Inferring genetic pathways associated with Schizophrenia using Bayesian Structure Learning

Daniel de Vassimon Manela[1]

Computational Statistics and Machine Learning

Supervisors: Prof. Mark Herbster and Dr. Karim Malki

Submission date: 25th September 2020

Abstract

The central aim of this thesis is to learn genetic pathways associated with Schizophrenia utilising Bayesian structure-learning algorithms. The thesis first presents a comprehensive review of both traditional and state of the art algorithms, whose performance is assessed using well established synthetic and experimental benchmark datasets. The results of these experiments, with optimal hyperparameter selection, show that the H2PC structure learning algorithm outperforms its peers. Motivated by the fact that many biological systems do not conform to Gaussian assumptions, the performance of all algorithms at learning networks from non-Gaussian data is assessed. In order to derive confidence measures for the inferred networks, Bayesian model averaging is used. Statistically significant edges are identified by using shrinkage operators akin to lasso regularisation in the domain of linear regression. The thesis provides a novel exploration to the problem of learning Bayesian networks using data with an incomplete feature set. This is tackled from both experimental and theoretical approaches. We demonstrate that the learned network may not capture dependencies between variables that are adjacent in the true network and may instead be more likely to contain links between variables that are not directly dependent on each other. This finding is especially significant for researchers who wish to apply structure learning algorithms to data from a reduced feature space. Finally, we apply the H2PC algorithm to Schizophrenia gene expression data derived from two RNASeq datasets generated independently by the Lieber Institute and CommonMind Consortium. Cross-validation of the learned networks results identifies fourteen edges that are hypothesized to be representative of the true genetic network.

Acknowledgements

I would like to thank Professor Mark Herbster, Dr. Karim Malki, and Dr. Nico Kist for their excellent continuous support, encouragement, guidance, and for being readily available to discuss my work. I would also like to acknowledge Professor David Barber for fruitful discussions.

I am also grateful to Dr. Marco Scutari for his technical and theoretical guidance on the computational aspects of Bayesian structural learning.

Finally, I would like to thank Andrew Strange for his suggestions and comments on the written material.

Contents

Chapter 1

Introduction

Understanding the behaviour of complex genetic networks is of vital importance to society. Degenerative cerebral conditions are the leading cause of death in the United Kingdom and this is forecasted to increase over the coming years [1]. A secondary, financial problem is the vast amount of money that is spent on caring for sufferers of mental illness [2]. Both direct indirect and direct costs are estimated to double by 2030, and that does not include associated costs associated with mental disorders outside of the healthcare systems, such as legal costs caused by illicit drug abuse [3, 4].

A central requirement to any sort of medical treatment is the identification of the genetic pathways responsible for these illnesses. While critical genetic loci have been identified for several mental diseases, it may not always be possible to target specific genes. For this reason, knowledge of the genetic network around these genes is of significant value, as regulation of critical genes can be achieved by treating upstream or downstream genetic sites.

The goal of this thesis is the identification of genetic networks pertaining to Schizophrenia. Schizophrenia affects more than 2% of the UK population, and is a major psychiatric disorder that alters the sufferer's perception, thoughts, and behaviour. It manifests itself during the formative childhood to teen years of an individual's life, and current drug treatments seek to control the symptoms rather than treat the root causes of the disease. Currently, there is limited evidence of the efficacy of current drug treatments in children and younger patients with Schizophrenia [5]. There is therefore a clear need to identify more effective forms of treatment that inhibit the progression of the illness at an early stage.

In this thesis, Bayesian networks (BNs) are used to partially construct a Schizophrenia genetic network. BNs are graphical representations of probabilistic relationships between a set of variables and have become increasingly popular over recent years across a variety of bioinformatic, economic, and statistical domains. They have demonstrated success at both validating known genetic connections and identifying new pathways that have been shown to be correct through later pharmaceutical experiments. In order to construct a model of the Schizophrenia genetic network, we use two recent datasets published by the Lieber Institute for Brain Development (LIBD) and the CommonMind Consortium (CMC) [6, 7]. While genetic studies have identified genomic regions associated with disease risk, these experimental datasets were developed with the aim of identifying the specific mechanisms which lead to Schizophrenia. The genomic data consists of RNAseq

measurements taken from a combined total of around 600 post-mortem human brains, with gene expression measurements taken from brain regions associated with Schizophrenia degeneracy.

Since the turn of the 21st century, a variety of methods have been derived to learn BN structures. These methods fall within four general categories: constraint-based, score-based, hybrid, and regression algorithms. Constraint-based methods aim to infer dependencies between variables using hypothesis tests, whereas score-based algorithms use heuristic search techniques to find a structure which maximises a predetermined score function. Hybrid methods synthesise both constraint and score-based algorithms in order to mitigate their respective weaknesses, whereas regression methods involve the minimisation of a loss function using optimisation techniques.

Structure learning can be performed on both discrete and continuous datasets. When dealing with the latter, most structure learning algorithms assume that the data is drawn from a linear multivariate Gaussian distribution [8]. In practice, the Gaussian assumption may not necessarily hold, and as we will demonstrate, distributions of gene expression are non-negative and often heavily positively skewed. One approach for dealing with non-Gaussian continuous data involves discretising the data and learning a BN using discrete algorithms. We use the popular Hartemink discretisation technique that seeks to maintain as much information in the dataset as possible [9]. Its performance is validated against the standard RAF Sachs dataset commonly used to benchmark structure learning algorithm performance [10]. Whilst validation on the RAF benchmark shows that a greater number of true edges from the RAF network can be identified by data discretisation, we show that this conclusion does not necessarily hold on other datasets. When structure learning was performed on discretised Schizophrenia data, the learned network was small and sparse. In contrast, learning with continuous data yielded a much larger network, which contained the discrete network as a subgraph.

One of the key challenges in learning an unknown network is the difficulty of assessing the accuracy of a learned network without a source of truth to cross reference it against. In this thesis, Bayesian model averaging is used to derive confidence measures for the inferred network. We then calculate a significance threshold to identify statistically significant edges in the learned model. This is done using an L1 shrinkage operator, akin to lasso regression in the domain of linear regression [11].

A further challenge relates to learning highly dimensional networks. In practice, genetic datasets may contain a few hundred observations from tens of thousands of genes. Structure learning methods are known to suffer considerably in both accuracy and runtime when the dimensions of the dataset are very large. One approach to overcome this limitation is to learn networks from a reduced subset of features, which may identify dependencies between nodes which do not have any direct dependency in the true network. This thesis offers a novel perspective to the above problem from both experimental and theoretical approaches. We demonstrate that the learned network may not capture dependencies between variables that are adjacent in the true network and may instead be more likely to contain links between variables that are not directly dependent on each other.

1.1 Main Contributions

As well as offering a systematic review of structure learning methods, this thesis presents the results of hyperparameter optimisation on learning biological networks. It also investigates the impact of dataset and graph size on the performance of structure learning algorithms. Furthermore, it supports the findings reported in the literature that hybrid algorithms can be more effective than their constituent counterparts. We demonstrate that a novel hybrid version of a state of the art algorithm can improve the performance on medium to large datasets. The results reported here may be of interest for researchers investigating which algorithm to use (along with optimal hyperparameterisation) when learning from a particular dataset.

This thesis also highlights the inadequacies of the commonly used RAF benchmark dataset as a means of validating the performance of algorithms when applied to non-Gaussian data. The conclusions drawn from the RAF benchmark do not carry to the non-Gaussian Schizophrenia data; in fact, they are contradictory. Validation on the RAF benchmark shows that a greater number of true edges from the RAF network can be identified by data discretisation. However, the network learned from discretised Schizophrenia data was small and sparse. In contrast, the network learned from continuous Schizophrenia data yielded a much larger network, which contained the discrete network as a subgraph. This highlights the importance of establishing a robust set of non-Gaussian benchmark datasets spanning different graph and dataset sizes.

The thesis also offers a novel perspective to the problem of feature reduction when learning from highly dimensional datasets. We show that learned networks may not capture dependencies between directly dependent variables in the true networks and may be more likely to infer links between variables that are not directly dependent on each other. This finding is especially important for researchers who wish to apply structure learning algorithms to data from a reduced feature space.

To the best of my knowledge, this thesis is the first to apply structure learning methods to infer Schizophrenia pathways from RNASeq data and identifies 14 genetic links that we hypothesise occur in the true genetic network. It is hoped that this finding will be of value to those involved in the development of Schizophrenia treatments.

1.2 Thesis Organisation

The structure of this thesis is as follows. Chapter 2 reviews the theoretical background behind graphical models and outlines the structure learning algorithms investigated in the thesis. Chapter 3 describes the metrics used for performance validation and presents the results of hyperparameter selection for each algorithm. The impact of dataset and graph size on algorithm performance is also assessed. Chapter 4 introduces Bayesian Model Averaging and investigates the effectiveness of data discretisation on learning BNs from non-Gaussian data. The impact of structure learning on feature reduced datasets is investigated in Chapter 5. The results from learning genetic pathways from Schizophrenia data are presented in Chapter 6. Conclusions are drawn in Chapter 7, along with suggestions for further research.

Chapter 2

Literature Review

In this chapter, we present a review of the current machine learning and statistical theory central to Bayesian Structure Learning. Firstly, we review the concepts of conditional and marginal independencies and contextualise them in the framework of probabilistic graphical modelling. Following this, we present the constraint-based, score-based, hybrid, and regression structure learning algorithms used in the thesis.

2.1 Graphical Models

In several domains, such as speech recognition, finance, and bioinformatics, complex systems with many potentially interrelated features are frequently encountered and can at first appear intractable. In these fields, a parsimonious representation of the system's dynamics is of great value. A clear example of this is in the medical domain, where diagnosing the exact illness contracted by a patient given their symptoms is of critical importance to medical professionals. Identifying the structural relationship between covariates can be a life or death question.

This task is further complicated by the fact that many systems involve a considerable degree of stochasticity. In several real-world applications, it is often impossible to obtain a complete measurement of a system, and given limited observations, it is impossible to accurately characterise relationships within models as deterministic. Because of this, it is necessary to use probabilistic models which provide a degree of flexibility in capturing the effects of stochasticity in a parsimonious fashion.

Graphical Models provide an elegant framework for representing the kind of probabilistic structural models discussed in the above paragraphs. They provide a simple visual representation of a model's structure, as well as providing key probabilistic insights into the probabilistic properties of a model such as conditional and marginal independence relationships. These graphs represent the random variables of a system as the graph's nodes (or vertices), with the edges describing the independence relationships between these variables. In what follows, a graph, its nodes and edges will be represented by G, V and E respectively.

Before outlining the general theory behind Graphical Models, it is worth reviewing the definitions of conditional and marginal independence.

DEFINITION 1. *Two events α and β are marginally independent if $P(\alpha|\beta) = P(\alpha)$ and $P(\beta|\alpha) = P(\beta)$.*

Marginal independence relationships are rarely encountered in real world systems. More commonly encountered are events which are independent given some additional information. Consider the following example. A coin is randomly selected from two coins: a fair coin and a two-headed coin. Select one coin and toss it twice. Let X and Y be the outcome that the first and second tosses respectively are tails, and let Z be the outcome that the fair coin is selected. Even though X is not independent of Y, X is *conditionally independent* of Y given Z, since knowledge of Y's outcome provides no further information on the likelihood of X if Z is known. This example illustrates how two marginally dependent variables become conditionally independent of each other given additional information.

DEFINITION 2. *Event α is conditionally independent from event β given event γ if $P(\alpha|\beta,\gamma) = P(\alpha|\gamma)$ and $P(\alpha|\beta,\gamma) = P(\alpha|\gamma)$. These expressions are symmetrical with respect to α and β.*

As mentioned above, a graphical model provides a representation of the independencies which characterise by a set of random variables. Another alternative interpretation involves considering the edges as representing a factorised decomposition of the full joint distribution into smaller factors, each of which is associated with a much smaller subgraph. In the end, these approaches are equivalent - the independencies which characterise a distribution naturally result in a factorised representation and vice versa [12]. Such graphical representations fall under two general categories - Undirected and Directed Graphical Models.

2.1.1 Undirected Graphical Models

Edges in **Undirected Graphical Models** do not carry any directional significance. Each edge corresponds to a direct probabilistic interaction between the connected nodes. These models explicitly capture conditional independencies only. To identify these probabilistic dependencies, we consider whether paths between nodes on the exist. In an undirected graph, nodes which are conditioned upon "block" a path. If all paths between nodes A and B are blocked by the set of variables S, $A \perp\!\!\!\perp B \mid S$. Consider the example in Figure 2.1. Node C is conditionally independent from all other nodes in the graph when conditioned upon the set of nodes $M = \{A, B, D\}$, i.e. $C \perp\!\!\!\perp X \setminus M \mid M$ where X are all nodes in the graph not in M or C. This nicely leads towards the concept of the Markov Blanket.

DEFINITION 3. *The **Markov Blanket** of a node X in a graph G, $MB_G(X)$ is any subset of variables S which when conditioned upon, makes X independent from all other nodes in the graph not in S. The minimal set of S which satisfies this condition is known as the **Markov Boundary**.*

For an undirected network, the Markov Boundary of a node is the set of nodes which lie directly adjacent to it. This is otherwise referred to as the *neighbourhood* of the node.

As mentioned previously, these conditional independence relations can be derived from the factorised representation of the graph. Since each undirected edge represents a single probabilistic factor, the joint distribution of the variables in the graph in fig. 2.1 is:

$$P(A, B, C, D, E, F, G) = \phi_1(A, C, D)\phi_2(B, C)\phi_3(D, G)\phi_4(B, E)\phi_5(E, F)\phi_6(F, G)\phi_7(D, G) \quad (2.1)$$

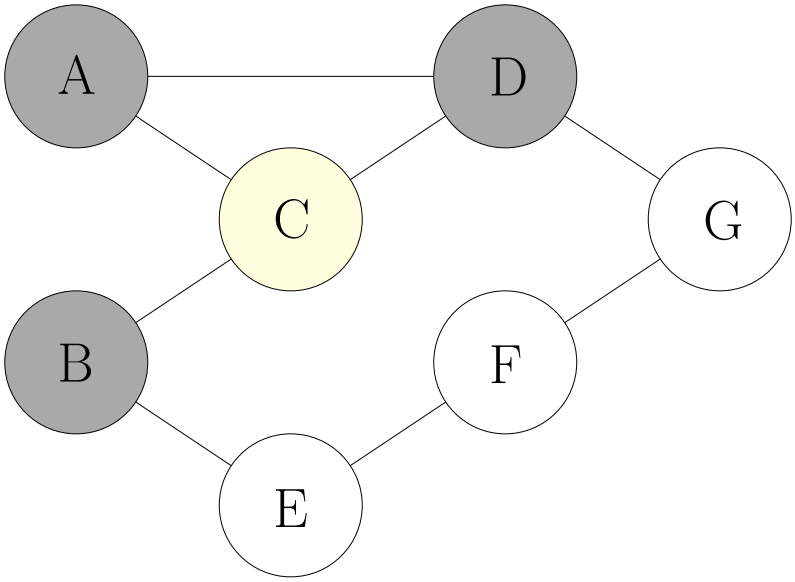

Figure 2.1: Illustration of the Markov Blanket concept for an undirected graphical model. The target node C (yellow) is blocked off from all non-neighbouring nodes if we condition upon A, B and D (grey). The grey nodes are the Markov Boundary for node C.

where $\phi_i(\cdot)$ represent unspecified normalised probabilistic factors.

The conditional distribution of C given all other nodes $N \setminus C$ is

$$P(C|N \setminus C) = \frac{P(A,B,C,D,E,F,G)}{P(A,B,D,E,F,G)} = \frac{P(A,B,C,D,E,F,G)}{\sum_C P(A,B,D,E,F,G)} \tag{2.2}$$

$$= \frac{\prod_{i=1}^{7} \phi_i(X_i|ne(X_i)}{\sum_C \prod_{i=1}^{7} \phi_i(X_i|ne(X_i)} \tag{2.3}$$

$$= \frac{\phi_1(A,C,D)\phi_2(B,C)}{\sum_C \phi_1(A,C,D)\phi_2(B,C)} \tag{2.4}$$

where $ne(X_i)$ is the neighbourhood of X_i (the parents and children of X_i).

The only factors that remain are the terms with a dependency on C, which happen to be factors included within the $MB(C)$. Note that this conditional distribution is only a function of the Markov Boundary of C (and C itself), and so the functional form of this conditional probability will be the same for all sets larger than $MB(C)$. The minimum size set N where this conditional relation holds is when N is equal to the Markov Boundary, and so it can be seen that the interpretation of edges as factors and as characterising direct conditional dependencies are equivalent [8].

In summary, the correspondence between the topology of an undirected network and its dependence relationships are as follows.

THEOREM 4. *If a subset of graph nodes Z in an undirected network blocks all paths between nodes X and Y, then X and Y are independent given Z.*

2.1.2 Bayesian Networks

Bayesian Networks (sometimes referred to as Directed Graphical Models) represent a set of random variables and their dependencies via a Directed Acyclic Graph.

DEFINITION 5. *A Directed Acyclic Graph is a directed graph with no directed cycles, such that there is no directed path from a node X that returns back to X.*

Directed networks provide a more natural representation for capturing causal dependencies between nodes. More subtly, they can capture probabilistic relationships between nodes which undirected networks cannot. To demonstrate this, consider the following famous "sprinkler" example adapted from Judea Pearl's *"Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference"* [13, 14]. Consider the network in fig. 2.2. Observing that the ground is shiny, or that one's shoes are wet provide evidence that the grass is wet. That is, if a directed edge points from node $A \rightarrow B$, the rational is that having evidence of outcome of B provides evidence on the outcome of A. Similarly, if it rains the previous night, then it is very probable that the grass will be wet the following day.

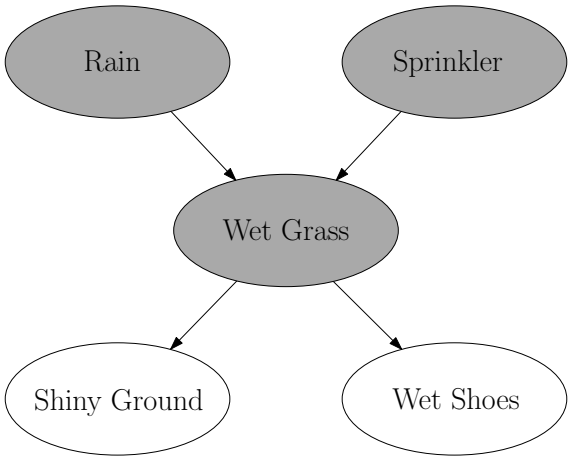

Figure 2.2: *Explaining Away* example adapted from Pearl [14, 15]. The greyed subgraph is an example of a V-Collider which characterises probabilistic relations which simply cannot be captured by undirected graphs.

A unique feature of BNs are the presence of "V-Colliders" in the network, which capture probabilistic relationships which cannot be expressed in undirected systems.

DEFINITION 6. *The set of nodes $\{X, Y, Z\}$ from a Bayesian Network G is a V-Collider if G has $X \rightarrow Y \leftarrow Z$ and X and Z are non-adjacent.*

Consider the V-Collider subgraph in Figure 2.2, coloured in grey. Without any additional information, confirmation of rain on the previous night does not provide any evidence for sprinklers having been active. However, if the observer had prior information that the grass was wet, and confirmation that it had rained the previous night would reduce the probability that the sprinkler had been active. This is known as the process of *Explaining Away* - knowledge of a causal variable (the rain) explains the observed effect (the wet grass) and so reduces the probability that other

causes were involved. This "Occam's Razor" approach reduces the probability of additional causal hypotheses if they are unnecessary to account for the evidence [14].

Directed edges in BNs have more restrictive meaning than their undirected counterparts when considering the factor decomposition approach of graphical models.

DEFINITION 7. *Define the* ***parents*** *of a node X $pa(X)$ as nodes which have directed edges pointing towards X, and the* ***children*** *of X $ch(X)$ being the nodes which have directed edges pointing towards them from X.*

In a BN, the contributing factors of the full global distribution are all local conditional probability distributions. The factor each node provides to the joint distribution is the distribution of that node, conditional on the values of its parents, such that the global distribution of all K nodes $\{X\}_k$ is

$$P(X_1, \ldots, X_K) = \prod_{k=1}^{K} P\left(X_k | pa(X_k)\right) \tag{2.5}$$

This factorisation can be used to formally characterise the independence relations of the V-Collider discussed above. Consider again the greyed subgraph in fig. 2.2. Since root nodes have no parents, their factors are simply marginal probabilities, and so the full joint distribution of the V-Collider subgraph is

$$P(\text{Rain}, \text{Sprinkler}, \text{Wet Grass}) = P(\text{Rain})P(\text{Sprinkler})P(\text{Wet Grass}|\text{Rain}, \text{Sprinkler}) \tag{2.6}$$

In the case where no variables are observed, the effect "Wet Grass" can be marginalised to yield

$$P(\text{Rain}, \text{Sprinkler}) = P(\text{Rain})P(\text{Sprinkler}) \tag{2.7}$$

which shows that in the absence of the child node, isolated coparents are marginally independent from one another. However, if the distribution is conditioned upon "Wet Grass", independence no longer holds

$$P(\text{Rain}, \text{Sprinkler}|\text{Wet Grass}) = \frac{P(\text{Rain}, \text{Sprinkler}, \text{Wet Grass})}{P(\text{Wet Grass})} \tag{2.8}$$

$$= \frac{P(\text{Rain})P(\text{Sprinkler})P(\text{Wet Grass}|\text{Rain}, \text{Sprinkler})}{P(\text{Wet Grass})} \tag{2.9}$$

which generally cannot be factored into $P(\text{Rain}|\text{Wet Grass})P(\text{Sprinkler}|\text{Wet Grass})$, implying that

$$\text{Rain} \not\perp\!\!\!\perp \text{Sprinkler} \mid \text{Wet Grass} \tag{2.10}$$

and so coparents of a node are not independent from one another conditioned upon their shared children.

This conditional relationship leads to a slightly different Markov Blanket structure compared to its undirected counterpart. For BNs, the Markov Blanket comprises the set of children, parents *and coparents* of a node, since explaining away means that observation of child nodes will not isolate their coparents from one another. The differences between the two are highlighted in Figure 2.3.

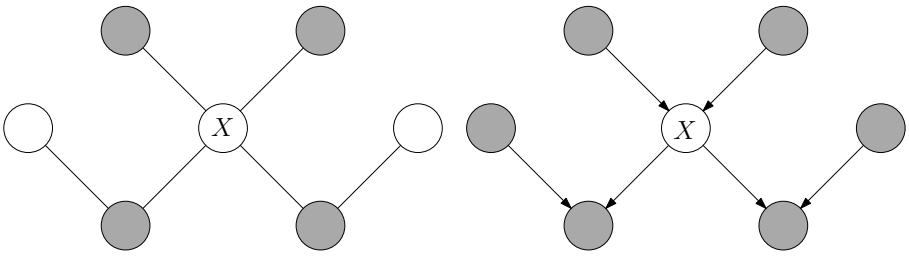

Figure 2.3: Differences between Markov Blankets of undirected and directed graphical models. The minimal Markov Blanket of node X in the undirected graph (left) is formed from only its parents and children. In the directed graph (right), the coparents of node X's children need to be included.

The relationships captured by V-Colliders requires an adjustment of the separation property defined in Definition 4.

DEFINITION 8. *If Z is a subset of nodes in a BN, then Z is said to d-separate nodes X and Y if there exists no path between X and Y where*

- *every node with converging arrows (i.e. the base of a V-Collider) is in Z or has a descendent in Z*
- *every other node is outside Z.*

Paths satisfying the conditions above are said to be active. Otherwise, they are said to be blocked by Z.

It can be formally proved that all d-separation criteria in a BN are also represented by the global probability distribution. An outline of the approach is explored in the Rain, Sprinkler, Wet Grass example mentioned above, but a more formal proof can be found in Pearl's original paper [13].While a BN is characterised by a unique product of local probabilistic factors, they do not uniquely characterise the same set of conditional and marginal independence relations. Consider the following example [16].

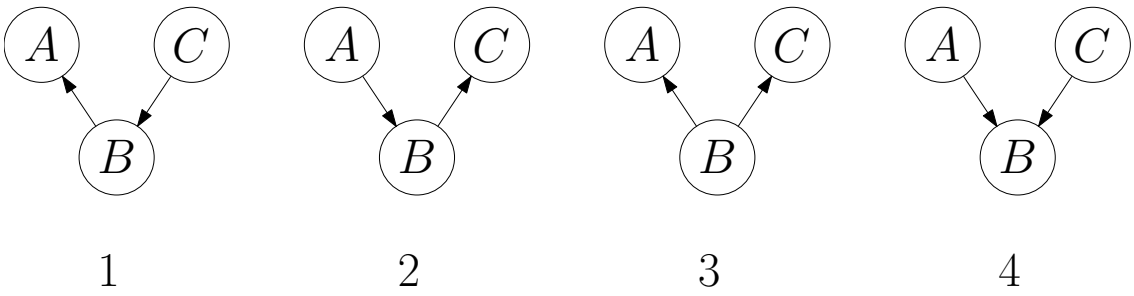

Figure 2.4: All possible permutations of BNs for three nodes (A, B, and C) with edges between AB and BC. The first three characterise the same set of probabilistic relationships and are part of the same equivalence class. The fourth, a *V-Collider*, cannot be determined from the other three, and characterises a set of both marginal and conditional probabilistic relations.

Fig. 2.4 contains four examples of causal models. The factorisation of the first BN is $P(C)P(B|C)P(A|B)$, whereas the second is $P(A)P(B|A)P(C|B)$. Using Bayes's Law, it can be

shown that these models are equivalent since $P(A)P(B|A) = P(A|B)P(B)$ and $P(B)P(C|B)$ and $P(B|C)P(C)$. Similarly, the third model factorised into $P(B)P(A|B)P(C|B)$ is equivalent to the first since $P(B)P(C|B) = P(B|C)P(C)$. However, the V-Collider in the fourth model is factorised as $P(A)P(C)P(B|A,C)$ which cannot be determined from the other three models. The equivalence of the first three model can be seen directly by noticing that they share the exact same independence relations by using the d-separation criteria in Definition 8. It can be shown that d-separation completely determines the equivalence of two causal models [16]. Before defining an equivalence class, the concept of a skeleton needs to be introduced.

DEFINITION 9. *The **skeleton** of a Bayesian Network is the undirected graph derived from removing directionality from all edges.*

DEFINITION 10. *Two Bayesian-network structures are said to belong to the same equivalence class if the set of distributions that can be represented with one of those structures is identical to the set of distributions that can be represented with the other.*

DEFINITION 11. *A set of Bayesian Networks belong to the same **equivalence class** only if they share the same skeleton and V-Collider structure.*

Since BNs do not uniquely capture the independence relationships in a system, it is worth introducing Partially and Completed Partially Directed Acyclic Graphs (PDAGs/CPDAGs), which uniquely represent a set of equivalent BNs. PDAGs are graphs which can contain both directed and undirected edges. No cycles which contain a directed edge are permitted in these graph classes. CPDAGs are a specific subclass of PDAGs, which only retain directionality of edges which form part of a V-Collider structure [13, 16]. Following Definition 11, it is clear that these graphs uniquely characterise an equivalence class, and so it naturally follows that graphical comparisons of CPDAGs should form the bedrock of the algorithm assessments conducted in Section 2.2.

At a first glance, it may be expected that since undirected graphical models characterise a set of conditional independence relationships, the skeleton of a BN captures the same conditional relationships as its directed counterpart. The V-Collider is a simple counter example to this hypothesis. Even though no direct edge connects the parent nodes, they are conditionally dependent on their shared children. To ensure consistency in conditional independence relationships between BNs and their undirected counterparts, edges need to be introduced between all coparents of V-Collider structures. This is known as *graph moralisation*.

DEFINITION 12. *A **Moral Graph** is the minimial undirected graph which captures the same conditional independence relationships as a BN.*

Comparison of the moral graphs derived from the structure learning algorithms is used in Chapter 3 to explicitly compare the conditional relationships between the graph's nodes, whereas comparisons between CPDAGs account for both marginal and conditional dependencies.

2.1.3 Conditional Probabilistic Representation of Bayesian Networks

The selection of which Conditional Probability Distribution (CPD) to model the factors of BNs brings the discussion of network model specification to a close. This is a critical step in the

modelling procedure as it specifies the nature of conditional dependencies as well as the parameterisation of the model's representation.

Discrete CPDs can be simply represented by a tabular form, where a row corresponds to a node's outcome and the columns the configuration of its parents. Whilst this paramaterisation is simple and exact, it scales exponentially with network's size and the number of possible values each node can take. This makes it computationally difficult to work with when dealing with larger systems. One alternative for modelling discrete/categorical data using lower dimensional representations is to use Sigmoid Belief Networks (SBNs). SBNs use log-linear probabilistic representations for categorical variables and have found popularity in language and image modelling when utilised in Deep Bayesian Networks [17–19].

In many applications, the observed data are continuous and are modelled using probability density functions. The ubiquitous Gaussian distribution is commonly used to represent such systems, partially due to its well documented probabilistic properties as well as the large number of Gaussian hypothesis tests. These make Gaussian BNs highly receptive to be learned using a variety of different methods, as will be shown in Section 2.2. As well as the requirement of multivariate normal CPDs, Gaussian BNs assume that the mean of a child node is a linear sum of its parents:

$$P(X_{ch}|Pa(X_{ch})) = P(X_{ch}|\mu(Pa(X_{ch})), \sigma_{ch}) \propto \exp\left(-\frac{\left(X_{ch} - \mu(Pa(X_{ch}))^2\right)}{2\sigma_{ch}^2}\right) \tag{2.11}$$

where the mean

$$\mu(Pa(X_{ch})) = \sum_{X_i \in Pa(X_{ch})} w_{X_i} X_i + c. \tag{2.12}$$

For Gaussian BN's, the edge weights of the network connecting X_{ch} to $X_i \in Pa(X_{ch})$ are defined to be the linear weight w_{X_i} associated with the mean calculation in Equation 2.12.

DEFINITION 13. *In a multivariate Gaussian Bayesian Network, the weights $w_{i,ch}$ of directed edges pointing from $X_i \to X_{ch}$ $\forall$ $X_i \in Pa(X_{ch})$ are equal to the linear constants which characterise the mean of X_{ch}, as a function of its parents $X_i \in Pa(X_{ch})$.*

The assumption of linearity also implies that the global graph is modelled as a multivariate Gaussian. Consider a node with only root nodes as its parents. Since the roots are independent marginally Gaussian distributed, any linear combination of the parents will also be a Gaussian random variable, that is $\mu_{ch} \sim \mathcal{N}(\mu(Pa(X_{ch})), \sigma_{ch})$. Under the above assumption that a likelihood factor is Gaussian with a known σ and unknown μ, it can be shown that the corresponding conjugate prior on the mean is also a Gaussian. Taking the proof from [8], assuming $X_{ch}|\mu(\mathbf{X}_{Pa}) \sim \mathcal{N}(\mu, \sigma)$ and $\mu \sim \mathcal{N}(\mu_0, \sigma_0)$, multiplying the two distributions together to find the joint $\mathcal{N}(X_{ch}, \mu)$ and marginalising out μ yields the marginal of $P(X_{ch})$ which is also Gaussian:

$$P(X_{ch}) = \int \mathcal{N}(X_{ch}|\mu, \sigma_{ch})\mathcal{N}(\mu|\mu_0, \sigma_0)d\mu \sim \mathcal{N}\left(X_{ch}|\mu_0, \left(\sigma_x^2 + \sigma_0^2\right)^{\frac{1}{2}}\right). \tag{2.13}$$

It is noteworthy to mention that this is not guaranteed to be the case for all continuous CPD choices. The mathematical elegance of the Gaussian is what allows us to use Gaussian likelihood factors when we observe that the marginal data distributions are observed to be normal.

