Department of Mathematics
TUM School of Computation, Information and Technology
Technical University of Munich

Y-Vine Copula Based Structure Learning for Continuous Bayesian Networks

Brian Zeiser Dietrich

Thesis for the attainment of the academic degree

Master of Science

at the TUM School of Computation, Information and Technology of the Technical University of Munich

Supervisor:
Prof. Claudia Czado, Ph.D.

Advisors:
Prof. Claudia Czado, Ph.D.

Submitted:
Munich, September 27, 2024

I hereby declare that this thesis is entirely the result of my own work except where otherwise indicated. I have only used the resources given in the list of references.

[illegible]

Munich, September 27, 2024 Brian Zeiser Dietrich

Abstract

Bayesian networks are powerful graphical models that capture the probabilistic dependencies between random variables. Pair-copula Bayesian networks (Bauer et al. 2011) extend the well-known Gaussian Bayesian networks by allowing for non-Gaussian distributions through the incorporation of bivariate copulas and univariate marginal distributions into the network structure.

A widely-used method for learning the structure of Bayesian networks is the constraint-based PC algorithm (Spirtes et al. 1993), which employs conditional independence tests to identify the absence of edges in an underlying directed acyclic graph (DAG). In the case of continuous data, Fisher's Z-test of partial correlation is commonly used as a benchmark for testing conditional independencies.

However, a significant limitation of this approach is its reliance on the assumption of a multivariate Gaussian distribution, under which partial and conditional correlations coincide, and vanishing correlations imply independence. To overcome this limitation, a novel conditional independence test based on Y-vine copulas is introduced.

Y-vine copulas, a subclass of regular vine copulas introduced by Tepegjozova and Czado (2023), are designed to model bivariate conditional distributions using only univariate distributions and bivariate copulas, thus avoiding the need for integration. Their inherent flexibility allows them to model data without making assumptions about the underlying distribution.

The modified PC algorithm, utilizing Y-vines to facilitate conditional independence testing, will be evaluated against its traditional counterpart in a simulation study using both Gaussian and non-Gaussian data generated from Bayesian networks. To produce non-Gaussian data, a new simulation procedure based on D-vines is proposed, which enables approximate sampling from a pair-copula Bayesian network without imposing constraints on the underlying structure or the order of the parent nodes. The results of the simulation study highlight that the Y-vine-based PC algorithm more accurately recovers the true underlying graphical structure than the Z-test benchmark, albeit with significantly increased computational effort.

Finally, it will be demonstrated that univariate D-vine-based regression can effectively learn the parent order of a node and parameters in pair-copula Bayesian networks. The combination of the Y-vine-based PC algorithm and the D-vine-based parameter learning method will be applied to estimate pair-copula Bayesian networks in real-world case studies.

Contents

1 Introduction and Overview

In many fields, modeling the complex relationships between variables is essential for capturing both their interdependencies and the inherent uncertainty in their interactions. To address this need, Bayesian networks offer a powerful approach. These graphical models are designed to represent the probabilistic relationships among variables, allowing for a structured framework that reflects how variables influence each other. By representing these dependencies visually and probabilistically, Bayesian networks enhance decision-making and prediction capabilities. However, while they are effective in many scenarios, traditional continuous Bayesian networks often rely on Gaussian assumptions that may not capture all types of real-world dependencies. This limitation has led to the development of more advanced models, such as pair-copula Bayesian networks (PCBNs), which use copulas to model non-Gaussian dependencies and better reflect complex, real-world interactions (Bauer et al. 2011).

A significant challenge with PCBNs is that they often involve conditional cumulative distribution functions in the factorization of the network's probability density function, which can only be computed through integration (Bauer 2013). Consequently, not all PCBNs can be represented using regular vines, and exact likelihood inference and sampling can become computationally intensive, particularly for large-dimensional networks.

This thesis addresses these challenges by investigating an alternative approximate method for likelihood inference and sampling through D-vine structural equation models (DV-SEMs) (Czado and Scharl 2021). The approach involves constructing a D-vine for each node with multiple parents, where the node of interest is treated as a leaf node followed by its parents in a specified order. In cases where the parent orders and specific pair-copulas of a PCBN are unknown, D-vine-based regression (Kraus and Czado 2017) is used to estimate all D-vines associated with the DV-SEM. This includes a forward selection algorithm that sequentially determines the covariate order in the first D-vine tree based on importance. By estimating all pair-copulas required for computing the density function of a specific node given its parents, this method facilitates integration-free sampling from the PCBN and likelihood computation.

Learning the parameters of a PCBN using D-vine-based regression presupposes knowledge of the network structure. While expert knowledge can sometimes provide this structure, it is not always available, and proposed structures may not always fit the dataset. Therefore, efficient and accurate structure learning algorithms are often necessary.

One commonly used method for structure learning is the PC algorithm (Spirtes et al. 1993), which identifies node relationships through conditional independence tests. Traditionally, Fisher's Z-test (Fisher 1924) has been used to assess conditional independence by analyzing partial correlations. However, this test is constrained by its assumption of normally distributed data. Specifically, Fisher's Z-test relies on the alignment of partial and conditional correlations, with a partial correlation of zero indicating conditional independence. This alignment is only guaranteed in Gaussian distributions (Baba et al. 2004). Consequently, if a PCBN exhibits non-Gaussian dependencies, Fisher's Z-test may not reliably detect conditional independencies.

To address these limitations, this thesis introduces a novel structure learning method based on Y-vine copulas (Tepegjozova and Czado 2023), a subclass of regular vine copulas. Y-vine copulas provide a more flexible approach for modeling non-Gaussian dependencies in conditional independence tests used by the PC algorithm. For testing conditional independence statements of the form $X_i \perp\!\!\!\perp X_j \mid \mathbf{X}_\mathrm{S}$, Y-vine-based regression is applied to data containing variables i, j, and S, with i and j as response variables and S as covariates. The conditional copula $C_{i,j;\mathrm{S}}$ is obtained from the final tree of the fitted Y-vine, and the Y-vine-based conditional independence test uses a simple upper bound on the estimated τ-value of this copula to assess conditional independence.

The thesis aims to evaluate whether Y-vine-based structure learning can outperform traditional Fisher's Z-test in both Gaussian and non-Gaussian data scenarios. This involves a simulation study across various dimensions and PCBN specifications, where data is sampled using the approximate method introduced. The fitted models are compared based on how closely their estimated structures match the true underlying structure, and the models are further compared on probability levels using D-vine-based parameter learning and approximate likelihood inference. The simulation results show that the PC algorithm incorporating Y-vine conditional independence testing more effectively recovers the true graphical structure compared to the Z-test-based PC algorithm, especially in scenarios with strong non-Gaussian dependence structures.

Ultimately, the PC algorithm, incorporating both the novel Y-vine-based conditional independence test and Fisher's Z-test for partial correlation, is applied to two real-world examples. The first example comes from aviation and includes 711 observations across 12 variables related to Boeing 747-8 landings at a specific airport. In this case, understanding the interdependencies among landing parameters is crucial for improving flight safety. The second example is drawn from biology, utilizing data from the Sachs dataset (Sachs et al. 2005), which comprises 911 observations of 11 variables. This experiment aims to investigate the causal relationships between various molecular entities in human cells. In both cases, the learned structures are compared to expert-knowledge-based models. After fitting the parameters of both the learned and expert Bayesian networks, it will be shown that the fitted, sparser structures achieve better penalized fit statistics, indicating that model complexity can be significantly reduced. Moreover, for the Sachs dataset, the Y-vine-based model outperforms the Z-test-based approach in terms of AIC and BIC.

2 Mathematical Foundations

2.1 Notation

- Random variables will be denoted by capital letters such as X, Y, and Z, and their realizations will be denoted by lowercase letters such as x, y, and z.
- All random variables are assumed to be absolutely continuous, and therefore all (conditional) densities exist.
- The letter f is used to denote a probability density function (pdf), and the corresponding cumulative distribution function (cdf) is denoted by the letter F.
- Random vectors in d dimensions will be denoted in bold letters such as $\mathbf{X} = (X_1, \dots, X_d)^\top$, and subsets of random vectors will have a subscript like $\mathbf{X}_S$, where $S \subseteq \{1, \dots, d\}$. Similarly, observations of random vectors will be in bold lowercase letters such as $\mathbf{x} = (x_1, \dots, x_d)^\top$.
- Marginal distributions will have a subscript that refers to the random variable. For example, $f_j(x_j)$ is the marginal density function of X_j in $\mathbf{X} = (X_1, \dots, X_d)^\top$, $j = 1, \dots, d$.
- Conditional densities and distribution functions use subscripts of the form $j \mid k$ to indicate the conditional distribution of X_j given X_k for $j \neq k$ (e.g., $f_{j|k}(x_j \mid x_k)$ is the conditional density function).
- If it is not clear from the context which distribution is being referred to, joint densities and distribution functions may contain multiple subscripts. For example, $f_{1234}(x_1, x_2, x_3, x_4)$ is the density function of $(X_1, X_2, X_3, X_4)^\top$, and $F_{134}(x_1, x_3, x_4)$ is the cdf of $(X_1, X_3, X_4)^\top$.
- The independence of two random variables X and Y is denoted by $X \perp\!\!\!\perp Y$. Similarly, dependence is denoted by $X \not\perp\!\!\!\perp Y$.
- Conditional independence is denoted by $X \perp\!\!\!\perp Y \mid \mathbf{Z}$, where $\mathbf{Z}$ is a possibly empty set of random variables. In the case of $\mathbf{Z} = \emptyset$, $X \perp\!\!\!\perp Y \mid \mathbf{Z}$ corresponds to the ordinary independence $X \perp\!\!\!\perp Y$.

2.2 Graphical Models

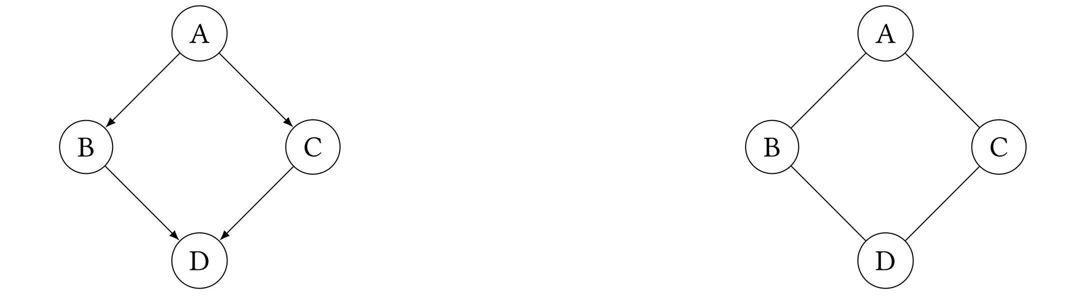
In graphical models, the graph's structure expresses the probabilistic dependencies among the variables it represents. Analyzing the graphical structure using tools from graph theory can reveal much about a model's properties. To facilitate this analysis, it is necessary to introduce some basic terminology, which will be sufficient for this thesis. This terminology is mainly adapted from Nagarajan et al. (2014) and Edwards (2000).

2.2.1 Graph Theory

Definition 2.2.1 (Graph)**.** A **graph**, $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, consists of a finite set $\mathcal{V}$ of **nodes** and a finite set $\mathcal{E}$ of **edges**, where $\mathcal{E} \subseteq \mathcal{V} \times \mathcal{V}$, and $(A, A) \notin \mathcal{E}$ for all $A \in \mathcal{V}$. A graph is said to be n-dimensional, $n \in \mathbb{N}$, if it contains exactly n nodes.

Definition 2.2.2 (Undirected/directed)**.** An edge $(A, B) \in \mathcal{E}$ is called **undirected** if $(B, A) \in \mathcal{E}$ holds as well. In this case, the notation $A - B$ is used to show that the order of the nodes is interchangeable. Similarly, an edge $(A, B) \in \mathcal{E}$ is called **directed** if $(B, A) \notin \mathcal{E}$. This is denoted by $A \longrightarrow B$. A graph is called undirected (resp. directed) if it contains only undirected (resp. directed) edges. A graph that may contain both undirected and directed edges is referred to as **partially directed**.

Definition 2.2.3 (Skeleton)**.** A **skeleton** is an undirected graph obtained from a (partially) directed graph by replacing all directed edges with undirected edges.



(a) A directed graph in four dimensions **(b)** The skeleton of the graph in (a)

Figure 2.1 Comparison of a directed graph and its skeleton

Figure 2.1a depicts a directed graph in four dimensions, while Figure 2.1b illustrates its skeleton, which is defined as an undirected graph.

Definition 2.2.4 (Path, trail)**.** Let $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ be a graph and $n \geq 2$.

i) A **path of length n from A_0 to A_n** is a sequence of edges $P(A_0; A_n) = \big((A_{i-1}, A_i)\big)_{i=1,\dots,n}$ connecting the nodes $A_0, A_1, \dots, A_n \in \mathcal{V}$ such that
   - $(A_{i-1}, A_i) \in \mathcal{E}$ for all $i = 1, \dots, n$.
   - All edges in $P(A_0; A_n)$ are unique (both directed and undirected).

ii) A **trail of length n from A_0 to A_n** is a path where the direction of the edges is ignored, i.e., a sequence of edges $T(A_0; A_n) = (e_i)_{i=1,\dots,n}$ such that
   - $e_i \in \{(A_{i-1}, A_i), (A_i, A_{i-1})\}$ for all $i = 1, \dots, n$.
   - $e_i \in \mathcal{E}$ for all $i = 1, \dots, n$.
   - All edges in $T(A_0; A_n)$ are unique (both directed and undirected).

In an undirected graph, a path from A_0 to A_n may also be denoted as $A_0 - A_1 - \cdots - A_n$. In a directed graph, it can be represented as $A_0 \longrightarrow A_1 \longrightarrow \cdots \longrightarrow A_n$. Additionally, in both cases, the sequence of nodes $A_0, A_1, \dots, A_n$ may simply be used as another notation.

Definition 2.2.5 (v-structure, diverging structure)**.** Let $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ be a n-dimensional graph where $n \geq 3$, and let $A, B, C \in \mathcal{V}$. A **v-structure** is a trail of the form $A \longrightarrow B \longleftarrow C$, and a **diverging structure** is a trail of the form $A \longleftarrow B \longrightarrow C$. In these cases, the trail is said to contain **a v-structure (or diverging structure) around B**.

Note that from the definition of a path, it follows that all directed edges forming the path must point in the same direction along the sequence; thus, diverging structures and v-structures cannot be part of the path. For this reason, there is, for example, no path from node B to node C or vice versa in Figure 2.1a. In contrast, trails do not impose any restrictions on the directions of the edges. They can be viewed as paths in the skeleton of a graph where the edge directions are inherited from the original graph. In the skeleton shown in Figure 2.1b, for instance, there exists the path $B - D - C$, which corresponds to the trail $B \longrightarrow D \longleftarrow C$ in Figure 2.1a.

Definition 2.2.6 (Cycle)**.** Let $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ be a graph. A **cycle** is a path of length $n \geq 3$ where the starting node and end node are identical, i.e., a path of the form $A_1, \dots, A_n, A_1$. A graph is called **acyclic** if it contains no cycles.

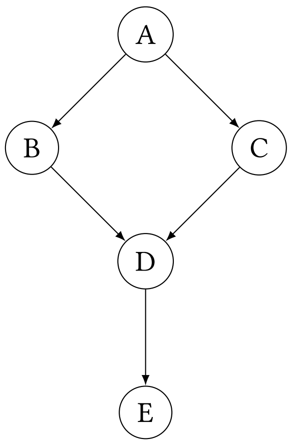


Figure 2.2 A five dimensional DAG. Node A blocks the trail $B \leftarrow A \rightarrow C$ and node E activates $B \rightarrow D \leftarrow C$

Throughout this thesis, the focus will be on a special class of graphs that serve as the foundation for Bayesian networks.

Definition 2.2.7 (Directed acyclic graph)**.** A graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ is called a **directed acyclic graph (DAG)** if it is directed and acyclic.

The directed graph in Figure 2.2 is acyclic and therefore a DAG.

Definition 2.2.8 (Ancestral relations)**.** Let $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ be a DAG. For any node $A \in \mathcal{V}$, consider the following ancestral relations:

i) **Parents of A:** $pa(A) := \{B \in \mathcal{V} \mid (B, A) \in \mathcal{E}\}$.

ii) **Ancestors of A:** $an(A) := pa(A) \cup \{B \in \mathcal{V} \mid \exists$ a path from B to $A\}$.

iii) **Children of A:** $ch(A) := \{B \in \mathcal{V} \mid (A, B) \in \mathcal{E}\}$.

iv) **Descendants of A:** $des(A) := ch(A) \cup \{B \in \mathcal{V} \mid \exists$ a path from A to $B\}$.

v) **Non-descendants of A:** $nd(A) := \mathcal{V} \setminus (des(A) \cup \{A\})$.

vi) **Neighbors/Adjacencies of A:** $adj(A) := pa(A) \cup ch(A)$.

Example 2.2.1. Consider the DAG in Figure 2.2. Its ancestral relations are given in Table 2.1.

