Radboud University Nijmegen

Master's Thesis Mathematics

Causal Discovery Algorithms and Real World Systems

Author:
Vincent Coumans

Supervisor:
Dr Tom Claassen
Second reader:
Dr Sebastiaan Terwijn

August 20, 2017

Preface

This thesis serves as a proof of competence for obtaining a master's degree in mathematics from the Radboud University Nijmegen. In a way, this thesis is a perfect conclusion of the pathway I took through the study Mathematics. Not only does the subject matter of this thesis involve various branches of mathematics, it also was a rewarding answer to my increasing desire to witness applications of mathematics. Hopefully reading this thesis is enjoyable and useful, writing it definitely was.

I am very grateful to Tom Claassen for his supervision and his inspiring remarks. Moreover, I appreciate his style of feedback very much as he held no punches, as requested. Furthermore, I want to thank Tom Heskes for inviting me to discuss subjects for a master's thesis and for introducing me to Tom Claassen.
I want to thank Serge Horbach and Alex Kolmus for their feedback and many interesting discussions. Finally, a word of thanks to Ina de Vries for being such a great study counsellor.

Table of Contents

1 Introduction

One of the main endeavours of many sciences is to identify causal relations. Given a phenomenon, researchers are interested in finding out which variables have causal influence on that phenomenon. Furthermore, scientists want to know how that phenomenon will change when one manipulates those variables. For instance, when a new drug is tested, it is examined whether it has a causal effect on improving the health of the patients. In these circumstances one uses randomized trials to assess the causal interaction. In this case that would amount to one group of test subjects receiving the drug and the other group receiving a placebo. Ideally, causal discovery is performed via these randomized trials. However, sometimes one cannot perform these type of experiments and one only has observational data, i.e. data obtained without any interventions. An example of this is research regarding gene activation levels in normally operating cells. Here we have data of the expression levels of various genes and we are interested in finding out which genes influence the expression of other genes.
Observational data is where causal discovery algorithms come into play. Causal discovery algorithms are algorithms that try to derive causal relations from observational data. That is, given a set of data, a causal discovery algorithm returns a set of statements regarding the causal interactions between the measured variables. Often, such a set of statements is represented with a graph. There are many causal inference algorithms which provably find the correct causal structure under certain ideal circumstances. Should these algorithms prove to be effective in practice then they are a very powerful instrument and then they would probably be among the standard tools in scientific research. As of yet, however, they are not. Apparently these algorithms do not function in practice as desired. Ergo, on the one hand we have the theoretical setting in which causal discovery algorithms fare well and on the other hand we have real world systems for which they apparently are not yet suited. Hence, bridging this gap between the theoretical correctness results and application in practice would be invaluable. This thesis tries to take some steps in bridging the gap by exploring it. That is, we will evaluate causal discovery algorithms on realistically simulated data. In doing so we hope to characterize mistakes causal discovery algorithms make in non ideal circumstances. Furthermore, we aim to be able to find ways of recognizing mistakes and ideally we find ways of avoiding said mistakes. Completely bridging the gap, however, is too great a task for a master's thesis. In this master's thesis we do hope to find topics for future research that seem fruitful for the endeavour of bridging the gap. This is done by considering the performance of causal discovery algorithms on several datasets. Starting with datasets consistent with the theoretical optimality results and moving onto increasingly more realistic datasets.
A typical problem of evaluating causal discovery algorithms on realistic datasets is *verifiability*. Assessing the correctness of the outcomes of the algorithms requires knowledge of the true causal interactions. Hence, we consider realistically *simulated* datasets for which some form of *ground truth* is available.
In the remainder of this introduction I briefly discuss some general aspects of causal discovery and various causal discovery algorithms, leaving the details for sections 2 and 3. Then I review (some deviations of the) ideal circumstances (for which we have theoretical correctness results). Lastly I give an outline of the thesis.

1.1 Causal Discovery Algorithms

We begin this subsection by considering several aspects and complications of causal discovery algorithms. When, for instance, we would investigate causal relations concerned with global warming, then we first choose the set V of variables to be considered. This set of variables would, for instance, include carbon dioxide levels, methane levels, etc. A desired property of this set of variables is that it is *causally sufficient*. This means that for every pair of variables $X, Y \in V$, if they have a *common cause*, i.e a variable that causes both X and Y, then that common cause is also in V. In the absence of causal sufficiency we might be inclined to infer a causal relation between variables X and Y, while they are not causally related but only have a common cause. For instance, consider the informal example of 'ice cream consumption' and 'drowning rates'. The following trend can be detected: when ice cream consumption increases, so do the drowning rates. Assuming that all variables required for a description of the causal relations are present, we would mistakenly conclude that either ice cream consumption causes drowning or vice versa. However, we

can also include the variable 'summer' in the description of the causal system. If it is summer and, hence, the average temperature is higher, people consume more ice cream. Furthermore, in the summer people swim more often at the beach and, hence, drowning rates also increase. In this case we see that ice cream consumption and drowning rates are not causally related but that they only share a common cause: warm weather.
Another property that might complicate causal discovery is *selection bias*, i.e. that the population is not random but that it is conditioned on a variable not in V. For instance, assume a drug with heavy side effects is tested on a group of test subjects. Then it might be that the health of some of these subjects is too poor in the sense that the side effects of the drugs would be dangerous if the treatment is continued. Hence, the tests on those subjects are stopped and the resulting dataset comprises only data from subjects who are healthy enough to withstand the side effects of the drug. This affects the assessment of the effect of the drug.
Some causal discovery algorithms can handle causal insufficiency and selection bias, others cannot.

Furthermore, one of the main problems for causal discovery algorithms is that one only has finite data. Hence, the observed distribution of the variables, on which many of these causal interaction statements are based, might significantly differ from the actual distribution. This may lead to errors in the causal discovery. Ideally we have a causal discovery algorithm that is *sound* and *complete*. A causal discovery algorithm is sound if the causal statements it returns are valid, it is complete if it in fact returns *all* valid causal statements. For many causal discovery algorithms there are theoretical results stating that that specific algorithm is sound and complete under certain assumptions. We will discuss these assumptions in more detail later.
There are two major classes of causal discovery algorithms, [27, p.1]. These classes are the so-called constraint-based algorithms and the so-called score-based algorithms. In this thesis we will evaluate one typical constraint-based algorithm and one typical score-based algorithm, hence representing most of the causal discovery algorithms.

1.1.1 Constraint-based algorithms

As the name suggests, constraint-based algorithms first identify several constraints which the underlying causal structure should satisfy. These constraints are then used to derive the causal structure. The forementioned constraints might, for instance, consist of *conditional independence* statements. Conditional independence statements are of the form X is independent of Y given W, where X, Y are stochasts and W is a set of stochasts. This is precisely the case if $p(X, Y|W) = p(X|W)p(Y|W)$ and we denote this by $X \perp\!\!\!\perp Y|W$.
The advantage of constraint-based algorithms is that they are generally applicable. However, a very large sample size may be needed to get the correct result, [26, p.11].
Examples of constraint-based algorithms are the PC[1] algorithm, [25], the modified PC algorithm, [25, 28] and the FCI[2] algorithm, [25, 28].
Although these algorithms are similar in some ways, there are differences between the algorithms which make some more suited for certain situations than others. First of all, PC can be shown to be sound and complete under certain circumstances. Furthermore, it is also rather efficient, [25]. However, it cannot handle causal insufficiency and selection bias. On the other hand, FCI can handle causal sufficiency and selection bias. Furthermore, it can be proven that under ideal circumstances it is sound and complete, [28]. However, it is not efficient enough to handle datasets with many variables. Modified PC is efficient and it can handle causal insufficiency and selection bias, however, it is not sound and complete, [25]. Luckily, the amount of mistakes modified PC makes in comparison to FCI is small, [7, section 3].
As the assumption of causal sufficiency proves to be inaccurate in many cases, [25], we choose to evaluate a constraint-based algorithm that can handle hidden common causes. Specifically, we choose modified PC as it is efficient enough to handle large sets of variables, albeit at the cost of not being provably sound and complete.

[1]The name is derived from the first letters of the first names of the inventors, Peter Spirtes and Clark Glymour.
[2]Here FCI stands for Fast Causal Inference.

1.1.2 Score-based algorithms

Score-based algorithms, as the name suggests, score possible explanations. They reward explanatory power but they penalize complexity. The concrete algorithm we will evaluate is Greedy Equivalence Search (GES). This algorithm searches a graph that can represents the correct causal statements. It does so by first adding edges to increase the explanatory power and then removing edges to reduce the complexity. The explanation with the highest score is found with a search procedure. The search procedure consists of states and each edge addition/removal moves GES from one state to the next.
Regarding the complexity of this search procedure, GES visits at most $n(n-1)$ different states, where n is the number of variables. Hence, the algorithmic complexity of the search is mainly dependent on the amount of states one can reach from a certain state. Unfortunately, learning the optimal structure using the so-called Bayesian scoring criterion, cf. section 3.2.1, is NP-hard, as Chickering proved, [4]. However, the worst case scenarios do not occur often in practice and steps can be taken to make the search as efficient as possible, [5].
One of the advantages of score-based algorithms is that we are not only dependent on the conditional independence statements, but that the score considers the explanation as a whole. Furthermore, there are also score-based methods that return multiple high scoring explanations instead of only the highest scoring one. Greedy equivalence search, however, is designed only to return the highest scoring explanation.

1.1.3 Variations and other causal discovery algorithms

In this subsection we summarize other causal discovery algorithms. This gives an idea of the variety of causal discovery algorithms.
First of all, there are several variations of the algorithms mentioned in the above. For instance, for some constraint-based algorithms there are variations that can work with background knowledge, [16]. This background knowledge is in the form of statements as 'variable X causes Y' or 'variable X is not a cause of Y' and is used to represent causal interactions known to be valid, prior to the causal inference. Ways of determining (whether there exists) a model explaining both the data and the background knowledge are investigated in [16]. These methods consist of two steps, first a causal model is determined with a constraint-based method, then the background information is brought into the equation. In this thesis we focus on the first step and hence we do not consider methods for including background knowledge.
Furthermore, there are several modifications for FCI. For instance, a *conservative* version of FCI exists in which additional conditional independency tests are performed to prevent mistakes, [20]. The *majority rule* variant of FCI, [8], is similar to the conservative version as it also performs additional tests. However, the interpretation of these tests is less strict than conservative FCI. There are variants of FCI in which the order of the conditional independency test is relevant and variants which are order independent, [8].
For GES there are variations that improve the efficiency of the search procedure at the cost of making assumptions regarding the structure of the causal system, for instance that every node has at most k parents, for a certain number k, [6]. For using causal discovery algorithms in practice we aim not to make said assumptions. Therefore, we do not use this variation of GES.
Apart from variations of the algorithms described in the above, there are also algorithms that are methods for specific problems. For instance, there is an algorithm specifically designed for distinguishing cause from effect. Given two variables, X and Y, suppose one only knows that they are causally related, then it is non trivial to differentiate between cause and effect. In [29] a method is proposed to identify cause and effect in such a situation. They make certain assumptions that if X is cause of Y then Y can be expressed in X in a certain fashion. In order to check this, X and Y are transformed to stochasts X' and Y' such that X is a cause of Y if X' and Y' are independent. Furthermore, in [29], it is shown that for specific circumstances this approach can indeed separate cause from effect.
Although both GES and modified PC cannot output cyclic causal structures, there are methods that can. For instance, Cyclic Causal Determination, [21], is an algorithm designed to handle causal systems that have cyclic components. This algorithm is in essence a constraint-based algorithm with additional rules to interpret these constraints so cycles can occur. Although provably sound and complete, there are a number

of limitations to this algorithm, [14], making the algorithm intractable in certain circumstances.
Lastly, GES and modified PC thrive when applied to Gaussian data. However, for non-Gaussian data a method using LiNGAM models can be used, [23]. In these models, variables are expressed as a linear combination of other variables and non-Gaussian noise components. Methods exist to retrieve these functional dependencies from the dataset and they are provably sound and complete, [23].

1.2 The ideal circumstance and deviations from this ideal setting

As mentioned, some of the algorithms can be proven to be sound and complete under certain circumstances. Properties of these circumstances consist of two categories, statistical assumptions and the assumptions regarding causality.
The statistical assumptions are the assumptions that regard the statistical difficulties. For instance, with causal inference we only have a finite amount of data and we score models or conduct tests for conditional independence based on this finite data. In the theoretical optimality results it is, however, assumed that we have an oracle for the conditional independence tests. Furthermore, the optimality results regarding the score-based algorithms state that "in the limit of large sample size" the output is correct.
Second of all, both the score and the conditional independence test used in this thesis assume a certain distribution of the variables. In practice, these variables need not have that distribution. Therefore, in practice, the output of the algorithms may not be correct.
Regarding the causality assumption: each causal discovery algorithms has a certain interpretation of causality. For instance, the ones considered in the remainder of this thesis assume that, at the core, the causal system can be represented by a directed acyclic graph (DAG). This implies that a causal system cannot contain feedback, i.e. two variables having causal influence on each other, or self-loops, i.e. a variable having a causal influence on itself. In practice, however, there are causal systems which contain feedback or self-loops. We shall consider the forementioned optimality results in more detail in section 3.

1.3 Outline of the thesis

In section 2 we evaluate the notion of causality. There we discuss the task of defining causality. Then we consider the entire process of causal inference from observational data, starting with causal systems underlying variables to the output of the algorithms. The precise descriptions of modified PC and GES are considered in section 3.
In section 3 we review the algorithms, which we evaluate later, in detail. This section also mentions several theoretical results regarding the validity of these algorithms, i.e. the forementioned optimality results.
In section 4 we start the process of evaluating causal discovery algorithms to datasets. As mentioned, we begin by evaluating modified PC and GES in circumstances which are as ideal as possible, given the optimality results. These circumstances are referred to as *standard assumptions* and their datasets are referred to as *standard data*. Furthermore, we test the impact of various parameters, such as sample size and the number of variables, on the performance of these algorithms. These results give us a baseline with which we can compare the performance of modified PC and GES on non-standard datasets. Furthermore, as it turns out, even in these circumstances the algorithms usually do not give perfect results.
In section 5 we take a small step away from the standard data and test the performance of the algorithms on simulated functional magnetic resonance imaging (fMRI) datasets. Functional magnetic resonance imaging is a technique used in research of the structure of the brain. Hence, being able to successfully apply causal discovery algorithms to this type of data is valuable. In addition, the simulated dataset is still on many, but not all, levels conform the standard assumptions. Hence, we can see the performance of modified PC and GES on datasets that have similar properties as the standard data.
In section 6 we turn to flux balance analysis (FBA) and apply the algorithms to the so-called fluxes of interconnected chemical reactions. Flux balance analysis is a tool used to simulate systems of chemical reactions and to predict behavior of this system under certain interventions. Informally speaking, flux is the speed of a reaction. Knowing the flux of every reaction in the system is then informative for the structure underlying the system of reactions. This dataset is of interest for bridging the forementioned gap since both

the distribution of the variables and the underlying causal structure are non-standard.
In section 7 we consider gene regulatory networks. These are networks used to simulate the expression levels of genes and influences between these expression levels. These networks can be used to predict the behavior of the expression levels under interventions, for instance the administration of a drug. We use causal discovery algorithms to so-called *reverse-engineer* the structure of the gene regulatory network from the data of the expression levels. This dataset is of interest for our main goal since the distribution of the variables is closer to the distribution of standard data, even though the underlying structure contains both cycles (i.e. feedback) and self-loops.
In section 8 we conclude the thesis with interpreting the results from sections 4 - 7 and we discuss topics for future research.

2 Causality

When one investigates causal discovery algorithms one would, at first sight, need a precise definition of causality, i.e. when do we say that an event A causes an event B. This definition is needed in order to prove whether or not the outcome of such an algorithm does indeed return proper causal relations. Unfortunately there still is not a widely accepted definition of causality and there are various ways of considering causality. For these various considerations, the interested reader is referred to [19].
Given the significant amount of literature on the subject of defining causality, we will not consider a precise definition of causality in this thesis and use our intuitive understanding of causality instead. Devices have been set in place to still have a meaningful discussion of causality, irregardless of the precise definition of causality. Hence, such a precise definition is no longer needed in order to evaluate causal discovery algorithms. Furthermore, using an intuitive understanding of causality does no harm if we do not consider borderline cases of causality. A satisfying definition of causality should be compatible with our intuitive understanding of causality and should give clarification in the case of doubt. Hence, should we avoid cases of doubt, the use of our intuitive understanding of causality suffices.
This section is outlined as follows. First we consider various assumptions regarding the nature of causality and show that although these assumptions are reasonable, there are cases in which these assumptions are violated. Hence, when one wants to evaluate how well the causal discovery algorithms work in practice, the performance of the algorithms we consider on data that violates these assumptions becomes a point of interest. Especially when these algorithms are based on the violated assumptions. Secondly, although leaving the description of modified PC and GES to section 3, we consider the theoretical background of every step in the process from causal structures underlying the data to the outcome of the causal discovery algorithms.

2.1 The nature of causality

The causal discovery algorithm we will evaluate assume that causality is irreflexive, transitive and antisymmetric. This means that an event A cannot cause itself (irreflexivity), if A is a cause of B and B a cause of C, then A is a cause of C (transitivity) and if A is a cause of B, then B is not a cause of A. There are, however, situations imaginable in which these properties are not valid for causality.
Consider the following counterexample from [9]. Assume a person is in a room with a heater. Now assume this person turns the heater up. Then first the external temperature increases and then his core temperature. This increase of core temperature activates the homeostatic system which decreases the core temperature to normal. If transitivity holds, then in this chain of events, turning up the heater would cause the core temperature to decrease back to normal. This is a counter intuitive statement. Furthermore, consider the following example. Let Aaron be a student, let Fail be the number of exams failed by Aaron and let Conf be the level of confidence of Aaron. Then an increase of Fail would cause the confidence of Aaron to drop and that drop of confidence might increase the number of exams Aaron fails. In this system one would say that Fail causes Conf and that Conf causes Fail. Furthermore, should we assume transitivity, than Fail would be a cause of itself.
These counterexamples indicate that the nature of causality is not always captured by these assumptions. Hence, it is of interest to evaluate how causal discovery algorithms based on these assumptions handle causal systems that violate said assumptions. These violations include, among others, the presence of feedback, i.e. when A causes B and B in turn causes A, and self-loops, i.e. when an event has a causal effect on itself.

2.2 (Representing) causal structures

For describing the entire process from the causal structures to the outcome of the algorithms we start by describing the representation of causal structures.

2.2.1 DAGs and CPDAGs

Let V be a finite set of variables. Then one is interested in finding causal relations between these variables. Assuming causality is transitive, irreflexive and anti-symmetric the causal system can be, as we will see, represented by a directed acyclic graph (DAG). Now, given such a set of variables V and a variable $X \in V$, we can consider the so-called direct causes of X.

Definition 2.2.1 (Direct cause)
*Given a set of variables V and let $X, Y \in V$. Then X is a **direct cause** of Y if there is a subset C of $V \setminus \{Y\}$ such that (i) C contains X, C consists solely of causes of Y, C completely determines Y and (ii) there is no $S \subsetneq C$ that satisfies (i).*

For a set of variables V we can consider the causal system (V, E) where V is the set of variables and

$$E = \{(X, Y) \in V \times V | X \text{ is a direct cause of } Y\}.$$

These causal systems (V, E) can be depicted in a directed graph on vertex set V where a directed arrow is placed from X to Y iff $(X, Y) \in E$. Assuming causality is transitive, irreflexive and anti-symmetric, this graph is a DAG. A DAG representing a causal system is called a **causal DAG**.

Example 2.2.2 (Causal structure and causal DAG)
Consider the causal structure defined by

$$V = \{\textit{Summer},\ \textit{Swimming},\ \textit{Ice cream consumption}, \textit{Drowning}\}$$

and

$$E = \{(\textit{Summer}, \textit{Swimming}), (\textit{Summer}, \textit{Ice cream consumption}), (\textit{Swimming}, \textit{Drowning})\}.$$

This causal system is represented by the DAG in figure 1.

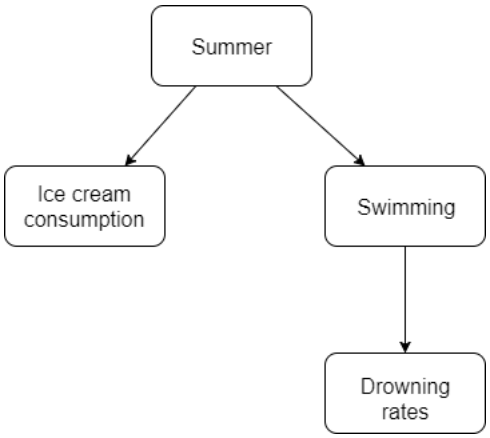


Figure 1: Example of a causal DAG regarding ice cream and drowning.
The intuition is that when it is summer, people consume more ice cream and swim more often. Furthermore, when people swim more often, the drowning rate increases.

Given a DAG representing a causal system, we want to link the DAG (or, equivalently, the causal system it represents) to the joint probability distribution of the variables of that causal system.
The Causal Markov assumption and the Faithfulness condition relate DAGs and the causal system they represent to probability distributions. First, we consider an example introducing the Causal Markov assumption, then we will make both conditions precise.

Example 2.2.3 (Causal Markov Assumption)
Consider the causal system from 2.2.2. Given that more people will swim, then it is irrelevant whether or not it is summer; the drowning rates will increase. We conclude that $(Summer \perp\!\!\!\perp Drowning)|Swimming$. *The Causal Markov Assumption gives us a way to read conditional independence statements from DAGs in such a fashion. The faithfulness condition guarantees that the conditional independence statements that can be read from the DAG are indeed the only ones that are valid for the probability distribution.*

- A DAG G and a probability distribution p satisfy the **Causal Markov Assumption** iff each variable is conditionally independent of its non-descendants given its parents. In this case we say that p is generated by G.
- A DAG G and a probability distribution p that satisfy the Causal Markov Assumption satisfy the **Faithfulness Assumption** iff each conditional independence statement valid for p is implied by the Causal Markov Assumption. In this case we say that G and p are faithful to one another.

Given a DAG G that is faithful to a probability distribution p, one can read conditional independence statements from the DAG via d-separation. The definition of d-separation goes beyond the scope of this text. Consider [25] for a detailed description of d-separation.
Now we can ask ourselves whether causal faithfulness and the Causal Markov assumption uniquely determine a DAG, i.e. whether for each probability distribution p there is at most one DAG G such that G and p are faithful to one another. The answer to this question is a negative. Given a probability distribution p, there is, in fact, an entire class of DAGs that are faithful to p, the so-called Markov Equivalency Class.

Definition 2.2.4 (Markov Equivalence)
Given DAGs G *and* H, *then* G *and* H *are said to be Markov equivalent iff for every probability distribution* p *we have that* p *and* G *are faithful to one another if and only if* p *and* H *are faithful to one another.*

Markov equivalent DAGs can also be characterized graphically. Among others, this notion is based on so-called v-structures.

Definition 2.2.5 (v-structures)
Let G *be a DAG, then a* v*-structure is a triple of vertices* X, Y, Z *such that the subgraph generated by* X, Y *and* Z *in* G *is* $X \to Y \leftarrow Z$.

Lemma 2.2.6
Let G *and* H *be DAGs, then* G *and* H *are Markov equivalent iff they have the same adjacencies and the same* v*-structures.*

A Markov equivalence class can be uniquely characterized by a so-called **CPDAG**, or **pattern**. The notion of a CPDAG relies on the notion of skeleton.

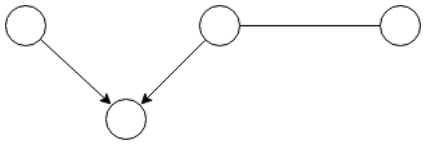
Definition 2.2.7 (Skeleton)
Let G *be a graph on nodes* V. *Then the* ***skeleton*** *of* G *is the undirected graph on* V *such that for every pair of nodes* $X, Y \in V$ *one has that* X *and* Y *are adjacent in* G *iff there is an edge between* X *and* Y *in the skeleton of* G.

Note that every pair of Markov equivalent DAGs has the same skeleton, as is implied by the previous lemma. This provides justification for the following definition.

Definition 2.2.8 (CPDAG/Pattern)
Let M *be a Markov Equivalence class of DAGs. Then the pattern representing* M *is a graph whose skeleton is the same as the skeleton of all the DAGs in that Markov equivalence class and such that for every pair of adjacent nodes* X, Y *we have that that edge between* X *and* Y *is a directed edge* $X \to Y$ *iff for every DAG in the equivalence class that edge is oriented as* $X \to Y$, *otherwise it is an undirected edge.*

Example 2.2.9
The following DAGs are Markov equivalent.

The CPDAG representing their equivalence class is:



2.2.2 MAGs and PAGs

Given a causal structure, then the observational data can be incomplete in the sense that certain variables from the causal structure are not observed. One can write the set of variables V of such a structure as the disjoint union

$$V = O \sqcup S \sqcup L.$$

Here O is the set of observed variables, S is the set of unobserved variables upon which is conditioned (the selection variables) and L is the set of remaining unobserved variables. Should we assume causal sufficiency, when, in fact, there are hidden common causes, this might lead to erroneous results.

Example 2.2.10 (Hidden common cause)
In example 2.2.2 we have a causal system. Now assume variable "Summer" is unobserved. Then one would still find a link between swimming and ice cream consumption. However, when we assume causal sufficiency, then swimming must be a cause of the increase of ice cream consumption or vice versa. In this concrete example this is not the case.

There are several possibilities to express the presence of hidden common causes in graphs. However, the algorithms we evaluate and that can handle hidden common causes, output a so-called PAG. Hence, we will work towards that definition, starting with the definition of a mixed graph.

Definition 2.2.11 (Mixed graph)
*A **mixed graph** is a graph that can contain three types of edges: directed arrows, bidirected arrows and undirected arrows. Let X, Y be two nodes from a mixed graph then there is an **almost directed cycle** from X to Y if X and Y are connected by a bidirected arrow and if there is a directed path from X to Y. Nodes that are connected by an undirected edge are called **neighbors**, nodes connected by a bidirected arrow are called **spouses**.*

There is a certain analogy between the representation of causally sufficient structures and the representations of, perhaps, causally insufficient structures. In this analogy, mixed graphs correspond to directed graphs. The DAGs correspond to so-called maximal ancestral graphs (MAG), which, as the name suggests, is a type of ancestral graph.

Definition 2.2.12 (Ancestral graph)
*A mixed graph is called **ancestral** if it contains no directed cycles and no almost directed cycles and if for any undirected edge $X - Y$, X and Y have no parents or spouses.*

As one can read off conditional independence statements from DAGs via d-separation, one can read conditional independence statements from ancestral graphs using so-called m-separation which is a generalization of d-separation to ancestral graphs. A detailed treatment of m-separation is beyond the scope of this text, for such a treatment consider [28]. Nonetheless, this notion is used in the definition of maximal ancestral graph.

Definition 2.2.13 (Maximal Ancestral Graph)
An ancestral graph is a ***maximal ancestral graph*** *iff for every pair of nodes* (X, Y) *we have that* X *and* Y *are not adjacent iff there is a set of nodes* S *such that* X *and* Y *are* m*-separated given* S

As the conditional independence constraints (or equivalent, d-separation statements) do not uniquely determine a DAG, so do m-separation statements not uniquely determine a MAG. One can generalize the notion of Markov Equivalence to the situations of MAGs and then one can represent the equivalence class of a MAG with a so-called **partial ancestral graph**, or PAG for short. Given such an equivalence class, then an edge mark is **invariant** if it is the same in every member of the equivalence class. It can be shown that equivalent MAGs have the same adjacencies. This fact provides justification for the following definition.

Definition 2.2.14 (Partial Ancestral Graphs)
Given a Markov Equivalence class of a MAG M*, then the PAG* P *representing that class is a graph with the same adjacencies as* M *but with three different edge marks: arrowheads, tails and circles. Furthermore, it must hold that every non-circle mark in* P *is an invariant mark. If, in addition, every circle mark represents an variant mark, then* P *is called a maximally informative PAG for the equivalence class of* M*.*

Figure 2 shows an example of a PAG.

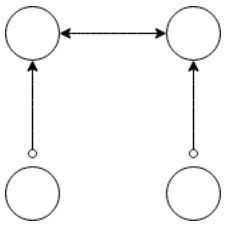


Figure 2: An example of a PAG.