More critically the linear Gaussian assumption implies that the global distribution is also Gaussian.

THEOREM 14. *For any arbitrary DAG over D variables in which node i represents a single continuous random variable X_i, let the factor corresponding to X_i be normally distributed with a mean equal to a linear combination of its parent nodes $Pa(X_i)$:*

$$P(X_i|Pa(X_i)) \sim \mathcal{N}\left(\sum_{j \in Pa(X_i)} w_{ij} X_j + b_i, \sigma_i^2\right) \tag{2.14}$$

where w_{ij} and b_i parameterise the mean, and σ_i^2 is the variance of the conditional distribution. The full joint distribution of the graph is equal to the product of these factors:

$$P(\mathbf{X}) = \prod_{i=1}^{D} P(X_i|Pa(X_i)) \tag{2.15}$$

$$= A \prod_{i=1}^{D} \exp\left[-\frac{1}{2\sigma_i^2}\left(X_i - \sum_{j \in Pa(X_i)} w_{ij} X_j - b_i\right)^2\right] \tag{2.16}$$

where $\mathbf{X}$ is a vector of the network's nodes, and A is a normalisation constant. The joint distribution is the natural exponent of quadratic functions of the components in $\mathbf{X}$. Hence, the full joint distribution $P(\mathbf{X})$ of a BN parametrised by linear Gaussian factors is a multivariate Gaussian. (Taken from [8].)

Calculating the mean and covariance parameters of the multivariate Gaussian joint distribution $P(\mathbf{X}) \sim \mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma})$ requires the manipulation of marginal and conditional Gaussian distributions:

THEOREM 15. *If the marginal distribution of Y and the conditional distribution of vector variables $\mathbf{X}$ are:*

$$P(\mathbf{X}) = \mathcal{N}(\boldsymbol{\mu}, \Sigma_X) \tag{2.17}$$

$$P(Y) = \mathcal{N}(\mathbf{w} \cdot \mathbf{X} + c, \sigma_Y^2) \tag{2.18}$$

then the joint distribution is also Gaussian:

$$\begin{pmatrix} \mathbf{X} \\ Y \end{pmatrix} = \mathcal{N}\left(\begin{pmatrix} \boldsymbol{\mu} \\ \mathbf{w} \cdot \boldsymbol{\mu} + c \end{pmatrix}, \begin{pmatrix} \Sigma_X & \Sigma_X \mathbf{w}^\intercal \\ \mathbf{w}\Sigma_X & \sigma_Y^2 + \mathbf{w}\Sigma_X \mathbf{w}^\intercal \end{pmatrix}\right). \tag{2.19}$$

(Proof found in [8].)

Theorems 14 and 15 can be used to recursively build a global joint distribution across the entire distribution, and hence calculate the algebraic forms of the parameters $\boldsymbol{\mu}$ and $\boldsymbol{\Sigma}$ as a function of the weights and variances of the Gaussian factors.

2.2 Structure Learning

This section introduces the set of structure learning algorithms which aim to infer the graphical structure that best models a dataset. A central reason for learning a model's structure include *knowledge discovery*, since independence relationships between variables can be directly inferred from the network's structure. Early structure learning algorithms such as the Chow-Liu method, imposed restrictions (on top of acyclicity) that the learned graph is a tree shaped network [12, 20].

From the end of the 20th century, this restriction on returning a tree-structured network has been relaxed, and a variety of different methods for structure learning have been developed. They fall within four general categories:

1. **Constraint Based Algorithms**, which utilise conditional independence tests to identify conditional independence properties and then learn a network from them.

2. **Score Based Algorithms**, which use a goodness-of-fit criterion (or score) as a target such that maximisation of that score yields the best network fit to the data. The algorithms search through the space of possible networks and aim to identify the network with the maximum score

3. **Hybrid Algorithms**, which utilise a mixture of structure learning algorithms to initially reduce the search space (often by identifying the neighbourhood or Markov Blanket of each node) and then use a score algorithm to search over this reduced space and narrow down on the best fitted network.

4. **Regression Algorithms**, which involve optimisation of a loss function with a global optimum.

2.2.1 Constraint Algorithms

Constraint based approaches aim to identify sets of conditional independence properties and then attempt to narrow down and identify a network structure that satisfies all these constraints. To identify these independence relations, conditional hypotheses tests are used, with the null hypothesis being that two nodes are independent when conditioned on a node subset. Inductive Causation (IC) was the first constraint algorithm proposed by Verma and Pearl. It provided a framework that provably recovered the equivalence class of a network assuming a perfect conditional independence test (no type-1 or type-2 error) exists [15]. However, the number of independence tests required scaled exponentially with the size of the network, making it unusable on larger graphs. This motivated the development of algorithms which seek to improve on this exponential scaling. In the following subsections, we outline some of the more successful algorithms before validating them experimentally in Section 3.

DEFINITION 16. *An triplet of nodes $\{X, Y, Z\}$ are called an unshielded triple if both X and Z are both adjacent to Y but X is not adjacent to Z.*

PC-Stable

The PC algorithm (named after its creators Peter Spirtes Clark Glymour) was the first practical improvement of the IC algorithm [21]. It initially builds a complete undirected graph on the vertex set, and iteratively removes edges from the network by incrementally building the conditioning set.

Algorithm 1: PC-Stable Algorithm

Step 1: Find the skeleton $\mathcal{S}$ and separation sets using Algorithm 2;
Step 2: Orient unshielded triples in the skeleton $\mathcal{C}$ based on the separation sets;
Step 3: In $\mathcal{C}$ orient as many of the remaining undirected edges as possible by repeated applications of rules R1-R3;
R1: Orient $X_j - X_k$ into $X_j \to X_k$ whenever there is a directed edge $X_i \to X_j$ such that X_i and X_k are not adjacent;
R2: Orient $X_i - X_j$ into $X_i \to X_j$ whenever there is a chain $X_i \to X_k \to X_j$;
R3: Orient $X_i - X_j$ into $X_i \to X_j$ whenever there are two chains $X_i - X_k \to X_j$ and $X_i - X_l \to X_j$ such that X_k and X_l are not adjacent;
return *Output graph* $(\mathcal{C})$ *and separation sets*

Rather than scaling exponentially as per the IC algorithm, the worst case number of conditional hypothesis tests are

$$2\binom{n}{2}\sum_{l=1}^{k}\binom{n-1}{i} \leq \frac{n^2(n-1)^{k-1}}{(k-1)!} \sim O(n^{k+1}) \tag{2.20}$$

where k is the maximal degree of any vertex and n is the total number of vertices. The left hand term can be intuited by considering that in the worst case, all nodes have maximal degree k. For a particular conditioning set cardinality l, the worst case scenario is that all possible hypothesis tests need to be derived yielding the $\binom{n-1}{i}$ term. This is summed over k when considering the worst case scenario for all l. These tests are calculated for all ordered of pairs of adjacent vertices in the network which in the worst case is equal to $2\binom{n}{2}$, giving the result. This upper bound assumes that in the worst case for n and k, no two variables are d-separated by a set of cardinality fewer than k. Many real world graphs do not share this property, so in practice this upper bound is extremely loose [21].

The output of the PC algorithm is in general order dependent since the inferred conditional independencies depend on the order in which the variables are given, which can lead to highly variable results when learning high dimensional networks. The PC-Stable algorithm, an improved implementation of the original PC algorithm, mitigates this by removing edges only after the conditional dependencies of all nodes have been measured [22]. This improved implementation is used in the thesis. Algorithms 1 and 2 outline the core steps in the PC-stable algorithm.

The Grow-Shrink (GS) Algorithm

The GS algorithm seeks to learn a network structure by first determining the Markov Blanket of each node, and then resolving the set of blankets into a consistent PDAG [23]. As the name suggests, the construction of the MB consists of a simple forward selection process (growth phase) before pruning the blankets to construct the Markov Boundary. Algorithm 3 outlines the growth-shrink phases for identifying the MBs, and Algorithm 4 outlines the entire structure learning

Algorithm 2: Step 1 of the PC-Stable Algorithm

Find the skeleton and separation sets (sepsets) where $sepset(X_i, X_j)$ is the MB experimentally derived by the algorithm. Obtain conditional independence information among all variables in $\mathbf{V}$ and an ordering order (V) of the variables;
Form the complete undirected graph $\mathcal{C}$ on the vertex set $\mathbf{V}$;
Let $l = -1$;
repeat
  Let $l = l + 1$;
  forall *vertices* X_i *in* $\mathbf{C}$ **do**
    Let $a(X_i) = \text{adj}(\mathcal{C}, X_i)$
  end
  repeat
    Select a (new) ordered pair of vertices (X_i, X_j) that are adjacent in $\mathcal{C}$ and satisfy $|\text{adj}(\mathcal{C}, X_i) \setminus \{X_j\})| \geq l$, using order($\mathbf{V}$);
    repeat
      Choose a (new) set $\mathbf{S} \subseteq \text{adj}(\mathcal{C}, X_i) \setminus \{X_j\}$ with $|\mathbf{S}| = l$, using order($\mathbf{V}$);
      if X_i *and* X_j *are conditionally independent given* $\mathbf{S}$ **then**
        Delete edge $X_i - X_j$ from $\mathcal{C}$;
        Let $\text{sepset}(X_i, X_j) = \text{sepset}(X_j, X_i) = \mathbf{S}$;
      end
    until X_i *and* X_j *are no longer adjacent in* $\mathcal{C}$ *or all* $\mathbf{S} \subseteq adj(\mathcal{C}, X_i) \setminus \{X_j\}$ *with* $|\mathbf{S}| = l$ *have been considered*;
  until X_i *and* X_j *are no longer adjacent in* $\mathcal{C}$ *or all* $\mathbf{S} \subseteq adj(\mathcal{C}, X_i) \setminus \{X_j\}$ *with* $|\mathbf{S} = l$ *have been considered*;
until *all ordered pairs of adjacent vertices* (X_i, X_j) *in* $\mathcal{C}$ *with* $|adj(\mathcal{C}, X_i) \setminus \{X_j\} \geq l$ *have been ordered*;
return $\mathcal{C}$, *sepset*

process.

Algorithm 3: GS Estimation of Markov Blanket for the variable X

Initialise $\mathcal{S} = \emptyset$;
while *there exists* $Y \in G \setminus X$ *such that* $Y \not\perp\!\!\!\perp X \mid \mathcal{S}$ **do**
  $\mathcal{S} = \mathcal{S} \cup Y$
end
while *there exists* $Y \in \mathcal{S}$ *such that* $Y \not\perp\!\!\!\perp X \mid \mathcal{S} - Y$ **do**
  $\mathcal{S} = \mathcal{S} \setminus Y$
end
return *The Markov Blanket of* X: $MB(X) = \mathcal{S}$

Whilst the upper bound for the complexity of the GS algorithm is exponential with respect to the maximum degree of the nodes (for similar reasons as the PC algorithm), it can be shown that GS scales quadratically rather than polynomially with the number of nodes in (i.e. the size of) the graph. This yields a significant improvement in performance for large, sparse graphs (of which biological networks tend to be). Later adaptations of the algorithm sought to reduce the number of tests in the shrinking phase by ordering the variable inclusion in the grow phase by a probabilistic measure of dependence between two nodes, such as the p-value of a dependence test or the mutual independence between the two variables. One weakness of this approach is that when the MB of a node X contains coparents of X. In this case the coparents are very weakly

Algorithm 4: Markov Blanket Algorithm Template

forall $X \in G$ **do**
  Compute the Markov Blanket $MB(X)$
end
forall $X \in G$ **do**
  forall $Y \in MB(X)$ **do**
    Assign Y to be a direct neighbour of X if X and Y are dependent given $\mathcal{S}$ for all $\mathbf{T}$ where $\mathbf{T}$ is the smaller of $MB(X) \setminus Y$ and $MB(Y) \setminus X$
  end
end
forall $X \in G$ **do**
  forall $Y \in ne(X)$ **do**
    Orient $Y \rightarrow X$ if there exists a variable $Z \in ne(X) \setminus [ne(Y) \cup Y]$ such that Y and Z are dependent given $\mathbf{S} \cup X$ for all $\mathbf{S} \subseteq \mathbf{T}$ where $\mathbf{T}$ is the smaller of $MB(Y) \setminus X, |$ and $MB(Z) \setminus X, Y$
  end
end
while *Cycles still exist in the graph* **do**
  Compute the set of edges $\mathcal{C}$ that are part of a cycle;
  Remove from the current graph the edge in $\mathcal{C}$ that is part of the greatest number of cycles and put it in $\mathcal{R}$;
end
Insert each edge from $\mathcal{R}$ in the graph in reverse order of removal in the previous step, with the edge directionality reversed;
forall $X \in G$ **do**
  forall $Y \in ne(X)$ **do**
    if *no edge exists between* X, Y **then**
      if *there exists a directed path from* X *to* Y **then**
        Orient $X \rightarrow Y$
      end
    end
  end
end

associated with X when conditioned on the empty set. Recall that coparents in a V-Collider are marginally independent from one another, even though they are both in each other's MB (see Figure 2.3). This increases the chances of more false positive edges being included in the grow phase.

The Incremental Association Markov Blanket (IAMB) Algorithm

The IAMB family of algorithms build upon the GS algorithm by also aiming to discover the MBs of the graph's nodes before constructing a probabilistically consistent network [24, 25]. It is similar to the GS algorithm in the sense that it follows the same two-phase grow/shrink process. However, it seeks to mitigate the higher false positive rate by calculating the dependence of variables conditioned on the updated MB rather than on the empty set. This approach allows for the inclusion of coparents to be included in the MB much earlier than using the GS Algorithm. Algorithm 5 outlines the general procedure.

Algorithm 5: The IAMB Algorithm

Forward Phase;
forall $X \in |G|$ **do**
  Initialise an empty Markov Boundary set $MB(X) = \emptyset$;
  Initialise a set of eligible variables $\mathbf{E} = \mathbf{V} \setminus X$;
  repeat
    $Y = \text{argmax}_{Z \in \mathbf{E}} I(X, Z | MB(X))$;
    $\mathbf{E} = \mathbf{E} \setminus Y$;
    if $X \not\perp\!\!\!\perp Y \mid MB(X)$ **then**
      $MB(X) = MB(X) \cup Y$;
      $\mathbf{E} = \mathbf{V} \setminus X$
  until $\mathbf{E}$ *is empty*;
Backward Phase;
forall $A \in MB(X)$ **do**
  if $X \not\perp\!\!\!\perp A \mid (MB(X) \setminus A)$ **then**
    $MB(X) = MB(X) \setminus A$
return $MB(X)$

A number of modified IAMB algorithms exist that improve accuracy and runtime. The Interleaved IAMB (InterIAMB) algorithm interleaves the grow and shrink phases together to keep the estimates of the MB sets as small as possible [24]. After the addition of a node to the candidate $MB(X)$, the dependence of all nodes $Y \in MB(X)$ is tested. If the independence test deems $Y \not\perp\!\!\!\perp X \mid MB(X) \setminus Y$, then Y is removed from the candidate MB. The FastIAMB algorithm aims to increase algorithm speed by speculatively adding one or more nodes of highest significance to the MB during the grow phase without resorting after each modification [26]. An attempt to further reduce the False Discovery Rate was made with the IAMB-FDR algorithm, which replaces the condition for null hypothesis rejection $p_i < \alpha$ with a Benjamini-Hochberg adjustment:

$$\text{Reject } H_0 \text{ if: } p_i \leq \alpha \text{ (IAMB)} \tag{2.21}$$

$$\text{Reject } H_0 \text{ if: } p_i \cdot \frac{n}{i} \cdot \sum_{k=1} n \frac{1}{k} \leq \alpha \text{ (IAMBFDR)} \tag{2.22}$$

which, whilst being more conservative in edge selection, aims to provide a greater degree of confidence in the validity of identified edges [27, 28].

2.2.2 Score Based Algorithms

Score-based algorithms utilise heuristic search methods to identify networks which best satisfy some quantitative metric. Central to this approach is the selection of a score function that can score each network structure with respect to the training data. These algorithms then seek to search over the domain of possible networks for the network which maximises the score.

Given the exponential scaling of possible network configurations with the graph size, exploring every single network candidate is computationally unfeasible. The problem of finding the optimal BN is provably NP-hard, and so score-based methods utilise local heuristic search techniques to find good network fits [12]. Some of the candidate techniques that have been applied to the problem of structure learning include:

- Greedy Local Search algorithms, which explore the search space by making small modifications of the current network [29].
- Genetic Algorithms, which iteratively selecting the best models from a generation of candidate networks, before hybridising their features and creating a subsequent generation of candidate networks which have been randomly modified by a small amount [30].
- Simulated Annealing algorithms accept changes to the proposed network which decrease the network score with a probability inversely proportional to the decrease of the score [31]. This idea borrows heavily from the Metropolis-Hastings sampling algorithm.
- Reinforcement Learning (RL) algorithms treat the score as a reward which an agent seeks to maximise. Here, RL methods are used to derive a search strategy in contrast to conventional RL algorithms which generally aim to learn a policy [32].

This thesis investigates the effectiveness of two greedy local search algorithms: Hill Climbing (HC) and Tabu search. Greedy search algorithms involve the specification of an initial network and a computation of its score. *Adjacent* networks in the search space are identified and scored, with the highest scoring neighbour becoming the new candidate network. The choice of how the graph space is interconnected and what classifies as an adjacent/neighbouring network is a critical one. Too restrictive a criterion will limit the number of candidate options, whereas too many neighbours makes it computationally challenging to determine the next step in the search. A favoured connectivity criterion defines neighbouring networks as those which differ by either a single edge *addition*, *removal*, or *reversal* - in other words, all networks which differ from the current by a single edge modification. In the case of BN structure learning, only acyclic networks are considered in this step.

Score Functions

As mentioned above, the choice of score function is critical for the success of score-based algorithms. An intuitive but naive choice for the function is the global log likelihood of the network. We show that such a score function, while being information theoretically optimal, fails to return sparse

networks. The following derivation assumes a discrete system. Note that its conclusions are also valid for continuous distributions since such PDFs can be discretised into arbitrarily many buckets, with continuity being reached when the bucket widths tends towards zero. Let $N(x_i, \mathbf{y}_i)$ be the number of data points where node $X_i = x_i$ and the parents of X_i take the values $Pa(X_i) = \mathbf{y}_i$. The likelihood is

$$l(G, \hat{\theta_G}; \mathcal{D}) = \sum_{i=1}^{n} \left[\sum_{x_i} \sum_{\mathbf{y}_i \in Val(Pa(X_i))} N(x_i, \mathbf{y}_i) \log \hat{P}\left(x_i | \mathbf{y}_i, \hat{\theta}_{x_i}\right) \right] \tag{2.23}$$

where $\hat{P}\left(x_i | \mathbf{y}_i, \hat{\theta}_{x_i}\right)$ is the probability of observing x_i given $\mathbf{y}_i$ given the maximum likelihood estimate (MLE) parameterisation of the system. Note that for discrete distributions, this is simply equal to the empirical distribution.

The likelihood can be rearranged with the aim of isolating terms corresponding to the mutual information between variables in the graph given the distribution $\hat{P}$:

$$l(G, \hat{\theta_G}; \mathcal{D}) = \sum_{i=1}^{n} \left[\sum_{x_i} \sum_{\mathbf{y}_i \in Val(Pa(X_i))} N\hat{P}\left(x_i, \mathbf{y}_i, \hat{\theta}_{x_i}\right) \log \hat{P}\left(x_i | \mathbf{y}_i, \hat{\theta}_{x_i}\right) \right] \tag{2.24}$$

$$= N \sum_{i=1}^{n} \left[\sum_{x_i} \sum_{\mathbf{y}_i \in Val(Pa(X_i))} \hat{P}\left(x_i, \mathbf{y}_i, \hat{\theta}_{x_i}\right) \log \left(\frac{\hat{P}\left(x_i, \mathbf{y}_i | \hat{\theta}_{x_i}\right) \hat{P}\left(x_i | \hat{\theta}_{x_i}\right)}{\hat{P}\left(\mathbf{y}_i | \hat{\theta}_{x_i}\right) \hat{P}\left(x_i | \hat{\theta}_{x_i}\right)} \right) \right] \tag{2.25}$$

$$= N \sum_{i=1}^{n} \left[\sum_{x_i} \sum_{\mathbf{y}_i \in Val(Pa(X_i))} \hat{P}\left(x_i, \mathbf{y}_i, \hat{\theta}_{x_i}\right) \log \frac{\hat{P}\left(x_i, \mathbf{y}_i | \hat{\theta}_{x_i}\right)}{\hat{P}\left(\mathbf{y}_i | \hat{\theta}_{x_i}\right) \hat{P}\left(x_i | \hat{\theta}_{x_i}\right)} + \cdots \right. \tag{2.26}$$

$$\left. \cdots + \hat{P}\left(x_i | \hat{\theta}_{x_i}\right) \log \hat{P}\left(x_i | \hat{\theta}_{x_i}\right) \right] \tag{2.27}$$

Recognising the first term as the mutual information between X_i and $Pa(X_i)$ ($MI_P(X_i; Pa(X_i))$) and the second as the negative entropy of X_i ($H_P(X_i)$), the likelihood can be written as:

$$l(G, \hat{\theta}_G; \mathcal{D}) = N \sum_{i=1}^{D} \left[MI_P(X_i; Pa(X_i)) - H_P(X_i) \right] \tag{2.28}$$

The entropy terms are independent from the network structure and so can be ignored in network analysis. The mutual information terms (which are always non-negative) are a measure of dependence between X_i and $Pa(X_i)$, equalling zero only when they are independent. Therefore, using the likelihood as a score function will favour the selection of networks with edges connecting the most dependent variables.

Whilst using the likelihood as a score metric will lead to the insertion of edges between nodes which manifest some dependency characterised by the empirical distribution of the training data, these dependencies may not generalise effectively to new examples. More rigorously, it can be shown that introducing edges never decreases the likelihood by considering the resultant change to the mutual information. Before proving this, we quote a theorem taken from [33]:

THEOREM 17. *The Chain Rule for mutual information states that*

$$MI_P(Y; X_1, \ldots, X_N) = \sum_{i=1}^{N} MI_P(Y; X_i | X_1, \ldots, X_{i-1}) \tag{2.29}$$

where

$$MI_P(Y; X|Z) = \sum_{x,y,z} P(X, Y, Z) \log \frac{P(X, Y|Z)}{P(Y|Z)P(X|Z)} \tag{2.30}$$

Using the above, we can now prove the following:

THEOREM 18. *For any set of variables* $\mathbf{X}, \mathbf{Y}, \mathbf{Z}$ *and distribution* P, $MI_P(\mathbf{X}; \mathbf{Y} \cup \mathbf{Z}) \geq MI_P(\mathbf{X}; \mathbf{Y})$

Proof. Application of the chain rule in Theorem 17 allows for the following rearrangement:

$$MI_P(\mathbf{X}; \mathbf{Y} \cup \mathbf{Z}) = MI_P(\mathbf{X}; \mathbf{Y}) + MI_P(\mathbf{X}; \mathbf{Z}|\mathbf{Y}). \tag{2.31}$$

Since all mutual information terms are non-negative, the $MI_P(\mathbf{X}; \mathbf{Z}|\mathbf{Y}) \geq 0$. Applying this condition into the equation above gives the desired result. □

It can be concluded that the network which maximises the network will demonstrate conditional independence relations only when that relation holds *exactly* in the empirical (training data) distribution. Since in practice this rarely holds, the network that maximises the likelihood will be fully corrected. This is an example of overfitting in the domain of network structure learning.

The development of Bayesian scoring functions aim to mitigate this issue. Prior uncertainty over the domain of possible network structures and their parameters is captured in a structure prior $P(G)$, and a parameter prior $P(\theta|G)$ respectively. The Bayesian "goal" is to identify the posterior distribution of the graph; the distribution of graphs conditioned on the observed data, $\mathcal{D}$. Using Bayes rule:

$$P(G|\mathcal{D}) = \frac{P(\mathcal{D}|G)P(G)}{P(\mathcal{D})} \tag{2.32}$$

$$\implies \log P(G|\mathcal{D}) = \log P(\mathcal{D}|G) + \log P(G) - \log P(\mathcal{D}) \tag{2.33}$$

where the $P(G|\mathcal{D})$ marginalises out the unknown parameters and is known as the marginal likelihood of the data given the structure:

$$P(\mathcal{D}|G) = \int P(\mathcal{D}|\theta, G)P(\theta|G)d\theta \tag{2.34}$$

Since the $P(\mathcal{D})$ term in Equation 2.33 is independent of graph structure, it suggests that a suitable Bayesian score functions could be:

$$Score(G|\mathcal{D}) = \log P(\mathcal{D}|G) + \log P(G) \tag{2.35}$$

The $\log P(G)$ allows for the construction of an informative structure prior. For example, one may have a preference for sparse rather than dense graphical structures. In practice however, this prior has an insignificant effect on the Bayesian score, which is dominated by the $\log P(\mathcal{D}|G)$ term.

It is important to emphasise the difference between maximising with respect to the marginal likelihood $P(\mathcal{D}|G)$ rather than the standard likelihood $P(\mathcal{D}|\theta, G)$. The former is the expected

value of the standard likelihood taken over $P(\theta|G)$ and so is more resilient to isolated maxima in the likelihood. Bayesian Occam's Razor provides a justification for why sparser models tend to be returned [34]. Since simpler models are likely to generate a smaller number of datasets, such models will concentrate their probability around this small subset whereas more complex models will have a less peaked marginal likelihood. This automatically penalises the selection of more complex models and encourages the selection of simpler models if they can well explain the data. Another intuitive justification for the above is that by averaging over all model parameterisation, model selection is less affected by the single best fitting parameters and cares more about the fit of all possible parameterisations of a BN.

The log marginal likelihood can be decomposed across the network edges as follows:

$$\log P(\mathcal{D}|G) = \sum_{i=1}^{D} \int P(X_i|Pa(X_i), \theta_i) P(\theta_i|G) d\theta_i. \tag{2.36}$$

We now outline two posterior score functions that have exact solutions for categorical and Gaussian CPDs. In the case of discrete multinomial data, a common choice of prior on the parameters is the Dirichlet distribution:

$$P(\theta|\alpha) = \frac{1}{B(\alpha)} \prod_{i=1}^{K} \theta_i^{\alpha_i - 1} \tag{2.37}$$

where $B(\cdot)$ is the Beta function. Each node in a multinomial network has a set of distributions for each set of values conditioned on the parents $\theta_{X_i|Pa(X_i)}$ with a hyperparameter vector $\alpha_{X_i|Pa(X_i)}$. The **Bayesian Dirichlet Equivalence** (BDe) score assigns to the hyperparameters the value:

$$\alpha_{ijk} = \alpha_{X_i=k|Pa(X_i)=\mathbf{y}_j} = N' \tilde{P}(X_i = k, Pa(X_i) = \mathbf{y}_i | G_{prior}) \tag{2.38}$$

$$\alpha_{ij} = \sum_k \alpha_{ijk} \tag{2.39}$$

where the indexes denote the situation where the i^{th} node takes the value k under the j^{th} configuration of X_i's parents. $\tilde{P}$ is represented by a BN with prior structure G_{prior}, and N' is referred to as the imaginary dataset size [12, 31]. N' effectively parameterises the sparsity of the Dirichlet prior, with low values pushing the sampling distribution to extremes where the majority of the probability weight is assigned to a single outcome. Higher N' parametrises the scenario where probabilities for the multinomial distribution are evenly distributed across all outcomes.

In practice, this prior structure G_{prior} is set to an empty graph to represent the scenario where no prior knowledge of the graph is known. $\tilde{P}$ then becomes a uniform marginal variable over each variable:

$$\tilde{P}(X_i = x_i, Pa(X_i) = \mathbf{y}_i | G_{prior}) = \frac{1}{|\theta_i|}. \tag{2.40}$$

An analytical solution for the marginal likelihood exists under these assumptions and is known as the BDe score:

$$BGe = \prod_{i=1}^{D} \prod_{j \in Pa(X_i)}^{M_i} \frac{\Gamma(\alpha_{ij})}{\Gamma(\alpha_{ij} + N_{ij})} \prod_{k}^{R_i} \frac{\Gamma(\alpha_{ij} + N_{ijk})}{\Gamma(\alpha_{ijk})} \tag{2.41}$$

where R_i is the number of categories X_i can take, M_i is the total set of values the parents $Pa(X_i)$ can take. Let N_{ij} be the number of times that the parents $Pa(X_i)$ take the value j, and N_{ijk} be the number of times $X_i = k$ given a parent configuration j.

The **Bayesian Gaussian Equivalence (BGe)** score is another exact derivation of the marginal likelihood originally derived for a multivariate Gaussian BN. It is, however, more generalisable and can be used to derive solutions for any marginal likelihood model if the global probability distribution satisfies the following five assumptions:

1. Complete model equivalence between families of CPDAGs
2. For every two equivalent DAG models there exists a one-to-one mapping between their parameters such that their likelihoods are equal
3. For every two DAG models such that a node X_i has the same parents in both nodes, the local factor distributions are the same
4. For every two DAG models such that X_i have the same parents, the local distributions for X_i in both models are the same
5. For every DAG model, the global parameter priors are fully independent, $p(\theta_G|M) = \prod_{i=1}^{n} p(\theta_i|M)$.

These assumptions (which are satisfied by multinomial and multivariate t-distribution BNs as well as multivariate Gaussians) permit the construction of parameter priors from which the marginal likelihood can be calculated. For Gaussian DAGs, an analytical expression for the marginal likelihood can be derived by assigning a normal-Wishart prior on the precision matrix and thus on the network structure. The derivation and final analytical form are more complex than the BDe and is not be stated here - interested readers can find it in its flagship papers referenced here [35, 36].

Another popular score function is the **Bayesian Information Criterion** (BIC) which has been used as a measure of model fit across a variety of statistical domains. Whilst we outline below how it can naturally arise in the domain of structure learning, note that it is frequently used for modelling both discrete and continuous data as opposed to the raw likelihood. In the context of Bayesian Network structure learning, it can be derived by considering the BDe score in the limit of infinite data:

$$\log P(\mathcal{D}|G) \xrightarrow{N\to\infty} \log P(\mathcal{D}|\hat{\theta}, \mathcal{D}) - \frac{\log N}{2}|\theta| + O(1) \tag{2.42}$$

where $|\theta|$ is the number of parameters used by the network. This approximation helps define the BIC score:

$$BIC(G|\mathcal{D}) = l(\hat{\theta}; G, \mathcal{D}) - \frac{\log N}{2}|\theta| \tag{2.43}$$

The second term acts as a regularisation term which becomes more significant as the training data increases.

When deriving scoring functions, one must consider how they score networks from the same equivalence class.

DEFINITION 19. *A scoring function $S(G|\mathcal{D})$ satisfies score equivalence if all networks from the same equivalence class as G have the same score for all datasets $\mathcal{D}$.*

It can be shown that the likelihood, BIC, BDe and BGe are all score equivalent. For proofs, we refer the reader to [12, 35, 36].

