Seminar Modeling under Dependence

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

Department of Statistics
Ludwig-Maximilians-Universität München

Theresa Meier

Supervisor: Dr. Cornelius Fritz

March 1st, 2023

Abstract

The PC-algorithm is a constraint-based method to detect conditional independence relationships in observational data that can be represented as a directed acyclic graph (DAG). It assumes that the joint distribution of the variables which symbolize nodes is faithful to a DAG and multivariate Gaussian. In this report, the PC-algorithm and various extensions to improve the goodness of fit, run time and robustness to non-Gaussian distributions are presented and compared in a simulation study. In addition, the algorithm is applied to weather time series data. Overall, the PC-algorithm and its extensions prove to work very well especially for high-dimensional data.

Table of Contents

List of Figures

List of Tables

List of Algorithms

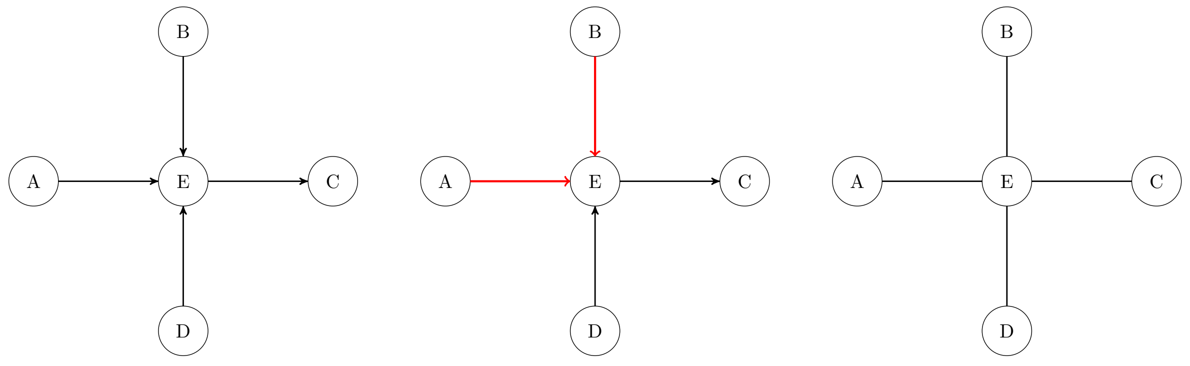
1 Introduction

Estimating conditional independence relationships in observational data is a task that many scientists attempted to solve in the last decades. As a result lots of algorithms exist but when it comes to high-dimensional data some of them reach their limits for various reasons such as run time or accuracy. Many of them have in common that the basis of the algorithm is a probabilistic graphical model, i.e. a probabilistic model for which a graph is used to express the conditional dependency structure between random variables. Finding the optimal graph underlying the data is a difficult and computationally costly challenge since the size of the corresponding space increases super-exponential with the number of nodes. To face this task, two main approaches exist: search and score algorithms and constraint-based methods. While search and score methods like the Greedy Equivalence Search Algorithm presented by Chickering (2003) search for all possible graphs and rank them by a scoring metric according to their goodness of fit, constraint-based methods are based on conditional independence tests between random variables, i.e. nodes of the graph. These tests aim to identify conditional independence relationships by removing edges in a structured way.

One of the state-of-the-art algorithms belonging to constraint-based methods is the so called PC-algorithm presented by Spirtes, Glymour, and Scheines (2000) which will be examined in this paper. The PC-algorithm starts with a fully connected graph and gradually deletes edges when conditional independence is detected. It serves as a starting point for further research and by now, multiple variations of the PC-algorithm exist, e.g. to make it computationally more efficient.

Exploring conditional independence relationships is crucial in various application fields. For instance, gene expression data is analyzed in order to find genetic causes of cancer in oncology research or in epidemiology. Another example of use is spatial data where countries or other spatial classifications are represented as random variables and relationships between different areas are analyzed (see Section 6). In general, the prerequisite is that a graph structure is present in the observed data.

In order to explore the PC-algorithm, this report starts with introducing basic definitions in Section 2. That will build the foundation for the procedure of the PC-algorithm which is explained in detail in Section 3. Since the algorithm in its original form is subject to some limitations, in Section 4 three extensions are presented, namely the stable, the parallel and the rank PC-algorithm. To evaluate all presented algorithms numerical experiments and their outcomes are summarized in Section 5. For an example of application, the algorithm is tested on weather time series data in Section 6 and the final discussion in Section 7 concludes this paper.



(a) DAG $\mathcal{G}$ (b) Collider in $\mathcal{G}$ (c) Skeleton of $\mathcal{G}$

Figure 1: Example of a DAG $\mathcal{G}$ (a) with a collider (b) and its corresponding skeleton (c).

2 Basic Definitions

Before introducing the PC-algorithm some fundamental definitions are required to fully understand the underlying ideas and procedures. First, directed acyclic graphs (DAGs) and their properties are examined and in a second step, relations within and between DAGs are explored.

2.1 Directed Acyclic Graphs

One important assumption for the underlying data is that it can be presented as a graph structure. One special graph is the DAG.

Definition 2.1 (Directed Acyclic Graph). *Let $\mathcal{G} = (\mathcal{N}, \mathcal{E})$ be a graph consisting of*

- *a set of nodes $\mathcal{N}$ and*
- *a set of edges $\mathcal{E} \subset \mathcal{N} \times \mathcal{N}$.*

*$\mathcal{G}$ is a **directed acyclic graph** (DAG) if $\mathcal{G}$ contains only directed edges and has no directed circle.*

An example of a DAG is displayed in Figure 1a. All following definitions in this chapter are defined for DAGs but equivalently hold true for graphs in general.

Definition 2.2 (Path). *A **path** in a DAG $\mathcal{G}$ from node A_0 to A_n is a sequence of distinct nodes $(A_0, A_1, ..., A_n)$ such that for all $1 \leq k \leq n$ either*

$$A_{k\text{-}1} \leftarrow A_k \in \mathcal{E} \textit{ or } A_{k\text{-}1} \rightarrow A_k \in \mathcal{E}.$$

Paths can be either directed or undirected. For instance, in Figure 1a the path $B \rightarrow E \rightarrow C$ is directed, because all edges are aligned, while the path $B \rightarrow E \leftarrow D$ is undirected.

Adjacent nodes are a special subset of nodes:

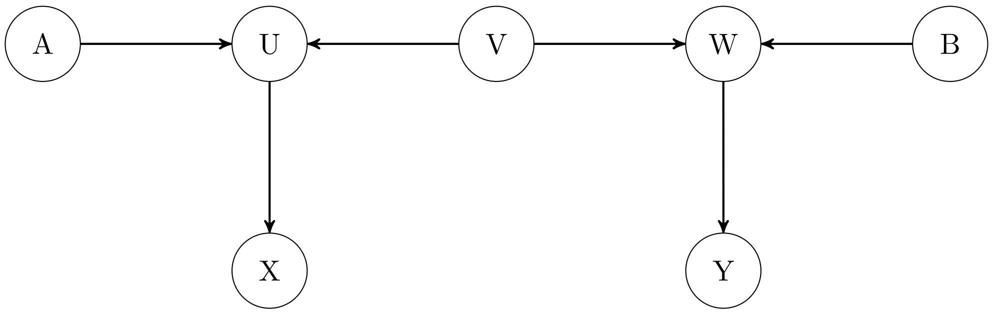


Figure 2: Example of d-separation and d-connection, respectively. Given $\mathcal{S} = \{U, W\}$, A and B are d-connected. Increasing the separating set to $\mathcal{S} = \{U, W, V\}$, A and B are d-separated given $\mathcal{S}$.

Definition 2.3 (Adjacency Set)**.** *The set of nodes that are* ***adjacent*** *to a node A in DAG $\mathcal{G}$ is defined as*

$$adj(A, \mathcal{G}) = \{B : (A, B) \in \mathcal{E} \text{ or } (B, A) \in \mathcal{E}\}.$$

In Figure 1a the adjacency set of node A consists only of node E. In addition, the concepts of colliders and skeletons are important for the PC-algorithm.

Definition 2.4 (Collider)**.** *A* ***collider*** *or* ***v-structure*** *is a triple of nodes $(A, B, C) \in \mathcal{N}^3$ that induces the subgraph*

$$A \rightarrow B \leftarrow C$$

where A and C are not adjacent i.e. there is no edge between A and C.

Figure 1b shows an example of a collider. In the following, only the term collider is used.

Definition 2.5 (Skeleton)**.** *The* ***skeleton*** *of a DAG is the undirected graph obtained by converting each directed edge into an undirected edge.*

In Figure 1c the skeleton of the example graph is shown.

2.2 Relations within and between DAGs

In the following consider a DAG $\mathcal{G} = (\mathcal{N}, \mathcal{E})$.

Definition 2.6 (d-Separation)**.** *Let $A, B \in \mathcal{N}$, $A \neq B$, and $\mathcal{S} \subset \mathcal{N}$ a subset of nodes not containing A and B. A and B are* ***d-connected*** *given a separating set $\mathcal{S}$ if and only if there exists an undirected path p between A and B such that*

- *every collider on p has a descendant in $\mathcal{S}$ or is in $\mathcal{S}$ itself, and*
- *no other node on p is in $\mathcal{S}$.*

Otherwise A and B are called ***d-separated*** *given $\mathcal{S}$.*

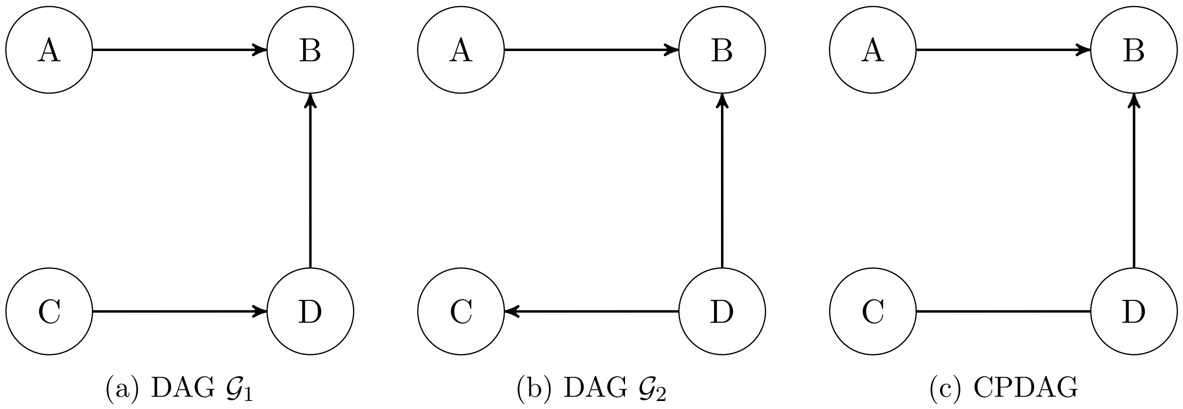


Figure 3: Two DAGs $\mathcal{G}_1$ (a) and $\mathcal{G}_2$ (b) with the corresponding CPDAG (c) under the assumption, that the respective Markov equivalence class consists of these two DAGs only. The CPDAG shares the same skeleton and the same collider ($A \to B \leftarrow D$) as the two DAGs. As there exists edge $C \to D$ in DAG $\mathcal{G}_1$ and edge $D \to C$ in DAG $\mathcal{G}_2$, the respective edge in the CPDAG is undirected.

