Estimating high-dimensional dependence structures with the PC-algorithm

A dissertation submitted to the
SWISS FEDERAL INSTITUTE OF TECHNOLOGY
ZURICH

for the degree of
Doctor of Sciences

presented by
MARKUS KALISCH
Dipl. Phys. ETH, MSc Applied Statistics
born December 28, 1977
citizen of Germany

accepted on the recommendation of
Prof. Dr. Peter Bühlmann, examiner
Prof. Dr. Marloes Maathuis, co-examiner

2008
Acknowledgements

First, I would like to thank Prof. Peter Bühlmann for supervising this thesis and giving me excellent help and guidance. I was always welcome to discuss ideas and issues. When in need for technical support or conceptual ideas, I found that Prof. Bühlmann has the talent of creating a very motivating and reassuring atmosphere while providing the help that is needed. Despite his authority, he manages to be a primus inter pares - well, apart from the SOLA, where he remains an outlier.

I also thank Prof. Marloes Maathuis for the nice collaborations, refereeing my thesis, for acting as co-examiner and, anyway, for spending time with an old man.

I also thank Dr. Martin Mächler for his patient and extensive support while creating the R-library, erhh, -package `pcalg`.

Furthermore, I want to thank the whole SfS for creating such a cordial atmosphere. Whether the secretaries, whose organised and yet affectionate qualities counterbalance the sometimes rather chaotic behavior of a standard PhD student, or Puma, who regularly provided brain food for ferociously hungry PhD brains, or the not yet mentioned professors, who always have clever advise when asked for and miraculously manage to produce questions even after the most cryptic seminar talk, or the fellow PhD students, with whom I could pet sharks, miss Philadelphia by 60 miles, make relativistic experiments on Rigi, pump iron like a pro or train regularly for the chocolate-cake-speed-eating contest: They all contributed to the wonderful atmosphere at the SfS.

Last but not least, I want to thank my parents for their continuous support both during my studies and during my PhD thesis and of course my little big brother for being a companion in so many situations.
Contents

Abstract ix

Zusammenfassung xi

1 Introduction 1

1.1 Different Graphs 1

1.2 Different Rules 3

1.3 Graphical Models in Practice 4

2 Estimating High-Dimensional Directed Acyclic Graphs with the PC-Algorithm 7

2.1 Introduction 7

2.2 Finding the Equivalence Class of a DAG 10

2.2.1 Definitions and Preliminaries 10

2.2.2 The PC-Algorithm for Finding the Skeleton 13

2.2.3 Extending the Skeleton to the Equivalence Class 17
<table>
<thead>
<tr>
<th>Section</th>
<th>Title</th>
<th>Pages</th>
</tr>
</thead>
<tbody>
<tr>
<td>2.3</td>
<td>Consistency for High-Dimensional Data</td>
<td>18</td>
</tr>
<tr>
<td>2.3.1</td>
<td>Finding the Skeleton</td>
<td>18</td>
</tr>
<tr>
<td>2.3.2</td>
<td>Extending the Skeleton to the Equivalence Class</td>
<td>19</td>
</tr>
<tr>
<td>2.4</td>
<td>Numerical Examples</td>
<td>20</td>
</tr>
<tr>
<td>2.4.1</td>
<td>Simulating Data</td>
<td>20</td>
</tr>
<tr>
<td>2.4.2</td>
<td>Choice of Significance Level</td>
<td>22</td>
</tr>
<tr>
<td>2.4.3</td>
<td>Performance for Different Parameter Settings</td>
<td>24</td>
</tr>
<tr>
<td>2.4.4</td>
<td>Properties in High-Dimensional Setting</td>
<td>24</td>
</tr>
<tr>
<td>2.4.5</td>
<td>Computational Complexity</td>
<td>27</td>
</tr>
<tr>
<td>2.5</td>
<td>R-Package <code>pcalg</code></td>
<td>28</td>
</tr>
<tr>
<td>2.6</td>
<td>Conclusions</td>
<td>32</td>
</tr>
<tr>
<td>2.7</td>
<td>Proofs and Appendix</td>
<td>33</td>
</tr>
<tr>
<td>2.7.1</td>
<td>Proof of Proposition 1</td>
<td>33</td>
</tr>
<tr>
<td>2.7.2</td>
<td>Analysis of the PC-Algorithm</td>
<td>34</td>
</tr>
</tbody>
</table>

3 Robust PC-Algorithm 43

<table>
<thead>
<tr>
<th>Section</th>
<th>Title</th>
<th>Pages</th>
</tr>
</thead>
<tbody>
<tr>
<td>3.1</td>
<td>Introduction</td>
<td>43</td>
</tr>
<tr>
<td>3.2</td>
<td>Robustification of the PC-Algorithm</td>
<td>44</td>
</tr>
<tr>
<td>3.3</td>
<td>Consistency of the PC-Algorithm</td>
<td>45</td>
</tr>
<tr>
<td>3.4</td>
<td>Numerical Results</td>
<td>47</td>
</tr>
<tr>
<td>3.4.1</td>
<td>Simulating Data</td>
<td>47</td>
</tr>
<tr>
<td>3.4.2</td>
<td>Performance of the Robust PC-Algorithm</td>
<td>48</td>
</tr>
</tbody>
</table>
### Contents

3.4.3 Computational Complexity .......................... 52  
3.4.4 Decision Heuristic ................................. 54  
3.5 R-Package pcalg ................................. 55  
3.6 Proofs .............................................. 57  
  3.6.1 Proof of Theorem 3 and Theorem 4 .............. 57  
  3.6.2 Proof of Theorem 4 .............................. 59  
  3.6.3 Proof of Lemma 6 ............................... 59  
  3.6.4 Proof of Proposition 3 ......................... 59  

4 Variable Selection for High-Dimensional Models: Partial Faithful Distributions, Strong Associations and the PC-Algorithm 61  
4.1 Introduction ........................................ 61  
4.2 Gaussian Linear Model and Partial Faithfulness ........ 64  
  4.2.1 Partial Correlation Screening Using Partial Faithfulness ........................................ 66  
4.3 Estimation Using the PC-Algorithm ................. 68  
  4.3.1 The Population Version of the PC-Algorithm . 68  
  4.3.2 Sample Version of the PC-Algorithm .......... 69  
4.4 Failure of Partial Faithfulness and Measures of Association 70  
4.5 Asymptotic Consistency in High Dimensions .......... 74  
  4.5.1 Consistency with Partially Faithful Distributions 74  
  4.5.2 Asymptotic Behavior for Non-Partial-Faithful Distributions ................................. 78
Abstract

We consider the PC-algorithm (Spirtes et al., 2000) for estimating the skeleton and equivalence class of a very high-dimensional directed acyclic graph (DAG) with corresponding Gaussian distribution. The PC-algorithm is computationally feasible and often very fast for sparse problems with many nodes (variables), and it has the attractive property to automatically achieve high computational efficiency as a function of sparseness of the true underlying DAG. We prove uniform consistency of the algorithm for very high-dimensional, sparse DAGs where the number of nodes is allowed to quickly grow with sample size $n$, as fast as $O(n^a)$ for any $0 < a < \infty$. The sparseness assumption is rather minimal requiring only that the neighborhoods in the DAG are of lower order than sample size $n$.

Moreover, a computationally efficient robustification of the PC-algorithm is proposed and we prove its consistency. Furthermore, we compare the robustified and standard version of the PC-algorithm on simulated data.

Finally, we consider the application of the PC-algorithm to the problem of variable selection in high-dimensional linear models where the number of covariates greatly exceeds the sample size. We present the concept of partially faithful distributions and discuss their role for inferring associations between the response and the covariates. For partially faithful distributions, a simplified version of the PC-algorithm, which
is computationally feasible even with thousands of covariates, yields consistency for high-dimensional variable selection under weak conditions on the (random) design matrix. Our assumptions are of a different nature than coherence conditions for penalty-based approaches like the Lasso. If partial faithfulness does not hold, we show that the PC-algorithm still consistently identifies some strong associations which are related to notions of causality. We also provide an efficient implementation of our (simplified) PC-algorithm, which is tailored for variable selection and demonstrate the method on simulated and real data.

All of the above mentioned methods are available in the R-package `pcalg`
Zusammenfassung

Wir untersuchen den PC-Algorithmus (Spirtes et al., 2000) zur Schätzung des Skeletts und der Äquivalenzklasse eines sehr hochdimensionalen Gerichteten Azyklischen Graphen (DAG) mit zugehöriger Normalverteilung. Der PC-Algorithmus ist für Schätzprobleme mit vielen Knoten und kleiner maximaler Nachbarschaft (spärliche Besetzung) gut geeignet. Die Rechenzeit ist dabei eine Funktion der spärlichen Besetzung des zugrundeliegenden, wahren DAG. Wir beweisen uniforme Konsistenz des Algorithmus für sehr hochdimensionale, spärlich besetzte DAG, bei denen die Anzahl Knoten verglichen mit der Anzahl Beobachtungen $n$ schnell wachsen kann ($O(n^a)$ für beliebiges $0 < a < \infty$). Die Annahme der spärlichen Besetzung verlangt, dass die Nachbarschaften des DAG von kleinerer Ordnung als die Anzahl Beobachtungen sind und ist somit relativ schwach.


Schliesslich betrachten wir die Anwendung des PC-Algorithmus auf das Problem der Variablenselektion in hochdimensionalen linearen Modellen, in denen die Anzahl der erklärenden Variablen die Anzahl der Beobachtungen übersteigt. Dabei stellen wir das Konzept der partiell sinngemäßen Verteilungen vor und diskutieren deren Rolle bei der

Alle der oben genannten Methoden sind im R-Paket `{pcalg}` verfügbar.
Chapter 1

Introduction

Graphical models are a very popular tool for analyzing and visualizing dependence structures. Given data on several variables, the goal is to find a graph, that represents some aspects of the dependence structure of the underlying probability distribution just as a hiking map represents some aspects of the terrain in real nature. An example of a graph is given in Fig.1(a). The nodes represent the random variables and the edges encode some sort of dependence, which has to be specified in addition to the graph. So, in order to encode dependence information in a graph, one has to have both some form of a graph and some rule of how to interpret the graph.

1.1 Different Graphs

In statistics, there are mainly two different types of graphs which are used for graphical modelling:

- Undirected graphs, as in Fig.1(a), consist of nodes and undirected edges between nodes.
Figure 1.1: Examples of graphs used in graphical modelling. For representing a dependence structure using graphical models, one needs a graph and a rule for interpreting the graph.
Directed graphs, as in Fig.1(b), consist of nodes and only directed edges between nodes.

Partially directed graphs, as in Fig.1(c), consist of nodes and directed and undirected edges between nodes. For convenience, we will also use bidirected edges to represent undirected edges.

1.2 Different Rules

There are many different rules for interpreting the graphs. We will give two examples.

In the first example, we will consider the Conditional Independence Graph (CIG). The interpretation rule of a CIG applies to undirected graphs. It says that two nodes, that are not directly connected by an edge are independent given the set of all remaining nodes. For instance, if we assume that Fig.1(a) is a CIG, the according rule for interpretation would tell us, that random variable 2 and 3 are independent when conditioning on the set of random variables $\{1, 4, 5\}$, since they are not connected by an edge. On the other hand, the random variables 3 and 5 are dependent given the set of random variables $\{1, 2, 4\}$, since there is an edge connecting node 3 and node 5.

In the second example, we will consider the skeleton of a Directed Acyclic Graph (DAG). A DAG is a directed graph, where one cannot trace a circle by following the directions of the arrows (see Fig.1(b)). When ignoring the directions of the arrows, one obtains the skeleton of the DAG (see Fig.1(a)). Thus, the skeleton of a DAG is an undirected graph. The interpretation rule of a skeleton states that two nodes are connected if they are dependent even if conditioning on any subset of the remaining nodes. In the last example, we interpreted Fig.1(a) as a CIG. Now we will interpret it as a skeleton. The according rule for interpretation would then tell us, that for example random variable 2 and 3 are independent if conditioning on some subset of $\{1, 4, 5\}$, since they are not connected. On the other hand, the rule for interpretation
would tell us that random variables 3 and 5 are dependent even if conditioning on any subset of \( \{1, 2, 4\} \), since they are connected. Note that the dependence implied by the skeleton is stricter than the dependence implied by a CIG: The edge between 3 and 5 in the CIG implies that 3 and 5 are dependent given the set \( \{1, 2, 4\} \). However, the edge between 3 and 5 in the skeleton implies that 3 and 5 are dependent when conditioning on \( \{1, 2, 4\} \) but also when conditioning on \( \{\}, \{1\}, \{2\}, \{4\}, \{1, 2\}, \{1, 4\}, \{2, 4\} \) (which are all subsets of \( \{1, 2, 4\} \)). In fact, the association between two connected nodes in a skeleton is so strong, that one is tempted to think of causality. This has been done in e.g. Spirtes et al. (2000) and we will also pick up this idea in chapter 4. We note in passing that there are also concise interpretation rules for directed and partially directed graphs. However, these rules are more complicated and we refer for a more detailed explanation and further references to chapter 2.

1.3 Graphical Models in Practice

Thus, in practice, a typical goal in graphical modelling is the following: Given a data set, which type of graph (directed or undirected; detailed structure) and which set of rules are most appropriate to represent the dependence structure described by the data?

In this thesis, we will focus on trying to find the skeleton or the DAG (as far as possible, see chapter 2) that fits the dependence structure of the data best. The basic idea for finding the skeleton is very simple: As we have seen above, an edge between two nodes occurs if and only if the two corresponding random variables (represented by the nodes) are dependent even if conditioning on every subset of the remaining nodes. The dependency of two random variables given a set of other random variables can be analyzed by using a statistical test. Hence, in practice, one could for each possible edge in a skeleton compute all tests that are necessary and thus decide, whether the edge should be present or absent. For a moderate or big graph, this will be a huge computational
effort. By rearranging the procedure a little bit, Spirtes et al. (2000) developed the PC-algorithm which reduces the computational burden immensely. Moreover, Spirtes et al. (2000) showed, that the algorithm is correct when the dependency information (which is used as input for the algorithm) is correct.

In practice, however, the dependency information of data is never known perfectly but has to be estimated. In the above example, to decide whether there should be an edge between node 3 and node 5, we would have to test the dependency given each subset of \{1, 2, 4\}. Since there are eight possible subsets, there are eight tests to compute in order to decide on this edge. In practice, every test has a small error, since we are not given the perfect dependency information but only data that represents a snapshot of the truth. Therefore it is natural to ask whether the errors add up so that finding the skeleton will go wrong when using real data. This problem gets much more severe in high dimensions, i.e., in asymptotic considerations where the number of nodes increases a lot and the number of samples does not increase as much. Imagine, we wanted to decide on the edge between node 3 and node 5 in a skeleton which consists of 100 nodes. Then we would have to make a test for each subset out of \{1, 2, 4, 6, 7, ..., 100\}. Since there are on the order of $10^{30}$ possible subsets, we would have to perform on the order of $10^{30}$ tests so that there could be many errors to add up. Fortunately, the PC-algorithm manages to reduce the number of tests a bit, but the basic problem remains. In chapter 2, we will show that for the case of multivariate normal distributions and under an assumption called “faithfulness”, the PC-algorithm still works, even if the input is estimated and thus not perfect, and that this result holds even in high dimensions. In chapter 3 we will analyze what happens if the underlying distribution is not Gaussian, i.e., we will analyze robust versions of the PC-algorithm. Finally, in chapter 4 we analyze what happens if the faithfulness assumption is violated. Moreover, we explore the potential of the PC-algorithm for variable selection, especially in the context of strong associations and causality.

The work of chapters 2, 3 and 4 are published in Kalisch and Bühlmann
(2007), Kalisch and Bühlmann (2008a) and Kalisch and Bühlmann (2008b), respectively.
Chapter 2

Estimating High-Dimensional Directed Acyclic Graphs with the PC-Algorithm

2.1 Introduction

Graphical models are a popular probabilistic tool to analyze and visualize conditional independence relationships between random variables (see Edwards, 2000; Lauritzen, 1996; Neapolitan, 2004). Major building blocks of the models are nodes (or vertices), which represent random variables and edges, which encode conditional dependence relations of the enclosing vertices. The structure of conditional independence among the random variables can be explored using Markov properties.

Of particular current interest are directed acyclic graphs (DAGs),
Chapter 2. Estimating High-Dimensional DAGs

containing directed rather than undirected edges, which restrict in a sense the conditional dependence relations. These graphs can be interpreted by applying the directed Markov property (see Lauritzen, 1996). When ignoring the directions of a DAG, we get the skeleton of a DAG. In general, it is different from the conditional independence graph (CIG), see Section 2.2.1. (Thus, estimation methods for directed graphs cannot be easily borrowed from approaches for undirected CIGs.) As we will see in Section 2.2.1, the skeleton can be interpreted easily and thus yields interesting insights into the dependence structure of the data.

Estimation of a DAG from data is difficult and computationally non-trivial due to the enormous size of the space of DAGs: the number of possible DAGs is super-exponential in the number of nodes (see Robinson, 1973). Nevertheless, there are quite successful search-and-score methods for problems where the number of nodes is small or moderate. For example, the search space may be restricted to trees as in MWST (Maximum Weight Spanning Trees; see Chow and Liu, 1968; Heckerman et al., 1995), or a greedy search is employed. The greedy DAG search can be improved by exploiting probabilistic equivalence relations, and the search space can be reduced from individual DAGs to equivalence classes, as proposed in GES (Greedy Equivalent Search, see Chickering, 2002b). Although this method seems quite promising when having few or a moderate number of nodes, it is limited by the fact that the space of equivalence classes is conjectured to grow super-exponentially in the nodes as well (Gillispie and Perlman, 2001). Bayesian approaches for DAGs, which are computationally very intensive, include Spiegelhalter et al. (1993) and Heckerman et al. (1995).

An interesting alternative to greedy or structurally restricted approaches is the PC-algorithm (after its authors, Peter and Clark) from Spirtes et al. (2000). It starts from a complete, undirected graph and deletes recursively edges based on conditional independence decisions. This yields an undirected graph which can then be partially directed and further extended to represent the underlying DAG (see Sections 2.2.2 and 2.2.3). The PC-algorithm runs in the worst case in exponential time (as a function of the number of nodes), but if the true underlying
2.1. Introduction

DAG is sparse, which is often a reasonable assumption, this reduces to a polynomial runtime.

In the past, interesting hybrid methods have been developed. Very recently, Tsamardinos et al. (2006) proposed a computationally very competitive algorithm. We also refer to their paper for a quite exhaustive numerical comparison study among a wide range of algorithms.

We focus in this paper on estimating the equivalence class and the skeleton of DAGs (corresponding to multivariate Gaussian distributions) in the high-dimensional context, that is, the number of nodes $p$ may be much larger than sample size $n$. We prove that the PC-algorithm consistently estimates the equivalence class and the skeleton of an underlying sparse DAG, as sample size $n \to \infty$, even if $p = p_n = O(n^a)$ ($0 \leq a < \infty$) is allowed to grow very quickly as a function of $n$.

Our implementation of the PC-algorithm is surprisingly fast, as illustrated in section 2.4.5, and it allows us to estimate a sparse DAG even if $p$ is in the thousands. For the high-dimensional setting with $p \gg n$, sparsity of the underlying DAG is crucial for statistical consistency and computational feasibility. Our analysis seems to be the first establishing a provable correct algorithm (in an asymptotic sense) for high-dimensional DAGs which is computationally feasible.

The question of consistency of a class of methods including the PC algorithm has been treated in Spirtes et al. (2000) and Robins et al. (2003) in the context of causal inference. They show that, assuming only faithfulness (explained in section 2), uniform consistency cannot be achieved, but pointwise consistency can. In this chapter, we extend this in two ways: We provide a set of assumptions which renders the PC-algorithm to be uniformly consistent. More importantly, we show that consistency holds even as the number of nodes and neighbors increases and the size of the smallest non-zero partial correlations decrease as a function of the sample size. Stricter assumptions than the faithfulness condition that render uniform consistency possible have been also pro-
posed in Zhang and Spirtes (2003). A rather general discussion on how many samples are needed to learn the correct structure of a Bayesian Network can be found in Zuk et al. (2006).

The problem of finding the equivalence class of a DAG has a substantial overlap with the problem of feature selection: If the equivalence class is found, the Markov Blanket of any variable (node) can be read of easily. Given a set of nodes $V$ and suppose that $M$ is the Markov Blanket of node $X$, then $X$ is conditionally independent of $V \setminus M$ given $M$. Thus, $M$ contains all and only the relevant features for $X$. In recent years, many other approaches to feature selection have been developed for high dimensions. See for example Goldenberg and Moore (2004) for an approach dealing with very high dimensions or Ng (1998) for a rather general approach dealing with bounds for generalization errors.

## 2.2 Finding the Equivalence Class of a DAG

### 2.2.1 Definitions and Preliminaries

A graph $G = (V, E)$ consists of a set of nodes or vertices $V = \{1, \ldots, p\}$ and a set of edges $E \subseteq V \times V$, that is, the edge set is a subset of ordered pairs of distinct nodes. In our setting, the set of nodes corresponds to the components of a random vector $\mathbf{X} = (X^{(1)}, \ldots, X^{(p)}) \in \mathbb{R}^p$. An edge $(i, j) \in E$ is called directed if $(i, j) \in E$ but $(j, i) \notin E$; we then use the notation $i \to j$. If both $(i, j) \in E$ and $(j, i) \in E$, the edge is called undirected; we then use the notation $i - j$. A directed acyclic graph (DAG) is a graph $G$ where all edges are directed and not containing any directed cycle.