A further consideration is whether the score function is maximised by the true network (i.e. whether we are guarenteed to recover the true network in the limit of infinite data). This motivates the necessity for scoring functions to be *consistent*.

DEFINITION 20. *Assume dataset $\mathcal{D}$ is generated from a distribution which is perfectly mapped by the BN G. A scoring function is said to be* ***consistent*** *if, in the limit of infinite data, the structure G almost certainly maximises the score, and all structures which are not in the equivalence class as G have a strictly lower score. Consistent scoring functions are necessarily score equivalent.*

We outline a proof taken from [12] for the consistency of the BIC network below.

Proof. Let a dataset $\mathcal{D}$ be drawn from a network G that is a map of the true underlying distribution P. Consider a graph G^* that implies an independence assumption that G does not hold. G^* cannot be in the equivalence class of G, and hence it cannot be a maximum likelihood model. Therefore, we must have:

$$\sum_{i=1}^{D} MI_P(X_i; Pa(X_i)^G) > \sum_{i=1}^{D} MI_P(X_i; Pa(X_i)^{G^*}) \tag{2.44}$$

where $Pa(X_i)^G$ are the parents of X_i in the graph G.

As the datasize N tends to infinity, the empirical distribution $\hat{P}$ will almost certainly converge to the true distribution P. Therefore, we can use Equation 2.28 to express the differences in likelihood score between G and G^* for large N:

$$l(G, \hat{\theta}_G; \mathcal{D}) - l(G^*, \hat{\theta}_{G^*}; \mathcal{D}) \approx N \sum_{i=1}^{D} \left[MI_P(X_i; Pa(X_i)^G) - MI_P(X_i; Pa(X_i)^{G^*}) \right] = N\Delta \tag{2.45}$$

where $\Delta = \sum_{i=1}^{D} \left[MI_P(X_i; Pa(X_i)^G) - MI_P(X_i; Pa(X_i)^{G^*}) \right]$. Therefore, by using Equation 2.43, we can express the asymptotic difference between the BICs of the two graphs as:

$$BIC(G; \mathcal{D}) - BIC(G^*; \mathcal{D}) = N\Delta + \frac{\log N}{2} \left(|G^*| - |G| \right) \tag{2.46}$$

where $|G|$ is the number of independent parameters used to specify the model G. Since the first term grows much quicker than the second, its effect will dominate for large N and so $BIC(G; \mathcal{D}) > BIC(G^*; \mathcal{D})$.

Now, assume that G^* implies all independence relationships in G, but that G implies an independence relation that G does not. In this case, G^* can represent any distribution that G can, and so Theorem 18 implies that:

$$l(G, \hat{\theta}_G) - l(G^*, \hat{\theta}_{G^*}) \xrightarrow{N \to \infty} 0 \tag{2.47}$$

and so asymptotically, we have

$$BIC(G;\mathcal{D}) - BIC(G^*;\mathcal{D}) \xrightarrow{N\to\infty} \frac{\log N}{2}(|G^*| - |G|) \tag{2.48}$$

Since G^* makes fewer independence assumptions than G, it must be parameterised by a larger set of parameters (since it is more flexible at modelling a wider variety of distributions). Therefore, $|G^*| > |G|$ and so $BIC(G;\mathcal{D}) > BIC(G^*;\mathcal{D})$.

We have shown that in both cases where G^* implies an independence assumption that G does not hold and vice versa, $BIC(G;\mathcal{D}) > BIC(G^*;\mathcal{D})$. Hence, in the limit where $N \to \infty$, the BIC is maximised by the true network G (and networks in its equivalence class) and so the BIC is a consistent score function. □

A corollary of the above is that that since the Bayesian score is asymptotically equal to the BIC, all Bayesian scores (including the BDe and BGe) are consistent score functions. This theoretical property justifies their use in score-based algorithms, although it is worth noting that the above proof does not provide much information about networks learned with small datasets.

Hill Climbing

The Hill Climbing (HC) search algorithm makes progressive modifications to the candidate network if the new network has a higher score than the prior candidate. There is no record of previously visited networks, nor does it look ahead beyond the immediate neighbours of the current state.

Algorithm 6: Hill Climbing Algorithm

```
Initialise a network C over a vertex set V and compute its score S_C = Score(C)
Set maxscore = S_C repeat
    forall neighbouring graphs with a single graph addition, deletion, or reversal that does
     not yield an acyclic network. do
        Calculate the score of the candidate network C^cand, S_{C^cand} = Score(C^cand)
        if S_{C^cand} > S_C then
            Set C = C^cand and S_C = S_{C^cand}
        end
    end
until S_C < maxscore;
return Output graph (C)
```

A critical problem with the HC algorithm is its tendency to identify and subsequently fail to escape from local maxima. To mitigate this, HC is often initialised from multiple random restarts. Once a single pass of an algorithm returns a locally optimal graph, a number of graph edges in the locally optimised graph are randomly perturbed to construct a new starting network, and a fresh HC is run.

Tabu Search

Tabu Search is a variant of the HC algorithm. It seeks to improve search efficiency and allow for the escaping of local maxima through use of a *tabu list*. This list (of length l) maintains a record of the previous l network states visited during the search and prevents them from being revisited

while they remain on the list. Many academic reviews have investigated the optimal choice of tabu list length. In general, it has been found that a small tabu list length allows for greater exploration of search space, whereas larger lengths allow for the breaking away from local maxima since many states around this maxima are on the list. A further consensus in the literature is that the optimal length of the tabu list is roughly in the order of the size of the vertex set. As a result, the hyperparameter search in subsequent sections explore tabu ranges which are fractions of the graph size.

2.2.3 Hybrid Algorithms

The development of hybridised structure learning algorithms sought to combine the benefits of both constraint and score-based learning algorithms and by doing so offset their respective weaknesses. As mentioned in Section 2.2.2, one problem with heuristic search methods is that the search space of possible networks is exponentially large. Particularly in the case of limited data, these methods can identify non-global optima that may differ significantly from the true network and so learned networks may vary considerably when multiple iterations are run. On the otherhand, constraint algorithms seek to identify a set of Markov Blankets without quantitatively assessing the importance of specific networks in the context of the global network.

Hybrid methods consist of an initial *restrict* phase which reduces the set of candidate parents for each node by using a constraint algorithm, greatly reducing the search space of potential networks. The second *maximise* phase uses score or regression (see Section 2.2.4) optimisation methods to maximise a score or loss function within the reduced subspace identified in the restrict phase. The Sparse Candidate (SC) algorithm repeats the restrict-maximise phases until either the learned network at each phase converges or no network improves the score. In this thesis, only a single experimental iteration of the restrict/maximise phase will be conducted due to time constraints, but it should be noted that repeated iterations can sometimes lead to better results if navigating a search space where several local optima exist.

Generally, the restriction algorithms that have shown success are local-discovery constraint algorithms that learn an undirected skeleton. The **Max-Min Parents and Children** algorithm constructs a candidate set of parent and children (CPC) for target node X using forward selection which progressively includes the node which exhibits the maximum association to X conditioned on CPC set. The association is often measured using a statistical measure such as α or mutual information. After these sets have converged, a backwards removal phase is run which removes a node in the candidate set if it can be d-separated from X on all subsets of the CPC. The MMPC method is outlined in Algorithm 7.

The **Semi-Interleaved HITON PC** (SI-HITON-PC) algorithm uses a forward selection phase ordered by the pairwise marginal associations with the target node X and the candidate node for inclusion, Y. If a newly included node in the CPC is found to be conditionally dependent on X conditioned on any of the CPC subsets, the node is then eliminated from consideration. SI-HITON-PC is a comparatively fast algorithm that scales linearly with the size of the node set, and scales as $O(|PC(X)|^2 2^{|PC(X)|})$ with the size of the Markov Blanket. This exponentiality is not an issue if the graph is sparse, but the run time becomes a significant issue with denser networks. The SI-HITON-PC method is outlined in Algorithm 8.

Algorithm 7: MMPC Algorithm

Forward Phase:
forall *nodes* $X \in G$ **do**
  Initialise $CPC_X = \emptyset$
  repeat
    forall *variables* $Y \in G$ **do**
      Find $minassocset(Y)$ which is the subset $\mathbf{s}$ of CPC_X that minimises the independence association between X and Y, conditional on $\mathbf{s}$: $I(X, Y|\mathbf{s})$
      Identify the node $Z \in G \setminus (X \cup CPC_X)$ which maximises the independence association
      if X *is dependent on* Y *conditional on* $minassocset(Y)$ **then**
        $CPC_X \leftarrow CPC_X \cup Y$
      end
    end
  until CPC_X *is unchanged*;
end
Backward Phase:
forall $Y \in CPC_X$ **do**
  if *for every subset* $\mathbf{s} \subseteq CPC_X$ *such that* $Y \perp\!\!\!\perp X \mid \mathbf{s}$ **then**
    Remove Y from CPC_X
  end
end
return CPC_{X_i} *sets for all* $X_i \in G$

Algorithm 8: SI-HITON-PC Algorithm

forall *nodes* $X \in G$ **do**
  repeat
    Initialise $CPC_X = \emptyset$
    Identify the node $Y \in G \setminus CPC_X$ which maximises the marginal dependence association between Y and X and add it to CPC_X: $CPC_X \leftarrow Y \cup CPC_X$
    if *for any subset* $\mathbf{s} \subseteq CPC_X$ *there exists a node* $Y \in CPC_X$ *such that* $X \perp\!\!\!\perp Y \mid (CPC_X \setminus Y)$ **then**
      Remove Y from CPC_X and do not consider X for further admission to the set
    end
  until *No further nodes are left to consider*;
end
return CPC_{X_i} *sets for all* $X_i \in G$

A recent addition to this family of methods is the **Hybrid Parents and Children** (HPC) algorithm which can be viewed as an ensemble method which combines multiple Parent-Children (PC) learners to construct a strong learner. It starts by identifying a superset of PC nodes (SPC) and Parent-Spouse nodes (SPS), the latter being the set of nodes which share children with the target node X. It then runs the Incremental Association Parent Child FDR (IAPCFDR) algorithm (an undirected variant of the IAMBFDR algorithm, see Section 2.2.1) on the union $SPC \cup SPS$ to minimise the number of false positives. If IAPCFDR identifies a node as being in the neighbourhood of the target node, it is included in the final PC set. The pseudocode for HPC is outlines in Algorithm 9. A more detailed breakdown can be found in the original paper [37]. The HPC algorithm when hybridised with HC is referred to by its creators as the H2PC algorithm.

Algorithm 9: HPC Algorithm (Pseudocode)

forall *nodes* $X \in$ **do**
  Initialise $PC_X = \emptyset$
  Calculate the superset of parents and children (SPC) of X using a weak forward-backward selection process based on SI-HITON-PC. For each node in SPC, identify whether they can be d-separated from X
  Calculate the superset of spouses of X (SPS) by assessing whether nodes in SPC outside of the become conditionally dependent using the SPC and d-separation results from the previous step
  forall *nodes* $Y \in T \cup SPC \cup SPS$ **do**
    if X *is a member of the neighbourhood identified by IAPCFDR* **then**
      $PC_X \leftarrow PC_X \cup Y$
    end
  end
end
return PC_{X_i} *sets for all* $X_i \in G$

2.2.4 Regression Methods

The approach of learning BN structures using regression methods is a fairly novel addition to the family of structure learning algorithms. This class of algorithms are characterised by defining an objective function with a global global minimum that under theoretical limits converges to the true network structure. This continuous function is then minimised using optimisation methods such as gradient descent. When applied to learning Gaussian BNs, the chosen objective function is generally some form of the least squares (LS) loss:

$$l(W) = ||X - XW||^2 + \lambda||W||_1 \tag{2.49}$$

where X is the data matrix and W is the network's edge weight matrix. To ensure graph sparsity, a lasso regularisation term is chosen such that some of the edges are driven to zero rather than very small but non-zero values. Assuming the true network is Gaussian, choice of the above loss function is constructed under the assumption that the mean of a node is equal to a linear combination of its parents, and so aligns with the previous assumptions of earlier structure learning methods that a linear Gaussian BN accurately models a target system [19].

This report assesses a very recent state of the art method called the NOTEARS (Non-combinatorial Optimization via Trace Exponential and Augmented lagRangian for Structure learning) algorithm which reframes the problem from a regression, subject to combinatorical graph constraints, to a fully continuous Lagrangian optimisation problem [38]. The authors propose the single smooth equality constraint $h(W) = 0$ which hold iff W is acyclic, and has computable derivatives. The chosen function is

$$h(W) = \text{Trace}\left(\exp(W \circ W)\right) - d = 0 \tag{2.50}$$

where d is the number of features/nodes in the network, and W are the linear weights parameterising the BN.

The motivation behind this can be intuited from the traced Taylor expansion of $\exp(W \circ W)$:

$$\text{Trace}\left(\exp(W \circ W)\right) = \text{Trace}(I) + \text{Trace}(W \circ W) + \frac{\text{Trace}((W \circ W)^2)}{2} + \dots \tag{2.51}$$

Since $W \circ W \in \mathbb{R}_+^{d \times d}$, the diagonal elements of $(W \circ W)^k$ will be zero if there are no k cycles in the system, which requires $\text{Trace}((W \circ W)^k) = 0$. Ensuring that there are no cycles of any order in the system, we have that a Directed Acyclic Network (DAG) satisfies

$$\text{Trace}\left(\exp(W \circ W)\right) = d \tag{2.52}$$

where the d here arises from the trace of the identity. Further implementation details of the optimisation process and proofs of validity are beyond the scope of this report and can be found in the original paper.

It is worth noting that the NOTEARS algorithm can be used with any smooth loss function. However, the extensively studied statistical properties of the LS loss makes is an very suitable choice for the algorithm. It can be shown that the LS loss function provably recovers the true DAG with high probability on finite samples for both Gaussian and non-Gaussian linear systems - the only required assumption is that the expected value of a child is a linear combination of the parent. The generalisability of the NOTEARS algorithm to non-Gaussian linear systems has been demonstrated through synthetic experiments. These results can be found in the original paper.

Chapter 3

Synthetic Data Experiments

This chapter aims to answer the following questions:

- How does the choice of hyperparameter affect algorithm performance?
- How does network size affect structure learning for a fixed number of samples?
- How does sample size affect the accuracy of a learned network?

Prior to applying the algorithms defined in Section 2.2 to experimental Schizophrenia data, their performances were compared using simulated data generated from known networks. These fall in two categories:

1. Randomly created DAGs which are artificially generated given a sparsity parameter η and number of nodes. The sparsity controls the probability of an edge being present between any two nodes.
2. The ECOLI biological network (46 nodes, 70 edges) that has been inferred through biological experiments [39]. This network was sourced from the *bnlearn* R package [40].

The algorithm used to generate the randomly generated DAGs is as follows [41]. To construct a random DAG and generate N data samples from it, the following procedure was followed:

1. Initialise an empty graph of D nodes, labelling them $d \in \{1, \ldots, D\}$
2. Recursively iterate in ascending order from $d \in \{1, \ldots, D\}$ and assign a directed edge from each node d to nodes $d+1, \ldots, D$ with a probability η. This ensures the resultant adjacency matrix, $\boldsymbol{M}$ is upper triangular, and so is acyclic.
3. For each edge $M_{ij} \neq 0$, assign a weight to each edge connection to create a weight matrix W_{ij}. For simplicity, the implementation of the algorithm used in this thesis assigns a constant value of 0.5 to each edge (i.e. $W_{ij} = 0.5M_{ij}$).
4. For each data sample n from $n = 1, \ldots, N$, randomly draw the data sample for the first node from a standardised normal: $X_{n1} \sim \mathcal{N}(0, 1)$. Generate subsequent node data recursively: $X_{nd} = \sum_{i<d} M_{id} X_{ni} + \epsilon_{nd}(0, 1)$ where $\epsilon_{nd}(0, 1) \sim \mathcal{N}(0, 1)$.

The data from the established biological networks were generated from experimentally derived weights and standard deviations. To simulate the stochasticity of real data, additional noise added to the linear combination of parent values was instead drawn from $\epsilon_{nd} \sim \mathcal{N}(\mu_d, \sigma_d^2)$, where μ_d and σ_d^2 were calculated using linear regression. This contrasts with the data from the random DAGs which have fixed constant weights $w = 0.5$ and noise drawn from standard Gaussians.

The choice of drawing synthetic data from a continuous multivariate Gaussian distribution was made since these graphs are most similar to the kind of networks observed in biological domains. It is worth emphasising that true biological networks do not follow a linear multivariate Gaussian distribution and can sometimes deviate quite significantly from such assumptions. The approach of how to deal with such exceptions (such as highly skewed data) remains an open question. One example of the challenges posed from trying to construct continuous probabilistic graphical models from non-Gaussian data is examined in Section 4.3 and Chapter 7. However, since all the algorithms reviewed in Section 2.2 have been theoretically constructed under the assumption that the underlying model is a multivariate Gaussian, it makes sense to validate their performance against data generated from such a model. As it is possible that these algorithms will be applied to high dimensional datasets, an understanding of which algorithms perform best with particular datasizes is essential before applying them to experimental data.

The structure of the analysis is as follows. Datasets are drawn from a variety of generated DAGs with datasize N and node number D being sampled uniformly from $N \in \{40, \ldots, 200\}$ and $D \in \{20, \ldots, 120\}$. Since many biological networks are sparse ($|E| \sim O(D)$), the sparsity was set to $\eta = \frac{3}{2D}$ as the ratio $\frac{|E|}{|D|}$ emulates those seen in real biological systems as closely as possible (see Figure 3.1). These datasets are used to perform hyperparameter searches on each of the algorithms. The effect of datasize on the learned network is investigated using datasizes of sizes $N \in \{50, 100, 250, 500\}$, and is covered in Section 3.3. The impact of graph size on algorithm performance is assessed in Section 3.4.

3.1 Validation Metrics

This section outlines the key metrics used in this thesis for validating algorithm performance. The Hamming Distance (HD) provides a global measure of structure learning success by counting the number of edge differences between the undirected edges of two BN skeletons. The Structural Hamming Distance (SHD) accounts for differences in edge directionality by counting mismatched edges, where an edge is defined as *mismatched* if there is:

(a) An edge between two nodes $d_i \to d_j$ in G_1 that is not present in G_2 or vice versa

(b) An edge between $d_i \to d_j$ exists in both G_1 and G_2, but the direction of the edge differs between the two (either they point in different directions, or one edge is directed but the other is undirected)

The HD can be used to measure the accuracy of inferred conditional dependencies between two BNs by measuring the HD between their moralised graphs. To account for differences in network size, we report the normalised SHD $\left(\frac{\text{SHD}}{|E|}\right)$ and HD $\left(\frac{\text{HD}}{|E|}\right)$ which normalise over the edge count of the true network.

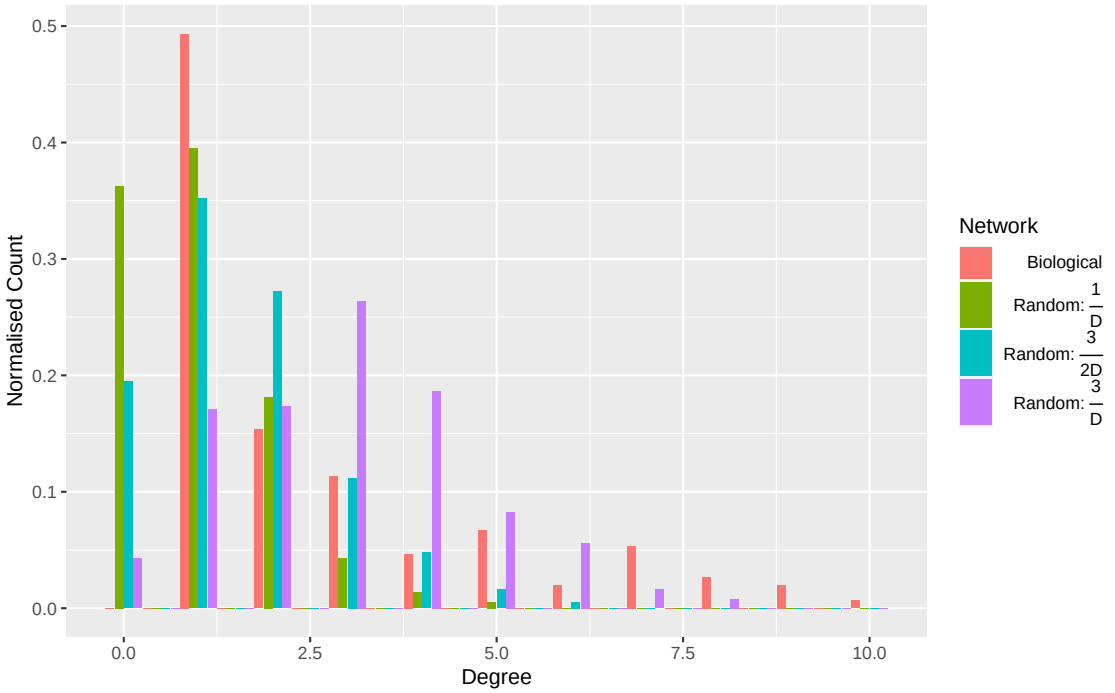

Figure 3.1: Comparison of network degrees between sparse randomly generated DAGs and experimentally validated biological networks taken from the *bnlearn* package. Note that half the nodes in the biological networks have a degree of one with the remaining nodes being distributed relatively evenly between the higher degrees. Conversely, the random DAGs have a more continuous variation in degree. Notably, these networks have a significant number of empty nodes, a property not observed in their biological counterparts.

We also use more specific metrics which attempt to characterise how responsive a method is at recovering existing edges as well as how prone it is to learning incorrect pathways. By framing the assignment of each edge into a classification problem, conventional classification terms can be used to quantitatively assess the accuracy of a learned network:

- True Positives (TP): the sum of directed edges which are present in both the true and learned networks
- True Negatives (TN): the sum of edges absent in both the true and learned networks
- False Positives (FP): the sum of edges present in the true network which have not been identified in the learned network
- False Negatives (FN): the sum of edges identified in the learned network which are absent in the true graph.

Since biological networks tend to be sparse ($|E| \sim O(D)$) rather than dense ($|E| \sim O(D^2)$), the number of TNs will vastly outnumber the other three metrics since in both cases the majority of possible edges will be absent in both the true and learned graphs. This imbalance makes it challenging to use metrics such as a F1 score or the more robust Matthews Correlation Coefficient (MCC), as these are dominated by the TNs and would not be sensitive enough to capture subtle variations in the TPs, FPs, and FNs [42].

As a result, this thesis uses the *False Discovery Rate* (FDR) and *Sensitivity* which are conventionally used in the literature and aim to effectively summarise the success of structure learning without using TNs:

$$\text{FDR} = \frac{\text{FP}}{\text{FP} + \text{TP}} \qquad \text{Sensitivity} = \frac{TP}{TP + FN} \tag{3.1}$$

Note that the above metrics map closely to the conventional Precision/Recall metrics (FDR = 1 − Precision and Sensitivity = Recall), but the thesis will use the former terms to align closer with the terminology used in the literature.

Apart from the NOTEARS algorithm, selecting the optimal hyperparameter by observing the histograms of key metrics proved a challenge, since their standard deviations were much larger than the differences between their means. Both two-sample mean t-tests and non-parametric tests with α of 0.8 were unable to distinguish between the hyperparameter performances of constraint, score, and hybrid based algorithms. Because of the similarity in performance variability across algorithm performance, the optimal hyperparameter was chosen to be that which minimised the mean SHD, which is illustrated as a vertical line in the relevant histograms. As well as providing general guidance to readers on how the assessed algorithms perform under different conditions, the conclusions of this chaper are used to decide which algorithms to apply to the experimental Schizophrenia data.

3.2 Hyperparameter Search

Prior to analysing the performance of all algorithms in detail, a summary of the best performing algorithms along with their optimal hyperparameter settings are outlined here.

Family	Best Algorithm	Random DAG $\frac{SHD}{\lvert E\rvert}$	Random DAG $\frac{HD}{\lvert E\rvert}$	Random DAG FDR	Random DAG Sensitivity	ECOLI $\frac{SHD}{\lvert E\rvert}$	ECOLI $\frac{HD}{\lvert E\rvert}$	ECOLI FDR	ECOLI Sensitivity
Constraint	InterIAMB	0.93	1.49	0.33	0.64	1.16	1.29	0.28	0.50
Score	HC	0.85	0.95	0.33	**0.79**	1.26	2.89	0.37	**0.62**
Hybrid	H2PC	0.53	0.99	0.18	0.74	**1.09**	**1.06**	**0.27**	0.57
Regression	NOTEARS	**0.48**	**0.84**	**0.12**	0.76	1.22	1.10	0.35	0.48

Table 3.2: This table presents the metrics from the best performing algorithms from each family. These results were obtained using the optimal hyperparameters which are presented in Table 3.2. The best performing metrics are emboldened.

Best Algorithm	Optimal Hyperparameters
InterIAMB	$\alpha = 0.05$
Hill Climbing	Restarts= 50 (Max), Perturbations=50 (Max), BGe, Empty Initialisation
H2PC	Restrict Phase: $\alpha = 0.2$, Maximise Phase: Use HC hyperparameters
NOTEARS	$\lambda_1 = 0.1$

Table 3.3: Optimal hyperparameter settings from the best performing algorithms in each family.

3.2.1 Constraint Algorithms

Constraint based algorithms rely on hypothesis-testing to determine the condition dependencies of nodes in the graph. The test's p-value or α for rejection of these tests acts as a hyperparameter for these algorithms. It may be expected that alpha acts as a sort of regularisation - using very small α often yield network connections with very high confidence, but this high selectivity risks returning a much sparser network with many FPs. Conversely, an algorithm with a large α may be identify many more TPs but increases the amount of FPs than its lower p counterpart. Since the cost of experimentally validating a genetic pathway is high, a key priority is identifying algorithms with low FDR values.

Figure 3.4 plots the SHD, HD, FDR and sensitivity for each algorithm across the hyperparameter range $\alpha \in \{0.001, 0.005, 0.01, 0.05, 0.1, 0.2\}$. Whilst the distribution of SHD across the multiple α parameters appear very similar and may be indistinguishable at first glance, note that across all algorithms, the setting of $\alpha = 0.05$ yields the lowest average SHD across all the tested datasets. It is very closely followed by the setting $\alpha = 0.1$. This trend is carried forward to the HD distributions where $\alpha = 0.05,\ 0.1$ are the optimal choices and are visually indistinguishable given the scale of the graph. The general FDR trend for the PC-Algorithm is that it increases with α which agrees with the hypothesis that a more lenient selection parameter introduces a larger number of FPs. However, this trend is not observed across the other algorithms, where α on the extremes of the hyperparameter range do worse than $\alpha = 0.1, 0.05$. The unexpectedly high FDR for $\alpha = 0.001$ is explained by noting that whilst this most restrictive selection criteria minimises the number of FPs, it also rejects a larger number of true edges, and so the ratio of FPs to TPs (which the FDR characterised) increases. Sensitivity increases with α across all constraint algorithms as expected since more lenient algorithms classify a greater ratio of positives, and so we may expect the ratio of TPs to FNs will decrease.

While the SHD and HD trends are carried over to the ECOLI case study in Figure 3.5, there are notable differences in the FDR and sensitivities. While the lenient $\alpha = 0.2,\ 0.1$ hyperparameters have the largest FDR across all algorithms the remaining choices are all clustered closely together.

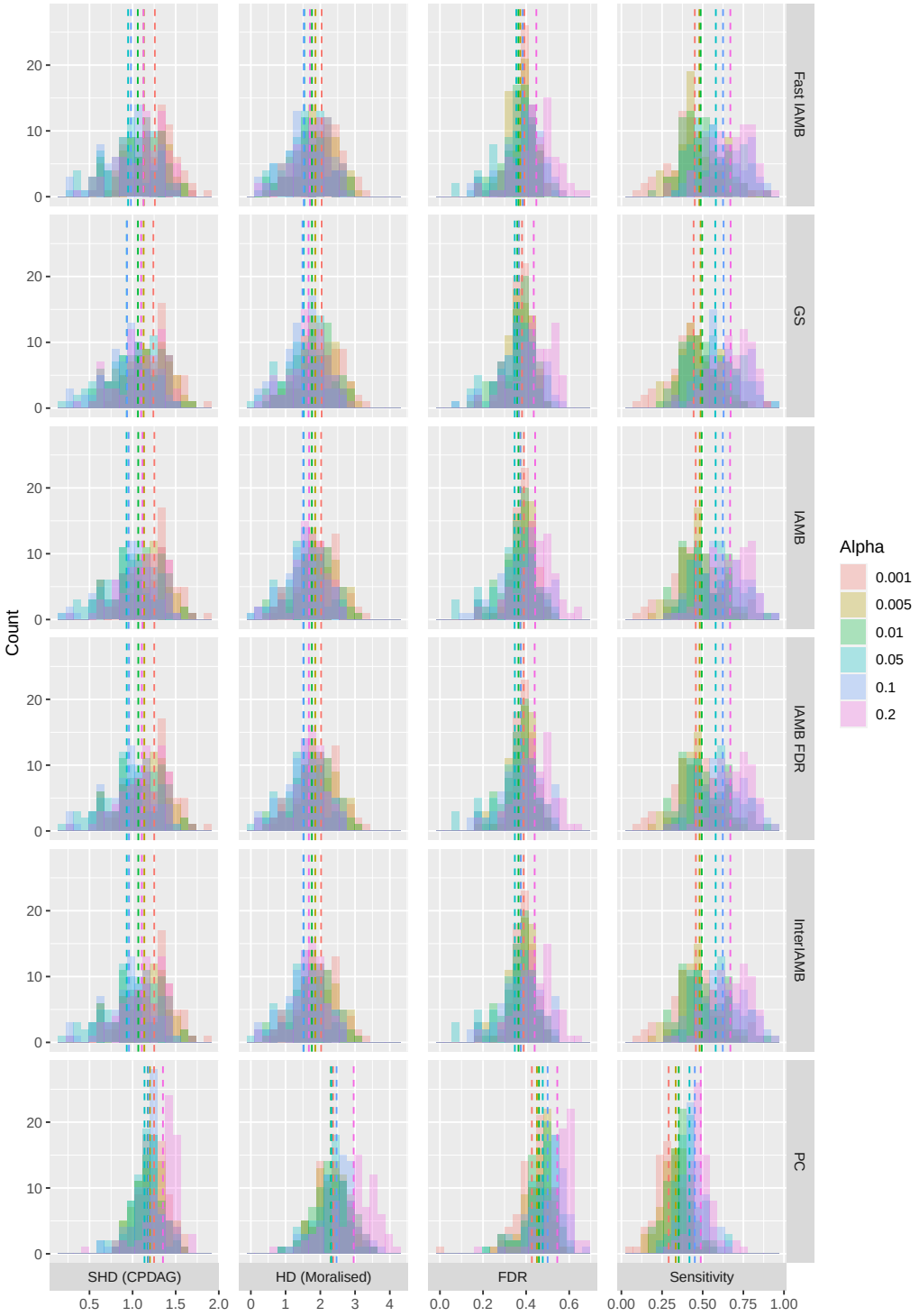

Figure 3.4: **Random DAGs**: SHD, HD, FDR and Sensitivity analysis from hyperparameter optimisation of constraint-based algorithms. The means for the distributions are indicated by vertical dashed lines. The parameterisation of $\alpha = 0.05$ outperforms the other tested values across all algorithms. Apart from the PC algorithm (which performs the worst), all other methods demonstrate very similar levels of performance.