Figure 2 shows an example to gain a better understanding of the concept of two d-separated nodes. Consider nodes A and B. The only path p between these two nodes is via

$$A \to U \leftarrow V \to W \leftarrow B,$$

where U and W are two colliders on p. For instance, setting $\mathcal{S} = \{U, W\}$, both conditions for d-connection are fulfilled as no other nodes except for all colliders on p are in $\mathcal{S}$. On the other hand, if $\mathcal{S} = \{U, W, V\}$, then the second condition is no longer fulfilled and A and B are d-separated given $\mathcal{S}$.

The space of DAGs can be partitioned into disjoint equvalence classes.

Definition 2.7 (Markov equivalence)**.** *Two DAGs with the same node set $\mathcal{N}$ are called* ***Markov equivalent*** *if and only if both share the same skeleton and the same colliders.*

Since the skeleton and colliders determine the data-generating distribution, all members of an equivalence class encode the same statistical model. Harris and Drton (2013) state that in particular they all share the same conditional independence information and the same d-separation relationships.

For a visualization of a Markov equivalence class a new kind of DAG is introduced:

Definition 2.8 (CPDAG)**.** *Let $[\mathcal{G}]$ be the Markov equivalence class of $\mathcal{G}$ and define the edge set*

$$[\mathcal{E}] = \bigcup_{\mathcal{H} \in [\mathcal{E}]} \mathcal{E}(\mathcal{H}),$$

where $\mathcal{E}(\mathcal{H})$ denotes the set of edges of a DAG $\mathcal{H} \in [\mathcal{G}]$.

The graph $\mathcal{C}(\mathcal{G}) = (\mathcal{N}, [\mathcal{E}])$ is called the ***completed partially directed acyclic graph*** *(CPDAG).*

The CPDAG may consist of both directed and undirected edges, therefore it is partially directed. For an example of a Markov equivalence class visualized by a CPDAG see Figure 3.

Now, let the nodes in $\mathcal{N}$ represent random variables $\mathbf{X} = (X_N)_{N\in\mathcal{N}}$.

Definition 2.9 (Faithfulness)**.** *The joint distribution $\mathcal{P}$ of $\boldsymbol{X}$ is **faithful** to DAG $\mathcal{G}$ if for any triple of pairwise disjoint subsets $\mathcal{A}, \mathcal{B}, \mathcal{S} \subset \mathcal{N}$, the following holds:*

$\mathcal{A}$ and $\mathcal{B}$ are d-separated given $\mathcal{S}$ if and only if $X_\mathcal{A}$ and $X_\mathcal{B}$ are conditionally independent given $X_\mathcal{S}$, i.e. $X_\mathcal{A} \perp X_\mathcal{B} | X_\mathcal{S}$.

Meek (1995b) showed that the non-faithful distributions of the multivariate Gaussian family form a Lebesgue null set in the space of distributions associated with a DAG, so faithfulness is not a strict assumption in practice. According to Harris and Drton (2013) an advantage of a faithful distribution is that statistical tests of conditional independence can be used to determine d-separation relationships in a DAG. In addition, assuming faithfulness allows to read conditional independence directly off a graph.

Extended theory about graphs and their properties can be found in Koller and N. Friedman (2009) and Spirtes, Glymour, and Scheines (2000).

3 PC-Algorithm

In this section, the PC-algorithm is presented in its original form together with its theoretical foundations, assumptions and details of the pseudo code.

3.1 Theoretical Foundation

The algorithm is based on two main theorems, both taken from Spirtes, Glymour, and Scheines (2000).

Theorem 3.1. *If nodes A and B are not adjacent in a DAG $\mathcal{G}$, then there is a set of nodes $\mathcal{S}$ which is either a subset of*

$$adj(A, \mathcal{G}) \quad \text{or of} \quad adj(B, \mathcal{G})$$

such that $\mathcal{S}$ d-separates A and B in $\mathcal{G}$.

The theorem implies that $\mathcal{S}$ disconnects A and B. The proof begins with the assumption that nodes A and B are not adjacent and A is not an ancestor of B. Let the total order Ord on the variables in $\mathcal{G}$ be such that all ancestors of A and all ancestors of B except for A are prior to A, and all other nodes are after A. Hence one can show that if B is not in the d-separation set of (A, B) then the d-separation set of (A, B) d-separates A from B in $\mathcal{G}$. B is in the d-separation set of (A, B) if and only if there is a path from A to B in which each node except the endpoints is a collider on the path, and each node on the path is an ancestor of A or B. But then there is an inducing path between A and B and one can show that A and B are adjacent, contrary to the assumption.

Theorem 3.1 could lead to the naive approach to exhaustively search for all possible neighbors of non-adjacent nodes in the DAG and perform conditional independence tests because a subset of the adjacency set will separate them. This is the concept of the SGS-algorithm which was also presented by Spirtes, Glymour, and Scheines (2000). However, this algorithm is rather inefficient because lots of conditional independence tests are needed. The PC-algorithm improves the SGS-algorithm by structuring the conditional independence tests with the goal to exploit the sparsity of the DAG (see Colombo and Maathuis (2012)) and to reduce the number of required conditional independence tests.

The second theorem is an important result about the output of the PC-algorithm:

Theorem 3.2. *Let the nodes in $\mathcal{N}$ represent random variables $\boldsymbol{X} = (X_N)_{N\in\mathcal{N}}$ which are multivariate Gaussian distributed and let the joint distribution $\mathcal{P}$ of $\boldsymbol{X}$ be faithful to DAG $\mathcal{G}$. Assume that the perfect conditional independence information about all pairs of variables (A, B) in $\mathcal{N}$ given subsets $\mathcal{S}$ is given.*
Then, the output of the PC-algorithm is the CPDAG that represents $\mathcal{G}$.

The following sketch of the proof of Theorem 3.2 can be found in Colombo and Maathuis (2012). Under faithfulness conditional independence in the joint distribution of the nodes is equivalent to d-separation relations. One can show that two variables in a DAG are d-separated by a subset of the remaining variables if and only if they are d-separated by their parent nodes. This can be used to determine the skeleton. The PC-algorithm is guaranteed to check the conditional independencies: at any point in the algorithm the current graph is a supergraph of the true CPDAG and the algorithm checks conditional independencies given all subsets of adjacency sets which obviously include the parent sets.

It follows that even given infinite amount of data, the PC-algorithm is not able to identify the true DAG but only its Markov equivalence class. In practice this circumstance is not restrictive, as all members of one Markov equivalence class encode the same statistical model and therefore the same data-generating distribution.

Theorem 3.2 not only delivers a result about the output but also of the assumptions of the PC-algorithm. These comprise on the one hand the faithfulness of the underlying joint distribution of the nodes in order to be able to read off conditional independence relations from the graph. On the other hand the joint distribution is assumed to be multivariate Gaussian. This has one important advantage when it comes to partial correlations:

Theorem 3.3. *For a multivariate normal distributed random vector $\boldsymbol{X} \in \mathbb{R}^p$ denote its partial correlation between components $\boldsymbol{X}^{(i)}$ and $\boldsymbol{X}^{(j)}$ $(i \neq j)$ given set*

$$\{\boldsymbol{X}^{(r)},\, r \in R \subset \{1, \dots, p\}\backslash\{i, j\}\}$$

by $\rho_{i,j|R}$. Then it holds:

$$\rho_{i,j|R} = 0 \textit{ if and only if } \boldsymbol{X}^{(i)} \textit{ and } \boldsymbol{X}^{(j)} \textit{ are conditionally independent given } \{\boldsymbol{X}^{(r)}, r \in R\}.$$

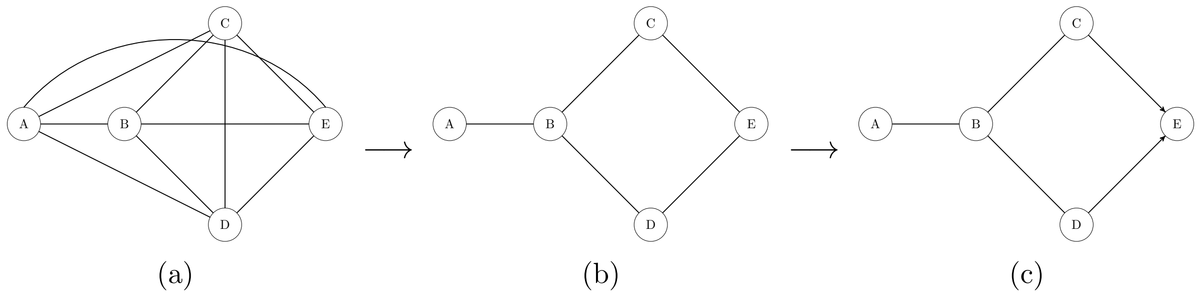


Figure 4: Example of the two-step computation of the PC-algorithm (see Spirtes, Glymour, and Scheines (2000)). Graph (a) shows the starting point of the first step, the complete undirected graph. The output of step 1 is the approximated skeleton (b) of the true underlying DAG. The edges are partially oriented in step 2 to finally obtain the approximated CPDAG (c).

This theorem is a popular property of the multivariate Gaussian distributions. Assuming a multivariate Gaussian distribution for the nodes of the DAG of which one wants to find the conditional independence relationships makes it possible to derive conditional independence from partial correlations. In practice, given enough data, the central limit theorem implies that this assumption is not very restrictive. However, in Section 4.3 one method to deal with non-Gaussian data is presented.