Let P be a PAG and let X, Y be nodes of P, then one should interpret an edge

$$X* \rightarrow Y$$ [3]

as Y is not a cause X or of any selection variable. Hence, a bidirected edge, for instance the top edge from figure 2, indicates the presence of a hidden common cause. An edge

$$X - *Y$$

should be interpreted as X causes Y or some selection variable.

[3]The asterisk indicates that at that spot there can be any edge mark.

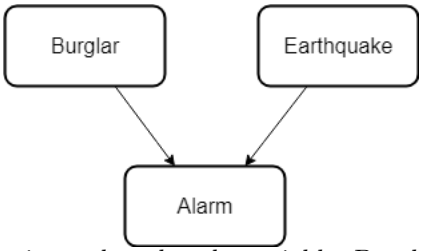
3 Causal Discovery Algorithms

As mentioned in the introduction, there are two main classes of causal discovery algorithms, namely constraint-based algorithms and score-based algorithms. From these classes we choose modified PC and GES to evaluate on realistically simulated data, using implementations of these algorithms from the R-package *pcalg*, [11]. Recall that the set-up is that we have a dataset on a set of variables V and that the algorithms estimate the causal structure. Their outputs are PAGs and CPDAGs, respectively. Both of these algorithms assume causality is transitive, irreflexive and anti-symmetric.
In this section we first describe modified PC and then we consider GES. Furthermore, their optimality results are stated (without proof).

3.1 Modified PC

The following is based on [25] and [28]. The algorithm modified PC consists of two phases: the skeleton phase and the orientation phase. In the skeleton phase conditional independence statements are derived from the data. This is done via a statistical test, we will review this in detail later on. The resulting skeleton satisfies the following property: for every pair of vertices X and Y, we have that X and Y are adjacent iff for every subset C of vertices not including X and Y we have X is not conditionally independent from Y given C.
The orientation phase starts with orienting the v-structures. For orienting the v-structures, the unshielded triples are considered, i.e. triples (X, Y, Z) such that X and Y are adjacent and Y and Z are adjacent, but X and Z are not. The device is that if X and Z are dependent given Y, then (X, Y, Z) should be oriented as a v-structure. This property is based on the so-called *explaining away principle*.

Example 3.1.1 (Explaining away)
Assume we have an alarm that goes off when there is burglar in the house or when there is an earthquake. Furthermore, assume that the occurrence of a burglar or an earthquake are independent. This can be represented by the following causal DAG.



However, knowing that the alarm is triggered, makes the variables Burglar and Earthquake dependent. That is, given that the alarm is triggered, then if you know that there has been an earthquake, then the possibility that there is a burglar becomes less likely.

Therefore, for an unshielded triple (X, Y, Z), if X and Z are not rendered independent given a set that contains Y, then that triple is oriented as a v-structure.
The orientation phase is finished by applying ten orientation rules. Discussing these orientation rules goes beyond the scope of this text and a detailed description can be found in [28].
A description of modified PC in pseudocode can be found in figure 1.

input : A dataset D over variable set V consisting of n variables
output: A PAG on V

C← Complete graph over V where every edge is oriented as o-o;
for *k from 0 to n* **do**
  for X, Y *distinct adjacent nodes in* C*, such that the number of nodes adjacent to* X *is greater than* k **do**
    for *Every set* $W \subset V \setminus \{Y\}$ *of nodes adjacent to* X*, with* $\#(W)=k$ **do**
      if *The conditional independence test on* (X, Y, W) *returns TRUE* **then**
        Erase the edge between X and Y;
        Remember W;
      end
    end
  end
end
for *Unshielded triples* $X * - * Y * - * Z$ *in* C **do**
  if Y *is not in one of the sets encountered that renders* X *and* Z *independent* **then**
    Orient $X * - * Y * - * Z$ as $X * \rightarrow Y \leftarrow * Z$
  end
end
Apply further orientation rules until no more edges can be oriented.
return C

Algorithm 1: Modified PC.

Note that the output is a PAG and, hence, it can indicate hidden common causes and selection bias. Therefore, modified PC is suited for handling causal structures with hidden common causes and selection bias.

Let D be a dataset over a set of variables V. Then, when applying modified PC to this dataset, it is repeatedly tested whether distinct variables X and Y are independent given another set of variables not containing X and not containing Y. Under the assumption that the dataset consists of random draws from a multivariate Gaussian, we have that, [10], for two variables X and Y and a set W of variables, not including X and Y that

$$X \perp\!\!\!\perp Y|W \iff \rho_{XY.W} = 0.$$

Here $\rho_{XY.W}$ denotes the *partial correlation* of X and Y given W.

The conditional independence test used in this thesis is described in algorithm 2 and is based on [10].

input : A dataset D over variable set V consisting of N samples, distinct variables $X, Y \in V$, a set $W \subseteq V \setminus \{X, Y\}$ and the threshold α
output: TRUE or FALSE

CondIndependent← FALSE;
r_{X_j,X_k,X_W} ←sample partial correlation of X_j, X_k on the set $\{X_p : p \in W\}$;
$c = \frac{1}{\sqrt{N-\#(W)-3}} |\arctan(r_{X_j,X_k,X_W})|$;
if $c \leq$ *Norm*$^{-1}$*(1-*$\frac{\alpha}{2}$*)* **then**
  CondIndependent:=TRUE
end
return *CondIndependent*

Algorithm 2: Conditional Independence Test.

In this algorithm, Norm indicates the cumulative distribution function of the standard normal distribution. Note that for low thresholds, variables are more likely to be conditionally independent than for high thresholds.

This conditional independency test is based on the following theorem, [11].

Theorem 3.1.2
Let $(X_1^i, ..., X_n^i)_{i=1}^N$ be an iid sequence of samples of a multivariate Gaussian distribution. Let $j, k \in \{1, ..., n\}$ be distinct and let W be a subset of $\{1, ..., n\}$ not including j or k. Then, under the assumption $\rho_{X_j, X_k, \{X_p : p \in W\}} = 0$, we have that

$$\frac{1}{\sqrt{N - \#(W) - 3}} |\arctan(r_{X_j, X_k, X_W})|$$

is asymptotically distributed by a standard normal distribution. Here N is the sample size and r_{X_j, X_k, X_W} is the sample partial correlation of X_j, X_k on the set $\{X_p : p \in W\}$.

We conclude this subsection with one of the optimality results. As mentioned, modified PC is not sound and complete. However, the output of modified PC is in most cases the same as the outcome of the algorithm FCI and for FCI one has the following optimality result, [28].

Theorem 3.1.3 (FCI is sound and complete)
Given a perfect conditional independence oracle, the FCI algorithm returns the maximally informative PAG for the true causal MAG.

Note that since one does not have a perfect conditional independence oracle, but a test that assumes a multivariate Gaussian distribution, this result should be interpreted as: given a dataset that is generated by a multivariate Gaussian distribution, then in the limit of large sample size, FCI returns the maximally informative PAG for the true causal MAG.

3.2 Greedy Equivalence Search

Greedy equivalence search is a score-based algorithm. It scores possible explanations and returns the explanation with the highest outcome. The outcome of the algorithm is a CPDAG. As mentioned in the introduction, the score rewards explanatory power and penalizes complexity. There are two facets of importance for such an algorithm. On the one hand a score is needed such that the CPDAG with the highest score is indeed the true model. On the other hand, evaluating the score of every possible CPDAG is not feasible since the amount of CPDAGs is exponential in the number of variables (the amount of undirected graphs on n nodes is already 2^n). Hence, one has to search for the optimal CPDAG in an efficient fashion.

3.2.1 Scoring and the Bayesian Information Criterium

The score used in the implementation of GES used in this thesis is based on the so-called Bayesian Information Criterium, [3]. The concrete expression of the score as a function of data D and a DAG G is:

$$\mathrm{S}(D, G) = p(D|\hat{\theta}, G^h) - \frac{1}{2} M \log(N). \tag{1}$$

Here G^h is the hypothesis that the distribution is faithful to DAG G, $\hat{\theta}$ is the value of θ that maximizes the likelihood $p(D|\theta, G^h)$, M is the amount of parameters in θ and N is the number of samples in D. The first term of the score measures the explanatory power. The second part concerns the complexity of the model. As the model becomes more complex, the number of parameters increases as well. Hence, the score becomes lower as the complexity of the model increases. In the implementation of GES in the package *pcalg* used in this thesis, [11], the *penalty term* $\frac{1}{2}\log(N)$ can be altered. Note that in this description of the score, DAGs are scored and not CPDAGs. However, on equivalent DAGs, the score is equal. Hence, this score translates uniquely to a score for CPDAGs. It can be shown, [5], that in the limit of large sample size, the true CPDAG has the highest score. For more information regarding the score, consider [5, 11].

3.2.2 Search procedure

The search procedure of greedy equivalence search is based on two operations: Add and Erase. Intuitively speaking, GES adds edges as long as the score increases. When a local maximum has been attained, GES starts removing edges as long as the score increases. The operation Add is described in pseudocode in algorithm 3.

```
   input  : A dataset D and a CPDAG G
   output: A CPDAG H
1  Consider the set S of CPDAGs obtainable from G by adding one edge;
2  Let K be a CPDAG from S with the highest score S(D, K);
3  if S(D,G)<S(D,K) then
4  |  H←K
5  else
6  |  H←G
7  end
8  return H
```

Algorithm 3: Operation Add.

The operation Erase is described in pseudocode in algorithm 4.

```
   input  : A dataset D and a CPDAG G
   output: A CPDAG H
1  Consider the set S of CPDAGs obtainable from G by removing one edge;
2  Let K be a CPDAG from S with the highest score S(D, K);
3  if S(D,G)<S(D,K) then
4  |  H←K
5  else
6  |  H←G
7  end
8  return H
```

Algorithm 4: Operation Remove.

Greedy equivalence search starts with the empty graph and repeats operation Add until the score does not increase anymore. Then operation Remove is repeated until the score does no longer increment. The description of GES can be considered in pseudocode in algorithm 5.

```
   input  : A dataset D
   output: A CPDAG G
1  G← Empty graph;
2  while S(D,Add(D,G))>S(D,G) do
3  |  G←Add(D,G)
4  end
5  while S(D,Erase(D,G))>S(D,G) do
6  |  G←Erase(D,G)
7  end
8  return G
```

Algorithm 5: GES.

The search can be made more efficient by restricting the CPDAGs evaluated in the Add and the Erase operations. The specific improvements are beyond the scope of this text. Futhermore, note that the output of GES is a CPDAG. Hence, GES is not suited for datasets with hidden common causes and selection bias. We conclude this subsection by stating an optimality result for GES, [5].

Theorem 3.2.1 (Soundness and completeness for GES)
Let D be a dataset and let m be the amount of records in D. Then under the assumption that the data

is generated from a multivariate Gaussian distribution p, then in the limit of large m, applying GES to D results in the CPDAG of the class of DAGs faithful to p.

3.3 Non-standard data and ground truths

Under normal circumstances, the performance of a causal discovery algorithm is investigated as follows. One starts with the true causal system in a certain representation (called the ground truth) and data generated by that system. Then the causal discovery algorithm is applied to that dataset and the outcome is compared with the true causal system. For instance with GES, one has a DAG and data generated accordingly. Then one applies GES and receives a CPDAG and one can verify whether that CPDAG indeed represents the equivalence class of the DAG. If so, then GES produced the right outcome, if not, then GES made an error. The problem with data that is non-standard, in the sense that it is incompatible with the causal discovery algorithm, is that there is no obvious way to compare the ground truth with the outcome. For instance, when we apply GES to a system that has hidden common causes, representing the ground truth cannot be performed by writing it as a DAG since there are hidden common causes. One way of representing the ground truth, as mentioned, is by a PAG or MAG. Greedy equivalence search, on the other hand, outputs a CPDAG. These two structures are of different types and as a result, incomparable.
The situation becomes even more involved when the assumptions regarding the nature of causality (irreflexivity, antisymmetry and transitivity) do not hold in the causal system underlying a dataset. In section 6 this turns out to be a serious problem.
In this thesis, several situations occur when the ground truth is not compatible with the output of GES or modified PC. For these situations we choose alternative ways of evaluating the causal discovery algorithms and in some cases this means transforming the output of one of the algorithms.

4 Synthetic data

The main goal of this thesis is to evaluate causal discovery algorithms on real life systems in order to characterize, recognize and prevent mistakes made by causal discovery algorithms. This section serves as a preliminary step. Certain optimality results were considered in the previous section. This section tries to evaluate the causal discovery algorithms on data that satisfies the requirement of the optimality results of FCI. In this way we see how to interpret these theoretical results in the absence of an oracle and with a limited number of samples. We evaluate modified PC and GES on the same data. However, the data may include latent variables. Therefore, the data does not satisfy the requirements of the optimality results of GES.
Furthermore, in this section we also evaluate the impact of several parameters on the performance of the causal discovery algorithms. These parameters include among others the number of hidden variables, the sample size and the density of the causal DAG. The density of the causal DAG is expressed with the so-called expected neighborhood which gives the expected degree of the nodes.
This section starts with a description of the generation of the data and the causal DAGs. Then we apply the algorithms to a toy structure in order to see how modified PC and GES handle hidden common causes. After that we consider different scores of the output of the algorithms to see how well these algorithms perform.
The data is optimal for FCI. Hence, if the data size is sufficiently large, FCI should retrieve the full and correct ground truth. However, we test modified PC and not FCI. Since modified PC is similar to FCI and only slighty differs from FCI, it is expected that we can get near perfect outcomes when the sample size is sufficiently large. For GES this dataset is sub-optimal. The presence of hidden variables might compromise the outcomes. However, it still is the expectation that when the sample size is sufficiently large, the outcome will be near perfect. This is due to the fact that for parts of the ground truth in which there are no hidden common causes, GES should function perfectly. Since the score of GES can be decomposed, [5], these parts should be approximated perfectly for a sufficiently large sample size.
Finally, we conduct large scale evaluations using the previously scores.

4.1 Generating structures and data

The simulation of data is based on [8]. We generate linear Gaussian models. Given that we generate data on n variables, then we sample stochastic vectors x, satisfying

$$x = Ax + \varepsilon,$$

where x is an n-dimensional vector, A is an $n \times n$ lower triangular matrix and ε is an n-dimensional vector such that each component $(\varepsilon)_i \sim \mathcal{N}(0,1)$. Note that if $x = Ax + \varepsilon$, then $x = (I - A)^{-1}\varepsilon$. Furthermore, since $\varepsilon \sim \mathcal{N}(I, 0_n)$, we have that $x \sim \mathcal{N}(\Sigma, 0_n)$, where

$$\Sigma = (I - A)^{-1}((I - A)^{-1})^t.$$

The general procedure for data generation is given in pseudocode in algorithm 6. Here $\text{Bern}(x)$ stands for the Bernoulli distribution with parameter x and $\text{Un}(a, b)$ stands for the uniform distribution on the interval (a, b).

input : Number of variables n, number of hidden variables k, expected neighborhood EN and number of samples N
output: Dataset D

A← Square matrix of size n with zeros;
for $i, j : i < j \leq n$ **do**
  $A_{ij} \leftarrow$ Random draw from Bern($\frac{EN}{n-1}$)× Un(0.1,1)
end
$\Sigma \leftarrow (I - A)^{-1}((I - A)^{-1})^t$;
Temp_data← N samples from $Y \sim \mathcal{N}(\Sigma, 0_n)$;
Hidden_vars← random subset of size k from $\{1, ..., n\}$;
Erase the variables from Hidden_vars in Temp_data and call the result D;
return D

Algorithm 6: Data generation of synthetic data.

Depending on the parameter that is tested, the procedure is slighty altered to be as consistent as possible. For instance when the sample size is varied from M to K, then K samples are created and the dataset is restricted afterwards.

4.2 Toy structure

In this subsection we consider a toy structure in which we can see the influence of the hidden common cause requirement on GES. The matrix A and the corresponding causal DAG can be found in figure 3.

$$A := \begin{bmatrix} 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 & 0 \end{bmatrix},$$

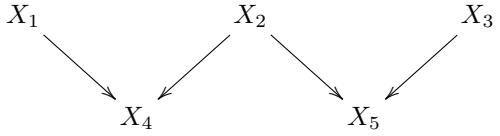


Figure 3: The matrix A and the corresponding causal DAG.

We take variable X_2 to be hidden. The PAG representing this causal system is depicted in figure 4.

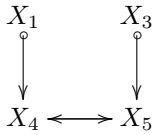
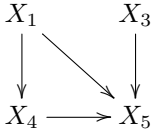
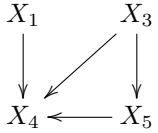


Figure 4: PAG representing the causal system of the toy structure.

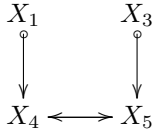
For this structure we generate 5000 samples and apply GES and modified PC to this dataset. For modified PC the threshold is 0.025. Iterating this procedure showed two different outcome for GES. The outcomes can be found in figure 5. In the outcome of GES an additional edge appeared. This is an example where the



(a) GES 1.



(b) GES 2.



(c) Modified PC.

Figure 5: Application of GES and modified PC to toy structure.

assumption that there are no hidden common causes affects the outcome of GES. In this case we see that, due to the hidden common cause, X_1 and X_5 are dependent given X_4 and X_3 and X_4 are dependent given X_5. Hence, the edge that is added raises the score. Modified PC does not add the additional edge since it is able to express the existence of a hidden common cause. This gives reason to believe that when the amount of hidden common causes increases, the amount of extra edges found by GES increases as well.

4.3 Scoring the outcomes

For large scale evaluation it is not feasible to consider the outcome of every application of the causal discovery algorithms. Hence we will score the outcomes. Furthermore, the outputs of GES and modified PC are different; modified PC outputs a PAG and GES outputs a CPDAG. Since the ground truth is a PAG we will transform the CPDAG of GES to a PAG and compare that PAG to the ground truth.
Ultimately we want algorithms that work well. It might be possible that in some circumstances modified PC performs better than GES and vice versa in other circumstances. Note that this comparison between modified PC and GES is not simply to determine which algorithm is best. Instead, noting which algorithm performs better in which circumstance might suggest a hybrid algorithm that performs better then either of these algorithms. In the end, the goal is to explore the gap and possible ways of bridging it, as discussed in the introduction.

Let G be the ground truth and let H be the PAG that is acquired via one of the the causal discovery algorithms. Then from the so-called **confusion matrix** of G and H we can derive various informative values. This confusion matrix is constructed as follows. Let 1 correspond to tails, let 2 correspond to arrowheads, let 3 correspond to circles and let 4 correspond to a blank. Then on entry i, j of the confusion matrix of (G, H) we have the *amount of times an edge mark corresponding to i has been interpreted as an edge mark corresponding to j*.
Given that one receives output H, then one is interested in the correctness of the result. That is, say, for instance that at some point H indicates the presence of an arrowhead. Then how likely is it that at that place in the graph there should actually be an arrowhead. This likeliness is expressed with the **precision**. Let C be the confusion matrix of (G, H) then the precision of the edge mark i is defined by

$$\text{Precision}(i) = \frac{C[i,i]}{\sum_{j=1}^{4} C[i,j]}. \tag{2}$$

This number should be interpreted as the fraction of the number of correct identifications of i and the total number of identifications of edge mark i.
Furthermore, we want to have an outcome that is as complete as possible, i.e. that as much information as possible about the ground truth is gathered. This is measured via the **recall**. The recall of the edge mark corresponding to i is defined by

$$\text{Recall}(i) = \frac{C[i,i]}{\sum_{j=1}^{4} C[j,i]}. \tag{3}$$

Furthermore, instead of just considering the correct orientation of edge marks, one is interested in finding out whether the determined adjacencies (i.e. causal interactions) are correct. This can be measured by comparing the skeletons of G and H. Recall that the skeleton of a graph J is the undirected graph on the nodes of J where there is an edge between nodes X and Y if and only if X and Y are adjacent in J.
For comparing the skeletons we want to know how much they are different and in what fashion. That is, we want to know how many edges one needs to alter to transform one graph into the other and we want to compare the amount of edges both skeletons have. The first is measured by the Structural Hamming Distance (SHD) for undirected graphs. The undirected SHD of graphs G and H is the amount of edges one has to remove/add in order to transform G into H. The latter is the value we call the average edge difference

(AED, for short). The average edge difference of graphs G and H is defined by

$$\text{AED}(G, H) = 100 \times \frac{E(G) - E(H)}{E(G) + 1},$$

where $E(X)$ denotes the amount of edges in graph X. The AED relates the amount of edges from G and H. Note that $\text{AED}(G, H)$ is positive iff G contains more edges than H. The value we compute in order to evaluate the skeleton is $\text{AED}(Skel(G), Skel(H))$, where G is the ground truth and H is the outcome of the algorithm.
Applying these scores to the structures from this section also serves as a test or validation of the used scores. This is useful for when we apply these scores to the relevant, real life systems.

4.4 Results

As mentioned, we vary several parameters. This is done by picking one parameter and while varying that one, keeping the other parameters on their default value. Concretely, we vary the sample size, the number of variables, the number of hidden variables, the expected neighborhood and the threshold. By default we use 5000 samples, fifteen variables, four of which are hidden, an expected neighborhood of 2.5 and a threshold of 0.025. For each setting the recall, precision, skeleton SHD and AED is averaged over 100 generated models. In the precision recall graphs only the first value of the varied parameter is displayed.

4.4.1 Sample size

The sample size takes values in $\{100, 500, 1000, 2000, 5000, 10000, 1000000, 10000000\}$. The results are depicted in figure 6 and 7.

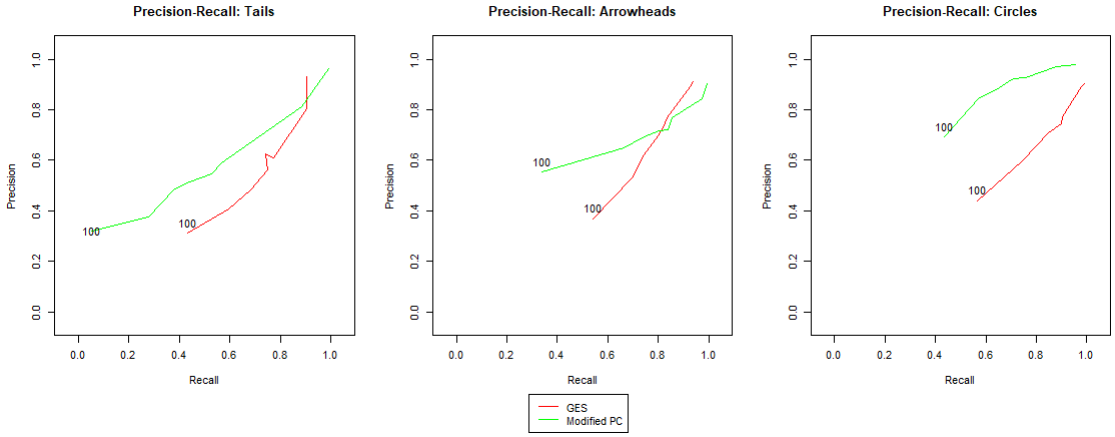


Figure 6: Precision recall plots resulting from varying the number of samples.

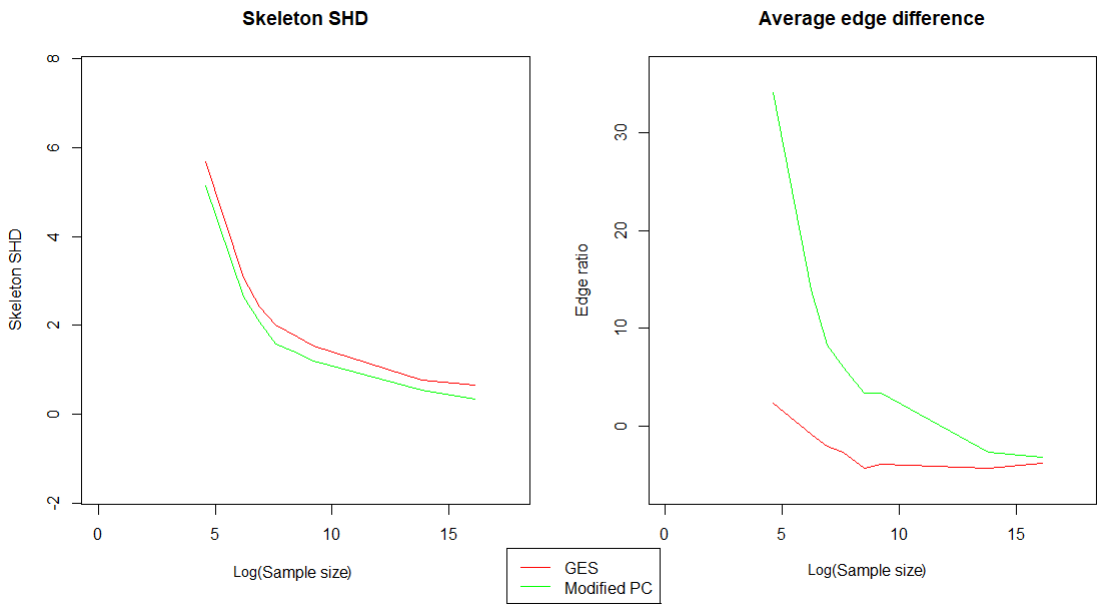


Figure 7: Skeleton results after varying the number of samples.

With respect to precision-recall plots, one can detect a clear trend: when the amount of samples increases, both the precision and recall increase. This is consistent with the expectation: when the sample size increases, the conditional independence tests become more accurate. Ergo, with enough samples it will return the same outcomes as the oracle and, hence, the precision and recall of the outcome of modified PC increase. For GES one can run a similar argument regarding the convergence of the score when the sample size increases.
However, when the sample size is 10000, the precision and recall of the tails are 50/50 and the precision and recall of arrowheads and circles is around 80 percent. This indicates that a sample size of 10000 is not large enough to have a near perfect outcome. Furthermore, the low precision and recall of tails in comparison to the arrowheads and circles is due to the fact that there are relatively little tails in PAGs. Hence, there are not many tails present in the ground truth and even the slightest mistake may result in a missed tail or an erroneous tail.
Regarding the skeleton outcomes, for both GES and modified PC the skeleton SHD decreases asymptotically to 0 as the sample size increases. Furthermore, modified PC first underestimates the amount of edges and then eventually overestimates the amount of edges. The graph also shows that greedy equivalence search is more likely to overestimate the amount of edges. This is consistent with what we found for the toy structure. There GES overestimated the amount of edges due to the presence of hidden common causes.

4.4.2 Number of variables

We take the number of variables to be 8, 10, 15, 20, 25 and 50. The results are depicted in figures 8 and 9.

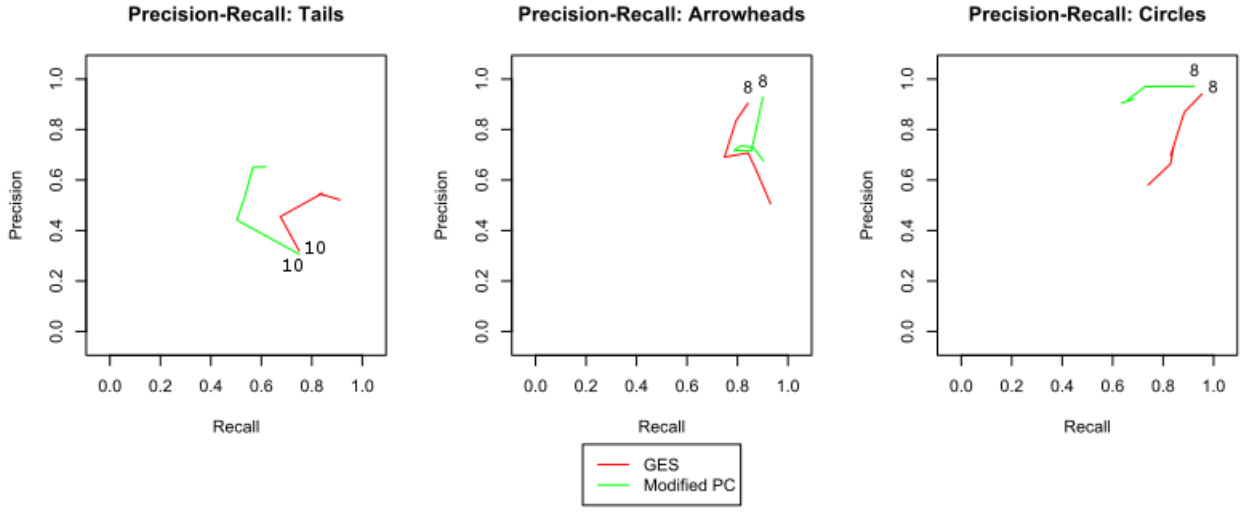


Figure 8: Precision recall plots resulting from varying the number of variables.