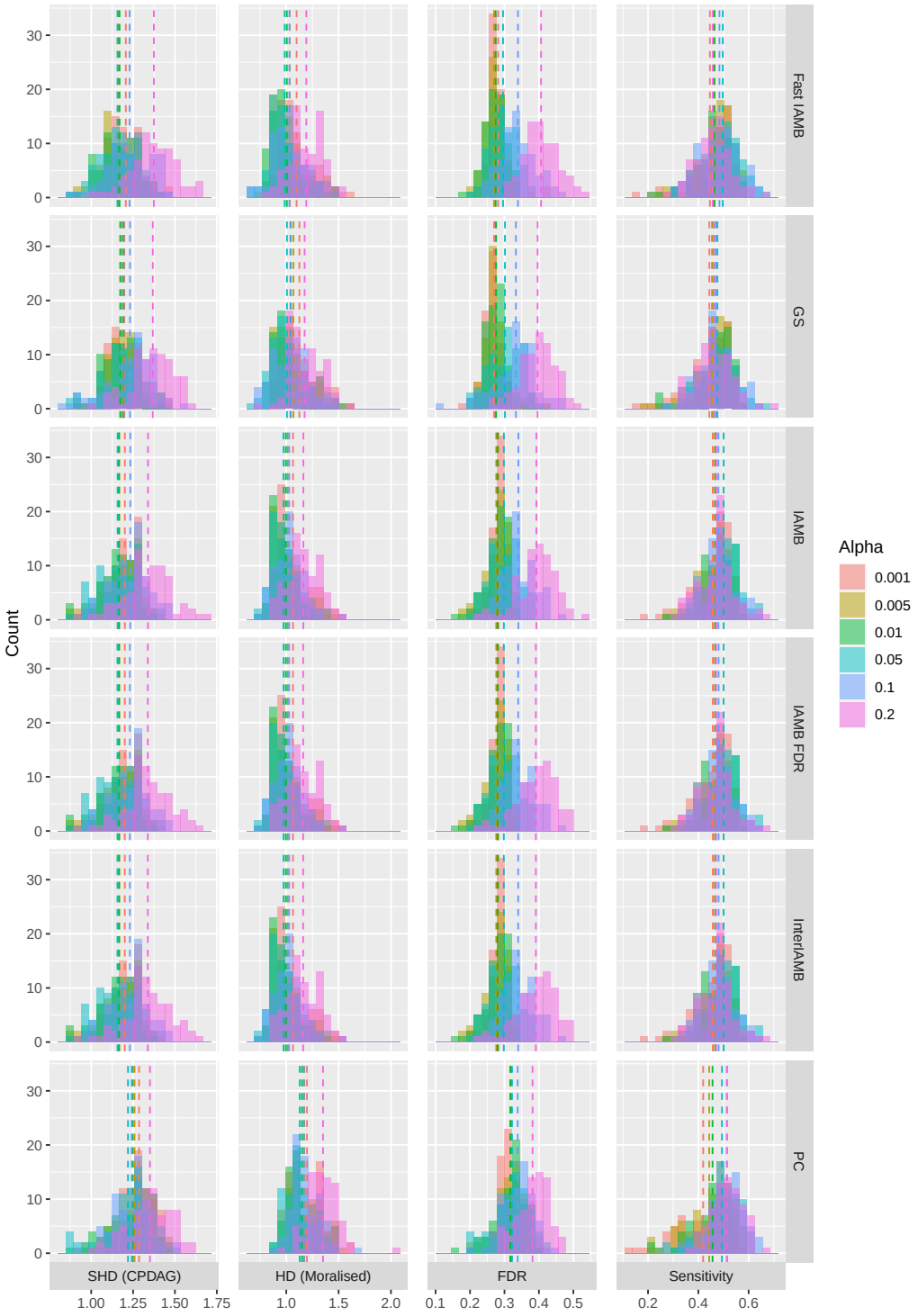

Figure 3.5: **ECOLI**: SHD, HD, FDR and Sensitivity analysis from hyperparameter optimisation of constraint-based algorithms. The means for the distributions are indicated by vertical dashed lines. The means for the distributions are indicated by vertical dashed lines. The parameterisation of $\alpha = 0.05$ outperforms the other tested values across all algorithms. Apart from the PC algorithm (which performs the worst), all other methods demonstrate very similar levels of performance.

$\alpha = 0.05$ is the next highest FDR, although it is much closer to the lowest values compared to its higher α counterparts. The sensitivity plots demonstrate that, apart from the PC-algorithm, the choice of $\alpha = 0.05$ identify the greatest number of TPs. It is worth attempting to diagnose the discrepancies between these results and the Random DAGs. Figure 3.1 demonstrates that the distributions of node degree's differ across artificial and biological networks and so more conservative α setting may be more appropriate for sparser networks. Secondly, the weights and standard deviations of the marginal and conditional Gaussian distributions also differ substantially. The weights and standard deviations of the Random DAG synthetic networks are all 0.5 and 1 respectively, whereas the corresponding values in the biological networks are more varied. It may be more challenging to distinguish immediately causal nodes via conditional hypothesis testing if the inclusion of a variable in a conditioning set does not clearly perturb the target distribution. These observations and the results in Figures 3.4 and 3.5 highlight that the algorithm performances are impacted significantly by both the graph shape and the parameterisation of the network. Great care must be taken when drawing conclusions about algorithm performance across different datasets, and validation should be conducted on artificial networks which replicated the target data as closely as possible.

3.2.2 Hill Climbing

Hill Climbing algorithms are parameterised by the number of edge perturbations, the number of local-search restarts, the graph initialisation of the algorithm, and the choice of score function. The restart and perturbation hyperparameters are investigated over the range $\{1, 10, 25, 50\}$. An empty and random graph initialisation will be investigated, as will the choice of BGe and BIC score functions.

Figures 3.14 and 3.15 illustrate that the SHD and HD decrease as the number of restarts and perturbations increase. This trend is echoed across the FDR and sensitivity metrics, which are lowered and increased respectively when a more comprehensive local search (higher restarts/perturbations) search is performed. It appears that the algorithm performs best when these parameters are as large as computationally possible, and so consideration of computational time should be accounted for when choosing these hyperparameters. Figure 3.11 demonstrates that hill-climbing algorithms are negatively affected when initialised with a random network as opposed to an empty graph. These figures also demonstrate that using the BGe score function returns networks with a much lower SHD and HD than using a Gaussian BIC function. The greatly increased FDR for BIC learned networks show that it does not naturally select sparse networks, suggesting that the BIC approximation of large datasizes has not been satisfied for examples studied here. The above trends are also manifested in the biological ECOLI networks.

3.2.3 Tabu Search

Tabu search algorithms are parameterised by the number Tabu list length, the graph initialisation of the algorithm, and the choice of score function. Commentary on local greedy search algorithms indicate that the optimal choice for a Tabu list is proportional to the number of nodes in the network and so the hyperparameter chosen to validate was the Tabu Fraction $\in \{0.25, 0.5, 1, 2, 5\}$, defined such that the Tabu Fraction multiplied by the number of network nodes is the Tabu list

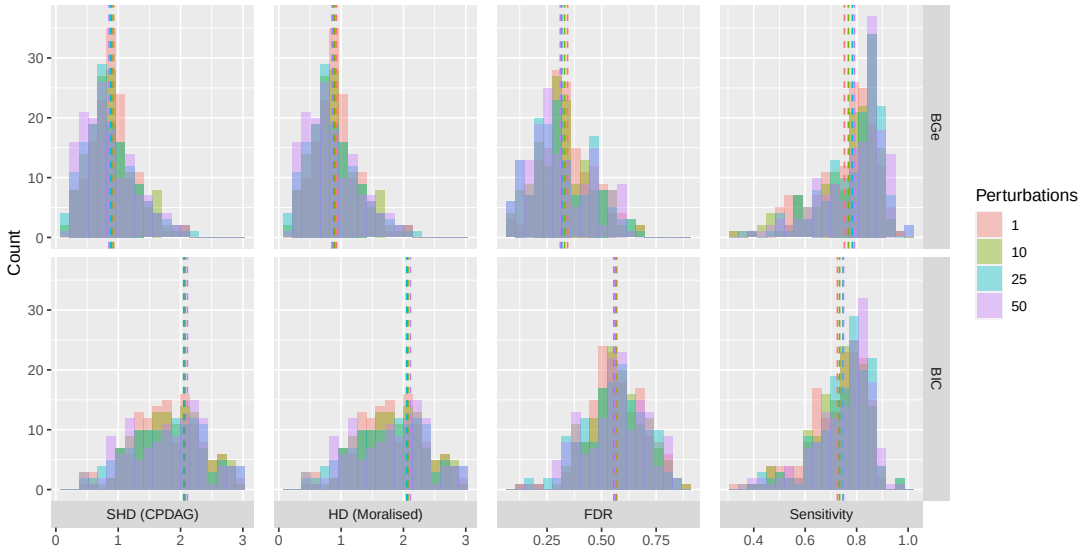

Figure 3.6: **Random DAG**: SHD, HD, FDR and Sensitivity analysis from perturbation number hyperparameter optimisation of HC based algorithms. The means for the distributions are indicated by vertical dashed lines. Note the trend that algorithm performance increases with the number of restarts.

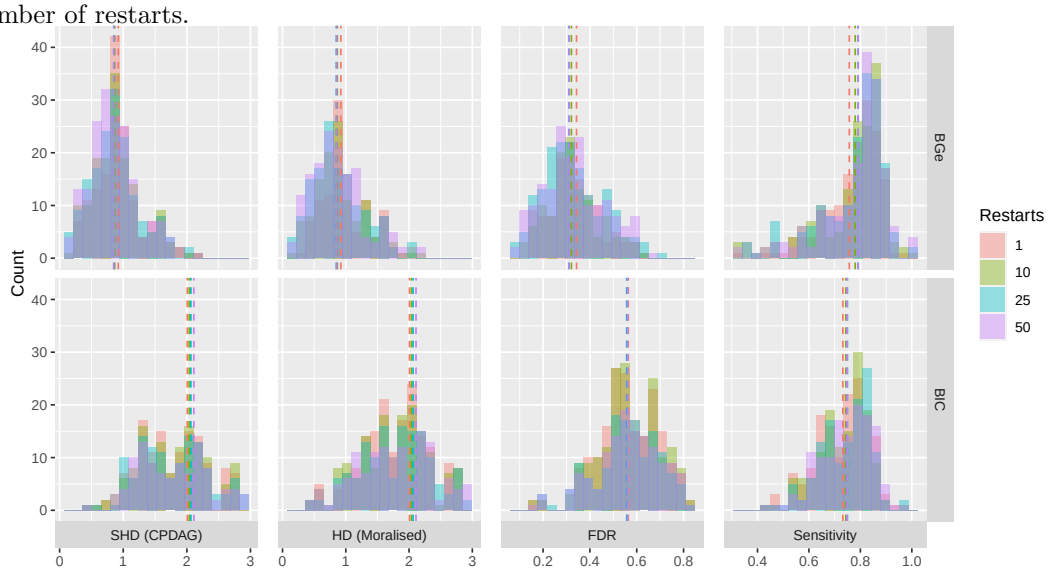

Figure 3.7: **Random DAG**: SHD, HD, FDR and Sensitivity analysis from restart number hyperparameter optimisation of HC based algorithms. The means for the distributions are indicated by vertical dashed lines. Note the trend that algorithm performance increases with the number of restarts.

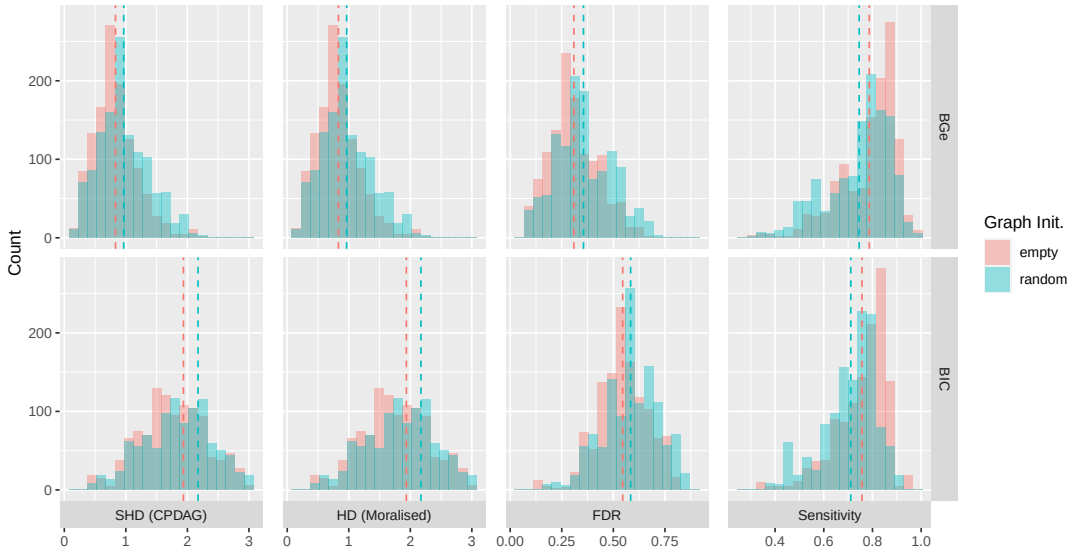

Figure 3.8: **Random DAG**: SHD, HD, FDR and Sensitivity analysis across empty and random initialisation states for the HC algorithm. Observe that empty graph initialisations outperform their random counterparts.

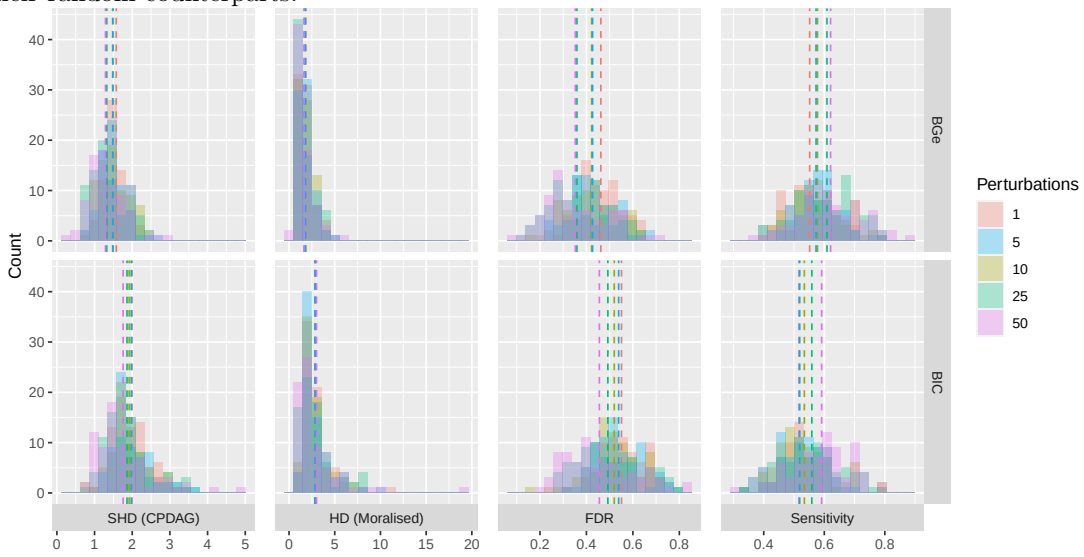

Figure 3.9: **ECOLI**: SHD, HD, FDR and Sensitivity analysis from perturbation hyperparameter optimisation of HC based algorithms. The means for the distributions are indicated by vertical dashed lines. Note the trend that algorithm performance increases with the number of perturbations.

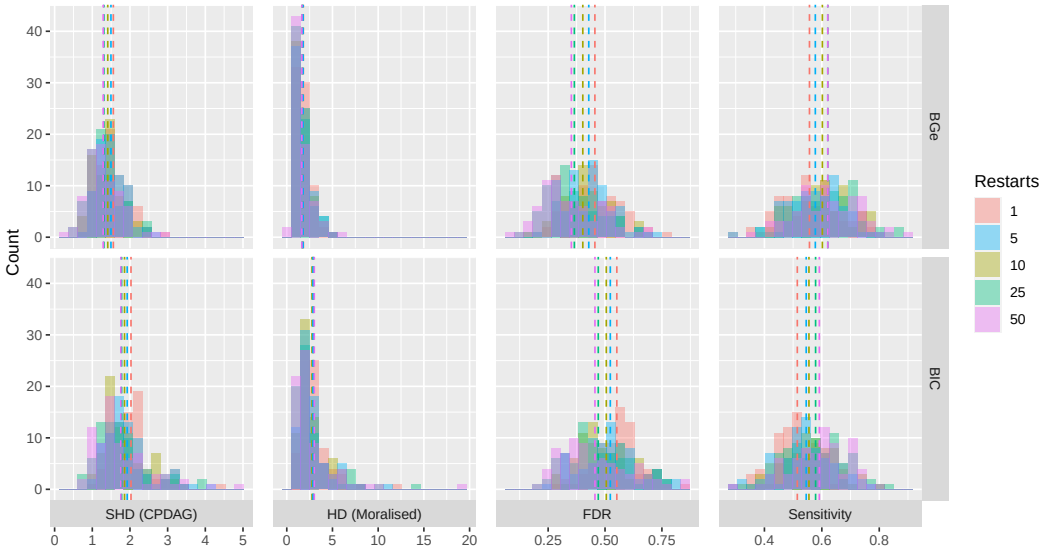

Figure 3.10: **ECOLI**: SHD, HD, FDR and Sensitivity analysis from hyperparameter optimisation of HC based algorithms. The means for the distributions are indicated by vertical dashed lines. The means for the distributions are indicated by vertical dashed lines. Note the trend that algorithm performance increases with the number of restarts.

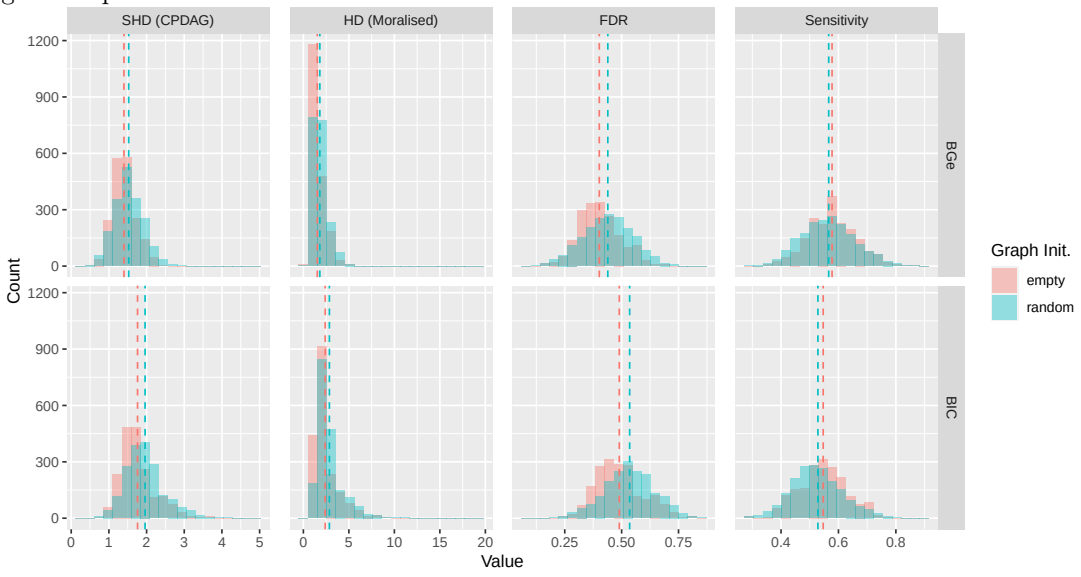

Figure 3.11: **ECOLI**: SHD, HD, FDR and Sensitivity analysis across empty and random initialisation states for the HC algorithm. The means for the distributions are indicated by vertical dashed lines. The figures show that an empty initialisation outperforms a random choice.

length. The graph initialisation and score functions are the same as those validated in section 3.2.2

Similarly to Hill Climbing, initialising the algorithm with an empty network yields slightly better results than initialising a random network. However, the difference between these two is markedly reduced compared to HC for both Random DAGs and the ECOLI dataset. The algorithm with the best performance (lowest SHD, HD and FDR, and highest sensitivity) is a choice of Tabu Fraction = 1, with larger and smaller fractions yielding worse results. This Goldilocks approach of having a hyperparameter that is "just right" is similar to the choice of α in constraint-based algorithms, whereas the performance of HC monotonically increases with increasing restarts and perturbations. The variation in metrics between Tabu Fractions is greater in the ECOLI network compared to the Random DAGs, which may again result from the differences in their graphical structure. It is well documented that the optimal choice of Tabu list depends on the problem at hand. Since a larger Tabu length means increases the number of iterations before the search returns to a node, larger lists are effective for breaking away from local minima whereas smaller list sizes excel at optimising the learned network structure once the algorithm has identified a global minima. Sparser networks have been shown to benefit from longer list lengths, and so this is also a parameter which requires fine tuning to the problem at hand. We also note that, just like with HC, the BGe score function returns networks with a much lower SHD and HD than using a Gaussian BIC function.

3.2.4 Hybrid Methods

Since in the above experiments Hill-Climbing outperformed Tabu search, the former was used for the maximisation phase of the hybrid algorithms assessed here with the BGe score function. The well established MMHC and H2PC algorithms were investigated, as well as a custom method which used InterIAMB for the restrict phase.

All hybrid algorithms significantly outperformed isolated constraint and score-based methods. Hybridised approaches appear to synergise the low FDR of constraint algorithms with the high sensitivity of score-based methods. The optimal hyperparameter choice of $\alpha = 0.2$ suggests that the restriction phase is there to greatly reduce the space of possible networks explorable by a score algorithm but does not need to return a highly accurate undirected network. Keeping in as many positive edges as possible whilst removing a significant proportion of negatives allows HC to be much more effective at optimising the minima of the restricted network space.

Whilst all methods performed equally well on the Random DAGs, H2PC notably excelled when applied to the ECOLI data. Another interesting observation is that while hybrid methods significantly outperformed constraint algorithms by reducing the normalised SHD by approximately 0.25, the reduction in the ECOLI scenario was smaller than 0.05, further illustrating that the performance of structure learning algorithms is significantly affected by the true network's dynamics.

3.2.5 Regression Methods

NOTEARS algorithm performance proved highly sensitive to the L1 regularisation parameter λ_1, which was validated over $\lambda_1 \in \{0.001, 0.01, 0.1, 1\}$. In both the Random DAG and ECOLI setting,

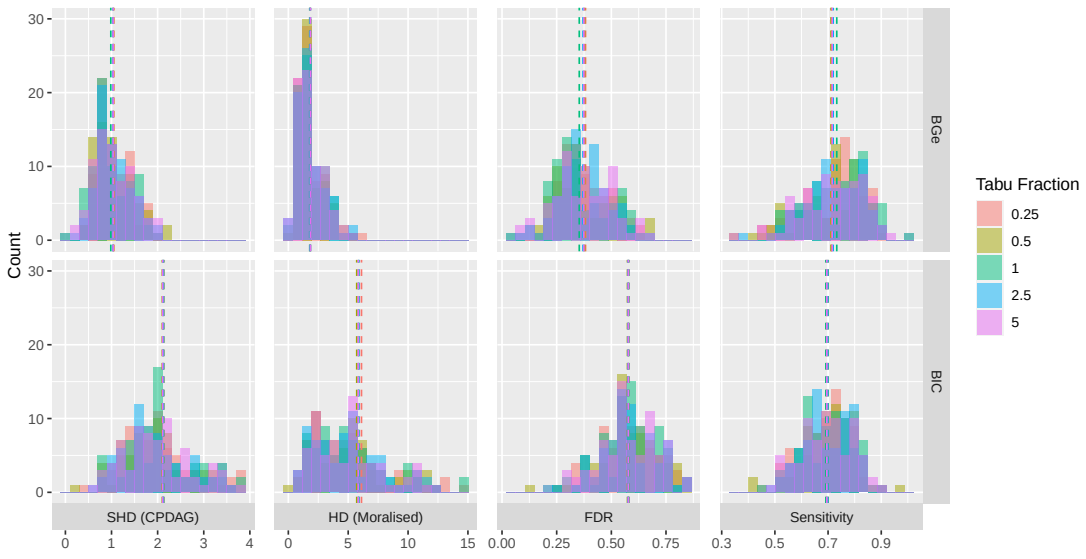

Figure 3.12: **Random DAG**: SHD, HD, FDR and Sensitivity analysis from hyperparameter optimisation of TABU based algorithms. The means for the distributions are indicated by vertical dashed lines. The hyperparameter choice of Tabu Fraction= 1 yields the best performance, although it is worth noting that the difference in performance across validated parameters is small.

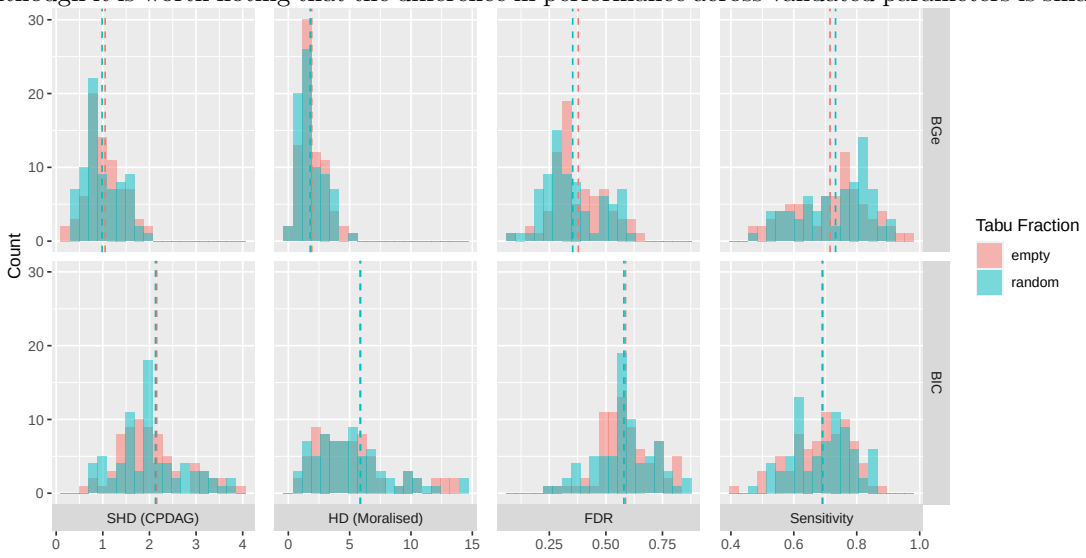

Figure 3.13: **Random DAG**: SHD, HD, FDR and Sensitivity analysis across empty and random initialisation states for the TABU search. The means for the distributions are indicated by vertical dashed lines. There is only a very slight discrepancy between empty and random initialisation.

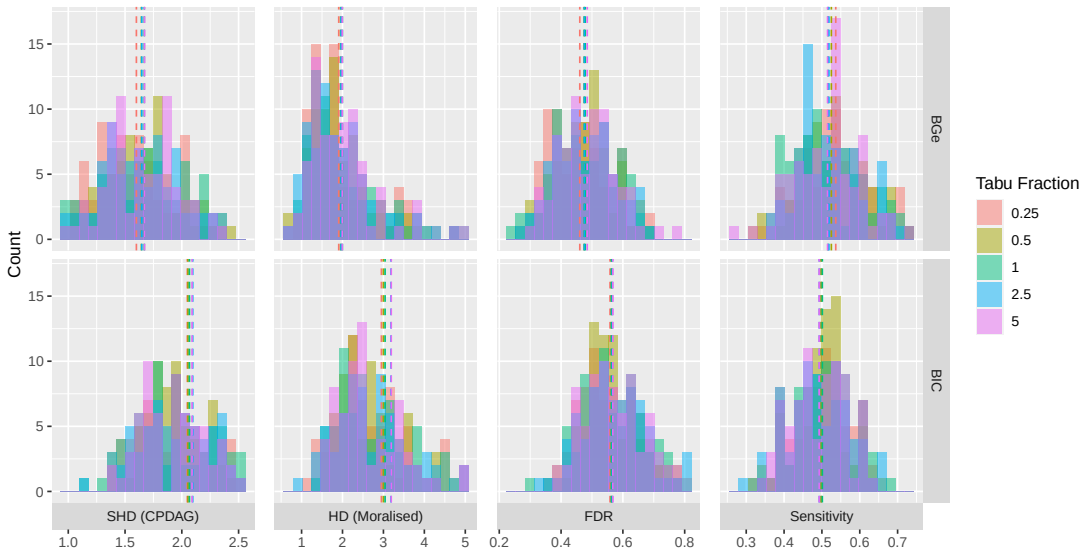

Figure 3.14: **ECOLI**: SHD, HD, FDR and Sensitivity analysis from hyperparameter optimisation of TABU based algorithms. The means for the distributions are indicated by vertical dashed lines. Similarly to the random DAG case, there is very little variation between hyperparameter choice, although the choice of Tabu Fraction = 0.25 appears to perform slightly better than the rest of the tested parameters.

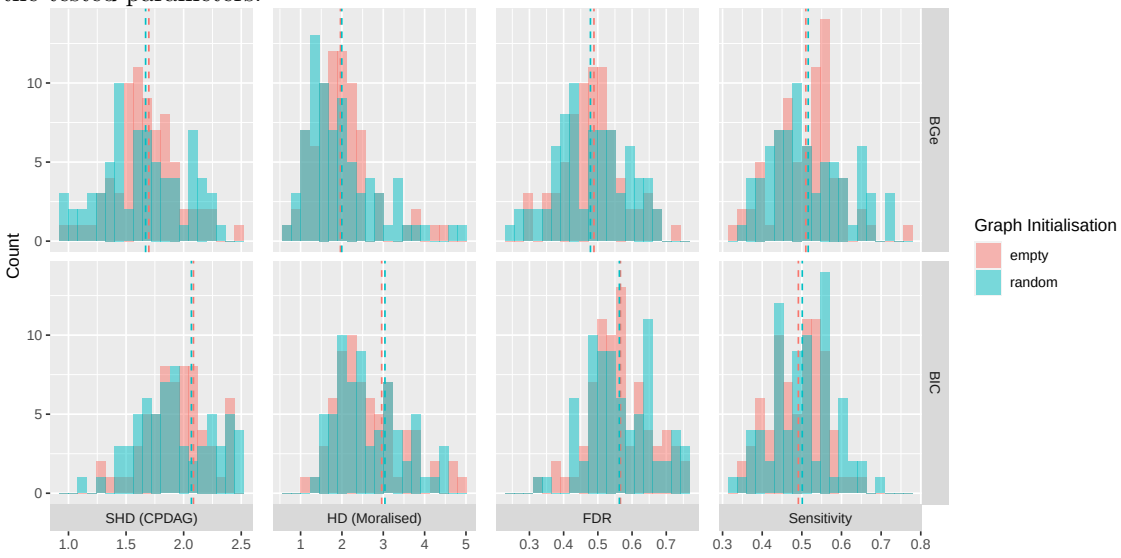

Figure 3.15: **ECOLI**: SHD, HD, FDR and Sensitivity analysis across empty and random initialisation states for the TABU algorithm. The means for the distributions are indicated by vertical dashed lines. Random initialisations outperform an empty choice, although this difference is extremely slight.