3.2 Algorithm in Detail

Recall the goal of the PC-algorithm to find iterative conditional independence relations in DAGs, i.e. the (directed) edges between nodes in a DAG. Therefore the PC-algorithm performs a two-step computation: In step 1, the algorithm starts with the complete undirected graph, i.e. the graph where every node is connected to each other node via an undirected edge. The aim of the first step is to find the underlying skeleton by performing conditional independence tests. In the second step these detected undirected edges are oriented in order to output the approximated CPDAG of the true underlying DAG. An example is shown in Figure 4. All inference is done in the first step which is by far the more computationally costly one while in the second step only deterministic rules are applied. Therefore research on how to improve the original PC-algorithm mainly focuses on the first part.

Step 1: Algorithm 1 shows the pseudo code of the first step, the derivation of the skeleton. It takes a data set where the variables symbolize nodes and a significance level for the conditional independence test as input and starts at level $d = 0$. The level indicates the size of the separating set of two adjacent nodes and is increased after every pair of adjacent nodes is tested for conditional independence. The starting level $d = 0$ implies the empty set. Then, for each pair of adjacent nodes where the adjacency set comprises at least d elements subsets of the adjacency set of size d are tested for conditional independence, i.e. if the two chosen adjacent nodes are conditionally independent given this separating set $\mathcal{S}$, $\mathcal{S} = \{\}$ in the first step.

Algorithm 1: Step 1: Learning the skeleton

```
Data: Data set $D$ with nodes $\mathcal{N}$, significance level $\alpha$
Result: Skeleton $\mathcal{G}_{Skel}$, seperating sets $S$
$d \leftarrow 0$
repeat
    for each ordered pair of adjacent nodes $X$ and $Y$ do
        if $|adj(X, \mathcal{G}) \setminus \{Y\}| \geq d$ then
            for each subset $S \subseteq adj(X, \mathcal{G}) \setminus \{Y\}$ and $|S| = d$ do
                Test $CI(X, Y|S)$ on significance level $\alpha$
                if $CI(X, Y|S)$ then
                    Remove edge between $X$ and $Y$
                    Save $S$ as separating set of $(X, Y)$
                    Update $\mathcal{G}$ and $\mathcal{E}$
                    break
                end
            end
        end
    end
    $d \leftarrow d + 1$
until $|adj(X, \mathcal{G}) \setminus \{Y\}| < d$ for every pair of adjacent nodes;
```

In practice, the perfect conditional independence information is not known, so the algorithm needs to estimate it from the data. Therefore, recall the property of the multivariate Gaussian distribution in Theorem 3.3 regarding partial correlations. The central idea of the conditional independence test is the estimation of $\rho_{A,B|\mathcal{S}}$ of two adjacent nodes A and B given separating set $\mathcal{S}$ and testing whether it is close to zero. Several possibilities for the estimation exist. For instance $\rho_{A,B|\mathcal{S}}$ can be estimated recursively, via regression or via inverting parts of the correlation matrix. The PC-algorithm is based on the following recursion for any $C \in \mathcal{S}$:

$$\rho_{A,B|\mathcal{S}} = \frac{\rho_{A,B|\mathcal{S}\setminus C} - \rho_{A,C|\mathcal{S}\setminus C} \cdot \rho_{B,C|\mathcal{S}\setminus C}}{\sqrt{(1-\rho^2_{A,C|\mathcal{S}\setminus C})(1-\rho^2_{B,C|\mathcal{S}\setminus C})}} \tag{1}$$

Note that this represents the Pearson correlation. For further details see Kalisch and Bühlmann (2005). To finally obtain a valid test statistic one needs to apply Fisher's z-transform

$$Z(A, B|\mathcal{S}) = \frac{1}{2}\log\left(\frac{1+\rho_{A,B|\mathcal{S}\setminus C}}{1-\rho_{A,B|\mathcal{S}\setminus C}}\right), \tag{2}$$

which is standard Gaussian distributed under the null hypothesis $H_0 : \rho_{A,B|\mathcal{S}} = 0$. The test rejects H_0 against $H_1 : \rho_{A,B|\mathcal{S}} \neq 0$ if

$$\sqrt{n - |\mathcal{S}| - 3}\;|Z(A, B|\mathcal{S})| > \phi^{-1}\left(1 - \frac{\alpha}{2}\right), \tag{3}$$

Removed Edge	Separating Set	Conditional Independence
$A - C$	$\{B\}$	$A \perp C \mid \{B\}$
$A - D$	$\{B\}$	$A \perp D \mid \{B\}$
$A - E$	$\{B\}$	$A \perp E \mid \{B\}$
$C - D$	$\{B\}$	$C \perp D \mid \{B\}$
$B - E$	$\{C, D\}$	$B \perp E \mid \{C, D\}$

Table 1: Intermediate results of the first step of the PC-algorithm for the example in Figure 4b. Edges in the first column are deleted from the edge set of the graph and the corresponding separating sets are stored for the second step.

where ϕ denotes the cumulative distribution function of a standard Gaussian distribution and n is the number of samples in the data set.

Coming back to the algorithm, conditional independence tests are conducted for each pair of adjacent nodes in the complete undirected graph. If the test is not rejected and cannot find evidence for conditional dependence, the edge between the corresponding nodes is deleted and the respective separating set is stored before the edge set and the graph is updated. After all nodes have been tested for level $d = 0$, the level is increased so that the separation sets now consist of one element of the adjacency set. Then the same procedure is performed as in the first level and the algorithm continues to increase the level until there are no pairs of adjacent nodes left whose adjacency set is larger than or equal to the level d.

For the example in Figure 4, the intermediate results of the first step are displayed in Table 1. They are the starting point for the second step of the algorithm.

Step 2: In Algorithm 2 the pseudo code of the second step is displayed. It aims at orienting the detected undirected edges by applying deterministic rules. The algorithm takes the estimated skeleton and the corresponding separating sets of step 1 as input and outputs the respective CPDAG. In the first loop colliders of the CPDAG are detected. Recall from Definition 2.8 that colliders are uniquely determined and are equal for all representations of the respective Markov equivalence class. The second part, (R1) to (R4), are based on not producing any invalid edges:

- All colliders are found in the first part of step 2. (R1) to (R4) should therefore not detect any new colliders.
- The resulting CPDAG is supposed to be acyclic. (R1) to (R4) are constructed such that no cycles are produced.

Figure 4c shows the final approximated CPDAG of the example graph. In this case, the PC-algorithm is only able to identify the single collider and not to orient any other edge.

Note that this step is order-dependent and may result in conflicting edges. Built-in implementations in R therefore include a parameter that determines how to deal with conflicts.

Algorithm 2: Step 2: Learning the CPDAG

```
Data: Skeleton $\mathcal{G}_{Skel}$, separating sets $S$
Result: CPDAG
for all pairs of non-adjacent nodes $X$ and $Y$ with common neighbor $U$ do
    if $U \notin S(X,Y)$ then
        Orient $X - U - Y$ as $X \to U \leftarrow Y$
    end
end
Orient as many undirected edges as possible by applying the following rules:
repeat
    (R1) Orient $Y - U$ into $Y \to U$ when there is edge $X \to Y$ s.t. $X$ and $U$ are non-adjacent.
    (R2) Orient $X - Y$ into $X \to Y$ when there is $X \to U \to Y$.
    (R3) Orient $X - Y$ into $X \to Y$ when there is $X \to U \to Y$ and $X \to V \to Y$ s.t. $U$ and $V$ are non-adjacent.
    (R4) Orient $X - Y$ into $X \to Y$ when there is $X \to U \to V$ and $U \to V \to Y$ s.t. $U$ and $Y$ are non-adjacent.
until no more edges can be oriented;
```

3.3 Consistency

Kalisch and Bühlmann (2005) showed in their work the consistency for both parts of the PC-algorithm even for high-dimensional data when the DAG is sparse:

Theorem 3.4. *Under the assumptions that*

- *the data are realisations from a multivariate Gaussian distribution,*
- *the joint distribution is faithful to a DAG $\mathcal{G}$,*
- *DAG $\mathcal{G}$ is suitably sparse and*
- *an additional regularity assumption,*

both step 1 and step 2 of the PC-algorithm are consistent for high-dimensional data, i.e. there exists $\alpha_n \to 0$ $(n \to \infty)$ and $\beta_n \to 0$ $(n \to \infty)$ such that

$$\mathbb{P}[\hat{\mathcal{G}}_{skel,n}(\alpha_n) = \mathcal{G}_{skel,n}] = 1 - \mathcal{O}(exp(-C_1 n^{1-2d_1})) \to 1 \ (n \to \infty)$$

and

$$\mathbb{P}[\hat{\mathcal{G}}_{CPDAG}(\alpha_n) = \mathcal{G}_{CPDAG}] = 1 - \mathcal{O}(exp(-C_2 n^{1-2d_2})) \to 1 \ (n \to \infty)$$

for some $0 < C_1, C_2 < \infty$ and $0 < d_1, d_2 < \frac{1}{2}$. $\hat{\mathcal{G}}_{skel,n}$ stands for the estimated skeleton of the first step and $\mathcal{G}_{skel,n}$ for the true skeleton, while $\hat{\mathcal{G}}_{CPDAG}$ corresponds to the estimated CPDAG of the second step and $\mathcal{G}_{CPDAG}$ to the true CPDAG of the true underlying DAG.

A proof of this theorem can be found in the Appendix of Kalisch and Bühlmann (2005). Following this, the authors proved that in theory, the most appropriate significance level α for the conditional independence tests is given by

$$\alpha = \alpha_n = 2(1 - \phi\left(\sqrt{n}\frac{c_n}{2}\right) \tag{4}$$

which depends on the unknown lower bound c_n of partial correlations

$$\inf\{|\rho_{\mathcal{A},\mathcal{B}|\mathcal{S}}| : \mathcal{A}, \mathcal{B}, \mathcal{S} \text{ with } \rho_{\mathcal{A},\mathcal{B}|\mathcal{S}} \neq 0\} \geq c_n.$$

Meek (1995a) found out that if the skeleton is determined correctly, the orientation step of the edges will never fail.

This result is crucial for the PC-algorithm and sets it apart from other algorithms that struggle with high-dimensional data.

3.4 Limitations of the PC-Algorithm

Nevertheless, the PC-algorithm in its original form faces three main problems which can lead to a poor fit or high computation time:

- In Algorithm 1 the first for loop chooses ordered pairs of adjacent nodes and performs the corresponding conditional independence tests in the given order. Changing the ordering of the variables could lead to a different result, since the conditional independence tests are conducted in a different order as well. A solution to this limitation is presented by Colombo and Maathuis (2012), the so called stable PC-algorithm (see Section 4.1).