Node $i \in \mathcal{V}$	$pa(i)$	$an(i)$	$ch(i)$	$des(i)$	$nd(i)$	$adj(i)$
A	$\emptyset$	$\emptyset$	$\{B, C\}$	$\{B, C, D, E\}$	$\emptyset$	$\{B, C\}$
B	$\{A\}$	$\{A\}$	$\{D\}$	$\{D, E\}$	$\{A, C\}$	$\{A, D\}$
C	$\{A\}$	$\{A\}$	$\{D\}$	$\{D, E\}$	$\{A, B\}$	$\{A, D\}$
D	$\{B, C\}$	$\{B, C, A\}$	$\{E\}$	$\{E\}$	$\{A, B, C\}$	$\{B, C, E\}$
E	$\{D\}$	$\{D, B, C, A\}$	$\emptyset$	$\emptyset$	$\{A, B, C, D\}$	$\{D\}$

Table 2.1 Ancestral Relations of the five dimensional DAG in Figure 2.2

An important characteristic of DAGs is that their structure inherently defines a topological order of the nodes.

Definition 2.2.9 (Topological order)**.** Let $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ be a n-dimensional DAG. A **topological order** is a total ordering of all nodes $A_1, A_2, \ldots, A_n$, such that for every edge $(A_i, A_j) \in \mathcal{E}$, it holds that $i < j$, where $i, j \in \{1, \ldots, n\}$.

Bang-Jensen and Gutin (2008) note that every DAG has a topological order of its nodes and provide an algorithm that can be used to determine such an order in linear time. The topological order is not necessarily unique, as illustrated in Figure 2.1a. Both the order $A < B < C < D$ and $A < C < B < D$ are valid topological orders since, by definition, it only needs to hold that $A < \{B, C\}$ and $\{B, C\} < D$.

To conclude this section on definitions from graph theory, the concept of d-separation as stated in Pearl (1988) will now be introduced. This concept will prove to be an important connection between the structure of a Bayesian network and the dependence relationships it exhibits.

Definition 2.2.10 (d-separation). Let $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ be a DAG and $\mathbf{X}, \mathbf{Y}$, and $\mathbf{Z}$ be three disjoint subsets of $\mathcal{V}$. $\mathbf{Z}$ **d-separates** $\mathbf{X}$ from $\mathbf{Y}$ in $\mathcal{G}$, denoted by $< \mathbf{X} \mid \mathbf{Z} \mid \mathbf{Y} >_{\mathcal{G}}$, if for all trails between a node in $\mathbf{X}$ and a node in $\mathbf{Y}$, there exists a node v satisfying one of the following conditions:

i) The trail contains a v-structure around v, and for all $w \in \{v\} \cup des(v)$, it holds that $w \notin \mathbf{Z}$.

ii) The trail does not contain a v-structure around v, and $v \in \mathbf{Z}$.

If a trail satisfies these conditions, it is said to be **blocked** by Z; otherwise, it is referred to as **activated** by Z.

Example 2.2.2. Consider the DAG directly taken from Pearl (1988) in Figure 2.2 and the sets $\mathbf{X} = \{B\}$, $\mathbf{Y} = \{C\}$, $\mathbf{Z} = \{A\}$, and $\mathbf{Z}' = \{A, E\}$. Suppose one wants to check if $< \mathbf{X} \mid \mathbf{Z} \mid \mathbf{Y} >_{\mathcal{G}}$ and $< \mathbf{X} \mid \mathbf{Z}' \mid \mathbf{Y} >_{\mathcal{G}}$ hold. First, note that there are two trails from $\mathbf{X}$ to $\mathbf{Y}$, namely $B \leftarrow A \rightarrow C$ and $B \rightarrow D \leftarrow C$.

i) $B \leftarrow A \rightarrow C$:
   - Diverging structure around A and $A \in \mathbf{Z} \Rightarrow$ trail is blocked by $\mathbf{Z}$.
   - Diverging structure around A and $A \in \mathbf{Z}' \Rightarrow$ trail is blocked by $\mathbf{Z}'$.

ii) $B \rightarrow D \leftarrow C$:
   - v-structure around D and for all $w \in \{D\} \cup des(D) = \{D, E\}$ it holds that $w \notin \mathbf{Z} \Rightarrow$ trail is blocked by $\mathbf{Z}$.
   - v-structure around D but for $E \in \{D\} \cup des(D) = \{D, E\}$ it holds that $E \in \mathbf{Z}' \Rightarrow$ trail is activated by $\mathbf{Z}'$.

Since both trails are blocked by $\mathbf{Z}$, the d-separation $< \mathbf{X} \mid \mathbf{Z} \mid \mathbf{Y} >_{\mathcal{G}}$ holds. However, since one trail is activated by $\mathbf{Z}'$, the sets $\mathbf{X}$ and $\mathbf{Y}$ are not d-separated by $\mathbf{Z}'$ in $\mathcal{G}$.

2.2.2 Bayesian Networks

Throughout this thesis, the focus will be on a special class of graphical models known as Bayesian networks. Bayesian networks display the probabilistic dependencies between a set of random variables $\mathbf{X} = \{X_1, \ldots, X_d\}$ using a d-dimensional DAG $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, where each node $i \in \mathcal{V}$, $i = 1, \ldots, d$, represents a variable $X_i \in \mathbf{X}$. More details on the content of this chapter can be found in Nagarajan et al. (2014), Lauritzen (1996), Pearl (1988) and Edwards (2000).

Definition 2.2.11 (Maps). A d-dimensional DAG, $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, $\mathcal{V} = \{1, \ldots, d\}$, is an **independence map (I-map)** of the distribution $\mathcal{P}$ of a d-dimensional random vector $\mathbf{X} = (X_v)_{v \in \mathcal{V}}$ if for every three disjoint subsets $A, B, C \subseteq \mathcal{V}$ it holds that

$$< \mathrm{A} \mid \mathrm{C} \mid \mathrm{B} >_{\mathcal{G}} \Longrightarrow \mathbf{X}_{\mathrm{A}} \perp\!\!\!\perp \mathbf{X}_{\mathrm{B}} \mid \mathbf{X}_{\mathrm{C}}. \tag{2.1}$$

An I-map is **minimal** if no edge in $\mathcal{E}$ can be removed without violating the property of being an I-map. Further, $\mathcal{G}$ is a **dependence map (D-map)** of $\mathcal{P}$ if

$$\mathbf{X}_{\mathrm{A}} \perp\!\!\!\perp \mathbf{X}_{\mathrm{B}} \mid \mathbf{X}_{\mathrm{C}} \Longrightarrow < \mathrm{A} \mid \mathrm{C} \mid \mathrm{B} >_{\mathcal{G}}.$$

holds. If $\mathcal{G}$ is an I-map and a D-map of $\mathcal{P}$, $\mathcal{P}$ is said to be **faithful** to $\mathcal{G}$, and $\mathcal{G}$ is said to be a **perfect map** of $\mathcal{P}$.

The Equation (2.1) is also widely known as the **global Markov property (global MP)** of a distribution $\mathcal{P}$ in relation to a DAG $\mathcal{G}$. The global MP states that d-separation in the graphical sense implies conditional independence in the probabilistic sense. In other words, this means that the DAG encodes all dependencies of $\mathcal{P}$. However, if the DAG is only an I-map and not a perfect map, there might still be independencies of random variables that are not represented by d-separation in the graph.

The following formal definition of a Bayesian network is according to Pearl (1988).

Definition 2.2.12 (Bayesian network). Given a probability distribution $\mathcal{P}$ on a set of variables $\mathbf{X}$, $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ is a **Bayesian network** of $\mathcal{P}$ if and only if $\mathcal{G}$ is a minimal I-map of $\mathcal{P}$.

While Pearl (1988) defines a Bayesian network via the global Markov property, it is interesting to note that there are different characterizations of Bayesian networks, as will be shown now.

Definition 2.2.13 (Local Markov Property). A probability distribution $\mathcal{P}$ obeys the **local Markov property (local MP)** with respect to a DAG $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, $\mathcal{V} = \{1, \ldots, d\}$, if for all $v \in \mathcal{V}$, it holds that

$$X_v \perp\!\!\!\perp \mathbf{X}_{nd(v)} \mid \mathbf{X}_{pa(v)}. \tag{2.2}$$

In fact, the global MP and the local MP are equivalent, as shown in Lauritzen (1996). A central result that will be used extensively in this thesis is the following:

Lemma 2.2.14 (Factorization). *A probability distribution $\mathcal{P}$ factorizes over a DAG $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ if its density satisfies*

$$f(\mathbf{x}) = \prod_{v \in \mathcal{V}} f_{v|pa(v)}(x_v \mid \mathbf{x}_{pa(v)}). \tag{2.3}$$

If $\mathcal{G}$ is a Bayesian network of $\mathcal{P}$, then $\mathcal{P}$ admits (2.3).

Proof. By definition, a Bayesian network obeys the global MP.

i) Global MP $\Longrightarrow$ Local MP
   Consider w.l.o.g. a d-dimensional Bayesian network with the topological order $1 < 2 < \cdots < d$. For any $\{i\} \in \mathcal{V}$, it is evident that $< \{i\} \mid pa(i) \mid nd(i) \setminus pa(i) >_{\mathcal{G}}$ holds because all trails from $\{i\}$ to $nd(i) \setminus pa(i)$ contain a node $v \in pa(i)$ with no v-structure around it, thereby blocking all the above trails. Therefore, by the global MP of a Bayesian network:

$$X_i \perp\!\!\!\perp \mathbf{X}_{nd(i)\setminus pa(i)} \mid \mathbf{X}_{pa(i)}$$

   which directly implies the local MP.

ii) Local MP $\Longrightarrow$ Factorization
   Using the chain rule and the specific topological order of the d-dimensional Bayesian network, the density f of $\mathbf{X}$ can be expressed as

$$f(\mathbf{x}) = f_1(x_1) \cdot f_{2|1}(x_2 \mid x_1) \cdot \cdots \cdot f_{d|1\ldots d-1}(x_d \mid x_1, \ldots, x_{d-1}). \tag{2.4}$$

   Define $L := \{1, \ldots, k-1\}$ for $k = 2, \ldots, d$. The function $f_{k|L}$ represents the conditional pdf of X_k given $\mathbf{X}_L$. By the topological order of the DAG, it is clear that $pa(k) \subseteq L \subseteq nd(k)$. According to the local MP, $X_k \perp\!\!\!\perp \mathbf{X}_{nd(k)} \mid \mathbf{X}_{pa(k)}$, which implies $X_k \perp\!\!\!\perp \mathbf{X}_L \mid \mathbf{X}_{pa(k)}$. Therefore, it holds that $f_{k|L}(x_k \mid \mathbf{x}_L) = f_{k|pa(k)}(x_k \mid \mathbf{x}_{pa(k)})$. Substituting this expression back into Equation (2.4) demonstrates that the factorization holds.

□

In fact, it even holds the following equivalence:

Theorem 2.2.15. *Let $\mathcal{G}$ be a DAG and $\mathcal{P}$ be a probability distribution with a density with respect to a product measure. It holds that*

$$\textit{Global MP} \Longleftrightarrow \textit{Local MP} \Longleftrightarrow \textit{Factorization.}$$

To complete the proof, it remains to show that factorization implies the global MP, as detailed in Lauritzen (1996, p. 51).

To understand Bayesian networks better, it is important to explore the concept of Markov equivalence, which identifies networks with the same patterns of independence.

Definition 2.2.16 (Markov equivalence)**.** Two DAGs, $\mathcal{G}_1 = (\mathcal{V}, \mathcal{E}_1)$ and $\mathcal{G}_2 = (\mathcal{V}, \mathcal{E}_2)$, are called **Markov equivalent** if they represent the same independence structure. This means that for every three disjoint subsets $A, B, C \subseteq \mathcal{V}$, $\mathcal{G}_1$ and $\mathcal{G}_2$ satisfy the same d-separation criteria:

$$< A \mid C \mid B >_{\mathcal{G}_1} \quad \Longleftrightarrow \quad < A \mid C \mid B >_{\mathcal{G}_2} .$$

The set of all DAGs that are Markov equivalent to each other is known as **Markov equivalence class**.

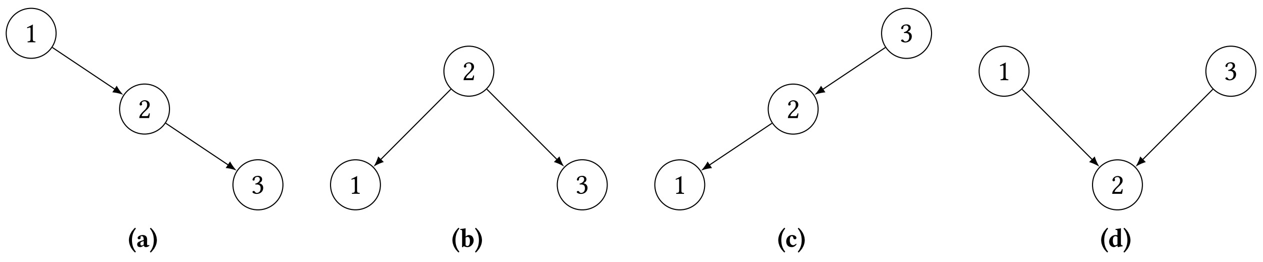


Figure 2.3 Comparison of various structures found in Bayesian networks

Example 2.2.3. Consider now the four different Bayesian networks in Figure 2.3, directly taken from Verma and Pearl (1990). Using the factorization (2.3), it holds that

(a) $f(\mathbf{x}) = f_1(x_1) \cdot f_{2|1}(x_2 \mid x_1) \cdot f_{3|2}(x_3 \mid x_2)$.

(b) $f(\mathbf{x}) = f_2(x_2) \cdot f_{1|2}(x_1 \mid x_2) \cdot f_{3|2}(x_3 \mid x_2)$.

(c) $f(\mathbf{x}) = f_3(x_3) \cdot f_{2|3}(x_2 \mid x_3) \cdot f_{1|2}(x_1 \mid x_2)$.

(d) $f(\mathbf{x}) = f_1(x_1) \cdot f_3(x_3) \cdot f_{2|13}(x_2 \mid x_1, x_3)$.

It is straightforward to see that models (a) to (c) are equivalent regarding the independence information encoded in the different DAGs. In all these DAGs, nodes 1 and 3 are d-separated given 2, hence $X_1 \perp\!\!\!\perp X_3 \mid X_2$ holds. However, Bayesian network (d) is different from the others as it represents the relationship $X_1 \perp\!\!\!\perp X_3$. In other words, only the Bayesian networks (a) to (c) belong to the same Markov equivalence class.

In Example 2.2.3 it becomes clear that v-structures play an important role when it comes to determining whether two Bayesian networks are Markov equivalent. The following result is due to Verma and Pearl (1990):

Lemma 2.2.17. *Two DAGs are Markov equivalent if and only if they have the same skeleton and v-structures.*

The Markov equivalence class of a DAG can be visualized by a graph that shares the same skeleton as all DAGs in the equivalence class. In this graph, edges are directed only if they have consistent directions across every DAG in the equivalence class. Such a graph is known as a **completed partially directed acyclic graph (CPDAG)**. Directed edges in a CPDAG either belong to v-structures or would introduce additional v-structures or cycles if their direction were altered. Later, when algorithms are introduced to recover the structure of a Bayesian network by testing for conditional independencies, it is important to note that such algorithms cannot determine the underlying DAG uniquely, but only the corresponding equivalence class.

2.2.3 Gaussian Bayesian Networks

So far, the specific distribution of the random variables in a Bayesian network has not been of particular interest. The following subsection will therefore focus on the Gaussian distribution. For further details, readers are referred to Koller and Friedman (2009). Note that the univariate and multivariate normal distribution, as well as any other continuous distribution that may appear in this thesis, are defined in Appendix A.

Definition 2.2.18 (Gaussian Bayesian network)**.** A **Gaussian Bayesian network (GBN)** is a Bayesian network where all conditional pdfs are linear Gaussians. More specifically, it holds that

$$X_i \mid \mathbf{X}_{pa(i)} \sim N\big(\beta_0 + \boldsymbol{\beta}^\top \mathbf{X}_{pa(i)}, \sigma_i^2\big),$$

for all $i = 1, \dots, d$, where β_0, $\boldsymbol{\beta}$ and σ_i are constants.

This special class of Bayesian networks is essentially another representation for multivariate Gaussian distributions, as will be demonstrated now.