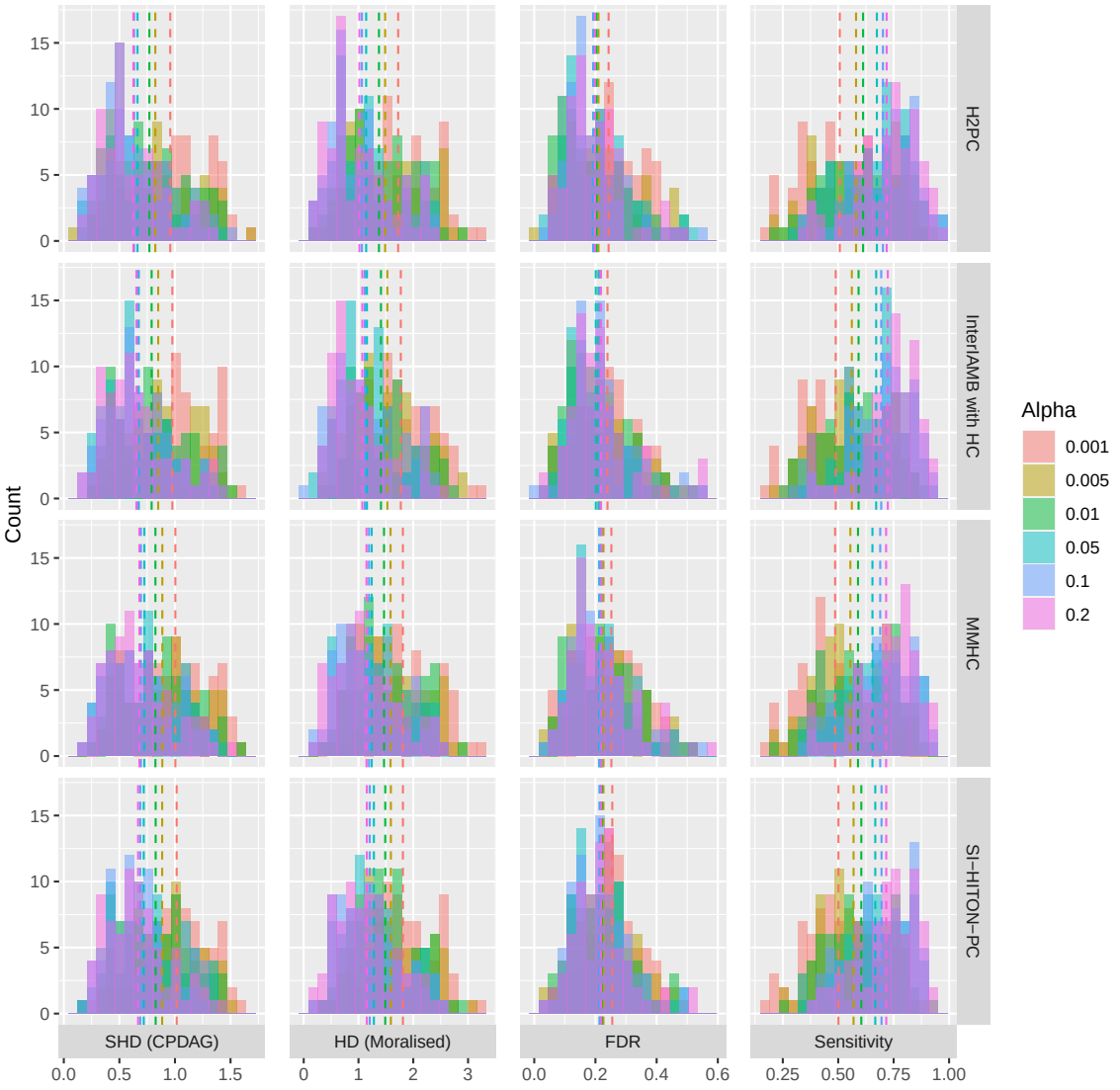

Figure 3.16: **Random DAG**: SHD, HD, FDR and Sensitivity analysis from hyperparameter optimisation of hybrid algorithms. The means for the distributions are indicated by vertical dashed lines. The hyperparameterisation of $\alpha = 0.2$ (the largest tested value) yields the highest performance on SHD, HD and Sensitivity across all hybrid algorithms. The FDR does not demonstrate much variation in across the range of α relative to the other measurements. The H2PC algorithm demonstrates the best performance across all hybrid algorithms.

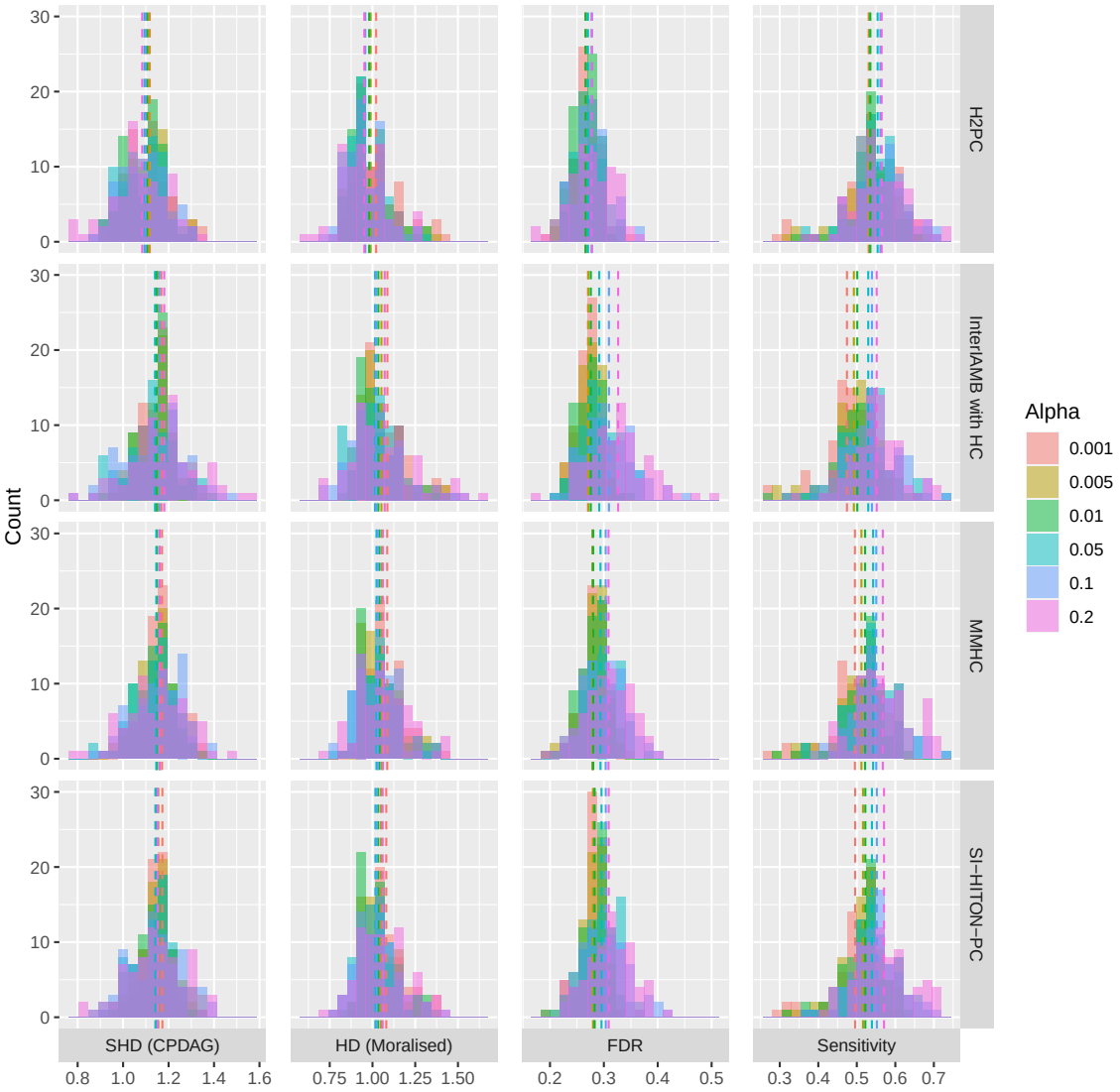

Figure 3.17: **ECOLI**: SHD, HD, FDR and Sensitivity analysis from hyperparameter optimisation of hybrid algorithms. The means for the distributions are indicated by vertical dashed lines. Note that the variation in these metrics is much lower than compared to the random DAG case. The exception is the sensitivity measurements which increase with respect to the choice of α. As in the random DAG experiment, the H2PC algorithm outperformed all its hybrid peers.

the optimal hyperparameter choice was clearly $\lambda_1 = 0.1$, which minimised the SHD, HD, FDR and maximised the sensitivity. NOTEARS yields the lowest SHD of all tested algorithms for the Random DAGs. However, it is outperformed by all hybrid algorithms in the ECOLI experiment.

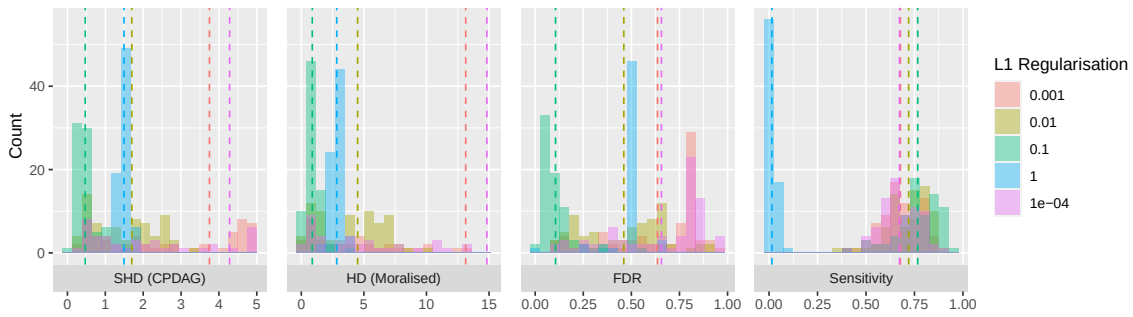

Figure 3.18: **Random DAG**: SHD, HD, FDR and Sensitivity analysis from hyperparameter optimisation of NOTEARS algorithms. The means for the distributions are indicated by vertical dashed lines. The choice of L1 regularisation parameter $\lambda_1 = 0.1$ clearly outperforms all other tested alternatives.

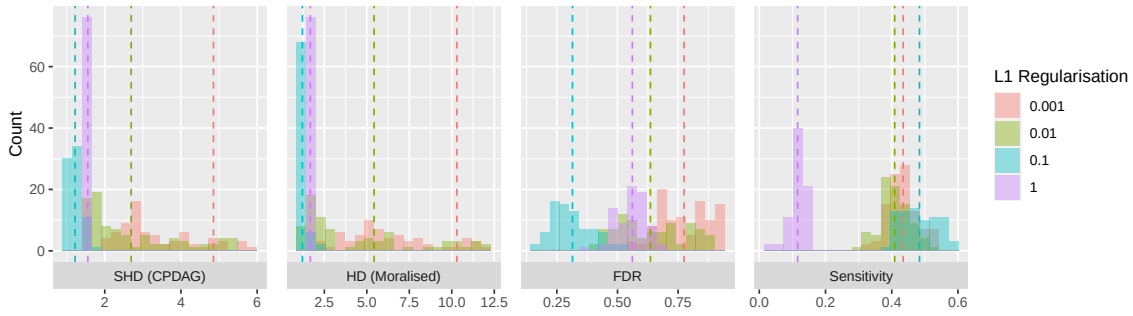

Figure 3.19: **ECOLI**: SHD, HD, FDR and Sensitivity analysis from hyperparameter optimisation of NOTEARS algorithms. The means for the distributions are indicated by vertical dashed lines. As in the Random DAG experiment, the choice of L1 regularisation parameter $\lambda_1 = 0.1$ clearly outperforms all other tested alternatives.

3.3 Effect of Data Size on Algorithm Performance

The number of data samples clearly impacts the algorithm performance since larger sample sizes provide more information for learning the target network. To assess the sensitivity of algorithms to structure datasize, 75 simulated datasets from the ECOLI network were generated for each of the following datasizes: $N \in \{50, 100, 250, 500, 1000\}$. The datasets were input to a subset the best performing algorithms (with optimal hyperparameter settings) and their performances were measured. A hybridised version of the NOTEARS algorithm, NOTEARS-H was also assessed, which uses the H2PC algorithm ($\alpha = 0.2$) and NOTEARS ($\lambda_1 = 0.1$) for the restrict and maximise phases, respectively. The results are shown in Figure 3.20.

For small datasizes, the mean SHDs for all algorithms are very similar. However, as the datasizes increase, the discrepancy between the mean SHDs grows larger. NOTEARS-H noticeably outperforms its competitors for up to $N = 250$, narrowly followed by H2PC. The family of NOTEARS algorithms fail to scale to much larger Gaussian datasets, where H2PC clearly distinguishes itself as the optimal algorithm to use. Perhaps one reason for this is due to HC's drastic rate of improvement across datasizes. Note that the mean SHD for HC does not even register on the plots for $N = 50, 100$ with mean SHDs of 2.81 and 1.91 for those datasizes respectively (the values were cut to aid the clarity of the plot) and for $N = 1000$, it is amongst the most competitive algorithms to use. This suggests that HC is an extremely effective heuristic search algorithm at large N and that this rate of improvement carries forward to hybridised algorithms also. It is encouraging to see that hybridisation of the NOTEARS algorithm with HPC yielded a noticeable improvement in performance across all datasizes, providing further evidence to the idea that hybridising constraint-based algorithms with score or regression methods yield significant performance improvements than their separate counterparts.

For small to medium datasizes of which can reasonably approximated as being Gaussian, the findings suggest that NOTEARS-H is the prime algorithm of choice. However, they are both significantly outclassed by HC methods at higher datasizes. It is worth repeating that the authors of NOTEARS demonstrated that it is consistent in its performance across a variety of different additive noises (Exponential and Gumbel distributions), and so NOTEARS may prove more useful at learning networks from large linear data with non-Gaussian noise [38].

However, it appears that NOTEARS suffers from this generalisability when working on known Gaussian data. By minimising a simple square loss function $||\mathbf{X} - \mathbf{WX}||^2$, the standard deviations of the CPDs are not considered. On the other hand, the score-based methods fit both the weights and variances for the Gaussian likelihood terms and so data which deviate significantly from the mean will be more heavily penalised if their corresponding distribution has a small variance as opposed to if they come from a wider distribution. NOTEARS does not consider this and would penalise the same residual equally across all possible nodes. A possible area for future work may be introduce a variance factor into the NOTEARS square loss and see if this improves algorithm performance.

As a further experiment, we quantitatively compare the outputs of these algorithms in the extreme case where data is plentiful. Fifteen different datasets of $15,000$ data points were generated from the ECOLI dataset. For each of the 15 datasets, each algorithm learned a network, and the SHDs of an algorithm's output to all other learned graphs were calculated. At this limit, it

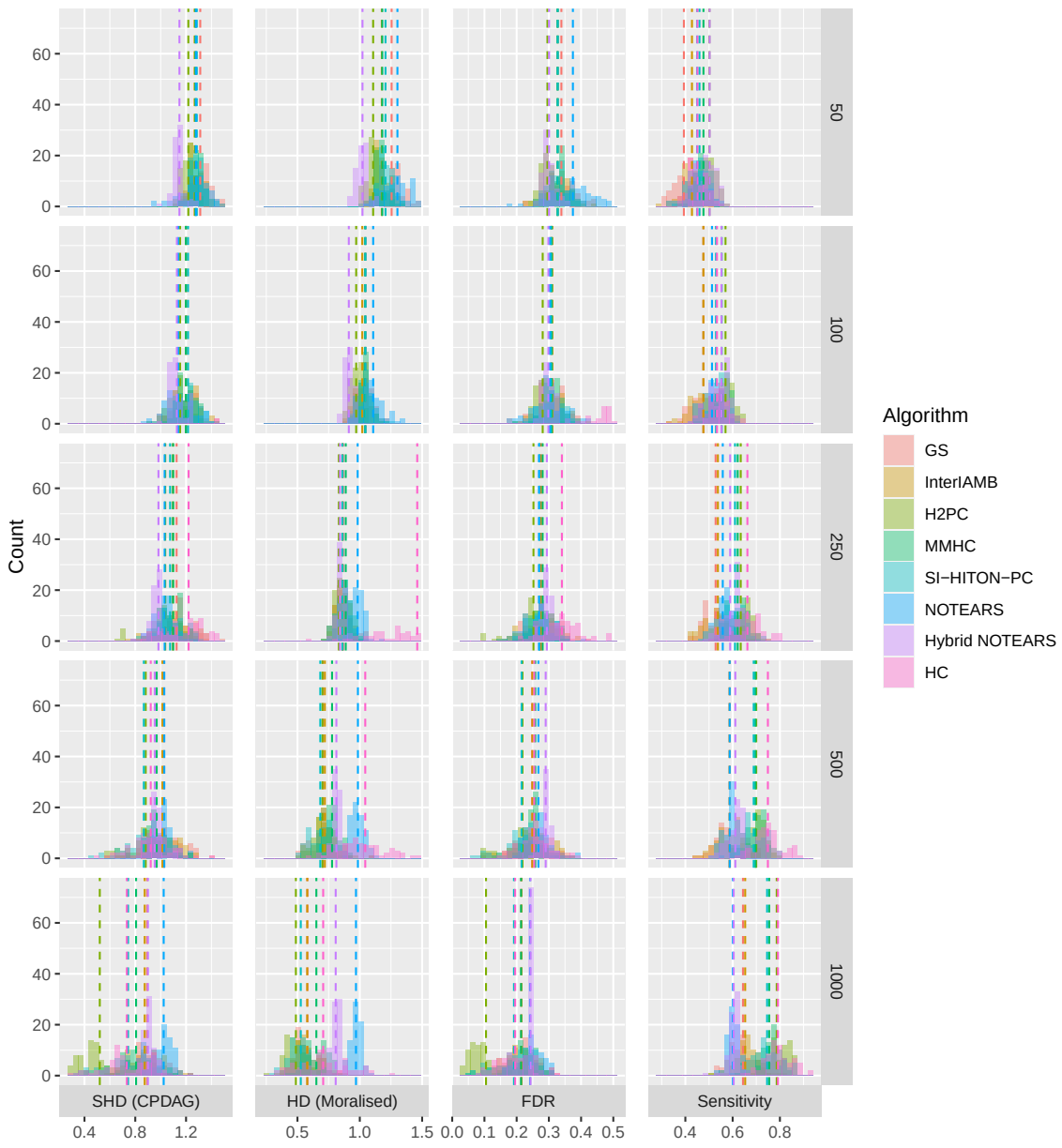

Figure 3.20: Impact of data size on ECOLI network. Each row corresponds to datasizes $N \in \{50, 100, 250, 500, 1000\}$. The means for the distributions are indicated by vertical dashed lines. The general trend is that algorithm performance tends to increase with datasize. Hybrid NOTEARS (NOTEARS-H) demonstrates itself to be most effective when learning from smaller datasets ($N = 50 - 250$), whereas H2PC performs best when learning larger datasets.

should be expected that the differences between the learned networks decreases, since all methods theoretically derive the true graph in the limit of full data. All of the above algorithms in fact have theoretically proven convergence criteria which show that in the limit of infinite complete data, the algorithm almost certainly reconstructs the true network (for proofs, see the flagship papers for each of the respective algorithms). The confusion matrix in fig. 3.21 plots the mean

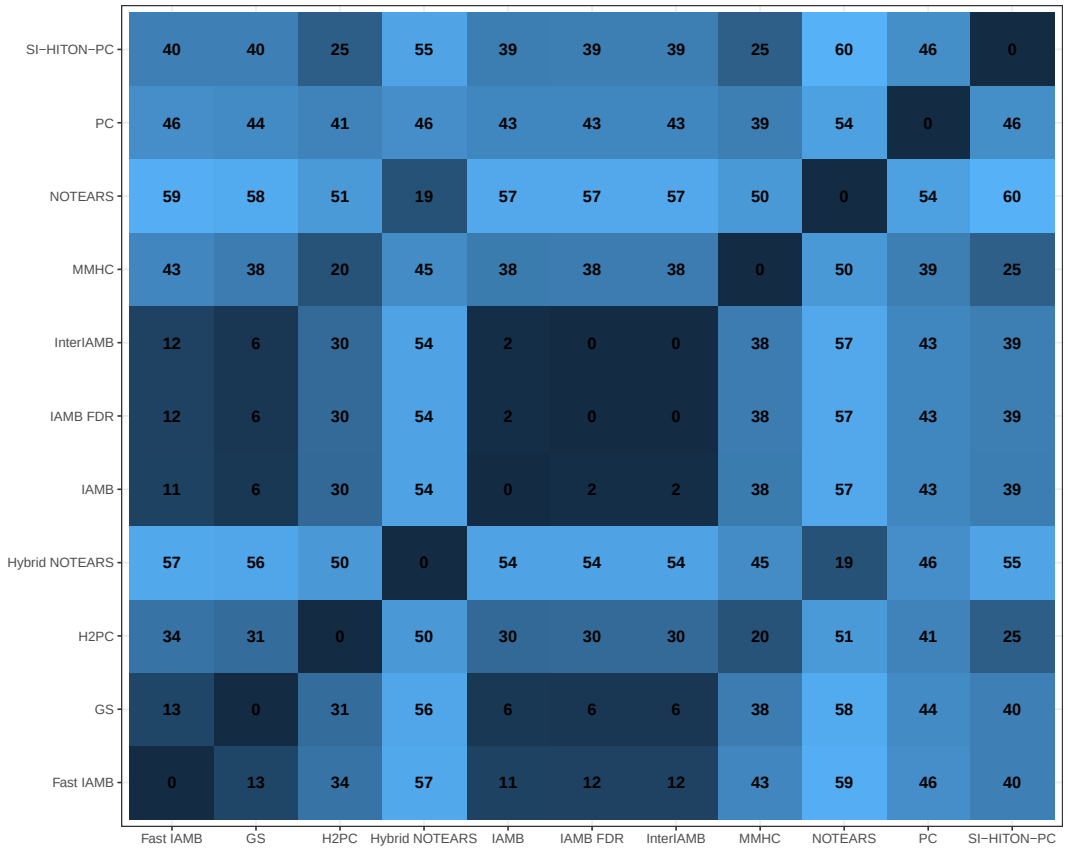

Figure 3.21: Confusion matrix of mean SHDs calculated from networks learned in the approximate limit of very large data sizes. We see that the algorithms that seek to learn the network's Markov Blanket (all IAMB algorithms, along with GS) learn very similar networks. The others differ quite significantly, with the NOTEARS algorithms and PC algorithms differing substantially compared to alternative algorithms.

SHDs for the learned networks and clearly demonstrates that, at least in the ECOLI experiment, these limits are not reached even for datasets where $N \gg D^2$.

Whilst significant differences exist between the algorithms in general, it is worth identifying families of algorithms whose graphs differ by only minor amounts. All the constraint algorithms which involve calculation of the Markov Blanket (IAMB and GS) have mean SHDs close to zero. Networks learned by hybrid algorithms which utilise HC maximisation are most similar to each other, as are networks using the NOTEARS algorithm. Each family of algorithm identifies distinct graphical structures which differ from alternative methods. A more in depth analysis of what exact features these are is limited due to time constraints but is an avenue of further research well worth exploring in the future.

3.4 Effect of Graph Size on Algorithm Performance

The impact of graph size on algorithm performance was assessed across the graph sizes $D \in \{25, 50, 100, 150\}$. Seventy-five randomly generated graphs generated datasets of size $N = 150$

were for all graph sizes apart from $D = 150$, of which only 25 graphs were used due to time restrictions. The algorithms were hyperparameterised using the optimal parameters identified in Section 3.2.

The results of this experiment are shown in Figure 3.22. The general observed trend is that the normalised SHDs and HDs tend to increase with network size. The exceptions here are the InterIAMB, GS, and MMHC algorithms which actually increase across the transitions $D = 25 \rightarrow 50$ and $D = 100 \rightarrow 150$. These trends are echoed in the FDR and Sensitivity plots, which also tend to improve as the graph size increases. Note that the NOTEARS family outperforms its peers across all dataset sizes. In particular, the original NOTEARS algorithms is most performant for smaller graph sizes, whereas the hybridised NOTEARS-H outperforms when learning medium to large graph sizes. This demonstrates the potential effectiveness of algorithm hybridisation, and we propose that NOTEARS-H should be considered as an effective algorithm when learning networks of size $D > 50$.

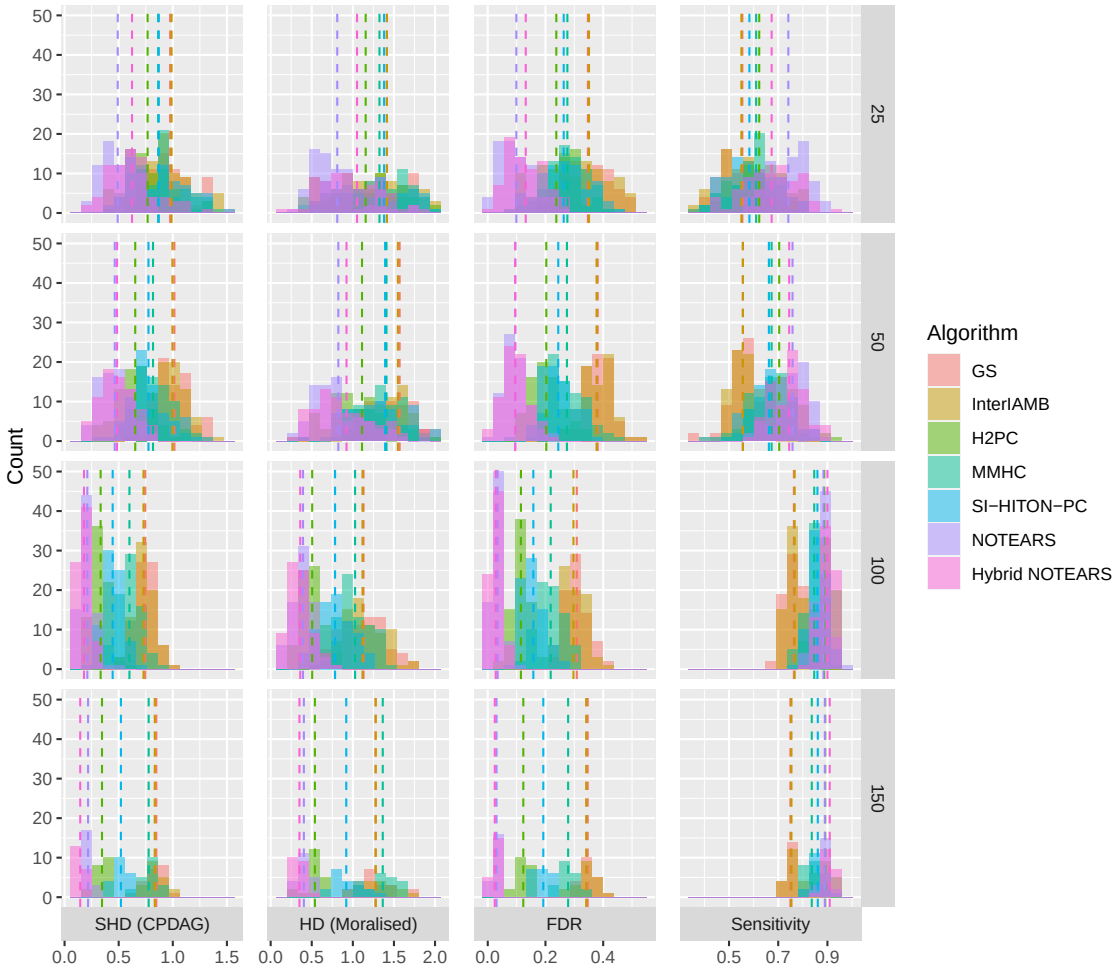

Figure 3.22: Impact of graph size on Gaussian data. Each row corresponds to data drawn from networks of sizes $D \in \{25, 50, 100, 150\}$. The normalised SHDs and HDs tend to decrease as D increases. InterIAMB, GS, and MMHC are the exceptions here, with an increase being observed moving from $D = 25 \rightarrow 50$ and $D = 100 \rightarrow 150$. For $D = 25, 50$, the original NOTEARS algorithm yields the lowest SHD and HD. For larger networks however, hybrid NOTEARS (NOTEARS-H) overtakes it in accuracy. These trends are echoed in the FDR and sensitivity measurements, which decrease and increase respectively with increasing datasizes for all algorithms apart from InterIAMB, GS, and MMHC.

Chapter 4

Structure Learning on RAF Pathway Data

The RAF dataset is a standard dataset for benchmarking performance of structure learning algorithms in recent studies [10]. This dataset, which records the protein expression levels for measured from a variety of independent samples, consists of 11 proteins (nodes), 20 edges, and is of size $N = 853$. The true network is presented in Figure 4.1. The network size is small so algorithm validation using this benchmark will not well measure the ability of algorithms to scale with network size. However, the marginal distributions of protein expressions are clearly non-Gaussian and highly skewed. Histograms of the unnormalised marginal distributions for each gene are presented in Figure 4.2. It is also unclear whether the assumption of linearity between a node's mean and its parents' values still exists. This section investigates how significant the impact of violating these assumptions can be. The performance impact of independence test choice on constraint-based algorithms on non-Gaussian data is also investigated.

In real world structure learning applications, the true network is often unknown. This can make validating the accuracy of a learned network a challenge since there is no ground truth to compare it with. In the case where there is no source of expert domain knowledge one can cross validate against, assessing the variability of an algorithm's output to subsets of training data can provide a measure of confidence in one's results. This is known as model averaging and is be outlined further in this section.