If there is a directed edge $i \to j$, node $i$ is said to be a parent of node $j$. The set of parents of node $j$ is denoted by $pa(j)$. The adjacency set of a node $j$ in graph $G$, denoted by $adj(G, j)$, are all nodes $i$ which are directly connected to $j$ by an edge (directed or undirected). The elements of $adj(G, j)$ are also called neighbors of or adjacent to $j$. 
A probability distribution \( P \) on \( \mathbb{R}^p \) is said to be faithful with respect to a graph \( G \) if conditional independencies of the distribution can be inferred from so-called d-separation in the graph \( G \) and vice-versa. More precisely: consider a random vector \( \mathbf{X} \sim P \). Faithfulness of \( P \) with respect to \( G \) means: for any \( i, j \in V \) with \( i \neq j \) and any set \( s \subseteq V \),

\[ \mathbf{X}^{(i)} \text{ and } \mathbf{X}^{(j)} \text{ are conditionally independent given } \{ \mathbf{X}^{(r)}; \ r \in s \} \]

\[ \iff \text{ node } i \text{ and node } j \text{ are d-separated by the set } s. \]

The notion of d-separation can be defined via moral graphs; details are described in Lauritzen (1996, Prop. 3.25). We remark here that faithfulness is ruling out some classes of probability distributions. An example of a non-faithful distribution is given in Spirtes et al. (2000, Chapter 3.5.2). On the other hand, non-faithful distributions of the multivariate normal family (which we will limit ourselves to) form a Lebesgue null-set in the space of distributions associated with a DAG \( G \), see Meek (1995a).

The skeleton of a DAG \( G \) is the undirected graph obtained from \( G \) by substituting undirected edges for directed edges. A v-structure in a (partially) directed graph \( G \) is an ordered triple of nodes \( (i, j, k) \) such that \( G \) contains the directed edges \( i \to j \) and \( k \to j \), and \( i \) and \( k \) are not adjacent in \( G \).

It is well known that for a probability distribution \( P \) which is generated from a DAG \( G \), there is a whole equivalence class of DAGs with corresponding distribution \( P \) (see Chickering, 2002b, Section 2.2). Even when having infinitely many observations, we cannot distinguish among the different DAGs of an equivalence class. Using a result from Verma and J. Pearl (1990), we can characterize equivalent classes more precisely: Two DAGs are equivalent if and only if they have the same skeleton and the same v-structures.

A common tool for visualizing equivalence classes of DAGs are completed partially directed acyclic graphs (CPDAG). A partially directed acyclic graph (PDAG) is a graph where some edges are directed and some are undirected and one cannot trace a cycle by following the di-
rection of directed edges and any direction for undirected edges. Equivalence among PDAGs or of PDAGs and DAGs can be decided as for DAGs by inspecting the skeletons and v-structures. A PDAG is completed, if (1) every directed edge exists also in every DAG belonging to the equivalence class of the PDAG and (2) for every undirected edge \(i - j\) there exists a DAG with \(i \rightarrow j\) and a DAG with \(i \leftarrow j\) in the equivalence class.

PDAGs encode all independence informations contained in the corresponding equivalence class. It was shown in Chickering (2002a) that two CPDAGs are identical if and only if they represent the same equivalence class, that is, they represent a equivalence class uniquely.

Although the main goal is to identify the CPDAG, the skeleton itself already contains interesting information. In particular, if \(P\) is faithful with respect to a DAG \(G\),

there is an edge between nodes \(i\) and \(j\) in the skeleton of DAG \(G\)
\[
\Leftrightarrow \quad \text{for all } s \subseteq V \setminus \{i, j\}, \quad X^{(i)} \text{ and } X^{(j)} \text{ are conditionally dependent given } \{X^{(r)}; \ r \in s\},
\]  

(Spirtes et al., 2000, Th. 3.4). This implies that if \(P\) is faithful with respect to a DAG \(G\), the skeleton of the DAG \(G\) is a subset (or equal) to the conditional independence graph (CIG) corresponding to \(P\). (The reason is that an edge in a CIG requires only conditional dependence given the set \(V \setminus \{i, j\}\)). More importantly, every edge in the skeleton indicates some strong dependence which cannot be explained away by accounting for other variables. We think, that this property is of value for exploratory analysis.

As we will see later in more detail, estimating the CPDAG consists of two main parts (which will naturally structure our analysis): (1) Estimation of the skeleton and (2) partial orientation of edges. All statistical inference is done in the first part, while the second is just application of deterministic rules on the results of the first part. Therefore, we will put much more emphasis on the analysis of the first part.
If the first part is done correctly, the second will never fail. If, however, there occur errors in the first part, the second part will be more sensitive to it, since it depends on the inferential results of part (1) in greater detail. Therefore, when dealing with a high-dimensional setting (large $p$, small $n$), the CPDAG is harder to recover than the skeleton. Moreover, the interpretation of the CPDAG depends much more on the global correctness of the graph. The interpretation of the skeleton, on the other hand, depends only on a local region and is thus more reliable.

We conclude that, if the true underlying probability mechanisms are generated from a DAG, finding the CPDAG is the main goal. The skeleton itself oftentimes already provides interesting insights, and in a high-dimensional setting it might be interesting to use the undirected skeleton as an alternative target to the CPDAG when finding a useful approximation of the CPDAG seems hopeless.

As mentioned before, we will in the following describe two main steps. First, we will discuss the part of the PC-algorithm that leads to the skeleton. Afterwards we will complete the algorithm by discussing the extensions for finding the CPDAG. We will use the same format when discussing theoretical properties of the PC-algorithm.

### 2.2.2 The PC-Algorithm for Finding the Skeleton

A naive strategy for finding the skeleton would be to check conditional independencies given all subsets $s \subseteq V \setminus \{i, j\}$ (see formula (2.1)), that is, all partial correlations in the case of multivariate normal distributions as first suggested by Verma and J.Pearl (1990). This would become computationally infeasible and statistically ill-posed for $p$ larger than sample size. A much better approach is used by the PC-algorithm which is able to exploit sparseness of the graph. More precisely, we apply the part of the PC-algorithm that identifies the undirected edges of the DAG.
Population Version

In the population version of the PC-algorithm, we assume that perfect knowledge about all necessary conditional independence relations is available. We refer here to the PC-algorithm what others call the first part of the PC-algorithm; the other part is described in Algorithm 2 in Section 2.2.3.

Algorithm 1 The PC\textsubscript{pop}-algorithm

1: **INPUT:** Vertex Set $V$, Conditional Independence Information
2: **OUTPUT:** Estimated skeleton $C$, separation sets $S$ (only needed when directing the skeleton afterwards)
3: Form the complete undirected graph $\tilde{C}$ on the vertex set $V$.
4: $\ell = -1$; $C = \tilde{C}$
5: **repeat**
6: $\ell = \ell + 1$
7: **repeat**
8: Select a (new) ordered pair of nodes $i, j$ that are adjacent in $C$
   such that $|\text{adj}(C, i) \setminus \{j\}| \geq \ell$
9: **repeat**
10: Choose (new) $k \subseteq \text{adj}(C, i) \setminus \{j\}$ with $|k| = \ell$.
11: **if** $i$ and $j$ are conditionally independent given $k$ **then**
12: Delete edge $i, j$
13: Denote this new graph by $C$
14: Save $k$ in $S(i, j)$ and $S(j, i)$
15: **end if**
16: **until** edge $i, j$ is deleted or all $k \subseteq \text{adj}(C, i) \setminus \{j\}$ with $|k| = \ell$
   have been chosen
17: **until** all ordered pairs of adjacent variables $i$ and $j$ such that $|\text{adj}(C, i) \setminus \{j\}| \geq \ell$ have been considered
18: **until** for each ordered pair of adjacent nodes $i,j$: $|\text{adj}(C, i) \setminus \{j\}| < \ell$.

The (first part of the) PC-algorithm is given in Algorithm 1. The
maximal value of $\ell$ in Algorithm 1 is denoted by
\[
m_{\text{reach}} = \text{maximal reached value of } \ell. \tag{2.2}
\]
The value of $m_{\text{reach}}$ depends on the underlying distribution.

A proof that this algorithm produces the correct skeleton can be easily deduced from Theorem 5.1 in Spirtes et al. (2000). We summarize the result as follows.

**Proposition 1.** Consider a DAG $G$ and assume that the distribution $P$ is faithful to $G$. Denote the maximal number of neighbors by $q = \max_{1 \leq j \leq p} |\text{adj}(G, j)|$. Then, the PC$_{\text{pop}}$-algorithm constructs the true skeleton of the DAG. Moreover, for the reached level: $m_{\text{reach}} \in \{q - 1, q\}$.

A proof is given in Section 2.7.

**Sample Version for the Skeleton**

For finite samples, we need to estimate conditional independencies. We limit ourselves to the Gaussian case, where all nodes correspond to random variables with a multivariate normal distribution. Furthermore, we assume faithful models, which means that the conditional independence relations correspond to d-separations (and so can be read off the graph) and vice versa; see Section 2.2.1.

In the Gaussian case, conditional independencies can be inferred from partial correlations.

**Proposition 2.** Assume that the distribution $P$ of the random vector $\mathbf{X}$ is multivariate normal. For $i \neq j \in \{1, \ldots, p\}$, $\mathbf{k} \subseteq \{1, \ldots, p\} \setminus \{i, j\}$, denote by $\rho_{i,j|\mathbf{k}}$ the partial correlation between $\mathbf{X}^{(i)}$ and $\mathbf{X}^{(j)}$ given $\{\mathbf{X}^{(r)}; \ r \in \mathbf{k}\}$. Then, $\rho_{i,j|\mathbf{k}} = 0$ if and only if $\mathbf{X}^{(i)}$ and $\mathbf{X}^{(j)}$ are conditionally independent given $\{\mathbf{X}^{(r)}; \ r \in \mathbf{k}\}$.
Proof: The claim is an elementary property of the multivariate normal distribution, cf. Lauritzen (1996, Prop. 5.2.).

We can thus estimate partial correlations to obtain estimates of conditional independencies. The sample partial correlation \( \hat{\rho}_{i,j|k} \) can be calculated via regression, inversion of parts of the covariance matrix or recursively by using the following identity: for some \( h \in k \),

\[
\rho_{i,j|k} = \frac{\rho_{i,j|k\setminus h} - \rho_{i,h|k\setminus h}\rho_{j,h|k\setminus h}}{\sqrt{(1 - \rho_{i,h|k\setminus h}^2)(1 - \rho_{j,h|k\setminus h}^2)}}. \tag{2.3}
\]

In the following, we will concentrate on the recursive approach. For testing whether a partial correlation is zero or not, we apply Fisher’s \( z \)-transform

\[
Z(i, j|k) = \frac{1}{2} \log \left( \frac{1 + \hat{\rho}_{i,j|k}}{1 - \hat{\rho}_{i,j|k}} \right). \tag{2.4}
\]

Classical decision theory yields then the following rule when using the significance level \( \alpha \). Reject the null-hypothesis \( H_0(i, j|k) : \rho_{i,j|k} = 0 \) against the two-sided alternative \( H_A(i, j|k) : \rho_{i,j|k} \neq 0 \) if

\[
\sqrt{n - |k| - 3}|Z(i, j|k)| > \Phi^{-1}(1 - \alpha/2), \text{ where } \Phi(\cdot) \text{ denotes the cdf of } \mathcal{N}(0, 1).
\]

The sample version of the PC-algorithm is almost identical to the population version in Section 2.2.2.

**The PC-algorithm**

Run the PC\(_{pop}(m)\)-algorithm as described in Section 2.2.2 but replace in line 11 of Algorithm 1 the if-statement by

\[
\text{if } \sqrt{n - |k| - 3}|Z(i, j|k)| \leq \Phi^{-1}(1 - \alpha/2) \text{ then.}
\]

The algorithm yields a data-dependent value \( \hat{m}_{reach,n} \) which is the sample version of (2.2).
2.2. Finding the Equivalence Class of a DAG

The only tuning parameter of the PC-algorithm is $\alpha$, which is the significance level for testing partial correlations. See Section 2.4 for further discussion.

As we will see below in Section 2.3, the algorithm is asymptotically consistent even if $p$ is much larger than $n$ but the DAG is sparse.

2.2.3 Extending the Skeleton to the Equivalence Class

While finding the skeleton as in Algorithm 1, we recorded the separation sets that made edges drop out in the variable denoted by $S$. This was not necessary for finding the skeleton itself, but will be essential for extending the skeleton to the equivalence class. In Algorithm 2 we describe the work of Pearl (2000, p.50f) to extend the skeleton to a PDAG belonging to the equivalence class of the underlying DAG.

Algorithm 2 Extending the skeleton to a PDAG

INPUT: Skeleton $G_{skel}$, separation sets $S$
OUTPUT: PDAG $G$

for all pairs of nonadjacent variables $i, j$ with common neighbour $k$ do
  if $k \notin S(i, j)$ then
    Replace $i - k - j$ in $G_{skel}$ by $i \rightarrow k \leftarrow j$
  end if
end for

In the resulting PDAG, try to orient as many undirected edges as possible by repeated application of the following three rules:

R1 Orient $j - k$ into $j \rightarrow k$ whenever there is an arrow $i \rightarrow j$ such that $i$ and $k$ are nonadjacent.

R2 Orient $i - j$ into $i \rightarrow j$ whenever there is a chain $i \rightarrow k \rightarrow j$.

R3 Orient $i - j$ into $i \rightarrow j$ whenever there are two chains $i - k \rightarrow j$ and $i - l \rightarrow j$ such that $k$ and $l$ are nonadjacent.

The output of Algorithm 2 is a CPDAG, which was first proved by Meek (1995b)
2.3 Consistency for High-Dimensional Data

As in Section 2.2, we will first deal with the problem of finding the skeleton. Then, we will extend the result to finding the CPDAG.

2.3.1 Finding the Skeleton

We will show that the PC-algorithm from Section 2.2.2 is asymptotically consistent for the skeleton of a DAG, even if \( p \) is much larger than \( n \) but the DAG is sparse. We assume that the data are realizations of i.i.d. random vectors \( X_1, \ldots, X_n \) with \( X_i \in \mathbb{R}^p \) from a DAG \( G \) with corresponding distribution \( P \). To capture high-dimensional behavior, we will let the dimension grow as a function of sample size: thus, \( p = p_n \) and also the DAG \( G = G_n \) and the distribution \( P = P_n \). Our assumptions are as follows.

(A1) The distribution \( P_n \) is multivariate Gaussian and faithful to the DAG \( G_n \) for all \( n \).

(A2) The dimension \( p_n = O(n^a) \) for some \( 0 \leq a < \infty \).

(A3) The maximal number of neighbors in the DAG \( G_n \) is denoted by \( q_n = \max_{1 \leq j \leq p_n} |adj(G, j)| \), with \( q_n = O(n^{1-b}) \) for some \( 0 < b \leq 1 \).

(A4) The partial correlations between \( X^{(i)} \) and \( X^{(j)} \) given \( \{X^{(r)}; r \in k\} \) for some set \( k \subseteq \{1, \ldots, p_n\} \setminus \{i, j\} \) are denoted by \( \rho_{n;i,j|k} \). Their absolute values are bounded from below and above:

\[
\inf \{ |\rho_{n;i,j|k}| ; i, j, k \text{ with } \rho_{n;i,j|k} \neq 0 \} \geq c_n, \quad c_n^{-1} = O(n^d),
\]

for some \( 0 < d < b/2 \),

\[
\sup_{n;i,j,k} |\rho_{n;i,j|k}| \leq M < 1,
\]

where \( 0 < b \leq 1 \) is as in (A3).
Assumption (A1) is an often used assumption in graphical modeling, although it does restrict the class of possible probability distributions (see also third paragraph of Section 2.2.1); (A2) allows for an arbitrary polynomial growth of dimension as a function of sample size, that is, high-dimensionality; (A3) is a sparseness assumption and (A4) is a regularity condition. Assumptions (A3) and (A4) are rather minimal: note that with $b = 1$ in (A3), for example fixed $q_n = q < \infty$, the partial correlations can decay as $n^{-1/2+\varepsilon}$ for any $0 \leq \varepsilon \leq 1/2$. If the dimension $p$ is fixed (with fixed DAG $G$ and fixed distribution $P$), (A2) and (A3) hold and (A1) and the second part of (A4) remain as the only conditions. Recently, for undirected graphs the Lasso has been proposed as a computationally efficient algorithm for estimating high-dimensional conditional independence graphs where the growth in dimensionality is as in (A2) (see Meinshausen and Bühlmann, 2006). However, the Lasso approach can be inconsistent, even with fixed dimension $p$, as discussed in detail in Zhao and Yu (2006).

**Theorem 1.** Assume (A1)-(A4). Denote by $\hat{G}_{\text{ske}}(\alpha_n)$ the estimate from the (first part of the) PC-algorithm in Section 2.2.2 and by $G_{\text{ske}}$ the true skeleton from the DAG $G_n$. Then, there exists $\alpha_n \to 0$ ($n \to \infty$), see below, such that

$$
P[\hat{G}_{\text{ske}}(\alpha_n) = G_{\text{ske}}] = 1 - O(\exp(-Cn^{1-2d})) \to 1 \text{ for some } 0 < C < \infty,
$$

where $d > 0$ is as in (A4).

A proof is given in the Section 2.7. A choice for the value of the significance level is $\alpha_n = 2(1 - \Phi(n^{1/2}c_n/2))$ which depends on the unknown lower bound of partial correlations in (A4).

### 2.3.2 Extending the Skeleton to the Equivalence Class

As mentioned before, all inference is done while finding the skeleton. If this part is completed perfectly, that is, if there was no error while
testing conditional independencies (it is not enough to assume that the skeleton was estimated correctly), the second part will never fail (see Meek, 1995b). Therefore, we easily obtain:

**Theorem 2.** Assume (A1)-(A4). Denote by \( \hat{G}_{CPDAG}(\alpha_n) \) the estimate from the entire PC-algorithm and by \( G_{CPDAG} \) the true CPDAG from the DAG \( G \). Then, there exists \( \alpha_n \to 0 \) \( (n \to \infty) \), see below, such that

\[
P[\hat{G}_{CPDAG}(\alpha_n) = G_{CPDAG}] = 1 - O(\exp(-Cn^{1-2d})) \to 1 \ (n \to \infty) \text{ for some } 0 < C < \infty,
\]

where \( d > 0 \) is as in (A4).

A proof, consisting of one short argument, is given in the Section 2.7. As for Theorem 2, we can choose \( \alpha_n = 2(1 - \Phi(n^{1/2}c_n/2)) \).

By inspecting the proofs of Theorem 1 and Theorem 2, one can derive explicit error bounds for the error probabilities. Roughly speaking, this bounding function is the product of a linearly increasing and an exponentially decreasing term (in \( n \)). The bound is loose but for completeness, we present it in the appendix.

### 2.4 Numerical Examples

We analyze the PC-algorithm for finding the skeleton and the CPDAG using various simulated data sets. The numerical results have been obtained using the R-package `pcalg`. For an extensive numerical comparison study of different algorithms, we refer to Tsamardinos et al. (2006).

#### 2.4.1 Simulating Data

In this section, we analyze the PC-algorithm for the skeleton using simulated data. In order to simulate data, we first construct an adjacency matrix \( A \) as follows:
2.4. Numerical Examples

1. Fix an ordering of the variables.

2. Fill the adjacency matrix $A$ with zeros.

3. Replace every matrix entry in the lower triangle (below the diagonal) by independent realizations of Bernoulli($s$) random variables with success probability $s$ where $0 < s < 1$. We will call $s$ the sparseness of the model.

4. Replace each entry with a 1 in the adjacency matrix by independent realizations of a Uniform([0, 1]) random variable.

This then yields a matrix $A$ whose entries are zero or in the range [0, 1]. The corresponding DAG draws a directed edge from node $i$ to node $j$ if $i < j$ and $A_{ji} \neq 0$. The DAGs (and skeletons thereof) that are created in this way have the following property: $\mathbb{E}[N_i] = s(p - 1)$, where $N_i$ is the number of neighbors of a node $i$.

Thus, a low sparseness parameter $s$ implies few neighbors and vice-versa. The matrix $A$ will be used to generate the data as follows. The value of the random variable $X^{(1)}$, corresponding to the first node, is given by

$$
\epsilon^{(1)} \sim N(0, 1) \\
X^{(1)} = \epsilon^{(1)}
$$

and the values of the next random variables (corresponding to the next nodes) can be computed recursively as

$$
\epsilon^{(i)} \sim N(0, 1) \\
X^{(i)} = \sum_{k=1}^{i-1} A_{ik} X^{(k)} + \epsilon^{(i)} \quad (i = 2, \ldots, p),
$$

where all $\epsilon^{(1)}, \ldots, \epsilon^{(p)}$ are independent.
2.4.2 Choice of Significance Level

In section 2.3 we provided a value of the significance level \( \alpha_n = 2(1 - \Phi(n^{1/2}c_n/2)) \). Unfortunately, this value is not constructive, since it depends on the unknown lower bound of partial correlations in (A4). To get a feeling for good values of the significance level in the domain of realistic parameter settings, we fitted a wide range of parameter settings and compared the quality of fit for different significance levels.

