-affected genes and gene regulation cascades. *Dna Research*, 10(1):19–25, 2003.

[71] Loïc Schwaller, Stéphane Robin, and Michael Stumpf. A closed-form approach to Bayesian inference in tree-structured graphical models. *arXiv preprint arXiv:1504.02723*, 2015.

[72] Marco Scutari. Learning Bayesian networks with the bnlearn R package. *arXiv preprint arXiv:0908.3817*, 2009.

[73] Ilya Shmulevich, Edward R Dougherty, Seungchan Kim, and Wei Zhang. Probabilistic Boolean networks: a rule-based uncertainty model for gene regulatory networks. *Bioinformatics*, 18(2):261–274, 2002.

[74] Peter Spirtes, Clark N Glymour, and Richard Scheines. *Causation, Prediction, and Search*. MIT Press, 2000.

[75] Oliver Stegle, Leopold Parts, Matias Piipari, John Winn, and Richard Durbin. Using probabilistic estimation of expression residuals (PEER) to obtain increased power and interpretability of gene expression analyses. *Nature Protocols*, 7(3):500–507, 2012.

[76] Robert Stojnic, Audrey Qiuyan Fu, and Boris Adryan. A graphical modelling approach to the dissection of highly correlated transcription factor binding site profiles. *PLoS computational biology*, 8(11), 2012.

[77] John D Storey and Robert Tibshirani. Statistical significance for genomewide studies. *Proceedings of the National Academy of Sciences*, 100(16):9440–9445, 2003.

[78] The GTEx Consortium. Genetic effects on gene expression across human tissues. *Nature*, 550(7675):204–213, 2017. PMC5776756.

[79] Tianhai Tian and Kevin Burrage. Stochastic models for regulatory networks of the genetic toggle switch. *Proceedings of the national Academy of Sciences*, 103(22):8372–8377, 2006.

[80] Dong Ling Tong, David J Boocock, Gopal Krishna R Dhondalay, Christophe Lemetre, and Graham R Ball. Artificial neural network inference (ANNI): a study on gene-gene interaction for biomarkers in childhood sarcomas. *PLoS One*, 9(7):e102483, 2014.

[81] SM Minhaz Ud-Dean and Rudiyanto Gunawan. Optimal design of gene knockout experiments for gene regulatory network inference. *Bioinformatics*, 32(6):875–883, 2016.

[82] Thomas Verma and Judea Pearl. Equivalence and synthesis of causal models. In *Proceedings of the Sixth Annual Conference on Uncertainty in Artificial Intelligence*, pages 255–270. Elsevier Science Inc., 1990.

[83] Jussi Viinikka and Mikko Koivisto. Layering-mcmc for structure learning in bayesian networks. In *Conference on Uncertainty in Artificial Intelligence*, pages 839–848. PMLR, 2020.

[84] Diego Villar, Paul Flicek, and Duncan T Odom. Evolution of transcription factor binding in metazoans-mechanisms and functional implications. *Nature Reviews Genetics*, 15(4):221–233, 2014.

[85] YX Rachel Wang and Haiyan Huang. Review on statistical methods for gene network reconstruction using expression data. *Journal of theoretical biology*, 362:53–61, 2014.

[86] Fan Yang, Jiebiao Wang, Brandon L Pierce, Lin S Chen, François Aguet, Kristin G Ardlie, Beryl B Cummings, Ellen T Gelfand, Gad Getz, Kane Hadley, et al. Identifying cis-mediators for trans-eQTLs across many human tissues using genomic mediation analysis. *Genome Research*, 2017.

[87] Chen Yao, Roby Joehanes, Andrew D Johnson, Tianxiao Huan, Chunyu Liu, Jane E Freedman, Peter J Munson, David E Hill, Marc Vidal, and Daniel Levy. Dynamic role of trans regulation of gene expression in relation to complex traits. *The American Journal of Human Genetics*, 100(4):571–580, 2017.

[88] Ka Yee Yeung, Kenneth M Dombek, Kenneth Lo, John E Mittler, Jun Zhu, Eric E Schadt, Roger E Bumgarner, and Adrian E Raftery. Construction of regulatory networks using expression time-series data of a genotyped population. *Proceedings of the National Academy of Sciences*, 108(48):19436–19441, 2011.

[89] William Chad Young, Adrian E Raftery, and Ka Yee Yeung. Fast Bayesian inference for gene regulatory networks using ScanBMA. *BMC systems biology*, 8(1):47, 2014.

[90] Stephane Zaffran, Axel Küchler, Hsiu-Hsiang Lee, and Manfred Frasch. biniou (foxf), a central component in a regulatory network controlling visceral mesoderm development and midgut morphogenesis in Drosophila. *Genes & development*, 15(21):2900–2915, 2001.

[91] Julia Zeitlinger, Robert P Zinzen, Alexander Stark, Manolis Kellis, Hailan Zhang, Richard A Young, and Michael Levine. Whole-genome chip–chip analysis of dorsal, twist, and snail suggests integration of diverse patterning processes in the Drosophila embryo. *Genes & development*, 21(4):385–390, 2007.

[92] Qiangfeng Cliff Zhang, Donald Petrey, Lei Deng, Li Qiang, Yu Shi, Chan Aye Thu, Brygida Bisikirska, Celine Lefebvre, Domenico Accili, Tony Hunter, et al. Structure-based prediction of protein–protein interactions on a genome-wide scale. *Nature*, 490(7421):556–560, 2012.

[93] Xiujun Zhang, Juan Zhao, Jin-Kao Hao, Xing-Ming Zhao, and Luonan Chen. Conditional mutual inclusive information enables accurate quantification of associations in gene regulatory networks. *Nucleic acids research*, 43(5):e31–e31, 2015.

[94] Robert P Zinzen, Charles Girardot, Julien Gagneur, Martina Braun, and Eileen EM Furlong. Combinatorial binding predicts spatio-temporal cis-regulatory activity. *Nature*, 462(7269):65–70, 2009.