4.1 Model Averaging

In the idealised case of infinite data, the network learned from the entire dataset would be equal to the true network for consistent score-based functions, and highly accurate for constraint algorithms (due to the inherent type I/II error rates which occur in hypothesis tests). Indeed, as Equations 2.45 and 2.46 demonstrate, the log posterior $\log P(\mathcal{D}|G)$ grows linearly with the datasize and so the differences between optimal and suboptimal graphs grows exponentially with the data. This makes identifying the maximum of Bayesian scores fairly easy in the limit of infinite data. When dealing with small datasizes however, multiple networks from different equivalence classes

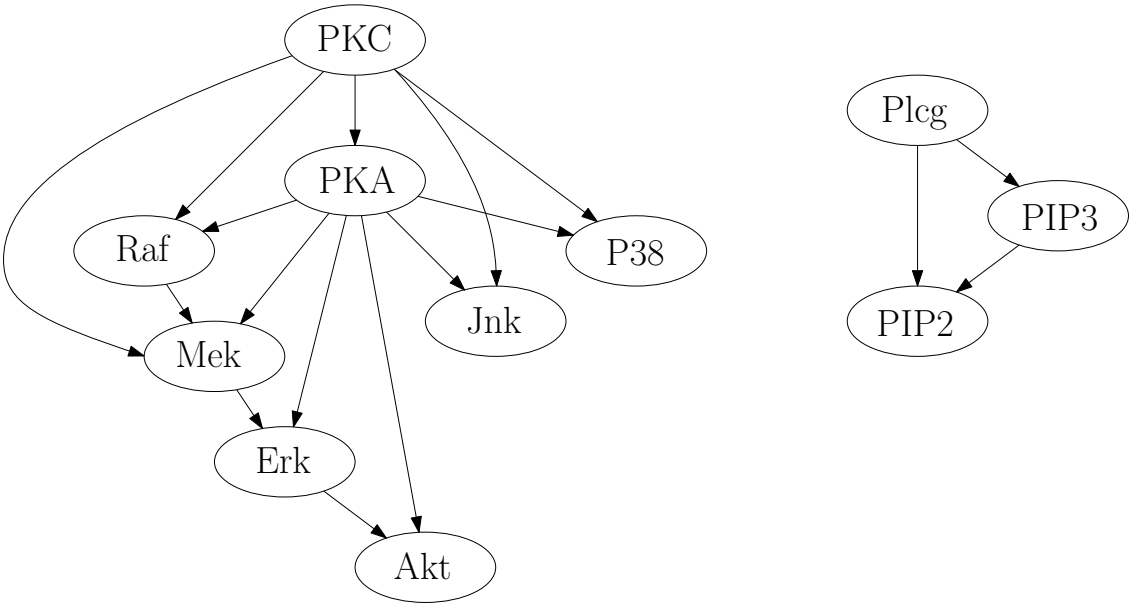

Figure 4.1: True RAF network as validated by [10].

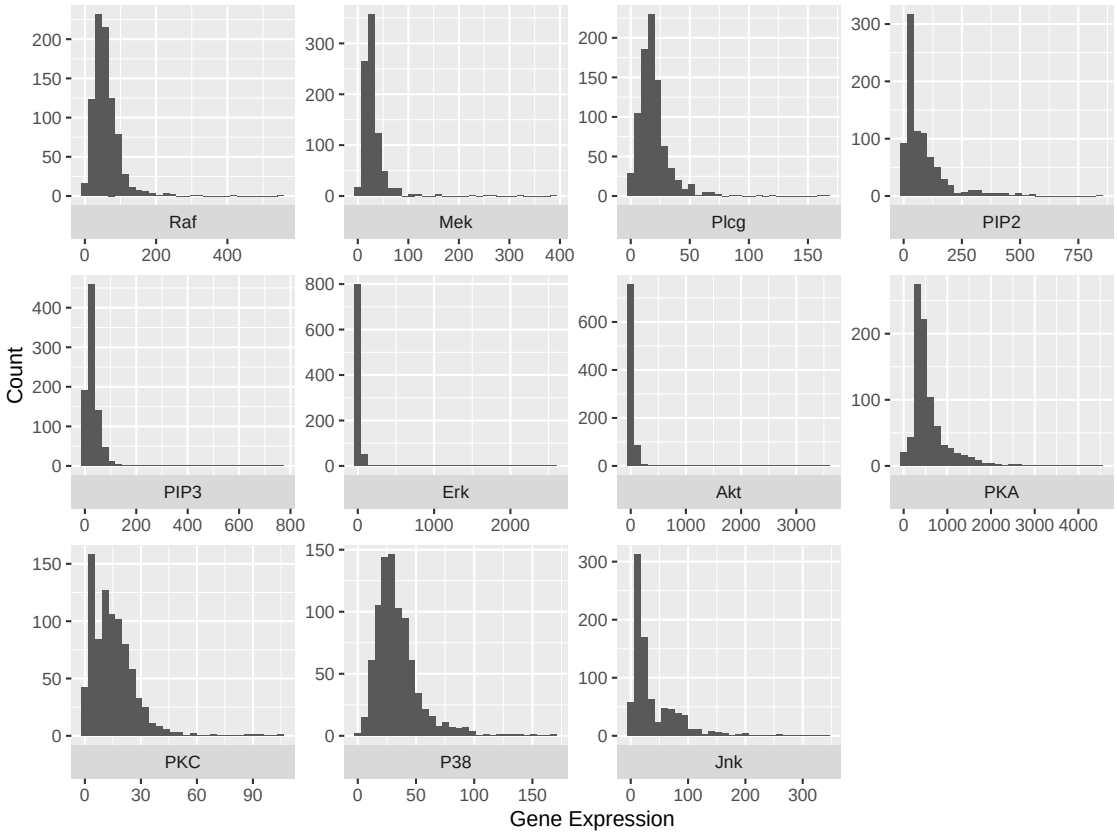

Figure 4.2: Histograms of gene expressions in the RAF Sachs dataset. None of the gene expressions appear to be normal, with all plots demonstrating a significant positive skew.

may have similar scores, suggesting that they are equally close to the true structure. For the problem of structure discovery, this is highly problematic as drawing conclusions on structure from a variety of contradictory networks is not possible. Ideally, a set of confidence estimates in the conclusions of the learned networks would be extremely valuable in the pharmaceutical domain, where experimental validation of network edges is costly. Such estimates can be derived by model averaging.

Let the function $f(G)$ represent a Boolean variable that is one when the edge is present in graph G and zero otherwise. From a Bayesian perspective, the metric that quantifies the confidence in the edge's presence given the data is the posterior:

$$P(f|\mathcal{D}) = \mathbb{E}_{P(G|\mathcal{D})} f(G) = \sum_{G} P(G|\mathcal{D}) f(G) \tag{4.1}$$

which counts the expected number of times an edge is present in graphs generated by the training data. One approach for approximating the above expectation uses the Monte Carlo estimate:

$$P(f|\mathcal{D}) \approx \frac{1}{N} \sum_{i=i}^{N} f(G_i) \tag{4.2}$$

where a set of graphs $\{G\}_N$ are sampled from $P(G|\mathcal{D})$. One approach constructs a Markov Chain Monte Carlo sampler that explores the space of graphs by considering the same single edge modifications as described in Section 2.2.2. The popular Metropolis-Hastings samples could be run with the transition probability:

$$\min \left[1, \frac{P(G^*, \mathcal{D}) T(G^* \to G)}{P(G, \mathcal{D}) T(G \to G^*)} \right] \tag{4.3}$$

where G and G^* are the current and proposed transition states respectively. $T(G \to G^*)$ is the proposal distribution over the set of transitions from G. A simple approach would be to assign a uniform distribution across all possible transition states [12].

Another approach is simply to bootstrap N various subsets of data from our training set ($\mathcal{B}_i \subseteq \mathcal{D}$) and apply a structure learning algorithm to each bootstrapped dataset. The collection of all learned BNs are then used to for the set $\{G\}_N$ used in the Monte Carlo expectation taken above. Given its simplicity of use and demonstrable effectiveness, bootstrapping will be the model averaging technique used in this thesis.

By generalising the above methods to all edges in the space of learnable graphs, we may obtain confidence measures (approximate a posteriori probabilities) for the entire learned network.

4.2 Choice of a Significance Threshold

The resultant feature posterior obtained from model averaging will likely have many features with a very small but non-zero probability of being present. Whilst there is no problem with this from a Bayesian perspective, it is often very complex to analyse such a densely populated network. This problem motivates the search for a "shrinkage" operator akin to lasso regularisation in the domain of linear regression.

Thresholding methods are one possible solution to select or shrink an averaged model. A popular method used in structure learning involves the selection of a threshold bound (between zero and one). Any edges with a posterior greater than the significance threshold are deemed significant and remain in the network whereas any that lie below this bound are rejected. In this thesis, significance thresholds are chosen using a method which seeks to minimise the L1 norm between the empirical CDF of edge posterior values and an idealised, unknown network with boolean edge posteriors 0 or 1. An outline of the proof is be presented in this thesis; further details can be found in the original paper [43]. For a set of empirical edge posterior $\hat{p}_{f_i}$ probabilities for edge features f_i, construct from it a vector of order statistic $\hat{\mathbf{p}}_{(\cdot)}$ where

$$\hat{\mathbf{p}}_{(\cdot)} = (\hat{p}_{(1)}, \hat{p}_{(2)}, \ldots, \hat{p}_{(K)}) \text{ where } \hat{p}_{(1)} \leq \hat{p}_{(2)} \leq \cdots \leq \hat{p}_{(K)}. \tag{4.4}$$

Define corresponding edge posteriors for an unknown target network with adjacency matrix W as

$$\tilde{p}_{(i)} = \begin{cases} 1 & \text{if } f_i \in W \\ 0 & \text{otherwise} \end{cases} \tag{4.5}$$

where $\tilde{p}_{(i)}$ is equal to 1 if the edge is significant, and zero otherwise. The vector of order statistics for the target network is therefore

$$\tilde{\mathbf{p}}_{(\cdot)} = \{0, \ldots, 0, 1, \ldots, 1\} \tag{4.6}$$

where the 0-1 boundary is set by significance threshold t placed on the empirical CDF. Selecting a threshold which minimises the "distance" between the empirical and idealised order statistic vectors is a sensible choice. One such measure of distance is the $L1$ norm:

$$L_1(t; \hat{\mathbf{p}}_{(\cdot)}) = \sum_x \left| F_{\hat{\mathbf{p}}_{(\cdot)}}(x) - F_{\tilde{\mathbf{p}}_{(\cdot)}}(x; t) \right|. \tag{4.7}$$

The value of t that minimises the above loss is then chosen as the significance threshold.

4.3 Discretisation of Continuous Data

If linear and Gaussian assumptions fail to characterise the distributions observed in the data, continuous structure learning methods can perform extremely poorly. One approach which has had some success in rectifying this issue involves discretising the data and treating the system as a multinomial probabilistic model. Hartemink discretisation seeks to bucket the data in such a way that minimises the loss of information in the data [9]. The pseudocode for this method is outlined in Algorithm 10.

To assess the improvement of performance after bucketing continuous data, the RAF dataset is discretised using the settings $m = 100$ and $n = 10$. The network is then learned using constraint-based (with discrete mutual information tests), score-based (with BDe score), and hybridised algorithms.

Algorithm 10: Hartemink Information Preserving Discretisation

Discretise a continuous dataset $\mathcal{D}_c$ into n different quartiles

repeat

  forall *nodes* $X_i \in G$ **do**

    forall *pairs* p *of adjacent intervals of* X_i **do**

      Collapse pair p *into a single interval for node* X_i *and calculate the cumulative mutual information between the collapsed node* X_i^p) *and all other nodes:*

$$M_{Cum}(X_i^p) = \sum_{j \neq i} MI(X_i^p, X_j) \tag{4.8}$$

      Assign $X_i \leftarrow argmax_p M_{Cum}(X_i^p)$

    end

  end

until *each variable has* $m \ll n$ *buckets*;

return *Discretised dataset* $\mathcal{D}_d$

4.3.1 Results

Figures 4.3, 4.4, 4.5, 4.6, and 4.7 illustrate the normalised SHD, normalised HD, FDR and sensitivity performance for InterIAMB, H2PC, TABU, HC and NOTEARS algorithms on continuous data. For any algorithms that utilise constraint-based methods, the standard t-test, Fisher Z-test and Gaussian mutual information (MI) test were used. Their respective non-parametric Monte Carlo alternatives were also investigated across multiple permutation numbers, $B \in \{100, 500\}$.

Figures 4.3, 4.4 show that the non-parametric MC tests do not demonstrate much improvement in performance when compared to their parametric counterparts. Given that MC hypothesis tests take significantly longer to run, this suggests that one does not necessarily suffer a reduction in performance when using Gaussian parametric tests on non-Gaussian data. While InterIAMB and H2PC demonstrate similar levels of sensitivity, certain parameterisations of H2PC do not detect any false edges which makes it exceptionally useful in applications where false positives are more costly than false negatives. Also note that while the Z-test yields the worst performance, there is not much variation between the t and MI tests. The exact t-test yielded the best performance for both parametric and non-parametric MC tests and so it shall be considered as the optimal test choice for subsequent experiments [44].

Figures 4.6 and 4.5 show that the performance between HC and TABU methods are almost identical, although all settings of HC methods outperform the TABU alternatives. The choice of perturbation number P was chosen to be 5 to reflect the small size of the RAF network.

Figure 4.7 shows that while NOTEARS algorithms detected no false positives, they detected few true positives also (substantially fewer than H2PC). The various hyperparameterisation did not significantly affect performance.

Figure 4.8 show the performance of a subset of algorithms against the RAF data after it was Hartemink discretised with $m = 5, n = 100$. Mutual information (with and without shrinkage) and χ^2 tests were compared for the constraint-based algorithms, but did not show any effect on performance. The score-based algorithms used the BDe score function for learning. All discrete methods had a much higher sensitivity than any of the other continuous methods whilst maintaining relatively low FDRs. The InterIAMB (with $\alpha = 0.05$), HC (with Restarts $= 50$, Perturbations $= 5$)

and TABU (with list length = 15) yielded higher sensitivities than H2PC.

The findings in this section demonstrate that discretisation can yield improvement in structure learning performance. However, their continuous counterparts, whilst demonstrating lower sensitivities, do not perform badly and can return networks with absolutely no false positives - a feat not achieved by any of the discrete methods.

The RAF data provides useful benchmark for validating structure learning methods against non-Gaussian data. However, its extremely small graph size limits our ability to draw inferences about performance to other practical applications. The small graph size also makes the evaluation metrics extremely sensitive to even a single difference in two learned networks since this can correspond to a significant proportional change in TPs and FPs which carries through to the FDR and sensitivity measures. Hence, while the above discussions are of use, one should be cautious before generalising these findings to wider dataset types.

Figure 4.3: Performance of InterIAMB algorithm on RAF data. We note that the FDR rates for the mutual information (MI) and t-test (Corr) are substantially lower than the Monte Carlo (MC) Fisher-Z tests (ZF). The exact Fisher-Z and Monte Carlo t-tests demonstrate the highest sensitivity metrics, with the Monte Carlo MI tests yielding the lowest FDR rate.

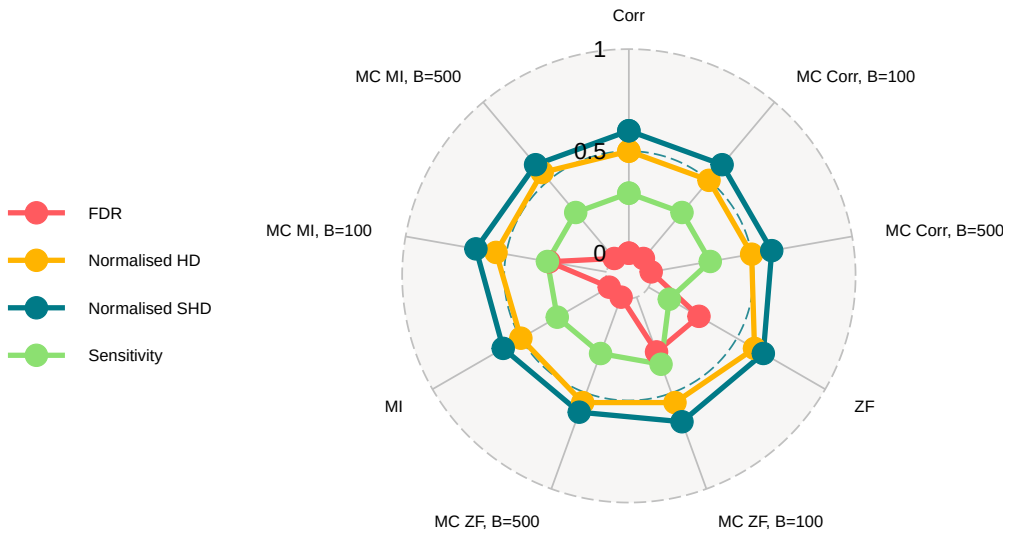

Figure 4.4: Performance of H2PC algorithm on RAF data. We see here that the t-tests (Corr) have a zero FDR measurement. All algorithms apart from the exact Z-Fisher test have very close similarities of around 0.25.

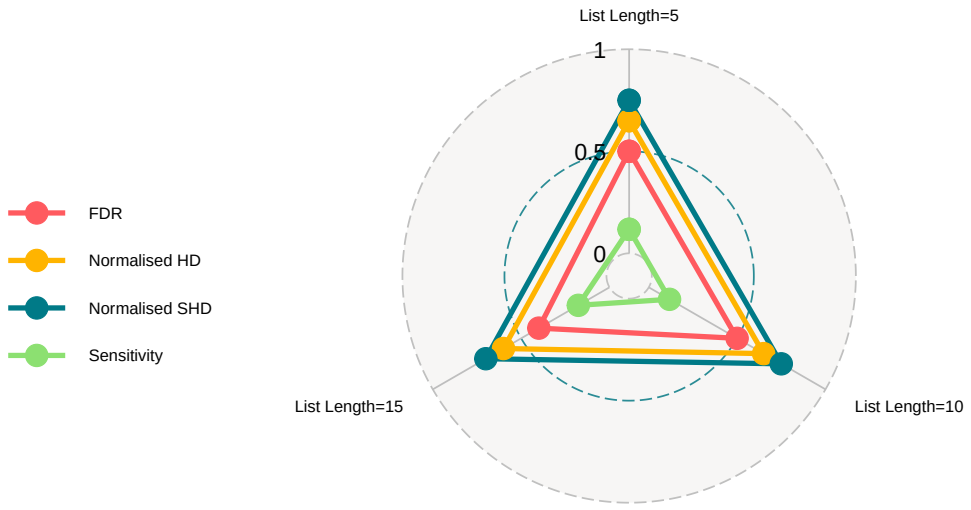

Figure 4.5: Performance of TABU algorithm on RAF data. The FDR values obtained are nearly twice as large as the corresponding sensitivity measurements, suggesting that any networks learned with non-Gaussian data would contain many false positives.

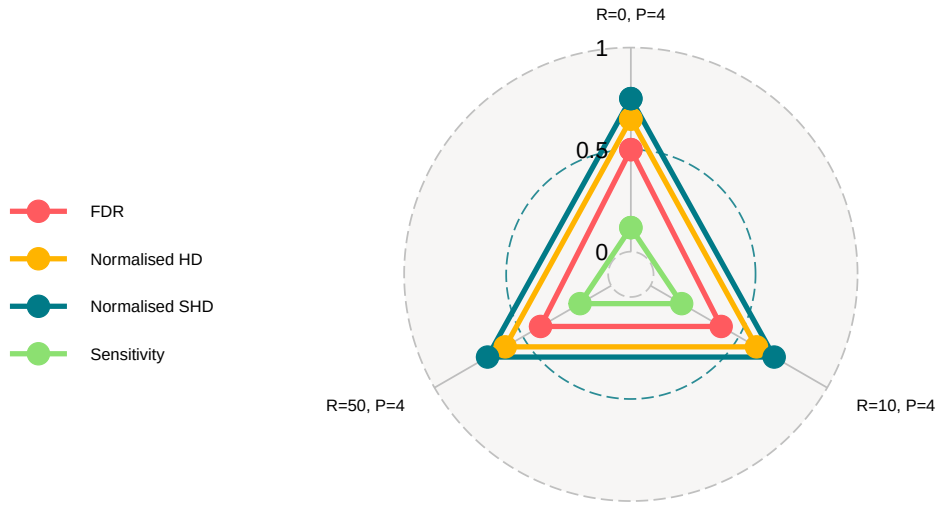

Figure 4.6: Performance of HC algorithm on RAF data. The FDR values obtained are nearly twice as large as the corresponding sensitivity measurements, suggesting that any networks learned with non-Gaussian data would contain many false positives.

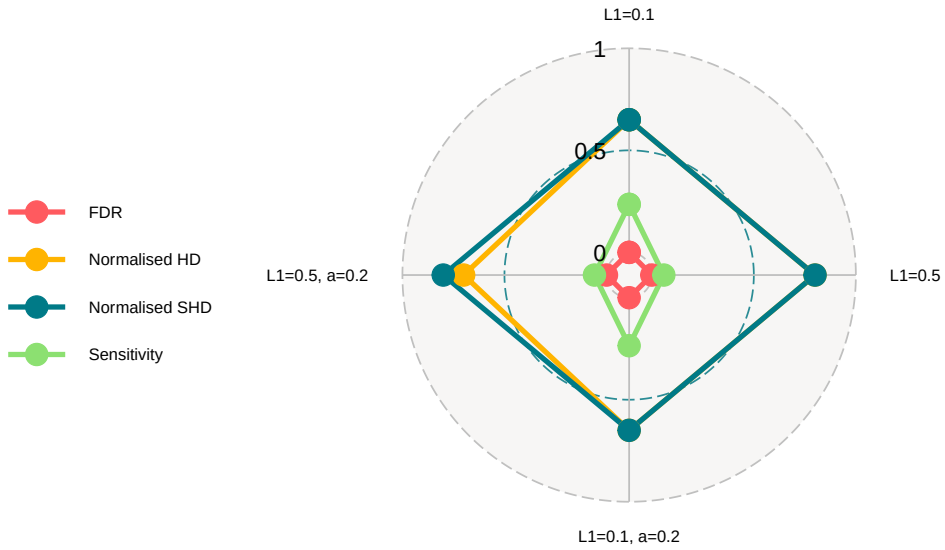

Figure 4.7: Performance of NOTEARS algorithm on RAF data. Both hybridised and non-hybridised methods demonstrated the lowest FDR and sensitivity measurements out of all algorithms, with very few edges being learned.

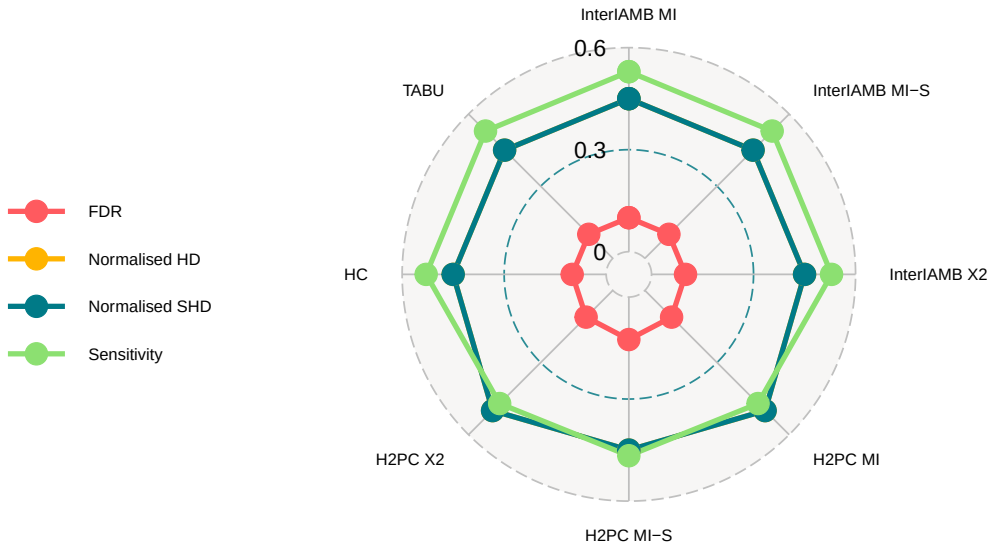

Figure 4.8: Performance of structure learning on discretised RAF data. All algorithms demonstrated a consistent FDR of around 0.1, which is higher than the continuous alternatives. However, we note that the sensitivity is much higher for all discrete algorithms than the continuous methods, with InterIAMB MI achieving the maximum sensitivity of 0.53.

Chapter 5

Impact of Hidden Network Pathways on Structure Learning

In practice, one may only have access to a subset of nodes from the true graph. Consider the case where nodes X and Y are not adjacent in a graph. If the set of nodes which d-separate X and Y are unknown, a dependency between the two exists. Here, we say that indirect pathways exist between X and Y. The number of indirect pathways between X and Y corresponds to the total number of unique paths in the graph that can take us from X to Y.

DEFINITION 21. *A **direct pathway** between nodes X and Y exists if the two are adjacent to each other in the graphical model. An **indirect pathway** between two nodes X and Y exists if the two nodes are marginally dependent. The nodes must be d-separated by a non-empty set of nodes. This set is referred to as the **d-separator set**.*

*The total number of unique indirect pathways which connect X and Y is defined as the **number of indirect pathways** between X and Y.*

Consider the case where the nodes which connect the multiple indirect pathways between $X \to Y$ are not experimentally observed. It is possible that a significant amount of noise will be introduced in the measurement of Y since each unobserved node in the indirect pathways introduces stochasticity into the system. This may hinder the detection of dependencies between nodes which are d-separated by many hidden nodes. To help illustrate this concept, consider the example in 5.1. The left hand network contains a direct pathway between $B \to D$ (they are adjacent in the graph). Two indirect pathways connect $A \to D$ through the routes that are d-separated by E and F, whereas four indirect pathways exist between $C \to D$. The latter are d-separated by the nodes G, H, I, and J. If the d-separator set for this graph is unobserved, these indirect pathways are effectively hidden since in the absence of further information, we do not know whether these pathways exist. The right hand network illustrates the BN that describes the system if all indirect pathways are hidden.

To assess this impact caused on the learned network by hidden indirect pathways, simulated data was generated from the network in Figure 5.1 with a variance of $\sigma^2 = 1$ assigned to each node and a fixed weight assigned to all edges. Data ($N = 1000$) was drawn from the network across the range of weights $w \in \{0.01, 0.05, 0.1, 0.5, 1, 5, 10, 25, 50, 100\}$ where the edges for a linear

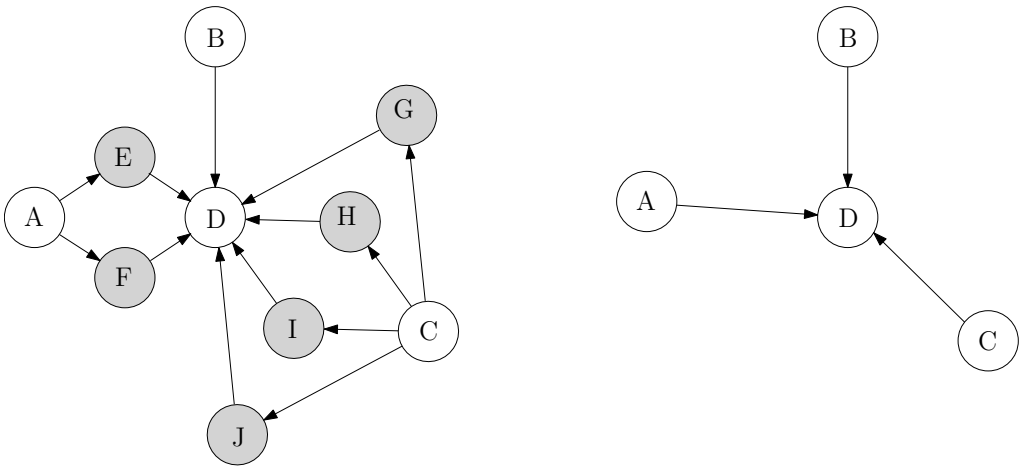

Figure 5.1: Bayesian Network used to assess the impact of indirected pathways on structure learning. Simulated data was drawn from the full linear Gaussian network (left) but structure learning was only performed on the data corresponding to nodes A, B, C, and D (right). The greyed out variables correspond to d-separator set of the network.

multivariate Gaussian BN are defined in Definition 13 and Equation 2.12. The network was learned with a H2PC algorithm with $\alpha = 0.2$, Restart number $= 50$ and perturbation number $= 2$. One hundred bootstraps of size $N_B = 200$ were used.

Figure 5.2 shows that for small edge weights ($w \lesssim 0.5$), the direct pathway BD is significantly more likely to be detected than the indirect connections between AD and CD. This effect is visible at $w = 0.1$ where $P(BD|\mathcal{D}) \approx 0.92$ whereas the next most likely edge is $P(CD|\mathcal{D}) \approx 0.18$. As the weights approach $w = 1$, all edges are detected in all bootstraps. However, the probability that the direct pathway is detected starts to drop at around $w = 5$ and is the least likely edge to be detected for $w > 5$. The connection CD with four indirect pathways is always more likely to be detected than AD, with only two indirect pathways. This suggests that connections between nodes with higher numbers of indirect pathways are easier to detect than those with a smaller number of indirect pathways.

In an attempt to explain these observations, the mutual information associated with insertion of edges for each case is considered. As discussed in Section 2.2.2, the increase in the likelihood by attaching an edge linking X and Y corresponds to their mutual information $MI(X, Y)$. Therefore, we consider the amount of information gained in adding a connection between the observed nodes in each case, and hypothesise that edges are more frequently assigned to nodes when they lead to a higher information gain. The theoretical framework used to investigate this observation is illustrated in Figure 5.3.

Consider first the example illustrated at the top half of Figure 5.3. Under the true model, there exist N pathways connecting the observed nodes X and Y with the node I_i connecting the i^{th} pathway between X and Y. Let the root node have a Gaussian distribution $X \sim \mathcal{N}(\mu_X, \sigma^2)$ and assume that all downstream nodes are linear Gaussians with linear weights, w. The conditional

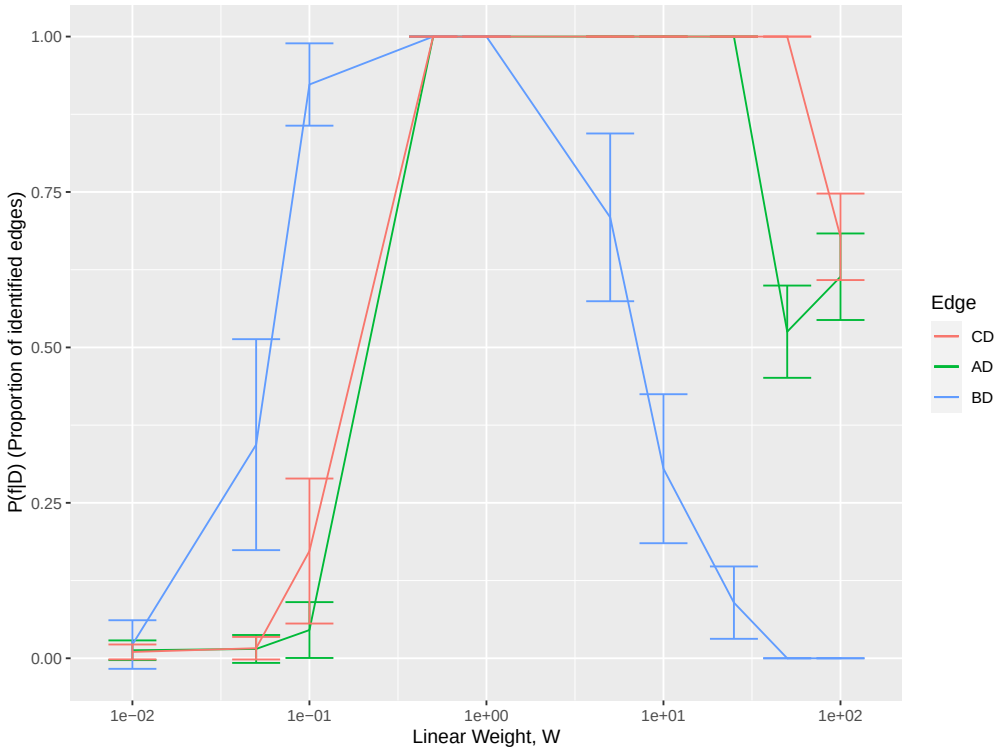

Figure 5.2: Proportion of bootstrapped models which identified a particular edge (approximating the posterior, $P(f|\mathcal{D})$) plotted against the linear weight of the linear Gaussian model. The uncertainty bounds correspond to the standard deviation of $P(f|\mathcal{D})$.

variables $I_i \mid X$ and $Y \mid \{I_1, \ldots, I_N\}$ can be written as.

$$I_i \mid X = wX + \epsilon_{I_i} \qquad Y \mid \{I_1, \ldots, I_N\} = w \cdot \sum_{i=1}^{N} I_i + \epsilon_Y \tag{5.1}$$

where $\epsilon_{I_i}, \epsilon_Y$ are independent Gaussian noise variables $\epsilon_i \sim \mathcal{N}(0, 1)$. We can use Equation 5.1 to express direct conditional relationship between Y and X as follows:

$$Y \mid X = Nw^2 X + w \sum_{i=1}^{N} \epsilon_{I_i} \tag{5.2}$$

$$\implies \mathbf{E}[Y|X] = Nw^2 X \quad \text{and} \quad \text{Var}[Y|X] = \sigma^2(Nw^2 + 1) \tag{5.3}$$

where the conditional expectations and variances follow naturally from the definition of conditional Gaussians. In the case where the intermediate nodes are omitted from the model (illustrated in the central BN in Figure 5.3), this corresponds to the true probabilistic representation that models X and Y.