Theorem 2.2.19. *If $X_i \mid \mathbf{X}_{pa(i)} \sim N\big(\beta_0 + \boldsymbol{\beta}^\top \mathbf{X}_{pa(i)}, \sigma_i^2\big)$, for $i = 1, \dots, d$, and if $\mathbf{X}_{pa(i)} \sim N\big(\boldsymbol{\mu}, \Sigma\big)$, then the following properties hold:*

i) X_i is normally distributed with mean $\mu_{X_i} = \beta_0 + \boldsymbol{\beta}^\top \boldsymbol{\mu}$ and variance $\sigma_{X_i}^2 = \sigma_i^2 + \boldsymbol{\beta}^\top \Sigma \boldsymbol{\beta}$.

ii) The joint distribution of $\{\mathbf{X}_{pa(i)}, X_i\}$ is also normal, with covariance given by

$$Cov(X_j, X_i) = \sum_{k=1}^{|pa(i)|} \beta_k \Sigma_{j,k}, \quad j = 1, \dots, |pa(i)|.$$

Proof. Only property (ii) will be proven. Consider $j \in pa(i)$ for some $i \in \{1, \dots, d\}$. Note that, by property (i), X_i can be written as a linear combination of its parents:

$$X_i = \beta_0 + \sum_{k=1}^{|pa(i)|} \beta_k X_k + \epsilon_i,$$

where $\epsilon_i \sim N(0, \sigma_i^2)$ is independent of $\mathbf{X}_{\mathbf{pa(i)}}$. Then the covariance between X_j and X_i is given by the following:

$$\begin{aligned} \mathrm{Cov}(X_j, X_i) &= \mathrm{Cov}\Big(X_j, \beta_0 + \sum_{k=1}^{|pa(i)|} \beta_k X_k + \epsilon\Big) \\ &\overset{X_j \perp\!\!\!\perp \epsilon_i}{=} \mathrm{Cov}\Big(X_j, \sum_{k=1}^{|pa(i)|} \beta_k X_k\Big) \\ &= \sum_{k=1}^{|pa(i)|} \beta_k \mathrm{Cov}(X_j, X_k) \\ &= \sum_{k=1}^{|pa(i)|} \beta_k \Sigma_{j,k}. \end{aligned}$$

□

The theorem establishes well-known properties of the normal distribution applied to the conditional distributions in a Bayesian network. By induction, it follows that a GBN defines a joint distribution that is multivariate Gaussian. It is important to recall that the converse is also true: the normal distribution is closed under conditioning.

Theorem 2.2.20. *Let* $\{\mathbf{X}, Y\}$, $\mathbf{X} \in \mathbb{R}^n$, $Y \in \mathbb{R}$, *have a joint normal distribution given by the mean vector*

$$\boldsymbol{\mu} = \begin{bmatrix} \boldsymbol{\mu}_X \\ \mu_Y \end{bmatrix} \quad \textit{and covariance matrix} \quad \Sigma = \begin{bmatrix} \Sigma_{XX} & \Sigma_{XY} \\ \Sigma_{YX} & \Sigma_{YY} \end{bmatrix},$$

then the conditional density of $Y \mid \mathbf{X}$ *is normal with mean*

$$\mu = \beta_0 + \boldsymbol{\beta}^\top \mathbf{X}, \; \textit{where } \beta_0 = \mu_Y - \Sigma_{YX}{\Sigma_{XX}}^{-1}\boldsymbol{\mu}_X \textit{ and } \boldsymbol{\beta} = {\Sigma_{XX}}^{-1}\Sigma_{YX},$$

and variance

$$\sigma^2 = \Sigma_{YY} - \Sigma_{YX}{\Sigma_{XX}}^{-1}\Sigma_{XY}.$$

Theorems (2.2.19) and (2.2.20) establish the interchangeability between a GBN and a joint Gaussian distribution, offering a concise representation. Specifically, they illustrate that a GBN can be constructed from multivariate Gaussian data by establishing a topological order, defining parent-child relationships for each variable, and computing the corresponding conditional distributions.

The next results are particularly beneficial when conducting independence tests in constraint-based structure learning, especially within the context of Gaussian distributions.

Definition 2.2.21 (Partial correlation)**.** Let $\mathbf{X} = (X_1, \dots, X_d)^\top$ be a d-dimensional random vector. Further, for distinct $i, j \in \{1, \dots, d\}$, define $\mathrm{S} := \{1, \dots, d\} \setminus \{i, j\}$ and $\mathbf{X}_{\tilde{\mathrm{S}}} := (1, {\mathbf{X}_\mathrm{S}}^\top)^\top$. The $(d-1)$-dimensional partial regression coefficients $\boldsymbol{\beta}_i^*$ and $\boldsymbol{\beta}_j^*$ are given by

$$\boldsymbol{\beta}_i^* = (\beta_{i,0}^*, \dots, \beta_{i,d-2}^*)^\top = \arg\min_{\boldsymbol{\beta}} \mathbb{E}\Big(\big(X_i - \mathbf{X}_{\tilde{\mathrm{S}}}^\top \boldsymbol{\beta}\big)^2\Big),$$

$$\boldsymbol{\beta}_j^* = (\beta_{j,0}^*, \dots, \beta_{j,d-2}^*)^\top = \arg\min_{\boldsymbol{\beta}} \mathbb{E}\Big(\big(X_j - \mathbf{X}_{\tilde{\mathrm{S}}}^\top \boldsymbol{\beta}\big)^2\Big).$$

The corresponding residuals of the linear regression are then given by

$$R_i = X_i - \mathbf{X}_{\tilde{\mathrm{S}}}^\top \boldsymbol{\beta}_{\mathrm{i}}^*,$$

and

$$R_j = X_j - \mathbf{X}_{\tilde{\mathrm{S}}}^\top \boldsymbol{\beta}_{\mathrm{j}}^*.$$

Finally, the **partial correlation** of (X_i, X_j) given $\mathbf{X}_\mathrm{S}$ is defined as the ordinary correlation of the residuals R_i and R_j, i.e.,

$$\rho_{i,j;\mathrm{S}} := \frac{\mathrm{Cov}(R_i, R_j)}{\sqrt{\mathrm{Var}(R_i)}\sqrt{\mathrm{Var}(R_j)}}.$$

Definition 2.2.21 is taken from Whittaker (1990). It establishes the connection between the partial correlation and linear regression. In simpler terms, $\rho_{ij;K}$ measures the dependence between two random variables X_i and X_j after removing the effect of $\mathbf{X}_K$, where $K \subseteq \{1, \dots, d\} \setminus \{i, j\}$. This can be efficiently computed using the following recursion, introduced by Yule (1917):

$$\rho_{ij;K} = \frac{\rho_{ij;K\setminus\{k_0\}} - \rho_{ik_0;K\setminus\{k_0\}}\rho_{k_0j;K\setminus\{k_0\}}}{\sqrt{1 - \rho^2_{ik_0;K\setminus\{k_0\}}}\sqrt{1 - \rho^2_{k_0j;K\setminus\{k_0\}}}}, \tag{2.5}$$

where $k_0 \in K$ and the initial values are given by the ordinary correlation coefficients ρ_{ij}, ρ_{ik_0}, and ρ_{k_0j}.

Definition 2.2.22 (Conditional correlation)**.** Let $X_1, \dots, X_d$ be random variables. The **conditional correlation** $\rho_{ij|K}$ measures the dependence between two random variables X_i and X_j after conditioning on the remaining variables $\mathbf{X}_K$, where $K \subseteq \{1, \dots, d\} \setminus \{i, j\}$. It is defined as

$$\rho_{ij|K} = \frac{\mathbb{E}(X_iX_j \mid \mathbf{X}_K) - \mathbb{E}(X_i \mid \mathbf{X}_K)\mathbb{E}(X_j \mid \mathbf{X}_K)}{\sqrt{\mathrm{Var}(X_i \mid \mathbf{X}_K)}\sqrt{\mathrm{Var}(X_j \mid \mathbf{X}_K)}}.$$

Baba et al. (2004) demonstrate that partial and conditional correlation do not generally coincide. However, there exists a certain class of distributions, including elliptical distributions, where they do align. They further show that the multivariate normal distribution is the only known distribution within the family of elliptical distributions where a partial correlation of zero indicates conditional independence. Consequently, in the special case of Gaussianity, it is reasonable to identify conditional independencies by testing for vanishing partial correlations.

2.2.4 Structure Learning

In practice, Bayesian networks are often constructed using an i.i.d. sample from a multivariate distribution. This process, known as learning, primarily involves two tasks: *structure learning* and *parameter learning*. The main objective of structure learning is to identify a directed acyclic graph (DAG) that best represents the independencies within the distribution $\mathcal{P}$ of the data. There are various algorithms designed for this purpose, which can be broadly categorized as *constraint-based*, *score-based*, or *hybrid* algorithms. This thesis focuses on constraint-based algorithms.

Constraint-based algorithms aim to identify the Markov equivalence class of a Bayesian network's structure by performing conditional independence tests. These algorithms rely on the faithfulness assumption, which posits that conditional independence and d-separation are equivalent. This assumption ensures that the conditional independence tests accurately reflect the underlying causal structure. Specifically, it assumes that all conditional independencies are captured by the Bayesian network's structure. Although this assumption might seem strong, Pearl (1988) demonstrated that for any DAG, there exists a distribution that is faithful to it. Additionally, Meek (1995) and Spirtes et al. (1993) showed that, in a measure-theoretic sense, almost all multinomial and normal distributions are faithful. The latter authors even suggest that faithfulness can only be violated, if at all, by a very specific choice of functional dependency between the random variables.

IC Algorithm

The *IC (Inductive Causation) algorithm*, introduced by Verma and Pearl (1990), serves as a foundational framework for various subsequent constraint-based algorithms. The associated pseudo-code is given in Algorithm 1. In the following, it is assumed that all conditional independence information among all variables is available, referred to as the *oracle version* of an algorithm. However, in practice, the oracle is replaced by statistical tests for conditional independence. Algorithms that use statistical testing based on i.i.d. observations will be referred to as the *empirical version*.

In Step 1 of Algorithm 1, the skeleton is identified by performing a series of conditional independence queries. Recall that two nodes can only be adjacent in the DAG if the two variables they represent are not independent given any other variables. If $\mathbf{X}$ is d-dimensional, there are $\binom{d}{2} = \frac{d(d-1)}{2}$ pairs of variables. Additionally, there are 2^{d-2} possible subsets of the remaining $d-2$ variables, including the empty set. Thus, a total of $N = d(d-1) \cdot 2^{d-1}$ independence queries are performed in this step.

Step 2 of the IC algorithm identifies the v-structures. As shown in Figure 2.3, a v-structure is unique because it is the only structure where two non-adjacent nodes are not conditionally independent given a common neighbor.

Step 3 concludes the algorithm by orienting all edges that can be oriented without introducing cycles or additional v-structures. After this step, the CPDAG representing the Markov equivalence class is fully identified (see Lemma 2.2.17).

As previously noted, the IC algorithm conducts an exponential search in the first step. In the worst case, this cannot be avoided to produce reliable results, as two variables can be conditionally dependent on some set but independent on a superset or subset of that set. Consequently, it is necessary to consider all 2^{d-2} subsets in the worst case. Additionally, in the empirical version of the algorithm, it is required to test higher-order conditional independencies up to order $d-2$. Determining higher-order conditional independencies is less reliable than determining lower-order conditional independencies (Spirtes et al. 1993). For these reasons, a practical implementation of the IC algorithm seems infeasible.

Algorithm 1 IC Algorithm (Inductive Causation), Oracle Version

```
1:  **Input:** $d$-dimensional set of variables $\mathbf{X}$, conditional independence information among all variables.
2:  **Output:** CPDAG $[\mathcal{G}]$
3:  **Step 1:** Skeleton identification
4:  Initialize an undirected graph $\mathcal{G}$ where each node represents a variable in $\mathbf{X}$ and each pair of nodes is
    connected by an edge.
5:  for each pair of variables $X_i$ and $X_j$, $i \neq j \in \{1, \ldots, d\}$ do
6:      if there exists a set $S \subseteq \{1, \ldots, d\} \backslash \{i, j\}$ including the empty set such that $X_i$ and $X_j$ are conditionally
        independent given $\mathbf{X}_S$ then
7:          Remove the edge between $i$ and $j$ in $\mathcal{G}$
8:      end if
9:  end for
10: **Step 2:** v-structure identification
11: for each pair of non-adjacent variables $X_i$ and $X_j$ with a common neighbor $X_k$, $i \neq j \neq k \in \{1, \ldots, d\}$
    do
12:     if $k \notin S$ from Step 1 then
13:         Orient $i \longrightarrow k \longleftarrow j$
14:     end if
15: end for
16: **Step 3:** Orientation of remaining edges
17: for each pair of variables $X_i$ and $X_j$ do
18:     Apply recursively the orientation rules:
19:     if $i$ is adjacent to $j$ and there is a path containing only directed edges from $i$ to $j$ then
20:         Orient $i \longrightarrow j$
21:     end if
22:     if $i$ is not adjacent to $j$ but it exists a $k$ such that $i \longrightarrow k$ and $k - j$ then
23:         Orient $k \longrightarrow j$
24:     end if
25: end for
26: Set $[\mathcal{G}] = \mathcal{G}$ and return the CPDAG $[\mathcal{G}]$.
```

The PC Algorithm: Skeleton Estimation and Complexity

A more practical application of the ideas presented by the IC algorithm was developed by Spirtes et al. (1993) and is known as the *PC algorithm*. As noted by Spirtes et al. (1993), the sample version of their algorithm is sensitive to the order in which variables are given. An order-independent variant called *PC-stable* was subsequently introduced by Colombo and Maathuis (2014). It is important to note that while the PC-stable algorithm achieves order independence in its initial step of skeleton estimation, as detailed later, subsequent steps still rely on variable ordering. Thus, the empirical version of the PC-stable algorithm, despite its name, does not achieve complete stability across all aspects. **To avoid confusion, the PC-stable algorithm will simply be referred to as the PC algorithm hereafter, acknowledging that its skeleton estimation is based on an order-independent approach.**

Algorithm 2 Step 1 of PC Algorithm, Oracle Version

```
1: Input: $d$-dimensional set of variables $\mathbf{X}$, conditional independence information among all variables, ordering $order(\mathbf{X})$ on the variables.
2: Output: Skeleton $\mathcal{G}^{skel}$, Separation sets $sepset$
3: Initialize $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ to the complete undirected graph on the set of variables $\mathbf{X}$.
4: Let $l = -1$
5: repeat
6:   Let $l = l + 1$
7:   for each node $i \in \mathcal{V}$ do
8:     Let $a(i) = adj(i)$, where the adjacencies are with respect to the current graph $\mathcal{G}$ at this stage.
9:   end for
10:  repeat
11:    Select a (new) ordered pair of adjacent nodes $(i, j)$ satisfying $|a(i) \setminus \{j\}| \geq l$, using $order(\mathbf{X})$;
12:    repeat
13:      Select a (new) $\mathrm{S} \subseteq a(i) \setminus \{j\}$ with $|\mathrm{S}| = l$, using $order(\mathbf{X})$
14:      if $X_i \perp\!\!\!\perp X_j \mid \mathbf{X}_\mathrm{S}$ then
15:        Remove the edge $i - j$ from $\mathcal{E}$
16:        Set $sepset(i, j) = sepset(j, i) = \mathrm{S}$
17:      end if
18:    until $(i, j)$ are no longer adjacent in $\mathcal{G}$ or all $\mathrm{S} \subseteq a(i) \setminus \{j\}$ with $|\mathrm{S}| = l$ have been considered
19:  until all pairs of adjacent nodes $(i, j)$ in $\mathcal{G}$ with $|a(i) \setminus \{j\}| \geq l$ have been considered
20: until all pairs of adjacent nodes $(i, j)$ in $\mathcal{G}$ satisfy $|a(i) \setminus \{j\}| \leq l$
21: Let $\mathcal{G}^{skel} = \mathcal{G}$
22: return $\mathcal{G}^{skel}$, $sepset$
```

The order-independent procedure, according to Colombo and Maathuis (2014), for finding a skeleton using known independence relations is detailed in Algorithm 2. It operates as follows:

$l = 0$: All pairs of nodes are queried for marginal independence. If $X_i \perp\!\!\!\perp X_j$ holds, the edge $i - j$ is removed, and $\text{sepset}(i, j) = \text{sepset}(j, i) = \emptyset$.

$l = 1$: Pairs of adjacent nodes (i, j) are considered. The algorithm checks if $X_i \perp\!\!\!\perp X_j \mid \mathbf{X}_\mathrm{S}$ holds for subsets S of size $l = 1$ from $a(i) \setminus \{j\}$. If such a subset is found, the edge $i - j$ is removed, and $\text{sepset}(i, j) = \text{sepset}(j, i) = \mathrm{S}$. Once all ordered pairs of adjacent nodes have been examined and conditional independencies for all subsets of size $l = 1$ have been queried, l is incremented by one.

$l > 1$: The algorithm continues similarly to the case of $l = 1$ until all adjacency sets in the DAG are smaller than l.

Example 2.2.4 (Skeleton Estimation using the PC-stable algorithm)**.** Consider the six dimensional example DAG $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ in Figure 2.4, which will be utilized throughout this thesis.