Assessing the quality of fit is not quite straightforward, since one has to examine simultaneously both the true positive rate (TPR) and false positive rate (FPR) for a meaningful comparison. We follow an approach suggested by Tsamardinos et al. (2006) and use the Structural Hamming Distance (SHD). Roughly speaking, this counts the number of edge insertions, deletions and flips in order to transfer the estimated CPDAG into the correct CPDAG. Thus, a large SHD indicates a poor fit, while a small SHD indicates a good fit.

We fitted 40 replicates to all combinations of

- \( \alpha \in \{0.00005, 0.0001, 0.0005, 0.001, 0.005, 0.01, 0.05, 0.1\} \)
- \( p \in \{7, 15, 40, 70, 100\} \)
- \( n \in \{30, 100, 300, 1000, 3000, 10000, 30000\} \)
- \( E[N] \in \{2, 5\} \)

(where \( E[N] \) is the average neighborhood size) and evaluated the SHD. Each value of \( \alpha \) was used 40 times on each of the 70 possible parameter settings, and we then computed the average SHD over the 70 parameter settings.

The result is shown in Figure 2.1. One can see that the average SHD achieves a minimum in the region around \( \alpha = 0.005 \) and \( \alpha = 0.01 \). For higher or lower significance levels, the average SHD increases; the increase for bigger significance levels is much more pronounced. We
2.4. Numerical Examples

Figure 2.1: Average Structural Hamming Distance (ave SHD) with 95% confidence intervals. For each value of $\alpha$, the average SHD was averaged over 70 parameter settings using 40 replicates each. One can see that the average SHD is minimized for significance levels between $\alpha = 0.005$ and $\alpha = 0.01$.

analyzed the results of the simulation (see Figure 2.1) using pairwise Wilcoxon-Tests and Bonferroni correction. It turns out that $\alpha = 0.005$ and $\alpha = 0.01$ yield significantly lower average SHD than the other values of $\alpha$. In contrast, there is no significant difference between $\alpha = 0.005$ and $\alpha = 0.001$ (without Bonferroni correction). Of course, if $n$ was of different order of magnitude, a reasonable $\alpha$ should be a function of $n$ with $\alpha = \alpha_n \to 0$ ($n \to \infty$).
2.4.3 Performance for Different Parameter Settings

In this section, we’d like to give an overview over the performance in terms of the true positive rate (TPR) and false positive rate (FPR) for the skeleton and the SHD for the CPDAG. In order to keep the overview at a manageable size, we restrict the significance level to $\alpha = 0.01$. This was one of the two settings minimizing the average SHD as described in the previous section. The remaining parameters will be chosen as follows:

- $p \in \{7, 40, 100\}$
- $n \in \{30, 100, 300, 1000, 3000, 10000, 30000\}$
- $E[N] \in \{2, 5\}$

The overview is given in Figure 2.2 (again 40 replicates). As expected, the fit for a dense graph (triangles; $E[N] = 5$) is worse than the fit for a sparse graph (circles; $E[N] = 2$). While the TPR and the SHD show a clear tendency with increasing sample size, the behavior of FPR is not so clear. The latter seems surprising at first sight but is due to the fact that we used the same $\alpha = 0.01$ for all $n$.

2.4.4 Properties in High-Dimensional Setting

In this section, we study the behaviour of the error rates in a high-dimensional setting. The number of variables increases exponentially, the number of samples increases linearly and the expected neighborhood size increases sub-linearly. By inspecting the theory, we would expect the error rates to stay constant or even decrease. Table 2.4.1 shows the parameter setting of a small numerical study addressing this question. Note that $p$ increases exponentially, $n$ increases linearly and the expected neighborhood size $E[N] = 0.2\sqrt{n}$ increases sub-linearly. We used $\alpha = 0.05$ and the results are based on 20 simulation runs.
2.4. Numerical Examples

Figure 2.2: Performance of the PC-algorithm for different parameter settings, showing the mean of TPR, FPR and SHD together with 95% confidence intervals. The triangles represent parameter settings where $E[N] = 5$, while the circles represent parameter settings where $E[N] = 2$ ($\alpha = 0.01$, 40 replicates).
Table 2.4.1: The number of variables $p$ increases exponentially, the sample size $n$ increases linearly and the expected neighborhood size $E[N]$ increases sub-linearly. As supported by theory, the TPR increases and the FPR decreases in this setting. The results are based on using $\alpha = 0.05$, 20 simulation runs, and standard deviations are given in brackets.

Figure 2.3: While the number of variables $p$ increases exponentially, the sample size $n$ increases linearly and the expected neighborhood size $E[N]$ increases sub-linearly, the TPR increases and the FPR decreases. See Table 2.4.1 for a more detailed specification of the parameters.
Figure 2.3 shows boxplots of the TPR and the FPR over 20 replicates of this study. One can easily see that the TPR increases and the FPR decreases with sample size, although the number of \( p = p_n \) grows fast and \( E[N] \) grows slowly with \( n \). This confirms our theory very clearly.

We should note, that while the number of neighbors to a given variable may be growing almost as fast as \( n \), so that the number of neighbors is increasing with sample size, the percentage of true among all possible edges is going down with \( n \). So in one sense, the sparsity in terms of percentage of true edges of the DAGs is decreasing, and in another sense the sparsity in terms of the neighborhood size is increasing with \( n \).

### 2.4.5 Computational Complexity

Our theoretical framework in section 2.3 allows for large values of \( p \). The computational complexity of the PC-algorithm is difficult to evaluate exactly, but the worst case is bounded by

\[
O(p^{r_{reach}}) \quad \text{which is with high probability bounded by} \quad O(p^q) \quad (2.5)
\]

as a function of dimensionality \( p \); here, \( q \) is the maximal size of the neighborhoods as described in assumption (A3) in Section 2.3. We note that the bound may be very loose for many distributions. Thus, for the worst case where the complexity bound is achieved, the algorithm is computationally feasible if \( q \) is small, say \( q \leq 3 \), even if \( p \) is large. For non-worst cases, however, we can still do the computations for much larger values of \( q \) and fairly dense graphs, for example some nodes \( j \) having neighborhoods of size up to \(|adj(G, j)| = 30\).

We provide a small example of the processor time for estimating a CPDAG by using the PC-algorithm. The runtime analysis was done on an AMD Athlon 64 X2 Dual Core Processor 5000+ with 2.6 GHz and 4 GB RAM running on Linux and using R 2.4.1. The number of variables varied between \( p = 10 \) and \( p = 1000 \) while the number of samples was fixed at \( n = 1000 \). The sparseness was either \( E[N] = 2 \) or \( E[N] = 8 \). For each parameter setting, 10 replicates were used. In each case, the
significance level used in the PC-algorithm was $\alpha = 0.01$. The average processor time together with its standard deviation for estimating both the skeleton and the CPDAG is given in Table 2.4.2. Networks of $p = 1000$ nodes and 8 neighbors on average could be estimated in about 25 minutes, while networks with up to $p = 100$ nodes could be estimated in about a second. The additional time spent for finding the CPDAG from the skeleton is comparable for both neighborhood sizes and varies between a couple to almost 100 percent of the time needed to estimate the skeleton. The percentage tends to decrease with increasing number of variables.

Figure 2.4 gives a graphical impression of the results of this example. The sparse graphs (solid line with circles) were estimated faster than the dense graphs. While the line for the dense graph is very straight, the line for the sparse graphs has a positive curvature. Note, that this is a log-log plot; therefore, the slope of the lines indicates the exponent of polynomial growth. In this case, both curves follow very roughly a line with slope two indicating quadratic growth. The positive curvature of the solid line would indicate exponential growth; theory tells us, that this is not possible. One possible explanation for the positive curvature is the fact, that with increasing $p$, the maximal neighborhood size (which was not controlled in the simulation) is likely to increase. This would gradually increase the exponent in the polynomial growth of the upper bound in (2.5), thus yielding a positive curvature.

## 2.5 R-Package pcalg

The R-package pcalg can be used to estimate from data the underlying skeleton or equivalence class of a DAG. To use this package, the statistics software R needs to be installed. Both R and the R-package pcalg are available free of charge at [http://www.r-project.org](http://www.r-project.org). There are a number of other implementations of the PC-algorithm that are also worth mentioning: Hugin at [http://www.hugin.com](http://www.hugin.com), Murphy’s Bayes Network toolbox at
Figure 2.4: Average processor time over 10 runs together with 95% confidence intervals. Triangles correspond to dense ($E[N] = 8$), circles to sparse ($E[N] = 2$) underlying DAGs. We used $\alpha = 0.01$ and sample size $n = 1000$. 
### Table 2.4.2: The processor time (Athl. 64, 2.6 GHz, 4 GB) for estimating the skeleton ($G_{skel}$) or the PDAG ($G_{PDAG}$) for different DAGs in seconds.

<table>
<thead>
<tr>
<th>$p$</th>
<th>$E[N]$</th>
<th>$G_{skel}$</th>
<th>$G_{PDAG}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>10</td>
<td>2</td>
<td>0.037 (0.004)</td>
<td>0.072 (0.005)</td>
</tr>
<tr>
<td>10</td>
<td>8</td>
<td>0.093 (0.005)</td>
<td>0.124 (0.006)</td>
</tr>
<tr>
<td>30</td>
<td>2</td>
<td>0.15 (0.02)</td>
<td>0.23 (0.02)</td>
</tr>
<tr>
<td>30</td>
<td>8</td>
<td>0.84 (0.05)</td>
<td>0.93 (0.05)</td>
</tr>
<tr>
<td>50</td>
<td>2</td>
<td>0.33 (0.01)</td>
<td>0.48 (0.02)</td>
</tr>
<tr>
<td>50</td>
<td>8</td>
<td>2.2 (0.06)</td>
<td>2.4 (0.06)</td>
</tr>
<tr>
<td>100</td>
<td>2</td>
<td>1.03 (0.05)</td>
<td>1.49 (0.05)</td>
</tr>
<tr>
<td>100</td>
<td>8</td>
<td>8.9 (0.3)</td>
<td>9.4 (0.27)</td>
</tr>
<tr>
<td>300</td>
<td>2</td>
<td>8.3 (0.1)</td>
<td>13.8 (0.13)</td>
</tr>
<tr>
<td>300</td>
<td>8</td>
<td>89 (3)</td>
<td>95 (3)</td>
</tr>
<tr>
<td>1000</td>
<td>2</td>
<td>116 (0.5)</td>
<td>262 (0.8)</td>
</tr>
<tr>
<td>1000</td>
<td>8</td>
<td>1300 (60)</td>
<td>1445 (59)</td>
</tr>
</tbody>
</table>