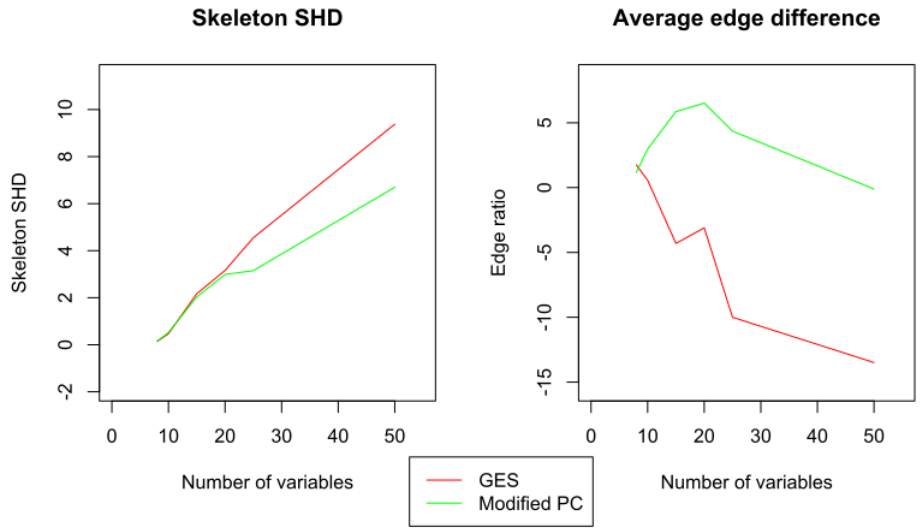


Figure 9: Skeleton results after from varying the number of variables.

For the arrowheads an increase in variables leads to a decrease of precision. This holds for both GES and modified PC. With regards to the circles, an increase in variables leads to a decrease in both precision and recall. For modified PC there is a slightly larger decrease in recall than in precision and for GES a larger decrease in recall than in precision. From the precision recall graph of the tails one cannot detect a clear trend. Note that the precision and recall of tails are roughly of the same size as in the situation of 5000 samples in section 4.4.1. Furthermore, recall that the amount of tails in PAGs is relatively low, hence it is likely that the impact of the number of variables on the precision and recall of tails is only marginal and that the variations are due to the fact that the amount of tails is so low.
Furthermore, note that when it comes to circles modified PC is very precise, whereas the precision of GES decreased. Therefore, when the amount of variables is large, one can very well trust the circles outputted

by modified PC.
The results regarding the skeletons illuminate the situation even more. The skeleton SHD of both GES and modified PC increases as the number of variables increases, whereas the AED seems to converge to 0 for modified PC and seems to decrease for GES. This suggests that modified PC does make mistakes with the determination of the skeleton but that it does not overestimate the amount of edges. On the other hand, GES does overestimate the amount of edges.
This decrease in skeleton is very likely due to the amount of samples. An argument can be made that as the number of variables increases, more conditional independence tests are executed and hence, there are more situations in which modified PC can make a mistake. Raising the amount of samples then reduces the amount of mistakes. Retrials of this test where not only the amount of variables are increased but also the number of samples, support this claim.

4.4.3 Number of hidden variables

The number of hidden variables takes values in $\{0, 2, 4, 6, 10\}$. The results are depicted in figure 10.

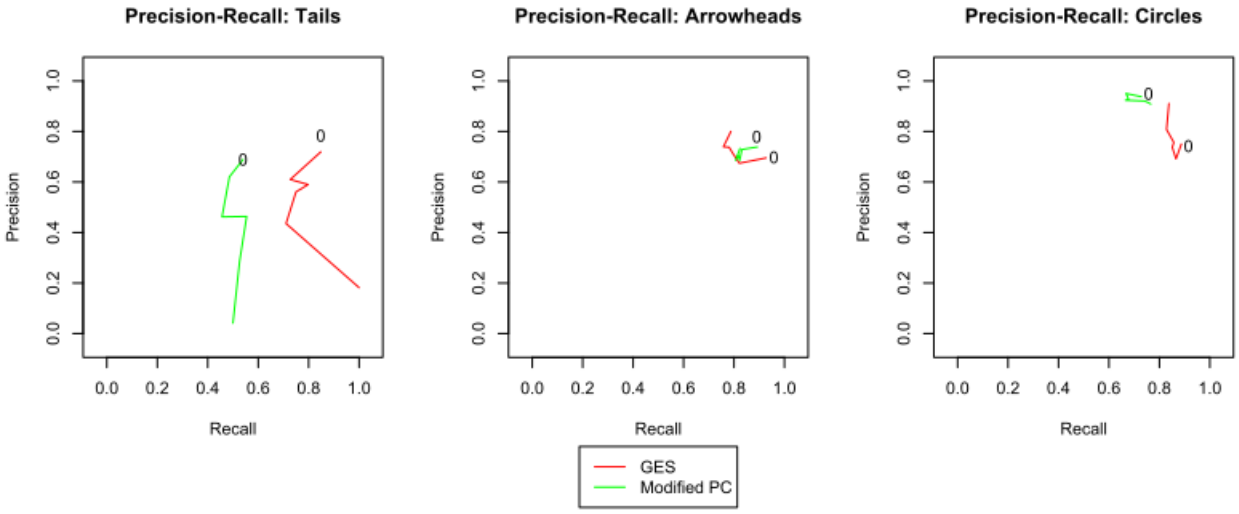


Figure 10: Precision recall plots resulting from varying the number of hidden variables.

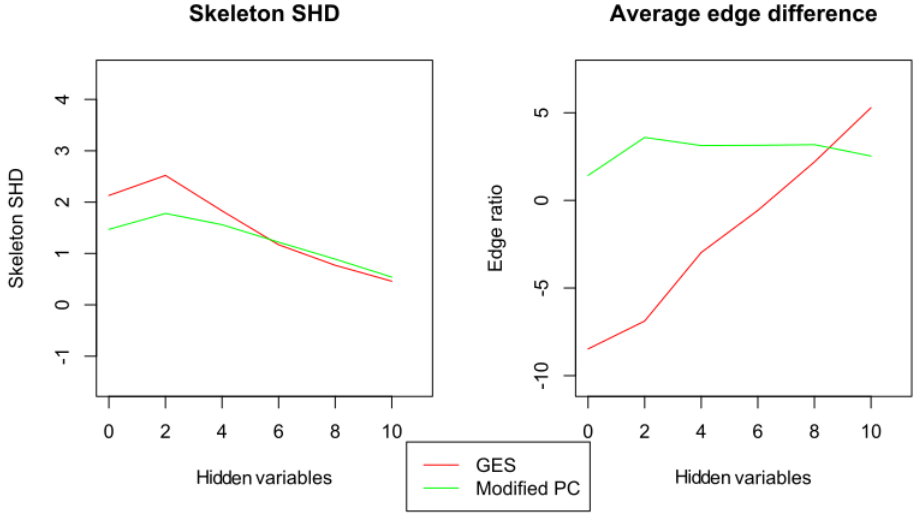


Figure 11: Skeleton results after varying the number of hidden variables.

We see that as the number of latent variables increases, the precision of the tails drops, the recall for modified PC stays roughly the same. For arrowheads and circles the increase in hidden variables seems to have very little effect on the precision and recall, except for an increase of the precision of GES on circles. The overall performance of GES is similar to modified PC, if not better. This seems strange as modified PC is able to handle hidden common causes and GES is supposed not to be able to. Note however that this test measures the impact of hidden variables. However, the amount of hidden *common* causes is not directly related to the amount of hidden variables. Not only can it be that variables are omitted that are not hidden common causes, but it can also be that the one of the effects of a common cause is removed.
The results regarding the skeleton provide additional information. When none of the variables are hidden, GES overestimates the amount of edges and as the amount of hidden variables increases, it increasingly underestimates that amount of edges. Modified PC consistently underestimates the amount of edges. The skeleton SHD, however, first increases and then decreases. Most notably for GES.

4.4.4 Threshold

The threshold takes values in $\{0.0001, 0.001, 0.01, 0.025, 0.05\}$. The results are depicted in 12. The outcome of GES is also depicted as a reference point.

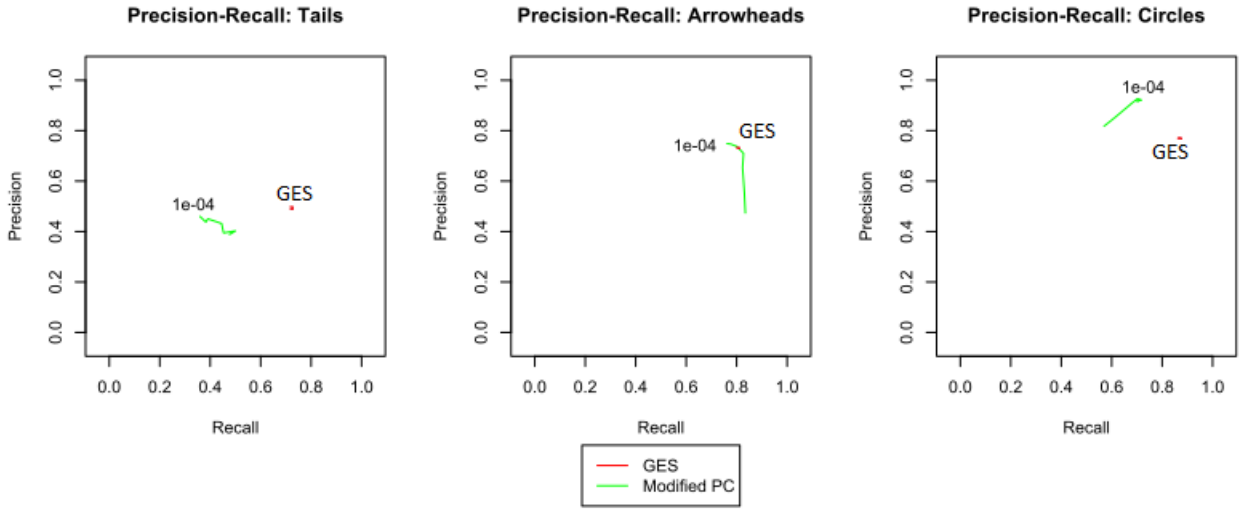


Figure 12: Precision recall plots resulting from varying the threshold.

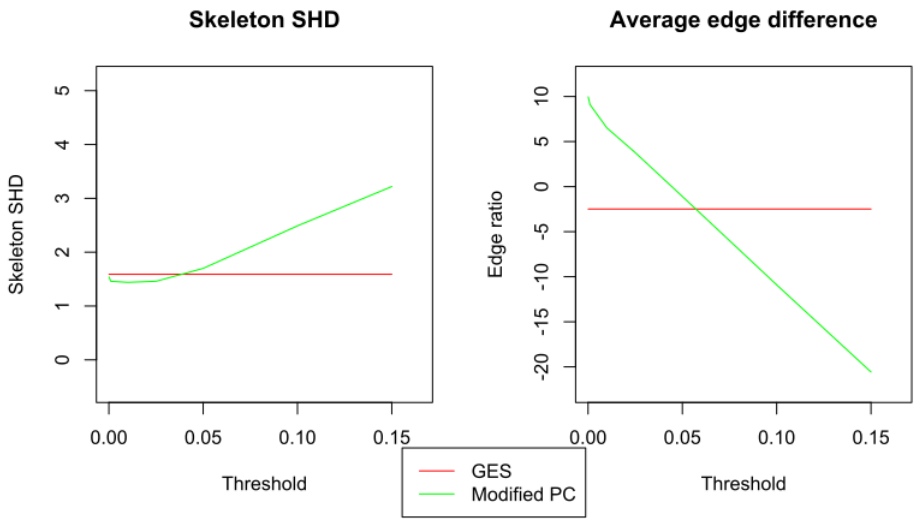


Figure 13: Skeleton results after varying the threshold.

One would expect that if the threshold is too high, one would have too many adjacencies and if one has a threshold too low, one would have too few adjacencies. The skeleton results are consistent with this view. There seems to be an optimal value of the AED for the threshold. Indeed, when the threshold is too low, modified PC underestimates the amount of edges. When the threshold is too high, modified PC overestimates the amount of edges. However, one would also expect to see a clear minimum for the skeleton SHD. However, there might be a slight minimum but it is not a convincing minimum. In the precision recall graphs of the tails and the arrowhead one can detect a mild optimal value of the threshold.
Iterations of the simulations show more convincing optimal values.

4.4.5 Density

We measure the density in the amount of expected neighbors each node has. The expected neighborhood will take the values $\{2, 2.5, 3, 3.5, 4\}$.

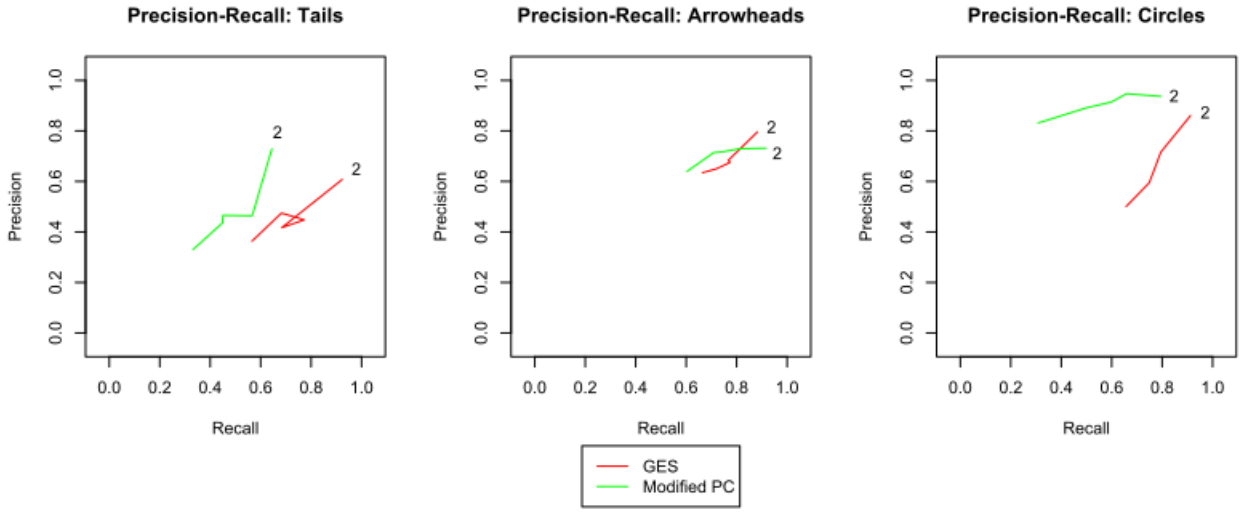


Figure 14: Precision recall plots resulting from varying the expected neighborhood.

From these results we can conclude that as the expected neighborhood increases, precision and recall decrease regarding the tails and arrowheads (both GES and modified PC). For the circles we see the same relation for GES. For modified PC, however, we see only a clear decrease in recall on the circles and only a slight drop in precision.
There are two possible explanations for this fact. First as the in-degree of certain nodes increases, the effect of the parents on that node decreases. This might both result in an erroneous score in GES or an erroneous conditional independence test. Furthermore, recall that the conditional independence tests are executed on neighbors. When there are more neighbors in the ground truth, the PC algorithms will conduct more conditional independence tests. This increases the chance on an error. Similarly, for GES, when there are more adjacent variables, the algorithm could be inclined to erase more edges, since there are more causes and GES tries to eliminate overdetermination. Both of these possible explanations should be visible in the results of the skeleton comparisons (figure 15); the amount of edges should be underestimated by GES and modified PC.

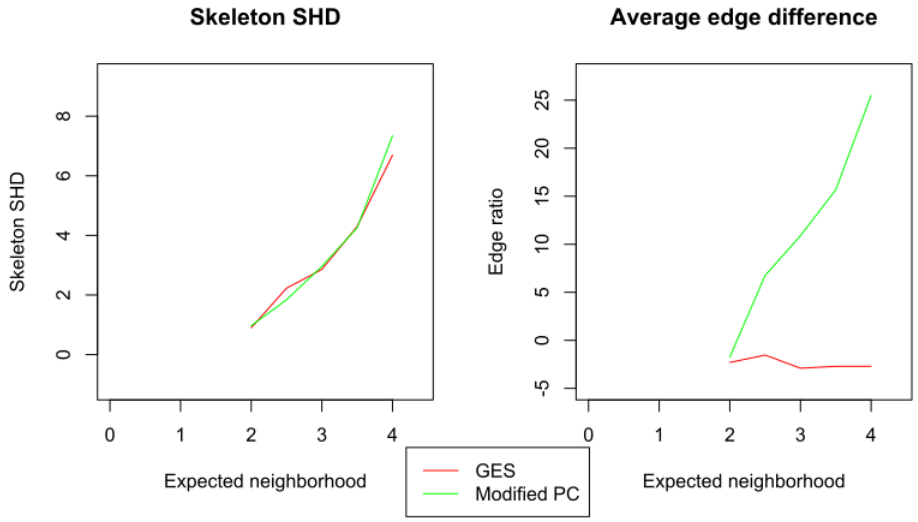


Figure 15: Skeleton results after varying the expected neighborhood.

As the expected neighborhood increases, modified PC underestimates the amount of edges. Greedy equivalence search however slightly overestimates the amount of edges and the amount of overestimation remains equal as the expected neighborhood increases. It could be that inclination of GES to increasingly overestimate the amount of edges as the number of latent common causes increases, balances out the increased erasing of edges of GES to avoid overdetermination. Indeed, the skeleton SHD of GES increases as the expected neighborhood increases. Combining this with the results of the AED, we conclude that either GES sometimes overestimates and sometimes underestimates the amount of edges or GES continually has the correct amount of edges. In the latter case GES placed edges where they should not belong and did not place edges where they should belong.

4.5 Conclusion and discussion

The results in this section show that the used scores for evaluating the outcomes of modified PC and GES are informative. Furthermore, we have established a base line with which we can compare the results from sections 5-7. The results in this section gave us an indication that GES is prone to overestimating the amount of edges, especially if the number of variables increases. Seeing the effects of non-standard data to this overestimation is a point of interest for the following sections. Furthermore, the effect of the sample size on the performance of modified PC and GES is significant and extremely large sample sizes are needed for a near perfect result. In practice, researchers seldom have these large quantities of samples and in the real life simulations we might find that the sample sizes are indeed too small to receive a near perfect outcome.

5 fMRI data

In this thesis we try to get a view of how GES and modified PC perform on realistic datasets. Furthermore, we intend to explore the gap between the theoretical optimality results and the application of causal discovery algorithms in practice. The previous section considered the performance on datasets that are close to ideal for modified PC, i.e. conform the theoretical optimality results. In the two following sections we evaluate modified PC and GES on simulated real world systems which are not conform the optimality results. This section focuses on a real world system that lies in between these two ends of the spectrum. Although the simulated data represents a real world system, the dataset still satisfies many requirements of the optimality results. The latter statement will be made more precise in the description of the generation of the data.
The dataset we consider regards brain research. Researchers try to understand the working of the brain by establishing relations between various regions of the brain. One tries to find out whether activity in one region causes activity in another region. Functional magnetic resonance imaging is a tool to do so. An fMRI scan measures blood flow in blood-oxygen-level-dependent (BOLD) data. Active regions in the brain demand blood with higher oxygen levels. Hence, by measuring the blood flow of oxygen rich blood of various brain regions one can gain insight in neural activity these regions. The precise connection between neural activity and blood flow depends on various parameters, the exact details of this connection are beyond the scope of this text. The picture in figure 16 schematically depicts the procedure of fMRI.

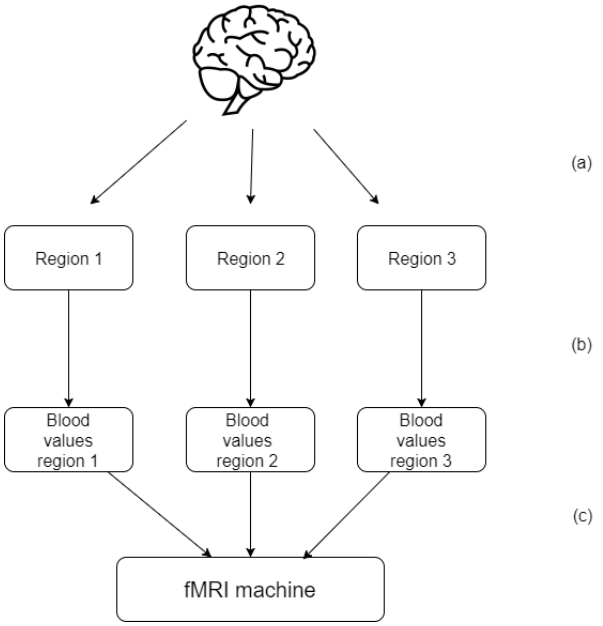


Figure 16: Schematic depiction of the process of fMRI. In (a) certain regions of the brain are determined. Neural activity influences the blood values near these regions (b). These variations in the blood values are then measured by the fMRI machine (c).

This section is outlined as follows. First we describe the generation of the datasets as performed in [24]. Then we consider how we will score the outcomes. After that we consider how the threshold influences the outcome and lastly we compare the outcomes of applying GES and modified PC to several datasets with a fixed threshold.

5.1 Data simulation

The datasets we consider are from [24]. A precise description of how this data is generated can be found in [24]. It suffices to mention that they generate a weighted DAG representing regions of the brain and the connectivity between these regions. This is consistent with the belief that causality is transitive, irreflexive

and antisymmetric. Therefore, this view is consistent with the standard assumptions. From this graph a simulation is determined for the neural activity of the regions. Each of the nodes has an on/off switch which gives a surge of electricity to that node. That signal then travels through the remaining nodes according to the underlying weighted DAG. This can be expressed by the following, simplified, differential equation:

$$\dot{z}(t) = Az(t) + u.$$

Here $z(t)$ is the vector with the neural activity as a function of the time, u represents the external input, A represents the connectivities between the nodes. Furthermore, the matrix A has -1 on the diagonal in order to simulate temporal decay of the electric signal.
As mentioned, fMRI does not measure the neural activity but BOLD data. Therefore, this simulation for neural activity is translated to a simulation for BOLD data. The resulting dataset consists of time series where these BOLD values are measured at several moments.
For each graph, time series of several so-called subjects are created, this represents having several test-subjects from which one has obtained fMRI data. For each of these subjects the true connectivity between the nodes in the graph is slightly varied, representing differences in neural structures between various individuals. An overview of the data generation is presented in figure 17.

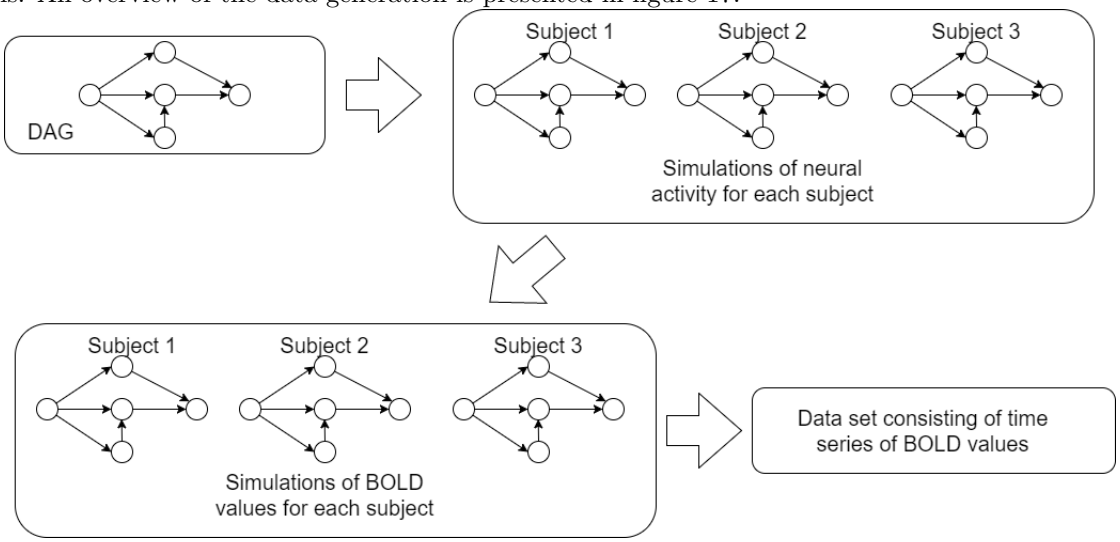


Figure 17: First a DAG is generated and then simulations of the neural activity are created for each test subject with slightly different parameters. Afterwards, this is translated to BOLD values which are consequently measured at different times.

We consider datasets on five, ten and fifty variables. These are generated by the DAGs from figure 18.

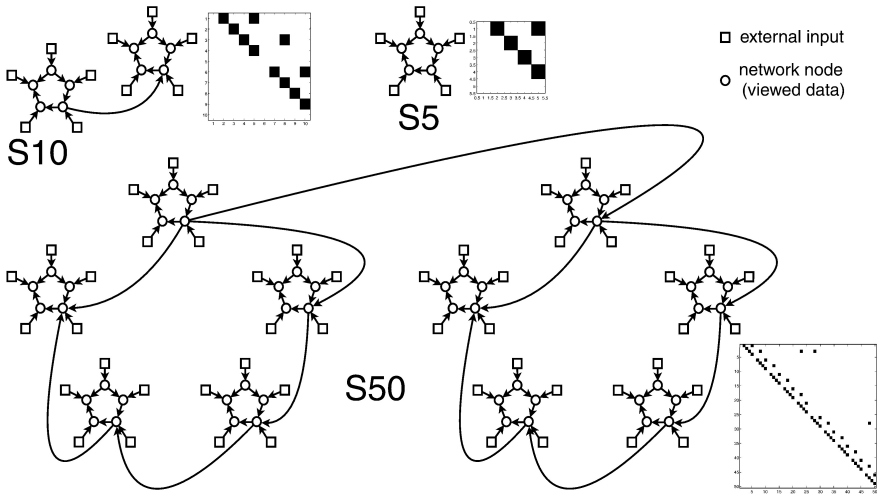


Figure 18: Graphs and matrices corresponding to the networks of five, ten and fifty nodes. The external inputs operate as off and on switches of neural activity. The neural activity then works through the other nodes according to the connections between the nodes. The figure is from [24].

Whereas the article from which the data is obtained, [24], considered 28 datasets in total, we only consider datasets 1, 2, 4, 6 and 7. These are datasets on five, ten and fifty nodes and they are datasets of BOLD values under normal circumstances. That is, no additional pertubations of the system occur (e.g. shared external inputs). Simulations 1 and 7 both are generated by the DAG on five nodes. However, simulation 1 only has 200 samples per subject and simulation 7 has 5000 samples per subject. This allows us to see the impact of the sample size on the outcomes of modified PC and GES. Similarly, simulations 2 and 6 are generated by the DAG on 10 nodes. However, simulation 2 has 200 samples per node and simulation 6 has 1200. Lastly, simulation 4 is on a graph of 50 nodes and has 200 samples per subject. Unfortunately there is no dataset available on 50 nodes and with a high number of samples per subject. Detailed descriptions of the data of each simulation can also be found in [24].
Similarly as in section 4, we transform both the underlying DAG as the outcome of GES to a PAG. This allows us to possibly see an improvement of GES and modified PC using a combination of the two.

5.2 Threshold variations

We now vary the threshold and evaluate how the threshold influences the results of modified PC. We do this for simulation 2, 4 and 6. For this evaluation we compute the precision and recall for tails, circles, arrowheads and the skeleton (i.e. the precision of the skeleton is the amount of correct edges divided by the amount of found edges, the recall of the skeleton is defined analogously). Furthermore, we consider the so-called **accuracy**. The accuracy is defined as the ratio of correct orientations and the total amount of orientations. The choice for a blank (i.e. no edge) is also considered as an orientation. The presented precision, recall and accuracy for each simulation are then averaged over the subjects.
For the data on the five node network, the results are depicted in figure 19.