- Especially for high-dimensional DAGs consisting of loads of nodes, the number of required conditional independence tests for determining the skeleton is very large. Kalisch and Bühlmann (2005) state that in the worst case, the run time increases exponential with the number of nodes. The original PC-algorithm conducts all tests in sequence on the same core of the computer's CPU that leads to a huge amount of time until all tests are done. Le et al. (2019) proposed the parallel PC-algorithm which is able to parallelize the testing procedure utilizing multiple cores and therefore decreases the run time drastically (see Section 4.2).

- So far, a Gaussian distribution was assumed for the joint distribution of the nodes. Both Theorem 3.3 and Theorem 3.4 only hold based on this assumption. If the joint distribution deviates remarkably from a multivariate Gaussian, the PC-algorithm will result in a poor fit (see Section 5.2). An extension for nonparanormal distributions, namely the rank PC-algorithm, was found by Harris and Drton (2013). Details can be found in Section 4.3.

In addition to these variations, there are many other algorithms based on and further improving the original PC-algorithm, and the potential in this area of research is far from being exhausted.

4 Extensions

In this section, the three previously mentioned extensions of the PC-algorithm are considered in more detail.

4.1 Stable PC-Algorithm

Let us revise the example in Figure 4 to gain a better understanding of why the result of the PC-algorithm is determined by the order of the conditional independence tests. Recall that the edge $A - C$ was deleted because $A \perp C|B$ and $B \in adj(A, \mathcal{G})\backslash C$ as portrayed in Table 1. If for instance the edge $A - B$ was mistakenly removed before performing the test for edge $A - C$, the algorithm could not detect the deletion of edge $A - C$ as B would no longer be in the adjacency set of A. On the other hand, changing the order of these tests would lead to the deletion of edge $A-C$. This means in particular that incorrectly removing or retaining edges results in changes of the adjacency sets and therefore in changes in the possible separating sets for the nodes that are considered afterwards.

To circumvent this limitation, Colombo and Maathuis (2012) modified in their stable PC-algorithm the part of updating the edge set and the graph in Algorithm 1. In detail, the idea consists of keeping the adjacency sets of the nodes unchanged within each level d and update DAG $\mathcal{G}$ and its edge set $\mathcal{E}$ at the end of level d. In addition, the adjacency sets of all nodes are stored for the search of the appropriate separating set $\mathcal{S}$. As a result, incorrectly removed edges do not lead to follow-up errors as the adjacency sets still contain the node to which the edge was deleted.

On the other hand, not decreasing the adjacency set in each step by removing edges leads to larger adjacency sets compared to the original PC-algorithm. This results in even more required conditional independence tests that in turn increases the run time even more. One possible solution is the parallel PC-algorithm presented in the next section.

Recall the consistency of the PC-algorithm in Section 3.3 for high-dimensional data under certain assumptions. This result holds equally for the stable PC-algorithm as the prove of Theorem 3.4 is based on conditional independence tests with separating sets of a size less than or equal to the degree of the DAG. The stable PC-algorithm does not violate this assumption and therefore the consistency result remains valid according to Colombo and Maathuis (2012).

4.2 Parallel PC-Algorithm

In order to accelerate the run time of the PC-algorithm, Le et al. (2019) developed an extension to parallelize the determination of the skeleton. To be precise, the so called parallel PC-algorithm aims to parallelize the most costly part by performing several conditional independence tests at the same time.

Parallelism in general combines the idea to break down a big task into several different smaller sub tasks and distribute them over different cores of the computer's CPU. One crucial prerequisite is the independence of the sub tasks. One naive approach could be to parallelize the different levels d of the Algorithm 1. However, this would violate the assumption of independent sub tasks as the results of the conditional independence tests of a particular level d influence the conditional independence test results of the following level $d+1$. The parallel PC-algorithm thus parallelizes the conditional independence tests within each level in a three-stage process:

- Step 1: Distribute the conditional independence tests evenly among the CPU cores by grouping conditional independence tests of the same edge together.
- Step 2: Each core performs its set of conditional independence tests in parallel with the other CPU cores.
- Step 3: Test results from all CPU cores are integrated into one global graph update.

Note that this procedure only works for the stable PC-algorithm as the edge set and the graph are updated at the end of each level d not in-between as in the original form of the PC-algorithm. For big data sets, the user can set an additional memory-efficient indicator to detect free memory.

As the stable and the parallel PC-algorithm both result in the same approximated CPDAG, the consistency of the stable PC-algorithm for high-dimensional data directly transfers to the parallel PC-algorithm.

4.3 Rank PC-Algorithm

So far, all results about high-dimensional consistency were based on the assumption that the joint distribution of the random variables associated with the nodes of a DAG is multivariate Gaussian. The goal of Harris and Drton (2013) was the extention of these results to a broader class of distributions, the so called nonparanormal distributions, which were introduced by Liu, Lafferty, and Wasserman (2009).

Definition 4.1. *Let $f = (f_N)_{N\in\mathcal{N}}$, $f_N : \mathbb{R} \to \mathbb{R}$, be strictly increasing functions and $\Sigma \in \mathbb{R}^{\mathcal{N}\times\mathcal{N}}$ a positive definite covariance matrix such that the random vector $\boldsymbol{Z} = (Z_N)_{N\in\mathcal{N}}$ is multivariate Gaussian distributed with zero mean and covariance matrix Σ. Then, the random vector $f(\boldsymbol{Z}) = (f_N(Z_N))_{N\in\mathcal{N}}$ is nonparanormal distributed, i. e.*

$$f(\boldsymbol{Z}) = (f_N(Z_N))_{N\in\mathcal{N}} \sim NPN(f, \Sigma).$$

Hence, nonparanormal random vectors are transformed Gaussian random vectors by some strictly increasing functions. To be precise, the nonparanormal distribution is a high-dimensional Gaussian copula with nonparametric marginals. If additionally the functions f_N are affine, all multivariate Gaussian distributions are nonparanormal as well.

In Figure 5 an example of a nonparanormal density is shown. For this purpose samples from a standard Gaussian distribution are drawn and the corresponding density is displayed in red.

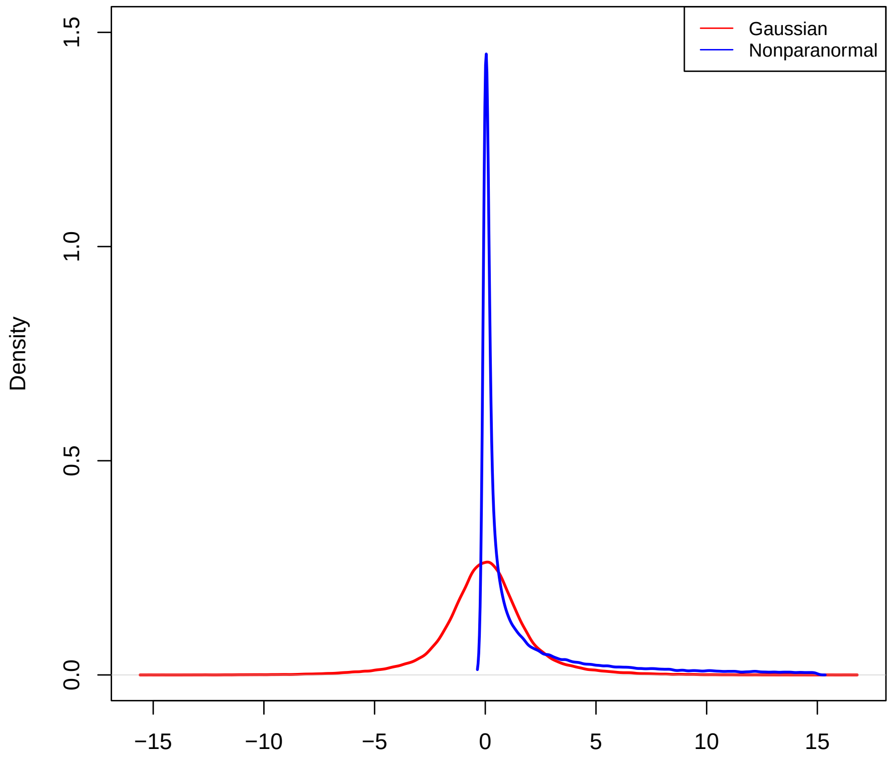


Figure 5: Densities of a standard Gaussian distribution (red) and the corresponding Gaussian copula (blue) obtained by transforming marginals of sampled Gaussian observations into an $F_{1,1}$-distribution.

In a second step the marginals of these samples are transformed into an $F_{1,1}$-distribution. The density of this Gaussian copula is visualized as blue curve in the figure.

For a simplification of notation, assume that $X \sim NPN(f, \Sigma)$ is the observed data and $Z \sim N(0, \Sigma)$ is the respective latent multivariate Gaussian vector. Then, for any pairwise disjoint subsets $\mathcal{A}, \mathcal{B}, \mathcal{S} \subset \mathcal{N}$ it holds:

$$X_{\mathcal{A}} \perp X_{\mathcal{B}} | X_{\mathcal{S}} \iff Z_{\mathcal{A}} \perp Z_{\mathcal{B}} | Z_{\mathcal{S}} \tag{5}$$

This implies that for two nodes A and B and separating set $\mathcal{S} \backslash \{A, B\}$, the result in Theorem 3.3 also holds for nonparanormal distributed variables:

$$X_A \perp X_B | X_{\mathcal{S}} \iff \rho_{A,B|\mathcal{S}} = 0. \tag{6}$$

Recall that in the Gaussian setting, the partial correlations were estimated by using standard Pearson-type empirical correlations where Equation 1 is one possible estimation approach.

Harris and Drton (2013) state that this estimation can be rewritten as

$$\rho_{A,B|\mathcal{S}} = -\frac{\Psi^{-1}_{A,B}}{\sqrt{\Psi^{-1}_{A,A}\Psi^{-1}_{B,B}}}, \tag{7}$$

where $\Psi = \Sigma_{(A,B,\mathcal{S}),(A,B,\mathcal{S})}$ is a part of the covariance matrix Σ. Thus, this approach is based on the estimation of the covariance matrix and partially inverting it to obtain partial correlations for conditional independence tests. The covariance matrix is obtained by transforming the estimated correlation matrix.