In the following, we show an example of how to generate a random DAG, draw samples and infer from data the skeleton and the equivalence class of the original DAG using the R-package `pcalg`. The line width of the edges in the resulting skeleton and CPDAG can be adjusted to correspond to the reliability of the estimated dependencies. (The line width is proportional to the smallest value of $\sqrt{n - |k| - 3 |Z(i,j,k)|}$ causing an edge, see also 2.4. Therefore, thick lines are reliable).

```r
library(pcalg)
## define parameters
p <- 10 # number of random variables
n <- 10000 # number of samples
s <- 0.4 # sparseness of the graph
```
For simulating data as described in Section 2.4.1:

```r
## generate random data
set.seed(42)
g <- randomDAG(p,s) # generate a random DAG
d <- rmvDAG(n,g) # generate random samples

Then we estimate the underlying skeleton by using the function pcAlgo
and extend the skeleton to the PDAG or CPDAG by using the function
udag2pdag and udag2cpdag, respectively.

gSkel <-
    pcAlgo(d,alpha=0.05) # estimate of the skeleton
gPDAG <-
    udag2pdag(gSkel)
gCPDAG <-
    udag2cpdag(gSkel)

The CPDAG can also be estimated directly using

gCPDAG <-
    pcAlgo(d,alpha=0.05, directed=TRUE) # estimate of the CPDAG

The results can be easily plotted using the following commands:

plot(g)
plot(gSkel,zvalue.lwd=TRUE)
plot(gCPDAG,zvalue.lwd=TRUE)

The original DAG is shown in Figure 2.5(a). The estimated skeleton
and the estimated CPDAG are shown in Figure 2.5(b) and Figure 2.5(c),
respectively. Note the differing line width, which indicates the reliability
(z-values as in (2.4)) of the involved statistical tests (thick lines are
reliable).
Chapter 2. Estimating High-Dimensional DAGs

Figure 2.5: Plots generated using the R-package pcalg as described in section 2.5. (a) The true DAG. (b) The estimated skeleton using the R-function pcAlgo with $\alpha = 0.05$ and $n = 10000$. Line width encodes the reliability (z-values) of the dependence estimates (thick lines are reliable). (c) The estimated CPDAG using the R-function udag2cpdag. Double-headed arrows indicate undirected edges.

2.6 Conclusions

We show that the PC-algorithm is asymptotically consistent for the equivalence class of the DAG (represented by the CPDAG) and its skeleton with corresponding very high-dimensional, sparse Gaussian distribution. Moreover, the PC-algorithm is computationally feasible for such high-dimensional, sparse problems. Putting these two facts together, the PC-algorithm is established as a method (so far the only one) which is computationally feasible and provably correct, in the sense of uniform consistency, for high-dimensional DAGs. Sparsity, in terms of the maximal size of the neighborhoods of the true underlying DAG, is crucial for statistical consistency (assumption (A3) and Theorems 1 and 2) and for computational feasibility with at most a polynomial complexity (see formula (2.5)) as a function of dimensionality.

We emphasize that the skeleton of a DAG oftentimes provides inter-
esting insights, and in a high-dimensional setting it is quite sensible to use the undirected skeleton as a simpler but more realistic target rather than the entire CPDAG. Software for the PC-algorithm is available as explained in Section 2.5

2.7 Proofs and Appendix

2.7.1 Proof of Proposition 1

Consider $X$ with distribution $P$. Since $P$ is faithful to the DAG $G$, conditional independence of $X^{(i)}$ and $X^{(j)}$ given $\{X^{(r)}; r \in k\}$ ($k \subseteq V \setminus \{i, j\}$) is equivalent to d-separation of nodes $i$ and $j$ given the set $k$ (see Spirtes et al., 2000, Th. 3.3). Thus, the population PC$_{pop}$-algorithm as formulated in Section 2.2.2 coincides with the one from Spirtes et al. (2000) which is using the concept of d-separation, and the first claim about correctness of the skeleton follows from Spirtes et al. (2000, Th. 5.1., Ch. 13).

The second claim about the value of $m_{reach}$ can be proved as follows. First, due to the definition of the PC$_{pop}$-algorithm and the fact that it constructs the correct skeleton, $m_{reach} \leq q$. We now argue that $m_{reach} \geq q - 1$. Suppose the contrary. Then, $m_{reach} \leq q - 2$: we could then continue with a further iteration in the algorithm since $m_{reach} + 1 \leq q - 1$ and there is at least one node $j$ with neighborhood-size $|\text{adj}(G, j)| = q$: that is, the reached stopping level would be at least $q - 1$ which is a contradiction to $m_{reach} \leq q - 2$. \hfill $\Box$
2.7.2 Analysis of the PC-Algorithm

Analysis of Partial Correlations

We first establish uniform consistency of estimated partial correlations. Denote by \( \hat{\rho}_{i,j} \) and \( \rho_{i,j} \) the sample and population correlation between \( X^{(i)} \) and \( X^{(j)} \). Likewise, \( \hat{\rho}_{i,j|k} \) and \( \rho_{i,j|k} \) denote the sample and population partial correlation between \( X^{(i)} \) and \( X^{(j)} \) given \( \{X^{(r)}; r \in k\} \), where \( k \subseteq \{1, \ldots, p_n\} \setminus \{i, j\} \).

Many partial correlations (and non-partial correlations) are tested for being zero during the run of the PC\((m_n)\)-algorithm. For a fixed ordered pair of nodes \( i, j \), the conditioning sets are elements of

\[
K_{i,j}^{m_n} = \{k \subseteq \{1, \ldots, p_n\} \setminus \{i, j\} : |k| \leq m_n\}
\]

whose cardinality is bounded by

\[
|K_{i,j}^{m_n}| \leq Bp_{m_n}^{m_n} \text{ for some } 0 < B < \infty. \tag{2.6}
\]

**Lemma 1.** Assume (A1) (without requiring faithfulness) and

\[
\sup_{n, i \neq j} |\rho_{n;i,j}| \leq M < 1 \text{ (compare with (A4))}. \]

Then, for any \( 0 < \gamma \leq 2 \),

\[
\sup_{i,j} \mathbb{P}[|\hat{\rho}_{n;i,j} - \rho_{n;i,j}| > \gamma] \leq C_1 (n - 2) \exp \left( (n - 4) \log \left( \frac{4 - \gamma^2}{4 + \gamma^2} \right) \right),
\]

for some constant \( 0 < C_1 < \infty \) depending on \( M \) only.

Proof: We make substantial use of Hotelling (1953)’s work. Denote by \( f_n(\hat{\rho}, \rho) \) the probability density function of the sample correlation \( \hat{\rho} = \hat{\rho}_{n+1;i,j} \) based on \( n + 1 \) observations and by \( \rho = \rho_{n+1;i,j} \) the population correlation. (It is notationally easier to work with sample size \( n + 1 \); and we just use the abbreviated notations with \( \hat{\rho} \) and \( \rho \)). For \( 0 < \gamma \leq 2 \),

\[
\mathbb{P}[|\hat{\rho} - \rho| > \gamma] = \mathbb{P}[\hat{\rho} < \rho - \gamma] + \mathbb{P}[\hat{\rho} > \rho + \gamma].
\]
It can be shown, that \( f_n(r, \rho) = f_n(-r, -\rho) \), see Hotelling (1953, p.201). This symmetry implies,

\[
P_{\rho}[\hat{\rho} < \rho - \gamma] = P_{\rho}[\hat{\rho} > \hat{\rho} + \gamma] \text{ with } \hat{\rho} = -\rho.
\] (2.7)

Thus, it suffices to show that \( P[\hat{\rho} > \rho + \gamma] = P_{\rho}[\hat{\rho} > \rho + \gamma] \) decays exponentially in \( n \), uniformly for all \( \rho \).

It has been shown (Hotelling, 1953, p.201, formula (29)), that for \(-1 < \rho < 1\),

\[
P[\hat{\rho} > \rho + \gamma] \leq \frac{(n-1)\Gamma(n)}{\sqrt{2\pi\Gamma(n+\frac{1}{2})}} M_0(\rho + \gamma)(1 + \frac{2}{1 - |\rho|})
\] (2.8)

with (for \( \rho + \gamma \leq 1 \) and \( M_0(\rho + \gamma) = 0 \) otherwise)

\[
M_0(\rho + \gamma) = \int_{\rho+\gamma}^{1} (1 - \rho^2)^{\frac{n}{2}} (1 - x^2)^{\frac{n-3}{2}} (1 - \rho x)^{-n+\frac{1}{2}} dx
\]

\[
= \int_{\rho+\gamma}^{1} (1 - \rho^2)^{\frac{n+3}{2}} (1 - x^2)^{\frac{5}{2}} (1 - \rho x)^{-n-\frac{5}{2}} dx \text{ (using } \tilde{n} = n - 3)\]

\[
\leq \frac{(1 - \rho^2)^{\frac{3}{2}}}{(1 - |\rho|)^{\frac{3}{2}}} \int_{\rho+\gamma}^{1} (\frac{\sqrt{1 - \rho^2} \sqrt{1 - x^2}}{1 - \rho x})^{\tilde{n}} dx
\]

\[
\leq \frac{(1 - \rho^2)^{\frac{3}{2}}}{(1 - |\rho|)^{\frac{3}{2}}} 2 \max_{\rho+\gamma \leq x \leq 1} (\frac{\sqrt{1 - \rho^2} \sqrt{1 - x^2}}{1 - \rho x})^{\tilde{n}}. \] (2.9)

We will show now that \( g_\rho(x) = \frac{\sqrt{1 - \rho^2} \sqrt{1 - x^2}}{1 - \rho x} < 1 \) for all \( \rho + \gamma \leq x \leq 1 \) and \(-1 < \rho < 1\) (in fact, \( \rho \leq 1 - \gamma \) due to the first restriction). Consider

\[
\sup_{-1 < \rho < 1; \rho + \gamma \leq x \leq 1} g_\rho(x) = \sup_{-1 < \rho \leq 1 - \gamma} \frac{\sqrt{1 - \rho^2} \sqrt{1 - (\rho + \gamma)^2}}{1 - \rho(\rho + \gamma)}
\]

\[
= \frac{\sqrt{1 - \gamma^2}}{4} \sqrt{\frac{1 - \gamma^2}{4}} = \frac{4 - \gamma^2}{4 + \gamma^2} < 1
\]

for all \( 0 < \gamma \leq 2 \). (2.10)

Therefore, for \(-1 < -M \leq \rho \leq M < 1\) (see assumption (A4)) and using (2.8)-(2.10) together with the fact that \( \frac{\Gamma(n)}{\Gamma(n+\frac{1}{2})} \leq \text{const.} \) with
respect to \( n \), we have

\[
\mathbb{P}[\hat{\rho} > \rho + \gamma] \\
\leq \frac{(n - 1)\Gamma(n)}{\sqrt{2\pi\Gamma(n + \frac{1}{2})}} \frac{(1 - \rho^2)^{\frac{3}{2}}}{(1 - |\rho|)^{\frac{1}{2}}} 2 \left( \frac{4 - \gamma^2}{4 + \gamma^2} \right)^\frac{n}{2}(1 + \frac{2}{1 - |\rho|}) \\
\leq \frac{(n - 1)\Gamma(n)}{\sqrt{2\pi\Gamma(n + \frac{1}{2})}} \frac{1}{(1 - M)^{\frac{1}{2}}} 2 \left( \frac{4 - \gamma^2}{4 + \gamma^2} \right)^\frac{n}{2}(1 + \frac{2}{1 - M}) \\
\leq C_1(n - 1)\left( \frac{4 - \gamma^2}{4 + \gamma^2} \right)^\frac{n}{2} = C_1(n - 1) \exp((n - 3) \log(\frac{4 - \gamma^2}{4 + \gamma^2}))
\]

where \( 0 < C_1 < \infty \) depends on \( M \) only, but not on \( \rho \) or \( \gamma \). By invoking (2.7), the proof is complete (note that the proof assumed sample size \( n + 1 \)). \( \square \)

Lemma 1 can be easily extended to partial correlations, as shown by Fisher (1924), using projections for Gaussian distributions.

**Lemma 2.** (Fisher, 1924)

Assume (A1) (without requiring faithfulness). If the cumulative distribution function of \( \hat{\rho}_{n;i,j} \) is denoted by \( F(\cdot|n, \rho_{n;i,j}) \), then the cdf of the sample partial correlation \( \hat{\rho}_{n;i,j|k} \) with \( |k| = m < n - 1 \) is \( F(\cdot|n - m, \rho_{n;i,j|k}) \). That is, the effective sample size is reduced by \( m \).

A proof can be found in Fisher (1924); see also Anderson (1984). \( \square \)

Lemma 1 and 2 yield then the following.

**Corollary 1.** Assume (the first part of) (A1) and (the upper bound in) (A4). Then, for any \( \gamma > 0 \),

\[
\sup_{i,j,k \in K_{i,j}^m} \mathbb{P}[|\hat{\rho}_{n;i,j|k} - \rho_{n;i,j|k}| > \gamma] \\
\leq C_1(n - 2 - m_n) \exp \left( (n - 4 - m_n) \log(\frac{4 - \gamma^2}{4 + \gamma^2}) \right),
\]

for some constant \( 0 < C_1 < \infty \) depending on \( M \) from (A4) only.
The PC-algorithm is testing partial correlations after the z-transform \(g(\rho) = 0.5 \log((1 + \rho)/(1 - \rho))\). Denote by \(Z_{n;i,j|k} = g(\hat{\rho}_{n;i,j|k})\) and by \(z_{n;i,j|k} = g(\rho_{n;i,j|k})\).

**Lemma 3.** Assume the conditions from Corollary 1. Then, for any \(\gamma > 0\),

\[
\sup_{i,j,k \in K_{i,j}^{m,n}} P[|Z_{n;i,j|k} - z_{n;i,j|k}| > \gamma] \\
\leq O(n - m_n)(\exp((n - 4 - m_n) \log(\frac{4 - (\gamma/L)^2}{4 + (\gamma/L)^2})) + \exp(-C_2(n - m_n)))
\]

for some constant \(0 < C_2 < \infty\) and \(L = 1/(1 - (1 + M)^2/4)\).

Proof: A Taylor expansion of the z-transform \(g(\rho) = 0.5 \log((1 + \rho)/(1 - \rho))\) yields:

\[
Z_{n;i,j|k} - z_{n;i,j|k} = g'(\hat{\rho}_{n;i,j|k})(\hat{\rho}_{n;i,j|k} - \rho_{n;i,j|k}),
\]

where \(|\hat{\rho}_{n;i,j|k} - \rho_{n;i,j|k}| \leq |\hat{\rho}_{n;i,j|k} - \rho_{n;i,j|k}|\). Moreover, \(g'(\rho) = 1/(1 - \rho^2)\). By applying Corollary 1 with \(\gamma = \kappa = (1 - M)/2\) we have

\[
\sup_{i,j,k \in K_{i,j}^{m,n}} P[|\hat{\rho}_{n;i,j|k} - \rho_{n;i,j|k}| \leq \kappa] \\
> 1 - C_1(n - 2 - m_n) \exp(-C_2(n - m_n)).
\]

Since

\[
g'(\hat{\rho}_{n;i,j|k}) = \frac{1}{1 - \hat{\rho}_{n;i,j|k}^2} = \frac{1}{1 - (\rho_{n;i,j|k} + (\hat{\rho}_{n;i,j|k} - \rho_{n;i,j|k}))^2} \\
\leq \frac{1}{1 - (M + \kappa)^2}\text{ if } |\hat{\rho}_{n;i,j|k} - \rho_{n;i,j|k}| \leq \kappa,
\]

where we also invoke (the second part of) assumption (A4) for the last inequality. Therefore, since \(\kappa = (1 - M)/2\) yielding \(1/(1 - (M + \kappa)^2) = L\), and using (2.12), we get

\[
\sup_{i,j,k \in K_{i,j}^{m,n}} P[|g'(\hat{\rho}_{n;i,j|k})| \leq L] \\
\geq 1 - C_1(n - 2 - m_n) \exp(-C_2(n - m_n)).
\]
Since $|g'(\rho)| \geq 1$ for all $\rho$, we obtain with (2.11):

\[
\sup_{i,j,k \in K_{i,j}^{m,n}} \mathbb{P}[|Z_{n;i,j}|k - z_{n;i,j}|k| > \gamma] \\
\leq \sup_{i,j,k \in K_{i,j}^{m,n}} \mathbb{P}[|g'(\hat{\rho}_{n;i,j}|k)| > L] \\
+ \sup_{i,j,k \in K_{i,j}^{m,n}} \mathbb{P}[|\hat{\rho}_{n;i,j}|k - \rho_{n;i,j}|k| > \gamma/L]. \quad (2.14)
\]

Formula (2.14) follows from elementary probability calculations: for two random variables $U, V$ with $|U| \geq 1$ ($|U|$ corresponding to $|g'(\hat{\rho})|$ and $|V|$ to the difference $|\hat{\rho} - \rho|$),

\[
\mathbb{P}[|UV| > \gamma] = \mathbb{P}[|UV| > \gamma, |U| > L] + \mathbb{P}[|UV| > \gamma, 1 \leq |U| \leq L] \\
\leq \mathbb{P}[|U| > L] + \mathbb{P}[|V| > \gamma/L].
\]

The statement then follows from (2.14), (2.13) and Corollary 1. □

**Proof of Theorem 1**

For the analysis of the PC-algorithm, it is useful to consider a more general version as shown in Algorithm 3.

The PC-algorithm in Section 2.2.2 equals the PC$_{pop}(m_{reach})$-algorithm. There is the obvious sample version, the PC$(m)$-algorithm, and the PC-algorithm in Section 2.2.2 is then the same as the PC$(\hat{m}_{reach})$-algorithm, where $\hat{m}_{reach}$ is the sample version of (2.2).

The population version PC$_{pop}(m_n)$-algorithm when stopped at level $m_n = m_{reach,n}$ constructs the true skeleton according to Proposition 1. Moreover, the PC$_{pop}(m)$-algorithm remains to be correct when using $m \geq m_{reach,n}$. The following Lemma extends this result to the sample PC$(m)$-algorithm.

**Lemma 4.** Assume $(A1), (A2), (A3)$ where $0 < b \leq 1$ and $(A4)$ where $0 < d < b/2$. Denote by $\hat{G}_{skel,n}(\alpha_n, m_n)$ the estimate from the PC$(m_n)$-
Algorithm 3 The PC$_{pop}(m)$-algorithm

**INPUT:** Stopping level $m$, Vertex Set $V$, Conditional Independence Information

**OUTPUT:** Estimated skeleton $C$, separation sets $S$ (only needed when directing the skeleton afterwards)

Form the complete undirected graph $\tilde{C}$ on the vertex set $V$.

$\ell = -1; \quad C = \tilde{C}$

repeat
  $\ell = \ell + 1$

  repeat
    Select a (new) ordered pair of nodes $i,j$ that are adjacent in $C$
    such that $|\text{adj}(C, i) \setminus \{j\}| \geq \ell$
    repeat
      Choose (new) $k \subseteq \text{adj}(C, i) \setminus \{j\}$ with $|k| = \ell$.
      if $i$ and $j$ are conditionally independent given $k$ then
        Delete edge $i,j$
        Denote this new graph by $C$.
        Save $k$ in $S(i,j)$ and $S(j,i)$
      end if
    until edge $i,j$ is deleted or all $k \subseteq \text{adj}(C, i) \setminus \{j\}$ with $|k| = \ell$
    have been chosen
  until all ordered pairs of adjacent variables $i$ and $j$ such that
  $|\text{adj}(C, i) \setminus \{j\}| \geq \ell$ have been considered

until $\ell = m$ or for each ordered pair of adjacent nodes $i,j$: $|\text{adj}(C, i) \setminus \{j\}| < \ell$. 
algorithm in Section 2.2.2 and by \( G_{\text{skel},n} \) the true skeleton from the DAG \( G_n \). Moreover, denote by \( m_{\text{reach},n} \) the value described in (2.2). Then, for \( m_n \geq m_{\text{reach},n} \), \( m_n = O(n^{1-b}) \) \( (n \to \infty) \), there exists \( \alpha_n \to 0 \) \( (n \to \infty) \) such that
\[
\mathbb{P}[G_{\text{skel},n}(\alpha_n, m_n) = G_{\text{skel},n}] = 1 - O(\exp(-Cn^{1-2d})) \to 1 \quad (n \to \infty)
\] for some \( 0 < C < \infty \).

Proof: An error occurs in the sample PC-algorithm if there is a pair of nodes \( i, j \) and a conditioning set \( k \in K_{i,j}^{m_n} \) (although the algorithm is typically only going through a random subset of \( K_{i,j}^{m_n} \)) where an error event \( E_{i,j|k} \) occurs; \( E_{i,j|k} \) denotes that “an error occurred when testing partial correlation for zero at nodes \( i, j \) with conditioning set \( k \)”. Thus,
\[
\mathbb{P}[\text{an error occurs in the PC}(m_n)\text{-algorithm}] \leq \mathbb{P}\left[ \bigcup_{i,j,k \in K_{i,j}^{m_n}} E_{i,j|k} \right] \leq O(p_n^{m_n+2}) \sup_{i,j,k \in K_{i,j}^{m_n}} \mathbb{P}[E_{i,j|k}], \quad (2.15)
\]
using that the cardinality of the set \( |\{i, j, k \in K_{i,j}^{m_n}\}| = O(p_n^{m_n+2}) \), see also formula (2.6). Now
\[
E_{i,j|k} = E_{i,j|k}^{I} \cup E_{i,j|k}^{II}, \quad (2.16)
\]
where

- **type I error** \( E_{i,j|k}^{I} \): \( \sqrt{n - |k| - 3} |Z_{i,j|k}| > \Phi^{-1}(1 - \alpha/2) \)
  and \( z_{i,j|k} = 0 \),

- **type II error** \( E_{i,j|k}^{II} \): \( \sqrt{n - |k| - 3} |Z_{i,j|k}| \leq \Phi^{-1}(1 - \alpha/2) \)
  and \( z_{i,j|k} \neq 0 \).

Choose \( \alpha = \alpha_n = 2(1 - \Phi(n^{1/2}c_n/2)) \), where \( c_n \) is from (A4). Then,
\[
\sup_{i,j,k \in K_{i,j}^{m_n}} \mathbb{P}[E_{i,j|k}^{I}] = \sup_{i,j,k \in K_{i,j}^{m_n}} \mathbb{P}[|Z_{i,j|k} - z_{i,j|k}| > (n/(n - |k| - 3))^{1/2}c_n/2]
\leq O(n - m_n) \exp(-C_3(n - m_n)c_n^2), \quad (2.17)
\]
for some $0 < C_3 < \infty$ using Lemma 3 and the fact that $\log(\frac{4-\delta^2}{4+\delta^2}) \sim -\delta^2/2$ as $\delta \to 0$. Furthermore, with the choice of $\alpha = \alpha_n$ above,

$$
\sup_{i,j,k \in K_{i,j}^{m_n}} \mathbb{P}[E_{i,j,k}^H] = \sup_{i,j,k \in K_{i,j}^{m_n}} \mathbb{P}[|Z_{i,j,k}| \leq \sqrt{n/(n - |k| - 3)c_n/2}]
$$

$$
\leq \sup_{i,j,k \in K_{i,j}^{m_n}} \mathbb{P}[|Z_{i,j,k} - z_{i,j,k}| > c_n(1 - \sqrt{n/(n - |k| - 3)/2})],
$$

because $\inf_{i,j,k \in K_{i,j}^{m_n}} |z_{i,j,k}| \geq c_n$ since $|g(\rho)| \geq |\rho|$ for all $\rho$ and using assumption (A4). By invoking Lemma 3 we then obtain:

$$
\sup_{i,j,k \in K_{i,j}^{m_n}} \mathbb{P}[E_{i,j,k}^H] \leq O(n - m_n) \exp(-C_4(n - m_n)c_n^2) \quad (2.18)
$$

for some $0 < C_4 < \infty$. Now, by (2.15)-(2.18) we get

$$
\mathbb{P}[\text{an error occurs in the PC}(m_n)\text{-algorithm}]
\leq O(n^{m_n+2}(n - m_n) \exp(-C_5(n - m_n)c_n^2))
\leq O(n^{a(m_n+2)+1} \exp(-C_5(n - m_n)n^{-2d}))
= O \left( \exp \left( a(m_n + 2) \log(n) + \log(n) - C_5(n^{-2d} - m_n n^{-2d}) \right) \right) = o(1),
$$

because $n^{1-2d}$ dominates all other terms in the argument of the exp-function due to the assumption in (A4) that $d < b/2$. This completes the proof. \qed

Lemma 4 leaves some flexibility for choosing $m_n$. The PC-algorithm yields a data-dependent reached stopping level $\hat{m}_{reach,n}$, that is, the sample version of (2.2).

**Lemma 5.** Assume (A1)-(A4). Then,

$$
\mathbb{P}[\hat{m}_{reach,n} = m_{reach,n}] = 1 - O(\exp(-Cn^{1-2d})) \to 1 \quad (n \to \infty)
$$

for some $0 < C < \infty$, where $d > 0$ is as in (A4).

Proof: Consider the population algorithm $\text{PC}_{pop}(m)$: the reached stopping level satisfies $m_{reach} \in \{q_n - 1, q_n\}$, see Proposition 1. The
sample PC($m_n$)-algorithm with stopping level in the range of $m_{\text{reach}} \leq m_n = O(n^{1-b})$, coincides with the population version on a set $A$ having probability $P[A] = 1 - O(\exp(-Cn^{1-2d}))$, see the last formula in the proof of Lemma 4. Hence, on the set $A$, $\hat{m}_{\text{reach},n} = m_{\text{reach}} \in \{q_n - 1, q_n\}$. The claim then follows from Lemma 4.

Lemma 4 and 5 together complete the proof of Theorem 1.

Because there are faithful distributions which require $m_n = m_{\text{reach},n} \in \{q_n - 1, q_n\}$ for consistent estimation with the PC($m$)-algorithm, Lemma 5 indicates that the PC-algorithm, stopping at $\hat{m}_{\text{reach},n}$, yields with high probability the smallest $m = m_n$ which is universally consistent for all faithful distributions.

**Proof of Theorem 2**

The proof of Theorem 1 also covers the issue of choosing the correct separation sets $S$, that is, the probability of having the correct set $S$ goes to one as $n \to \infty$. Hence, the proof of Theorem 2 is completed.

**Bound for error probability of PC-algorithm**

$M$ and $c$ are the upper and lower bounds for partial correlations, as defined in section 2.3.1. $p$ is the number of variables, $q$ is the maximal size of neighbors, $n$ is the sample size. The significance level is chosen as suggested in the proofs, that is, $\alpha = 2(1 - \Phi(n^{1/2}c/2))$. By closely inspecting the proofs, one can derive the following upper bound for the error probability of the PC-algorithm:

$$\mathbb{P}[\hat{G} \neq G] \leq p^{q+2}C_1(n-1-q)(\exp(-C_2(n-q)) + \exp((n-4-q)f(L, \frac{c}{2})))$$

where $L = \frac{1}{1-(1+M)^2/4}$, $C_1 = \frac{1+2/(1-M)}{(1-M)^{q/2}}$, $C_2 = -\log\left(\frac{16-16(M^2)}{16+16(M^2)}\right)$ and $f(x, y) = \log\left(\frac{4-(y/x)^2}{4+(y/x)^2}\right)$. 

Chapter 3

Robust PC-Algorithm

3.1 Introduction