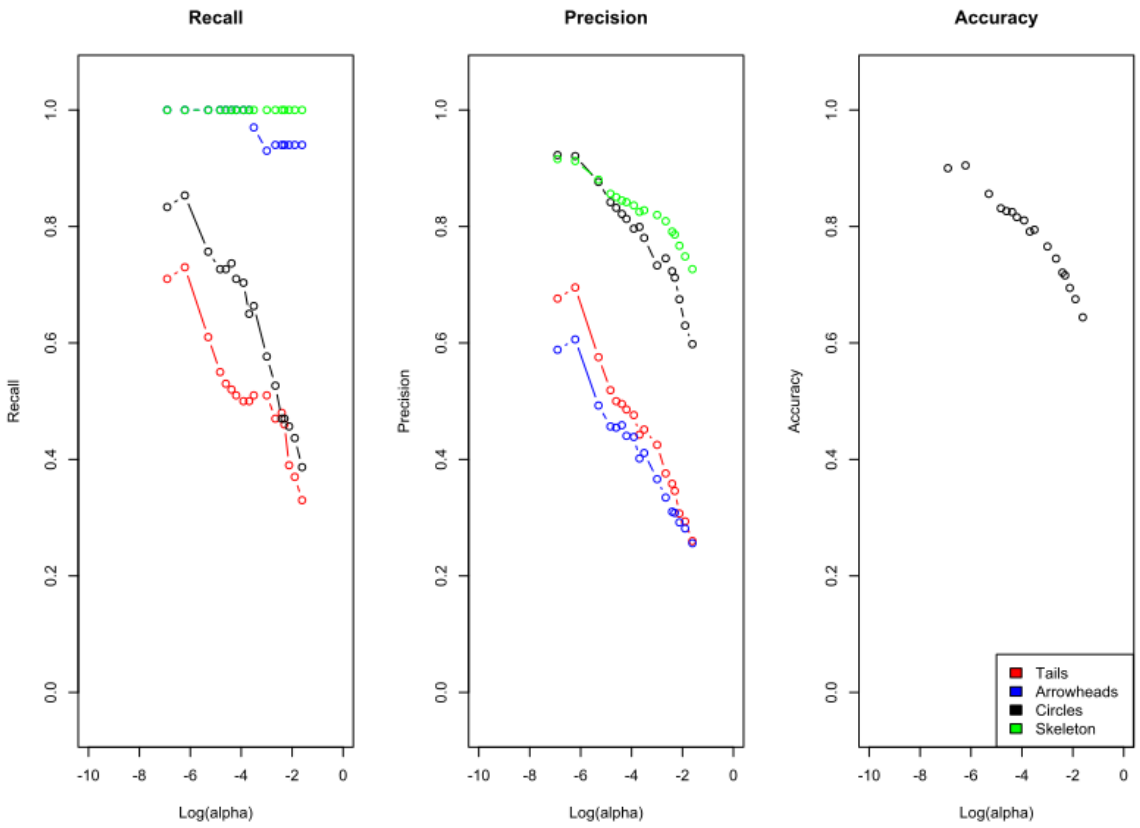


Figure 19: Recall, precision and accuracy resulting from varying the threshold for modified PC on simulation set 7. The x-axis displays the log of the threshold, indicated by log(alpha).

For the ten node network, the results are depicted in figure 20.

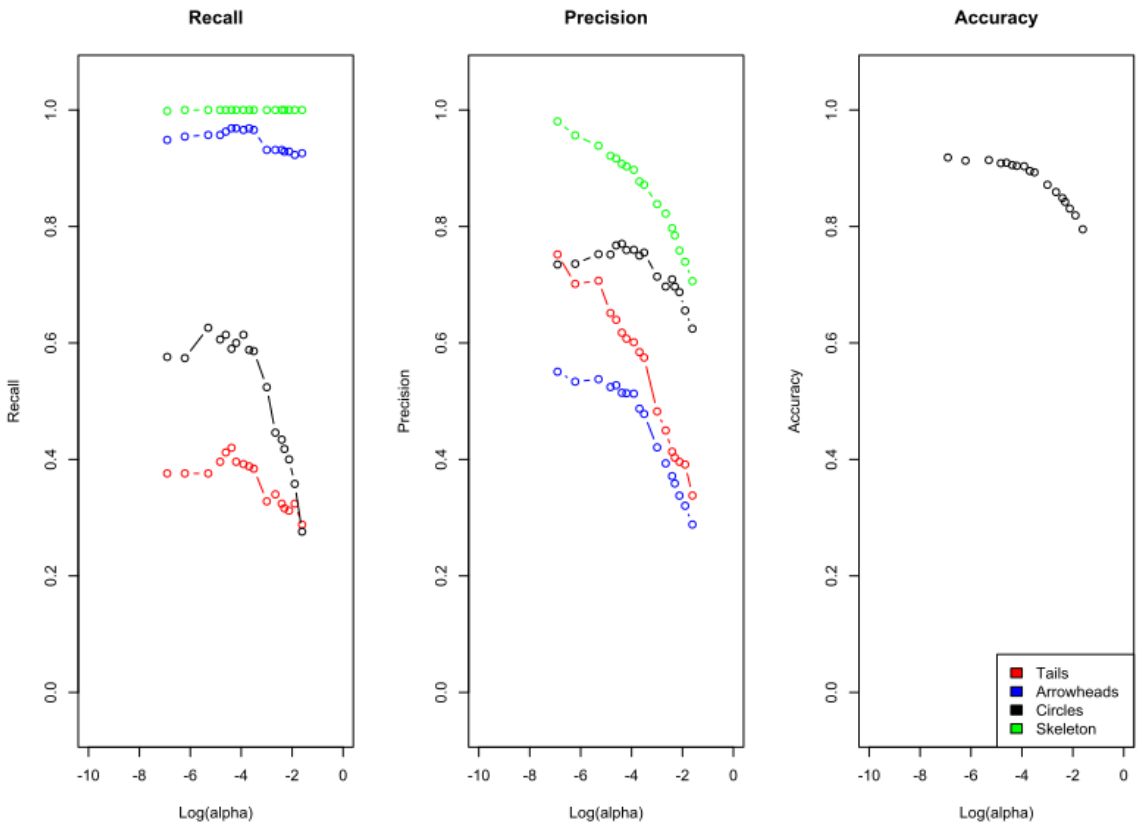


Figure 20: Recall, precision and accuracy resulting from varying the threshold for modified PC on simulation set 6. The x-axis displays the log of the threshold, indicated by log(alpha).

Note that the order of lines from top to bottom is similar between the five and the ten node network. Furthermore, in both cases it becomes clear that the amount of edges is overestimated by modified PC, since the recall is steady at 1 while the precision on the other hand decreases as the threshold increases. As mentioned, when the threshold decreases, two variables are more likely to be rendered conditionally independent. Combining the consistent recall of 1 with a precision that decreases as the threshold increases, we conclude that for the five node network, the threshold should be lower than the tested values in order to retrieve the perfect skeleton. For the ten node network, the skeleton seems to be approximated the best when log(alpha) is near -8.
On the other hand, for the ten node network, the orientations seem to indicate a local maximum in recall when log (alpha) is between -6 and -4. For the precision the circles seem to be optimal between -6 and -4, but the arrowheads and the tails seem to be optimal for log(alpha) near -8.

For simulation four, the results are displayed in figure 21.

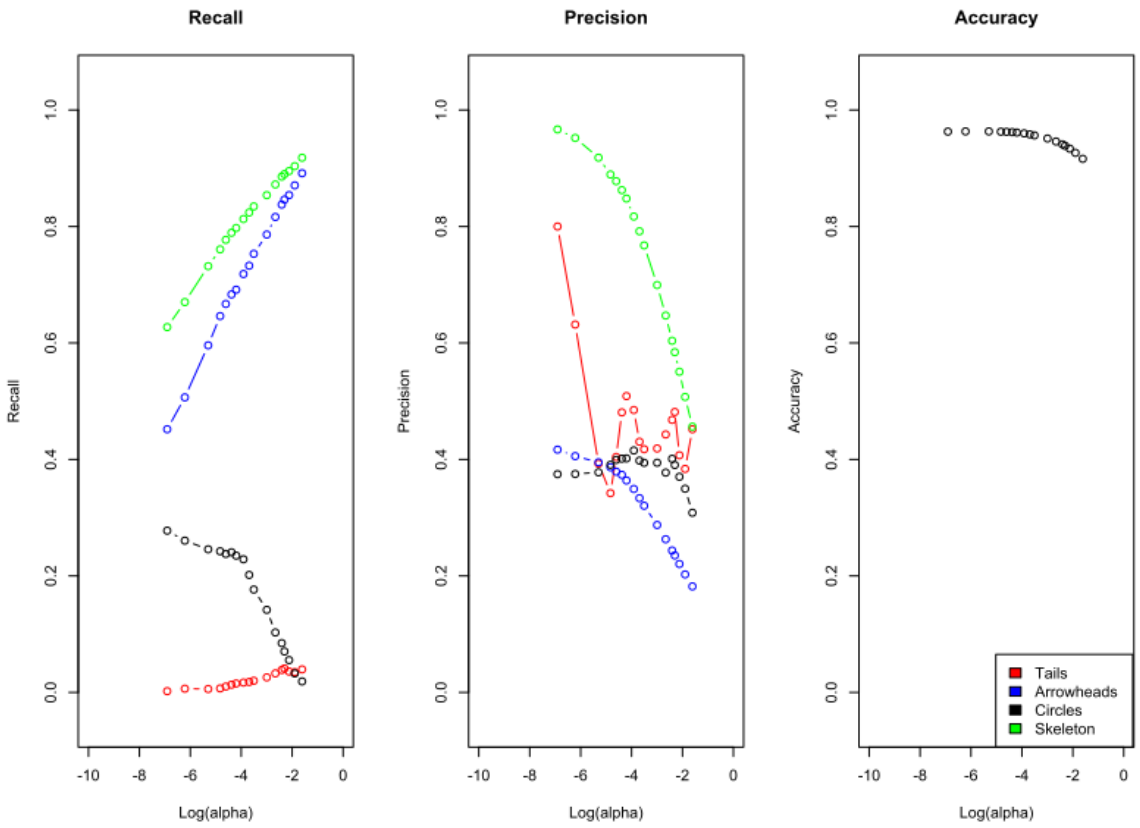


Figure 21: Recall, precision and accuracy resulting from varying the threshold for modified PC on simulation set 4. The x-axis displays the log of the threshold, indicated by log(alpha).

For the fifty node network, the recall of the skeleton is not fixed at 1, instead it increases as the threshold increases. The precision of the skeleton decreases as the threshold increases. This is as is to be expected, as mentioned before. The recall of tails is very low and the precision of tails decreases rapidly from 0.8 to roughly 0.4 and remains at that level.

5.3 Fixed threshold

As mentioned, we now fix the threshold. We choose to fix it at 0.025, since in section 4 that proved to be a reasonable choice. However, as the previous results show, this is a sub-optimal choice for the following dataset. On the other hand, a similar threshold optimization cannot be performed for real life data and, based on section 4, fixing the threshold at 0.025 is a reasonable choice.
As mentioned, we evaluate the performance of GES and modified PC on datasets 1, 2, 4, 6 and 7. Here we calculate the precision and recall of tails, arrowheads and circles as well as the skeleton SHD. The presented values are averaged over the test subjects. Furthermore we present the sum of the confusion matrices, cf. section 4.3, of each subject. For simulation 1 we get the following results.

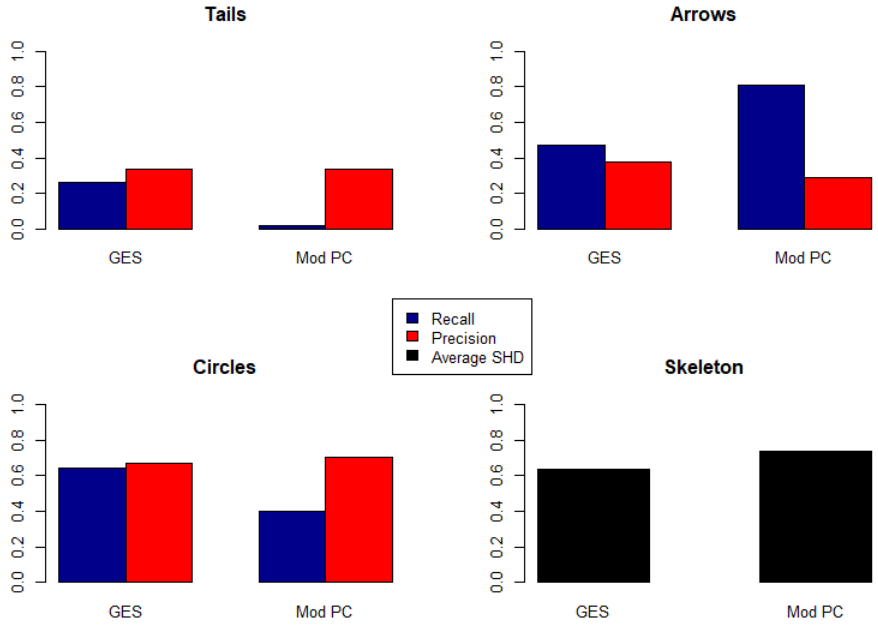


Figure 22: Precision and recall for tails, arrows and the average SHD resulting from applying modified PC and GES to simulation 1.

The tests yielded the following confusion matrices.

Modified PC

	Tail	Arrow	Circle	Blank
Tail	2	48	40	10
Arrow	0	81	9	10
Circle	3	138	121	38
Blank	1	13	2	734

GES

	Tail	Arrow	Circle	Blank
Tail	26	12	55	7
Arrow	10	47	36	7
Circle	31	53	194	22
Blank	11	13	4	722

Figure 23: Confusion matrices of modified PC and GES on simulation 1.

For this dataset the performance of modified PC is quite poor. The recall of tails with modified PC is extremely low. The confusion matrix shows that although there were 100 tails to be found, modified PC retrieved only two. For arrows, modified PC obtained a high recall but at the expense of precision.
For GES the precision is usually of the same size as the recall. The precision and recall of circles is, at around 0.6, relatively high. Still, the performance of GES is not as good as one would hope.
It is remarkable that the skeleton SHD for GES and modified PC is lower than 1. Combining these results, the skeleton is determined rather accurately by GES and modified PC, but the small error in skeleton has a major effect in the precision of the orientation.
One possible explanation for these disappointing results is that the amount of samples is too low. Indeed, considering the effects of sample size on the performance as measured in section 4, a sample size of 200 is far too low.
For that reason we consider simulation 7 which has 5000 samples per subject. The results for simulation 7 are depicted in figures 24 and 25.

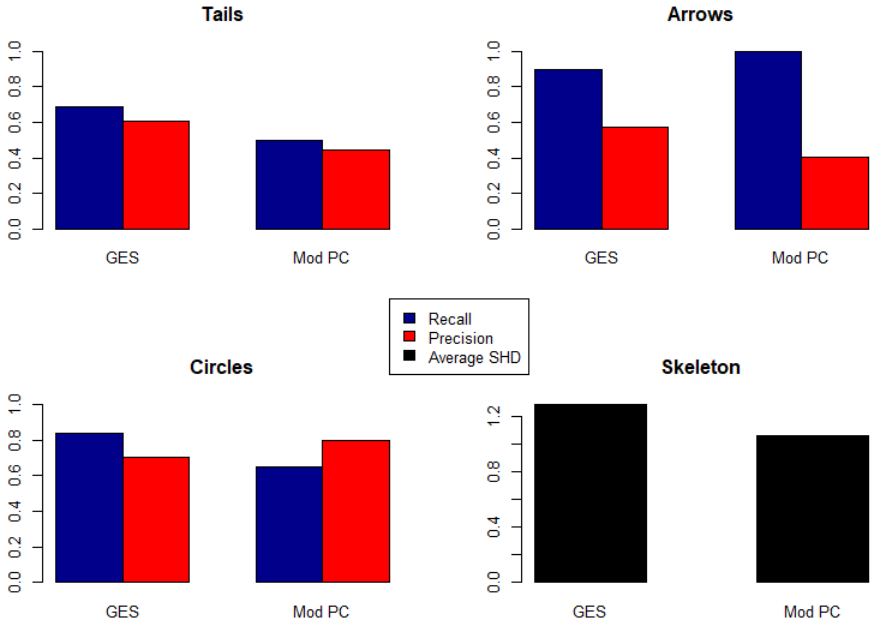


Figure 24: Precision and recall for tails, arrows and the average SHD resulting from applying modified PC and GES to the dataset of simulation 7, consisting of 5000 samples per subject.

Modified PC

	Tail	Arrow	Circle	Blank
Tail	50	20	30	0
Arrow	0	100	0	0
Circle	34	71	195	0
Blank	29	58	19	644

GES

	Tail	Arrow	Circle	Blank
Tail	69	1	30	0
Arrow	1	90	9	0
Circle	20	28	252	0
Blank	24	38	66	622

Figure 25: Confusion matrices of modified PC and GES on simulation 7.

The precision and recall of all edge marks for both GES and modified PC increased, in comparison to simulation 1. Especially the recall of modified PC for tails is far less extreme. Furthermore, the skeleton SHD of GES and modified PC increased slightly. Hence, although the skeleton is determined less accurately, the orientation is estimated far better. It also stands out that the number of blanks is underestimated[4] for both GES and modified PC, as can be seen in the last columns of the confusion matrices.
In comparison to the results of section 4.4.1, the recall of the arrows is too high and the precision of the arrows is too low.
Hence, we conclude that the performance of modified PC and GES for this dataset is slightly poorer than in the situation of 4.4.1 with 5000 samples, but still very close.

For the network on ten nodes, we consider simulations 2 and 6 of respectively 200 and 1200 samples per subject. The results of simulation 2 can be found in figures 26 and 27.

[4]Equivalently, the amount of adjacencies is overestimated.

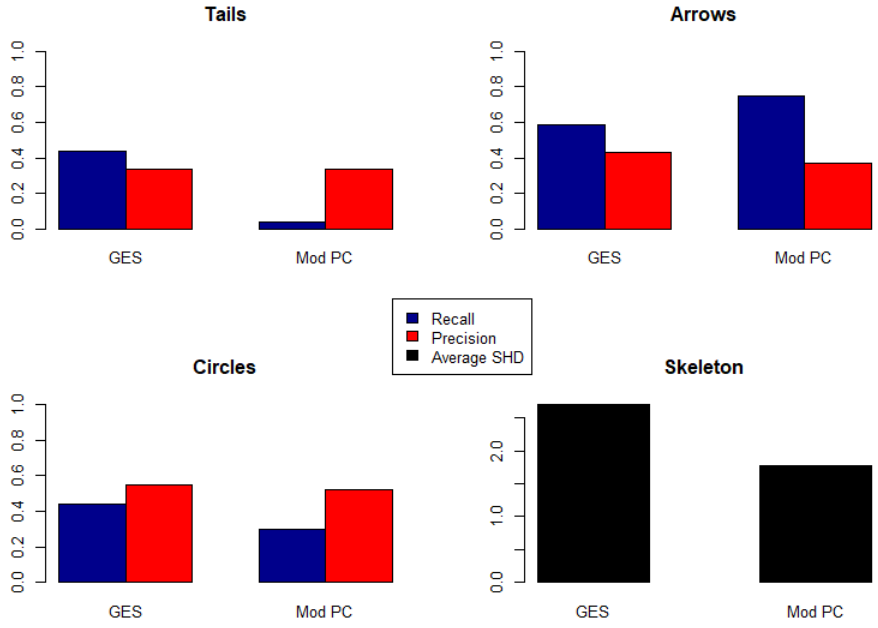


Figure 26: Precision and recall for tails, arrows and the average SHD resulting from applying modified PC and GES to the dataset of simulation 2, consisting of 200 samples per subject.

Modified PC

	Tail	Arrow	Circle	Blank
Tail	9	128	85	28
Arrow	0	263	47	40
Circle	17	278	151	54
Blank	1	47	8	3844

GES

	Tail	Arrow	Circle	Blank
Tail	110	41	80	19
Arrow	53	205	67	25
Circle	101	138	221	40
Blank	64	90	32	3714

Figure 27: Confusion matrices of modified PC and GES on simulation 2.

The orientation results are similar to the orientation results of simulation 1. The skeleton SHD, however, has doubled for modified PC and tripled for GES. This is consistent with the in section 4 measured effects of the number of variables on the skeleton SHD. We see that as the number of variables increases, the skeleton SHD increases faster for GES than for modified PC.

The results for simulation 6 can be found in figures 28 and 29.

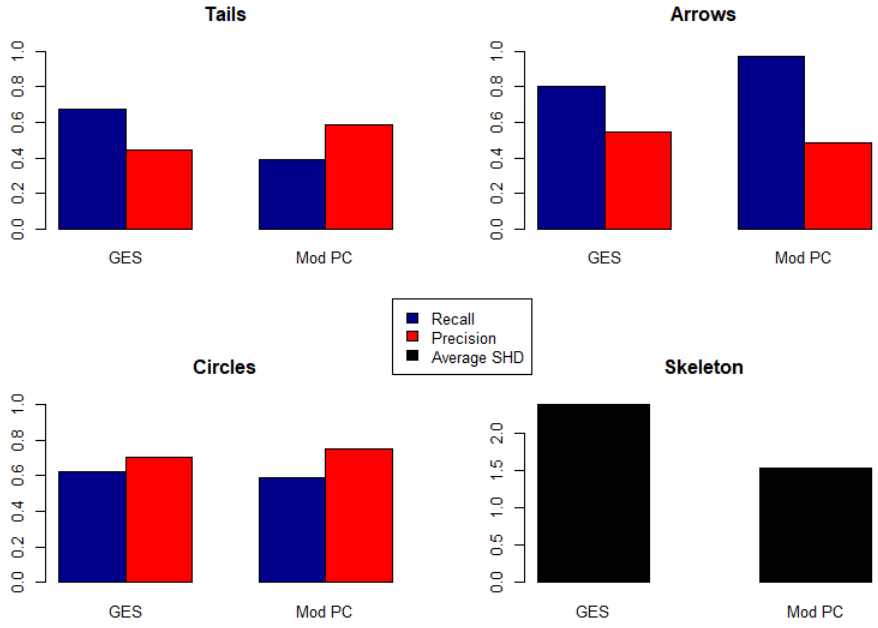


Figure 28: Precision and recall for tails, arrows and the average SHD resulting from applying modified PC and GES to simulation 6, consisting of 1200 samples per subject.

Modified PC

	Tail	Arrow	Circle	Blank
Tail	97	76	77	0
Arrow	3	339	8	0
Circle	56	150	294	0
Blank	10	131	13	3746

GES

	Tail	Arrow	Circle	Blank
Tail	168	22	60	0
Arrow	26	281	43	0
Circle	89	98	313	0
Blank	96	112	30	3662

Figure 29: Confusion matrices of modified PC and GES on simulation 6.

The precision and recall have improved, furthermore, the skeleton SHD has remained constant. Hence, we see that with more samples, the skeleton is estimated just as accurate as with simulation 2. However, for simulation 6, the impact of these errors is less than with simulation 2.
When comparing these results to the outcome of simulation 7, we see that the overall performance on simulation 6 is slightly lower. Based on the results of section 4 there are two possible causes for this slight decrease in performance, namely that the sample size of simulation 6 is lower than the sample size of simulation 7 and that the amount of variables has increased.
Furthermore, just as for simulation 7, the confusion matrices confirm that indeed both GES and modified PC overestimate the amount of adjacencies for simulation 6. An improvement of the skeleton can hence be achieved by raising the threshold.

We conclude with the network on 50 nodes. Based on the previous results, one can expect a very poor performance due to low amount of samples. Furthermore, since the number of variables has increased, we can expect the skeleton SHD of both GES and modified PC to increase as well. In particular the skeleton SHD of GES. The results can be found in figures 30 and 31.

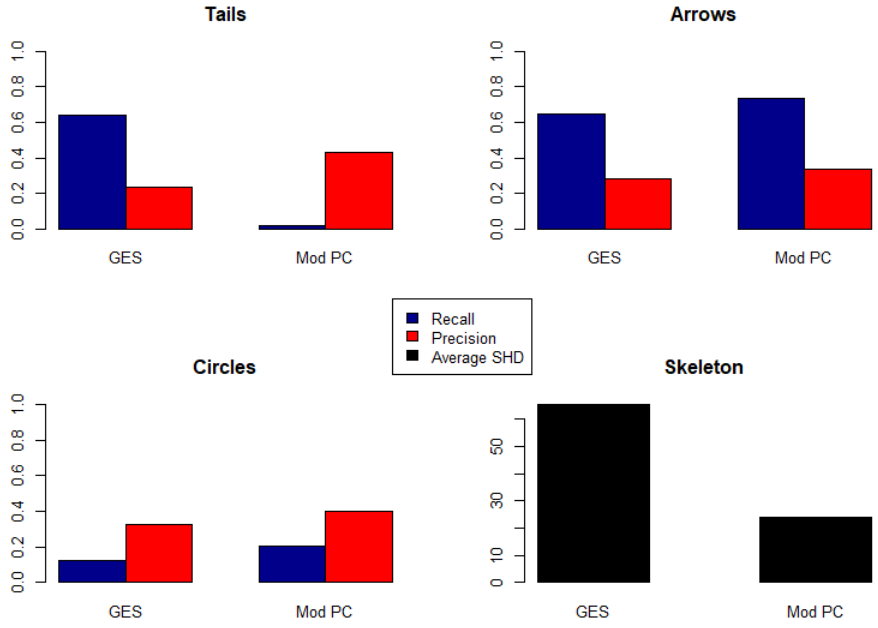


Figure 30: Precision and recall for tails, arrows and the average SHD resulting from applying modified PC and GES to simulation 4, consisting of 200 samples per subject.

Modified PC

	Tail	Arrow	Circle	Blank
Tail	34	1241	300	325
Arrow	16	1795	195	444
Circle	21	1071	353	305
Blank	8	1246	68	117578

GES

	Tail	Arrow	Circle	Blank
Tail	1223	414	102	161
Arrow	540	1580	97	233
Circle	658	692	216	184
Blank	2749	2945	250	112956

Figure 31: Confusion matrices of modified PC and GES on simulation 4.

Indeed, performance of GES and modified PC is low. Furthermore, the average skeleton SHD of GES and modified PC has increased. The skeleton SHD of GES is larger than the amount of edges present in the ground truth. As poor as this may be, it is consistent with the previously observated phenomenon that the amount of edges for GES increases as the amount of variables increases. The recall of tails and arrowheads for GES and the arrowheads for modified PC, however, are relatively high. Upon considering the confusion matrix we see that modified PC orients many edge marks as an arrow. This explains the high recall and low precision for arrows of modified PC. The algorithm GES is inclined to orient many edge marks as a tail or an arrow.

5.4 Conclusion and discussion

It becomes clear that when the sample size is too low, both modified PC and GES are not suited for this dataset. With a low sample size, the recall for arrows remains rather high, whereas the precision is low. For circles we see that the precision and recall are roughly the same for GES.
When the sample size is sufficiently large, the results are far better. However, the precision of tails and arrows remains around 50 percent. This seems to be rather low. For circles the precision is around 70 percent, however at the expense of a slight decrease of recall. The skeleton results for all simulations except simulation 4 are similar. The ground truth of the five and ten node networks contains 5 and 11 edges respectively.

For modified PC the skeleton is estimated with an error below 20 percent. For GES this percentage varies from 5 percent to 23 percent on the five and ten node networks. For the 50 node network, the ground truth contains 55 edges. Hence, modified PC is off 40 percent and GES is off more than 100 percent. Furthermore, GES overestimates the amount of edges increasingly as the amount of variable increases.
Generalizing the above conclusions to datasets which are of a similar intermediate level, we can use the following guidelines. If the sample size is too low, say, under 500, then the outcomes of the orientations of both GES and modified PC are not accurate enough to be valuable. The skeleton of modified PC and GES, however, are still rather accurately determined if the number of variables is around 10. When the number of variables increases to 50 then even the skeleton results of GES are not accurate enough.
When the sample size is at least 1000 and the number of variables is around ten, then the orientation results become fairly accurate, i.e. in comparison to the results from section 4. One should also assume that GES will overestimate the amount of edges. A possible method to avoid or correct this overestimation is to search for edges that one can remove without decreasing the score too much. Another possibility is to use modified PC as a second source to identify correct edges from overestimated edges. The viability of these methods is a topic for further research.