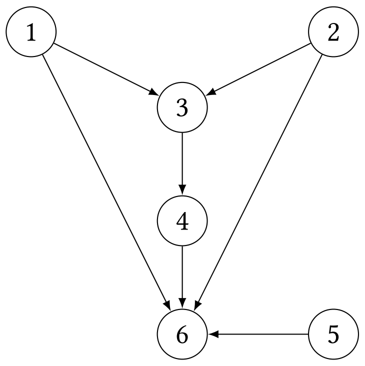


Figure 2.4 A six dimensional DAG

Node $i \in \mathcal{V}$	**pa(i)**	**adj(i)**
1	$\emptyset$	$\{3, 6\}$
2	$\emptyset$	$\{3, 6\}$
3	$\{1, 2\}$	$\{1, 2, 4\}$
4	$\{3\}$	$\{3, 6\}$
5	$\emptyset$	$\{6\}$
6	$\{1, 2, 4, 5\}$	$\{1, 2, 4, 5\}$

Table 2.2 Parents and adjacency sets of the DAG in Figure 2.4

Assume that the distribution of $\mathbf{X}$ is faithful to the DAG in Figure 2.4, and that all conditional independencies are known. They can be determined using the R-package `dagitty` (Textor et al. 2023) and the function within the package `impliedConditionalIndependencies` with the setting `type="all"`. In total, there are 45 implied conditional independencies, which are specified as follows:

i) $X_1 \perp\!\!\!\perp X_2 \mid \mathbf{X}_S$, where $S \in \{\emptyset, \{5\}\}$.

ii) $X_1 \perp\!\!\!\perp X_4 \mid \mathbf{X}_S$, where $S \in \{\{3\}, \{2, 3\}, \{3, 5\}, \{2, 3, 5\}\}$.

iii) $X_1 \perp\!\!\!\perp X_5 \mid \mathbf{X}_S$, where $S \in \{\emptyset, \{2\}, \{3\}, \{4\}, \{2, 3\}, \{2, 4\}, \{3, 4\}, \{2, 3, 4\}\}$.

iv) $X_2 \perp\!\!\!\perp X_4 \mid \mathbf{X}_S$, where $S \in \{\{3\}, \{1, 3\}, \{3, 5\}, \{1, 3, 5\}\}$.

v) $X_2 \perp\!\!\!\perp X_5 \mid \mathbf{X}_S$, where $S \in \{\emptyset, \{1\}, \{3\}, \{4\}, \{1, 3\}, \{1, 4\}, \{3, 4\}, \{1, 3, 4\}\}$.

vi) $X_3 \perp\!\!\!\perp X_5 \mid \mathbf{X}_S$, where $S \in \{\emptyset, \{1\}, \{2\}, \{4\}, \{1, 2\}, \{1, 4\}, \{2, 4\}, \{1, 2, 4\}, \{1, 2, 4, 6\}\}$.

vii) $X_3 \perp\!\!\!\perp X_6 \mid \mathbf{X}_S$, where $S \in \{\{1, 2, 4\}, \{1, 2, 4, 5\}\}$.

viii) $X_4 \perp\!\!\!\perp X_5 \mid \mathbf{X}_S$, where $S \in \{\emptyset, \{1\}, \{2\}, \{3\}, \{1, 2\}, \{1, 3\}, \{2, 3\}, \{1, 2, 3\}\}$.

Now, with the order of the variables set as $order(\mathbf{X}) = (1, 2, \ldots, 6)$, Algorithm 2 is applied to estimate the skeleton.

0.) **Initialization:**

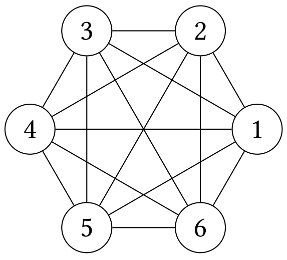


Figure 2.5 Complete undirected graph in six dimensions $\mathcal{G}_0 = (\mathcal{V}, \mathcal{E}_0)$

Node $i \in \mathcal{V}$	**a(i)**
1	$\{2, 3, 4, 5, 6\}$
2	$\{1, 3, 4, 5, 6\}$
3	$\{1, 2, 4, 5, 6\}$
4	$\{1, 2, 3, 5, 6\}$
5	$\{1, 2, 3, 4, 6\}$
6	$\{1, 2, 3, 4, 5\}$

Table 2.3 Adjacency sets of the graph in Figure 2.5

Figure 2.5 shows the initial complete undirected graph on all six nodes and the corresponding adjacency sets for the first iteration are given in Table 2.3.

1.) $\mathbf{l = 0}$: Set $\mathcal{E} = \mathcal{E}_0$. In this iteration, the condition $|a(i) \setminus \{j\}| \geq l$ holds for all $i \neq j$, ensuring that all pairs are considered. The conditioning set S is always the empty set since $|S| = l$ must be satisfied. Once an edge $i - j$ (where $i < j$) is deleted, the statement $X_j \perp\!\!\!\perp X_i$ is not queried again, but otherwise it is.

- Set $i = 1, j = 2$: $X_1 \perp\!\!\!\perp X_2$ holds $\Rightarrow$ remove $1 - 2$ from $\mathcal{E}$. Define $sepset(1, 2) = sepset(2, 1) = \emptyset$.
- Set $i = 1, j = 3$: $X_1 \perp\!\!\!\perp X_3$ does not hold.
- Set $i = 1, j = 4$: $X_1 \perp\!\!\!\perp X_4$ does not hold.
- Set $i = 1, j = 5$: $X_1 \perp\!\!\!\perp X_5$ holds $\Rightarrow$ remove $1 - 5$ from $\mathcal{E}$. Define $sepset(1, 5) = sepset(5, 1) = \emptyset$.
- Set $i = 1, j = 6$: $X_1 \perp\!\!\!\perp X_6$ does not hold.
- Set $i = 2, j = 3$: $X_2 \perp\!\!\!\perp X_3$ does not hold.
- Set $i = 2, j = 4$: $X_2 \perp\!\!\!\perp X_4$ does not hold.
- Set $i = 2, j = 5$: $X_2 \perp\!\!\!\perp X_5$ holds $\Rightarrow$ remove $2 - 5$ from $\mathcal{E}$. Define $sepset(2, 5) = sepset(5, 2) = \emptyset$.
- Set $i = 2, j = 6$: $X_2 \perp\!\!\!\perp X_6$ does not hold.
- Set $i = 3, j = 1$: $X_3 \perp\!\!\!\perp X_1$ does not hold.
- Set $i = 3, j = 2$: $X_3 \perp\!\!\!\perp X_2$ does not hold.
- Set $i = 3, j = 4$: $X_3 \perp\!\!\!\perp X_4$ does not hold.
- Set $i = 3, j = 5$: $X_3 \perp\!\!\!\perp X_5$ holds $\Rightarrow$ remove $3 - 5$ from $\mathcal{E}$. Define $sepset(3, 5) = sepset(5, 3) = \emptyset$.
- Set $i = 3, j = 6$: $X_3 \perp\!\!\!\perp X_6$ does not hold.
- Set $i = 4, j = 1$: $X_4 \perp\!\!\!\perp X_1$ does not hold.
- Set $i = 4, j = 2$: $X_4 \perp\!\!\!\perp X_2$ does not hold.
- Set $i = 4, j = 3$: $X_4 \perp\!\!\!\perp X_3$ does not hold.
- Set $i = 4, j = 5$: $X_4 \perp\!\!\!\perp X_5$ holds $\Rightarrow$ remove $4 - 5$ from $\mathcal{E}$. Define $sepset(4, 5) = sepset(5, 4) = \emptyset$.
- Set $i = 4, j = 6$: $X_4 \perp\!\!\!\perp X_6$ does not hold.
- Set $i = 5, j = 6$: $X_5 \perp\!\!\!\perp X_6$ does not hold.
- Set $i = 6, j = 1$: $X_6 \perp\!\!\!\perp X_1$ does not hold.
- Set $i = 6, j = 2$: $X_6 \perp\!\!\!\perp X_2$ does not hold.
- Set $i = 6, j = 3$: $X_6 \perp\!\!\!\perp X_3$ does not hold.
- Set $i = 6, j = 4$: $X_6 \perp\!\!\!\perp X_4$ does not hold.
- Set $i = 6, j = 5$: $X_6 \perp\!\!\!\perp X_5$ does not hold.

All ordered pairs have been checked and the output after the first iteration is shown in Figure 2.6 and Table 2.4.

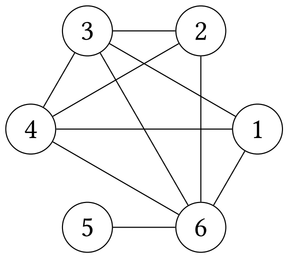


Figure 2.6 $\mathcal{G}_1 = (\mathcal{V}, \mathcal{E}_1)$ after the first iteration of the PC algorithm

Node $i \in \mathcal{V}$	$\mathbf{a(i)}$
1	$\{3, 4, 6\}$
2	$\{3, 4, 6\}$
3	$\{1, 2, 4, 6\}$
4	$\{1, 2, 3, 6\}$
5	$\{6\}$
6	$\{1, 2, 3, 4, 5\}$

Table 2.4 Adjacency sets of the graph in Figure 2.6

2.) $\mathbf{l = 1}$: Set $\mathcal{E} = \mathcal{E}_1$. In this iteration, the condition $|a(i) \setminus \{j\}| \geq l$ holds for all $i \neq j$ with the exception of $i = 5, j = 6$. To shorten notation, some statements will be summarized in the following and the number of statements will always be given in brackets.

- Set $i = 1, j = 3, S = 4$: $X_1 \perp\!\!\!\perp X_3 \mid X_4$ does not hold.

- Set $i = 1, j = 3, S = 6$: $X_1 \perp\!\!\!\perp X_3 \mid X_6$ does not hold.
- Set $i = 1, j = 4, S = 3$: $X_1 \perp\!\!\!\perp X_4 \mid X_3$ holds $\Rightarrow$ remove $1 - 4$ from $\mathcal{E}$. Define $sepset(1,4) = sepset(4,1) = \{3\}$.
- Set $i = 1, j = 6, S = 3$: $X_1 \perp\!\!\!\perp X_6 \mid X_3$ does not hold.
- Set $i = 1, j = 6, S = 4$: $X_1 \perp\!\!\!\perp X_6 \mid X_4$ does not hold.
- Set $i = 2, j = 3, S = 4$: $X_2 \perp\!\!\!\perp X_3 \mid X_4$ does not hold.
- Set $i = 2, j = 3, S = 6$: $X_2 \perp\!\!\!\perp X_3 \mid X_6$ does not hold.
- Set $i = 2, j = 4, S = 3$: $X_2 \perp\!\!\!\perp X_4 \mid X_3$ holds $\Rightarrow$ remove $2 - 4$ from $\mathcal{E}$. Define $sepset(2,4) = sepset(4,2) = \{3\}$.
- Set $i = 2, j = 6, S = 3$: $X_2 \perp\!\!\!\perp X_6 \mid X_3$ does not hold.
- Set $i = 2, j = 6, S = 4$: $X_2 \perp\!\!\!\perp X_6 \mid X_4$ does not hold.
- Set $i = 3, j \in \{1,2,4,6\}, S \in \{1,2,4,6\} \setminus \{j\}$: $X_3 \perp\!\!\!\perp X_j \mid X_S$ does not hold. ($4\binom{3}{1} = 12$ Statements)
- Set $i = 4, j \in \{3,6\}, S \in \{1,2,3,6\} \setminus \{j\}$: $X_4 \perp\!\!\!\perp X_j \mid X_S$ does not hold. ($2\binom{3}{1} = 6$ Statements)
- Set $i = 6, j \in \{1,2,3,4,5\}, S \in \{1,2,3,4,5\} \setminus \{j\}$: $X_6 \perp\!\!\!\perp X_j \mid X_S$ does not hold. ($5\binom{4}{1} = 20$ Statements)

All ordered pairs have been checked and the output after the second iteration is shown in Figure 2.7 and Table 2.5.

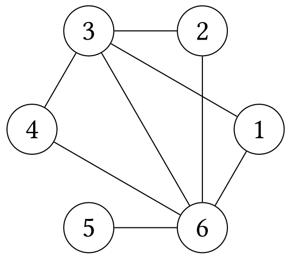


Figure 2.7 $\mathcal{G}_2 = (\mathcal{V}, \mathcal{E}_2)$ after the second iteration of the PC algorithm

Node $i \in \mathcal{V}$	**a(i)**
1	$\{3,6\}$
2	$\{3,6\}$
3	$\{1,2,4,6\}$
4	$\{3,6\}$
5	$\{6\}$
6	$\{1,2,3,4,5\}$

Table 2.5 Adjacency sets of the graph in Figure 2.7

3.) **l = 2** : Set $\mathcal{E} = \mathcal{E}_2$. In this iteration, the condition $|a(i) \setminus \{j\}| \geq l$ holds only for $i \in \{3,6\}$ and $j \in a(i)$.

- Set $i = 3, j \in \{1,2,4,6\}, S \in \{M \subseteq \{1,2,4,6\} \setminus \{j\} \mid |M| = 2\}$: $X_6 \perp\!\!\!\perp X_j \mid \mathbf{X_S}$ does not hold. ($4\binom{3}{2} = 12$ Statements)
- Set $i = 6, j \in \{1,2,3,4,5\}, S \in \{M \subseteq \{1,2,3,4,5\} \setminus \{j\} \mid |M| = 2\}$: $X_6 \perp\!\!\!\perp X_j \mid \mathbf{X_S}$ does not hold. ($5\binom{4}{2} = 30$ Statements)

During the third iteration, although 42 statements were queried, no edges were removed, and the graph along with its adjacency sets remain unchanged from those depicted in Figure 2.7 and Table 2.5 ($\mathcal{G}_3 = \mathcal{G}_2$).

4.) **l = 3** : In this iteration, the condition $|a(i) \setminus \{j\}| \geq l$ holds again only for $i \in \{3,6\}$ and $j \in a(i)$.

- Set $i = 3, j = 1, S = \{2,4,6\}$: $X_3 \perp\!\!\!\perp X_1 \mid X_2, X_4, X_6$ does not hold.
- Set $i = 3, j = 2, S = \{1,4,6\}$: $X_3 \perp\!\!\!\perp X_2 \mid X_1, X_4, X_6$ does not hold.
- Set $i = 3, j = 4, S = \{1,2,6\}$: $X_3 \perp\!\!\!\perp X_4 \mid X_1, X_2, X_6$ does not hold.
- Set $i = 3, j = 6, S = \{1,2,4\}$: $X_3 \perp\!\!\!\perp X_6 \mid X_1, X_2, X_4$ holds $\Rightarrow$ remove $3 - 6$ from $\mathcal{E}$. Define $sepset(3,6) = sepset(6,3) = \{1,2,4\}$
- Set $i = 6, j \in \{1,2,4,5\}, S \in \{M \subseteq \{1,2,3,4,5\} \setminus \{j\} \mid |M| = 3\}$: $X_6 \perp\!\!\!\perp X_j \mid \mathbf{X_S}$ does not hold. ($4\binom{4}{3} = 16$ Statements)

All ordered pairs have been checked and the output after the fourth iteration is shown in Figure 2.8 and Table 2.6.

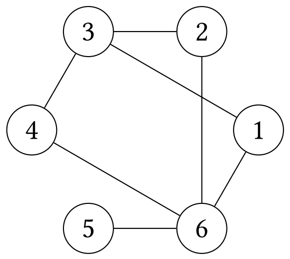


Figure 2.8 $\mathcal{G}_4 = (\mathcal{V}, \mathcal{E}_4)$ after the fourth iteration of the PC algorithm

Node $i \in \mathcal{V}$	$\mathbf{a(i)}$
1	$\{3, 6\}$
2	$\{3, 6\}$
3	$\{1, 2, 4\}$
4	$\{3, 6\}$
5	$\{6\}$
6	$\{1, 2, 4, 5\}$

Table 2.6 Adjacency sets of the graph in Figure 2.8

5.) $\mathbf{l = 4}$: Set $\mathcal{E} = \mathcal{E}_4$. In this iteration, $|a(i) \setminus \{j\}| \geq l$ does not hold for any $i \in \mathcal{V}$ and $j \in a(i)$. Therefore, Step 1 of the PC algorithm terminates and returns the skeleton $\mathcal{G}^{skel} = \mathcal{G}_4$ along with the separation sets in *sepset*.

The output in Figure 2.8 of Example 2.2.4 represents the correct skeleton of the underlying DAG depicted in Figure 2.4. Now, using the results from Example 2.2.4 and comparing them to a worst case scenario, the computational complexity of the first step of the PC algorithm will be analyzed. The worst case corresponds to a fully connected DAG, where there is a directed edge between each pair of nodes such that no cycles are formed. One can construct such a DAG by setting a directed edge $i \longrightarrow j$ if and only if $i < j$, for example. In this scenario, the adjacency set of each node is of maximal size $d - 1$. Furthermore, there are no conditional independencies present, and the PC algorithm will query the maximum number of statements.