In the nonparanormal setting however, Pearson-type empirical correlation estimates are no longer appropriate since one needs to estimate the correlation matrix from nonparanormal distributed data. The approach of Harris and Drton replaces these estimates by rank-based estimates, i.e. Spearman's rank correlation ρ^S and Kendall's τ. Since the transformation functions f_N are assumed to be strictly increasing, i.e. monotonic, ranks of the latent multivariate Gaussian vectors are preserved. Note that the data is assumed to be continuous so the probability for ties is equal to 0 and therefore ranks are well-defined. Liu, Han, et al. (2012) showed that simple trigonometric transformations of these rank-based estimates can be used to calculate accurate estimates for Pearson-type Gaussian correlations ρ:

$$\mathbb{P}\left(\left|2\sin\left(\frac{\pi}{6}\hat{\rho}^S\right) - \rho\right| > \epsilon\right) \leq 2\exp\left(-\frac{2}{9\pi^2}n\epsilon^2\right) \tag{8}$$

and

$$\mathbb{P}\left(\left|\sin\left(\frac{\pi}{2}\hat{\tau}\right) - \rho\right| > \epsilon\right) \leq 2\exp\left(-\frac{2}{\pi^2}n\epsilon^2\right), \tag{9}$$

for bivariate Gaussian distributed random variables (X, Y) with $\mathrm{Corr}(X, Y) = \rho$ and $\epsilon > 0$.

Since ranks are preserved, both formulas hold for bivariate nonparanormal distributed random vectors $(X, Y) \sim NPN(f, \Sigma)$ with Pearson correlation ρ in the corresponding latent bivariate Gaussian distribution. This result directly transfers to random vectors of higher dimensions.

The trigonometric transformations in Equation 8 and Equation 9 respectively are the basis for implementations of the rank PC-algorithm in R (see Section 5.2).

In conclusion, the only change in the procedure of the rank PC-algorithm in comparison to the original PC-algorithm is the estimation part of the correlation matrix which can be summarized as:

- Step 1: Estimate marginal correlations $\hat{\rho}_{A,B}$ for every pair of nodes (A, B) in time $\mathcal{O}(|\mathcal{N}|^2 n\log(n))$ with rank-based correlation estimators (by $\mathcal{O}(\log(n))$ larger than the estimation step in the original PC-algorithm).
- Step 2: Determine partial correlations via Equation 7, and perform conditional independence tests as in the original PC-algorithm.

Harris and Drton proved in their work an error bound for $\mathbb{P}[\hat{\mathcal{G}}_{CPDAG} \neq \mathcal{G}_{CPDAG}]$ and deduced the consistency of the rank PC-algorithm for high-dimensional data.

Cui, Groot, and Heskes (2016) extended the idea of the rank PC-algorithm to continuous and discrete data under the assumption that the data is drawn from a Gaussian copula model. Details on the so called Copula PC-algorithm can be found in their paper.

5 Numerical Experiments

The goal of this section is to show properties of all four presented algorithms when it comes to numerical experiments. Therefore, the first section deals with Gaussian data only, while the second part also takes nonparanormal distributions into account. In this part only simulated data is taken to evaluate the fit regarding the ground truth, while in Section 6 the PC-algorithm will be applied to a real world data set to discover relationships between weather gauging stations at rivers in Bavaria.

In R, multiple packages for running the different versions of the PC-algorithm exist. The most popular one is the package *pcalg* created by Kalisch, Mächler, et al. (2012) which also includes the stable PC-algorithm. The inventors of the parallel PC-algorithm implemented their version in the package *parallelPC*. In all these packages the build-in functions for the different PC-algorithms are constructed in the same way. Among others they take the estimated correlation matrix of the data, the type of conditional independence test, the significance level α and the method used for the estimation of the skeleton (original, stable, parallel) as input. In addition, the user can set how to deal with conflicts when it comes to orienting the found edges in step 2.

All following experiments were conducted on Linux system with an AMD Ryzen 5 5625U and 16 GB of RAM using R 4.2.2. For running the parallel PC-algorithm four cores of the computer's CPU were used. All codes, tables and plots can be found on GitHub[1].

5.1 Simulation Studies for multivariate Gaussian Data

In this section, the focus lies on the original, the stable and the parallel PC-algorithm assuming a multivariate Gaussian distribution for the data. An in-depth analysis regarding the rank PC-algorithm can be found in Section 5.2.

5.1.1 Study Design

Recall that the stable PC-algorithm is an extension to improve the fit of the original PC-algorithm while the parallel PC-algorithm is a faster version of the stable PC-algorithm. To numerically verify these properties, all three algorithms are applied to simulated data. Therefore, a random DAG is simulated multiple times in dependence of the number of nodes, i.e. the dimension p, and a sparsity parameter s. The sparsity determines the size of the

[1] https://github.com/TheresaMeier/Seminar_PC

Structural Hamming Distance	SHD	Number of edge insertions, deletions or flips in order to transform the estimated CPDAG to the true CPDAG (or vice versa)
True Positive Rate	TPR	Number of correctly found edges in estimated CPDAG divided by number of true edges in true CPDAG
False Positive Rate	FPR	Number of incorrectly found edges in estimated CPDAG divided by number of true gaps in true CPDAG
True Discovery Rate	TDR	Number of correctly found edges divided by number of found edges both in estimated CPDAG

Table 2: Evaluation measures used to determine the goodness of fit for different simulation studies.

expected neighborhood of each node, i.e. how many edges exist on average between one specific node and all remaining ones. In the second step, n random data samples are drawn from the simulated DAG assuming a multivariate Gaussian distribution. For details on how the simulation works in the R package *pcalg* see section 4.1 of Kalisch and Bühlmann (2005).

After running all three versions of the PC-algorithm for different settings regarding the dimension (Section 5.1.2), the sample size (Section 5.1.3) and the ordering of the variables (Section 5.1.4), the run time in seconds s and goodness of fit measures are compared. For a description of the chosen evaluation measures see Table 2.

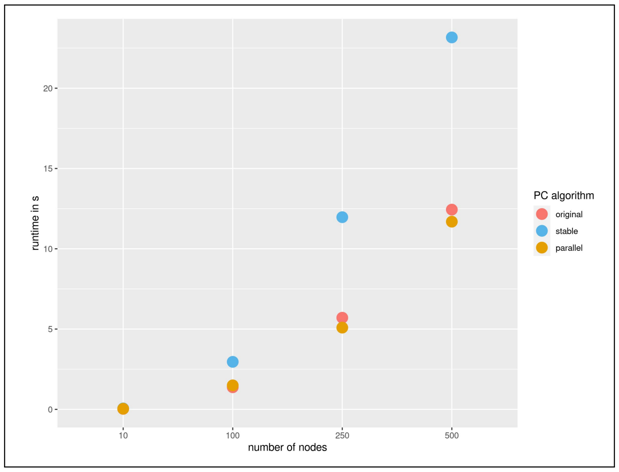
For all settings the significance level of the conditional independence tests is set to $\alpha = 0.01$ and 10 simulations are conducted and averaged to get comparable results.

5.1.2 Setting 1: Varying Dimension

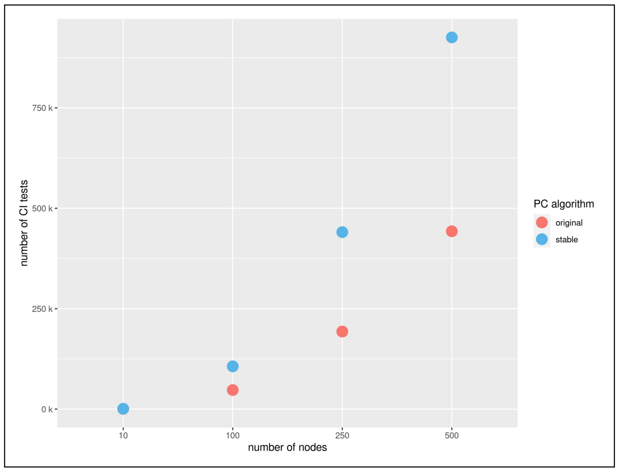
As a first step the performances of all three algorithms for different number of nodes $p \in \{10, 100, 250, 500\}$ are compared. For all 10 simulations 1000 samples are drawn randomly and the expected neighborhood size is set to five. Figure 6 shows the results.

Clearly, the run time in Figure 6a increases for increasing dimension. Interestingly, the run times of the original and the parallel PC-algorithm do not differ widely which is explained by Figure 6b. As declared in Section 4.1 the number of conditional independence tests is higher for the stable PC-algorithm because of possibly larger adjacency sets. As running conditional independence tests is the computationally most costly part, the stable PC-algorithm is the slowest of all three because no parallelization is done.

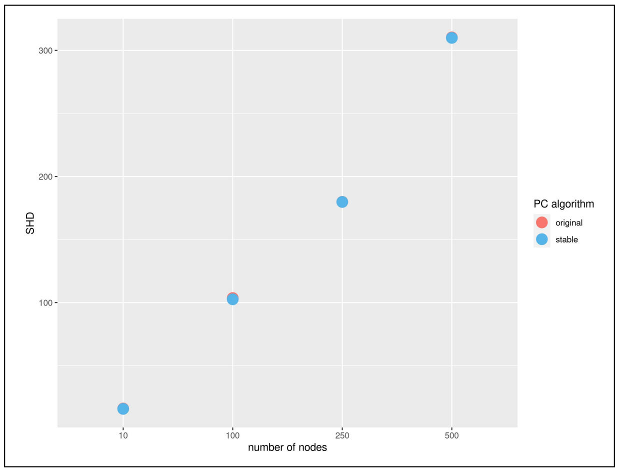
Regarding the goodness of fit, the plots only show the results for the original and the stable PC-algorithm, since both stable and parallel PC-algorithm produce the same output. As the number of nodes becomes larger the Structural Hamming Distance in Figure 6c increases, too, for all three algorithms. This can be explained by the fact, that for increasing dimensions, the total amount of edges goes up which rises the potential fore wrongly detected edges. Looking at the True Positive Rate (Figure 6d) the stable PC-algorithm tends to give slightly



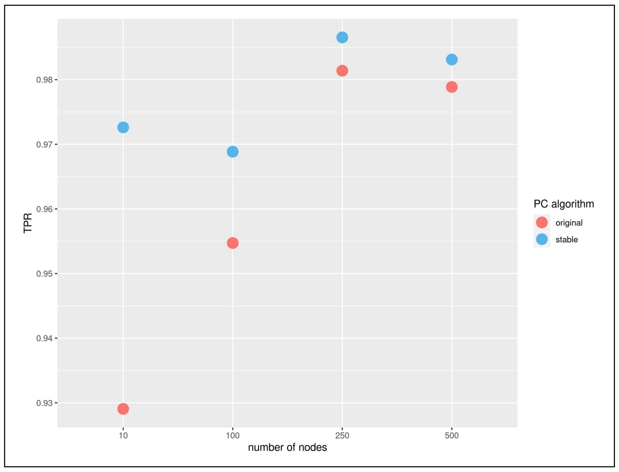
(a)