It was shown in Kalisch and Bühlmann (2007), that the PC-algorithm for Gaussian data is asymptotically consistent for finding the equivalence class or the skeleton of a DAG, even if the number of nodes is much larger than sample size. Here, we will propose a robustification of the Gaussian PC-algorithm. The latter is based on the maximum likelihood estimate for the covariance matrix of the variables. An effective robustification can be achieved by using a covariance estimator which is robust for every matrix element only: such elementwise robustness is feasible in high-dimensional settings and our results support that there is no need for simultaneous robustness of the whole covariance matrix. In fact, we prove asymptotic consistency and 50% breakdown point of the robustified PC-algorithm. Furthermore, we report on some finite sample numerical results. The robust PC-algorithm turns out to be very useful in presence of severely contaminated data or heavy outliers. Finally, our implementation of the robust PC-algorithm is computationally feasible for a large number of nodes, i.e., in the hundreds or thousands, if the underlying DAG is sparse.
3.2 Robustification of the PC-Algorithm

The only quantity that has to be estimated from data in order to run the PC-algorithm is the correlation among all pairs of variables $X^{(i)}$ and $X^{(j)}$. The correlation estimates are then used to iteratively compute estimates of partial correlations as in (2.3), which in turn are used to obtain z-values in (2.4). It is well known, that the standard sample correlation estimator (the Gaussian MLE) is not robust. Therefore, a small quantity of outliers suffices to completely distort the resulting graph. In order to robustify the PC-algorithm, we first note, that the correlation $\rho(X^{(i)}, X^{(j)}) = \rho_{i,j}$ can be written in terms of standard deviations $\sigma_X = \sqrt{Var(X)}$:

$$\rho_{i,j} = \rho_{X^{(i)}, X^{(j)}} = \frac{\sigma^2_{aX^{(i)}+bX^{(j)}} - \sigma^2_{aX^{(i)}-bX^{(j)}}}{\sigma^2_{aX^{(i)}+bX^{(j)}} + \sigma^2_{aX^{(i)}-bX^{(j)}}}$$  \hspace{1cm} (3.1)

where $a = \frac{1}{\sigma_{X^{(i)}}}$ and $b = \frac{1}{\sigma_{X^{(j)}}}$, see (Huber, 1981). To robustify the scale estimate, we replace the empirical standard deviation by a robust scale estimate. A good choice is the $Q_n$ estimator (Rousseeuw and Croux, 1993) which is defined as a scaled order statistics of differences in a sample of $n$ realisations $X_1, \ldots, X_n$:

$$Q_{n;X} = d\{|X_i - X_j|; i < j\}_{(k)},$$  \hspace{1cm} (3.2)

where $d = 2.2219$ is a constant factor and $k = \binom{h}{2} \approx \binom{\frac{n}{2}}{4}$ with $h = \lfloor n/2 \rfloor + 1$ roughly equal to half the number of observations. The $Q_n$ estimator has some attractive properties. Apart from its simple and explicit formula, the definition is suitable for asymmetric distributions. The breakdown point is 50% and the efficiency at Gaussian distributions is 82%. By using an algorithm by Rousseeuw and Croux (1993), $Q_n$ can be computed with $O(n)$ memory storage and $O(n \log n)$ essential operations. Thus, we obtain a robust estimate for the individual correlations by using

$$\hat{\rho}_{Q_n;i,j} = \frac{Q_{n;\alpha X^{(i)}+\beta X^{(j)}}^2 - Q_{n;\alpha X^{(i)}-\beta X^{(j)}}^2}{Q_{n;\alpha X^{(i)}+\beta X^{(j)}}^2 + Q_{n;\alpha X^{(i)}-\beta X^{(j)}}^2}$$  \hspace{1cm} (3.3)
3.3 Consistency of the PC-Algorithm

The robust version of partial correlation is, analogously to (2.3),

\[
\hat{\rho}_{Q_n;i,j|k} = \frac{\hat{\rho}_{Q_n;i,j|k|h} - \hat{\rho}_{Q_n;i,h|k|h}\hat{\rho}_{Q_n;j,h|k|h}}{\sqrt{(1 - \hat{\rho}_{Q_n;i,h|k|h}^2)(1 - \hat{\rho}_{Q_n;j,h|k|h}^2)}}. \tag{3.4}
\]

The corresponding z-value is then defined as

\[
Z_{Q_n}(i,j|k) = \frac{1}{2} \log \left( \frac{1 + \hat{\rho}_{Q_n;i,j|k}}{1 - \hat{\rho}_{Q_n;i,j|k}} \right). \tag{3.5}
\]

Finally, we define the robust PC-algorithm as follows.

**The robust PC-algorithm**

Run the PC-algorithm as described in section 2.2.2 but replace \(Z(i,j|k)\) by \(Z_{Q_n}(i,j|k)\).

3.3 Consistency of the PC-Algorithm

The standard PC algorithm on Gaussian data was shown to be asymptotically consistent for finding the equivalence class or the skeleton of a DAG (Kalisch and Buhlmann, 2007). This result also holds for high-dimensional settings where \(p_n\).

In this section, we will show that in the Gaussian case (i.e. uncontaminated model) and assuming a finite dimension \(p < \infty\), i.e. a stricter assumption than in Kalisch and Buhlmann (2007), the PC algorithm is consistent whenever every element of the covariance matrix is estimated consistently. Thus, the previous result is generalized under stricter conditions. This will yield the consistency of the robust PC-algorithm as a special case.

We will use the following assumptions:

(A1) The distribution \(P\) is multivariate Gaussian \(\mathcal{N}(\mu, \Sigma)\) with finite dimensionality \(p < \infty\).
(A2) The distribution $P$ is faithful to the DAG $G$.

(A3) The distribution $P$ is not degenerated, i.e., $\Sigma_{ii} > 0$ and $|\rho_{ij}| = |\Sigma_{ij}/(\Sigma_{ii}\Sigma_{jj})^{\frac{1}{2}}| < 1$ for all $i \neq j$.

Denote by $\hat{G}_{skel}(\alpha)$ the estimate of the skeleton using the PC-algorithm (standard or robust version) when using significance level $\alpha$. Moreover, we denote by $G_{skel}$ the skeleton of a DAG $G$.

**Theorem 3.** Assume (A1)-(A3) and let $\hat{\Sigma}_{ij}$ be any estimate of $\Sigma_{ij}$ which is (elementwise) consistent for all $i, j \in \{1, \ldots, p\}$ having convergence rate $n^{-r}$ ($r \leq \frac{1}{2}$), i.e., $n^r(\hat{\Sigma}_{ij} - \Sigma_{ij}) = O_p(1)$ for all $i, j$. Then, for any sequence $(\alpha_n)_{n \in \mathbb{N}}$ with $\frac{1-\Phi(n^{-r}+\frac{1}{2})}{\alpha_n} \to \infty$, $\frac{\alpha_n}{1-\Phi(\sqrt{n})} \to \infty$ ($n \to \infty$),

$$\mathbb{P}[\hat{G}_{skel}(\alpha_n) = G_{skel}] \to 1 \quad (n \to \infty).$$

A proof is given in section 3.6.

**Remark:** Typically, $r = \frac{1}{2}$ and hence we require $\alpha_n \to 0$, $\frac{\alpha_n}{1-\Phi(\sqrt{n})} \to \infty$ ($n \to \infty$).

The consistency result of Theorem 3 can be easily extended to the equivalence class of the DAG. We denote by $G_{CPDAG}$ the CPDAG of a DAG $G$.

**Theorem 4.** Assume (A1)-(A3) and let $\hat{\Sigma}_{ij}$ be any estimate of $\Sigma_{ij}$ which is (elementwise) consistent for all $i, j \in \{1, \ldots, p\}$ having convergence rate $n^{-r}$ ($r \leq \frac{1}{2}$), i.e., $n^r(\hat{\Sigma}_{ij} - \Sigma_{ij}) = O_p(1)$ for all $i, j$. Then, for any sequence $(\alpha_n)_{n \in \mathbb{N}}$ with $\frac{1-\Phi(n^{-r}+\frac{1}{2})}{\alpha_n} \to \infty$, $\frac{\alpha_n}{1-\Phi(\sqrt{n})} \to \infty$ ($n \to \infty$),

$$\mathbb{P}[\hat{G}_{CPDAG}(\alpha_n) = G_{CPDAG}] \to 1 \quad (n \to \infty).$$

A proof is given in section 3.6. Using Theorem 3, it is now easy to analyze the consistency properties of the robust PC-algorithm:

**Lemma 6.** Assume (A1). Then, the estimate $\hat{\rho}_{Q_n;i,j}$ defined in (3.3) and the corresponding covariance estimate $\hat{\Sigma}_{Q_n;i,j} = \hat{\rho}_{Q_n;i,j} Q_n;X^{(i)} Q_n;X^{(j)}$ are $\sqrt{n}$-consistent.
3.4. Numerical Results

A proof is given in section 3.6. Combining Theorem 3 and Lemma 6, we obtain:

**Corollary 2.** Assume (A1)-(A3). The robust PC-algorithm based on the $Q_n$-estimator is consistent in the sense of Theorem 3 and Theorem 4 (with $r = \frac{1}{2}$).

A proof is given in section 3.6. Furthermore, the breakdown point of the robust PC-algorithm can be easily found:

**Proposition 3.** The breakdown point of the robust PC-algorithm is 50%.

A proof is given in section 3.6.

3.4 Numerical Results

3.4.1 Simulating Data

The data simulating process is almost as in section 2.4.1. The only difference is the choice of error distributions. The value of the random variable $X^{(1)}$, corresponding to the first node, is given by

$$
\epsilon^{(1)} \sim F
$$

$$
X^{(1)} = \epsilon^{(1)}
$$

and the values of the next random variables (corresponding to the next nodes) can be computed recursively as

$$
X^{(i)} = \sum_{k=1}^{i-1} A_{ik} X^{(k)} + \epsilon^{(i)} (i = 2, \ldots, p),
$$

where all $\epsilon^{(1)}, \ldots, \epsilon^{(p)}$ are i.i.d., and $\epsilon^{(i)}$ is independent from $\{X^{(j)} : j < i\}$. Regarding the distribution $F$ of $\epsilon$, we consider either $N(0,1)$, $0.9N(0,1) + 0.1t_3(0,1)$ (i.e. 10% contamination by a t-distribution with three degrees of freedom) or $0.9N(0,1) + 0.1Cauchy(0,1)$ (i.e. 10% contamination by a standard Cauchy distribution).
3.4.2 Performance of the Robust PC-Algorithm

We compare the standard Gaussian and robust version of the PC-algorithm on Gaussian and contaminated Gaussian (10% $t_3$ or Cauchy) data on a wide range of parameter settings. To this end, we simulated each combination of the following variable settings:

- $n \in \{100, 500, 1000, 5000, 10000\}$
- $p \in \{10, 25, 50\}$
- $\alpha \in \{0.0001, 0.0003, 0.001, 0.003, 0.01, 0.03, 0.1, 0.3\}$
- $\mathbb{E}[N] \in \{3, 6\}$

For each of the possible 210 combinations, 30 replicates were produced. In order to investigate the influence of the single tuning parameter $\alpha$, we show the average Structural Hamming Distance (SHD; Tsamardinos et al., 2006) together with 95% confidence intervals of the simulation results grouped by various values of $\alpha$. Roughly speaking, the SHD counts the number of edge insertions, deletions and flips in order to transfer the estimated CPDAG into the correct CPDAG. Thus, a large SHD indicates a poor fit, while a small SHD indicates a good fit.

In Figure 3.1 (a) and Figure 3.1 (b) the rather similar results for the Gaussian distribution and the Gaussian distribution with 10% contamination from a $t_3$ distribution are shown. One can see that the standard Gaussian PC-algorithm performed better, since the average SHD is lower. The difference is highly significant (one-sided, paired Wilcoxon Test). For the standard Gaussian PC-algorithm, the values $\alpha = 0.01$ and $\alpha = 0.03$ yield a significantly lower average SHD (two sided Wilcoxon Test with Bonferroni correction) than for the remaining values of $\alpha$. For the robust PC-algorithm, the value of $\alpha = 0.01$ also seems to produce the lowest average SHD, but the result is not as clear (not significant). Therefore, when dealing with problems in the realm of the covered parameter settings, we would advocate $\alpha \approx 0.01$. 
### 3.4. Numerical Results

**Figure 3.1:** Comparison between standard Gaussian (triangles) and robust (circles) PC-algorithm: The average Structural Hamming Distance (SHD) over a wide range of parameter settings is shown versus the tuning parameter $\alpha$ (together with 95% confidence intervals). Without or with moderate outliers, the standard Gaussian PC-algorithm performs slightly better. If the outliers are very severe, the standard Gaussian PC-algorithm breaks down and the robust version is superior.
In Figure 3.1 (c), we show the corresponding result for Gaussian data with 10% contamination from a standard Cauchy distribution. The situation looks completely different than before. For values of $\alpha \leq 0.03$, the robust PC-algorithm performs significantly better than the standard Gaussian PC-algorithm while for $\alpha > 0.03$, the standard PC-algorithm performs better. For the standard Gaussian PC-algorithm, the average SHD for $\alpha = 0.1$ is significantly lower than for the remaining values of $\alpha$. For the robust PC-algorithm, $\alpha = 0.3$ and $\alpha = 0.1$ produce significantly worse results. For $\alpha \leq 0.03$, no choice of $\alpha$ leads to significantly better results than another. Note that by using the robust PC-algorithm, the SHD can be decreased substantially yielding a much better fit.

We also analyzed the behavior of the true positive rate (TPR) and the false positive rate (FPR) and show the typical results by using an example. We used $n = 100$, $p = 50$, $\alpha = 0.01$, average neighborhood size $\mathbb{E}[N] = 5$ and we conducted 100 repetitions. In each of the 100 replications, we chose a graphical structure and generated errors according to a standard normal distribution and according to a mixture of 90% standard normal and 10% standard Cauchy. We then used the standard and the robust PC-algorithm to estimate the underlying skeleton of the graph. The result is given in Figure 3.2. As can be seen in Figure 3.2(a), the TPR for the standard estimate drops dramatically if outliers are introduced, whereas it does not drop significantly for the robust method. Analogously, the FPR in Figure 3.2(b) increases significantly for the standard estimate, while it does not change significantly for the robust method. Furthermore, in the contaminated setting, the TPR for the robust method is significantly higher than for the standard method, while the FPR of both methods don’t vary much. Thus, the main advantage of the robust method lies in achieving a higher TPR while keeping the FPR rather constant.
3.4. Numerical Results

Figure 3.2: Example of typical behavior of TPR and FPR ($n = 100$, $p = 50$, $\alpha = 0.01$, $\mathbb{E}[N] = 5$ on 100 repetitions). In contrast to the standard method, the TPR and FPR of the robust method don’t change very much, when outliers are introduced. Moreover, the robust method achieves significantly higher TPR while keeping FPR at a level comparable to the standard method. (N: Standard Normal Error, C: 10% Cauchy, Std: Standard method, Qn: Robust method)
3.4.3 Computational Complexity

The computational complexity of the robust PC-algorithm is up to a \( \log(n) \)-factor the same as for the analogue with the standard Gaussian PC-algorithm, namely \( O(n \log(n) \max\{p^q, p^2\}) \) in the worst case.

We provide a small example of the processor time for estimating a CPDAG by using the PC-algorithm. The runtime analysis was done on an AMD Athlon 64 X2 Dual Core Processor 5000+ with 2.6 GHz and 4 GB RAM running on Linux and using R 2.4.1. The number of variables varied between \( p = 10 \) and \( p = 1000 \) while the number of samples was fixed at \( n = 1000 \). The sparseness was \( E[N] = 3 \). For each parameter setting, 3 replicates were used. In each case, the significance level used in the PC-algorithm was \( \alpha = 0.01 \). Figure 3.3 gives a graphical impression of the results of this example. The runtimes seem not to be influenced much by the type of contamination. For both uncontaminated and contaminated (\( t_3 \) or Cauchy) data, the standard Gaussian PC-algorithm is on average roughly by a factor of 25 faster. The average processor times together with their standard deviations for estimating the CPDAG are given in Table 3.4.1. We only give the values for Cauchy-contamination, since the values in the other cases are very similar.

<table>
<thead>
<tr>
<th>( p )</th>
<th>standard PC [s]</th>
<th>robust PC [s]</th>
</tr>
</thead>
<tbody>
<tr>
<td>10</td>
<td>0.07 (0.02)</td>
<td>1.53 (0.3)</td>
</tr>
<tr>
<td>30</td>
<td>0.59 (0.09)</td>
<td>14.4 (0.1)</td>
</tr>
<tr>
<td>100</td>
<td>3.4 (0.1)</td>
<td>151 (1)</td>
</tr>
<tr>
<td>300</td>
<td>31 (0.4)</td>
<td>1148 (192)</td>
</tr>
<tr>
<td>1000</td>
<td>614 (88)</td>
<td>10600 (860)</td>
</tr>
</tbody>
</table>

Table 3.4.1: Average processor time of the standard Gaussian and robust PC-algorithm for graphs of different sizes, with standard errors in parentheses. The standard PC-algorithm is on average roughly by a factor of 25 faster.

Using the standard and the robust PC-algorithm, graphs of \( p = 1000 \)
Figure 3.3: Average processor time of the standard Gaussian and robust PC-algorithm for graphs of different sizes and different kinds of contamination by outliers. The standard Gaussian PC-algorithm is on average roughly by a factor of 25 faster.
nodes could be estimated in about 10 minutes and 3 hours, while graphs with \( p = 100 \) nodes it took about 3 seconds and about 2 minutes, respectively.

### 3.4.4 Decision Heuristic

The simulation studies show that the standard PC-algorithm already is rather insensitive to outliers, provided, they are not too severe. The effect of very heavy outliers can be dramatically reduced by using the robust PC-algorithm; this increases the computational burden by roughly one order of magnitude.

We provide a simple method for deciding whether data at hand has worse outliers than a given reference distribution. Using this, we see two heuristics for deciding whether to use the robust version of the PC-algorithm or not. On the one hand, one could use the normal distribution as reference distribution and apply the robust PC-algorithm to all data that seem to contain more outliers than an appropriate normal distribution (Heuristic A). On the other hand, one could, inspired by the results of the simulation study in section 3.4.2, only want to apply the robust method in the case where the contamination is worse than a normal distribution with 10\% outliers from a \( t_3 \) distribution. Then, we would use a normal distribution with 10\% outliers from a \( t_3 \) distribution as reference distribution (Heuristic B).

In order to decide whether data has worse outliers than a given reference distribution, we proceed as follows. We compute a robust estimate of the covariance matrix of the data (e.g. OGK with Qn-estimator, see Maronna and Zamar (2002)) and simulate (several times) data from the reference distribution with this covariance matrix. For each dimension \( i \ (1 \leq i \leq p) \), we compute the ratio of standard deviation \( \sigma_i \) and a robust version of it \( s_i \) (e.g., Qn-estimator) and compute the average over all dimensions. (Since the main input for the PC-algorithm are correlation estimates which can be expressed in terms of scale estimates (as in section 3.2), we base our test statistics on scale estimates.) Thus, we ob-
tain the distribution of this averaged ratio \( R = \frac{1}{p} \sum_{i=1}^{p} \sigma_i/s_i \) under the null hypothesis that the data can be explained by the reference distribution with given covariance matrix. We now can test this null hypothesis by using the ratio computed with the current data set \( r = \frac{1}{p} \sum_{i=1}^{p} \tilde{\sigma}_i/\tilde{s}_i \) on a given significance level.

In order to test both heuristics, we tried to decide on using the robust PC-algorithm for samples drawn from a randomly generated DAG model with 10% contamination (\( t_3 \) and Cauchy for Heuristic A and Cauchy for Heuristic B), while the reference distribution is either normal for Heuristic A or normal with \( t_3 \) contamination for Heuristic B. As in section 3.4.2, the DAG models were generated using all combinations of the following variable settings:

- \( n \in \{100, 500, 1000, 5000, 10000\} \)
- \( p \in \{10, 25, 50\} \)
- \( \mathbb{E}[N] \in \{3, 6\} \)

For each possible combination, ten replicates were produced. When using Heuristic A, our method produces a true positive rate of 0.97 and a false positive rate of 0.05. For Heuristic B, the true positive rate is 1 and the false positive rate is 0.

For convenience, we provide the function \texttt{decHeuristic} in our R package which yields a boolean variable indicating whether the robust version should be used or not. Of course, our method only provides a heuristic and you might want to choose your favorite method for detecting outliers in high dimensions instead.

### 3.5 R-Package \texttt{pcalg}

As in section 2.5, we give a small example on how to use the robust functions of \texttt{pcalg}. 
library(pcalg)

## define parameters
p <- 10  # number of random variables
n <- 10000  # number of samples
s <- 0.2  # sparseness of the graph

For simulating data as described in Section 3.4.1:

## generate random data
set.seed(10)
g <- randomDAG(p,s)  # generate a random DAG
d <- rmvDAG(n,g,errDist=''mixt3'')  # generate random samples

Note that the option errDist allows for several different error distributions. We estimate the underlying skeleton by using the function pcAlgo and extend the skeleton to the CPDAG by using the function udag2cpdag.

# estimate of the skeleton
gSkel <- pcAlgo(d,alpha=0.05,corMethod=''QnStable'',
                directed=FALSE)

# extend the skeleton CPDAG
gCPDAG <- udag2cpdag(gSkel)

# estimate of the CPDAG directly
gSkel <- pcAlgo(d,alpha=0.05,corMethod=''QnStable'',
               directed=TRUE)

Note that the option corMethod allows for different robust and non-robust estimation methods for the correlation matrix. By setting the option directed to TRUE, one can directly estimate the CPDAG. The results can be easily plotted using the following commands:

plot(g)
plot(gSkel,zvalue.lwd=TRUE)
plot(gCPDAG,zvalue.lwd=TRUE)
The original DAG is shown in Figure 3.4(a). The estimated skeleton and the estimated CPDAG are shown in Figure 3.4(b) and Figure 3.4(c), respectively. Note the differing line width, which indicates the reliability (minimal z-values as in (2.4)) of the involved statistical tests (thick lines are reliable).

3.6 Proofs

3.6.1 Proof of Theorem 3 and Theorem 4

The z-values $Z_{i,j|k}$ are defined as in (2.4) or (3.5). Denote the empirical z-values by $\hat{Z}_{i,j|k}$, which are defined in terms of the corresponding empirical correlations $\hat{\rho}_{i,j|k}$, and they are functions of $\hat{\Sigma}_{i,j}$ themselves.

Since $\hat{\Sigma}_{i,j}$ is a consistent estimate, elementary calculations show that
for any $\gamma > 0$,