We may write the joint form using the distribution of X, the conditional linear Gaussian

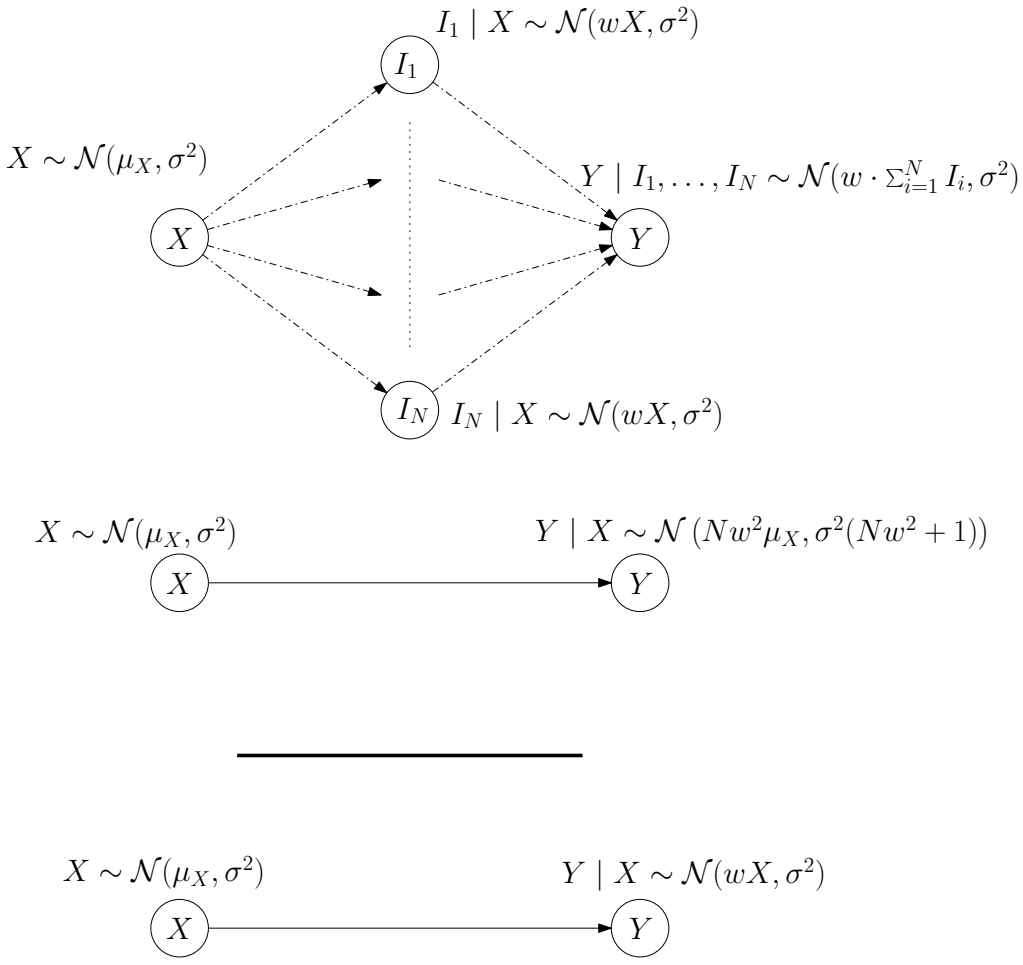

Figure 5.3: Probabilistic linear Gaussian model from the bootstrapped experiment. The top diagram connects X to Y by a set of N intermediate nodes I_n. The middle diagram illustrates the probabilistic model connecting X to Y in the absence of intermediate nodes. The bottom diagram illustrates the probabilistic graphical model between two immediately connected nodes.

likelihood $Y \mid X$, and Theorem 15:

$$\begin{pmatrix} X \\ Y \end{pmatrix} \sim \mathcal{N}\left(\begin{pmatrix} \mu_X \\ Nw^2\mu_X \end{pmatrix}, \sigma^2 \begin{pmatrix} 1 & Nw^2 \\ Nw^2 & N^2w^4 + Nw^2 + 1 \end{pmatrix}\right). \tag{5.4}$$

The mutual information associated with the central BN in Figure 5.3 can be readily calculated. The mutual information for two variables X and Y is given by:

$$MI(X,Y) = H(X) + H(Y) - H(X,Y) \tag{5.5}$$

and the joint entropy for a multivariate Gaussian:

$$h(\mathbf{X}) = \frac{1}{2} \ln\left(\det\left|2\pi e \boldsymbol{\Sigma}\right|\right) \tag{5.6}$$

$$= \frac{1}{2} \ln\left((2\pi e)^2 \left(\sigma_X^2 \sigma_Y^2 - \sigma_{XY}^4\right)\right) \tag{5.7}$$

where the second line is the joint entropy for the bivariate distributions considered in this example.

To assess the propensity for a structure learning algorithm to insert a conditional edge between two nodes, consider the difference in mutual information between a connected and unconnected model:

$$\begin{aligned} MI(X,Y)_{conn} - MI(X,Y)_{unconn} &= H(X,Y)_{unconn} - H(X,Y)_{conn} && (5.8)\\ &= \frac{1}{2}\ln\left(\sigma_X^2\sigma_Y^2\right) - \frac{1}{2}\ln\left(\sigma_X^2\sigma_Y^2 - \sigma_{XY}^4\right) && (5.9)\\ &= \frac{1}{2}\ln\left(\frac{\sigma_X^2\sigma_Y^2}{\sigma_X^2\sigma_Y^2 - \sigma_{XY}^4}\right) && (5.10)\\ &= \Delta MI(X,Y)_{undir} && (5.11) \end{aligned}$$

Substituting in the variance and covariance terms derived above:

$$\begin{aligned} \Delta MI(X,Y)_{undir} &= \frac{1}{2}\ln\left(\frac{N^2w^4 + Nw^2 + 1}{N^2w^4 + Nw^2 + 1 - (Nw^2)^2}\right) + \ln\sigma \\ &= \frac{1}{2}\ln\left(\frac{N^2w^4 + Nw^2 + 1}{Nw^2 + 1}\right) + \ln\sigma \\ &= \frac{1}{2}\ln\left(Nw^2 + \frac{1}{Nw^2 + 1}\right) + \ln\sigma. \end{aligned} \tag{5.12}$$

Note that $\Delta MI(X,Y)$ is a monotonically increasing function with respect to both N and w. This aligns with the experimental observation that the connection BD with four indirect pathways is always detected at a higher frequency than AD with only two. In the limit of large Nw^2, the separable relation between $N, \mid w$ and MI is given by:

$$\Delta MI(X,Y)_{undir} \xrightarrow{Nw^2\to\infty} \frac{1}{2}\ln\left(Nw^2\right) + \ln\sigma = \frac{1}{2}\ln(N) + \ln(w) + \ln(\sigma). \tag{5.13}$$

Now consider the mutual information gain from adding an edge to two nodes which are directly connected in the true network. Using the same Gaussian equations as in the indirect pathway case, the full joint can be expressed as:

$$\begin{pmatrix} X \\ Y \end{pmatrix} \sim \mathcal{N}\left(\begin{pmatrix} \mu_X \\ w\mu_X \end{pmatrix}, \sigma^2 \begin{pmatrix} 1 & w \\ w & w^2+1 \end{pmatrix}\right). \tag{5.14}$$

Here, the mutual information gain achieved by connecting X to Y is:

$$\begin{aligned} \Delta MI(X,Y)_{dir} &= \frac{1}{2}\ln\left(\frac{1+w^2}{1+w^2-w^2}\right) + \ln\sigma \\ &= \frac{1}{2}\ln\left(1+w^2\right) + \ln\sigma. \end{aligned} \tag{5.15}$$

The behaviour of $\Delta MI(X,Y)_{dir}$ and $\Delta MI(X,Y)_{undir}$ is such that for small weights, the directed pathways have a greater increase in MI when they are included, whereas pathways with numerous indirect pathways are more likely to be detected with higher linear weights. The critical

weight w^* where this transition occurs can be found by equating Equations 5.12 and 5.15:

$$1 + w^* = \frac{N^2 w^{*4} + N w^{*2} + 1}{N w^{*2} + 1} \tag{5.16}$$

$$\implies w^* = \frac{1}{\sqrt{N^2 - N}}. \tag{5.17}$$

Interestingly, this result can be be easily intuited by approaching the problem from a perspective of linear regression. For $w < 1$, the true linear relationship between $Y \mid X$ is $O(w^2)$ and thus smaller than the directed linear relationship $O(w)$. The reduction in unexplained variance would therefore be small (which is analogous to the increase of MI) whereas the proportion of explained variance would increase with larger w.

The experiment reported here highlights some critical considerations when structure learning is attempted on nodes drawn from a subset of the complete variable set. In the case of pharmaceutical data, there may be several thousand genes of which only a small fraction will be subject to structure learning. It is possible to predict that edges may be inferred between genes which are not directly connected. If the biological system is well modelled by linear weights $w > 5$, structure learning algorithms may be more likely to learn indirect pathways rather than direct ones. In the case of drug treatment targets where there may be a vast number of possible genetic pathways which can affect a particular gene, this poses a risk of unexpected side effects. Extreme care should be taken when using inferred models of biomedical systems which may contain indirect pathways between relevant nodes.

Chapter 6

Structure Learning on Schizophrenia Data

In this chapter, we utilise the results of the previous sections to learn genetic pathways associated with Schizophrenia (SCZ). We highlight 14 directional relationships that have been identified across two different experimental datasets which we recommend should be directly validated via additional experiments.

Several medical studies have been conducted with the aim of identifying key genetic factors related to SCZ. However, while specific risk loci are well documented in the literature, the exact genetic pathways manifesting from these genes have not been identified. This is a critical issue from a pharmaceutical perspective. While it is certainly essential to identify important genes related to the disease, it may not be possible to engineer drugs which can target them directly. Upstream or downstream genes may prove more amenable for drug targeting and by doing so may regulate the expression of downstream genes which are critical in the production of proteins that can cause SCZ. Furthermore, the over or under expression of a particular critical gene may not be a principle cause of an illness; it may impact the expression of a downstream gene in the network which may be the cause of the malady. Identifying the network pathways around these genes would be of immense value to the pharmaceutical sector.

In this section, we use gene expression data derived from an RNASeq dataset generated by the Liber Institute for Brain Development (LIBD) along with the H2PC algorithm to learn critical genetic pathways [6]. The LIBD dataset is not only extremely high-dimensional but is also far larger than any of the networks assessed in this report ($N \ll D$ at $N = 343, D = 16,828$). Not only would the accuracy of the returned networks be extremely dubious, but also the computational costs for learning the entire structure make it infeasible to do. To obtain a more manageable dataset, the most significantly expressed genes between SCZ and control (CTL) patients were identified. The marginal distributions for each of the genes is strictly non-negative, and often highly skewed. All marginal distributions caused rejection of the null hypothesis at the $p = 0.001$ level using the Shapiro-Wilks test of normality. As a result, differential expression was tested using the two-tailed nonparametric Mann-Whitney U rank test which tests the null hypothesis that the one distribution is stochastically greater/smaller than the other. The distribution for each gene

Sex\Race	African American	Caucasian
Female	64	43
Male	116	128

Table 6.1: Breakdown of race and sex characteristics for LIBD dataset.

Sex\Race	Non-Caucasian	Caucasian
Female	23	79
Male	37	192

Table 6.2: Breakdown of race and sex characteristics for CMC dataset.

was partitioned into CTL ($N = 196$) and SCZ ($N = 155$) patients and were subsequently tested. The genes were ranked by significance and the genes with the ten smallest p-values were selected.

6.1 Data Standardisation

It has been well established via experimental investigations that differences in gene expressions can arise based on human traits such as sex, age, and race. If no attempt is made to account for these, we risk drawing incorrect inferences about the conditional relations in the network. Ideally, networks could be learned from each subcategory of these factors, but unfortunately this would greatly reduce the size of our data. In an attempt to mitigate this, each gene was fitted with a linear model:

$$y_{expression} = \zeta_{Sex} + \zeta_{Race} + \beta_{Age} \cdot \text{Age} + \beta_{RIN} \cdot \text{RIN} + \beta_{PMI} \cdot \text{PMI} + \text{const} \tag{6.1}$$

where ζ_{Sex} and $\zeta Race$ are categorical offsets corresponding to the respective sex and race categories [45]. The breakdown for each of these categorical factors is documented in Table 6.1. Information on whether the subject was a smoker was also provided but was not deemed to be a significant feature ($p = 0.89$). The fixed variables modelled are the subject's age, their postmortem interval (PMI) and the RNA integrity number (RIN). An expected expression level was calculated for each subject, and was subtracted from the original values such that the network structure is learned from the residuals.

A more thorough analysis and linear model selection was limited due to time restrictions but is encouraged should this work be repeated.

6.2 Learning Pathways using Discretised Data

The key steps followed in practical implementation of structure learning algorithms are as follows. Firstly, a reduced set of around 10 to 15 critical nodes associated with SCZ is identified, and their Markov Blankets are calculated [45]. The network associated with the union of these genes is then calculated. We demonstrate that in this example, data discretisation in fact returns a much sparser network than continuous structure learning despite the non-Gaussian nature of the data.

To complete the set of modellable genes, the Markov Blankets for each of the top ten preferentially expressed genes were calculated. The SI-HITON-PC algorithm was used here as its runtime scales extremely well with large node set sizes. While it is expected that linear Gaussian assumptions do not hold in this scenario, the results in Section 4.3.1 suggest there was not a significant difference in performance when using non-parametric Monte Carlo tests versus their exact parametric counterparts. As a result, exact tests were used rather than their MC counterparts. The

hyperparameter $\alpha = 0.2$ was chosen such that the Markov Blankets were selected as generously as possible such that as many TPs were included in the modellable subset.

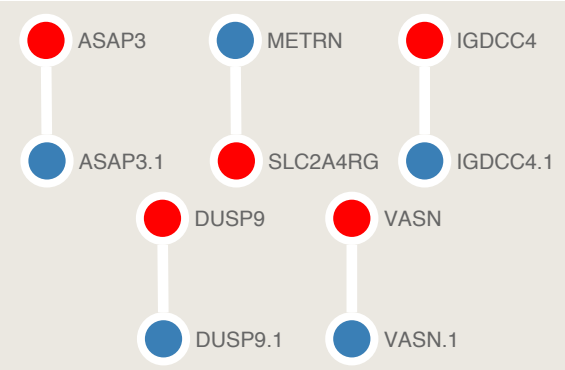

Figure 6.3: Network learned from discretised LIBD data.

A discrete H2PC algorithm was initially used to learn the network. The hypermeters used ($\alpha = 0.2$, Perturbations= 50, Restarts= 50) were those which performed optimally in Section 3. The result is shown in Figure 6.3 where red nodes represent the core nodes used to find the MB, and the edges representing the posterior probability of the connection. The returned model was both extremely sparse and very small ($D = 10$, Edges $= 5$), predicting five separate subgraphs each with only two nodes. This does not provide much illumination into the dynamics of the system, and so the continuous H2PC algorithm was run to learn a more descriptive network. This choice was motivated the fact that H2PC performed best on the ECOLI experiments (see Section 3.2), datasets larger than $N = 250$ (see Section 3.3) and on non-Gaussian data (see Section 4.3.1). The resultant network (shown in 6.4) was much more detailed ($D = 64$, Edges=67) with a subgraph of 27 nodes. Note that all probabilistic relations from the discrete network also manifested in the continuous case, making the latter consistent with and more descriptive than the former.

6.3 Learning on Experimentally Validated Schizophrenia Risk Loci

As a final experiment, we repeat this process but rather of selecting significant nodes via the Mann-Whitney test, we take a set of 13 key experimentally validated differentially expressed genes (CUL3, EP300, SLC7A6, NCAN, SGSM2, PTPRF, CCDC39, ESAM, ZDHHC5, DPEP2, GRIN2A, DPYD, ZNF536) from the Schizophrenia Working Group of the Psychiatric Genomics Consortium, 2014 [46]. In this step, an additional SCZ dataset generated by the CommonMind Consortium (CMC) is also analysed and is used to cross reference any key findings across different experimental studies [7]. This data consists of RNASeq measurements taken from 159 SCZ and 172 control patients. The data was normalised in the same way as documented in Section 6.1. The CMC dataset breakdown of race and sex is documented in Table 6.1. The networks inferred from the LIBD and CMC datasets are shown in Figures 6.5 and 6.6. The CMC network consists of only a single graph ($D = 64$, Edges $= 68$) whereas the LIBD network ($D = 65$, Edges $= 58$) consists of nine subgraphs with the largest consisting of 16 nodes. Unfortunately, there does

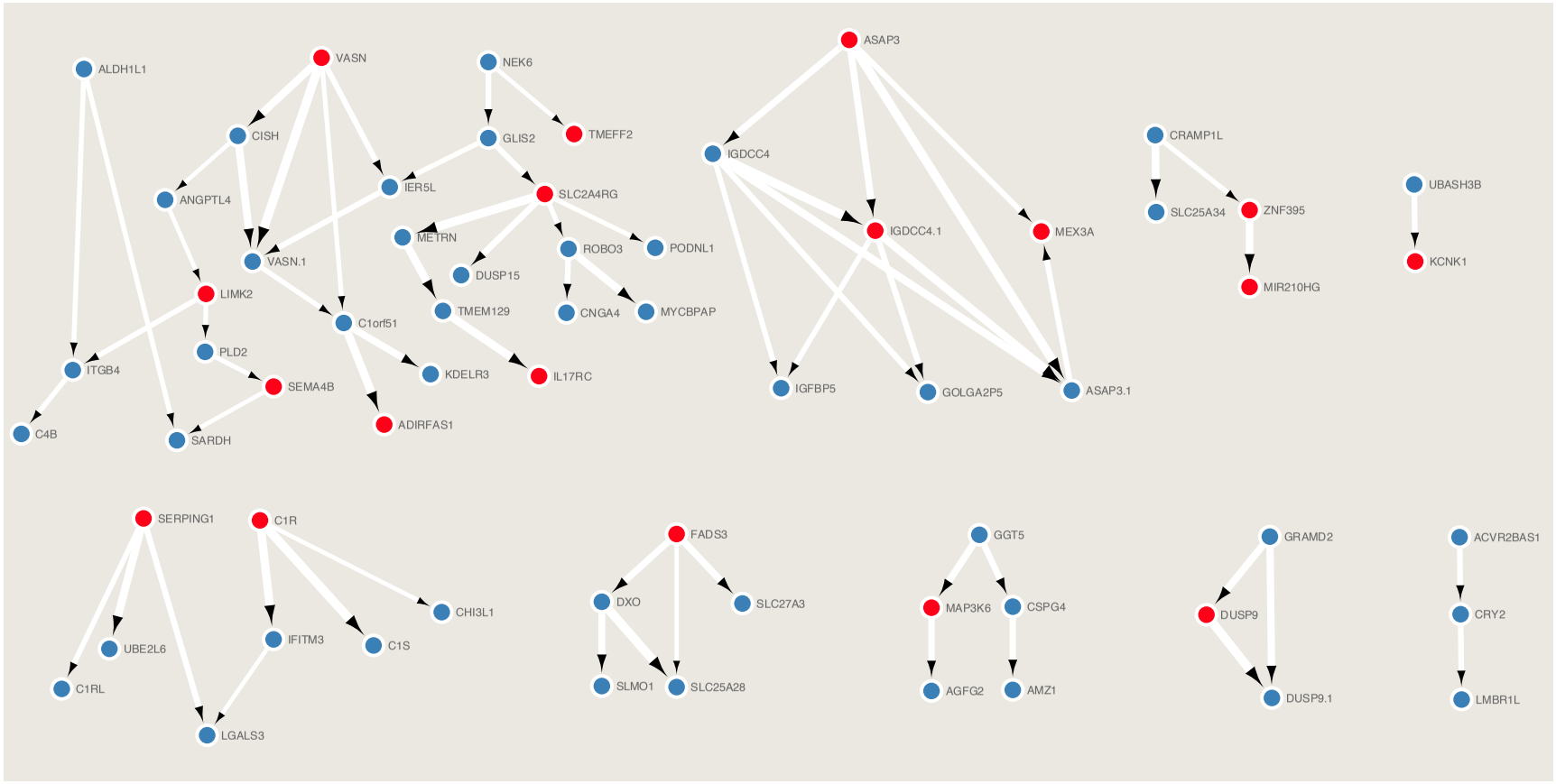

Figure 6.4: Network learned from continuous LIBD data.

not appear to be a great degree of similarity between the two. Apart from the core set of 13 genes, the networks only share two genes (MMD2, MINK1) which it identified via the Markov Blanket discovery (which were both approximately 60-70 nodes). It is encouraging to see that it identified the same probabilistic relationship in both cases (PTPRF→MINK1) and (NCAN→ MMD2), however it is surprising that more similarities were not discovered. Only three nodes occurred in both MB sets prior to structure learning, and it is possible that that this difference may be caused by inappropriate model specification during the gene expression normalisation. The vast number of genes implies that investigating this further is a time intensive exercise and beyond the scope of the thesis; however, it is recommended as a potential avenue of future research.

Given the vast discrepancy in MB detection, the experiment was repeated using the *union* of separate MB node sets (a total of 110 genes) as the feature set. The resultant networks are shown in Figures 6.7 and 6.8 which are of size ($D = 84,\ \text{Edges} = 76$) and ($D = 93,\ \text{Edges} = 92$) respectively. There are 14 directed edges that are identified across both networks; these are shown as blue arrows in the network figures. For clarity, these connections are plotted separately in Figure 6.9. The fact that these edges were identified across different experimental datasets as well as generally being quite strongly significant is extremely encouraging and could be first targets for experimental validation. However, it should be emphasised that the absence of nodes in one network that were observed in another do not imply that that such connections do not exist. The experimental differences in the LIBD and CMC as well as the aforementioned model specification may be partially to blame for such an inconsistency.

To summarise, this section presented a variety of different networks identified for differentially expressed genes identified using non-parametric tests and a set of 13 critical SCZ genes identified in the literature. While large differences exist across the networks inferred from the separate experimental data, they shared 14 edges which should be directly probed using further experimental investigations. Nevertheless, the differences between the two networks should be further studied and contextualised by a domain expert. It is possible that they can be combined to construct a more comprehensive description of the true underlying network.

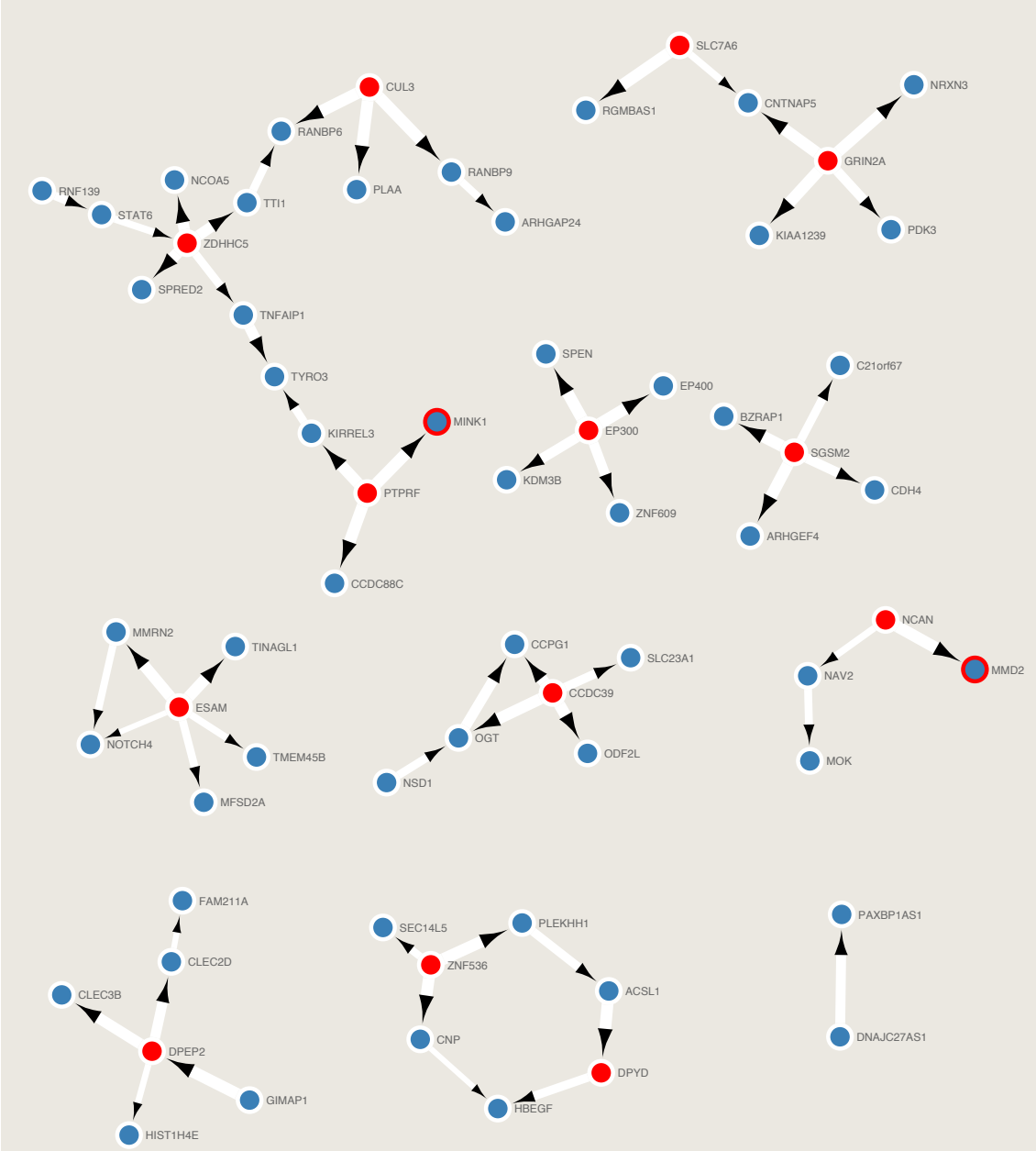

Figure 6.5: Network inferred from LIBD data. The filled red nodes are the 13 genes taken from [46]. The blue nodes with a red border are the nodes which were found through via Markov Blanket detection and structure learning across both LIBD and CMC datasets.

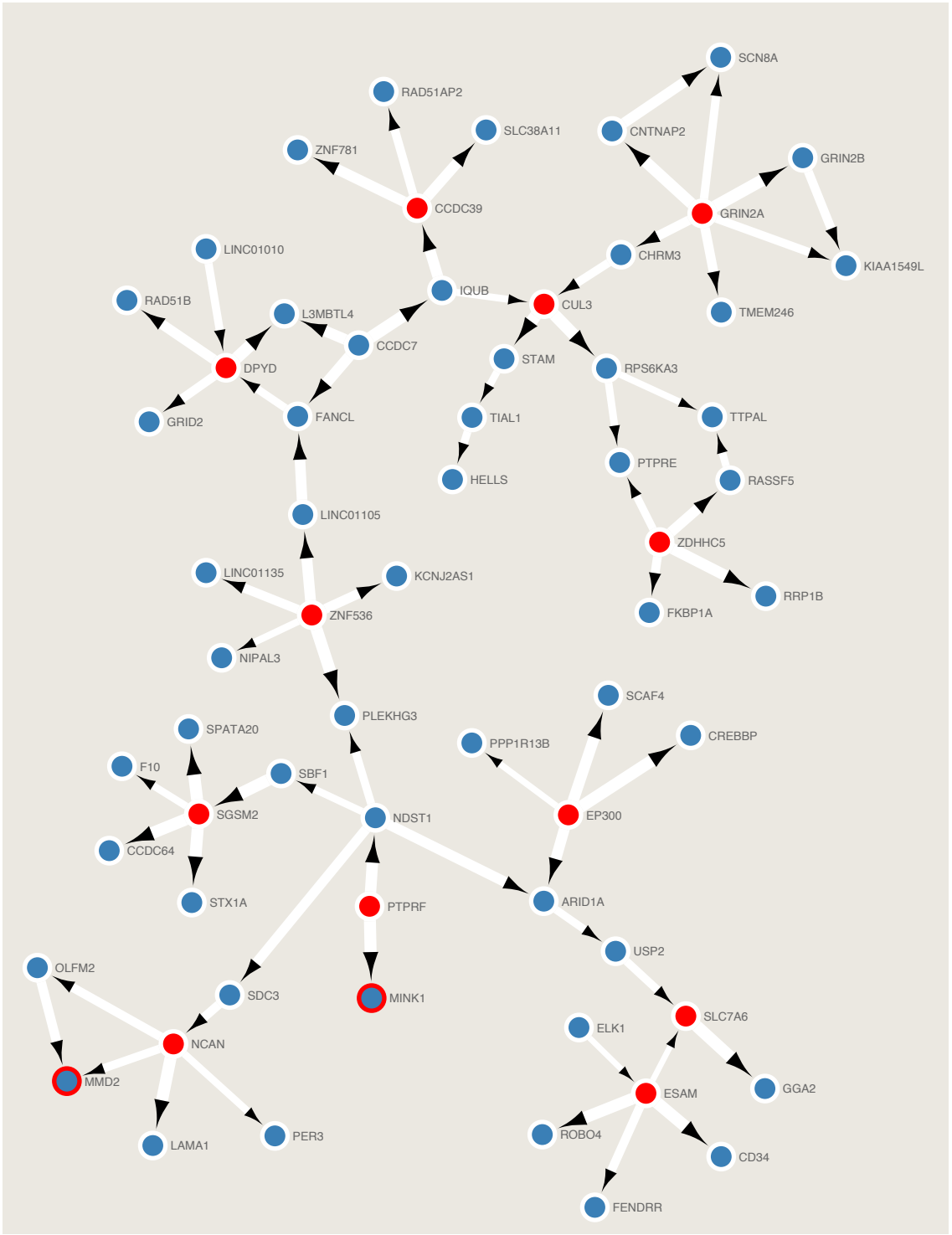

Figure 6.6: Network inferred from CMC data. The filled red nodes are the 13 genes taken from [46]. The blue nodes with a red border are the nodes which were found through via Markov Blanket detection and structure learning across both LIBD and CMC datasets.