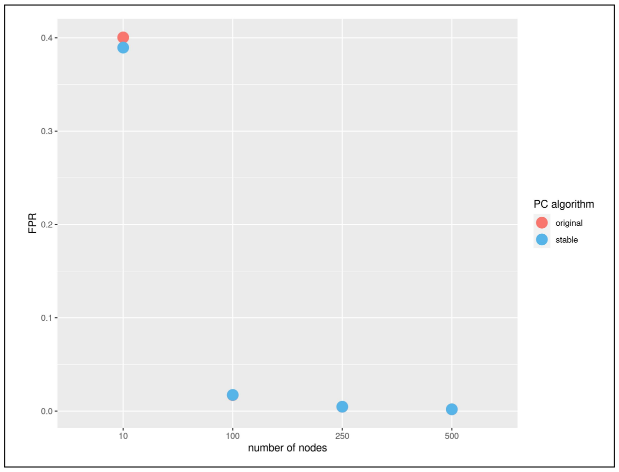
(b)



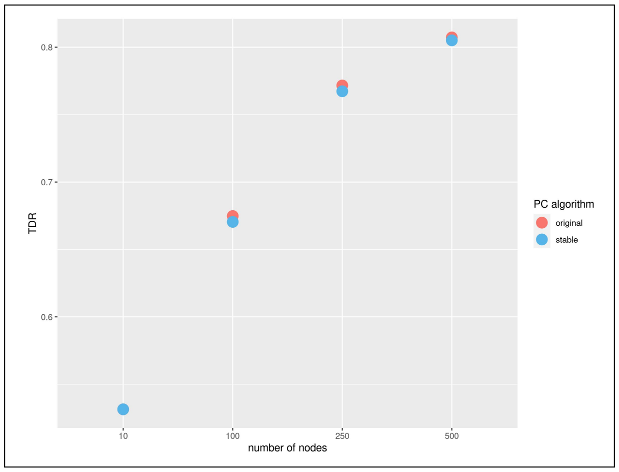
(c) SHD



(d) TPR



(e) FPR



(f) TDR

Figure 6: Runtime in seconds (a), number of conditional independence tests (b), SHD (c), TPR (d), FPR (e) and TDR (f) for varying number of nodes for all three algorithms, namely the original, stable and parallel PC-algorithm. As the stable and the parallel PC-algorithm produce the same result, the goodness of fit evaluated in (b) to (f) does not change.

better results than the original one. Interestingly, increasing the dimension leads to a better fit in the beginning but tends to deteriorate again for both algorithms. This result, which is surprising at first glance, can be explained by Equation 7. The "perfect" significance level α is dependent on a lower bound of partial correlations which itself is dependent on the dimension through $\mathcal{A}, \mathcal{B}$ and $\mathcal{S}$. As a result, ideally the significance level should be chosen according to the number of nodes (and sample size) which is hardly feasible in practice. Note however that the significance level in the setting of the PC-algorithm is taken as a tuning parameter that determines the sparsity of the output (further details in Kalisch and Bühlmann (2005)). The False Positive Rate in Figure 6e and the True Discovery Rate (Figure 6f) both improve for higher dimensions and are very similar for both algorithms. This leads to the conclusion that the PC-algorithm and its extensions are well-suited for high-dimensional data.

5.1.3 Setting 2: Varying Sample Size

In a second step the performance of the three algorithms is compared regarding different sample sizes $n \in \{50, 100, 500, 1000, 10000, 20000\}$. The number of nodes is set to 100 and the expected neighborhood size is once again set to five. The corresponding results are displayed in Figure 7.

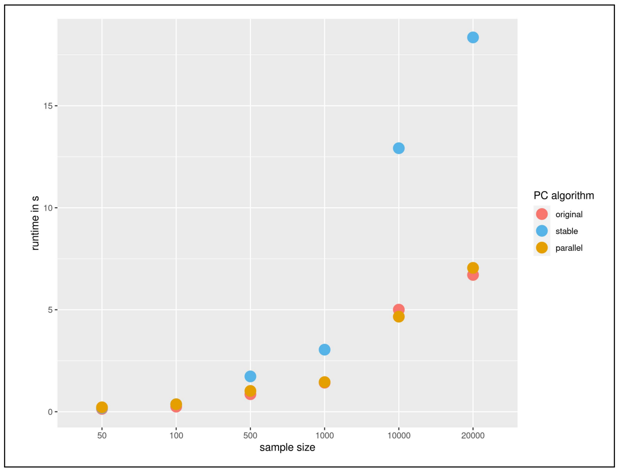
As one would expect, the run time of all algorithms in Figure 7a increases with increasing sample size, but once again, the effect for the stable PC-algorithm is the strongest one. As in the previous setting, the original and the parallel PC-algorithm do not differ substantially due to a lot more conditional independence tests for the stable PC-algorithm (Figure 7b).

Evaluating the fit using the Structural Hamming Distance, the False Positive Rate and the True Discovery Rate shows an improved fit for increased sample size and interestingly, the fit is similar for both original and stable PC-algorithm. As in the evaluation of the first setting, the True Positive Rate first increases and then decreases. This can be explained once again by the chosen significance level α in Equation 4 which is dependent on the sample size n and should be adapted in the ideal setting. Note that even if the rate decreases, the overall fit is still very good even but once again it is slightly better for the stable PC-algorithm.

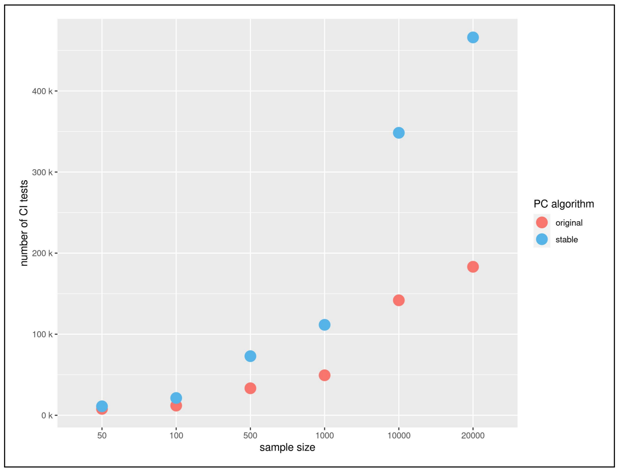
5.1.4 Setting 3: Varying Variable Order

In Section 3.4 the order-dependence of the PC-algorithm in its original form was discussed. While the simulation studies in the previous subsections did not detect any drastic differences concerning the fit between stable and original PC-algorithm, looking at the number of edges in the estimated skeletons reveals possible variation in the output of the PC-algorithm.

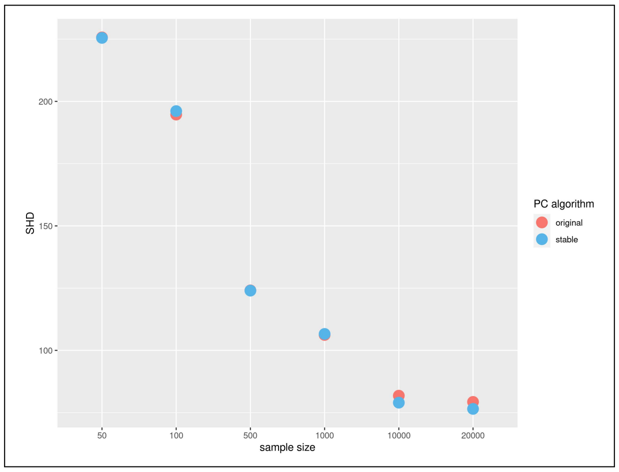
In Table 3 the number of edges of a simulation with 10 different orderings is displayed. For this example, 20 samples are drawn randomly from a DAG with 50 nodes and expected neighborhood size of three. The conditional independence tests are conducted on a significance level of $\alpha = 0.05$. Clearly, varying the ordering affects the total number of estimated edges in the original PC-algorithm which obviously could lead to a different result.



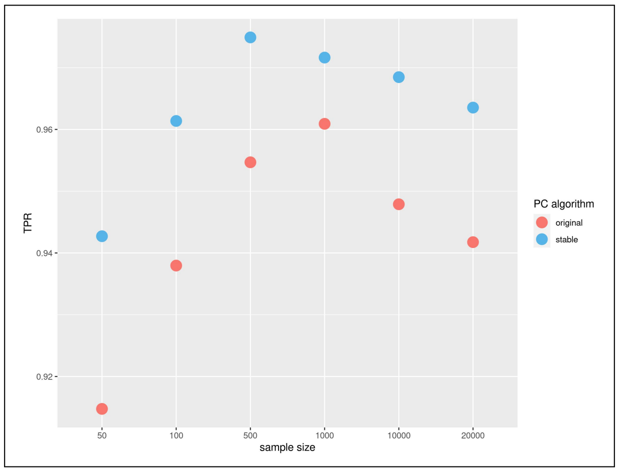
(a)



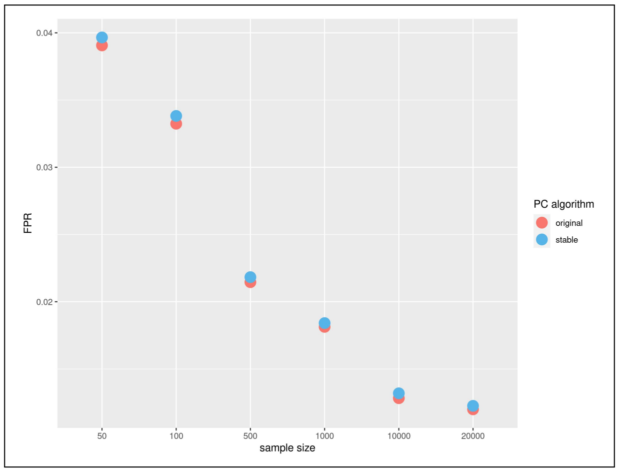
(b)



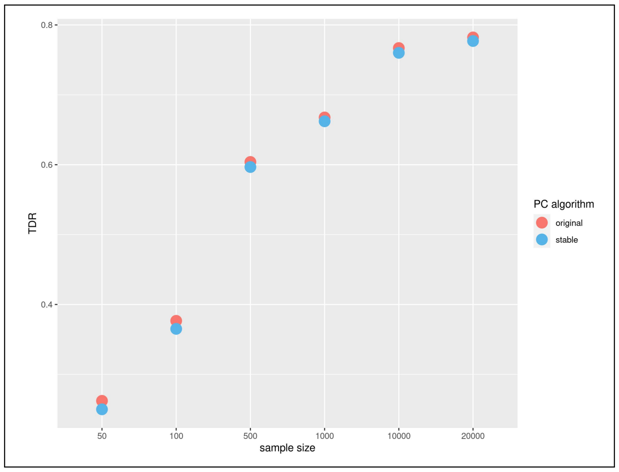
(c) SHD



(d) TPR



(e) FPR



(f) TDR

Figure 7: Runtime in seconds (a), number of conditional independence tests (b), SHD (c), TPR (d), FPR (e) and TDR (f) for varying number of random samples for all three algorithms, namely the original, stable and parallel PC-algorithm. As the stable and the parallel PC-algorithm produce the same result, the goodness of fit evaluated in (b) to (f) does not change.

ordering	1	2	3	4	5	6	7	8	9	10
original	106	108	110	110	110	108	110	110	110	106
stable	102	102	102	102	102	102	102	102	102	102

Table 3: Number of edges in the estimated CPDAG of the original and stable PC-algorithm for 10 different orderings of the variables.