From Example 2.2.4, it becomes evident that in the skeleton estimation only conditional independencies of order up to $k - 1$ are queried, where k is the maximum size of adjacency sets in the true underlying DAG. In the worst case, $k = d - 1$, and in the example, $k = d - 2 = 4$. Initially, the algorithm considers all ordered pairs of variables and tests them for ordinary independence, reducing the set only if edges are removed. In the worst case, all $2\binom{d}{2}$ statements are queried. In Example 2.2.4, not all $2\binom{6}{2} = 30$ statements for ordinary independence are checked, but only 25, as five edges are removed at this stage. In the second step of the algorithm, conditional independencies of order up to $l = 1$ are checked. In the worst case, there are again $2\binom{d}{2}$ pairs of variables, and since the sets $a(i) \setminus \{j\}$ will always have size $d - 2$, there are $\binom{d-2}{1}$ possibilities to choose one-dimensional subsets as conditioning sets. Therefore, in total, a maximum of $2\binom{d}{2}\binom{d-2}{1}$ statements are queried at this stage. This number decreases if edges were removed in the previous or same iteration of the algorithm. In Example 2.2.4, 48 conditional independencies are queried, compared to a worst-case scenario of 120. In the subsequent stages where $1 < l \leq d - 2$, the maximum number of checks, analogous to $l = 0, 1$, is given by $2\binom{d}{2}\binom{d-2}{l}$. This yields the following result on the computational complexity of the skeleton estimation in the PC algorithm.

Lemma 2.2.23 (Complexity of the first step of the PC algorithm)**.** *In the worst case, the number of independence statements queried by the PC algorithm is bounded by*

$$N^{max} = 2\binom{d}{2}\sum_{l=0}^{d-2}\binom{d-2}{l} = \binom{d}{2} \cdot 2^{d-1} = d \cdot (d-1) \cdot 2^{d-2}.$$

Lemma 2.2.23 shows that the computational requirements of the PC algorithm can grow exponentially with the number of nodes. In contrast to the IC algorithm, Spirtes et al. (1993) argue that this worst case is rarely realized because it requires that no two variables are independent given a set of variables that has cardinality less than k. In most practical cases, the average number of conditional independence queries is expected to be much smaller, allowing the algorithm to recover sparse graphs even in higher dimensions. This is confirmed by Example 2.2.4, where 135 conditional independence statements were queried compared to the worst case of 480 statements for $d = 6$. While conducting a complexity analysis for the average case is challenging, Kalisch and Bühlmann (2007) argue that if the true underlying DAG is sparse, the computational complexity of the PC algorithm reduces to polynomial time.

The PC Algorithm: Edge Orientations

After obtaining the skeleton and separation sets in the first step of the PC algorithm, the subsequent steps focus on identifying v-structures and orienting edges. Algorithm 3 illustrates the second and third steps

Algorithm 3 Step 2 and 3 of the PC algorithm, Oracle Version

```
1: **Input:** Skeleton $\mathcal{G}^{skel}$, Separation sets $sepset$
2: **Output:** CPDAG $[\mathcal{G}]$
3: Let $\mathcal{G} = \mathcal{G}^{skel}$.
4: **Step 2:** v-structure identification
5: **for** each pair of non-adjacent nodes $i$ and $j$ with a common neighbor $k$ **do**
6:     **if** $k \notin \text{sepset}(i, j)$ from Step 1 **then**
7:         Orient $i \longrightarrow k \longleftarrow j$
8:     **end if**
9: **end for**
10: **Step 3:** Orientation of remaining edges
11: Orient as many edges as possible by repeatedly applying one of the following rules:
12: **R1:** Orient $j - k$ into $j \longrightarrow k$ whenever there is an edge $i \longrightarrow j$ such that $i$ and $k$ are non-adjacent.
13: **R2:** Orient $i - j$ into $i \longrightarrow j$ whenever there is a structure $i \longrightarrow k \longrightarrow j$.
14: **R3:** Orient $i - j$ into $i \longrightarrow j$ whenever there are two structures $i - k \longrightarrow j$ and $i - l \longrightarrow j$ such that $k$ and $l$ are non-adjacent.
15: Set $[\mathcal{G}] = \mathcal{G}$
16: **return** $[\mathcal{G}]$.
```

of the PC algorithm. It is worth noting that both steps bear similarity to those found in Algorithm 1.

Example 2.2.5 (Example 2.2.4 continued). Consider now the skeleton and separation sets that were produced in Example 2.2.4.

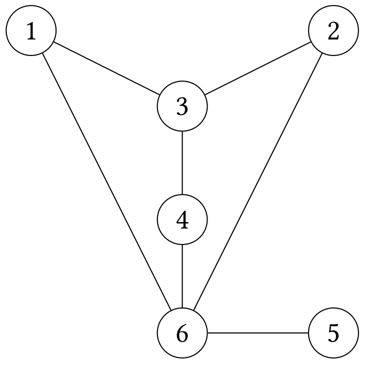


Figure 2.9 The skeleton of the DAG in Figure 2.4

Node $i \in \mathcal{V}$	adj(i)
1	$\{3, 6\}$
2	$\{3, 6\}$
3	$\{1, 2, 4\}$
4	$\{3, 6\}$
5	$\{6\}$
6	$\{1, 2, 4, 5\}$

Table 2.7 Adjacency sets of the skeleton in Figure 2.9

The separation sets are the following:

- $sepset(1, 2) = sepset(2, 1) = \emptyset$.
- $sepset(1, 5) = sepset(5, 1) = \emptyset$.
- $sepset(2, 5) = sepset(5, 2) = \emptyset$.

- $sepset(3,5) = sepset(5,3) = \emptyset$.
- $sepset(4,5) = sepset(5,4) = \emptyset$.
- $sepset(1,4) = sepset(4,1) = \{3\}$.
- $sepset(2,4) = sepset(4,2) = \{3\}$.
- $sepset(3,6) = sepset(6,3) = \{1,2,4\}$.

Step 2: v-structure identification. Common neighbors of two non-adjacent nodes i and j are given by $adj(i) \cap adj(j)$.

- $i = 1, j = 2, sepset(i,j) = \emptyset$ and $adj(i) \cap adj(j) = \{3,6\}$.
  - $k = 3$: $k \notin sepset(i,j) \Rightarrow$ v-structure $1 \longrightarrow 3 \longleftarrow 2$.
  - $k = 6$: $k \notin sepset(i,j) \Rightarrow$ v-structure $1 \longrightarrow 6 \longleftarrow 2$.
- $i = 1, j = 4, sepset(i,j) = \{3\}$ and $adj(i) \cap adj(j) = \{3,6\}$.
  - $k = 3$: $k \in sepset(i,j)$.
  - $k = 6$: $k \notin sepset(i,j) \Rightarrow$ v-structure $1 \longrightarrow 6 \longleftarrow 4$.
- $i = 1, j = 5, sepset(i,j) = \emptyset$ and $adj(i) \cap adj(j) = \{6\}$.
  - $k = 6$: $k \notin sepset(i,j) \Rightarrow$ v-structure $1 \longrightarrow 6 \longleftarrow 5$.
- $i = 2, j = 4, sepset(i,j) = \{3\}$ and $adj(i) \cap adj(j) = \{3,6\}$.
  - $k = 3$: $k \in sepset(i,j)$.
  - $k = 6$: $k \notin sepset(i,j) \Rightarrow$ v-structure $2 \longrightarrow 6 \longleftarrow 4$ (already present).
- $i = 2, j = 5, sepset(i,j) = \emptyset$ and $adj(i) \cap adj(j) = \{6\}$.
  - $k = 6$: $k \notin sepset(i,j) \Rightarrow$ v-structure $2 \longrightarrow 6 \longleftarrow 5$ (already present).
- $i = 3, j = 5, sepset(i,j) = \emptyset$ and $adj(i) \cap adj(j) = \emptyset$.
- $i = 3, j = 6, sepset(i,j) = \{1,2,4\}$ and $adj(i) \cap adj(j) = \{1,2,4\}$.
  - For all $k \in \{1,2,4\}$: $k \in sepset(i,j)$.
- $i = 4, j = 5, sepset(i,j) = \emptyset$ and $adj(i) \cap adj(j) = \{6\}$.
  - $k = 6$: $k \notin sepset(i,j) \Rightarrow$ v-structure $4 \longrightarrow 6 \longleftarrow 5$ (already present).

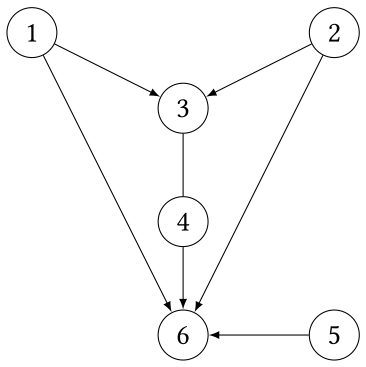


Figure 2.10 Estimated CPDAG of the DAG in Figure 2.4 after step 2 of the PC algorithm

Node $i \in \mathcal{V}$	**pa(i)**	**adj(i)**
1	$\emptyset$	$\{3,6\}$
2	$\emptyset$	$\{3,6\}$
3	$\{1,2\}$	$\{1,2,4\}$
4	$\emptyset$	$\{3,6\}$
5	$\emptyset$	$\{6\}$
6	$\{1,2,4,5\}$	$\{1,2,4,5\}$

Table 2.8 Parents and adjacency sets of the PDAG in Figure 2.10

Figure 2.10 and Table 2.8 show the resulting estimated CPDAG after Step 2 of the PC algorithm.

Step 3: Orientation of the remaining edges. As depicted in Figure 2.10, the only edge that remains undirected is $3 - 4$. Rules R1-R3 are now applied to determine if this edge can be directed.

R1: $j = 3, k = 4$.

- $i = 1$: the edge $i \longrightarrow j$ exists and i and k are non-adjacent $\Rightarrow$ Orient $3 \longrightarrow 4$.

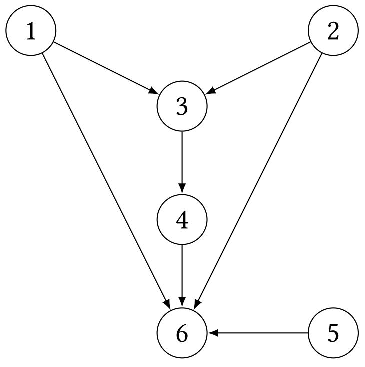


Figure 2.11 Estimated CPDAG of the DAG in Figure 2.4 after Step 3 of the PC algorithm

Node $i \in \mathcal{V}$	**pa(i)**	**adj(i)**
1	$\emptyset$	$\{3, 6\}$
2	$\emptyset$	$\{3, 6\}$
3	$\{1, 2\}$	$\{1, 2, 4\}$
4	$\{3\}$	$\{3, 6\}$
5	$\emptyset$	$\{6\}$
6	$\{1, 2, 4, 5\}$	$\{1, 2, 4, 5\}$

Table 2.9 Parents and adjacency sets of the CPDAG in Figure 2.11

There are no remaining undirected edges in this scenario, indicating the completion of the PC algorithm. It returns the estimated CPDAG $[\mathcal{G}]$, pictured in Figure 2.11 and detailed in Table 2.9.

The output in Example 2.2.5 corresponds to the true CPDAG of the underlying DAG in Figure 2.4. This is evident because changing any edge directions in the CPDAG or leaving some edges undirected would lead to additional v-structures. Therefore, in this specific case, the CPDAG precisely matches the underlying DAG. In other words, there are no other DAGs within the Markov equivalence class.

The skeleton estimation and edge orientation in Examples 2.2.4 and 2.2.5 yielded correct results, assuming faithfulness and complete information about all independence relations. The following general result was mentioned in Colombo and Maathuis (2014) and formally proven in Spirtes et al. (1993).

Theorem 2.2.24. *Let the distribution $\mathcal{P}$ of $\mathbf{X}$ be faithful to a DAG $\mathcal{G} = (\mathcal{V}, \mathcal{E})$. Further, assume that all conditional independence relations are known. Then the output of the PC algorithm is the CPDAG that represents the Markov equivalence class of $\mathcal{G}$.*

In this context, the oracle version of the PC algorithm can be considered sound and complete. However, as mentioned earlier, true independence relations among variables are seldom known in practice. Therefore, in general, the empirical version of the PC algorithm cannot guarantee a correct estimation of the equivalence class corresponding to the true underlying distribution of observed values. A more detailed discussion will follow now.

Stability of the PC Algorithm

Consider the empirical version of the PC algorithm, where the oracle is replaced by a test for conditional independence. Specifically, the null hypothesis $H_0 : X_i \perp\!\!\!\perp X_j \mid \mathbf{X}_S$ is tested against the alternative given by $H_1 : X_i \not\!\perp\!\!\!\perp X_j \mid \mathbf{X}_S$ for distinct $i, j \in \{1, \ldots, d\}$ and $S \subseteq \{1, \ldots, d\} \setminus \{i, j\}$. Let $T_\alpha(i, j, S) \in \{H_0, H_1\}$ denote the test decision at significance level α. A common example for such a test in the continuous case is *Fisher's Z-test*, which will be defined in Section 2.2.5. The test decision primarily influences the skeleton estimation, as depicted in Algorithm 2. If $T_\alpha(i, j, S) = H_0$, the edge $i - j$ is removed from the undirected graph, and the separation sets are defined by $sepset(i, j) = sepset(j, i) = S$.

A Type I error in the test decision can lead to falsely including edges in the skeleton, whereas a Type II error can result in falsely removing edges. The following example is directly taken from Spirtes et al. (1993) and shows how erroneously deleting an edge can lead to mistakenly including another edge.

Example 2.2.6. Consider the true underlying DAG depicted in Figure 2.12.

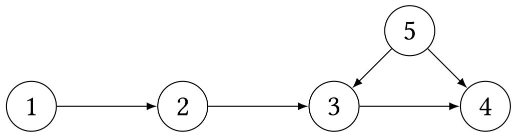


Figure 2.12 Example DAG in five dimensions

The only conditional independencies of order zero are given by $X_1 \perp\!\!\!\perp X_5$ and $X_2 \perp\!\!\!\perp X_5$. Suppose these are correctly identified by the test, but the test incorrectly concludes that $T_\alpha(4, 5, \emptyset) = H_0$, i.e., $X_4 \perp\!\!\!\perp X_5$. As a result, the edges $1-5$, $2-5$, and $4-5$ are removed at stage $l = 0$. In subsequent stages, node 5 is no longer in the adjacency sets of node 1, 2, and 4. If the test procedure correctly identifies all other independencies and dependencies, the edge $2-4$ will not be removed at any stage because it holds that $X_2 \not\perp\!\!\!\perp X_4 \mid \mathbf{X}_\mathbf{S}$ for any $\mathbf{S}$ that does not contain node 5. Since node 5 is no longer in the adjacency sets, the algorithm will not conclude that the edge $2-4$ should be removed. Hence, an error in removing $4-5$ leads to an error in keeping $2-4$.

Errors in the skeleton estimation can directly impact subsequent edge orientations. Incorrectly including or excluding edges in the skeleton can result in missing or newly appearing v-structures. Type I and Type II errors may cause instabilities in the second step of the PC algorithm, even if the skeleton is correct initially. This can occur due to deviations from the true separation sets. Additionally, sampling errors or hidden variables may lead to conflicting information about edge directions.

For example, suppose the algorithm determines that $1-2-3$ and $2-3-4$ should be v-structures. Consequently, the edge direction of $2-3$ becomes ambiguous. In this thesis, such conflicts are resolved by overwriting the edge direction according to the most recent information. However, this introduces order-dependence in the v-structure estimation. Separating sets can also be order-dependent, varying based on the order in which conditional independencies are checked. This directly affects the v-structure estimation since separating sets determine v-structures. Moreover, orientation rules might depend on the order of variables and errors made in independence tests.

As already mentioned, Algorithm 2 shows an order-independent version of the skeleton estimation. This is achieved by lines 7 to 9, which ensure adjacency sets are saved before each new iteration (i.e., before incrementing l). The original version does not contain these lines and updates adjacency sets immediately after each edge removal, influencing other decisions within the same iteration. Colombo and Maathuis (2014) also developed order-independent versions for the other steps of the PC algorithm. However, these versions are impractical for this thesis's application as they could result in *non-extendable CPDAGs* by marking some edges as ambiguous in their direction. A non-extendable CPDAG is a PDAG, where directing some of the undirected edges would introduce cycles or additional v-structures. Later, when fitting different Bayesian networks to simulated data using copula-based regression, complete parent-child relationships are needed, requiring extendable CPDAGs.