\[ \mathbb{P}[^{\hat{z}_{i,j} - z_{i,j}} > \gamma] \to 0 \quad (n \to \infty) \quad \forall i, j \in \{1, \ldots, p\}, \]
where $\hat{z}_{i,j}$ and $z_{i,j}$ are the z-transforms of $\hat{\rho}_{i,j}$ and $\rho_{i,j}$, respectively. Furthermore, since $p < \infty$ is finite, we obtain: for any $\gamma > 0$,
\[ \max_{i,j,k \in V \backslash \{i,j\}} \mathbb{P}[|^{\hat{z}_{i,j}|k} - z_{i,j}|k] > \gamma \to 0 \quad (n \to \infty), \quad (3.6) \]
analogous to the computation in (Wille and Bühlmann, 2006). An error occurs in the sample PC-algorithm if there exists a pair of nodes $i, j$ and a conditioning set $k \subset V \backslash \{i, j\}$ such that error event $E_{i,j|k} \cup E_{i,j|k}^{II}$ (see below) occurs. In more detail, for some suitable $c$ (see below), the error events are:

- type I error $E_{i,j|k}^I$: $|^{\hat{z}_{i,j}|k} > \frac{c}{2}$ and $z_{i,j}|k = 0$,
- type II error $E_{i,j|k}^{II}$: $|^{\hat{z}_{i,j}|k} \leq \frac{c}{2}$ and $z_{i,j}|k \neq 0$.

Using (3.6) and a proper choice of $c$ (see below)
\[ \max_{i,j,k} \mathbb{P}[E_{i,j|k}^I] = \max_{i,j,k} \mathbb{P}[|^{\hat{z}_{i,j}|k} - z_{i,j}|k] > \frac{c}{2} \to 0 \]
and
\[ \max_{i,j,k} \mathbb{P}[E_{i,j|k}^{II}] \leq \max_{i,j,k} \mathbb{P}[|^{\hat{z}_{i,j}|k} - z_{i,j}|k] > \frac{c}{2} \to 0. \]

Note, that $c$ exists (e.g. $c = \min_{i,j,k} \{ |z_{i,j}|k; z_{i,j}|k \neq 0 \}$). In fact, we can use any $c = c_n$ with $c_n \to 0$, $c_n n^r \to \infty$ ($n \to \infty$) ((3.6) also holds for $\gamma = \gamma_n$ with $\gamma_n n^r \to \infty$). This immediately translates to the specified range of the sequence $(\alpha_n)_{n \in \mathbb{N}}$ in Theorem 3.

Therefore, we finally get
\[ \mathbb{P}[^{\text{an error occurs in the PC-algorithm}}] \leq \mathbb{P}[ \bigcup_{i,j,k \in V \backslash \{i,j\}} E_{i,j|k}^I \cup E_{i,j|k}^{II} ] \leq C_p \max_{i,j,k \in V \backslash \{i,j\}} \mathbb{P}[E_{i,j|k}^I \cup E_{i,j|k}^{II}] \to 0 \quad (n \to \infty) \quad (3.7) \]
since $C_p < \infty$ using $p < \infty$. 


3.6.2 Proof of Theorem 4

As mentioned in section 2.2.3, due to the result of Meek (1995b), it is sufficient to estimate the correct skeleton and separation sets. The proof of Theorem 3 also covers the issue of choosing the correct separation sets $S$, that is, the probability of estimating the correct sets $S$ goes to one as $n \to \infty$. Hence, the proof of Theorem 4 is completed.

3.6.3 Proof of Lemma 6

Consider first a univariate sample $X_1, \ldots, X_n$ i.i.d. $\sim N(0, \sigma^2)$ (without loss of generality with mean 0). Define $h(X_i, X_j) = |X_i - X_j|$ and $H_n(y) = \frac{1}{n(n-1)} \sum_{i<j} I(h(X_i, X_j) \leq y)$. Then, the $Q_n$-estimator can be written as a functional $T$ of $H_n$:

$$Q_n = T(H_n) = dH_n^{-1}(q)$$

where $d = 2.2219$ is a constant and $q = \frac{h}{n} \binom{n/2}{h}$ with $h = \lfloor n/2 \rfloor + 1$. Due to the Gaussian assumption, the assumptions of Theorem 3.1 in Serfling (1984) hold, and we obtain

$$n^{1/2}[T(H_n) - \sigma] \to N(0, \tau_\infty^2)$$

for some limiting variance $\tau_\infty^2$. Therefore, $Q_n$ is a $\sqrt{n}$-consistent estimate of the standard deviation $\sigma$ of a univariate sample (choosing the appropriate constant $d$). It follows immediately that $\hat{\Sigma}_{Q_n}^{i,j}$ and $\hat{\rho}_{Q_n}^{i,j}$ are $\sqrt{n}$-consistent.

3.6.4 Proof of Proposition 3

Because the breakdown point of the $Q_n$-estimator for the scale of any of the $X^{(i)}$'s is 50%, also $\hat{\rho}_{Q_n}^{i,j}$ and $\hat{\rho}_{Q_n}^{i,j,k}$ in (3.3) and (3.4) respectively have breakdown point 50%. The claim then follows by the definition of the robust PC-algorithm.
Chapter 4

Variable Selection for High-Dimensional Models: Partial Faithful Distributions, Strong Associations and the PC-Algorithm

4.1 Introduction

The variable selection problem for high-dimensional models has recently gained a lot of attraction. A particular stream of research has focused on estimators and algorithms whose computation is feasible and provably correct (Meinshausen and Bühlmann, 2006; Zou, 2006; Zhao and Yu,
2006; Bunea et al., 2007; Candès and Tao, 2007; Meinshausen and Yu, 2007; van de Geer, 2007; Zhang and Huang, 2007; Huang et al., 2007; Wainwright, 2006; Wasserman and Roeder, 2007; Bickel et al., 2007; Candès and Plan, 2007). As such, these methods distinguish themselves very clearly from heuristic optimization of an objective function or stochastic simulation or search, e.g. MCMC, which are often not really exploiting a high-dimensional search space. Prominent examples of computationally feasible and provably correct (w.r.t. computation) methods are penalty-based approaches, including the Lasso (Tibshirani, 1996), the adaptive Lasso (Zou, 2006) or the Dantzig selector (Candes and Tao, 2007).

We propose here a method for linear models which is “diametrically opposed” to penalty-based schemes. Three reasons for another approach include the following: (i) from a theoretical perspective, we prove that in the framework of so-called partially faithful distributions, our method leads to consistent model selection for more general (random) design matrices than what has been shown for the Dantzig selector or the Lasso or the adaptive Lasso; (ii) it can be worthwhile to infer stronger concepts of associations than what is obtained from the usual regression coefficients; (iii) from a practical perspective, it can be very valuable to have a “diametrically opposed” method in the tool-kit for high-dimensional data analysis, raising the confidence for relevance of variables if they have been selected by say two or more very different methods. We will address all these reasons in our paper.

Our method is a simplification of the PC-algorithm (Spirtes et al., 2000) which has been proposed for estimating directed acyclic graphs. The simplification arises because selecting variables in a linear model is easier than assigning a directed association in a graphical model. We prove consistency for variable selection in high-dimensional linear models where the number of covariates can greatly exceed the sample size. For the ordinary problem of inferring the non-zero regression coefficients, we introduce and assume the framework of partially faithful distributions. Partial faithfulness is novel and we prove here that partial faithfulness arises naturally in the context of (high-dimensional)
linear models. Assuming such partial faithfulness in a linear model, which is arguably only a mild requirement, our simplified PC-algorithm is asymptotically consistent under rather ill-posed (random) designs; essentially, we only need that the variables are identifiable in the population case and there are no strong conditions on the coherence or minimal sparse eigenvalues of the design. The new results complement our earlier work on the PC-algorithm for high-dimensional acyclic directed graphs (Kalisch and Bühlmann, 2007). We focus here on regression which allows to change assumptions about directed associations and use the concept of partial faithfulness. Furthermore, we discuss examples whose distributions are not faithful. Causal relations and stronger notions of associations than what is represented by the regression coefficients can be important. In particular, when partial faithfulness fails to hold, these concepts distinguish themselves very clearly from the regression-type associations. We also prove that for non-partial-faithful distributions, the PC-algorithm is inferring some strong associations between the response variable and the covariates. Our approach can also be adapted for preliminary dimensionality reduction of the covariate space: we call it “correlation screening” and the method bears some relations to “sure independence screening” (Fan and Lv, 2007).

Moreover, the PC-algorithm is computationally feasible in high-dimensional problems: its computational complexity is crudely bounded by a polynomial in $p$, the dimension of the covariate space, and we illustrate that our implementation in R (CRAN, 1997 ff.) has about the same magnitude for computing time as the LARS-algorithm (Efron et al., 2004).

Finally, we compare our PC-algorithm with the Lasso and the elastic net (Zou and Hastie, 2005), and we demonstrate the usefulness of having “diametrically opposed” methods for analyzing a high-dimensional real data-set on riboflavin production from bacillus subtilis.
4.2 Gaussian Linear Model and Partial Faithfulness

We are considering here a class of probability distributions for linear models which satisfies a so-called partial faithfulness condition. Such a condition will be crucial for identifying the effective variables (in the sense of regression) with the PC-algorithm whose computational complexity is bounded by a polynomial in the number of covariates.

Consider the Gaussian linear model

$$X \sim \mathcal{N}_p(\mu_X, \Sigma_X),$$

$$Y|X \sim \mathcal{N}(\sum_{j=1}^{p} \beta_j X^{(j)}, \sigma^2)$$

(4.1)

First, we assume:

(A1) $\Sigma_X$ is strictly positive definite.

Note that (A1) implies identifiability of the regression parameters since $\beta = \Sigma_X^{-1} \gamma$, where $\beta = (\beta_1, \ldots, \beta_p)^T$ and $\gamma = (\text{Cov}(Y, X^{(1)}), \ldots, \text{Cov}(Y, X^{(p)}))^T$. Moreover, the following mild assumption is crucial for what follows. It is a condition on the structure of $\beta_j$ ($j = 1, \ldots, p$): to do so, we will use the framework where the non-zero coefficients are fixed realizations from a probability distribution.

(A2) Denote the active set by $\mathcal{A} \subseteq \{1, \ldots, p\}$ and by $\mathcal{A}^C$ its complement. The regression coefficients satisfy:

$$\beta_j = 0 \text{ for } j \in \mathcal{A}^C,$$

$$\{\beta_j; j \in \mathcal{A}\} \sim f(b)db,$$

where $f(\cdot)$ denotes a density in (a subset of) $\mathbb{R}^{\text{peff}}$, $\text{peff} = |\mathcal{A}|$, of an absolutely continuous distribution with respect to Lebesgue measure.
4.2. Gaussian Linear Model and Partial Faithfulness

Assumption (A2) says that the regression coefficients are either equal to zero or (fixed) realizations from an absolutely continuous distribution with respect to Lebesgue measure. Once the $\beta_j$’s are realized, we fix them such that they can be considered as deterministic in the Gaussian linear model (4.1). Our framework is different but loosely related to a Bayesian formulation treating the $\beta_j$’s as i.i.d. random variables from a prior distribution which is a mixture of point mass at zero and a density $f(\cdot)$ with respect to Lebesgue measure.

**Definition 1.** The Gaussian linear model (4.1) satisfies the **partial faithfulness** assumption if and only if for all $j \in \{1, \ldots, p\}$

$$\text{Parcor}(Y, X^{(j)}|X^{(S)}) = 0 \text{ for some } S \subseteq \{1, \ldots, p\} \setminus j \implies \beta_j = 0.$$  

**Theorem 1.** Consider the Gaussian linear model in (4.1) satisfying assumptions (A1) and (A2). Then, the partial faithfulness assumptions hold, almost surely (with respect to the distribution generating the non-zero regression coefficients, see assumption (A2)).

A proof is given in Section 4.8. Theorem 1 says that failure of partial faithfulness will have probability zero (i.e. Lebesgue measure zero). Our result is in the spirit of Spirtes et al. (2000, Th. 3.2), saying that non-partial-faithful Gaussian distributions for a directed acyclic graph have Lebesgue measure zero. To appreciate such results, consider the setting of our Theorem 1: the regression coefficients having values zero can arise in an arbitrary order (and they do concentrate on the value 0) and only the non-zero coefficients are required to arise from an absolutely continuous probability distribution where concentration on some particular value does not happen.

The concept of partial faithful distributions is often used in the graphical modeling literature. There, conditional dependencies of a probability distribution $P$ can be inferred from a graph thanks to some Markov condition. In general, the distribution $P$ may include other conditional independence relations than those entailed by or derived from the Markov condition. If that is not the case, i.e. if all conditional
dependencies can be read off the graph, the distribution is called faithful, see Spirtes et al. (2000). Since we focus only on partial correlations between the response \( Y \) and any other covariate \( X^{(j)} \) (but not some partial correlation between say \( X^{(j)} \) and \( X^{(k)} \) \((j \neq k)\)), we introduce in Definition 1 the new terminology of partial faithfulness. Note however, that the concepts of faithfulness and partial faithfulness are not directly compatible. In that sense, Theorem 1 is not a consequence of the result of Spirtes et al. (2000). A consequence of partial faithfulness is as follows.

**Proposition 1.** Consider the Gaussian linear model (4.1) satisfying the partial faithfulness condition. Then,

\[
\text{Parcor}(Y, X^{(j)}|X^{(S)}) \neq 0 \text{ for all } S \subseteq \{1, \ldots, p\} \setminus j \iff \beta_j \neq 0,
\]

for \( j \in \{1, \ldots, p\} \).

A proof is given in Section 4.8. Proposition 1 shows that an effective variable, which is an element of the active set \( \mathcal{A} = \{ j; \beta_j \neq 0 \} \) has a stronger interpretation in the sense that all corresponding partial correlations are different from zero when conditioning on any subset \( S \subseteq \{1, \ldots, p\} \setminus j \). In many applications, this is a desirable property, and a stronger concept for association which is linked more closely to some notion of causality (Spirtes et al., 2000); more details are given in Section 4.4.

### 4.2.1 Partial Correlation Screening Using Partial Faithfulness

If partial faithfulness holds, see Definition 1, we can exploit some immediate consequences for construction of algorithms for variable selection. We point out that popular methods like the Lasso (Tibshirani, 1996) or the Dantzig selector (Candès and Tao, 2007) are not taking advantage of partial faithfulness. Partial faithfulness says:

\[
\text{Parcor}(Y, X^{(j)}|X^{(S)}) = 0 \implies \beta_j = 0.
\]
The easiest relation, in particular when it comes to estimation, is with $S = \emptyset$:

$$\text{Cor}(Y, X^{(j)}) = 0 \implies \beta_j = 0. \quad (4.2)$$

We can do screening according to marginal correlations and build a first set of candidate active variables

$$\mathcal{A}^{[1]} = \{1 \leq j \leq p; \; \text{Cor}(Y, X^{(j)}) \neq 0\}.$$ We call this the step$_1$ active set or the correlation screening active set. We know by (4.2) that variables with corresponding correlations being equal to zero will be non-active, i.e. they can be dropped from the model. In other words, the true underlying active set $\mathcal{A} = \{j; \; \beta_j \neq 0\}$ satisfies

$$\mathcal{A} \subseteq \mathcal{A}^{[1]}.$$ Such covariance screening may reduce the dimensionality of the problem already by a substantial or even huge amount, and due to (4.3), we can use other variable selection methods on the reduced set of variables $\mathcal{A}^{[1]}$.

Furthermore, we can do screening with partial correlations of order one by using the relation: for $j \in \mathcal{A}^{[1]}$,

$$\text{Parcor}(Y, X^{(j)}|X^{(k)}) = 0 \text{ for some } k \neq j \implies \beta_j = 0. \quad (4.4)$$

That is, for checking whether the $j$th covariate remains in the model, we would additionally screen with all partial correlations of order one. As we will see in Section 4.3, it will be sufficient to use only conditioning variables $X^{(k)}$ which are elements of $\mathcal{A}^{[1]}$. Screening with partial correlations of order one using (4.4) leads to a smaller active set

$$\mathcal{A}^{[2]} = \{j \in \mathcal{A}^{[1]}; \; \text{Parcor}(Y, X^{(j)}|X^{(k)}) \neq 0 \text{ for all } k \in \mathcal{A}^{[1]}, \; k \neq j\} \subseteq \mathcal{A}^{[1]}.$$ This new step$_2$ active set $\mathcal{A}^{[2]}$ may have reduced the dimensionality of the original problem a lot. We can then continue screening using higher-order partial correlations, as will be described in Section 4.3.1, and we end up with a nested sequence of step$_m$ active sets

$$\mathcal{A}^{[1]} \supseteq \mathcal{A}^{[2]} \supseteq \ldots \supseteq \mathcal{A}^{[m]} \supseteq \ldots \supseteq \mathcal{A}. \quad (4.5)$$
A step\textsubscript{m} active set \(\mathcal{A}^{[m]}\) can be used as dimensionality reduction and any favored variable selection method could then be used for the reduced linear model with covariates corresponding to indices in \(\mathcal{A}^{[m]}\). Alternatively, we can use the sequence in (4.5) without applying additional variable selection methods. This will be described in Section 4.3.

### 4.3 Estimation Using the PC-Algorithm

A simplified version of the PC-algorithm (Spirtes et al., 2000) can be used to compute the sequence of step\textsubscript{m} active sets in (4.5).

#### 4.3.1 The Population Version of the PC-Algorithm

We assume first that perfect knowledge about partial correlations is available.

**Algorithm 4** The PC\textsubscript{pop}-algorithm

1: Start with the step\textsubscript{0} active set \(\mathcal{A}^{[0]} = \{1, \ldots, p\}\).
2: Set \(m = 1\). Do correlation screening, see (4.2), and build the step\textsubscript{1} active set
   \(\mathcal{A}^{[1]} = \{1 \leq j \leq p; \ Cor(Y, X^{(j)}) \neq 0\}\)
3: **repeat**
4: \(m = m + 1\). Construct the step\textsubscript{m} active set:
   \(\mathcal{A}^{[m]} = \{ j \in \mathcal{A}^{[m-1]}; \ Parcor(Y, X^{(j)}|X^{(S)}) \neq 0, \)
   for all \(S \subseteq \mathcal{A}^{[m-1]} \setminus \{j\} \) with \(|S| = m - 1\}.
5: **until** \(|\mathcal{A}^{[m]}| \leq m\).

The value of \(m\) which is reached by the algorithm is defined as
4.3. Estimation Using the PC-Algorithm

follows:

\[ m_{\text{reach}} = \min \{ m; |A^{[m]}| \leq m \}. \] (4.6)

**Proposition 2.** For the Gaussian linear model (4.1) satisfying (A1) and partial faithfulness, the population PC\textsubscript{pop}-algorithm identifies the true underlying active set, i.e. \( A^{[m_{\text{reach}}]} = A = \{ 1 \leq j \leq p; \beta_j \neq 0 \} \).

A proof is given in Section 4.8. Note that partial faithfulness is implied by assumption (A2). Correctness of the population PC\textsubscript{pop}-algorithm for directed acyclic graphs has been given by Spirtes et al. (2000, Th. 5.1).

### 4.3.2 Sample Version of the PC-Algorithm

For finite samples, we need to estimate partial correlations. The sample partial correlation \( \hat{\rho}_{Y,j|S} = \text{Parcor}(Y, X^{(j)}|X^{(S)}) \) and \( \hat{\rho}_{i,j|S} = \text{Parcor}(X^{(i)}, X^{(j)}|X^{(S)}) \) can be calculated recursively by using the following identity: for some \( k \in S \),

\[
\hat{\rho}_{Y,j|S} = \frac{\hat{\rho}_{Y,j|S \setminus k} - \hat{\rho}_{Y,k|S \setminus k} \hat{\rho}_{j,k|S \setminus k}}{\sqrt{(1 - \hat{\rho}_{Y,k|S \setminus k}^2)(1 - \hat{\rho}_{j,k|S \setminus k}^2)}}.
\]

For testing whether a partial correlation is zero or not, we apply Fisher’s Z-transform

\[
Z(Y, j|S) = \frac{1}{2} \log \left( \frac{1 + \hat{\rho}_{Y,j|S}}{1 - \hat{\rho}_{Y,j|S}} \right).
\] (4.7)

Classical decision theory yields then the following rule when using the significance level \( \alpha \) (Anderson, 1984, cf.). Reject the null-hypothesis \( H_0(Y, j|S) : \rho_{Y,j|S} = 0 \) against the two-sided alternative \( H_A(Y, j|S) : \rho_{Y,j|S} \neq 0 \) if \( \sqrt{n - |S| - 3} |Z(Y, j|S)| > \Phi^{-1}(1 - \alpha/2) \), where \( \Phi(\cdot) \) denotes the cdf of \( N(0, 1) \).

The sample version of the PC-algorithm is almost identical to the population version in Section 4.3.1.
The PC-algorithm

Run the PC$_{pop}$-algorithm as described in Section 4.3.1 but replace in steps 2 and 4 of Algorithm 4 the statements about Parcor($Y, X^{(j)}|X^{(S)}$) $\neq 0$ (including $S = \emptyset$) by

$$\sqrt{n - |S| - 3} |Z(Y, j|S)| > \Phi^{-1}(1 - \alpha/2).$$

The only tuning parameter of the PC-algorithm is $\alpha$, the significance level for testing partial correlations. The analogue to the reached value of $m$ in (4.6) is denoted by $\hat{m}_{reach}$.

The computational complexity of the PC-algorithm is difficult to evaluate exactly, but the worst case is bounded by

$$O(np^{\hat{m}_{reach}})$$

which is with high probability bounded by $O(np^{peff})$, (4.8)

where $peff = |A|$, see Kalisch and Bühlmann (2007). Thus, the PC-algorithm is polynomial in $p$. In fact, the bound in (4.8) is often extremely loose and we can easily use the algorithm for problems where $p \approx 100 - 5'000$, as demonstrated in Section 4.6.

4.4 Failure of Partial Faithfulness and Measures of Association

By Theorem 1, failure of partial faithfulness happens for very specific parameter constellations in the linear model (4.1), i.e. the non-zero coefficients do not arise from a continuous probability distribution. We give two examples.

\textbf{Example 1. Consider a Gaussian linear model}

$$Y = X^{(1)} - X^{(2)} + \varepsilon,$$

$$X^{(2)} = X^{(1)} + \gamma,$$
4.4. Failure of partial faithfulness