5.2 Simulation Study for nonparanormal Data

So far, in every experiment the data was assumed to be multivariate Gaussian distributed so the PC-algorithm in its original form or the stable/parallel PC-algorithm were appropriate according to Theorem 3.4. In this chapter, the PC-algorithm and its extension to nonparanormal data, the rank PC-algorithm, is applied to non Gaussian data to evaluate their fits.

In Section 4.3 two possibilities for estimating the partial correlations between nodes were presented: empirical estimates of Spearman's rank correlation ρ^S (Equation 8) and Kendall's τ (Equation 9). In implementations of the rank PC-algorithm usually Spearman's rank correlation is used but as stated by Harris and Drton (2013), the differences between both estimation techniques are very small. Nevertheless, in this simulation study both are applied and compared to the standard Pearson estimate that was used in all other sections. All three correlation estimation methods are evaluated for Gaussian data and nonparanormal data. For the latter the marginals of the Gaussian vectors are transformed to a $F_{1,1}$-distribution as in the example mentioned above in Figure 5.

As in Section 5.1.1, random samples are drawn from random DAGs, here with $p = 100$ and $n = 1000$. The significance level is chosen to be $\alpha = 0.01$ and the expected neighborhood size is five. For the reason of comparability, all following evaluation measures are averaged over 10 equivalent simulation runs.

In Table 4 evaluation measures for Gaussian data are summarized. As expected, all three correlation estimators work similarly good and even if the Pearson coefficient is best in all four evaluation measures, the differences are negligible. Running the algorithms with nonparanormal data instead, severe differences appear. Table 5 shows that once again the fits of the rank PC-algorithms run with Kendall's τ and Spearman's rank correlation respectively are very much alike and equally good. The fit of the standard PC-algorithm on the other hand has worsen drastically for all four evaluation measures. Especially a true discovery rate of 5.39% indicates a very poor fit.

6 Example of Application

In addition to experiments on simulated DAGs, in this section the PC-algorithm is applied to a real world data set. This data set is taken from climate simulations that were conducted within the ClimEx-II Study of Prof. Ralf Ludwig (Ludwig-Maximilians-Universität Munich, Department of Geography). For details on the simulation process of the climate data, see

	SHD	TPR	FPR	TDR
Pearson	106.6	0.9788	0.0149	0.7070
Kendall	110.4	0.9733	0.0150	0.7043
Spearman	108.0	0.9733	0.0149	0.7061

Table 4: Evaluation measures for Gaussian data

	SHD	TPR	FPR	TDR
Pearson	253.3	0.4650	0.0463	0.0539
Kendall	108.5	0.9745	0.0150	0.7043
Spearman	103.1	0.9739	0.0150	0.7061

Table 5: Evaluation measures for nonparanormal data

a similar work from the same research group (Leduc et al. (2019)). The pre-processed data consists of time series of 31 years (1990-2020) on daily temperature (in °C) and precipitation (in mm) at gauging stations at rivers in Bavaria and surrounding areas, the so called "hydrological Bavaria". The total area is partitioned into 98 catchments where measurements are averaged over different weather stations in each catchment. In Figure 8 the territory is illustrated.

Taking the catchments as nodes ($p = 98$) and 10000 randomly chosen days ($n = 10000$) with values of the variable of interest, the graph structure of the data becomes apparent. The parallel PC-algorithm is applied to both temperature and precipitation with a significance level of $\alpha = 0.01$ using four cores of the computer's CPU. The calculation of the CPDAG took the algorithm 4.987 seconds for the temperature data and 5.739 seconds for the precipitation data respectively.

For the precipitation data, the skeleton of the estimated CPDAG is portrayed in Figure 9b. Obviously, between every pair of nodes there exists a path which shows that the precipitation in every gauging station is somehow connected to the precipitation of all remaining stations. To visualize how strong these connections are, one specific catchment is taken as reference and the number of edges of the shortest path to every other catchment is calculated. Figure 9a shows the respective map of hydrological Bavaria and a catchment in the middle (black) as reference. This indicated that the PC-algorithm captures spatial dependencies quite well.

On the other hand, for the temperature data, the estimated CPDAG consists of several separate subgraphs as shown in Figure 10b. This leads to the idea that the PC-algorithm could be used for clustering in this setting. Figure 10a displays the different subgraphs as clusters where every color stands for a different cluster. Note that the three black clusters are clusters consisting of only one catchment. Once again, the regionality is clearly visible and one can assume that the temperature depends on geographical conditions (e.g. alpine regions in the south).

These examples show that the PC-algorithm not only works well for simulated data but

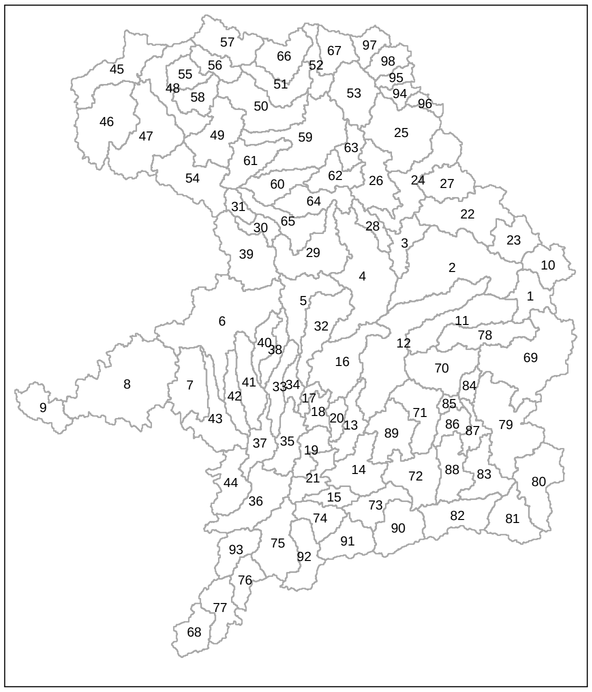


Figure 8: Map of hydrological Bavaria and corresponding gridcode of the 98 catchments.

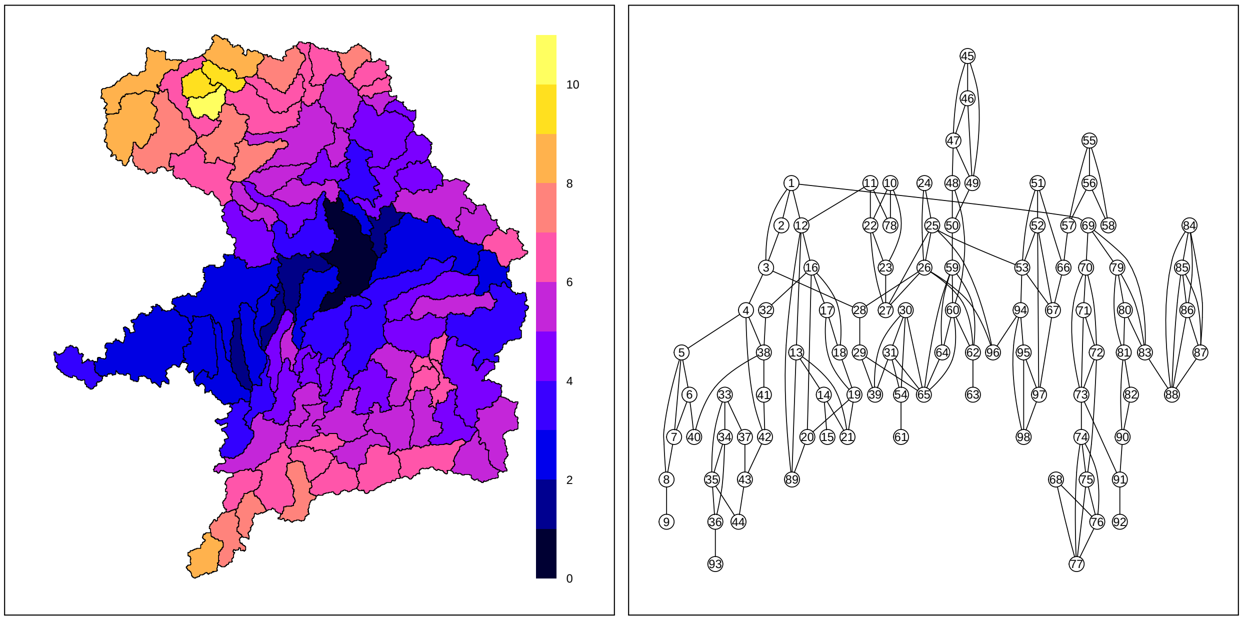
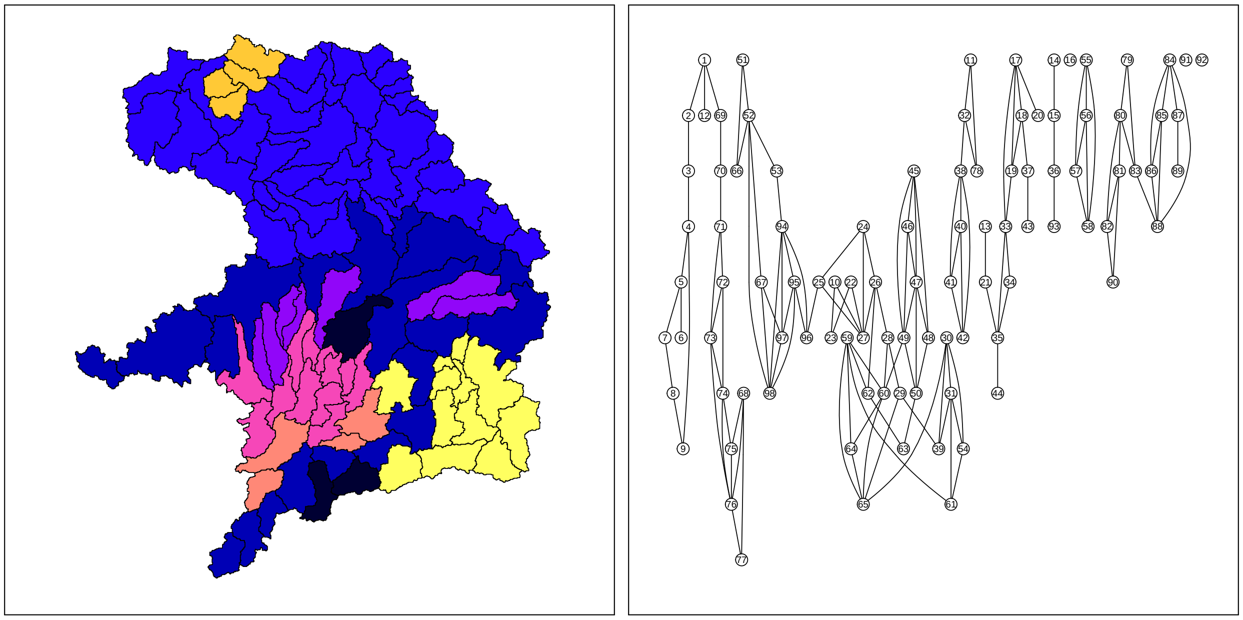