2.2.5 Conditional Independence Tests

Conditional independence tests are crucial for estimating the skeleton and v-structures in the empirical version of the PC algorithm. Stability analysis shows that accurately estimating the Markov equivalence class of the true underlying DAG necessitates a highly precise method for testing conditional independencies. For continuous data, the most commonly used statistical test relies on the classical asymptotic result of the sample correlation, introduced by Fisher (1915) and known as *Fisher's Z-transformation.*

Definition 2.2.25 (Fisher's Z-transformation)**.** Let $(\mathbf{x}, \mathbf{y})$ be a n-dimensional sample drawn from the bivariate random vector (X, Y). The sample correlation coefficient between X and Y is computed as

$$r := r(\mathbf{x}, \mathbf{y}) = \frac{\sum_{i=1}^{n}(x_i - \bar{x})(y_i - \bar{y})}{\sqrt{\sum_{i=1}^{n}(x_i - \bar{x})^2}\sqrt{\sum_{i=1}^{n}(y_i - \bar{y})^2}},$$

where $\bar{x} = \frac{1}{n}\sum_{i=1}^n x_i$ denotes the sample mean of X and $\bar{y} = \frac{1}{n}\sum_{i=1}^n y_i$ denotes the sample mean of Y. **Fisher's Z-transformation** is then defined as

$$z = \frac{1}{2}\ln\left(\frac{1+r}{1-r}\right) = \operatorname{artanh}(r). \tag{2.6}$$

Theorem 2.2.26 (Asymptotic distribution of Fisher's Z-transformation)**.** *Consider an i.i.d. sequence of random variables $\{(X_i, Y_i)\}_{i=1,\dots,n}$, where each (X_i, Y_i) follows a bivariate normal distribution with correlation given by $\operatorname{Cor}(X_i, Y_i) = \rho$ for all $i = 1, \dots, n$. Further, let the (random) sample correlation be given by*

$$R_n := \frac{\sum_{i=1}^n (X_i - \bar{X})(Y_i - \bar{Y})}{\sqrt{\sum_{i=1}^n (X_i - \bar{X})^2}\sqrt{\sum_{i=1}^n (Y_i - \bar{Y})^2}},$$

where $\bar{X} = \frac{1}{n}\sum_{i=1}^n X_i$ and $\bar{Y} = \frac{1}{n}\sum_{i=1}^n Y_i$. Consequently, a random version of Fisher's Z-transformation is given by

$$Z_n = \frac{1}{2}\ln\left(\frac{1+R_n}{1-R_n}\right) = \operatorname{artanh}(R_n).$$

It follows that the sequence of random variables $\{Z_n\}_{n\in\mathbb{N}}$ converges in distribution to a normally distributed random variable:

$$Z_n \xrightarrow{d} N\left(\frac{1}{2}\ln\left(\frac{1+\rho}{1-\rho}\right), \frac{1}{n-3}\right) \quad as \quad n \longrightarrow \infty. \tag{2.7}$$

In particular, if $\rho = 0$, it holds that

$$\sqrt{n-3}\cdot Z_n \xrightarrow{d} N(0,1) \quad as \quad n \longrightarrow \infty.$$

A derivation of the results presented in Theorem 2.2.26 can be found in Anderson (2003). Equation (2.7) demonstrates that Fisher's Z-transformation stabilizes the variance of the correlation coefficient. Further, the asymptotic normality facilitates statistical inference.

To apply Fisher's Z-transformation in the context of structure learning, the following proposition offers an alternative definition of the partial correlation, which is commonly used for estimating partial correlations in sample data.

Proposition 2.2.27. *Let $\mathbf{X} = (X_1, \dots, X_d)^\top$ be a d-dimensional random vector with positive definite covariance matrix $\Sigma = (\sigma_{kl})_{k,l=1,\dots,d}$. Further, let the concentration matrix be given by $P = \Sigma^{-1} = (p_{kl})_{k,l=1,\dots,d}$, and define the set $\mathrm{S} = \{1, \dots, d\} \setminus \{i, j\}$. Then the partial correlation of (X_i, X_j) given $\mathbf{X}_\mathrm{S}$ is given by*

$$\rho_{ij;\mathrm{S}} = -\frac{p_{ij}}{\sqrt{p_{ii}p_{jj}}}.$$

Proof. Using Definition 2.2.21, it holds that

$$\rho_{ij;\mathrm{S}} = \frac{\operatorname{Cov}(R_i, R_j)}{\sqrt{\operatorname{Var}(R_i)}\sqrt{\operatorname{Var}(R_j)}},$$

where

$$R_i = X_i - \mathbf{X}_{\tilde{\mathrm{S}}}^\top \boldsymbol{\beta}_\mathrm{i}^*$$
$$R_j = X_j - \mathbf{X}_{\tilde{\mathrm{S}}}^\top \boldsymbol{\beta}_\mathrm{j}^*,$$

with $\boldsymbol{\beta}_i^*$ (resp. $\boldsymbol{\beta}_j^*$) being the least squares estimator of the linear regression of X_i (resp. X_j) on $\mathbf{X}_\mathrm{S}$, and $\mathbf{X}_{\tilde{\mathrm{S}}}$ being the random vector $\mathbf{X}_\mathrm{S}$ augmented by 1 to allow for a constant term in the regression. After reordering the variables, the covariance matrix of $(X_i, X_j, \mathbf{X}_{\tilde{\mathrm{S}}}^\top)^\top$ is given by

$$\Sigma = \begin{bmatrix} \Sigma_{ii} & \Sigma_{ij} & \Sigma_{i\tilde{\mathrm{S}}} \\ \Sigma_{ji} & \Sigma_{jj} & \Sigma_{j\tilde{\mathrm{S}}} \\ \Sigma_{\tilde{\mathrm{S}}i} & \Sigma_{\tilde{\mathrm{S}}j} & \Sigma_{\tilde{\mathrm{S}}\tilde{\mathrm{S}}} \end{bmatrix} = \begin{bmatrix} C_{11} & C_{12} \\ C_{21} & C_{22} \end{bmatrix},$$

where

$$C_{11} = \begin{bmatrix} \Sigma_{ii} & \Sigma_{ij} \\ \Sigma_{ji} & \Sigma_{jj} \end{bmatrix}, \quad C_{12} = \begin{bmatrix} \Sigma_{i\tilde{S}} \\ \Sigma_{j\tilde{S}} \end{bmatrix}, \quad C_{21} = \begin{bmatrix} \Sigma_{\tilde{S}i} & \Sigma_{\tilde{S}j} \end{bmatrix}, \quad C_{22} = \Sigma_{\tilde{S}\tilde{S}}.$$

Further, by the standard results on the ordinary least square estimator, the residuals are given by

$$R_i = X_i - \mathbf{X}_{\tilde{S}}^\top \boldsymbol{\beta}_i^* = \mathrm{X_i} - \mathbf{X}_{\tilde{S}}^\top (\Sigma_{\tilde{S}\tilde{S}})^{-1}\Sigma_{\tilde{S}i},$$

$$R_j = X_j - \mathbf{X}_{\tilde{S}}^\top \boldsymbol{\beta}_j^* = \mathrm{X_j} - \mathbf{X}_{\tilde{S}}^\top (\Sigma_{\tilde{S}\tilde{S}})^{-1}\Sigma_{\tilde{S}j}.$$

Assume w.l.o.g. that $\mathbf{X}$ is centered. Using that $\mathbb{E}(R_i) = \mathbb{E}(R_j) = 0$, and $\mathbb{E}(\mathbf{X}) = \mathbf{0}$, the covariance of the residuals can be expressed as

$$\mathrm{Cov}(R_i, R_j) = \mathbb{E}(R_i R_j) = \cdots = \Sigma_{ij} - \Sigma_{i\tilde{S}}(\Sigma_{\tilde{S}\tilde{S}})^{-1}\Sigma_{\tilde{S}j}. \tag{2.8}$$

Finally, let the concentration matrix be denoted by

$$\Sigma^{-1} = P = \begin{bmatrix} P_{ii} & P_{ij} & P_{i\tilde{S}} \\ P_{ji} & P_{jj} & P_{j\tilde{S}} \\ P_{\tilde{S}i} & P_{\tilde{S}j} & P_{\tilde{S}\tilde{S}} \end{bmatrix} = \begin{bmatrix} K_{11} & K_{12} \\ K_{21} & K_{22} \end{bmatrix},$$

where

$$K_{11} = \begin{bmatrix} P_{ii} & P_{ij} \\ P_{ji} & P_{jj} \end{bmatrix}.$$

Using Schur's formula for matrix inversion and Equation (2.8), the inverse of K_{11} is given by

$$K_{11}^{-1} = C_{11} - C_{12}C_{22}^{-1}C_{21} = \begin{bmatrix} \Sigma_{ii} & \Sigma_{ij} \\ \Sigma_{ji} & \Sigma_{jj} \end{bmatrix} - \begin{bmatrix} \Sigma_{i\tilde{S}} \\ \Sigma_{j\tilde{S}} \end{bmatrix} (\Sigma_{\tilde{S}\tilde{S}})^{-1} \begin{bmatrix} \Sigma_{\tilde{S}i} & \Sigma_{\tilde{S}j} \end{bmatrix} = \begin{bmatrix} \mathrm{Cov}(R_i, R_i) & \mathrm{Cov}(R_i, R_j) \\ \mathrm{Cov}(R_j, R_i) & \mathrm{Cov}(R_j, R_j) \end{bmatrix}.$$

Further,

$$K_{11}^{-1} = \frac{1}{\det(K_{11})} \begin{bmatrix} [K_{11}]_{22} & -[K_{11}]_{12} \\ -[K_{11}]_{21} & [K_{11}]_{11} \end{bmatrix} = \frac{1}{\det(K_{11})} \begin{bmatrix} P_{jj} & -P_{ij} \\ -P_{ji} & P_{ii} \end{bmatrix}.$$

Lastly, combining the results yields

$$\rho_{ij;\mathrm{S}} = \frac{\mathrm{Cov}(R_i, R_j)}{\sqrt{\mathrm{Var}(R_i)}\sqrt{\mathrm{Var}(R_j)}} = \frac{-\frac{1}{\det(K_{11})}P_{ij}}{\sqrt{\frac{1}{\det(K_{11})}P_{jj}}\sqrt{\frac{1}{\det(K_{11})}P_{ii}}} = -\frac{P_{ij}}{\sqrt{P_{ii}}\sqrt{P_{jj}}}.$$

□

The proof is taken from Lauritzen (1996) and modified to handle non-Gaussian distributions. Proposition 2.2.27 provides an additional method (besides linear regression and the recursive formula) for computing partial correlations by inverting the covariance matrix. This approach is particularly computationally efficient in practice, especially when the number of conditioning variables is large, as it requires only a single matrix inversion. In the R-package `pcalg` (Kalisch et al. 2024), **sample partial correlations** are derived using the recursive formula (see Equation (2.5)) with ordinary sample correlations as initial values if the conditioning set contains only one variable. For larger conditioning sets, the matrix inversion method (see Proposition 2.2.27) is applied to the sample correlation matrix to compute partial correlations.

Theorem 2.2.28 (Distribution of the sample partial correlation). *Let* $\mathbf{X} = (X_1, \dots, X_d)^\top \sim N(\boldsymbol{\mu}_X, \Sigma_X)$ *be a random vector that follows a multivariate normal distribution. For* $i, j \in \{1, \dots, d\}$ *and* $\mathrm{S} \subseteq \{1, \dots, d\} \setminus \{i, j\}$, *let the (random) sample partial correlation of* X_i *and* X_j *after removing the effect of* $\mathbf{X}_\mathrm{S}$, *based on a sample size of* n, *be denoted by* $R_{ij;\mathrm{S}}(\mathbf{X}; n)$.

Further, let the true partial correlation coefficient be denoted by $\rho = \rho_{ij;\mathrm{S}}$. *Suppose additionally* $\mathrm{Y} = (Y_1, Y_2)^\top \sim N(\boldsymbol{\mu}_Y, \Sigma_Y)$ *follows a bivariate normal distribution with correlation* ρ. *Then, for the (random) sample correlation of* Y_1 *and* Y_2 *based on* $n - |\mathrm{S}|$ *observations, denoted by* $R_{12}(\mathrm{Y}; n - |\mathrm{S}|)$, *it follows that it has the same distribution as* $R_{ij;\mathrm{S}}(\mathbf{X}; n)$ *independently of which observations are removed:*

$$R_{ij;\mathrm{S}}(\mathbf{X}; n) \overset{d}{=} R_{12}(\mathrm{Y}; n - |\mathrm{S}|). \tag{2.9}$$

The result in Theorem 2.2.28 was first derived by Fisher (1924) and suggests that the asymptotic behavior of Fisher's Z-transformation is similar for both partial and ordinary correlations. A detailed derivation can be found in Anderson (2003).

Corollary 2.2.29 (Asymptotic distribution of Fisher's Z-transformation of the partial correlation)**.** *Let* $\{\mathbf{X}^{(i)}\}_{i=1,\ldots,n} = \{(X_1^{(i)}, \ldots, X_d^{(i)})^\top\}_{i=1,\ldots,n}$ *be an i.i.d. sequence of random variables, where each* $\mathbf{X}^{(i)}$ *follows a multivariate normal distribution with partial correlations* $\rho_{ij;\mathrm{S}}$*, for all* $i, j \in \{1, \ldots, d\}$ *and* $\mathrm{S} \subseteq \{1, \ldots, d\} \setminus \{i, j\}$*. Let the (random) sample partial correlations based on n observations be denoted by* $R_{i,j;\mathrm{S}}(\mathbf{X}; n)$*. Consequently, a random version of Fisher's Z-transformation of the partial correlation is given by*

$$Z_n^{par} = \frac{1}{2}\ln\left(\frac{1 + R_{i,j;\mathrm{S}}(\mathbf{X}; n)}{1 - R_{i,j;\mathrm{S}}(\mathbf{X}; n)}\right) = \operatorname{artanh}(R_{i,j;\mathrm{S}}(\mathbf{X}; n)).$$

It then follows that the sequence of random variables $\{Z_n^{par}\}_{n\in\mathbb{N}}$ *converges in distribution to a normally distributed random variable:*

$$Z_n^{par} \xrightarrow{d} N\left(\frac{1}{2}\ln\left(\frac{1 + \rho_{ij;\mathrm{S}}}{1 - \rho_{ij;\mathrm{S}}}\right), \frac{1}{n - |\mathrm{S}| - 3}\right) \quad \textit{as} \quad n \longrightarrow \infty.$$

In particular, if $\rho_{ij;\mathrm{S}} = 0$*, it holds that*

$$\sqrt{n - |\mathrm{S}| - 3} \cdot Z_n^{par} \xrightarrow{d} N(0, 1) \quad \textit{as} \quad n \longrightarrow \infty. \tag{2.10}$$

Clearly, Equation (2.10) is useful in the context of testing a hypothesis of the form $H_0 : \rho_{ij;\mathrm{S}} = 0$ against its two sided alternative for some i, j, and S.

Definition 2.2.30 (Fisher's Z-test of the partial correlation)**.** Consider a continuous d-dimensional random vector $\mathbf{X} = (X_1, \ldots, X_d)^\top$ with a n-dimensional realization $\mathbf{x}$. Let $i, j \in \{1, \ldots, d\}$ and $\mathrm{S} \subseteq \{1, \ldots, d\} \setminus \{i, j\}$. Denote by $\rho_{ij;\mathrm{S}}$ the true underlying partial correlation of X_i, X_j given $\mathbf{X}_\mathrm{S}$, and by $r_{ij;\mathrm{S}}$ the sample partial correlation based on the sample $\mathbf{x}$. Further, let Fisher's Z-transformation of the sample partial correlation be given by

$$z^{par} = \frac{1}{2}\ln\left(\frac{1 + r_{ij;\mathrm{S}}}{1 - r_{ij;\mathrm{S}}}\right).$$

Consider now the hypotheses

$$H_0 : \rho_{ij;\mathrm{S}} = 0 \quad \text{vs.} \quad H_1 : \rho_{ij;\mathrm{S}} \neq 0.$$

The null hypothesis can be tested against its alternative using the following test, called **Fisher's Z-test**:
Reject H_0 vs. H_1 at level α if and only if

$$\sqrt{n - |\mathrm{S}| - 3} \cdot z^{par} > \Phi^{-1}(1 - \alpha/2),$$

where $|\mathrm{S}| \leq d - 2$ is the cardinality of the set S, Φ^{-1} denotes the quantile function of the standard normal distribution, α is the significance level, and n is the number of observations.

As mentioned earlier in Section 2.2.3, in the special case where $\mathbf{X}$ follows a multivariate normal distribution, the partial correlation $\rho_{ij;\mathrm{S}}$ is equal to zero if and only if the conditional correlation $\rho_{ij|\mathrm{S}}$ is equal to zero. Furthermore, a conditional correlation of zero is equivalent to conditional independence in the case of the Gaussian distribution. Therefore, in this specific case, or when the data is at least approximately normal distributed, Fisher's Z-Test for vanishing partial correlations can be used as a conditional independence test.

While the Z-test application within the PC algorithm is highly computationally efficient, relying on Gaussian assumptions poses a significant drawback. When the dependence structure in the data deviates strongly from normality, it is anticipated that the test may yield misleading results. In subsequent sections of this thesis, a novel and more flexible copula-based method for conditional independence testing will be introduced and compared against the Z-test within the PC algorithm. This comparison will involve both Gaussian and non-Gaussian simulation setups, highlighting the method's robustness across different data distributions.

2.2.6 PDAG Extension

In practice, once the CPDAG is obtained using the empirical version of the PC algorithm, it often needs to be extended to fully fit a Bayesian network to the given data. Typically, undirected edges in the CPDAG can be oriented as long as they do not create new v-structures or cycles.