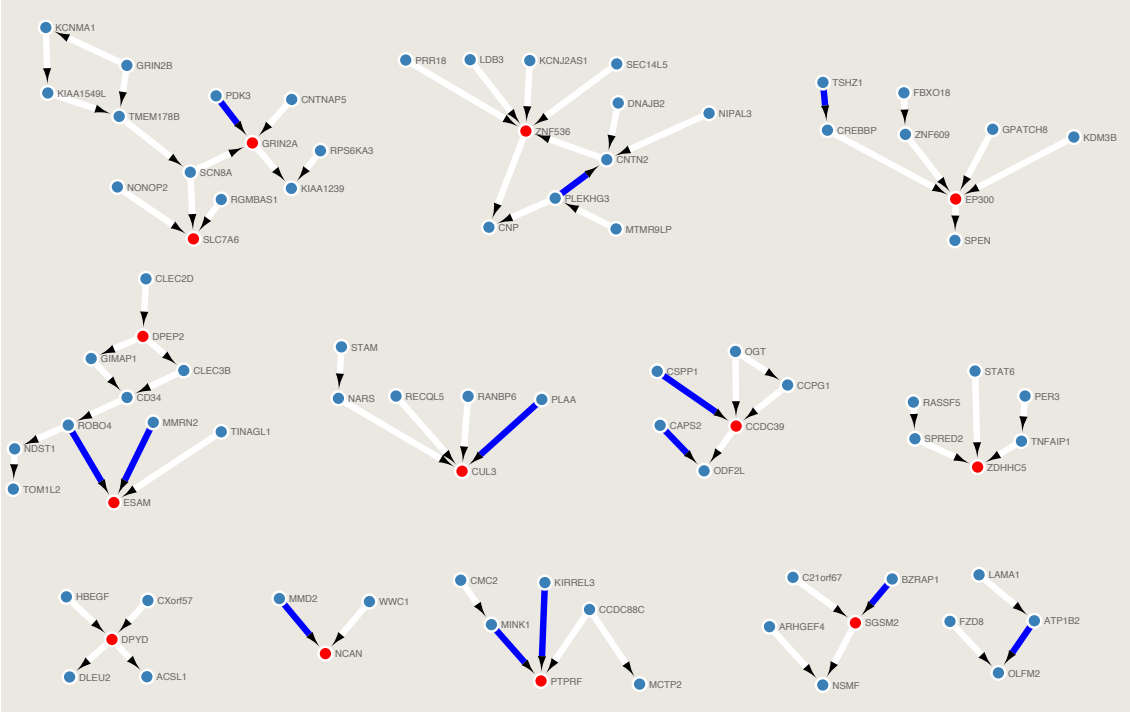

Figure 6.7: Network inferred from LIBD data using union of LIBD and CMC Markov Blankets. The filled red nodes are the 13 genes taken from [46]. The blue arrows correspond to edges shared between the CMC and LIBD inferred networks.

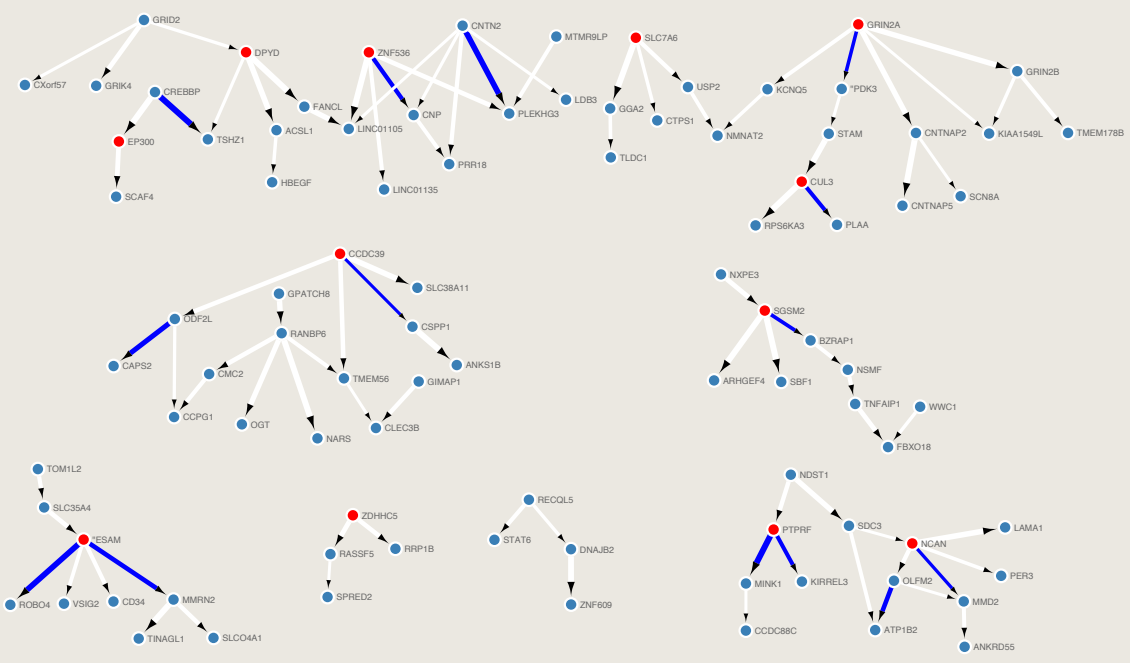

Figure 6.8: Network inferred from CMC data using union of LIBD and CMC Markov Blankets. The filled red nodes are the 13 genes taken from [46]. The blue arrows correspond to edges shared between the CMC and LIBD inferred networks.

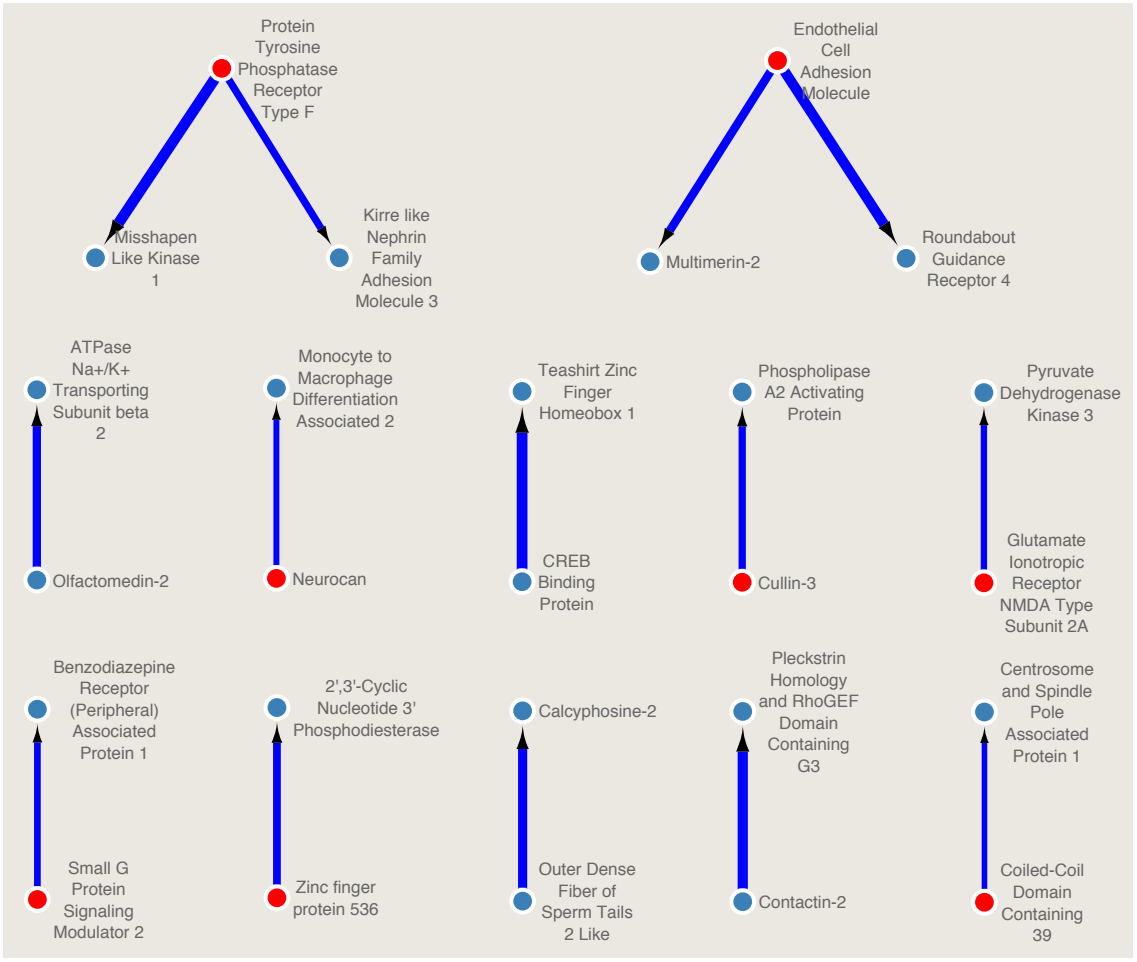

Figure 6.9: Shared edges identified across learned LIBD and CMC networks. The full names of the genes are used for additional legibility to biologically inclined readers.

Chapter 7

Conclusions and Further Work

The central goal of this report was to infer probabilistic Bayesian Networks for critical genetic pathways which occur in Schizophrenia patients. We began by presenting a review of graphical models and then contextualised the meanings of graph edges in probabilistic conditional relationships. Bayesian Networks were introduced as a more descriptive model which could capture both directed conditional dependencies and marginal relationships which are not inferable using their undirected counterparts. Concepts which are central to structure learning algorithms (such as d-separation and Markov Blankets) were also discussed.

A review of traditional and more state of the art structure learning algorithms followed. Constraint-based algorithms revolved around the identification of Markov Blankets prior to resolving them into consistent graphical models, whereas conventional score-based algorithms involve the specification of a score function and use heuristic local search algorithms to identify high scoring networks. The theoretical background behind appropriate score function specification was reviewed and shown to be consistent with identifying networks which maximise the mutual information of the system. Hybrid algorithms hybridised both constraint and score-based algorithms and were shown to consistently outperform their separate constituent methods. The performance of NOTEARS, a state of the art regression algorithm, was also assessed. We also propose a hybridised method, NOTEARS-H, which seeks to improve its performance further.

Initial experiments were conducted on synthetic Gaussian datasets generated from both randomly created DAGS and biological networks with experimentally verified edges, weights, and variances. Experiments were also conducted to investigate the relations between datasize and graph size with algorithm performance. It was observed that the hybrid H2PC algorithm was the most performant in a variety of cases when working with small-mid to large networks with larger datasets. The NOTEARS family of algorithms performed competitively, particularly when dealing with small networks and limited datasizes. The unadulterated NOTEARS method was most performant on small graphs ($D = 25 - 50$) whereas our proposed hybrid adaptation was shown to be more effective when dealing with larger networks, particularly when dealing with scenarios where the $N \gtrsim D$.

Additionally, the effect of variable omission in bootstrapped structure learning was investigated. A simulated linear Gaussian system was created, and experiments showed that while detection of direct connections was strongly favoured for weak linear edges, indirect connections with large

numbers of omitted pathways were favoured from more responsive systems. We showed that our results were consistent with the principle that the discovery of edges which most increase the mutual information of the system are more likely to be discovered, and derived critical transition points which aligned with the results from our experiments.

Bayesian model averaging was used as a way of assessing the stability of learned graphical models. From the set of model averaging techniques, this thesis used model bootstrapping with a suitable significance threshold to identify statistically significant edges. We also reviewed the approach of Hartemink data discretisation for dealing with clearly non-Gaussian continuous datasets. Both these approaches were utilised in investigating the performance of structure learning algorithms when dealing with highly non-Gaussian data. When applied to the standard RAF Sachs benchmarking dataset, discretisation was shown to yield the best balance of true positive to false positive identification. Out of all the continuous Gaussian cases, H2PC was shown to be the most performant, identifying no false positives whatsoever.

We then turned to learning the structure of Schizophrenia genetic pathways from two novel datasets. In this case, it was found that model discretisation actually limited the number of identiable edges, and was infact a subgraph of the network learned from continuous data. Two different sets of nodes which are found to be critical in the development of Schizophrenia were identified using hypotheses tests and also from biopharmaceutical literature. When dealing with the latter, 14 identical edges were identified from networks learned from the separate detasets. We propose that these findings form the basis of further pharmaceutical experimental research.

When considering areas for future research, it is clear that application of structure learning methods is limited by the central assumption of dealing with a linear Gaussian system. In particular, we identified that skewed, non-negative data are frequently encountered when dealing with distributions of genetic expression, and that discretisation does not guarantee performance improvement in all cases. One possible extension that was considered involved the construction of a multivariate Gamma BN. The Gamma distribution $\Gamma(\alpha,\ \beta)$ appropriately characterises the non-negative and skewed data. In an attempt to draw an analogy to the derivation of multivariate Gaussian models, we note that a Gamma prior on the rate parameter β is the conjugate prior of the Gamma likelihood (assuming α is known).

Under a linearity assumption, we propose that the value of β_i is a linear combination of parent nodes:

$$X_{child} \sim \Gamma(\alpha_{child}, \mathbf{w}_{child} \cdot \mathbf{X}_{parent}) \text{ where } w_i > 0 \quad \forall \quad w_i \in \mathbf{w}_{child} \tag{7.1}$$

where $\mathbf{w}_{child}$ is the row in the weight matrix corresponding to the node X_{child}.

Assuming that it is valid to use a Gamma likelihood as a loss function, the target loss function would be:

$$l(\boldsymbol{\alpha}, \boldsymbol{\beta}) = \sum_{j=1}^{d} \left[(a_j - 1) \sum_{i=1}^{n_j} \ln x_j - n_j \ln \Gamma(a_j) - n_j a_j \sum_{i=1}^{n_j} \ln(\beta_j) - \sum_{i=1}^{n} x_i \beta_j \right] \tag{7.2}$$

where $\boldsymbol{\alpha}, \boldsymbol{\beta}$ are the corresponding vectors of α, β parameters for each node. To generalise this to a

general loss function, consider likelihood for a single datapoint drawn from a Gamma distribution:

$$l(\beta, \alpha) = a - n \ln \Gamma(a) + a \ln(\beta) - x\beta \tag{7.3}$$
$$= a \ln \beta - x\beta + f(a) \tag{7.4}$$

where we assume that the likelihood is maximised with respect to β holding α constant.

Generalising this to n datapoints, which can be generalised to d nodes:

$$l = \sum_{n,d} [a_d \ln(\beta_{nd}) - x_{nd}\beta_{nd}] \tag{7.5}$$
$$= a_d \ln(XW)_{nd} - X_{nd}(XW)_{nd} \tag{7.6}$$

where we assume β is linearly dependent on the node's parents.

Differentiating it with respect to a single weight component:

$$\frac{\partial l}{\partial W_{ij}} = a_d \frac{\partial \ln(XW)_{nd}}{\partial W_{ij}} - X_{nd} \frac{\partial \ln(XW)_{nd}}{\partial W_{ij}} \tag{7.7}$$
$$= a_d \frac{X_{ni}}{\sum_z X_{nz} W_{zd}} \delta_{dj} - X_{nd} X_{ni} \delta_{jd} \tag{7.8}$$
$$= (X^\intercal)_{in} \frac{a_j}{\beta_{nj}} - (X^\intercal)_{in} X_{nj} \tag{7.9}$$

To show that an extremum exists, we must show that the loss is convex. Taking the above derivative and setting it to zero we find that:

$$X^\intercal \left(\frac{a}{\beta}\right) = X^\intercal X \implies \left(\frac{a}{\beta}\right) = X \tag{7.10}$$

assuming the columns of the data matrix X are linearly independent. This result aligns with the MLE for a single gamma function, where $\alpha\beta = \bar{x}$. Under the assumption that β is a non-negative linear sum of the parent nodes, the solution becomes:

$$\left(\frac{a}{\beta}\right)_{nd} = X_{nd} \implies \frac{a_d}{\sum_z X_{nz} W_{zd}} = X_{nd} \implies \frac{a_d}{X_{nd}} = \sum_z X_{nz} W_{zd} \tag{7.11}$$

which, for a particular fixed value of d, is a series of N equations with D unknowns. If $N > D$, then this is an overdetermined system, which will likely be inconsistent.

Whilst the above method recursively calculates β, it does not give any indication on how to calculate the shape term, α_j. One suggestion is to update it using the mixture:

$$\alpha_{n+1} = \lambda \alpha_n + (1 - \lambda) \alpha_{update} \tag{7.12}$$

where λ is chosen to be small such that the risk of convergence issues is minimised.

A proposal for further research would specifically forgo the assumption of linearity in Gaussian systems. In the calculation of scores involving likelihood terms, the linear weights and variances for the Gaussian components is calculated via linear regression. One possible relaxation would be to model each node as a Gaussian Process (GP) instead, with the child node as the response, and

the parent nodes as independent input variables. The likelihood component calculated from the fitted posterior GP would contribute to the score.

The key problem with both these approaches is that they do not ensure that the full network represents a multivariate Gaussian or Gamma distribution. In the case of the linear Gamma model considered above, consider the following model:

$$X \sim \Gamma(\alpha, \beta) \qquad Y \mid X \sim \Gamma(\gamma, wX) \tag{7.13}$$

where w is a positive constant. The full joint is:

$$\begin{aligned} P(X,Y) &= P(X)P(Y|X) && (7.14)\\ &= \frac{(wX)^{\gamma}}{\Gamma(\gamma)} Y^{\gamma-1} e^{-wXY} \cdot \frac{\beta^{\alpha}}{\Gamma(\alpha)} X^{\alpha-1} e^{-\beta X} && (7.15)\\ &\propto (wX)^{\gamma} Y^{\gamma-1} e^{-wXY} X^{\alpha-1} e^{-\beta X} && (7.16)\\ &\propto Y^{\gamma-1} X^{\gamma+\alpha-1} e^{-X(\beta+wY)} && (7.17) \end{aligned}$$

To find the marginal distribution of node Y, we integrate over X:

$$\begin{aligned} P(Y) &= \int P(X,Y) dX && (7.18)\\ &\propto Y^{\gamma-1} \int X^{\gamma+\alpha-1} e^{-X(\beta+wY)} dX. && (7.19) \end{aligned}$$

Noting that the term inside the integral is an unnormalised Gamma distribution with parameters $\alpha = \gamma, \beta = wX$, the unnormalised marginal is equal to:

$$\begin{aligned} P(Y) &\propto Y^{\gamma-1} \cdot \frac{\Gamma(\gamma+\alpha)}{(\beta+wY)^{\gamma+\alpha}} && (7.20)\\ &\propto \frac{Y^{\gamma-1}}{(\beta+wY)^{\gamma+\alpha}}. && (7.21) \end{aligned}$$

$P(Y)$, being the marginal distribution of the observed child node, is not skewed and thus is not an appropriate function for modelling the kind of data observed earlier in this thesis. This highlights the need to ensure that the distributions used to construct the probabilistic factors result in a marginal distribution that is consistent with the observed data. This is also an issue if GPs are used to model the mean of the distribution, as the loss of linearity would imply that the prior and thus the marginal of the child will no longer be normal.

The above discussions highlight the difficulty with deviating from a linear Gaussian model when working with continuous structure learning algorithm. However, finding a method which copes well with skewed data would certainly allow for the application of graphical structure learning to a wide variety of biological datasets and is a avenue of research well worth exploring.

When it comes to further developing constraint-based algorithms, we look to the development of independence tests which can deviate from linear Gaussian assumptions. Measuring statistical dependence using Hilbert-Schmidt norms shows significant promise in resolving these issues. In particular, the application of the Hilbert-Schmidt Independence Criterion as an independence measure appears to be a very promising development [47]. Adapting existing constraint-based

algorithms to incorporate these methods would be an extremely valuable area of further research and appears considerably more achievable than the Gamma/GP developments mentioned above.

Bibliography

[1] Office for National Statistics. *Leading Causes of Death, UK: 2001 to 2018*. Latest Release: 27 March 2020.

[2] Monika Szkultecka-Debek et al. "Schizophrenia causes significant burden to patients' and caregivers' lives". In: *Psychiatria Danubina* 28.2 (2016), pp. 104–110. ISSN: 03535053.

[3] A. Fasseeh et al. "A systematic review of the indirect costs of schizophrenia in Europe". In: *European Journal of Public Health* 28.6 (2018), pp. 1043–1049. ISSN: 1464360X. DOI: 10.1093/eurpub/cky231.

[4] G. Kovács et al. "Direct healthcare cost of schizophrenia – European overview". In: *European Psychiatry* 48 (2018), pp. 79–92. ISSN: 17783585. DOI: 10.1016/j.eurpsy.2017.10.008.

[5] Graham Thornicroft et al. "The personal impact of schizophrenia in Europe". In: *Schizophrenia Research* 69.2-3 (2004), pp. 125–132. ISSN: 09209964. DOI: 10.1016/S0920-9964(03)00191-9.

[6] Giulio Pergola et al. "Gene Co-Expression Reveals Pathways of Convergence of Schizophrenia Risk Genes". In: *European Neuropsychopharmacology* 29 (2019), pg1013. ISSN: 0924977X. DOI: 10.1016/j.euroneuro.2017.08.413. URL: http://dx.doi.org/10.1016/j.euroneuro.2017.08.413.

[7] Gabriel E. Hoffman et al. "CommonMind Consortium provides transcriptomic and epigenomic data for Schizophrenia and Bipolar Disorder". In: *Scientific data* 6.1 (2019), p. 180. ISSN: 20524463. DOI: 10.1038/s41597-019-0183-6. URL: http://dx.doi.org/10.1038/s41597-019-0183-6.

[8] Christopher Bishop. *Pattern Recognition and Machine Learning*. 2006. ISBN: 978-0-387-31073-2.

[9] Alexander Hartemink. "Principled computational methods for the validation and discovery of genetic regulatory networks. Massachusetts Institute of Technology". PhD thesis. MIT, 2001.

[10] Karen Sachs et al. "Causal protein-signaling networks derived from multiparameter single-cell data". In: *Science* 308.5721 (2005), pp. 523–529. ISSN: 00368075. DOI: 10.1126/science.1105809.

[11] Trevor Hastie, Robert Tibshirani, and Jerome Friedman. *The Elements of Statistical Learning*. Springer Series in Statistics. New York, NY, USA: Springer New York Inc., 2001.

[12] Dalphne Koller and Nir Friedman. *Probabilistic Graphical Models*. 2009. ISBN: 978-0262013192.

[13] Judea Pearl. *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference*. Vol. 8. 1. 1992, pp. 73–75. ISBN: 978-1558604797. DOI: 10.1016/0167-9236(92)90038-Q.

[14] Michael P. Wellman and Max Henrion. "Explaining "Explaining Away"". In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 15.3 (1993), pp. 287–292. ISSN: 01628828. DOI: 10.1109/34.204911.

[15] Judea Pearl and Tom. S Verma. "A Theory of Inferred Causation". In: *Principles of Knowledge Representation and Reasoning* (1991).

[16] Tom. S Verma and Judea Pearl. "On the Equivalence of Causal Models". In: (1990). arXiv: 1304.1108.

[17] Lawrence K. Saul, Tommi Jaakkola, and Michael I. Jordan. "Mean field theory for sigmoid belief networks". In: *Journal of Artificial Intelligence Research* 4 (1996), pp. 61–76. ISSN: 10769757. DOI: 10.1613/jair.251.

[18] James Henderson and Ivan Titov. "Incremental sigmoid belief networks for grammar learning". In: *Journal of Machine Learning Research* 11 (2010), pp. 3541–3570. ISSN: 15324435.

[19] Yang Zhou. "Structure Learning of Probabilistic Graphical Models: A Comprehensive Survey". In: (2011). arXiv: 1111.6925. URL: http://arxiv.org/abs/1111.6925.

[20] Kevin Murphy. *Machine Learning: A Probabilistic Perspective*. 2012. ISBN: 9780262018029.

[21] Peter Spirtes and Clark Glymour. *Causation, Prediction, and Search*. ISBN: 0262194406.

[22] Diego Colombo and Marloes H Maathuis. "Order-independent constraint-based causal structure learning". In: 15 (2012), pp. 3921–3962. arXiv: arXiv:1211.3295v2. URL: https://arxiv.org/pdf/1211.3295.pdf.

[23] Dimitris Margaritis. "Learning Bayesian Network Model Structure from Data". PhD thesis. 2003.

[24] Ioannis Tsamardinos, Constantin F. Aliferis, and Alexander Statnikov. "Time and sample efficient discovery of Markov blankets and direct causal relations". In: *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining* (2003), pp. 673–678. DOI: 10.1145/956750.956838.

[25] I Tsamardinos et al. "Algorithms for Large Scale Markov Blanket Discovery". In: *FLAIRS Conference* i (2003), pp. 376–381. DOI: 10.1609/aimag.v12i4.918. arXiv: 1855670437. URL: http://www.aaai.org/Papers/FLAIRS/2003/Flairs03-073.pdf.

[26] Sandeep Yaramakala and Dimitris Margaritis. "Speculative Markov blanket discovery for optimal feature selection". In: *Proceedings - IEEE International Conference on Data Mining, ICDM* Icdm (2005), pp. 809–812. ISSN: 15504786. DOI: 10.1109/ICDM.2005.134.

[27] Jose M. Peña. "Learning Gaussian graphical models of gene networks with false discovery rate control". In: *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)* 4973 LNCS (2008), pp. 165–176. ISSN: 03029743. DOI: 10.1007/978-3-540-78757-0_15.

[28] Yosef Hochberg. "Controlling the False Discovery Rate : A Practical and Powerful Approach to Multiple Testing". In: *Journal of the Royal Statistical Society . Series B (Methodological), Vol . 57 , No . 1 (1995)* 57.1 (1995), pp. 289–300.

[29] Marco Scutari and Jean-Baptiste Denis. *Bayesian Networks with Examples*. Chapman & Hall, CRC Press, 2015. ISBN: 9781482225594.

[30] Pedro Larranaga et al. "Structure learning of bayesian networks by genetic algorithms: A performance analysis of control parameters". In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 18.9 (1996), pp. 912–926. ISSN: 01628828. DOI: 10.1109/34.537345.

[31] Radhakrishnan Nagarajan and Marco Scutari. *Bayesian Networks in R*. ISBN: 9781461464457.

[32] Shengyu Zhu, Ignavier Ng, and Zhitang Chen. "Causal Discovery with Reinforcement Learning". In: (2019), pp. 1–17. arXiv: 1906.04477. URL: http://arxiv.org/abs/1906.04477.

[33] Thomas M. Cover and Joy A. Thomas. *Elements of Information Theory*. 2005, pp. 1–748. ISBN: 9780471241959. DOI: 10.1002/047174882X.

[34] Zoubin Ghahramani and Iain Murray. "A note on the evidence and Bayesian Occam's razor". In: *Gatsby Unit Technical Report* August (2005). DOI: http://eprints.pascal-network.org/archive/00001165/01/occam.pdf. URL: http://www.gatsby.ucl.ac.uk/.

[35] Jack Kuipers, Giusi Moffa, and David Heckerman. "Deriving and Simplifying the BGe Score". In: *Suppliment to "Addendum on the Scoring of Gaussian Directed Acyclic Graphical Models"* (2010), pp. 1–9.

[36] Jack Kuipers, Giusi Moffa, and David Heckerman. "Addendum on the scoring of Gaussian directed acyclic graphical models". In: *The Annals of Statistics* 42.4 (2014), pp. 1689–1691. ISSN: 0090-5364. DOI: 10.1214/14-aos1217.

[37] Maxime Gasse, Alex Aussem, and Haytham Elghazel. "A hybrid algorithm for Bayesian network structure learning with application to multi-label learning". In: *Expert Systems with Applications* 41.15 (2014), pp. 6755–6772. ISSN: 09574174. DOI: 10.1016/j.eswa.2014.04.032. arXiv: 1506.05692. URL: http://dx.doi.org/10.1016/j.eswa.2014.04.032.

[38] Xun Zheng et al. "Dags with no tears: Continuous optimization for structure learning". In: *Advances in Neural Information Processing Systems* 2018-Decem.1 (2018), pp. 9472–9483. ISSN: 10495258.

[39] Juliane Schafer and Korbinian Strimmer. "Statistical Applications in Genetics and Molecular Biology A Shrinkage Approach to Large-Scale Covariance Matrix Estimation and Implications for Functional Genomics A Shrinkage Approach to Large-Scale Covariance Matrix Estimation and Implications for Func". In: 4.1 (2005).

[40] Marco Scutari. "Bayesian network constraint-based structure learning algorithms: Parallel and optimized implementations in the bnlearn R package". In: *Journal of Statistical Software* 77.1 (2017). ISSN: 15487660. DOI: 10.18637/jss.v077.i02. arXiv: 1406.7648.

[41] Markus Kalisch and Peter Bühlmann. "Estimating high-dimensional directed acyclic graphs with the PC-algorithm". In: *Journal of Machine Learning Research* 8 (2007), pp. 613–636. ISSN: 15324435. arXiv: 0510436 [math].

[42] Pierre Baldi et al. "Assessing the accuracy of prediction algorithms for classification: An overview". In: *Bioinformatics* 16.5 (2000), pp. 412–424. ISSN: 13674803. DOI: 10.1093/bioinformatics/16.5.412.

[43] Marco Scutari and Radhakrishnan Nagarajan. "Artificial Intelligence in Medicine Identifying significant edges in graphical models of molecular networks". In: *Artificial Intelligence In Medicine* 57.3 (2013), pp. 207–217. ISSN: 0933-3657. DOI: 10.1016/j.artmed.2012.12.006. URL: http://dx.doi.org/10.1016/j.artmed.2012.12.006.

[44] Marco Scutari. "Learning Bayesian networks with the bnlearn R Package". In: *Journal of Statistical Software* 35.3 (2010), pp. 1–22. ISSN: 15487660. DOI: 10.18637/jss.v035.i03. arXiv: 0908.3817.

[45] Marco Scutari et al. "Multiple Quantitative Trait Analysis Using Bayesian Networks". In: *Genetics* 198.1 (2014), pp. 129–137. ISSN: 19432631. DOI: 10.1534/genetics.114.165704. arXiv: 1402.2905.

[46] Schizophrenia Working Group of the Psychiatric Genomics Consortium. "Biological Insights From 108 Schizophrenia-Associated Genetic Loci". In: *Nature* 511.7510 (2014), pp. 421–427. DOI: 10.1038/nature13595.Biological.

[47] Arthur Gretton et al. "Measuring statistical dependence with Hilbert-Schmidt norms". In: *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)* 3734 LNAI (2005), pp. 63–77. ISSN: 03029743. DOI: 10.1007/11564089_7.

Appendix A

Code

The code written for this thesis was predominantly in R, with a smattering of Python. The following libraries and packages were used:

- **R**:
  - *bnlearn*: for Bayesian Network framework and structure learning algorithms.
  - *reshape2, purrr, plyr*: for dataframe manipulation.
  - *Rgraphviz, ggplot2*: for data and network visualisation.
  - *reticulate*: for interfacing with Python code
- **Python**:
  - *numpy*: for manipulating numerical data structures
  - *NOTEARS*: A working implementation of the NOTEARS algorithm from its authors. The code can be found on https://github.com/xunzheng/notears

The Cytoscape software was used to plot the Schizophrenia networks plotted in Chapter 6.

The codebase written for this project is made publicly available:
https://github.com/12kleingordon34/mastersproject2020.