Figure 9: Visualization of the shortest paths (number of nodes on the path) between the black catchment and all other ones (a) and the skeleton estimated by the parallel PC-algorithm (b) for the precipitation data of hydrological Bavaria at 98 gauging stations.



(a) (b)

Figure 10: Visualization of the clusters (a) induced by the subgraphs of the skeleton estimated by the parallel PC-algorithm (b) for the temperature data of the hydrological Bavaria at 98 gauging stations. White clusters mean that these clusters consist of one catchment only (see nodes 16, 91, 92 in the skeleton).

also for real world applications. Further examples can be looked up in Colombo and Maathuis (2012) and Harris and Drton (2013).

7 Discussion

In summary, the PC-algorithm is an effective method to detect conditional independence relationships between nodes in a DAG whose underlying data generating distribution is faithful and multivariate Gaussian. Starting from a fully connected DAG, edges are iteratively eliminated by performing conditional independence tests for each pair of adjacent nodes to obtain the skeleton of the underlying DAG. In a second step, the algorithm tries to orient undirected edges by applying deterministic rules. As a result it outputs the corresponding CPDAG, i.e. a partially directed DAG that is Markov equivalent to the approximated true DAG and therefore encodes the same statistical model.

However, the PC-algorithm is subject to three limitations. First, it is order dependent on the input variables which is solved by the stable PC-algorithm which on the down side has an even longer run time. In order to tackle this task the parallel PC-algorithm speeds up the stable PC-algorithm by parallelizing conditional independence tests using multiple cores of the computer's CPU. Third, applying the original PC-algorithm to non-Gaussian data results in a poor fit. The rank PC-algorithm delivers an alternative for nonparanormal data by modifying the estimation process of marginal correlations.

Comparing the run time of the original, the stable and the parallel PC-algorithm in a simulation study showed that the latter is in deed the fastest. All three algorithms proved to be very similar regarding the goodness of fit, although the stable (or parallel) PC-algorithm performs better in terms of the True Positive Rate. Since the original PC-algorithm is in addition order-dependent I would recommend the stable or parallel PC-algorithm in practice. For nonparanormal data, the PC-algorithm based on Pearson-estimates gives a very poor fit in comparison to the rank PC-algorithm, while the fits for Gaussian data are comparably good. Applying the PC-algorithm to weather time series data showed that the PC-algorithm works quite well for spatial data since it manages to detect spatial dependencies.

Interestingly, CPDAGs estimated with the PC-algorithm not only offer the possibility to draw conclusions regarding conditional independence, but also to draw causal conclusions whenever the direction of edges is detected. With undirected edges, one can only infer a dependency of the nodes which includes confounding or bicausal relationships. However, it is important to note that the causal relationships inferred from the CPDAG are based on assumptions used to construct the graph and may not always reflect the true causal relationships in the observed data. One possibility to evaluate the strength of evidence for causal relationships are the Bradford Hill Criteria proposed by Hill (1965). Although not originally developed specifically for methods such as the PC-algorithm, they can serve as a general framework for assessing plausibility and strength of association.

In addition to the partial correlation estimation techniques presented in this report, there are several other methods. The graphical lasso proposed by J. Friedman, Hastie, and Tibshirani (2007) is an alternative approach to estimate the inverse covariance matrix. It is based on a regularized maximum likelihood estimator that solves an optimization problem by using an L1 penalty to ensure sparsity in the estimated matrix. Especially for data with many variables this technique is shown to perform very well.

Besides the PC-algorithm, there are various other algorithms to detect causal relationships in graph structures. As mentioned in the beginning, score-based methods can be used to determine CPDAGs. One example of these is the DAGs with no tears algorithm developed by Zheng et al. (2018) which is based on structural equation models. The authors of this paper extend the original algorithm to the so called Non-combinatorial Optimization via Trace Exponential and Augmented lagRangian for Structure learning (NOTears) algorithm. Here, the objective is to find the DAG that maximizes the likelihood of the observed data subject to certain constraints. The NOTears algorithm solves this optimization problem by minimizing the L1 penalty term and the augmented Lagrangian term.

To conclude, as shown in this report, the PC-algorithm has been extended and modified in many ways since its introduction. Nevertheless, it remains an active area of research in the field to further improve its overall fit and run time.

Acknowledgements

Special thanks to Prof. Ralf Ludwig, Andrea Böhnisch and Alexander Sasse for providing data for the application example in Section 6.

References

Chickering, David Maxwell (2003). "Optimal Structure Identification with Greedy Search". In: *J. Mach. Learn. Res.* 3(null), pp. 507–554. DOI: 10.1162/153244303321897717. URL: https://doi.org/10.1162/153244303321897717.

Colombo, Diego and Marloes H. Maathuis (2012). *Order-independent constraint-based causal structure learning*. DOI: 10.48550/ARXIV.1211.3295. URL: https://arxiv.org/abs/1211.3295.

Cui, Ruifei, Perry Groot, and Tom M. Heskes (2016). "Copula PC Algorithm for Causal Discovery from Mixed Data". In: *ECML/PKDD*.

Friedman, Jerome, Trevor Hastie, and Robert Tibshirani (2007). "Sparse inverse covariance estimation with the graphical lasso". In: *Biostatistics* 9(3), pp. 432–441.

Harris, Naftali and Mathias Drton (2013). "PC Algorithm for Nonparanormal Graphical Models". In: *Journal of Machine Learning Research* 14. URL: http://jmlr.org/papers/volume14/harris13a/harris13a.pdf.

Hill, Ab (1965). "The Environment and Disease: Association or Causation?" In: *Proceedings of the Royal Society of Medicine* 58, pp. 295–300. ISSN: 0035-9157. DOI: 10.1177/003591576505800503. URL: https://europepmc.org/articles/PMC1898525.

Kalisch, Markus and Peter Bühlmann (2005). *Estimating high-dimensional directed acyclic graphs with the PC-algorithm*. DOI: 10.48550/ARXIV.MATH/0510436. URL: https://arxiv.org/abs/math/0510436.

Kalisch, Markus, Martin Mächler, et al. (2012). "Causal Inference Using Graphical Models with the R Package pcalg". In: *Journal of Statistical Software* 47(11), pp. 1–26. URL: https://www.jstatsoft.org/index.php/jss/article/view/v047i11.

Koller, D. and N. Friedman (2009). *Probabilistic Graphical Models: Principles and Techniques*. Adaptive computation and machine learning. MIT Press. ISBN: 9780262013192. URL: https://books.google.co.in/books?id=7dzpHCHzNQ4C.

Le, Thuc Duy et al. (Sept. 2019). "A Fast PC Algorithm for High Dimensional Causal Discovery with Multi-Core PCs". In: *IEEE/ACM Transactions on Computational Biology and Bioinformatics* 16(5), pp. 1483–1495. URL: https://doi.org/10.1109%5C%2Ftcbb.2016.2591526.

Leduc, Martin et al. (2019). "The ClimEx Project: A 50-Member Ensemble of Climate Change Projections at 12-km Resolution over Europe and Northeastern North America with the Canadian Regional Climate Model (CRCM5)". In: *Journal of Applied Meteorology and Climatology* 58, pp. 663–693. URL: https://doi.org/10.1175/JAMC-D-18-0021.1.

Liu, Han, Fang Han, et al. (2012). "High-dimensional semiparametric Gaussian copula graphical models". In: *The Annals of Statistics* 40(4), pp. 2293–2326. DOI: 10.1214/12-AOS1037. URL: https://doi.org/10.1214/12-AOS1037.

Liu, Han, John Lafferty, and Larry Wasserman (2009). "The Nonparanormal: Semiparametric Estimation of High Dimensional Undirected Graphs". In: *Journal of Machine Learning Research* 10. URL: http://www.jmlr.org/papers/volume10/liu09a/liu09a.pdf.

Meek, Christopher (1995a). "Causal Inference and Causal Explanation with Background Knowledge". In: *Proceedings of the Eleventh Conference on Uncertainty in Artificial In-*

telligence. UAI'95. Morgan Kaufmann Publishers Inc.: Montréal, Qué, Canada, pp. 403–410. ISBN: 1558603859.

Meek, Christopher (1995b). *Strong Completeness and Faithfulness in Bayesian Networks*. DOI: 10.48550/ARXIV.1302.4973. URL: https://arxiv.org/abs/1302.4973.

Spirtes, P., C. Glymour, and R. Scheines (2000). *Causation, Prediction, and Search*. 2nd. MIT press.

Zheng, Xun et al. (2018). *DAGs with NO TEARS: Continuous Optimization for Structure Learning*. DOI: 10.48550/ARXIV.1803.01422. URL: https://arxiv.org/abs/1803.01422.

Declaration of authorship

I hereby declare that the report submitted is my own unaided work. All direct or indirect sources used are acknowledged as references. I am aware that the report in digital form can be examined for the use of unauthorized aid and in order to determine whether the report as a whole or parts incorporated in it may be deemed as plagiarism. For the comparison of my work with existing sources I agree that it shall be entered in a database where it shall also remain after examination, to enable comparison with future reports submitted. Further rights of reproduction and usage, however, are not granted here. This paper was not previously presented to another examination board and has not been published.

Munich, March 1st, 2023

Theresa Meier

Theresa Meier