Algorithm 4 Extension of a Partially Directed Acyclic Graph (PDAG)

```
1:  Input: PDAG $\mathcal{G}$
2:  Output: Extended PDAG $\mathcal{G}'$
3:  Initialize $\mathcal{G}' := \mathcal{G}$, $\mathcal{A} := \mathcal{G}$
4:  while $\mathcal{A}$ is not empty do
5:      Select a node $i$ which satisfies the following properties in subgraph $\mathcal{A}$:
            a)  No edge $(i, j)$ in $\mathcal{A}$ is directed outward from $i$.
            b)  For every node $j$ adjacent to $i$ with $i - j$ undirected in $\mathcal{A}$, $j$ is adjacent to all other nodes
                adjacent to $i$.
6:      if such $i$ is not found then
7:          Stop and return a negative answer (graph $\mathcal{G}$ does not admit any extension).
8:      end if
9:      Direct all edges incident to $i$ (meaning all edges containing the node $i$) in $\mathcal{A}$ toward $i$ in $\mathcal{G}'$.
10:     $\mathcal{A} := \mathcal{A} - i$ (remove $i$ and all edges incident to $i$ from $\mathcal{A}$).
11: end while
12: return $\mathcal{G}'$ (an extension of the input PDAG $\mathcal{G}$)
```

Algorithm 4, introduced by Dor and Tarsi (1992), aims to extend a PDAG to a DAG while preserving the PDAG's structure and avoiding the creation of new v-structures, provided the PDAG admits an extension. The fundamental idea of the algorithm is that every DAG contains a *sink*, i.e., a node with no outward-directed edges. Property a) in line 5 aims to identify a sink. Property b) ensures that directing edges towards the sink does not create a new v-structure. Thus, both properties are necessary for the existence of an extension. For a more detailed justification of the recursion, refer to Dor and Tarsi (1992). Lastly, it is worth noting that the algorithm runs in polynomial time, as the number of iterations equals the number of nodes, and each edge is examined at most twice.

2.3 Vine Copulas

Copulas are mathematical functions used to describe the dependency structure between random variables, allowing for the separation of marginal distributions from their joint behavior. Originating from *Sklar's Theorem*, copulas enable precise modeling of complex, non-linear dependencies and tail behaviors in multivariate data. This thesis explores the theoretical foundations of copulas, their various types, and their applications in pair-copula Bayesian networks. The primary source for this section's content is Czado (2019), with additional insights drawn from Joe (2014), Kurowicka (2006), and Kurowicka and Joe (2011), among others.

2.3.1 Sklar's Theorem

Definition 2.3.1 (Copula, copula density)**.** A d-dimensional **copula** C is a multivariate distribution function $C : [0,1]^d \to [0,1]$ with uniformly distributed marginals. In the case of absolute continuity, the **copula density** is given by

$$c(u_1, \dots, u_d) = \frac{\partial^d}{\partial u_1 \dots \partial u_d} C(u_1, \dots, u_d),$$

for all $\mathbf{u} \in [0,1]^d$.

A frequently used transformation in the context of copulas is given by the following:

Theorem 2.3.2 (Probability Integral Transform). *Let X be a continuous random variable with cdf F_X. Then the random variable $U = F_X(X)$ follows a standard uniform distribution. Conversely, let U follow a standard uniform distribution. Then the random variable $F_X^{-1}(U)$, where F_X^{-1} is the quantile function of X, has the same distribution as X.*

The following foundational result in the theory of copulas, first proven by Sklar (1959), provides the key link between multivariate distribution functions, their marginal distributions, and dependence structure.

Theorem 2.3.3 (Sklar's Theorem). *Let $\mathbf{X}$ be a d-dimensional random vector with joint distribution function F and marginal distribution functions $F_1, \dots, F_d$. Then there exists a d-dimensional copula C with density c such that for all $\mathbf{x} \in \mathbb{R}^d$,*

$$F(x_1, \dots, x_d) = C(F_1(x_1), \dots, F_d(x_d)). \tag{2.11}$$

Further, the joint density function of $\mathbf{X}$ can be expressed as

$$f(x_1, \dots, x_d) = c(F_1(x_1), \dots, F_d(x_d)) \cdot f_1(x_1) \cdots f_d(x_d). \tag{2.12}$$

If the marginal distributions are absolutely continuous, then C is unique.

Proof. In the case of absolute continuity, consider the probability integral transform (PIT) $U_j = F_j(X_j)$, for $j = 1, \dots, d$. Clearly, the copula $C(u_1, \dots, u_d)$ represents the joint distribution function of $\mathbf{U} = (U_1, \dots, U_d)$. □

Theorem 2.3.3 can be employed to independently model marginal distributions and the dependence structure. For instance, given an i.i.d. sample $\mathbf{x}_i = (x_{i1}, \dots, x_{id})^\top$, for $i = 1, \dots, n$, the *pseudo-copula data* $u_{ij} = \hat{F}_j(x_{ij})$, where $\hat{F}_j$ is an estimate for the marginal distribution of j, for $j = 1, \dots, d$, can be utilized to model the copula C. Later in this thesis, an estimate $\hat{F}_j$ for the j-th margin is obtained via *kernel density estimation*. Further, by applying Equations (2.11) and (2.12) to arbitrary marginal distributions and a chosen copula C, one can construct a new multivariate distribution known as *meta distribution*.

2.3.2 Dependence Measures

A frequently used measure of linear dependence is the *Pearson product-moment correlation*. For two random variables X and Y with finite second moments, it is defined by

$$\rho := \rho(X, Y) := \operatorname{Cor}(X, Y) = \frac{\operatorname{Cov}(X, Y)}{\sqrt{\operatorname{Var}(X)}\sqrt{\operatorname{Var}(Y)}}.$$

In Section 2.2.5, for example, its empirical version was used to define Fisher's Z-transformation. Despite its popularity, however, Pearson's product-moment correlation possesses significant drawbacks. First, it is not defined for distributions with non-finite second moments. Second, and more importantly, it is not invariant under monotone increasing transformations of the margins, and its value might depend on the marginal distributions of X and Y. Hence, in most cases, there is no direct relationship between the bivariate copula and Pearson's correlation.

An alternative measure is provided by *Kendall's tau*, denoted simply by τ. It is invariant under monotone transformations of the marginals and can be expressed solely in terms of the associated copula, as its value remains independent of the marginal distributions.

Definition 2.3.4 (Kendall's τ). The **Kendall's τ** between two continuous random variables X and Y is defined as the difference between the probability of concordance and the probability of discordance:

$$\tau(X, Y) = \mathbb{P}((X_1 - X_2)(Y_1 - Y_2) > 0) - \mathbb{P}((X_1 - X_2)(Y_1 - Y_2) < 0),$$

where (X_1, Y_1) and (X_2, Y_2) are i.i.d. copies of (X, Y).

Assume now that an i.i.d. sample of (X, Y) is given by (x_i, y_i), $i = 1, \dots, n$. In total, there are $\frac{n(n-1)}{2}$ unordered pairs (x_i, y_i), (x_j, y_j), where $i, j = 1, \dots, n$ and $i \neq j$. A pair is called *concordant* if $(x_i - x_j)(y_i - y_j) > 0$ and *discordant* if $(x_i - x_j)(y_i - y_j) < 0$. Note that in the continuous case, $(x_i - x_j)(y_i - y_j) \neq 0$ almost surely ($\mathbb{P}$-a.s.), i.e., ties occur with probability zero.

Definition 2.3.5 (Estimate of Kendall's τ)**.** Let N_c be the number of concordant pairs and N_d be the number of discordant pairs from a sample (x_i, y_i), $i = 1, \dots, n$, from the joint distribution of (X, Y). An **estimate of Kendall's** τ is then given by

$$\hat{\tau}_n := \frac{2(N_c - N_d)}{n(n-1)}.$$

The number of concordant and discordant pairs can be computed using only the ranks of the observations. Thus, Kendall's τ is a *rank-based dependence measure* that does not rely on the marginal distributions.

Theorem 2.3.6 (Kendall's τ in terms of the copula)**.** *Let (X_1, X_2) be continuous random variables with associated copula C. Then Kendall's τ can be expressed as*

$$\tau = 4 \int_{[0,1]^2} C(u_1, u_2) \mathrm{d}C(u_1, u_2) - 1.$$

A proof of Theorem 2.3.6 can be found in Czado (2019). While Kendall's τ measures the overall strength and direction of the dependence structure, it does not explicitly provide insights into the behavior of the joint distribution in the tails. Specifically, the strength of dependence may increase in the tails, and furthermore, it may not be symmetric between its upper and lower tail.

Definition 2.3.7 (Tail dependence)**.** Let (X, Y) be random variables following a bivariate distribution with copula C. The **upper tail dependence coefficient** of (X, Y) is given by

$$\lambda^u = \lim_{t \to 1^-} \mathbb{P}(Y > F_Y^{-1}(t) \mid X > F_X^{-1}(t)) = \lim_{t \to 1^-} \frac{1 - 2t + C(t,t)}{1-t},$$

where F_X is the marginal distribution of X, and F_Y is the marginal distribution of Y. The **lower tail dependence coefficient** of (X, Y) is defined as

$$\lambda^l = \lim_{t \to 0^+} \mathbb{P}(Y \leq F_Y^{-1}(t) \mid X \leq F_X^{-1}(t)) = \lim_{t \to 0^+} \frac{C(t,t)}{t}.$$

2.3.3 Bivariate Copula Classes

Throughout this thesis, two classes of bivariate copulas will be considered: *elliptical* and *Archimedean copulas*. Elliptical copulas arise from applying the probability integral transform to the margins of well-known multivariate distributions such as the Gaussian or Student's t-distribution. Archimedean copulas are constructed using generator functions. A special subclass of the Archimedean copulas is the BB copulas, which blend the properties from multiple simpler copulas and are characterized by two parameters. All copulas used in this thesis are defined in Appendix B. This section will provide basic properties of these copulas, with a detailed overview available in Joe (2014).

Table 2.10 demonstrates that for many copula families, simple expressions for Kendall's τ in terms of the copula parameters exist. However, not all copula families can directly model negative dependence, i.e., negative values of τ. One possible solution to expand the range of dependence is through rotations.

Definition 2.3.8 (Rotated copulas)**.** A **rotated copula** is constructed by a counterclockwise rotation of the underlying copula density $c(\cdot, \cdot)$:

i) 90°: $c_{90}(u_1, u_2) := c(1 - u_2, u_1)$.

ii) 180°: $c_{180}(u_1, u_2) := c(1 - u_1, 1 - u_2)$.

iii) 270°: $c_{270}(u_1, u_2) := c(u_2, 1 - u_1)$.

Family	**Kendall's τ**	**Range of τ**
Independence	$\tau = 0$	0
Gaussian	$\tau = \frac{2}{\pi}\arcsin(\rho)$	$[-1, 1]$
t	$\tau = \frac{2}{\pi}\arcsin(\rho)$	$[-1, 1]$
Clayton	$\tau = \frac{\delta}{\delta+2}$	$[0, 1]$
Gumbel	$\tau = 1 - \frac{1}{\delta}$	$[0, 1]$
Frank	$\tau = 1 - \frac{4}{\delta}\left(1 - \frac{1}{\delta}\int_0^\delta \frac{t}{e^t-1}\,dt\right)$	$[-1, 1]$
Joe	$\tau = 1 + \frac{2}{2-\delta}(\psi(2) - \psi(\frac{2}{\delta}+1))$, where ψ denotes the digamma function	$[0, 1]$
BB1	$\tau = 1 - \frac{2}{\delta(\theta+2)}$	$[0, 1]$
BB6	No simple closed-form formula	$[0, 1]$
BB7	$\tau = 1 - \frac{2}{\delta(2-\theta)} + \frac{4}{\theta^2\delta}B(\frac{2-2\theta}{\theta}+1, \delta+2)$ for $1 < \theta < 2$ and beta function B	$[0, 1]$
BB8	No simple closed-form formula	$[0, 1]$

Table 2.10 Kendall's τ as a function of the copula parameters for different bivariate families

Family	**Upper Tail Dependence λ^u**	**Lower Tail Dependence λ^l**
Independence	0	0
Gaussian	0	0
t	$2t_{\nu+1}\left(-\sqrt{\frac{(\nu+1)(1-\rho)}{1+\rho}}\right)$	$2t_{\nu+1}\left(-\sqrt{\frac{(\nu+1)(1-\rho)}{1+\rho}}\right)$
Clayton	0	$2^{-1/\delta}$
Gumbel	$2 - 2^{1/\delta}$	0
Frank	0	0
Joe	$2 - 2^{1/\delta}$	0
BB1	$2 - 2^{1/\delta}$	$2^{-1/(\delta\theta)}$
BB6	$2 - 2^{1/(\delta\theta)}$	0
BB7	$2 - 2^{1/\theta}$	$2^{-1/\delta}$
BB8	$\begin{cases} 2 - 2^{1/\vartheta}, & \text{if } \delta = 1 \\ 0, & \text{else} \end{cases}$	0

Table 2.11 Upper and lower tail dependence as a function of copula parameters for different bivariate families

Table 2.11 presents upper and lower tail dependence formulas for various copula families, each characterized by a simple closed-form expression. The copulas exhibit significant differences in their tail behavior. For instance, copulas like the Gaussian and Frank copulas exhibit no tail dependence. In contrast, the t-copula displays symmetric tail dependencies, while copulas such as the Clayton or Gumbel copulas feature asymmetric tail dependencies. This demonstrates the versatility of copulas in modeling multivariate distributions compared to traditional approaches like the multivariate Gaussian or Student's t-distribution. Additionally, rotations of copulas not only extend the range of τ but also provide options for modeling tail dependencies. For example, a 180-degree rotation of the Clayton copula shifts tail dependency from the lower to the upper tail.

The dependence structure of a specific bivariate copula can be visualized using *normalized contour plots*. These plots are generated by transforming the marginal distributions to standard normal distributions and adjusting the copula density accordingly. Normalized contour plots for various copula families and parameters are provided in Appendix B. Typically, three different scales are employed throughout this thesis.

Definition 2.3.9 (Variable scales)**.** Consider the following variable scales:

i) **x-scale:** Original scale (X_1, X_2) with density $f(x_1, x_2)$.

ii) **u-scale:** Copula scale (U_1, U_2), where $U_i := F_i(X_i)$, $i = 1, 2$, and copula density $c(u_1, u_2)$.

iii) **z-scale:** Marginal normalized scale (Z_1, Z_2), where $Z_i := \Phi^{-1}(U_i) = \Phi^{-1}(F_i(X_i))$, $i = 1, 2$, and density given by
$$g(z_1, z_2) = c(\Phi(z_1), \Phi(z_2))\phi(z_1)\phi(z_2),$$
where $\Phi(\cdot)$ denotes the cdf of the standard normal distribution, and $\phi(\cdot)$ its pdf.

2.3.4 Regular Vines

Regular vines (R-vines) are a flexible class of multivariate copula models that decompose the dependence structure of high-dimensional data into a cascade of bivariate copulas. The first R-vine was introduced by Joe (1996) and belongs to the subclass of D-vines (drawable vines). The hierarchical approach of R-vines allows for the modeling of complex dependencies using bivariate building blocks, making R-vines particularly useful for high-dimensional applications. To illustrate, consider the following three-dimensional example:

Example 2.3.1. Let X_1, X_2, and X_3 be three random variables with joint density function $f(\cdot)$. The density can be factorized as follows:

$$f(x_1, x_2, x_3) = f_1(x_1) \cdot f_{2|1}(x_2 \mid x_1) \cdot f_{3|12}(x_3 \mid x_1, x_2). \tag{2.13}$$

The factorization is unique up to a reordering of the variables. According to Sklar's Theorem (2.12), it holds for the bivariate distribution of (X_1, X_2):

$$f_{12}(x_1, x_2) = c_{12}(F_1(x_1), F_2(x_2)) \cdot f_1(x_1) \cdot f_2(x_2).$$

Applying (2.12) to the bivariate distribution of (X_1, X_3) given X_2 yields

$$f_{13|2}(x_1, x_3 \mid x_2) = c_{13;2}(F_{1|2}(x_1 \mid x_2), F_{3|2}(x_3 \mid x_2); x_2) \cdot f_{1|2}(x_1 \mid x_2) \cdot f_{3|2}(x_3 \mid x_2),$$

where $c_{13;2}(\cdot, \cdot; x_2)$ denotes the copula associated with the distribution of (X_1, X_3) given X_2. Using

$$f_{2|1}(x_2 \mid x_1) = \frac{f_{12}(x_1, x_2)}{f_1(x_1)} = c_{12}(F_1(x_1), F_2(x_2)) \cdot f_2(x_2),$$

$$f_{3|12}(x_3 \mid x_1, x_2) = \frac{f_{13|2}(x_1, x_3 \mid x_2)}{f_{1|2}(x_1 \mid x_2)} = c_{13;2}(F_{1|2}(x_1 \mid x_2), F_{3|2}(x_3 \mid x_2); x_2) \cdot f_{3|2}(x_3 \mid x_2),$$

and

$$f_{3|2}(x_3 \mid x_2) = \frac{f_{23}(x_2, x_3)}{f_2(x_2)} = c_{23}(F_2(x_2), F_3(x_3)) \cdot f_3(x_3),$$

the three-dimensional joint density $f(\cdot)$ can be expressed in terms of bivariate copulas and conditional distribution functions as follows:

$$f(x_1, x_2, x_3) = c_{13;2}(F_{1|2}(x_1 \mid x_2), F_{3|2}(x_3 \mid x_2); x_2) \cdot c_{23}(F_2(x_2), F_3(x_3)) \cdot c_{12}(F_1(x_1), F_2(x_2)) \cdot \prod_{i=1}^{3} f_i(x_i). \quad (2.14)$$

Equation (2.14) is referred to as a *pair copula decomposition.* It is important to note that in this decomposition, the copula $c_{13;2}(\cdot, \cdot; x_2)$ may depend on the conditioning value x_2. Throughout this thesis, it is assumed that this dependence can be ignored, i.e., $c_{13;2}(\cdot, \cdot; x_2) = c_{13;2}(\cdot, \cdot)$ holds. This assumption is known as the *simplifying assumption*, and under this condition, Equation (2.14) is termed *pair copula construction (PCC).* Since the variables in (2.13) can be reordered, the decomposition (2.14) is not unique. For examples of PCCs in higher dimensions, readers are referred to Aas et al. (2009). The number of possible PCCs grows significantly with increasing dimensions; while there are only three possibilities in the three-dimensional example, there are 240 different ways to construct a five-dimensional density. To organize and visualize different PCCs in higher dimensions, Bedford and Cooke (2002) introduced a graphical model known as *regular vine (R-vine).* Further definitions from graph theory are necessary to fully define this model.