In order to compare the result of GES with modified PC we transformed the CPDAGs or both the GES outcomes as the ground truth to a PAG. This transformation might have led to erroneous orientations. It might very well be that when we do not transform the ground truth and the GES outcome, that GES performs rather well for this dataset. However, we are interested in the results on real life data. For real life data we do not have the guarantee of causal sufficiency. Hence, we choose for the PAG representation instead of the DAG representation.
Regarding the dataset, we choose to treat each subject separately in order to get averaged results. However, the low sample size of simulations 1, 2 and 4 resulted in very poor results. The results of simulations 6 and 7 indicate that when we consider the dataset of measurements on all subjects, the results will improve.

6 Causal discovery and flux balance analysis

As mentioned, this thesis is centered around exploring the gap between the theoretical side of causal discovery algorithms and applying causal discovery algorithms in practice. In section 4 we considered the performance of causal discovery algorithms on datasets that satisfy the standard requirements. Then in section 5 we took a small step away from these standard requirements. In this section we take an even larger step from the standard requirements by considering a system where not only the distribution of the variables is not a multivariate Gaussian, but also the causal structure is non-standard. We will consider these aspects in more detail later.
The datasets we consider in this section involve so-called metabolic networks. Metabolic networks are sets of chemical reactions in a cell, [15]. The molecules involved in these reactions are referred to as metabolites. These metabolic networks can be studied in silico with so-called flux balance analysis, [18]. Later in this section we will consider flux balance in more detail. It suffices to say at this point that flux balance analysis is a way to simulate metabolic networks and to measure effects on the system by external influences. Many of the utilities of flux balance analysis are beyond the scope of this text. In this thesis flux balance analysis is used to generate a dataset on which to apply causal discovery algorithms. This data generation is done with the so-called COBRA-toolbox, [2, 22], and it concerns the metabolic network of the Escherichia coli bacteria.
This section is outlined as follows. First, we consecutively consider the basics of flux balance analysis, the causal interpretation of metabolic network and the process of sampling the data. Then we evaluate modified PC and GES on various substructures of the metabolic network of the E. Coli bacteria. These substructures are chosen on the basis of having different structural properties. First we apply modified PC and GES to small substructures consisting of two or three reactions. Seeing how modified PC and GES handle these small structures may illuminate the behavior of modified PC and GES on larger structures. Moreover, if modified PC and GES return very unlikely results on these small structures then it is improbable that they return meaningful results on larger structures.
Having considered these small substructures we consider three larger structures. The first is the so-called Citric Acid Cycle (CAC) and, as the name suggest, it is shaped as a cycle. The second structure we consider is not a cycle but a sequence. This structure is concerned with so-called Glycolysis (Gly). Details of glycolysis are beyond the scope of this text. The last structure we consider is the Nitrogen Metabolism (NM). This is neither a cycle nor a line but a highly connected graph on fewer nodes than CAC and Gly.
The purpose of considering these different structures is in twofold. Firstly, the variety of structures enables us to consider in which circumstances modified PC and GES thrive and in which they do not. For instance, it might be the case that they are suited for the linear sequence of Gly but not for the cycle structure of CAC. Secondly, considering different structures allows us to generalize performance results by looking at the common properties of the structures.

6.1 Flux balance analysis

The following is based on [18]. Flux balance analysis (FBA) is used to simulate metabolic systems and to predict its behavior under specified circumstances. There are two main facets of FBA, *constraints* and *optimization*. One first specifies the constraints regarding the reactions of the metabolic network and then one determines the optimal production of certain metabolites under said constraints.
In this thesis we consider the metabolic network of Escherichia coli bacteria as described in [17]. Details regarding this network are acquired via its so-called BiGG model, [12]. These models are centralized repositories for data regarding metabolic networks and they have an implemented visualization tool.
The description of a metabolic network consists of metabolites, chemical reactions involving these metabolites and certain bounds regarding the reactions. Metabolites are resources of the cell and reactions convert one group of metabolites into another.
Let us first consider how the constraints for FBA are determined. The reactions of a metabolic network

can be written as equations. For example, the reaction fumarase[5] from the metabolic system of the E. Coli bacteria is written as:

$$FUM_c + H_2O_c \leftrightarrow MAL\_L_c.$$

Here the bidirected arrow indicates that the reaction is reversible. These reactions can be represented by an $m \times n$-matrix S, called the **stoichiometric matrix**. Here m denotes the number of metabolites and n denotes the number of reactions. This matrix is constructed as follows. Each of the irreversible reactions has an orientation (i.e. a clear distinction between the reactants and the products) and each of the reversible reactions can be equipped with such an orientation. Let a_{ij} be the coefficient of metabolite i in reaction j. Then we define S_{ij} to be $-a_{ij}$ if metabolite i is a reactant and $S_{ij} := a_{ij}$ if it is one of the products.
For example, consider the following set of fictitious reactions:

$$2A + 3B \to 1C + 3D$$
$$10A + C \to D$$

There are 4 metabolites and 2 reactions. Hence, the stoichiometric matrix is a 4×2 matrix. If we let A correspond to 1 and B to 2, etc., then the stoichiometric matrix becomes:

$$\begin{pmatrix} -2 & -10 \\ -3 & 0 \\ 1 & -1 \\ 3 & 1 \end{pmatrix}$$

Suppose one has n reactions and m metabolites and suppose one were to write the concentrations of each metabolite in an m dimensional vector C. Then the concentrations of the metabolites are related to the stoichiometric matrix via the equation

$$\frac{dC}{dt} = Sv,$$

where v indicates the speed of the reactions. The i-th component of vector v is the flux of the i-th reaction. Flux is measured in mmol per gram dry weight per hour (mmol/gDW/h). These *flux vectors* are studied in flux balance analysis.
One part of the forementioned constraints used in flux balance analysis is generated by the steady state assumption:

$$Sv = 0.$$

In flux balance analysis one only studies the flux vectors v for which $Sv = 0$. The other part of the constraints are bounds on the fluxes of every reaction. For instance if a reaction is irreversible, then this is expressed by requiring the flux of that reaction to be positive. The space consisting of flux vectors satisfying both parts of the constraints is referred to as the *solution space*.
Regarding the optimization part of FBA, note that the constraints for FBA are all linear equations. Now one can define an objective function to be minimized or maximized, i.e. a function $f(v)$ of the flux vectors for which one wants a flux vector v_0 from the solution space for which $f(v)$ is maximal. This set up is suited for linear programming methods, i.e. one can minimize or maximize the objective function of the metabolic network under the constraints by using linear programming methods. In particular, one can predict how the metabolic system behaves under interventions of the network, which is the main goal of flux balance analysis. These interventions can be translated in the second part of the constraints, i.e. the bounds on the fluxes of each reaction. For instance, one can postulate that a certain reaction does not occur by demanding the flux of that reaction to be 0. Hence by applying methods from linear programming one can, for instance, determine the maximal growth of the E. Coli bacteria under anaerobic or aerobic circumstances.
The COBRA-toolbox is a Matlab toolbox specifically designed for flux balance analysis. Although this

[5]The details of this reaction are beyond the scope of this text, however, the interested reader is referred to [17] for more information regarding the specific reactions of the E. Coli metabolism.

toolbox has many functions, we only use it to generate the data.
This concludes the description of flux balance analysis and the interested reader is referred to [18] for more details on FBA. Although in the above description only the metabolic network of the E. Coli bacteria is mentioned, flux balance analysis can be applied to all sorts of metabolic networks.

6.2 Causal interpretation

Although FBA is mainly used for prediction, we only use it to generate a dataset of flux vectors without having a function to minimize or maximize. This is done by sampling vectors uniformly distributed over the solution space. The procedure for sampling is an interesting algorithm in its own right and we will discuss it in section 6.3.
We apply modified PC and GES to this dataset with the goal of retrieving causal relations between the reactions. The data consists of flux vectors and hence, one can only find causal relations between the reactions of the metabolic network. However, as mentioned in section 3.3, we need a ground truth in order to evaluate modified PC and GES on this dataset. This raises the question: when are two reactions causally related? One might argue that two reactions are related iff they share a metabolite which is pivotal for one of the reactions. However, we cannot a priori know which metabolite is pivotal in which reaction. In order to differentiate pivotal metabolites from non-pivotal ones, we consider the pathway network displaying the metabolic system of the Escherichia coli bacteria. This network is determined in [17]. In this network, certain metabolites are present at more than one place. This seems to indicate that such a metabolite is not pivotal for either of these reactions. Hence, we use the pathway network to identify which reactions are causally related.
However, if two reactions are causally related, it is not obvious what the directionality is, i.e. which reaction is the cause and which one the effect. This is illustrated by the following example. Assume we have the following reactions:

$$A \to B, \tag{4}$$
$$B \to C. \tag{5}$$

In this case is it not clear whether an increased flux of reaction 4 causes an increased flux of reaction 5 or vice versa.
Due to this complication we refrain from determining a ground truth with directionality and we only evaluate the algorithms by comparing the outcomes with the pathway network. As a consequence, the focus of this section lies on the skeletons outputted by modified PC and GES.
The forementioned pathway network can be seen in figure 32, for zoom options and additional feature, the reader is referred to the BiGG model, [12, 17].
When we evaluate GES and modified PC to subsystems of this network, we are only interested in the structure of the network and not the precise names of the involved metabolites. Hence, we consider a schematic depiction of that subsystem, disregarding many details (for instance the names of the metabolites). However, there are metabolites that occur in several places in the subsystem. Details on these metabolites and the reactions in which they occur are considered seperately. N.B. in the pathway network in figure 32 and in the following schematic depiction of its subsystems, nodes represent metabolites and arrows represent reactions. It is also worth mentioning that the COBRA model of the E. Coli metabolic system works with abbreviations instead of the full names of the reactions and metabolites. The full names can be found at its BiGG model, [12, 17].

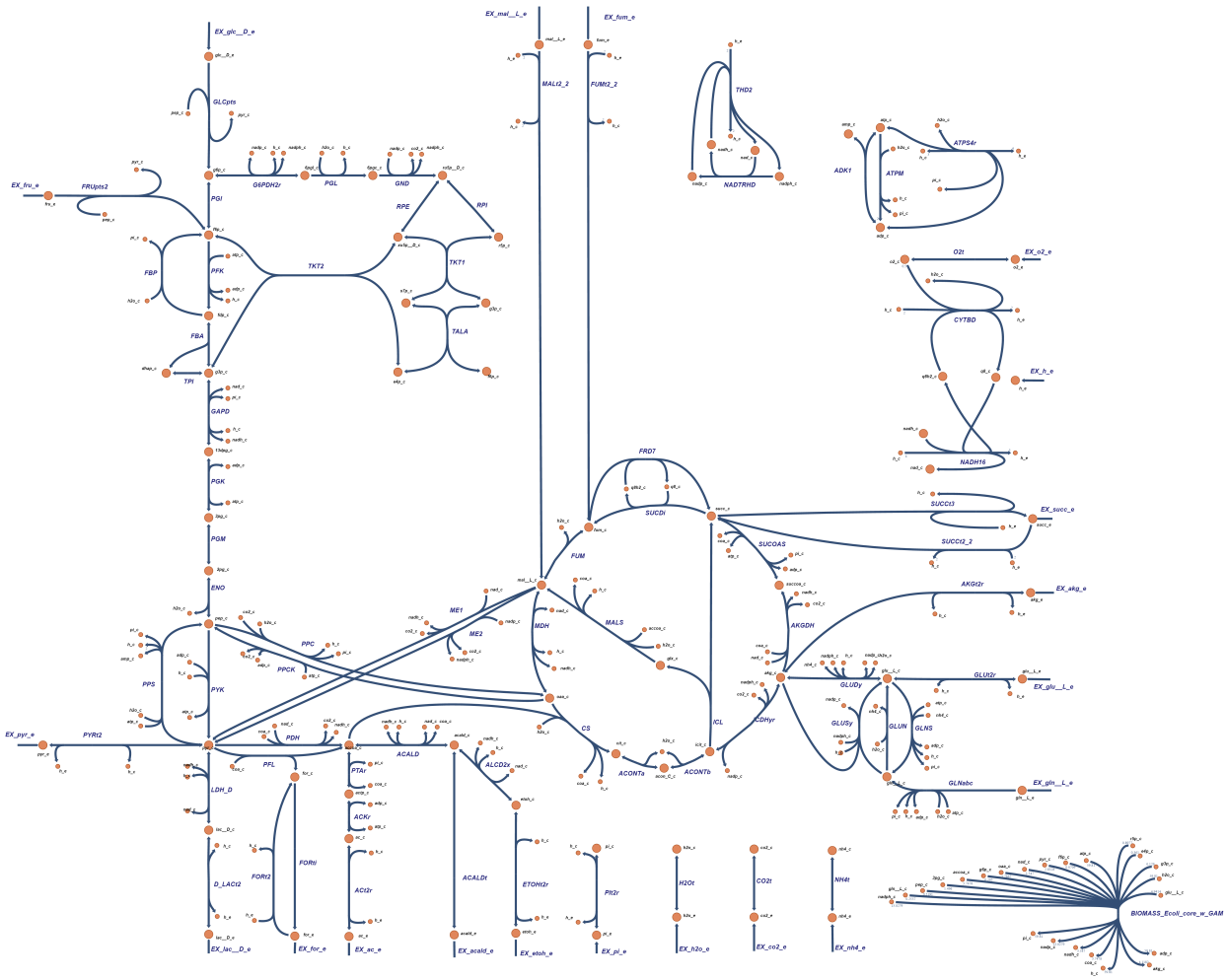


Figure 32: Ecoli pathway network. One can detect various cycles in this pathway graph and many reversible reactions. These cycles and bidirected arrows form a point of interest for the causal discovery algorithms, i.e. we are interested how the algorithms interpret this circularity. Again, note that for this dataset the causal discovery algorithms attempt to find causal relations between reactions, not between the metabolites.

6.3 Sampling the solution space

The data is generated by sampling vectors uniformly distributed over the solution space. Sampling the solution space is done by the so-called hit and run algorithm. Suppose we want to create a set of N samples with k steps (the notion step will be made precise). Then the sampling function from the COBRA toolbox, GpSampler, first generates N points in the middle of the solution space. Then these points are relocated through the solution space in several steps. Figure 33 gives an example of such a step.

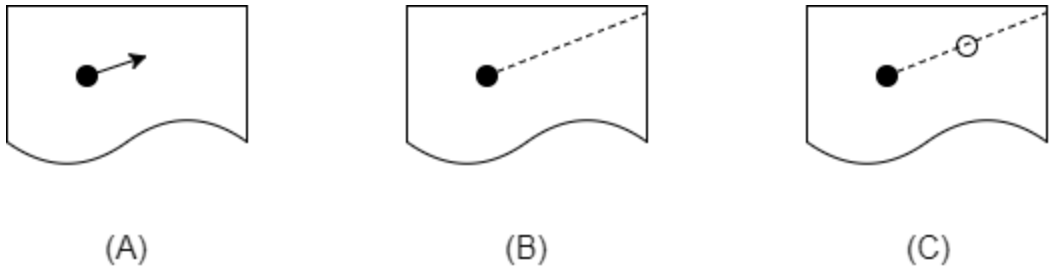


Figure 33: A step of the hit and run algorithm. (A) Given a point in the solution space, one chooses a random direction. (B) Then the line from the point to the edge of the solution space is determined and (C) a new point is chosen uniformly on that line.

The operation 'step' is given in pseudocode in algorithm 7.

input : Solution space U and $X \in U$
output: Point $Y \in U$
1 old_point$\leftarrow X$;
2 dir$\leftarrow$ random direction;
3 line$\leftarrow$ line in direction dir starting in old_point ending on the edge of the solution space;
4 new_point$\leftarrow$ random point on line;
5 **return** *new_point*

Algorithm 7: Hit and run step.

The idea is that when the number of steps increases, the set of points will be distributed more uniformly across the solution space. The dataset we use consists of 5000 samples, each of which is generated in 2000 steps. This means that initially the algorithm starts with 5000 points in the center of the solution space, $X_0^1, X_0^2, ..., X_0^{5000}$ and that for each of these points one creates a sequence $X_0^i, X_1^i, X_2^i, ..., X_{2000}^i$ such that for each j between 0 and 1999 we have that X_{j+1}^i is obtained by executing the operation described in algorithm 7 on X_j^i. The resulting dataset consists of the points $X_{2000}^1, X_{2000}^2, ..., X_{2000}^{5000}$. This concludes the description of the generation of the data.

6.4 Small structures of the network

As mentioned, for evaluating modified PC and GES datasets acquired through flux balance analysis, we start by considering small structures of the metabolic network of the E. Coli bacteria. Regarding the used threshold of modified PC, we choose the threshold to be 0.025 based on the results of section 4. We use the same threshold for each of the following substructures.
The first of these small structures consists of the reactions G6PDH2r, PGL and GND. We call this structure the GPG-structure. The corresponding part of the pathway network is depicted in figure 34.

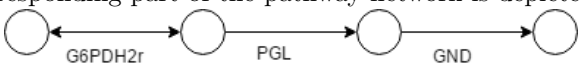


Figure 34: Schematic depiction of the GPG-structure.

Note that the nodes represent sets of metabolites and the (bidirected) arrows represent reactions. This subsystem consists of two irreversible reactions and one reversible reaction.
The following table indicates which other mutual metabolites occur and in which reactions.

Metabolites	Reactions
h_c	G6PDH2r, PGL
nadp_c	G6PDH2r, GND
napdh_c	G6PDH2r, GND

Figure 35: Table with the mutual metabolites.

Upon investigation we find that the data satisfies the following equalities:

$$\text{Flux(G6PDH2r)} = \text{Flux(GND)} = \text{Flux(PGL)}.$$

Applying GES to the dataset yields the results depicted in figure 36.

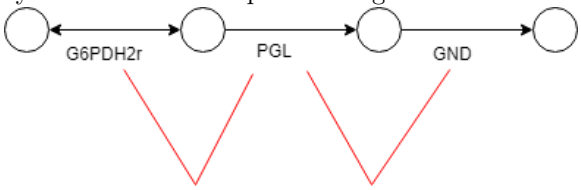


Figure 36: Result of applying GES to the GPG-structure.

It stands out that there is no edge between G6PDH2r and GND. This is remarkable since, objectively, if the fluxes of these reactions are equal we cannot differentiate between cause and effect. The only explanation is that the decrease in the complexity of the model outweighs the decrease in explanatory power of the model when the edge between G6PDH2r and GND is omitted.

The result of applying modified PC with threshold 0.025 to this dataset is depicted in figure 37.

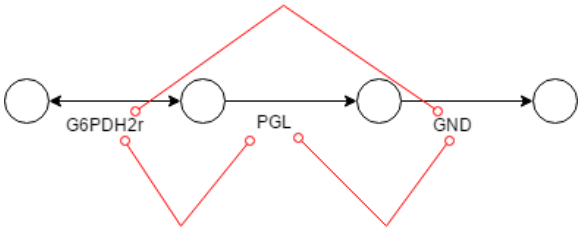


Figure 37: Result of applying modified PC with threshold 0.025 to the GPG-structure.

These results show that modified PC could not render any pair of variables conditionally independent. Hence, it could not orient any of the edges. This contradicts the theory since, counterintuitive as it may sound, PGL and GND should be rendered conditionally independent given G6PDH2r. This can be concluded from the following lemma.

Lemma 6.4.1
Let X, Y and Z be stochasts such that $X = Y = Z$, then X and Y are independent given Z

Proof: The following statements holds

$$\begin{aligned}
&p(X = x, Y = y|Z = z) = 1 \text{ iff } x = y = z \\
&p(X = x, Y = y|Z = z) = 0 \text{ iff } x \neq z \text{ or } x \neq z \\
&p(X = x|Z = z) = 1 \text{ iff } x = z,\ p(Y = y|Z = z) = 1 \text{ iff } y = z \\
&p(X = x|Z = z) = 0 \text{ iff } x \neq z,\ p(Y = y|Z = z) = 0 \text{ iff } y \neq z.
\end{aligned}$$

Hence, if $x = y = z$ then

$$p(X = x, Y = y|Z = z) = 1 = 1 \cdot 1 = p(X = x|Z = z)p(Y = y|Z = z).$$

If $x \neq z$ or $y \neq z$, say without loss of generality $x \neq z$, then we have that

$$p(X = x, Y = y|Z = z) = 0 = 0 \cdot p(Y = y|Z = z) = p(X = x|Z = z)p(Y = y|Z = z).$$

This proves the claimed conditional independence.

□

Therefore, since

$$\text{Flux(G6PDH2r)} = \text{Flux(GND)} = \text{Flux(PGL)},$$

we have that PGL and GND are independent given G6PDH2r and, hence, it is remarkable that no pair of variables is rendered independent conditioning on the remaining variable by modified PC. Upon further examination we find that in the software R-package *pcalg*, some rounding off occurs. Due to this rounding off, the entries of the correlation matrix which is used for the Independence tests are not precisely one. This results in the apparent dependencies.
Regarding the orientations, the result of GES allows that the edges can be oriented both as a chain from left to right or as its reversed counterpart, but also as the structure G6PDH2r←PGL→GND.
However, the orientation results of modified PC are inconclusive in a sense that the algorithm was not able to orient any edge. This result is consistent with the intuition that one cannot differentiate between cause and effect since the three fluxes are equal.

The second substructure is a small cycle consisting of the reactions ADK1 and ATPM as depicted in figure 38. We call this structure the AA-structure. Ideally we consider a cycle consisting of more than two reactions. However, in this specific model, there is no small cycle of more than two reactions. Therefore, we consider the AA-structure. The bidirected arrow is omitted from consideration.

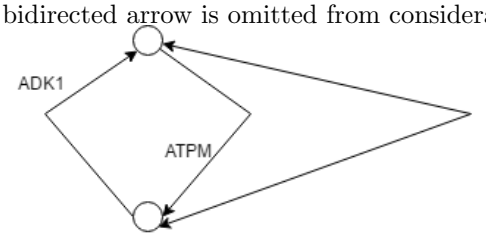


Figure 38: Schematic depiction of the AA-structure.

The result of applying modified PC with threshold 0.025 to the dataset of the AA-structure is depicted in 39.

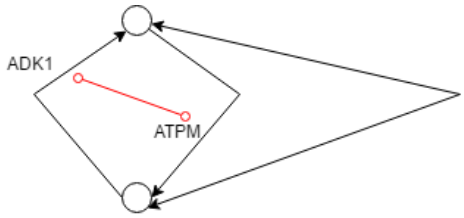


Figure 39: Outcome of applying modified PC with threshold 0.025 to the AA-structure.

Modified PC returns one edge with circle edge ends. One would expect a connection between ADK1 and ATPM, since they share a metabolite in the pathway network. Furthermore, this connection should not be oriented since there are no orientation rules involving precisely one arrow. Therefore, the outcome of modified PC is consistent with our expectations. Figure 40 displays the result of applying GES to this dataset.

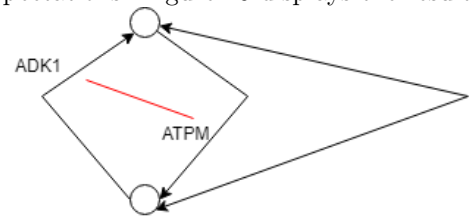


Figure 40: Outcome of applying GES to the AA-structure.

In theory, GES was able to return two possible models, either one with no edge or one with precisely one edge. Since they share a metabolite in the pathway network and GES identified both reactions as causally related, GES returned the correct model.

6.5 The citric acid cycle

Having evaluated modified PC and GES on the small structures we now consider the first of the large substructures, the citric acid cycle. The citric acid cycle is schematically depicted in figure 6.5. This is based on the pathway network from [17], more details can be found there.

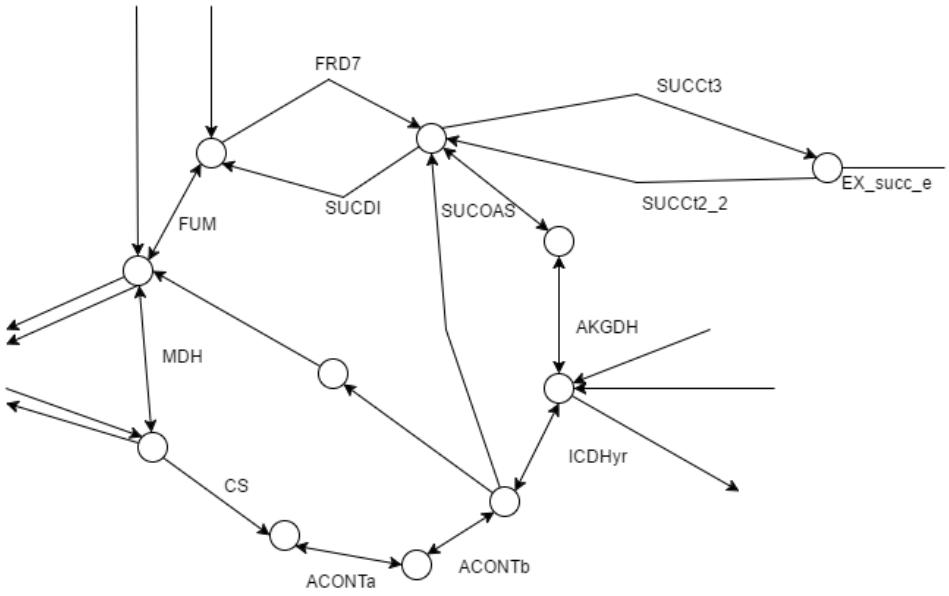


Figure 41: Schematic depiction of the citric acid cycle.

The following table indicates the remaining mutual metabolites and the reactions in which they occur.

Metabolites	Reactions
co2_c	AKGDH, ICDHyr
h2o_c	ACONTa, ACONTb, CS, FUM
h_c	CS, MDH
nad_c	AKGDH, MDH
nadh_c	AKGDH, MDH

Figure 42: Table with the mutual metabolites in the CAC structure.

6.5.1 Modified PC and CAC: part 1

Applying modified PC with a threshold of 0.025 to the dataset yields the following result.

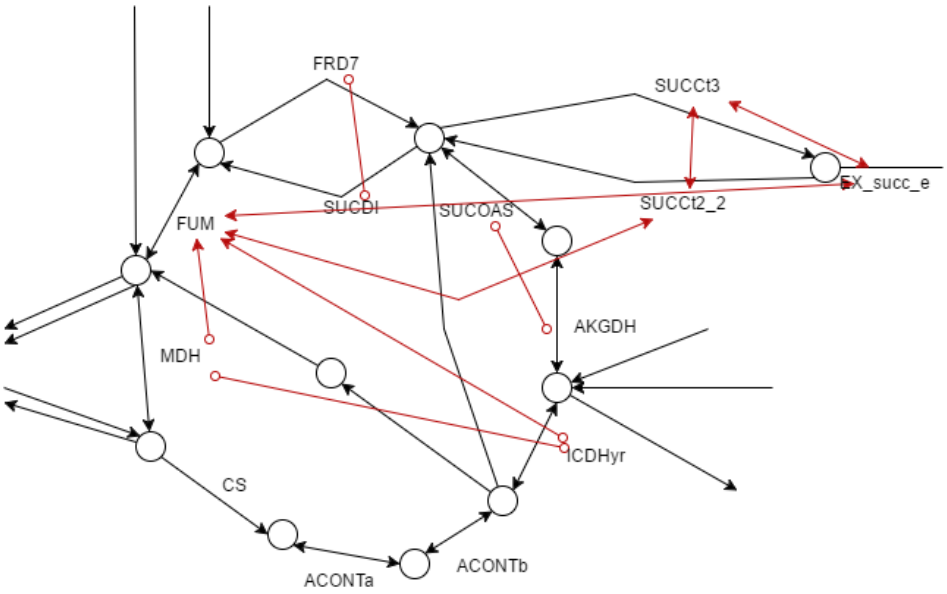


Figure 43: Outcome of applying modified PC with threshold 0.025 to the CAC.

This outcome displays a problem that, as we prove later, is a structural problem regarding deterministic relations between variables. Note that CS, ACONTa and ACONTb are not connected with any reaction. Furthermore, SUCOAS and AKGDH are connected with each other but not with other reactions.
Investigation of the dataset shows that the following equations hold:

$$\text{Flux(ACONTa)} = \text{Flux(ACONTb)=Flux(CS)},$$
$$\text{Flux(AKGDH)} = -\text{Flux(SUCOAS)}.$$

With lemma 6.4.1 one can conclude that this explains why CS, ACONTa and ACONTb are pairwise non adjacent.[6] The following lemma explains why the reactions SUCOAS and AKGDH and the reactions CS, ACONTa and ACONTb are not connected with any of the remaining reactions.