where $X^{(1)}$, $\gamma$, $\varepsilon$ are i.i.d. $\sim N(0, \sigma^2)$. This is a linear model as in (4.1) with a specific parameter constellation for the regression parameters. It can be easily calculated that

$$\text{Cor}(Y, X^{(1)}) = 0, \quad \text{Parcor}(Y, X^{(1)}|X^{(2)}) \neq 0,$$

and hence, partial faithfulness fails to hold.

**Example 2.** Consider a Gaussian moving average model from time series:

$$X_t = \theta_1 \varepsilon_{t-1} + \varepsilon_t, \quad t \in \mathbb{Z},$$

where $\{\varepsilon_t; \ t \in \mathbb{Z}\}$ is a sequence of i.i.d. variables $\varepsilon_t \sim N(0, \sigma^2)$, and $|\theta_1| < 1$ a parameter. In terms of (auto-)regression, the model can be written as

$$X_t = \sum_{j=1}^{\infty} (-\theta_1)^j X_{t-j} + \varepsilon_t, \quad t \in \mathbb{Z}$$

and hence, using $Y = X_t$, this is a linear model with $p = \infty$. We focus now only on three variables $\{Y = X_t, X_{t-1}, X_{t-2}\}$ corresponding to one response and two covariates. It is well known that

$$\text{Cor}(Y, X_{t-2}) = \text{Cor}(X_t, X_{t-2}) = 0, \quad \text{Parcor}(Y, X_{t-2}|X_{t-1}) = \text{Parcor}(X_t, X_{t-2}|X_{t-1}) \neq 0,$$

(Brockwell and Davis, 1991, cf.). Thus, this is another example where partial faithfulness does not hold.

The PC-algorithm would fail in both examples: it would drop the variable $X^{(1)}$ in Example 1 or $X_{t-2}$ in Example 2 from the active set because the corresponding correlation is zero. The reason for failure though is - from a certain perspective - not undesirable. In fact, as described below in the continuation of Example 1, there is no causal relation between the variables $Y$ and $X^{(1)}$ in the sense of the intervention framework with the $\text{do}(\cdot)$-operator from Pearl (2000). Therefore, in a causal sense, the PC-algorithm would correctly declare no relation.
The following definitions of associations between the response $Y$ and some of the covariates $X^{(j)}$ are useful:

\[ A = \{ j; \ Parcor(Y, X^{(j)}|X^{(1,\ldots,p)\setminus j}) \neq 0 \} = \{ j; \beta_j \neq 0 \}, \]
\[ A_{\text{strong}} = \{ j; \ Parcor(Y, X^{(j)}|X^{(S)}) \neq 0 \text{ for all } S \subseteq \{1,\ldots,p\} \setminus j \}, \]
\[ A_{\text{strong-endo}} = \max\{ B \subseteq \{1,\ldots,p\}; \ Parcor(Y, X^{(j)}|X^{(S)}) \neq 0 \text{ for all } j \in B \text{ and all } S \subseteq B \setminus j \}. \]

The set $A$ is the usual active set from regression containing the covariates having regression coefficients different from zero; the set $A_{\text{strong}}$ contains associations with a stronger notion, requiring that partial correlations remain non-zero when conditioning on any subset of covariates; and finally, the set $A_{\text{strong-endo}}$ requires that partial correlations remain zero when conditioning on any subset of “endogenous” covariates which are associated with the response $Y$. Because there are fewer conditioning sets involved in $A$ or $A_{\text{strong-endo}}$ than in $A_{\text{strong}}$, the following holds in general:

\[ A_{\text{strong}} \subseteq A, \quad A_{\text{strong}} \subseteq A_{\text{strong-endo}}. \quad (4.9) \]

Furthermore,

\[ A_{\text{strong}} = A_{\text{strong-endo}} = A \text{ for partial faithful distributions.} \quad (4.10) \]

The equality $A = A_{\text{strong}}$ follows from Proposition 1, and the equality $A_{\text{strong-endo}} = A$ follows exactly as in the proof of Proposition 1. For non-partial-faithful distributions, the equalities in (4.10) fail.

In general (for non-partial-faithful distributions), the notions of associations in $A_{\text{strong}}$ and $A_{\text{strong-endo}}$ are more of a causal nature than in $A$. In fact, $A_{\text{strong-endo}}$ is in Example 1 a strong enough measure for causality.

**Example 1 (continued).**

For the linear model in Example 1, it is easy to see that $A_{\text{strong}} = A_{\text{strong-endo}} = \{2\}$. That is, only the second covariate $X^{(2)}$ is strongly
associated with $Y$. In addition, if assuming a directed acyclic graph as in Figure 4.1 for generating the model, $A_{\text{strong}} = A_{\text{strong-endo}}$ coincides with the set of causal variables in the sense of the do($\cdot$) operator from Pearl (2000). That is, for the distribution of $Y$ with and without intervention, $P(Y|\text{do}(X^{(1)} = u)) = P(Y)$ for all values $u$ while $P(Y|\text{do}(X^{(2)} = u)) \neq P(Y)$ for some value $u$.

![Directed acyclic graphs corresponding to Example 1.](image)

**Figure 4.1:** Directed acyclic graphs corresponding to Example 1.

The following holds in the context of potentially non-partial-faithful distributions.

**Proposition 3.** Consider the Gaussian linear model (4.1) satisfying (A1). Then, the population PC_{pop}-algorithm satisfies

$$A_{\text{strong}} \subseteq A^{[\text{mreach}]} \subseteq A_{\text{strong-endo}}.$$  

A proof is given in Section 4.8. Proposition 3 says that in the context of potentially non-partial-faithful distributions, the PC-algorithm identifies stronger associations than what is given by $A_{\text{strong-endo}}$. Note that for the example, the strong-endogenous associations coincide with the strong associations and with the “causal” effects.
4.5 Asymptotic Consistency in High Dimensions

We will show that the PC-algorithm from Section 4.3.2 is asymptotically consistent for variable selection, even if $p$ is much larger than $n$ but assuming that the true underlying linear model is sparse. We consider the Gaussian linear model in (4.1). To capture high-dimensional behavior, we will let the dimension grow as a function of sample size: thus, $p = p_n$ and also the distribution of

$$(Y, X) \sim P_n = N_{p_n+1}(\mu_{Y, X; n}, \Sigma_{Y, X; n})$$

changes with $n$ which includes that the regression coefficient vectors $\beta = \beta_n$ are depending on $n$.

4.5.1 Consistency with Partially Faithful Distributions

Our assumptions are as follows.

(B1) The distribution $P_n$ satisfies the partial faithfulness condition (see Definition 1) and assumption (A1) for all $n$.

(B2) The dimension $p_n = O(n^a)$ for some $0 < a < \infty$.

(B3) The cardinality of the active set $\text{peff}_n = |A_n| = |\{1 \leq j \leq p_n; \beta_{j,n} \neq 0\}|$ satisfies: $\text{peff}_n = O(n^{1-b})$ for some $0 < b \leq 1$.

(B4) The partial correlations $\text{Parcor}_n(Y, X(j)|X(S)) = \rho_n(Y, j|S)$ satisfy:

$$\inf\{|\rho_n(Y, j|S)|; 1 \leq j \leq p_n, S \subseteq \{1, \ldots, p_n\} \setminus j$$

with $\rho_n(Y, j|S) \neq 0 \} \geq c_n, c_n^{-1} = O(n^d)$ for some $0 < d < b/2$,

where $0 < b \leq 1$ is as in (A3).
(B5) The partial correlations \( \text{Parcor}_n(Y, X^{(j)}|X^{(S)}) = \rho_n(Y, j|S) \) satisfy:

\[
\sup_{n,j,S \subseteq \{1, \ldots, p_n\} \setminus j} |\rho_n(Y, j|S)| \leq M < 1.
\]

A more detailed discussion of these assumptions is given in Section 4.5.1.

Denote the active set by \( A_n = \{1 \leq j \leq p_n : \beta_{j,n} \neq 0\} \) and by \( \hat{A}_n(\alpha) \) the estimate from the PC-algorithm in Section 4.3.2 with tuning parameter \( \alpha \).

**Theorem 2.** Consider the Gaussian linear model (4.1) and assume (B1)-(B5). Then, there exists \( \alpha_n \to 0 \) \((n \to \infty)\), see below, such that the PC-algorithm satisfies:

\[
P[\hat{A}_n(\alpha) = A_n] = 1 - O(\exp(-Cn^{1-2d})) \to 1 \quad (n \to \infty)
\]

for some \( 0 < C < \infty \), where \( d > 0 \) is as in (B4).

A proof is given in Section 4.8. It should be noted that for distributions which satisfy the partial faithfulness condition, as required by assumption (B1), the strong and the usual measures of association agree, i.e. \( A_{\text{strong},n} = A_{\text{strong-endo},n} = A_n \) and hence, the PC-algorithm consistently infers the strong associations. A choice for the value of the significance level, leading to consistency, is \( \alpha_n = 2(1 - \Phi(n^{1/2}c_n/2)) \) which depends on the unknown lower bound of partial correlations in (B4).

Theorem 2 is complementing our earlier work on the PC-algorithm for high-dimensional acyclic directed graphs. Here, we assume undirected associations from regression, which are much more widely used in statistical practice, and we require partial instead of full faithfulness of a multivariate distribution.

**Discussion and Comparison of Conditions**

There is a substantial amount of recent work on high-dimensional and computationally tractable variable selection. Most of these works con-
sider (versions of) the Lasso (Tibshirani, 1996) but some discuss also the Dantzig selector (Candès and Tao, 2007). None of these two methods exploits partial faithfulness and thus, it is interesting to compare our conditions with existing results about penalty-based methods.

For the Lasso, it is proved in (Meinshausen and Bühlmann, 2006) that a so-called “neighborhood stability” condition is sufficient and “almost” necessary for consistent variable selection (the word “almost” refers to the fact that a strict inequality “<” appears in the sufficient condition whereas for necessity, the corresponding relation is a “≤” relation). In Zou (2006) and Zhao and Yu (2006), a different, equivalent condition is condition which is termed in the latter work the “irrepresentable” condition. We point out that the neighborhood stability or irrepresentable condition can quite easily fail to hold which, due to the “almost” necessity of the condition, implies inconsistency of the Lasso for variable selection. For details about the irrepresentable condition, we refer to Zhao and Yu (2006).

Let us compare with our conditions. Regarding assumption (B1) we note the following. The inclusion of (A1) is weak since we do not require explicitly any behaviour of the covariance matrix $\Sigma_X = \Sigma_{X;n}$ in the sequence of distributions $P_n$ ($n \in \mathbb{N}$), except strict positive definiteness for all $n$ (but no explicit bound on the minimal eigenvalue). The partial faithfulness conditions follows from e.g. assuming (A2) in Section 4.2 for every $n$. It is also interesting to note that we require partial faithfulness only: dependence relations among covariates enter only indirectly via conditioning sets $S \subseteq \{1, \ldots, p\} \setminus j$ for a partial correlation between the response $Y$ and some covariate $X^{(j)}$. As a word of caution, the result in (Robins et al., 2003) indicates that uniform consistency for variable selection can fail to hold due to “nearly faithful” distributions. Assumption (B2) allows for an arbitrary polynomial growth of dimension as a function of sample size, i.e. high-dimensionality, while (B3) is a sparseness assumption in terms of the number of effective variables. Both (B2) and (B3) are fairly standard assumptions in high-dimensional asymptotics. Assumption (B4) is a regularity condition, saying that the non-zero partial correlations have to be of larger order
than $1/\sqrt{n}$. Without such a condition, one gets into the domain of super-efficiency, e.g. the behavior of the Hodges-Lehmann estimator. Assumptions (B3) and (B4) are rather minimal: note that with $b = 1$ in (B3), for example fixed $\text{peff}_n = \text{peff} < \infty$, the partial correlations can decay as $n^{-1/2+\varepsilon}$ for any $0 < \varepsilon \leq 1/2$. Finally, assumption (B5) is excluding perfect collinearity: since we require all partial correlations to be bounded by a constant $M < 1$ for all $n \in \mathbb{N}$, this yields some relatively mild restrictions on the covariance matrix $\Sigma_{Y,X} = \Sigma_{Y,X:n}$.

Although our assumptions are not directly comparable to the neighborhood stability or irrepresentable condition for the Lasso in general, our conditions seem much weaker if one is willing to assume partial faithfulness, e.g. assuming (A2) in Section 4.2. This is supported by our simple Example 3 below. If the dimension $p$ is fixed (with fixed distribution $P$ in the Gaussian linear model), (B2), (B3) and (B4) hold, and (B1) and (B5) remain as the only conditions.

Example 3. Consider the Gaussian linear model from (4.1) with

$$p = 4, \quad \text{peff} = |\mathcal{A}| = 3,$$

$$\beta_1, \beta_2, \beta_3 \text{ fixed i.i.d. realizations from } \mathcal{N}(0,1),$$

$$\beta_4 = 0, \quad \sigma^2 = 1, \quad \mu_X = 0,$$

$$\Sigma_X = \begin{pmatrix}
1 & \rho_1 & \rho_1 & \rho_2 \\
\rho_1 & 1 & \rho_1 & \rho_2 \\
\rho_1 & \rho_1 & 1 & \rho_2 \\
\rho_2 & \rho_2 & \rho_2 & 1
\end{pmatrix}, \quad \rho_1 = -0.4, \quad \rho_2 = 0.2.$$ 

It is shown in Zou (2006, Cor. 1) that the Lasso is inconsistent for this model. On the other hand, (B1) holds, because of (A2), and also (B5) is true (which are all the conditions for the PC-algorithm for a fixed distribution $P$). Hence, the PC-algorithm is consistent for variable selection. It should be noted though that also the adaptive Lasso (Zou, 2006) is consistent for this example.

Inconsistency of the Lasso typically occurs because of over-estimation, i.e. the Lasso selects too many variables. This has been made more pre-
cise with (asymptotic) results on the $\ell^1$- or $\ell^2$-norm of

$$\|\hat{\beta} - \beta\|_q (q = 1, 2),$$

see Bunea et al. (2007); Zhang and Huang (2007); van de Geer (2007); Meinshausen and Yu (2007). Also Candès and Tao (2007) prove, under restrictive conditions on the design, an $\ell^2$-norm result for the Dantzig selector which is another penalty-style estimation method. If the $\ell^q$-norm in (4.11) goes to zero, and under some conditions for the size of the non-zero coefficients, it holds that $\hat{A} = \{j; \hat{\beta}_j \neq 0\} \supseteq \mathcal{A}$ (Meinshausen and Yu, 2007, cf.). Furthermore, an additional stage of thresholding or using the more sophisticated adaptive Lasso (Zou, 2006) yield consistency of such two-stage procedures in the high-dimensional setting (Meinshausen and Yu, 2007; Huang et al., 2007). All of these works assume some conditions on either the minimal eigenvalues of the empirical covariance of the design, the coherence of the fixed design or the population correlations of the random design: these quantities measure the “ill-posedness” of the high-dimensional design matrix. Some of these conditions are substantially weaker than the neighborhood stability or the irrepresentable condition mentioned above. Our conditions for the PC-algorithm seem to be even substantially weaker, if one is willing to assume (A2), since we require only (A1) and (B5) regarding the regularity of the (random) design matrix.

### 4.5.2 Asymptotic Behavior for Non-Partial-Faithful Distributions

We have discussed in Proposition 3 that the PC$_{pop}$-algorithm identifies a set of associations which is between $\mathcal{A}_{\text{strong}}$ and $\mathcal{A}_{\text{strong-endo}}$, as described in (4.10), representing a more “causal” notion of association than the usual active set $\mathcal{A}$. The asymptotic arguments in the non-partial-faithful case are very similar to the analysis before. We assume:

(C1) The distribution $P_n$ satisfies assumption (A1) for all $n$. 

(C2) as assumption (B2).

(C3) The cardinality of set $A_{\text{strong-endo};n}$ satisfies: $|A_{\text{strong-endo};n}| = O(n^{1-b})$ for some $0 < b \leq 1$.

(C4) as assumption (B4).

(C5) as assumption (B5).

**Theorem 3.** Consider the Gaussian linear model (4.1) and assume (C1)-(C5). Then, there exists $\alpha_n \to 0$ ($n \to \infty$), see below, such that the PC-algorithm satisfies:

$$
P[ A_{\text{strong};n} \subseteq \hat{A}_n(\alpha) \subseteq A_{\text{strong-endo};n} ] = 1 - O(\exp(-Cn^{1-2d})) \to 1 \ (n \to \infty) \text{ for some } 0 < C < \infty,$$

where $d > 0$ is as in (C4).

Theorem 3 follows from Proposition 3 and analogous to the proof of Theorem 2. A possible choice of the tuning parameter is $\alpha = \alpha_n = 2(1 - \Phi(n^{1/2}c_n/2))$.

### 4.5.3 Asymptotic Behavior of Correlation Screening

For correlation screening, see formula (4.3), we do not require any sparsity. Related to our approach of correlation screening is the “Sure independence screening” by Fan and Lv (2007), but our reasoning, assumptions and mathematical derivations via partial faithfulness are very different. We assume:

(D1) as assumption (C1).

(D2) as assumption (B2).

(D3) as assumption (B4) but for marginal correlations $\text{Cor}(Y, X^{(j)}) = \rho_n(Y, j)$ only.
(D4) as assumption (B5) but for marginal correlations $\text{Cor}(Y, X^{(j)}) = \rho_n(Y, j)$ only.

Denote by $\widehat{A}_n^{[1]}(\alpha)$ the correlation screening active set estimated from data using tuning parameter $\alpha$ (i.e. the second step in the sample version of the PC-algorithm) and by $A_{\text{strong-endo};n}$, $A_{\text{strong};n}$ the set of variables from the stronger notions of associations as described in Section 4.4.

**Theorem 4.** Consider the Gaussian linear model (4.1) and assume (D1)-(D4). Then, there exists $\alpha_n \to 0$ $(n \to \infty)$, see below, such that:

\[
P[\widehat{A}_n^{[1]}(\alpha) \supseteq A_{\text{strong-endo};n}] = 1 - O(\exp(-Cn^{1-2d})) \to 1 (n \to \infty) \text{ for some } 0 < C < \infty,
\]

where $d > 0$ is as in (D3).

A proof is given in Section 4.8. We point out that $A_{\text{strong-endo};n} \supseteq A_{\text{strong};n}$, see formula (4.9). Moreover, for partial faithful distributions, i.e. assuming (B1) instead of (D1), Theorem 4 says that $P[\widehat{A}_n^{[1]}(\alpha) \supseteq A_n] \to 1 (n \to \infty)$. A possible choice of $\alpha$ is $\alpha_n = 2(1 - \Phi(n^{1/2}c_n/2))$. As pointed out above, we do not make any assumptions on sparsity. However, for non-sparse problems, many correlations may be non-zero and hence, $\widehat{A}^{[1]}$ could still be large, e.g. almost as large as the full set $\{1 \leq j \leq p\}$, and no effective dimensionality reduction would happen.

Under some condition on the covariance $\Sigma_X$ of the random design, it is shown in Fan and Lv (2007) that correlation screening, which they call sure independence screening, is overestimating the active set $A$. In general, this is not true. However, Theorem 4 describes that without essentially any assumption on $\Sigma_X$, correlation screening is overestimating the set of strong endogenous associations $A_{\text{strong-endo}}$ (and it is also overestimating the strong associations $A_{\text{strong}}$). This result may justify correlation screening as a more powerful tool than what it appears to be in the restrictive setting of Fan and Lv (2007).
4.6 Numerical Results

We analyze the variable selection properties using simulated and some high-dimensional real data.

4.6.1 Models Satisfying the Partial Faithfulness Condition

We consider first the classical association target only, namely the active set $\mathcal{A} = \{j; \beta_j \neq 0\}$. This enables a fair comparison of our PC-method with various versions of $\ell^1$-penalized approaches. In addition to reporting on goodness of fit measures for estimating associations, we give an overview of the runtime of the different methods.

ROC Analysis

We evaluate here the performance of the methods using ROC curves which measure the capacities for variable selection, independently from the issue to select good tuning parameters. We compare our simplified version of the PC-algorithm (PC, our own R-package `pcalg`) with the Lasso using the LARS algorithm (Efron et al., 2004) (LARS, R-package `lars`) and with the Elastic Net (Zou and Hastie, 2005) (ENET, R-package `elasticnet`). For the latter, we vary the $\ell^1$-penalty parameter only while keeping the $\ell^2$-penalty parameter fixed at the default value from the R-package `enet` to construct the ROC curve. In the ROC plots to be followed, horizontal and vertical bars indicate 95%-confidence intervals for the false positive rate (FPR) and the true positive rate (TPR), respectively; definitions of FPR and TPR are given in Section 4.6.2. In our PC-algorithm, the proposed default value for the tuning parameter is $\alpha = 0.05$: its performance is indicated by the intersection of a vertical line and the ROC curve.

We simulate data according to the Gaussian linear model (4.1) hav-
ing $p$ covariates with $\mu_X = 0$ and covariance matrix $\text{Cov}(X^{(i)}, X^{(j)}) = \Sigma_{X; i,j} = \rho^{|i-j|}$. The errors are generated as in model (4.1). In order to generate values for $\beta$, we follow (A2): a certain number $p_{\text{eff}}$ of coefficients $\beta_j$ have a value different from zero. The values of the nonzero $\beta_j$s are sampled independently from a standard normal distribution and the indices of the nonzero $\beta_j$s are evenly spaced between 1 and $p$. We consider a low- and a high-dimensional setting as follows:

low-dimensional: $p = 19$, $p_{\text{eff}} = 3$, $n = 100$; $\rho \in \{0, 0.3, 0.6\}$;

high-dimensional: $p = 499$, $p_{\text{eff}} = 10$, $n = 100$; $\rho \in \{0, 0.3, 0.6\}$.

Results for the low-dimensional case, based on 1000 independent simulations, are reported in Figures 4.2 to 4.4 which show a clear pattern. For small false positive rates (FPR), our PC method is clearly dominating LARS and ENET. If the correlation among the covariates increases, the performance of ENET gets worse, whereas the performances of PC and LARS don’t vary much. When focusing on values