Lemma 6.5.1
Let X, Y, Z be stochasts such that $Y = Z$. Then $X \perp\!\!\!\perp Y|Z$.

Proof: For x, y and z we have that if $y = z$, then

$$p(X = x, Y = y|Z = z) = p(X = x|Z = z) = p(X = x|Z = z)p(Y = y|Z = z),$$

since $p(Y = y|Z = z) = 1$. If $y \neq z$, then we have that

$$0 = p(X = x, Y = y|Z = z) = p(X = x|Z = z) \cdot 0 = p(X = x|Z = z)p(Y = y|Z = z).$$

Hence, $X \perp\!\!\!\perp Y|Z$.

□

This shows that when two variables X and Y are equal, then one cannot derive causal relation between X and a variable Z which is not equal to either X or Y. Furthermore, the same conclusion holds whenever two variables are almost equal.

[6] Lemma 6.4.1 only shows that $X \perp\!\!\!\perp Y|Z$ if $X = Y = Z$. A similar argument can be made when X and Y are functions of Z. In particular, when $X = Y = -Z$.

6.5.2 Grouping of variables

In the above we recognized the following structural problem.

> **Problem**: When multiple variables are correlated too strongly, modified PC cannot causally relate these variables to other variables.

This can be solved by grouping together the variables which are too correlated. However, this raises two difficulties. First of all, when are two variables 'too' correlated? That is, when is the correlation between two variables strong enough to affect the connections with the remaining variables? At this point we skip this question and only consider variables that are functionally related, i.e. when a variable X can be written as a function of Y. Should this prove to be fruitful, then answering the now skipped question is a topic for future research. Note that functionally related variables are correlated enough to be grouped together.
The second difficulty is that finding out which variables are functionally related is not a trivial procedure. One has to account for all sorts of possible functional relation. However, we limit this search and define an equivalence relation on the variables where $X \sim Y$ whenever either $X = Y$ or $X = -Y$. We group the variables that are equivalent.
One has to check, however, whether grouping the variables and in the process of doing so, the erasal of information, is harmful for discovering the causal relations. I claim however that this is not the case. Whenever two variables, for example, are equal then one cannot differentiate between cause and effect. Grouping respects this and then allows causal relations to be found between the group of identical variables and the remaining variables. Hence, grouping can only lead to additional information.
Concretely, in the situation of CAC the grouping amounts to grouping ACONTa, ACONTb and CS and grouping AKGDH and SUCOAS together. This is done by removing the variables ACONTa, ACONTb and SUCOAS from consideration. In the graph this is represented by displaying grouped variables with a colored area. In this fashion figure 6.5 becomes figure 44.

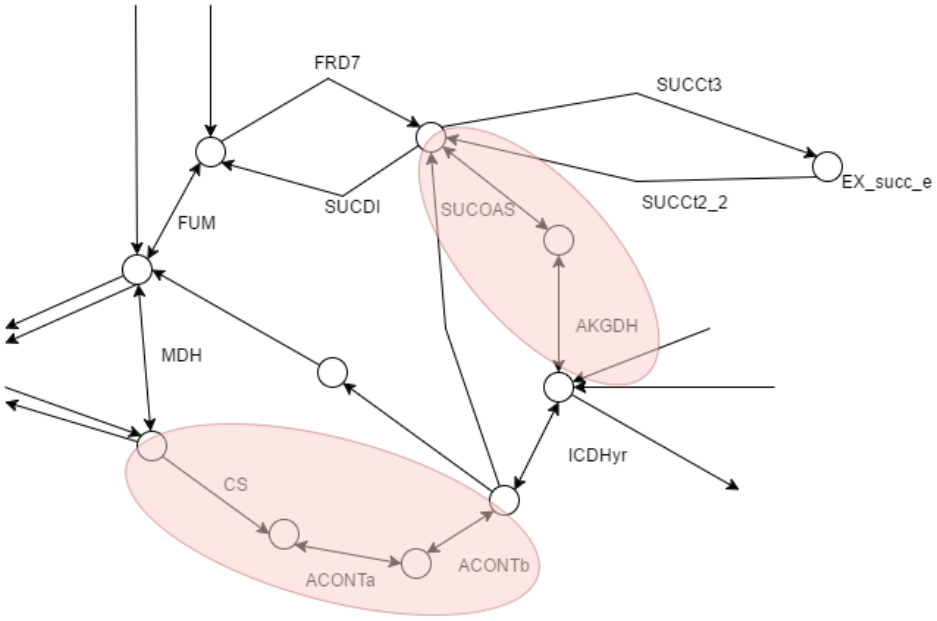


Figure 44: Schematic depiction of CAC with colored areas to indicate groupings.

6.5.3 Modified PC and CAC: part 2

Applying modified PC to the grouped dataset of CAC yields the following result.

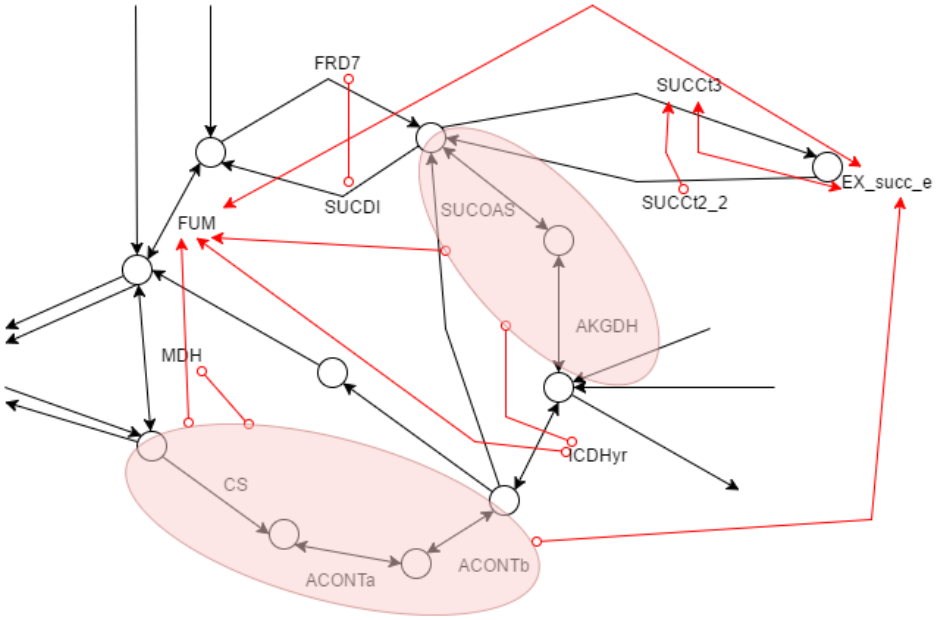


Figure 45: Modified PC applied to the grouped CAC.

The problems that arised from the functional dependencies of the CS group and of the SUCOAS grouped are indeed fixed. Edges have appeared between the CS group and MDH and between the AKGDH group and ICDHyr. In comparison to figure 43, the result obtained from adding the groupings is more desirable. Hence, from now on we first group reactions that have the same or opposite fluxes.

Improvement: before applying the causal discovery algorithm, group the variables appropriately.

Figure 45 shows that some of the reactions adjacent in the modified PC output are indeed close to each other in the pathway network (e.g. FRD7 and SUCDI). However, the results from figure 45 also display several peculiarities. They can be categorized in two classes.

1. Reactions that are adjacent in the modified PC output but remote in the pathway network
2. Reactions that are not adjacent in the modified PC output, despite having nodes in common in the pathway network.

Examples of the first category include the pairs (FUM, EX_succ_e) and (CS group, EX_succ_e). The pairs (FRD7, FUM), (SUCDI, FUM) and (SUCCt3, SUCOAS group) are examples of the second category. For the second category, an explanation can be found in lemma 6.5.1. For instance, FRD7 and SUCDI are strongly correlated, as can be seen from the scatterplot displayed in figure 46.

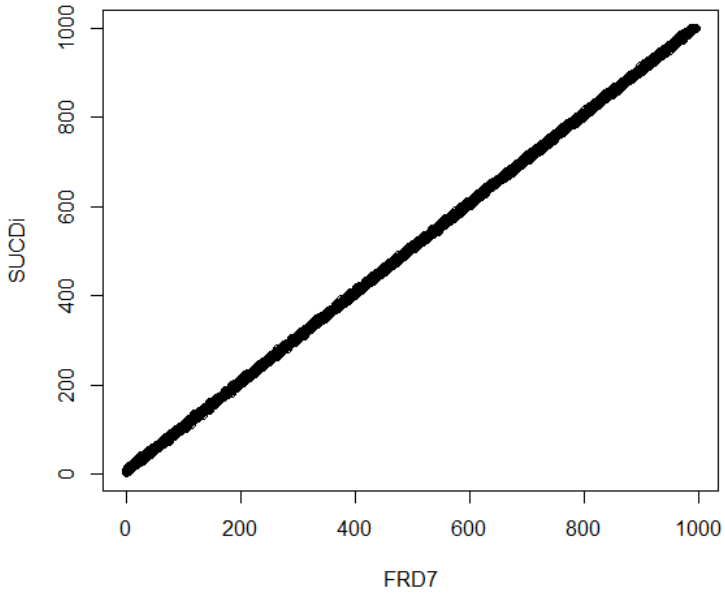


Figure 46: Scatterplot of the reactions FRD7 and SUCDi.

As mentioned, such a correlation causes FRD7 and SUCDi to be connected with each other but not with any of the remaining reactions.

For the first category an explanation can be found in deterministic relations, i.e. functional relations between variables without noise components. Several deterministic connections are present between the reactions due to the steady state assumption. In order to illustrate the effect of these relations, consider the following graph, where several arrows are given letters.

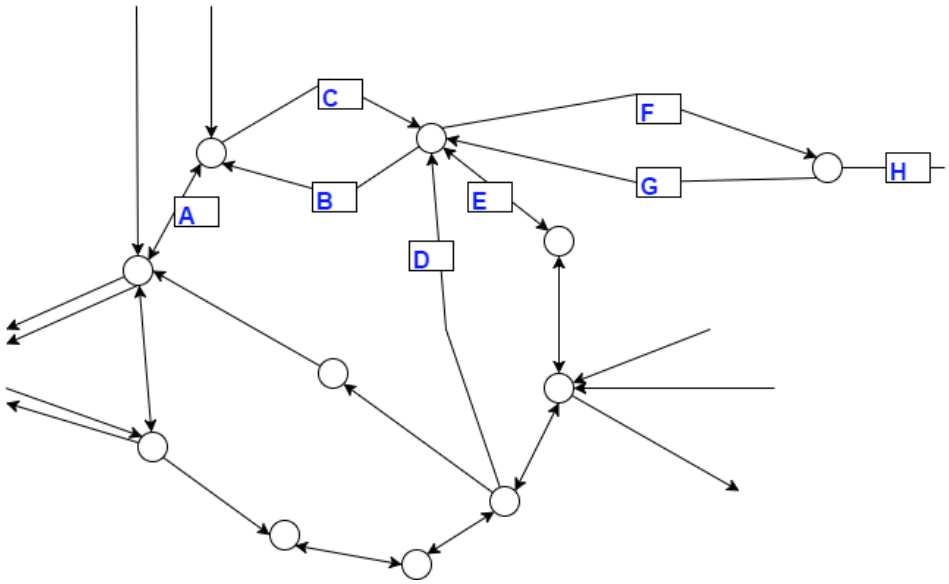


Figure 47: The citric acid cycle with letters.

The dataset shows that the following relations hold:

$$A = G - F + E + D$$
$$A \approx C - B$$
$$H = F - G$$

Hence, it becomes clear that since

$$\text{EX\_succ\_e} = H = F - G$$

and since $F - G$ is in the decomposition of A, that EX_succ_e is related to FUM. Similarly, the SUCOAS group is a significant part of the decomposition of A and hence modified PC found a connection between FUM and SUCOAS.
This shows that in this situation, modified PC does find reasonable connections, but that they do not represent the chain of the pathway network and hence are not useful.

6.5.4 GES and CAC

We continue with evaluating GES. As mentioned before, the ungrouped dataset gives no extra information. Hence, we apply GES to the grouped dataset. The results can be found in figure 48.

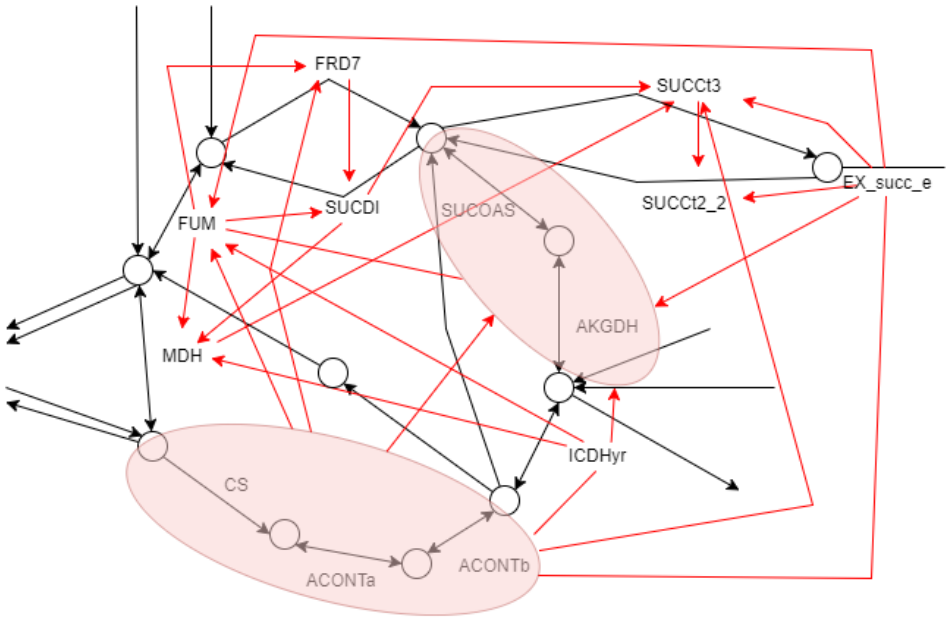


Figure 48: Outcome of applying GES to the CAC with groups.

The outcome contains a high amount of edges, i.e. the average degree of each reaction is relatively high. This might be the result of the deterministic relations between the reactions. Since most of the edges from the modified PC outcome are also found in figure 48, this seems very probable. The high amount of edges make the GES outcome rather uninformative. One is interested in the relevant connections. Apparently the complexity of the model is not reduced enough by the BIC score. This suggests that in order to decrease the amount of irrelevant connections, one should increase the penalty factor of BIC. Indeed, raising the penalty term leads to fewer edges in the GES outcome. For instance, multiplying the penalty factor by 40 results in the graph from figure 49.

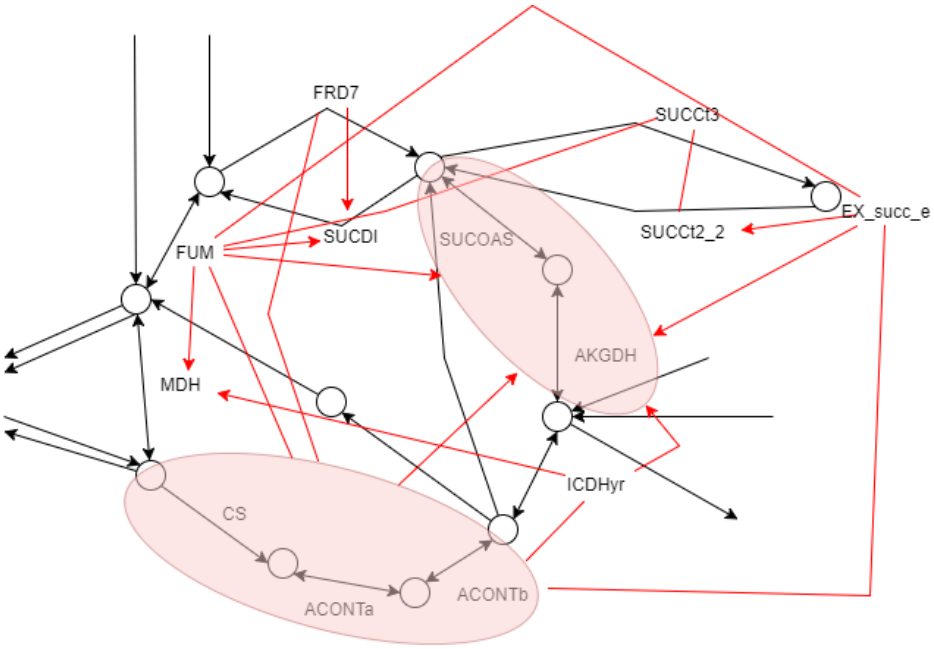


Figure 49: Outcome of applying GES with an increased penalty term, i.e. multiplied by 40, to the grouped CAC structure.

The amount of edges is indeed reduced and now the outcome of GES is similar to the outcome of modified PC. Moreover, GES exhibits similar peculiarities as modified PC on this dataset. That is, for instance, FRD7 and FUM are not connected by an edge in the GES outcome, whereas they are adjacent in the pathway network. Furthermore, both the CS group and EX_succ_e are connected by an edge as are FUM and EX_succ_e. A significant difference between the GES outcome and modified PC is that whereas FRD7 and SUCDI were only connected to each other in the outcome of modified PC, SUCDI is connected to FUM and FRD7 is connected to the CS group in the GES outcome. This shows that although modified PC cannot causally relate variables that are too correlated, GES, in fact, can. Hence, by using both GES and modified PC one can detect pairs of variables that are too correlated for modified PC so that they can become grouped.
We conclude the evaluation of GES on CAC by remarking that raising the penalty improved the result of GES and that we will discuss this in more detail in section 8.4.

6.6 Glycolysis

In the previously subsection we considered the citric acid cycle. This substructure was in the shape of a cycle. In this subsection we consider the substructure of glycolysis (Gly), depicted in figure 50. This structure is roughly shaped as a straight line.

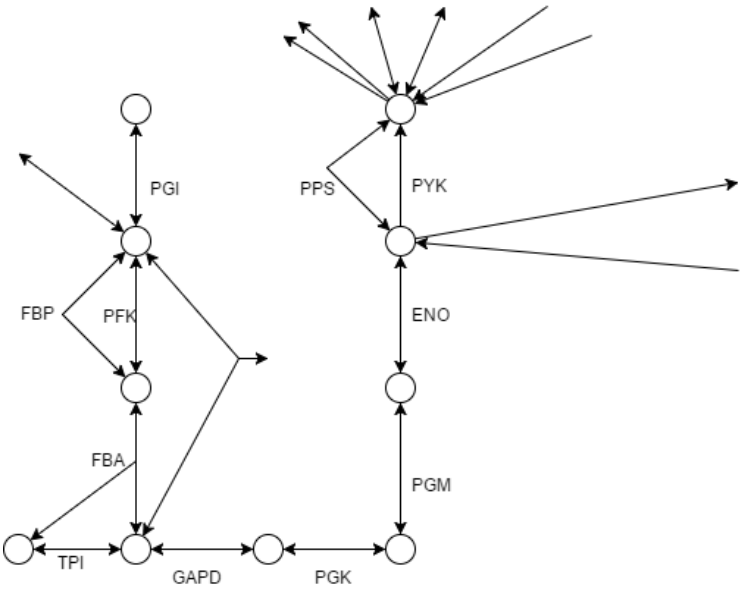


Figure 50: Schematic depiction of the substructure of glycolysis.

The remaining details on shared metabolites is given in the following table.

Metabolites	Reactions
atp_c	PFK, PGK, PPS, PYK
adp_c	PFK, PGK, PYK
h2o_c	ENO, FBP, PPS
h_c	GAPD, PFK, PPS, PYK
pi_c	FBP, GAPD, PPS

Figure 51: Table with the mutual metabolites in the Glycolysis structure.

The datasets exhibit the following equalities.

$$\text{Flux(ENO)}=\text{-Flux(PGM)}$$
$$\text{Flux(GAPD)}=\text{-Flux(PGK)}$$
$$\text{Flux(FBA)}=\text{Flux(TPI)}$$

We group the dataset accordingly by removing PGK, PGM and TPI from the dataset. The graph with groupings is depicted in figure 52.

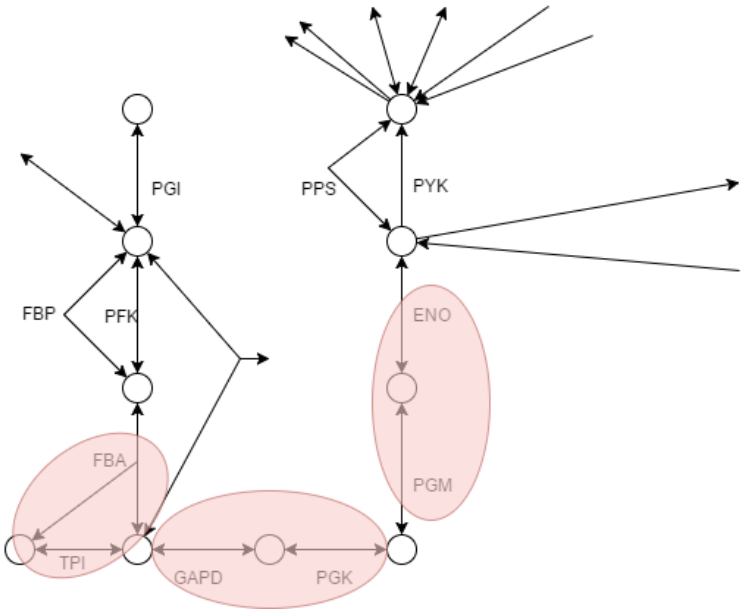


Figure 52: Schematic depiction of glycolysis with groupings.

6.6.1 Modified PC and Glycolysis

When we apply modified PC with threshold 0.025 to the grouped data we get the results depicted in figure 53.

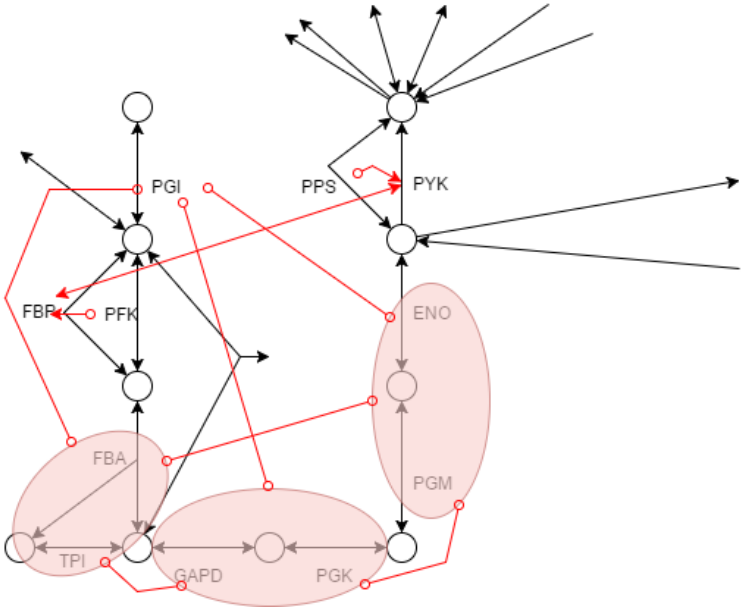


Figure 53: Outcome of applying modified PC with threshold 0.025 to the grouped glycolysis structure.

The lines of the outcome can be categorized into two classes. One class consists of the complete graph on the groups of FBA, GAPD and ENO and the reaction PGI. The second class is made up from FBP, PFK, PPS and PYK.
The complete part is not oriented in the sense that no changes are made to the orientation after the identification of the skeleton. Regarding the other class, four arrowheads have emerged during the orientation of

the v-structures.
Note that modified PC returned several edges between remote reactions. For instance FBP and PYK are connected and so are PGI and the GAPD group. These edges seem to contradict the pathway network. However, they can be explained by noting that on the one hand the substructure is governed by deterministic relations (similarly as in the case of CAC) and on the other hand, the substructure is in the shape of a line. Hence, it is hard to differentiate between adjacent reactions and distant reactions. In the next subsection we consider a structure which it not in the shape of a line and which is more dense. If the conclusion that deterministic relations and the shape of a line do in fact result in distant reactions being identified as adjacent by modified PC, then that problem should not occur with the next substructure.

6.6.2 GES and Glycolysis

The result of applying GES to the grouped dataset is shown in figure 54.

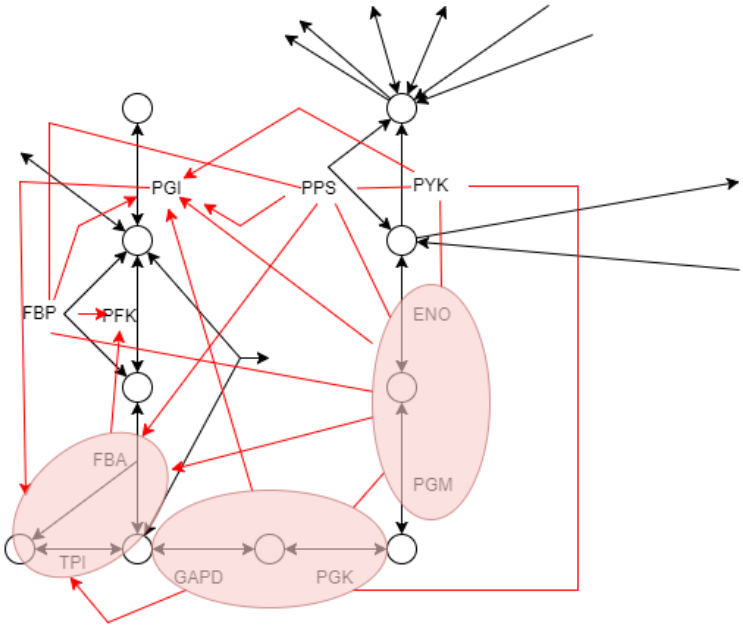


Figure 54: Outcome of GES on glycolysis with groups.

The number of edges in this outcome is significantly higher than in the outcome of modified PC. There are many edges between adjacent reactions. However, there are also many edges between separated reactions. When we compare the skeleton of this graph to the skeleton of the outcome of modified PC we see that there is no edge between PYK and FBP in this graph, but the other adjacencies found in the modified PC outcome are also found in this graph.
The overestimation of edges again might be due to deterministic relations between variables and to a penalty term in BIC that is too low. Indeed, raising the penalty term decreases the amount of edges. For example, multiplying the penalty term by 40 results in the outcome depicted in figure 55.

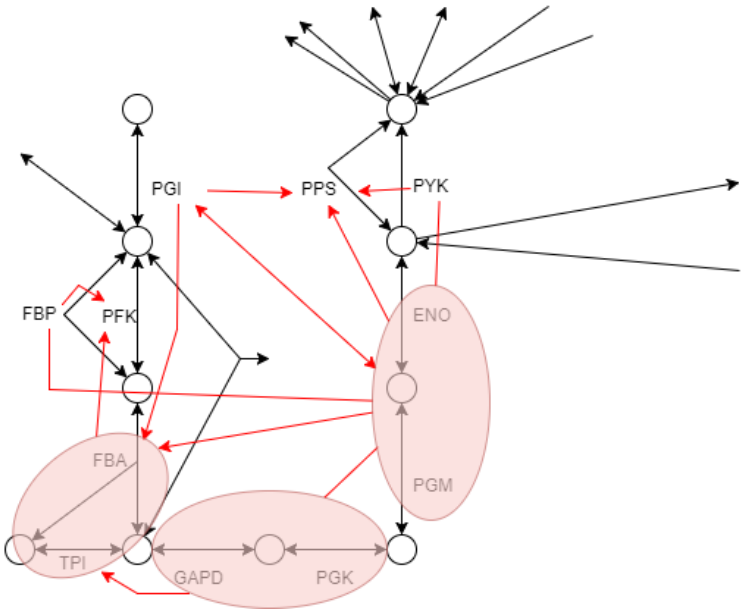


Figure 55: Outcome of applying GES with an increased penalty term, i.e. multiplied by 40, to the grouped Glycolysis structure.

The amount of edges has been reduced. Similar structures as in the outcome of modified PC arise, such as class of the FBA group, the GAPD group, the ENO group and the reaction PGI. However, several remote nodes have also been connected. Nonetheless, increasing the penalty term improved the outcome of GES.

6.7 Nitrogen Metabolism

Lastly we consider the nitrogen metabolism (NM) subsystem of the E. Coli model. This system is adjacent to the citric acid cycle in the pathway network of figure 32. It consists of eight reactions. The main reason for considering this structure is that both CAC and Gly are roughly in the shape of a sequence. In CAC this sequence is a cycle and with Gly this sequence is a line with a beginning and an end. The structure NM, on the other hand, is not a sequence, as can be seen in figure 56. If some of the remote connections are indeed caused by the combination of deterministic relations and the fact that the structure is a sequence, then applying modified PC to NM should not connect remote reactions with an edge.

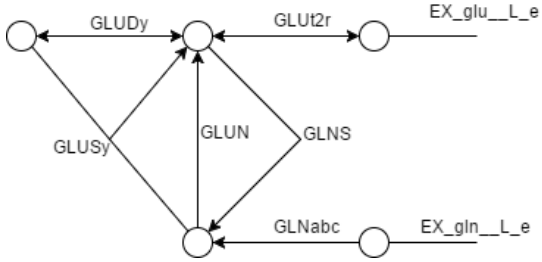


Figure 56: Schematic depiction of the structure of nitrogen metabolism.

The remaining information regarding the mutual metabolites can be found in the following table.

Metabolites	Reactions
atp_c	GLNabc, GLNS
adp_c	GLNabc, GLNS
h2o_c	GLNabc, GLUN
h_c	GLNabc, GLNS, GLUDy, GLUSy, GLUt2r
nadph_c	GLUDy, GLUSy
pi_c	GLNabc, GLNS

Figure 57: Table with the mutual metabolites in the NM structure.

Examination of the data shows that

$$\text{Flux(GLUt2r)} = -\text{Flux(EX\_glu\_\_L\_e)}.$$

Grouping the data accordingly yields the network from figure 58.

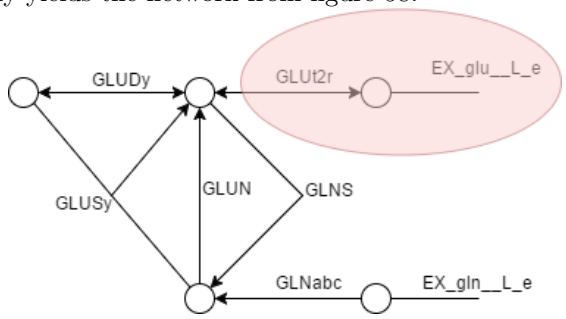


Figure 58: Schematic depiction of NM with groups.

6.7.1 Modified PC and NM

Applying modified PC with threshold 0.025 to this altered dataset yields the following results from figure 59.

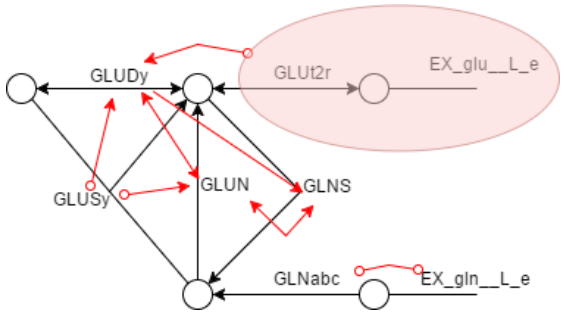


Figure 59: Outcome of applying modified PC with threshold 0.025 to the grouped nitrogen metabolism structure.

All the adjacencies in this result are compatible with the pathway network. However, it must be remarked that there are not many remote reactions in this structure. The only possibilities being either GLUDy or the GLUt2r group and GLNabc or EX_gln__L_e. However, GLNabc and EX_gln__L_e, are isolated. This is explained by the strong correlation between GLNabc and EX_gln__L_e and lemma 6.5.1. The strong correlation is shown in the scatterplot in figure 60. Furthermore, the impact of that strong correlation is amplified by the fluxes of these reactions. The fluxes of GLNabc and EX_gln__L_e are in the order of magnitude of 10^{-12}. This stands in contrast to the fluxes of the other reactions which are in absolute value greater than 10^{-2}.

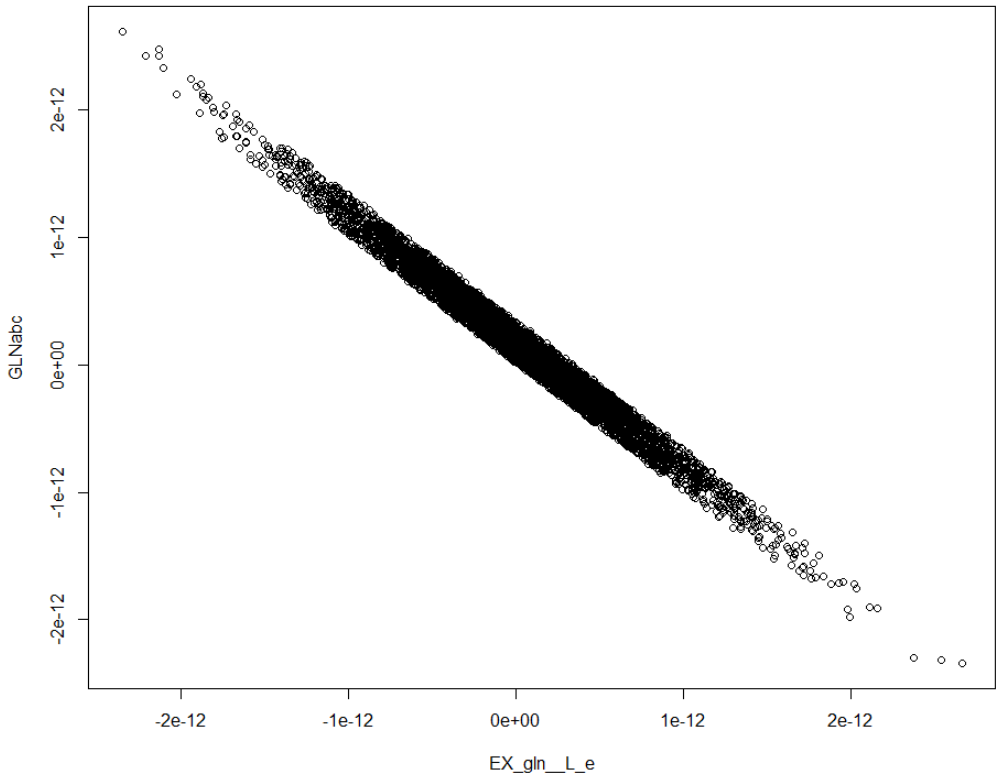


Figure 60: Scatterplot of the reactions GLNabc and EX_gln__L_e.

6.7.2 GES and NM

The results of applying GES to the grouped data are shown in figure 61.

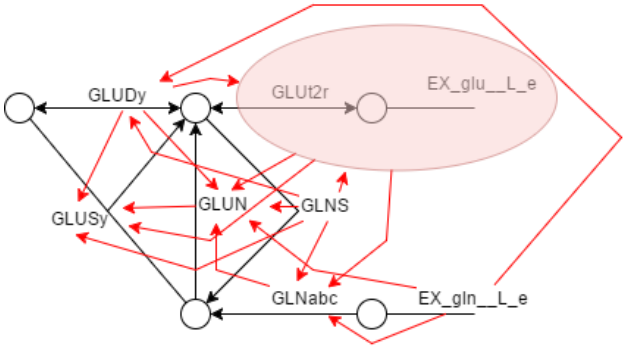


Figure 61: Outcome of applying GES to NM with groups.

Again, GES produces many arrows and also arrows between edges that are separated in the pathway network. The abundance of arrows makes the result uninformative. Raising the penalty term decreases the amount of edges. For example, multiplying the penalty factor with 40 gives the result depicted in figure 62.

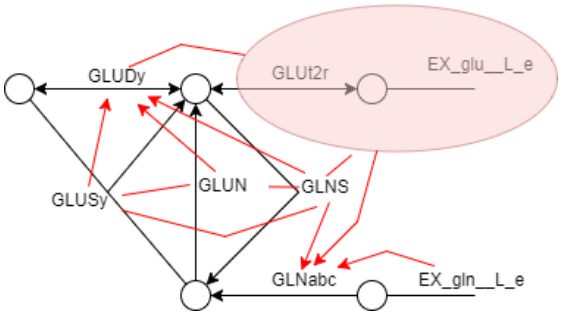


Figure 62: Outcome of GES applied to NM with groups with an increased penalty term, i.e. multiplied by 40.

Raising the penalty resulted in a higher concentration of relevant edges. Furthermore, the adjacencies GES found with the increased penalty term were all found when the penalty term was lower. Most of the edges are consistent with the pathway network. Note however that there are also several edges between remote reactions, such as the GLUt2r group and the reaction GLNabc. This shows that it is likely that the structure of NM did indeed reduce the effects of the deterministic relations.
We conclude that raising the penalty term improved the outcome of GES.

6.8 Conclusion and discussion

In this section we saw that this type of data presents several challenges for modified PC and GES. Both performances suffered heavily under the deterministic relations between variables. When the causal structure is shaped in a sequence, then these deterministic relations lead to remote reactions being connected by modified PC and GES. The impact of these deterministic relations seems to be less when the structure is more dense.
Furthermore, the impact of these deterministic relations in the shape of strongly correlated variables can in some cases be dealt with by grouping. Hence, given a dataset one can improve the results acquired by modified PC and GES by first searching for deterministic relations in the data and highly correlated groups of variables. Grouping these variables then leads to more informative causal structures.
Furthermore, one can get a suggestion of which groups of variables are too correlated by comparing the output of GES and modified PC. If a group of variables is isolated in modified PC but not in the outcome of GES, then this is an indication that the variables of that group are correlated too heavily.
Another unexpected result is that the outcome of GES in many cases was informative due to a high amount of edges in the result. Hence, it proved to be useful to increase the penalty term in the BIC score in order to receive fewer edges in the outcome. This is unexpected as there are optimality results of GES with the BIC score. We consider this in more detail in section 8.4.
One of the difficulties of the model of FBA was that we did not have a clear ground truth. There was a pathway network. However, this does not perfectly show which edges one can expect.
Furthermore, considering three different larger structures proved to be effective in the sense that it gave several perspectives on the mistakes made by the algorithms.
In the end it seems that this type of data is too different from the standard data in order to be suited to improve modified PC and GES. As a result it seems more fruitful, given the goal of bridging the gap, to consider a dataset that is a little closer to the standard assumption. We do this in the following section.

7 Gene regulatory networks

In the previous section we considered a dataset derived from flux balance analysis. Deterministic relations between variables complicated the matter and led to the formation of improbable adjacencies. In this section we consider a dataset that does not exhibit these deterministic relations. The dataset we consider in this section regards so-called gene regulatory networks. These gene regulatory networks are collections of regulators that determine so-called gene expression levels. Genes are encoded in the DNA and gene expression is the process by which the information encoded in the genes is used in the production of, for instance, proteins. The regulators in a regulatory network influence not only the expression of a certain gene but also the concentrations of other regulators. The branch reverse engineering of genetic networks is occupied with inferring the structure of a gene regulatory network based on the expression levels of its genes. That is, given data of gene expression levels one aims to derive the structure of the gene regulatory network.In this section we evaluate modified PC and GES as means to do so. In order to do this we apply these algorithms to simulated data of a generated gene regulatory network.
This section is outlined as follows. First we describe how the data is generated. Here we emphasize on the structural properties of the gene regulatory networks. Then we apply modified PC and GES to multiple networks. We end this section with a conclusion and discussion in which we interpret the results of applying modified PC and GES to the networks.

7.1 Data generation for simulated gene regulatory networks

Amongst other approaches, cf. [13], gene regulatory networks can be simulated by systems of ordinary differential equations (ODEs). The expression levels of a gene are represented by ODEs. Concretely, the expression of a gene is measured in the concentration of so-called RNA transcripts. For each of the genes the rate of synthesis of its transcripts is represented as a linear combination of the concentrations of the transcripts. More precisely, let $x(t)$ be a vector containing the concentrations of several transcripts on time t, then

$$\frac{d}{dt}x(t) = Ax + b,$$

for a matrix A. Here b represents a pertubation of the network. Given such a system of ODEs one can consider the expression levels over time and one can make predictions regarding the behavior of the gene regulatory network under certain interventions, such as pertubations.
The specific datasets used in this section are from [1]. They are generated as follows. For a generated gene regulatory network and a system of ODEs governing it, pertubations are executed on the entire system. This corresponds to changing the vector b in the description of the system of ODEs. Given such a vector b a steady state solution is determined, i.e. a vector x such that $Ax + b = 0$. If the matrix A is invertible, such a steady state x can be computed by

$$x = -A^{-1}b.$$

Hence, for a set of N samples, N pertubation vectors are generated. Afterwards, noise is added by summing to each component X_i of the steady-state solutions the value $|X_i|\varepsilon$, where ε is noise with zero mean and a standard deviation of 0.1. Note, however, that this noise is added in the final stage of the data generation. Hence, the error does not accumulate over the set of variables (as is the case with the linear Gaussian model from section 4).
The datasets used in this section are referred to as 'static global pertubation datasets' in [1]. There are twenty static global pertubation datasets and for each of these datasets there is a network (i.e. there are twenty different matrices A) for which data has been generated. The twenty networks have similar properties. The average degree is roughly the same and the number of variables is the same for each network, i.e. ten. In this section we consider three of these networks in detail. The number of networks considered is a compromise between the desire to consider many networks in detail and the feasibility of considering many networks in detail. Since the networks are roughly the same, we consider networks 1, 2 and 3.
In contrast to the situation of the previous section, for these networks there is an obvious candidate for

the ground truth. The ground truth for such a network is determined by the following property: there is a directed edge from X_i to X_j iff $A_{ji} \neq 0$, i.e. if the derivative of X_j is dependent of X_i.
Note that the data generation is similar to the data generation from section 4. However, considering this dataset as well is not only insightful because the dataset is a realistically simulated one, but also does the ground truth exhibit cycles and self-loops (i.e. an edge from a node to itself). Modified PC and GES are not designed to be able to identify cycles and self-loops and hence, it becomes of interest to consider how they handle this dataset. In the following graphs these self-loops are indicated by a bold edge of the node in order to increase the readability of the graph.
As mentioned, the networks we consider all consist of ten nodes. However, for network 3, we make one variable latent by erasing it from the dataset. Each of the considered datasets consists of 1000 samples.

7.2 Network 1

The ground truth with the results of applying modified PC with threshold 0.025 to the dataset of network 1 is displayed in figure 63. Note that every node has a selfloop.

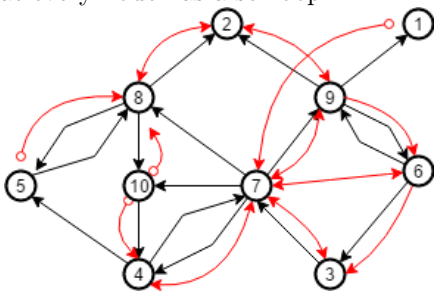


Figure 63: Result of applying modified PC with threshold 0.025 to network 1.

The results are very promising. The algorithm finds twelve adjacencies, two of which are erroneous. The precision is 83 percent. The ground truth contains fourteen adjacencies (not considering the self-loops and counting cycles of length two as a single adjacency). Therefore, modified PC has a recall of 71 percent. Hence, the skeleton is approximated rather accurately. However, we are also interested in discovering why some of the edges are not identified by modified PC. In order to do this we consider the differential equations of the targets of these edges. The particular edges are from from 9 to 1, from 4 to 5, from 7 to 8 and from 7 to 10. The ODEs for these targets are

$$\begin{aligned}
\frac{d}{dt}X_1 &= -1.025365X_1 + 0.013458X_9 \\
\frac{d}{dt}X_5 &= 0.151477X_4 - 0.694908X_5 - 0.167680X_8. \\
\frac{d}{dt}X_8 &= -0.162610X_5 - 0.023827X_7 - 0.812990X_8 \\
\frac{d}{dt}X_{10} &= 0.145298X_7 - 0.252668X_8 - 0.890607X_{10}
\end{aligned}$$

The absolute value of the coefficient of the missing sources (i.e. of nodes $9, 4$ and 7) is the lowest in each of the ODEs. However, the coefficient of X_4 in the derivative of X_5 and the coefficient of X_7 in the derivative of X_{10} are large enough to be noticed by modified PC. In the other cases, we see that the self-loops become relevant. Their influence outweighs some of the other influences, resulting in errors in the skeleton.
Note that, as modified PC cannot return self-loops, the self-loops are missed entirely by the algorithm. Similarly, none of the cycles are returned. Again, this is consistent with the algorithm. However, it is not clear how this can be detected from the outcome. In fact, these results show that it is unlikely that one is able to detect the presence self-loops with modified PC from this dataset. Note that although the interpretations of the two datasets are different, the data generation for gene inference is reminiscent of the data generation

from section 4. In this case however, the dataset represents a system in which there can be self-loops, whereas in section 4 the causal system did not contain self-loops. Hence, since the datasets are generated in a similar, albeit not identical, fashion but the interpretations are different, it is unlikely to be able to detect self-loops with modified PC for this dataset.

Regarding the orientations, we can evaluate these in two ways. We can interpret the ground truth in order to check whether the result is what we expect or not, or we can interpret the outcome and compare its statements with the ground truth. We start with the latter approach.
There are many bidirected arrows in the outcome. For PAGs a bidirected arrow indicates the presence of a latent common cause. However, there are no latent common causes for this dataset. Hence, those orientations are wrong.
The orientation of v-structures contributes heavily to these bidirected arrows. However, the orientation of these wrong v-structures can be prevented. One can derive conditional independence statements from v-structures in the outcome of the PAG, cf. 3.1.1. These claims can be tested. Should these statements prove to be invalid, then the v-structure orientation is false.
For instance, consider the triple (10,8,2). Modified PC oriented (10,8,2) as a v-structure using the conditional independency constraints. However, this would imply that 10 is dependent of 2, given 8. This can be checked. The independence test returns that 10 is independent of 2 given 8. Hence, triple $(10, 8, 2)$ should not have been identified as a v-structure.
In this case the so-called conservative PC algorithm would have noticed this mistake. The conservative variant for the PC algorithm checks all these conditional independence statements entailed by the orientation of v-structures prior to orienting the v-structures. For a detailed description of the conservative PC algorithm, consider [20].
The triples (5,8,10), (10,4,7) and (2,9,7) imply similar claims. The independence tests, however, state that 5 is independent of 10 given 8 with p-value 0.619, 7 is independent of 10 given 4 with p-value 0.718 and 2 is independent of 7 given 9 with p-value 0.046. Note that the latter p-value is only slightly above the threshold, hence this measured independency could also have been caused by a wrongly chosen threshold.

For comparison sake we display the result of applying FCI[7] with the majority rule to this dataset, [8]. The majority rule FCI is a version of conservative FCI where one only requires that 50 percent of the conditional independence statements entailed by orienting a v-structure are satisfied in order to actually orient a triple as a v-structure. The result is shown in figure 64.

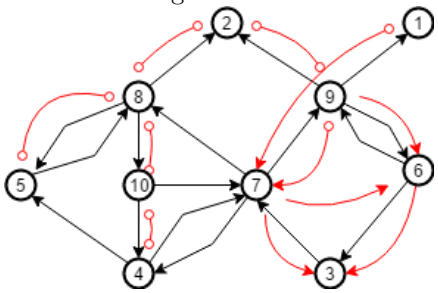


Figure 64: Result of applying FCI with majority rule and threshold 0.025 to network 1.

The majority rule resulted in an, orientation-wise, uninformative graph: there are very few orientations. However, an argument can be made that this is more desirable. Whereas the results from figure 63 erroneously indicate the presence of latent common causes, the outcome from figure 64 makes no such claims. Hence, figure 64 is more consistent with the ground truth. Figure 64 shows the result of applying FCI with the majority rule. One could also apply FCI with the conservative rule. However, since this algorithm is more strict than FCI with the majority rule, the outcome would likely be even less informative orientation-wise.

[7]The choice of applying FCI here instead of modified PC is that there is already a built-in version of the majority rule for FCI in the R-package pcalg. Furthermore, based on the skeleton of the outcome, the result should be the same.

Now we try to interpret the ground truth and find why modified PC identifies wrong v-structures from figure 63. Consider triple (10,8,2). Should we assume that one can reason with d-separation, then we would have that 10 is independent of 2 given 8, but 10 and 2 should be dependent. However, the independence test returns that 2 is independent of 10. Again, this is due to the self-loops at 8. This self-loop reduces the connection between 2 and 10.
Similarly we would have that 7 and 2 are independent given 9, but that 2 and 7 are dependent. Indeed, according to the conditional independence test we have that 7 and 2 are indeed independent given 9 and 2 and 7 are dependent. However, it also returns that 2 and 7 are independent given 3. This is the reason (2,9,7) is oriented as a v-structure: when modified PC finds that 2 and 7 are independent given 9, it does not test whether 2 and 7 are independent given 9. Hence, we see that the order of the independence test now affects the orientation of the v-structure.[8] Note, however, that the majority rule version of FCI is not order independent concerning the orientation of v-structures.

Applying GES to this dataset yields the results from figure 65.

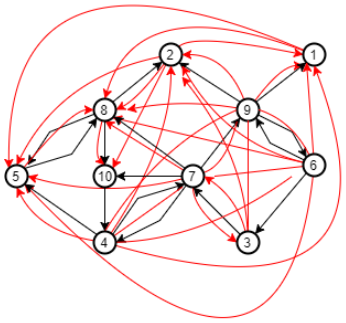


Figure 65: Outcome of applying GES to network 1.

Greedy equivalence search returns too many edges. This overestimation makes the outcome highly uninformative. The penalty term is $\frac{1}{2}\log(\text{no\_samples})$ by default. However, in this case it presented us with a dense graph. When we raise the penalty term to, for instance, $10\log(\text{no\_samples})$ we get the results from figure 66.

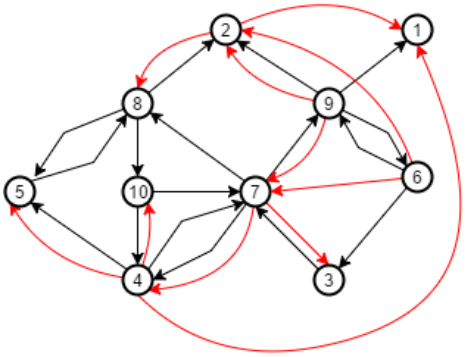


Figure 66: Outcome of applying GES with a raised penalty term, multiplied by 20, to network 1.

With this raised penalty term, GES returns eleven edges, four of which are erroneous. Hence, we have a precision of 63 percent and a recall of 50 percent (again we count the cycles of length two as a single adjacency and we do not consider the self-loops for the precision and recall). These results are not as good as the results of modified PC but significantly more informative than the dense graph. Furthermore, applying GES with

[8] In section 2 we mentioned using the stable PC algorithm. However, this is only used for the determination of the skeleton, the rest is still order dependent.

penalty terms $\frac{1}{2}\log(\text{no\_samples})$, $0.7\log(\text{no\_samples})$, $\log(\text{no\_samples})$, $2\log(\text{no\_samples})$, $8\log(\text{no\_samples})$ and $10\log(\text{no\_samples})$ shows that as the penalty term increases, edges are omitted but not added. This shows that GES is quite stable and that the remaining edges in fact do have the highest likelihood according to GES.
Just as for modified PC, GES misses all of the self-loops. Regarding the orientation results, when GES returns a correct adjacency, there is a 66 percent chance that it is orientated in the wrong direction (if there is a clear orientation in the ground truth). This is a very poor result and an obvious improvement in this case would be to reverse all the arrows. It is not clear whether this is a systematic error for GES or that in this case it made a mistake. The other networks we will consider hopefully shed light on this.

7.3 Network 2

The result of applying modified PC with threshold 0.025 to network 2 is depicted in the following graph.

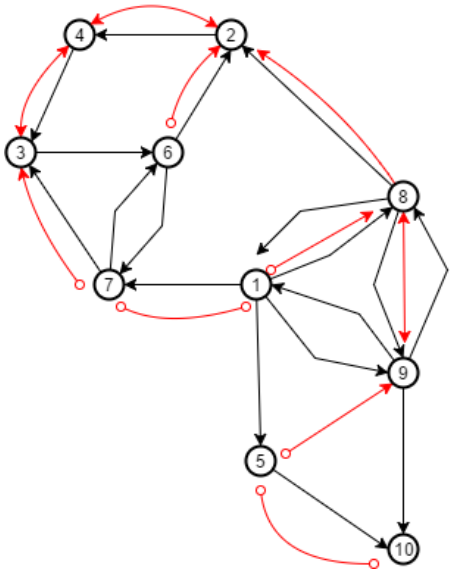


Figure 67: Result of applying modified PC with threshold 0.025 to network 2.

Modified PC finds 10 adjacencies, nine of which are correct, resulting in a precision of 90 percent. The ground truth contains 13 edges. Hence, the recall is 69 percent.
The adjacencies modified PC did not find are between nodes 3 and 6, between nodes 7 and 6, between 1 and 5, between 1 and 9 and between 9 and 10. The differential equations regarding 5, 6 and 10 are

$$\begin{aligned}
\frac{d}{dt}X_5 &= 0.076311X_1 - 3.848697X_5 + 0.801236X_9 \\
\frac{d}{dt}X_6 &= -0.235698X_3 - 3.719127X_6 - 0.037688X_7 \\
\frac{d}{dt}X_{10} &= 0.561379X_5 + 0.267959X_9 - 1.550797X_{10}
\end{aligned}$$

Again, the coefficients standing in front of the missing sources are the smallest in absolute value for each equation. However, the coefficient in front of X_9 in the derivative of X_{10} is not small enough to account for the missing edge.
Regarding the orientations, there are three bidirected arrows. Since there are no hidden common causes, there should not be any bidirected arrow. The bidirected arrows are between 2 and 4, between 3 and 4 and between 8 and 9.

The triples (8,2,4), (2,4,3) and (4,3,7) are identified as v-structures and only 4 and 7 are indeed dependent given 3. Nodes 4 and 8 are independent given 2 with p-value 0.073 and 2 and 3 are independent given 4 with p-value 0.360. This is consistent with the ground truth. Only (4,3,7) should be oriented as a v-structure. Regarding the v-structure (5,9,8), the conditional independency test concluded that 5 and 8 are independent given 9 with p-value 0.962. Hence, (5,9,8) should not have been oriented as a v-structure.
For comparison sake, the result of applying FCI with the majority rule to this dataset is depicted in figure 68.

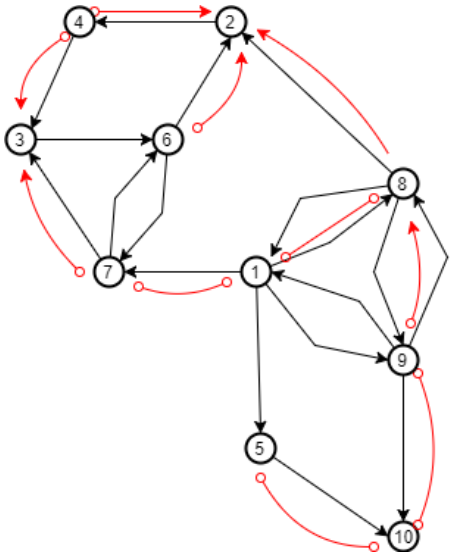


Figure 68: Result of applying FCI with the majority rules and with threshold 0.025 to network 2.

In this instance the outcome of FCI does indeed differ from the outcome of modified PC. The edge between 5 and 9 is not present in the FCI outcome. We even see that the precision has increased to 100 percent. Furthermore, both the improvement in precision as the majority rule have made the outcome free from bidirected arrows and most arrowheads are consistent with the ground truth.

The result of GES with the default penalty term contained too many edges. Raising the penalty term to $10 \log(\text{no\_samples})$ gave the results from figure 69.

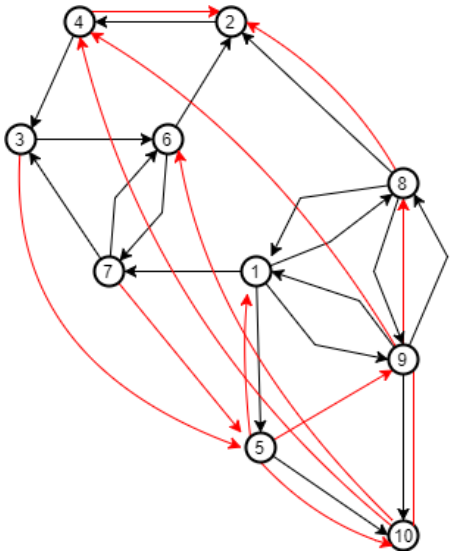


Figure 69: Result of applying GES with an increased penalty term, multiplied by 20, to network 2.