**Figure 4.2:** Low dimensional: $p = 19$, $\rho = 0$. Vertical line indicates performance of PC using the default $\alpha = 0.05$. 
of FPR arising from the default value for $\alpha$ in our method, PC outperforms LARS and ENET by a large margin. Note that many application
areas call for a small FPR, as discussed also in Section 4.6.4.

**Figure 4.5:** High dimensional: $p = 499$, $\rho = 0$. Vertical line indicates performance of PC using the default $\alpha = 0.05$.

**Figure 4.6:** High dimensional: $p = 499$, $\rho = 0.3$. Vertical line indicates performance of PC using the default $\alpha = 0.05$.

For the high-dimensional case, the resulting ROC curves, based on
300 independent simulations, are given in Figures 4.5 to 4.7. For small false positive rates (FPR), the difference between the methods is not very big. LARS seems to perform best, PC is close to LARS, while ENET is worst. For larger FPR, this effect gets stronger. Up to the FPR which arises by the default value of $\alpha = 0.05$, PC is never significantly outperformed by either LARS or ENET.

**Runtime**

All calculations were done on a Dual Core Processor with 2.6 GHz and 32 GB RAM running on Linux and using R 2.5.1. The processor times were averaged in the low and high-dimensional example over 1000 and 300 replications, respectively. The average processor times and standard errors are given in Table 4.6.1.

We should avoid the conclusion that PC is faster than LARS or ENET since the runtimes for PC were measured using the default of $\alpha = 0.05$ only whereas LARS and ENET compute a whole path of
solutions. The purpose of Table 4.6.1 is to show that PC is certainly feasible for high-dimensional problems. In addition, when using PC on say 10 different (small) values of $\alpha$, the computation is about of the same order of magnitude than LARS or ENET for the whole solution path.

<table>
<thead>
<tr>
<th>$p$</th>
<th>$\rho$</th>
<th>$ave(t_{PC})$ [s]</th>
<th>$ave(t_{LARS})$ [s]</th>
<th>$ave(t_{ENET})$ [s]</th>
</tr>
</thead>
<tbody>
<tr>
<td>19</td>
<td>0</td>
<td>0.004 (4e-5)</td>
<td>0.016 (3e-5)</td>
<td>0.024 (3e-5)</td>
</tr>
<tr>
<td>19</td>
<td>0.3</td>
<td>0.004 (4e-5)</td>
<td>0.016 (3e-5)</td>
<td>0.024 (3e-5)</td>
</tr>
<tr>
<td>19</td>
<td>0.6</td>
<td>0.005 (5e-5)</td>
<td>0.016 (3e-5)</td>
<td>0.024 (3e-5)</td>
</tr>
<tr>
<td>499</td>
<td>0</td>
<td>0.164 (0.003)</td>
<td>0.795 (0.006)</td>
<td>13.23 (0.03)</td>
</tr>
<tr>
<td>499</td>
<td>0.3</td>
<td>0.163 (0.002)</td>
<td>0.838 (0.007)</td>
<td>13.41 (0.03)</td>
</tr>
<tr>
<td>499</td>
<td>0.6</td>
<td>0.160 (0.002)</td>
<td>0.902 (0.006)</td>
<td>12.91 (0.02)</td>
</tr>
</tbody>
</table>

Table 4.6.1: Average runtime in seconds over 1000 and 300 repetitions for $p = 19$ and $p = 499$, respectively. The runtimes for PC were measured using the default of $\alpha = 0.05$ while LARS and ENET compute a whole path of solutions.

### 4.6.2 Prediction Optimal Tuned Methods

We compare here different methods when using prediction optimal tuning. It is known that the prediction-optimal tuned Lasso overestimates the true model (Meinshausen and Bühlmann, 2006). The adaptive Lasso Zou (2006) and the relaxed Lasso Meinshausen (2007) correct Lasso’s overestimation behavior. Furthermore, we use our simplified version of the PC-algorithm for variable selection and use then the Lasso or the adaptive Lasso to estimate coefficients for the sub-model selected by the PC-method. For simplicity, we do not show results for the elastic net (which was found to be worse in terms of ROC-curves than adaptive or relaxed Lasso).

The methods are used as follows. Prediction optimal tuning is pursued with a validation set having the same size as the training data.
The Lasso is computed using the \texttt{lars}-package from \texttt{R}. For the adaptive Lasso, we first compute a prediction-optimal Lasso as initial estimator $\hat{\beta}_{\text{init}}$, and the adaptive Lasso is then computed with penalty $\lambda \sum_{j=1}^{p} |\beta_j|/|\hat{\beta}_{\text{init},j}|$ where $\lambda$ is chosen again in a prediction-optimal way. The computations are done with the \texttt{lars}-package from \texttt{R}, using re-scaled covariates for the adaptive step. The relaxed Lasso is computed with the \texttt{relaxo}-package from \texttt{R}. Our simplified version of the PC-algorithm with the Lasso for estimating coefficients is straightforward to do using the \texttt{pcalg}- and \texttt{lars}-packages from \texttt{R}: optimal tuning is with respect to the $\alpha$-parameter for the PC-algorithm and the penalty parameter for Lasso. For the simplified version of the PC-algorithm with the adaptive Lasso, we first compute the weights $w_j$ as follows: $w_j = 0$ if the variables has not been selected; and if the variable has been selected, $w_j = \text{minimum value of the test statistic } \sqrt{n - 3 - |S|Z(Y, j|S)}$ (see Section 4.3.2) over all iterations of the PC-algorithm. Then, we compute the adaptive Lasso with penalty $\lambda \sum_{j=1}^{p} w_j^{-1} |\beta_j|$, i.e. the weights for the adaptive step are from the PC-algorithm.

We are considering the following performance measures:

$$\|\hat{\beta} - \beta\|^2_2 = \sum_{j=1}^{p} (\hat{\beta}_j - \beta_j)^2 \quad \text{(MSE Coeff)},$$

$$E_X[(X^T(\hat{\beta} - \beta))^2] = (\hat{\beta} - \beta)^T \Sigma (\hat{\beta} - \beta), \quad \Sigma = \text{Cov}(X) \quad \text{(MSE Pred)},$$

$$\sum_{j=1}^{p} I(\hat{\beta}_j \neq 0) I(\beta_j \neq 0) / \sum_{j=1}^{p} I(\beta_j \neq 0) \quad \text{(true positive rate (TPR))},$$

$$\sum_{j=1}^{p} I(\hat{\beta}_j \neq 0) I(\beta_j = 0) / \sum_{j=1}^{p} I(\beta_j = 0) \quad \text{(false positive rate (FPR))}.$$ 

We simulate from a Gaussian linear model as in (4.1) with $p = 1000$, $\text{peff} = 20$, $n = 100$ and:

$$\beta_1, \ldots, \beta_{20} \text{ i.d. } \sim \mathcal{N}(0, 1), \quad \beta_{21} = \ldots = \beta_{1000} = 0,$$

$$\mu_X = 0, \quad \Sigma_{X;i,j} = 0.5^{|i-j|}, \quad \sigma^2 = 1,$$

with 100 replicates.
Figure 4.8: Prediction optimal tuned methods. Boxplots of performance measures as described in (4.12) and runtimes, based on 100 simulated model realizations. The PC-algorithm with Lasso coefficient estimation (PCl), the PC-algorithm with adaptive Lasso (PCal), Adaptive Lasso (al), Relaxed Lasso (r) and Lasso (l).

Figure 4.8 displays the results. As expected, the Lasso is yielding too many false positives while the adaptive Lasso and the relaxed Lasso have much better variable selection properties. The PC-based methods have clearly lowest false positive rates (FPR) while paying a price in terms of power, the true positive rate (TPR), and in terms of mean squared errors (MSE and prediction MSE).

In quite many applications, a low false positive rate is highly desirable even when paying a price in terms of power. For example, in molecular biology where a covariate represents a gene, only a limited number of selected genes (covariates) can be experimentally validated and hence, methods with a low false positive rate are preferred. This
strategy is briefly sketched in Section 4.6.4.

4.6.3 Model where the Partial Faithfulness Condition Fails to Hold

We consider a version of Example 2. Denote by

$$U_t = (0.95\varepsilon_{t-1} + \varepsilon_t)/\sqrt{1 + 0.95^2} \ (t = 2, 3, 4, 5),$$
$$\varepsilon_1, \varepsilon_2, \ldots, \varepsilon_5 \text{ i.i.d. } \sim \mathcal{N}(0, 1)$$

a Gaussian MA(1) process with marginal variance 1. Define

$$Y = U_5 + 0.15X^{(4)} + 0.15X^{(5)} + 0.15X^{(6)},$$
$$X^{(1)} = U_4, \ X^{(2)} = U_3, \ X^{(3)} = U_2,$$
$$X^{(4)}, X^{(5)}, X^{(6)} \text{ i.i.d. } \sim \mathcal{N}(0, 1) \text{ independent from } \{X^{(1)}, X^{(2)}, X^{(3)}\},$$
$$X^{(7)}, \ldots, X^{(20)} \sim \mathcal{N}_{14}(0, \Sigma) \text{ independent from } \{X^{(j)}; \ j = 1, 2, \ldots, 6\},$$
$$\Sigma_{ij} = 0.5|i-j| \quad (4.13)$$

The covariates $X^{(7)}, \ldots, X^{(20)}$ are ineffective and all partial correlations with the response $Y$ are zero. Furthermore, the model has the property that

$$\text{Cor}(Y, X^{(j)}) = 0 \text{ for } j = 2, 3,$$

while the partial correlations Parcor($Y, X^{(2)}|X^{(1)}$) $\neq 0$ and Parcor($Y, X^{(3)}|X^{(1)}, X^{(2)}$) $\neq 0$. Thus, the partial faithfulness condition fails to hold. Finally, the model exhibits relatively weak (partial) correlations of $Y$ with $X^{(4)}, X^{(5)}$ and $X^{(6)}$. The active set from standard regression, the strong endogenous and strong associations are

$$\mathcal{A} = \{1, 2, 3, 4, 5, 6\}, \ \mathcal{A}_{\text{strong-endo}} = \mathcal{A}_{\text{strong}} = \{1, 4, 5, 6\}.$$  

From Theorem 3 we know that the simplified PC-algorithm will identify the set $\mathcal{A}_{\text{strong-endo}} = \mathcal{A}_{\text{strong}} = \{1, 4, 5, 6\}$ whereas regression-type variable selection methods such as the Lasso or the elastic net yield the active set $\mathcal{A}$ as sample size $n$ tends to infinity.
Figure 4.9: Target $A_{\text{strong-endo}} = A_{\text{strong}}$ in model (4.13). Based on sample size $n = 1000$. Vertical line indicates performance of PC using the default $\alpha = 0.05$.

Figure 4.10: Target $A$ (active set in regression) in model (4.13). Based on sample size $n = 1000$. Vertical line indicates performance of PC using the default $\alpha = 0.05$. 
4.6. Numerical Results

We show in Figure 4.9 and 4.10, for the model in (4.13), the ROC curves of the simplified PC-algorithm, the Lasso and the elastic net for estimating $A_{strong-endo} = A_{strong}$ and for $A$, respectively. The results are based on sample size $n = 1000$ and 300 independent simulations from the model. As expected, we see very clearly that the PC-algorithm is better for estimating the set $A_{strong-endo} = A_{strong}$ while the Lasso or elastic net are superior for finding the active set $A$.

4.6.4 Real Data: Riboflavin Production from Bacillus Subtilis

We consider a high-dimensional real dataset about riboflavin production in Bacillus Subtilis, provided by DSM Nutritional Products. There is a continuous response variable $Y$ which measures the production rate of riboflavin, and there are $p = 4088$ covariates corresponding to the expression levels of genes. One of the major goals is to genetically modify Bacillus Subtilis in order to increase its production rate for riboflavin. An important step to achieve this goal is to find some genes which are most relevant for the production rate. We pursue this by variable (i.e. gene) selection in a linear model.

We use the methods PC, LARS and ENET as for simulated data. We run PC on the full data set, with various values of $\alpha$. Then, we compute LARS and ENET and choose the tuning parameters such that the same number of selected variables arise as for PC. We show the results from a genetically homogeneous group of $n = 72$ individuals.

Table 4.6.2 indicates that LARS and ENET are more similar variable selection methods than PC and any of those two. Thus, the PC-algorithm seems to extract information, i.e. selects genes, in a “rather different” way than the penalized methods LARS and ENET. We view this property as very desirable: for any large-scale problem, we want to see different aspects of the problem by using different methods; and ideally, results from different methods can be combined to obtain better results than what is achievable with a single procedure. We remark
that we still find a remarkable overlap of the few selected genes among $p = 4088$ candidates and in fact, it is highly significant when calibrating with a null-distribution which consists of pure random noise only. A very stringent rule for variable selection tailored towards low false positive findings is the intersection set of selected variables from all three methods: there is one interesting gene (anonymized) in this intersection set which is biologically “plausible” and which has not been genetically modified so far and hence, it is an interesting candidate for a biological intervention experiment.

### 4.7 Conclusions

The (simplified version of the) PC-algorithm is a very valuable method for inferring associations in a high-dimensional (but sparse) linear model where the number of covariates can greatly exceed the sample size. For partial faithful distributions, and under mild assumptions allowing for ill-posed (random) designs, we prove consistency for inferring the covariates with corresponding regression coefficients being non-zero (Theorem 2). Furthermore, we show that partial faithful distributions arise quite naturally (Theorem 1). In addition, even if the weak assumption about partial faithfulness fails to hold, we prove that the PC-algorithm is con-
sistent for some stronger notions of associations (Theorem 3) and we describe in Section 4.4 some connections to the concept of causality.

We also provide an efficient implementation of our (simplified) PC-algorithm in the R-package \texttt{pcalg}. The method is computationally feasible for high-dimensional problems with thousands of covariates, and we illustrate some results on simulated and real data in comparison to the Lasso and the Elastic Net.

4.8 Proofs

Proof of Theorem 1. Consider first the case for partial faithfulness. Then, Theorem 1 reads:

\[
\text{Cov}(Y, X^{(j)}|X^{(S)}) = 0 \text{ for some } S \subseteq \{1, \ldots, p\} \setminus \{j\} \Rightarrow \beta_j = 0. \tag{4.14}
\]

For proving (4.14), we use the contra-position and assume that $\beta_j \neq 0$.

Then:

\[
\begin{align*}
\text{Cov}(Y, X^{(j)}|X^{(S)}) &= \sum_{r \in A \cap S^c} \beta_r \Sigma_{X|S;r,j} \\
&= \beta_j \text{Var}(X^{(j)}|X^{(S)}) + \sum_{r \in A \cap S^c, r \neq j} \beta_r \Sigma_{X|S;r,j},
\end{align*}
\]

where $A = \{1 \leq r \leq p; \beta_r \neq 0\}$ and $\Sigma_{X|S} = \text{Cov}(X|X^{(S)})$ (which has degenerate entries for indices in $S$). In the Gaussian case, conditional covariances are constant, cf. Anderson (1984, Th. 2.5.1). Thus, the first quantity on the right-hand side equals some deterministic real-valued number $a_j \neq 0$. Therefore, the only way that the covariance $\text{Cov}(Y, X^{(j)}|X^{(S)})$ would equal zero would be:

\[
\sum_{r \in A \cap S^c, r \neq j} \beta_r \Sigma_{X|S;r,j} + a_j = 0. \tag{4.15}
\]
But this cannot happen, because (4.15) describes a hyperplane for \( \{ \beta_r; r \in A \cap S^C, r \neq j \} \) whose probability is zero since the \( \beta_r \)'s are from an absolutely continuous distribution with respect to Lebesgue measure; i.e. the set of \( \beta_r \)'s for which (4.15) holds has Lebesgue measure zero. This proves (4.14).

\[ \square \]

**Proof of Proposition 1.** The implication “\( \implies \)” obviously holds by considering the set \( S = \{1, \ldots, p \} \setminus j \).

For the other implication “\( \Leftarrow \)” we use contra-position. Assume that \( \text{Parcor}(Y, X^{(j)}|X^{(S)}) = 0 \) for some \( S \subseteq \{1, \ldots, p \} \setminus j \), and we want to show that \( \beta_j = 0 \). But this follows by definition of partial faithfulness.

\[ \square \]

**Proof of Proposition 2.** By definition and partial faithfulness, \( A \subseteq A^{[m_{\text{reach}}]} \). Thus, it remains to show that \( A^{[m_{\text{reach}}]} \subseteq A \).

Consider \( j \in A^{[m_{\text{reach}}]} \). The value of \( m_{\text{reach}} \) is such that

\[
\begin{align*}
\text{Parcor}(Y, X^{(j)}|X^{(S)}) &\neq 0 \text{ for all } S \subseteq A^{[m_{\text{reach}}-1]} \setminus j \supseteq A \setminus j, \\
|S| &\leq m_{\text{reach}} - 1. 
\end{align*}
\]

(4.16)

Regarding the last inequality: by definition of PC-algorithm, conditioning sets of size \( |S| = m_{\text{reach}} - 1 \) are considered in iteration \( m_{\text{reach}} \). In previous iterations of the algorithm, sets \( S \) of lower cardinality \( |S| \leq m_{\text{reach}} - 1 \) are considered, and in particular (because \( A^{[1]} \supseteq A^{[2]} \supseteq \ldots \)), all subsets \( S \subseteq A^{[m_{\text{reach}}-1]} \) with \( |S| \leq m_{\text{reach}} - 1 \) are considered.

Suppose that \( \beta_j = 0 \). It holds that \( |A \setminus j| \leq m_{\text{reach}} - 1 \) (because \( A \subseteq A^{[m_{\text{reach}}]} \) and \( |A^{[m_{\text{reach}}]}| \leq m_{\text{reach}} \)). In particular, using (4.16),

\[ \text{Parcor}(Y, X^{(j)}|X^{(A) \setminus j}) \neq 0. \]

(4.17)

Then, by definition of the linear model and the active set \( A \) and since \( \beta_j = 0 \),

\[ \text{Cov}(Y, X^{(j)}|X^{(A) \setminus j}) = 0 \]
4.8. Proofs

which is a contradiction to (4.17). Hence, it must hold that \( \beta_j \neq 0 \) and therefore \( \mathcal{A}^{[m_{\text{reach}}]} \subseteq \mathcal{A} \).

\[ \square \]

**Proof of Proposition 3.** Consider the set

\[ \tilde{\mathcal{A}}^{[m_{\text{reach}}]} = \{ j; \, \text{Parcor}(Y, X^{(j)}|X^{(S)}) \neq 0 \text{ for all } S \in \mathcal{A}^{[m_{\text{reach}}]} \setminus j \}. \]

Obviously, since \( \mathcal{A}^{[m_1]} \supseteq \mathcal{A}^{[m_2]} \supseteq \ldots \supseteq \mathcal{A}^{[m_{\text{reach}}]} \), there are fewer conditioning sets \( S \) occurring in \( \tilde{\mathcal{A}}^{[m_{\text{reach}}]} \) than in \( \mathcal{A}^{[m_{\text{reach}}]} \) and hence

\[ \mathcal{A}^{[m_{\text{reach}}]} \subseteq \tilde{\mathcal{A}}^{[m_{\text{reach}}]} \]  \hspace{1cm} (4.18)

Moreover, for every \( j \in \tilde{\mathcal{A}}^{[m_{\text{reach}}]} \):

\[ \text{Parcor}(Y, X^{(j)}|X^{(S)}) \neq 0 \text{ for all } S \in \tilde{\mathcal{A}}^{[m_{\text{reach}}]} \setminus j. \]

Thus, \( \tilde{\mathcal{A}}^{[m_{\text{reach}}]} \) is a set \( \mathcal{B} \) as in the definition of \( \mathcal{A}_{\text{strong-endo}} \), but it may not be maximal. Therefore, \( \tilde{\mathcal{A}}^{[m_{\text{reach}}]} \subseteq \mathcal{A}_{\text{strong-endo}} \) which, together with (4.18), completes the proof. \[ \square \]

**Proof of Theorem 2.** According to Proposition 2, the population PC\(_{\text{pop}}\)-algorithm identifies the active set \( \mathcal{A} = \mathcal{A}_n \). The probability that the PC-algorithm is yielding different variables than the population version can be bounded using three key steps which are analogous to the arguments in Kalisch and Bühlmann (2007, Theorem 1).

First, a uniform exponential inequality for \( Z \)-transformed estimated partial correlations up to order \( m = m_n = O(n^{1-b}) \), \( 0 < b \leq 1 \) as in (B3), can be established. See Kalisch and Bühlmann (2007, Lemma 3).

Second, we can consider a pseudo PC\((m)\)-algorithm which we run for \( m \) iterations. That is, if \( m_{\text{reach}} \leq m \), the pseudo PC\((m)\)-algorithm coincides with the PC-algorithm; and if \( m_{\text{reach}} > m \), the pseudo PC\((m)\)-algorithm stops earlier than the PC-algorithm. It can be shown, using the exponential inequality for \( Z \)-transformed estimated partial correlations (see above) and the union bound, that the pseudo PC\((m)\)-algorithm with \( m = m_n \geq m_{\text{reach}, n} \times \text{peff}_n = O(n^{1-b}) \) iterations

...
identifies the active set $A = A_n$. The reasoning is analogous to Kalisch and Bühlmann (2007, Lemma 4).

Finally, one can show that $\mathbb{P}[\hat{m}_{reach,n} = m_{reach,n}] \to 1$ $(n \to \infty)$. The arguments are as in Kalisch and Bühlmann (2007, Lemma 5).

Since the pseudo PC($\hat{m}_{reach,n}$)-algorithm coincides with the PC-algorithm, the second and third statement then complete the proof of Theorem 2.

\textbf{Proof of Theorem 4.} By definition, $A_{\text{strong-endo}} \subseteq A^{[1]}$, where the latter is the set of variables from correlation screening.

Denote by $Z_n(Y,j)$ the quantity as in (4.7) with $S = \emptyset$ and by $z_n(Y,j)$ its population analogue. i.e. the $Z$-transformed correlation. An error occurs when screening the $j$th variable if $Z_n(Y,j)$ has been tested to be zero but in fact $z_n(Y,j) \neq 0$. We denote such an error event by $E_{j}^{II}$ whose probability can be bounded as

$$\sup_j \mathbb{P}[E_{j}^{II}] \leq O(n) \exp(-C_1nc_n^2),$$

for some $0 < C_1 < \infty$, see Kalisch and Bühlmann (2007, formula (17)) (no sparsity assumption is used for this derivation). Thus, the probability of an error occurring in the correlation screening procedure is bounded by

$$\mathbb{P}[\bigcup_{1 \leq p_n} E_{j}^{II}] = O(p_n n) \exp(-C_1nc_n^2)$$
$$= O(\exp((1 + a) \log(n) - C_1n^{1-2d}))$$
$$= O(\exp(-C_2n^{1-2d}))$$

for some $0 < C_2 < \infty$. This completes the proof. \qed
Bibliography


ZUK, O., MARGEL, S. and DOMANY, E. (2006). On the number of samples needed to learn the correct structure of a bayesian network. In UAI.
Curriculum Vitae

I was born on December 28, 1977 in Hamburg. After moving to Nürnberg and attending primary school from 1984 to 1988, I spent eight years at the Sigmund-Schuckert Gymnasium in Nürnberg/Eibach and obtained Abitur in 1997.

From September 1997 to September 1998 I served within civil service as Rettungssanitäter and instructor for first aid courses at the Malteser Hilfsdienst in Nürnberg/Eibach.


In November 2004 I enrolled as a PhD student and worked as a teaching assistant at the SfS.