Although the graph is sparse enough to be informative, it returns twelve adjacencies, only six of which are correct. Hence, we have a precision of 50 percent and a recall of 46 percent. Furthermore, from the correct adjacencies GES found where the ground truth as the GES outcome have a definitive orientation, GES oriented only 50 percent correct.

7.4 Network 3

For network 3 we consider variable X_5 to be latent in order to see how modified PC and GES handle this. The latent variable is represented with a dotted node in figure 70.

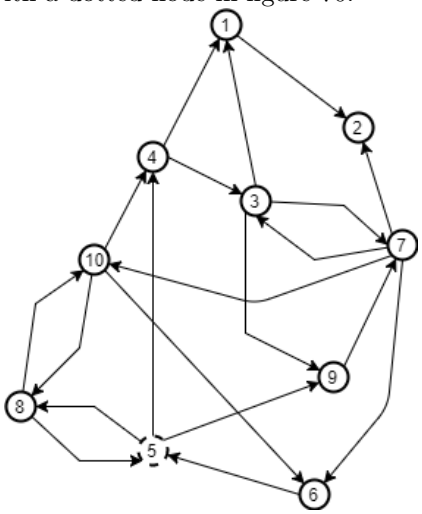


Figure 70: Schematic depiction of network 3.

Applying modified PC with threshold 0.025 yields the result from figure 71.

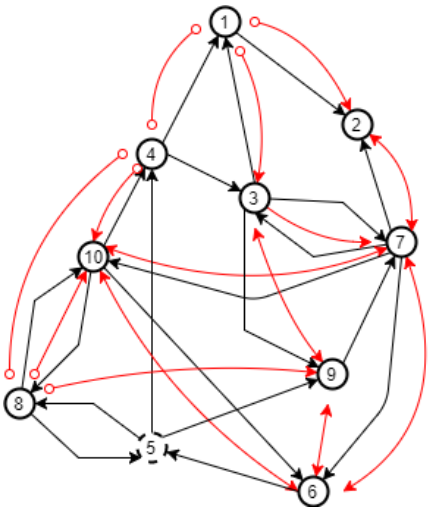


Figure 71: Result of applying modified PC with threshold 0.025 to network 3.

Apart from the edges between 4 and 8 and between 8 and 9 from the output, every identified adjacency is correct. Furthermore, since both 4 and 8 and 8 and 9 are connected in the ground truth via node 5, this retrieved adjacencies are correct as well. Hence, we have a precision of 100 percent. Furthermore, we aimed to find all the adjacencies in the ground truth between two non-latent variables and an edge for every pair of non-latent variables both connected with the hidden variable. This amounts to nineteen edges. However, we retrieved fourteen, thus we have a recall of 82 percent.
Regarding the orientations, however, there are no correctly identified bidirected arrows. Therefore, should we have interpreted the outcome PAG, then we would not have noticed the influence of variable X_5 but we would have erroneously identified the presence of several other hidden common causes.
When we apply FCI with the majority rule, then we get the result from figure 72.

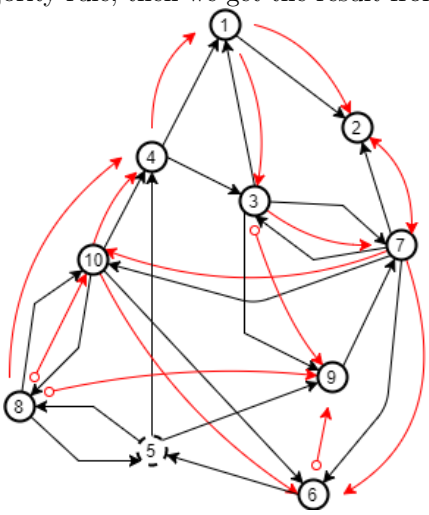


Figure 72: Outcome of applying FCI with majority rule and with threshold 0.025 to network 3.

Still there is no bidirected edge indicating the presence of variable 5, but the amount of erroneous bidirected arrows has been reduced to one. Furthermore, the prevention of orienting several v-structures has allowed the algorithm to orient edges in a different way than before. For instance there is a directed edge from 4 to 1 in the FCI outcome where there was an edge with two circles between 4 and 1 in the modified PC outcome.

Applying GES to this dataset with the standard penalty term is, again, too dense. Applying GES with

penalty term $10 \log(\text{no\_samples})$ yields the result from figure 73.

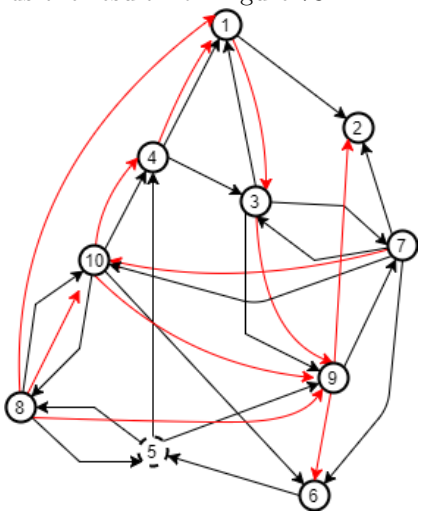


Figure 73: Result of applying GES with an increased penalty term, multiplied by 20, to network 3.

From the eleven edges GES found, eight are consistent with the ground truth. This gives us a precision of 73 percent. Furthermore, we have a recall of 47 percent.
For the correctly identified adjacencies for which there was a clear orientation in the ground truth, GES oriented that edge correctly in 75 percent of the cases.

7.5 Conclusion and discussion

Overall we see that modified PC can handle this dataset rather well, GES on the other hand is a little less successful. It turned out that modified PC can approximate the skeleton rather accurately, obtaining precisions of higher than 83 percent and recalls of higher than 69 percent. The performance of modified PC on orienting the edges was not good: not only does modified PC conclude the presence of several hidden common cause where there in fact are none, modified PC also missed the presence of the hidden variable in network 3. Applying the majority rule improved the result in the sense that less hidden common causes are outputted, but then the output became less informative. Hence, on the output of modified PC on gene regulatory networks one can have faith that the skeletons are rather accurate but one should not put too much faith in the orientation results. Based on the results from this section I would even recommend using the majority rule for this type of data.
Greedy equivalence search had the problem of overestimating the amount of edges. Raising the penalty term made the graph sparser and the result more informative. With a raised penalty term, the determination of the skeleton was less successful than for modified PC and the orientation results were, when they could be verified, not perfectly accurate. In the evaluation of network 1 it was mentioned that further evaluation was needed in order to conclude whether it was systematic. Based on the results of the remaining networks, it seems that sometimes the orientation is relatively accurate and sometimes it is not. Hence, reversing the arrows is not a solution to the problem.
Nonetheless, the results of both modified PC and GES on gene regulatory network were a significant improvement of the situation with flux balance analysis. The reason for this difference in performance lies in the generation of the data. Whereas the dataset of flux balance analysis displayed several deterministic relations, the distribution of the data for gene regulatory networks is more in line with the data generation of section 4. Indeed, the data from section 4 was generated by generating vectors x from the multivariate Gaussian distribution and then calculating $(I - A)^{-1}x$, for a given lower triangular matrix A. For the gene regulatory network, however, vectors $-A^{-1}b$ were determined where the vectors b were, most likely, randomly drawn from $[0, 10]^{10}$. Hence, the datasets for gene regulatory networks were more consistent with the standard assumption of modified PC and GES than the datasets from flux balance analysis. Nonetheless, modified

PC made several mistakes with the orientation. These mistakes have three likely causes. First of all, the influence of the self-loops and cycles reduced the influence of several variables, as mentioned in evaluations of modified PC and GES on the network. Secondly, the datasets consist of 1000 samples. Given that the data generation is similar to the one from section 4, we can get an impression of how accurate modified PC and GES are from considering the precision recall plots for the sample size from section 4. It becomes apparent that a sample size of 1000 is then too low in order to obtain perfect results. Lastly, since the vectors b are most likely generated from the uniform distribution of $[0, 10]^{10}$, the resulting distribution of the sample set is not a multivariate Gaussian, hence the conditional independence test is not ideal and the scoring is neither.

8 General observations, conclusions and future work

We start this section by answering the main questions as presented in the introduction.

1. How can the mistakes modified PC and GES make be characterized?
2. How can these mistakes be recognized?
3. How can one avoid these mistakes?

8.1 Characterization of mistakes

Regarding the characterization of mistakes, for modified PC the mistakes can be categorized into two classes: mistakes in the determination of the skeleton and mistakes in determining the v-structures. Mistakes are also made in the remaining part of the orientation phase, but considerably less and of a less damaging nature. Both the mistakes in the determination of the skeleton and of the v-structures are due to errors in the conditional independency tests. These errors are the results of a combination of three factors. First of all, the sample size is too small. Even in the situation where the standard assumptions were valid, the sample size needed to be around 10^7 in order to receive a near perfect result. Furthermore, if the data is generated by non-standard distributions, then the used conditional independence test is not always valid, resulting in mistakes in the conditional independency tests. Lastly, as seen in the dataset regarding the gene regulatory networks, feedback and, in particular, self-loops can result in a diminished influence of other variables. Therefore, modified PC might miss the impact of these variables.
Furthermore, when the dataset is devoid of deterministic relations between the variables, the skeleton can be approximated rather accurately. This becomes apparent from every dataset considered in this thesis, with the exception of flux balance analysis. If deterministic relations between the variables are present, then modified PC returns unlikely adjacencies. However, if the structure of the ground truth is less like a sequence but of a more interwoven structure, then the adjacencies modified PC returns are far more likely. It also becomes apparent that when variables are correlated too heavily, their connections to the other variables is influenced.
The orientation results of modified PC are only meaningful when the skeleton is approximated rather accurately. In the cases of the datasets from section 4 and section 5 the orientations are rather accurate. This stands in contrast to orientation results for the datasets of gene regulatory networks. The implications of those orientations are not consistent with the ground truth.
For GES the results show that when the dataset is Gaussianly distributed, then GES is fairly accurate in determining the skeleton. Even if the dataset is first generated by a multivariate Gaussian distribution and then pertubated, as is the case in section 5, then GES is still able to recover most of the skeleton correctly. However, mistakes made by GES regarding the skeleton are most notable for the dataset of FBA and gene regulatory networks. Greedy equivalence search consistently overestimates the amount of edges. Raising the penalty term reduced the amount of edges. We will consider this phenomenon later in more detail.
In the case of deterministic relations between the variables, when one raised the penalty term GES produced results that are similar to the outcome of modified PC. This includes the unlikely adjacencies.

8.2 Recognizing mistakes

For modified PC one can detect mistakes due to a correlation which is too high by looking for isolated groups of variables. These groups of variables should either have no edges between its variables or the subgraph of such a group of variables should be complete, as can be deduced from lemma 6.4.1 and 6.5.1. Then one considers the correlation between members of this group of variables. If the correlation is 1 in absolute value then the correlation is too high and most likely mistakes have been made as a result. For deducing whether or not the correlation is too high one can also apply GES and see whether GES finds a connection between variables from that group and variables outside of the group. This gives an indication of whether or not the correlation is too high.

Regarding the erroneous orientation v-structures one can recognize mistakes by performing additional tests. The presence of a v-structure implies various conditional independency statements which can be tested. Should most of these conditional independency statements prove to be not true, then it is likely that the orientation of that v-structure is erroneous.

8.3 Avoiding mistakes

Avoiding the erroneous orientation of v-structure can be done with the majority rule, where one only orients a v-structure if more than 50% of the conditional independency statements entailed by that v-structure are correct. As seen in section 7, this results in a graph where only few edges are oriented.
Mistakes due to a correlation that is too high can be avoided by considering such a group as a single variable and see how that group is related to the other nodes. Afterwards one can choose to applying further algorithms to the individual groups.

8.4 The penalty term of GES

As we saw in sections 6 and 7, GES overestimates the amount of edges and the resulting is too dense. Increasing the penalty term then proved to be fruitful since it reduced the amount of edges. On the one hand it is straightforward that raising the penalty term results in a sparser graph. Raising the penalty term leads to a higher penalty of complexity. Sparser graphs have a lower complexity than dense graphs. Hence, when the penalty term is sufficiently high, the lower complexity of the sparser graph outweighs the explanatory power of the denser one. As a result, raising the penalty term leads to a sparser graph.
On the other hand, the fact that the output of the standard penalty term is too dense is unexpected. The optimality results use the BIC score and therefore, the standard penalty term. Due to the results of section 4 the explanation for this deviation cannot be found in a small sample size. However, this deviations are likely due to the fact that the distributions are non-standard. Concretely, for flux balance analysis the datasets consisted of vectors sampled uniformly from the solution space and for the gene regulatory network the data without noise consists of a uniform draw from $[0, 1]^{10}$ which is multiplicated with a matrix.
In this thesis the increased penalty terms were found by trial and error, i.e. by the proces of increasing the penalty term until the outcome became reasonable. Finding methods for estimating the most desired penalty term is a topic for further research.

8.5 Multiple outcomes

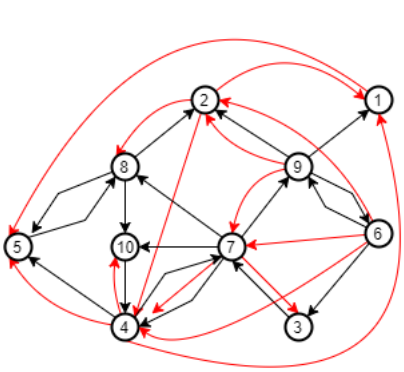
In the introduction it was mentioned that some score-based algorithms return several high scoring outcomes. It might be that combining these multiple outcomes can yield a better result than using only the highest scoring one. In this subsection we try to evaluate whether this approach is likely to be fruitful. If this approach proves to be useful, then it is a subject for further research. The search procedure of GES is specifically designed to find the highest scoring CPDAG. Altering this algorithm in order to reduce the top N highest scoring CPDAGs, however, is non trivial. Hence, we use a method called *bootstrapping* in order to obtain multiple similar, although not identical, datasets. We will consider the method later. Applying GES to each of these datasets might give different outcomes that one can compare in order to improve the outcome.
This approach is also applicable to modified PC. However, as modified PC uses conditional independency tests, it might be that the data obtained through bootstrapping is too similar to the original dataset in the sense that the conditional independency tests yield the same results. On the other hand, it might be that borderline cases of conditional independency are varied.
The dataset we will reevaluate with this approach is the one from network 1 of the gene regulatory networks, cf. section 7.2. This dataset was promising in that it produced reasonable but imperfect outcomes of modified PC and GES. We will now first consider bootstrapping and then we will execute the above procedure, first for GES and then for modified PC.

8.5.1 Bootstrapping

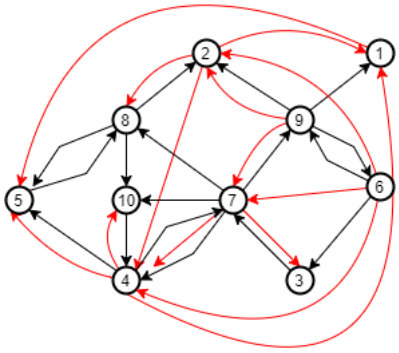
Bootstrapping can be described as follows. Let D be an obtained dataset with N samples. Then one can create a new, temporary, dataset S of M samples by drawing M samples from D. Then one can replace the first M elements from D with S, this results in a data set D', which in the limit of large sample size has the same distribution as D. However, in practice there are slight differences between D and D'. The chosen M indicates how much of the original dataset is resampled. In the following we will take $M = N$.

8.5.2 GES and multiple outcomes

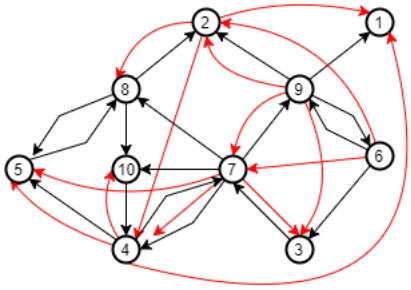
Applying GES to both the original dataset and two datasets obtained by bootstrapping the original data gives the results from figure 74. The used penalty term is $5 \log(\text{no\_samples})$. We use this penalty term and not the default penalty term, since for the default penalty term, the output was too dense. However, using the increased penalty term from section 7, might give an outcome that is too sparse if one also wants to erase unlikely edges.



(a) Original dataset



(b) Resampled dataset 1



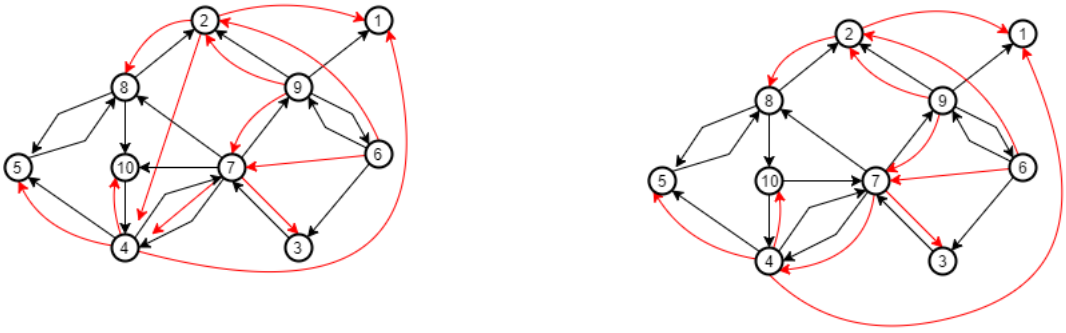
(c) Resampled dataset 2

Figure 74: The outcomes of GES applied to the original dataset and two datasets obtained by bootstrapping.

The outcomes of these three CPDAGs can be used to create a new outcome by only allowing the arrows that

are present in all three CPDAGs. In figure 75 we depict the result of this procedure next to the result of applying GES with the penalty term 10log(no_samples), i.e. the result obtained in section 7.2 and displayed in figure 66.

Combining the results of the CPDAGs did not result in a better outcome than raising the penalty term to



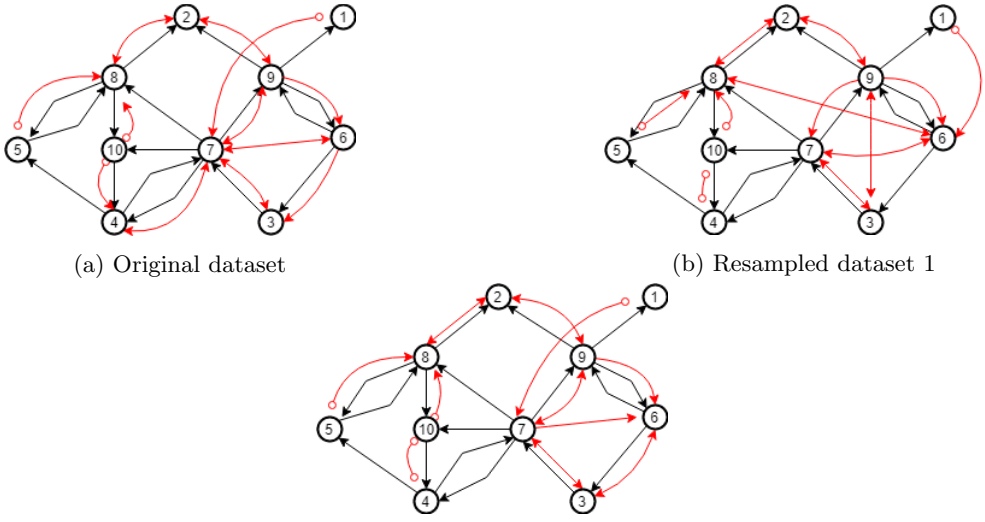
(a) (b)

Figure 75: (a) Result from combining the CPDAG outcomes and (b) the result of GES with increased penalty term from section 7.2 for comparison.

10log(no_samples). The result of combining the CPDAGs is the same as the outcome of raising the penalty term, with the exception of the edge between 2 and 4 that is presented in the combination of the CPDAGs. This edge is not consistent with the ground truth. We conclude that this multiple output approach does not seem to be fruitful for GES.

8.5.3 Modified PC and multiple outcomes

For modified PC we use the same datasets as the ones used for testing GES and multiple outcomes. The threshold is set on 0.025 and the results are displayed in figure 76. We now combine these PAGs into a single



(a) Original dataset (b) Resampled dataset 1

(c) Resampled dataset 2

Figure 76: The outcomes of modified PC applied to the original dataset and two datasets obtained by bootstrapping.

graph. This graph is constructed as follows. Two nodes are adjacent if they are adjacent in every outcome from figure 76. Furthermore, if two edges are adjacent then we orient the edge so that it is consistent with most of the outcomes from figure 76. If there is no orientation consistent with most of the outcomes from figure 76, then we place a dotted line for that adjacency. The resulting graph can be found in figure 77.

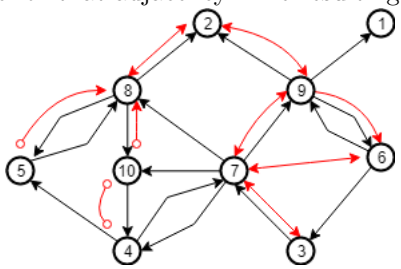


Figure 77: Result of combining the outcomes from figure 76.

In comparison with applying modified PC to the original data, the skeleton of this combination has 2 correct edges less (namely, the one between 3 and 6 and the one between 4 and 7). However, it also has one wrong edge less (i.e. the one between 1 and 7). The impact of bootstrapping on the outcome of modified PC had more effect than was expected. Due to this result, it seems that using bootstrapping in combination with modified PC might be fruitful. Hence, this is a topic for further research.

8.6 Future work

In this final subsection we conclude this thesis with a list of topics for future research.

- In section 6 and 7 it turned out that the standard penalty term of the score for GES was not sufficient. The outcome was too dense. In those sections we raised the penalty terms to specific values. These values were determined through trail and error: raise the term until the outcome is sparse enough. A topic for future research is finding a method to estimate a penalty term that is sufficiently large.

- In section 6 the datasets were filled with deterministic relations between the variables. In this thesis sketches were made of methods to deal with these relations. For instance, approaching the problem via grouping the variables and applying separate causal discovery algorithms might be a solution. Furthermore, a combination of GES and modified PC might be used to discovery which variables should be grouped. Working out these methods in details and testing them is a topic for future work.

- The multiple outcome approach considered in the above is also a topic for future research. In particular, finding methods of assessing which edges are likely to be valid and which edges are dubious is of interest. This is more likely to be fruitful for modified PC than for GES. For GES, however, one can look for ways of altering the search procedure so that GES can return multiple outcomes.

- In section 6 it occurred that several variables were too correlated. A question for further research is to quantify the notion 'too correlated' and for finding ways of identifying these variables.

References

[1] M. Bansal, V. Belcastro, A. Ambesi-Impiombato, and D. Di Bernardo. How to infer gene networks from expression profiles. *Molecular systems biology*, 3(1):78, 2007.

[2] S. A. Becker, A. M. Feist, M. L. Mo, G. Hannum, B. Ø. Palsson, and M. J. Herrgard. Quantitative prediction of cellular metabolism with constraint-based models: the cobra toolbox. *Nature protocols*, 2(3):727, 2007.

[3] C. M. Bishop. *Pattern Recognition and Machine Learning (Information Science and Statistics)*. Springer-Verlag New York, Inc., Secaucus, NJ, USA, 2006.

[4] D. M. Chickering. Learning bayesian networks is np-complete. *Learning from data: Artificial intelligence and statistics V*, 112:121–130, 1996.

[5] D. M. Chickering. Optimal structure identification with greedy search. *Journal of machine learning research*, 3(Nov):507–554, 2002.

[6] D. M. Chickering and C. Meek. Selective greedy equivalence search: finding optimal bayesian networks using a polynomial number of score evaluations. In *Proceedings of the Thirty-First Conference on Uncertainty in Artificial Intelligence*, pages 211–219. AUAI Press, 2015.

[7] T. Claassen, J. M. Mooij, and T. Heskes. Learning sparse causal models is not np-hard. In *Proceedings of the Twenty-Ninth Conference on Uncertainty in Artificial Intelligence*, pages 172–181. AUAI Press, 2013.

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

[9] J. Y. Halpern. Sufficient conditions for causality to be transitive. *Philosophy of Science*, 83(2):213–226, 2016.

[10] M. Kalisch and P. Bühlmann. Estimating high-dimensional directed acyclic graphs with the pc-algorithm. *Journal of Machine Learning Research*, 8(Mar):613–636, 2007.

[11] M. Kalisch, A. Hauser, M. Maechler, D. Colombo, D. Entner, P. Hoyer, A. Hyttinen, J. Peters, N. Andri, E. Perkovic, et al. Package 'pcalg'. 2017.

[12] Z. A. King, J. Lu, A. Dräger, P. Miller, S. Federowicz, J. A. Lerman, A. Ebrahim, B. O. Palsson, and N. E. Lewis. Bigg models: A platform for integrating, standardizing and sharing genome-scale models. *Nucleic acids research*, 44(D1):D515–D522, 2016.

[13] E. Klipp, R. Herwig, A. Kowald, C. Wierling, and H. Lehrach. *Systems biology in practice: concepts, implementation and application*. John Wiley & Sons, 2008.

[14] G. Lacerda, P. Spirtes, J. Ramsey, and P. O. Hoyer. Discovering cyclic causal models by independent components analysis. In *Proceedings of the Twenty-Fourth Conference on Uncertainty in Artificial Intelligence*, pages 366–374. AUAI Press, 2008.

[15] V. Lacroix, L. Cottret, P. Thébault, and M.-F. Sagot. An introduction to metabolic networks and their structural analysis. *IEEE/ACM transactions on computational biology and bioinformatics*, 5(4):594–617, 2008.

[16] C. Meek. Causal inference and causal explanation with background knowledge. In *Proceedings of the Eleventh conference on Uncertainty in artificial intelligence*, pages 403–410. Morgan Kaufmann Publishers Inc., 1995.

[17] J. D. Orth, R. M. Fleming, and B. Ø. Palsson. Reconstruction and use of microbial metabolic networks: the core escherichia coli metabolic model as an educational guide. *EcoSal plus*, 4(1), 2010.

[18] J. D. Orth, I. Thiele, and B. Ø. Palsson. What is flux balance analysis? *Nature biotechnology*, 28(3):245–248, 2010.

[19] J. Pearl. *Causality*. Cambridge university press, 2009.

[20] J. Ramsey, P. Spirtes, and J. Zhang. Adjacency-faithfulness and conservative causal inference. In *Proceedings of the Twenty-Second Conference on Uncertainty in Artificial Intelligence*, pages 401–408. AUAI Press, 2006.

[21] T. Richardson. A discovery algorithm for directed cyclic graphs. In *Proceedings of the Twelfth international conference on Uncertainty in artificial intelligence*, pages 454–461. Morgan Kaufmann Publishers Inc., 1996.

[22] J. Schellenberger, R. Que, R. M. Fleming, I. Thiele, J. D. Orth, A. M. Feist, D. C. Zielinski, A. Bordbar, N. E. Lewis, S. Rahmanian, et al. Quantitative prediction of cellular metabolism with constraint-based models: the cobra toolbox v2. 0. *Nature protocols*, 6(9):1290, 2011.

[23] S. Shimizu. Lingam: Non-gaussian methods for estimating causal structures. *Behaviormetrika*, 41(1):65–98, 2014.

[24] S. M. Smith, K. L. Miller, G. Salimi-Khorshidi, M. Webster, C. F. Beckmann, T. E. Nichols, J. D. Ramsey, and M. W. Woolrich. Network modelling methods for fmri. *Neuroimage*, 54(2):875–891, 2011.

[25] P. Spirtes, C. N. Glymour, and R. Scheines. *Causation, prediction, and search*. MIT press, 2000.

[26] P. Spirtes and K. Zhang. Causal discovery and inference: concepts and recent methodological advances. *Applied Informatics*, 3(1):3, Feb 2016.

[27] S. Triantafillou and I. Tsamardinos. Score-based vs constraint-based causal learning in the presence of confounders. In *CFA@ UAI*, pages 59–67, 2016.

[28] J. Zhang. On the completeness of orientation rules for causal discovery in the presence of latent confounders and selection bias. *Artificial Intelligence*, 172(16):1873–1896, 2008.

[29] K. Zhang and A. Hyvärinen. Distinguishing causes from effects using nonlinear acyclic causal models. In *Causality: Objectives and Assessment*, pages 157–164, 2010.