Definition 2.3.10 (Tree)**.** A **tree** is an undirected graph $T = (V, E)$ that is connected, meaning there is a path between every pair of nodes, and acyclic.

Since trees only contain undirected edges, the notation $\{a, b\}$ may be used to represent an edge $a - b$, instead of using (a, b) together with (b, a).

Definition 2.3.11 (R-vine tree sequence)**.** A **R-vine tree sequence** on d elements is a sequence of trees $\mathcal{T} = (T_i)_{i=1,\dots,d-1}$ such that:

i) $T_1 = (N_1, E_1)$ is a tree with $N_1 = \{1, \dots, d\}$.

ii) For $j \geq 2$, $T_j = (N_j, E_j)$ is a tree with $N_j = E_{j-1}$.

iii) Proximity condition: For $j \geq 2$ and $\{a, b\} \in E_j$, it must hold that $|a \cap b| = 1$, meaning a and b share a common node in T_{j-1}.

The objective is to associate each edge with a pair copula. This necessitates the following definition:

Definition 2.3.12 (Complete union and conditioned sets)**.** Let $e = \{a, b\} \in E_i$ be an edge in the i-th tree of a R-vine tree sequence. Define the **complete union** of e as

$$A_e := \left\{ j \in N_1 \mid \exists e_1 \in E_1, \dots, e_{i-1} \in E_{i-1} \text{ such that } j \in e_1 \in \cdots \in e_{i-1} \in e \right\}.$$

The **conditioning set** of e is given by

$$D_e := A_a \cap A_b,$$

and the **conditioned sets** are defined as

$$C_{e,a} := A_a \setminus D_e \text{ and } C_{e,b} := A_b \setminus D_e.$$

The edge $e = (C_{e,a}, C_{e,b}; D_e)$ may be abbreviated as

$$e = (a_e, b_e; D_e),$$

and if $D_e = \emptyset$, simply as $e = (a_e, b_e)$.

The next definition will bridge the gap between purely graphical R-vine tree sequences and their application in constructing multivariate distributions using pair copulas.

Definition 2.3.13 (R-vine distribution)**.** A d-dimensional random vector $\mathbf{X}$ follows a **regular vine distribution** if a triplet $(\mathcal{F}, \mathcal{T}, \mathcal{B})$ can be specified as follows:

i) **Margins:** $\mathcal{F} = (F_1, \dots, F_d)$ is a vector of continuous marginal distributions, where F_i corresponds to the marginal distribution of X_i for $i = 1, \dots, d$.

ii) **R-vine tree sequence:** $\mathcal{T}$ is a R-vine tree sequence on d elements.

iii) **Bivariate copulas:** $\mathcal{B} = \{C_e \mid e \in E_i, i = 1, \dots, d\}$ is a set of symmetric bivariate copulas, where E_i denotes the edge set in tree T_i, $i = 1, \dots, d-1$.

iv) **Link between $\mathcal{T}$ and $\mathcal{B}$:** For each $e_i \in E_i$, $i = 1, \dots, d-1$, C_e corresponds to the copula associated with the conditional distribution of (X_{a_e}, X_{b_e}) given X_{D_e}. The simplifying assumption is assumed to hold, and the copula C_e will be denoted by $C_{a_e,b_e;D_e}$ and its density by $c_{a_e,b_e;D_e}$.

Definition 2.3.13 is restricted to symmetric copulas, i.e., copulas for which $c(u_1, u_2) = c(u_2, u_1)$ holds. Many copulas, especially when considering rotations, are non-symmetric. However, the same principle applies to these types of copulas. In practical applications, special attention needs to be given to the order of (a_e, b_e) in the context of non-symmetric copulas. A regular vine distribution with uniformly distributed margins is referred to as a *regular vine copula.*

Bedford and Cooke (2002) show the existence and uniqueness of a regular vine distribution given a triplet $(\mathcal{F}, \mathcal{T}, \mathcal{B})$.

Theorem 2.3.14 (Existence of a R-vine distribution)**.** *Given a triplet $(\mathcal{F}, \mathcal{T}, \mathcal{B})$ satisfying the conditions stated in Definition 2.3.13, there exists a unique d-dimensional distribution F with density*

$$f(x_1, \dots, x_d) = \Big(\prod_{i=1}^{d} f_i(x_i)\Big)\Big(\prod_{i=1}^{d-1}\prod_{e \in E_i} c_{a_e,b_e;D_e}\big(F_{a_e|D_e}(x_{a_e} \mid \mathbf{x}_{D_e}), F_{b_e|D_e}(x_{b_e} \mid \mathbf{x}_{D_e})\big)\Big). \tag{2.15}$$

Theorem 2.3.14 proves to be very useful as it states that regular vine distributions can be defined by specifying a triplet $(\mathcal{F}, \mathcal{T}, \mathcal{B})$. The resulting joint density can then be computed using Equation (2.15). To evaluate this density, it is essential to determine the univariate conditional distributions $F_{C_{e,a}|D_e}$ and $F_{C_{e,b}|D_e}$. Following the chain rule of differentiation, Joe (1996) provided the following recursion:

Lemma 2.3.15 (Recursion for the conditional distribution functions)**.** *Let $K \subset V_1 = \{1, \dots, d\}$, $v \in V_1 \setminus K$, and $K_{-w} = K \setminus \{w\}$. Then, for all $w \in K$, the conditional distribution of X_v given $\mathbf{X}_K$ is given by*

$$F_{v|K}(x_v \mid \mathbf{x}_K) = \frac{\partial C_{v,w;K_{-w}}\big(F_{v|K_{-w}}(x_v \mid \mathbf{x}_{K_{-w}}), F_{w|K_{-w}}(x_w \mid \mathbf{x}_{K_{-w}})\big)}{\partial F_{w|K_{-w}}(x_w \mid \mathbf{x}_{K_{-w}})}.$$

Using a bottom-up approach, the copulas $C_{v,w;K_{-w}}$ can always be chosen such that they are included in $\mathcal{B}$, thereby eliminating the need for integration. A more concise notation for evaluating conditional distribution functions was introduced in Aas et al. (2009).

Definition 2.3.16 (h-function)**.** The **h-functions** corresponding to a bivariate copula C_{12} are defined as follows:

$$h_{1|2}(u_1, u_2) := \frac{\partial}{\partial u_2} C_{12}(u_1, u_2),$$

$$h_{2|1}(u_2, u_1) := \frac{\partial}{\partial u_1} C_{12}(u_1, u_2),$$

where $(u_1, u_2) \in [0, 1]^2$.

Before presenting an explicit example of a R-vine distribution, consider the following subclass of R-vines that holds particular importance in this thesis:

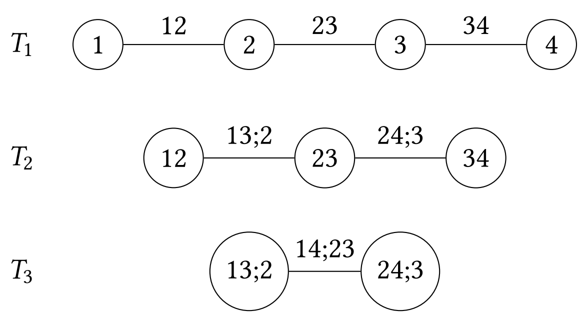


Figure 2.13 D-vine tree sequence on $d = 4$ elements

Definition 2.3.17 (D-vine tree sequence)**.** A regular vine tree sequence on d elements $\mathcal{T} = (T_i)_{i=1,\dots,d-1}$ is termed a **drawable vine (D-vine) tree sequence** if, for each node $n \in N_i$, it holds that

$$|\{e \in E_i \mid n \in e\}| \leq 2.$$

In other words, D-vine tree sequences are a subclass of R-vine tree sequences where each node has at most two neighbors. From the proximity condition of R-vine tree sequences, it follows that the first tree of a D-vine tree sequence completely determines all other trees. A regular vine distribution with a D-vine tree structure will be referred to as a *drawable vine (D-vine) distribution.*

Example 2.3.2 (Four dimensional D-vine distribution)**.** Consider the D-vine tree sequence on $d = 4$ elements in Figure 2.13. The D-vine density corresponding to the tree structure in Figure 2.13 is given by

$$\begin{aligned} f(x_1,\dots,x_4) = {} & c_{14;23}(F_{1|23}(x_1 \mid x_2,x_3), F_{4|23}(x_4 \mid x_2,x_3)) \cdot c_{24;3}(F_{2|3}(x_2 \mid x_3), F_{4|3}(x_4 \mid x_3)) \\ & \cdot c_{13;2}(F_{1|2}(x_1 \mid x_2), F_{3|2}(x_3 \mid x_2)) \cdot c_{34}(F_3(x_3), F_4(x_4)) \cdot c_{23}(F_2(x_2), F_3(x_3)) \\ & \cdot c_{12}(F_1(x_1), F_2(x_2)) \cdot f_4(x_4) \cdot f_3(x_3) \cdot f_2(x_2) \cdot f_1(x_1). \end{aligned} \tag{2.16}$$

Further, the conditional cdfs are given by the recursion

$$F_{1|2}(x_1 \mid x_2) = \frac{\partial C_{12}(F_1(x_1), F_2(x_2))}{\partial F_2(x_2)} =: h_{1|2}(F_1(x_1) \mid F_2(x_2)),$$

$$F_{3|2}(x_3 \mid x_2) = \frac{\partial C_{23}(F_2(x_2), F_3(x_3))}{\partial F_2(x_2)} =: h_{3|2}(F_3(x_3) \mid F_2(x_2)),$$

$$F_{2|3}(x_2 \mid x_3) = \frac{\partial C_{23}(F_2(x_2), F_3(x_3))}{\partial F_3(x_3)} =: h_{2|3}(F_2(x_2) \mid F_3(x_3)),$$

$$F_{4|3}(x_4 \mid x_3) = \frac{\partial C_{34}(F_3(x_3), F_4(x_4))}{\partial F_3(x_3)} =: h_{4|3}(F_4(x_4) \mid F_3(x_3)),$$

$$\begin{aligned} F_{1|23}(x_1 \mid x_2, x_3) &= \frac{\partial C_{13;2}(F_{1|2}(x_1 \mid x_2), F_{3|2}(x_3 \mid x_2))}{F_{3|2}(x_3 \mid x_2)} \\ &=: h_{1|3;2}(h_{1|2}(F_1(x_1) \mid F_2(x_2)) \mid h_{3|2}(F_3(x_3) \mid F_2(x_2))), \end{aligned}$$

$$\begin{aligned} F_{4|23}(x_1 \mid x_2, x_3) &= \frac{\partial C_{24;3}(F_{2|3}(x_2 \mid x_3), F_{4|3}(x_4 \mid x_3))}{F_{2|3}(x_2 \mid x_3)} \\ &=: h_{4|2;3}(h_{4|3}(F_4(x_4) \mid F_3(x_3)) \mid h_{2|3}(F_2(x_2) \mid F_3(x_3))). \end{aligned}$$

2.3.5 Family Selection and Parameter Estimation

The goal of this section is to estimate the families and parameters for a fixed tree structure given an i.i.d. d-dimensional sample of size n from a R-vine copula

$$\mathbf{u} := (\mathbf{u}_1^\top, \dots, \mathbf{u}_n^\top), \text{ where } \mathbf{u}_k := (u_{k,1}, \dots, u_{k,d})^\top \text{ for } k = 1, \dots, n.$$

Equation (2.15), transformed to the u-level, can be readily used to compute the likelihood of the simplified R-vine copula.

Definition 2.3.18 (Likelihood of a simplified R-vine copula)**.** The **likelihood of a simplified R-vine copula** with parameter set $\boldsymbol{\theta} = \{\theta_e, e \in E\}$ and sample $\mathbf{u}$ is given by

$$\ell(\boldsymbol{\theta}; \mathbf{u}) = \prod_{k=1}^{n} \prod_{i=1}^{d-1} \prod_{e \in E_i} c_{a_e,b_e;D_e}\big(C_{a_e|D_e}(u_{k,a_e} \mid \mathbf{u}_{k,D_e}), C_{b_e|D_e}(u_{k,b_e} \mid \mathbf{u}_{k,D_e}); \theta_{a_e,b_e;D_e}\big). \tag{2.17}$$

Note that in Equation (2.17), not only does the pair copula $c_{a_e,b_e;D_e}$ depend on a parameter from $\boldsymbol{\theta}$, but also its arguments $C_{a_e|D_e}$ and $C_{b_e|D_e}$. In fact, they depend on all parameters of the pair copulas needed to recursively define them. To simplify notation, this dependence is omitted in (2.17).

For a d-dimensional vine copula with a fixed tree structure $\mathcal{T}$, there are $d(d-1)/2$ bivariate copula families $\mathcal{B}(\mathcal{T})$ to estimate. Let $\mathcal{B}^{all}$ denote the set of all bivariate copulas that are considered in the estimation process for each edge. Unless stated otherwise, in this thesis, $\mathcal{B}^{all}$ includes all copulas defined in Appendix B, including rotated versions. The parameters estimated are denoted by $\Theta(\mathcal{B}(\mathcal{T}))$, and the number of parameters ranges from 0 (for only independence copulas) to $d(d-1)$ (for only copulas with two parameters each). To estimate the family and parameters for each edge, a sequential approach based on the **Akaike information criterion (AIC)** (Akaike 1998) will be employed. Alternatively, the **Bayesian information criterion (BIC)** (Schwarz 1978) could also be used. The methodology is outlined as follows:

1.) Consider tree $T_1 = (V_1, E_1)$ of the tree structure $\mathcal{T}$. For any edge $e = (a_e, b_e) \in E_1$, fit a copula C^B from $\mathcal{B}^{all}$ using the copula data $\mathbf{u}_e := \{u_{k,a_e}, u_{k,b_e}; k = 1, \dots, n\}$. This is done for all copulas in $\mathcal{B}^{all}$ using maximum likelihood estimation, i.e.,

$$\ell^B(\boldsymbol{\theta}^B; \mathbf{u}_e) = \prod_{k=1}^{n} c^B(u_{k,a_e}, u_{k,b_e}; \boldsymbol{\theta}^B)$$

is maximized for each $C^B \in \mathcal{B}^{all}$, yielding an estimate $\hat{\boldsymbol{\theta}}^B$. In the next step, the AIC for each C^B and corresponding $\hat{\boldsymbol{\theta}}^B$ is computed as:

$$\text{AIC}^B(\hat{\boldsymbol{\theta}}^B; \mathbf{u}_e) := -2 \sum_{k=1}^{n} \ln(c^B(u_{k,a_e}, u_{k,b_e}; \hat{\boldsymbol{\theta}}^B)) + 2k^B,$$

where $k^B := |\hat{\boldsymbol{\theta}}^B|$ denotes the number of estimated parameters. Finally, for edge e, the copula C^e with parameter $\boldsymbol{\theta}^e$ is chosen that minimizes $\text{AIC}^B(\hat{\boldsymbol{\theta}}^B; \mathbf{u}_e)$ over $\mathcal{B}^{all}$.

2.) Consider tree $T_i = (V_i, E_i)$ for $i > 1$. Suppose that all families and parameters for trees $T_1, \dots, T_{i-1}$ have already been estimated. The collection of parameter estimates is denoted by $\hat{\boldsymbol{\theta}}(T_{1,\dots,i-1})$. Further, for edge $e = (a_e, b_e; D_e) \in E_i$, define the *pseudo-observations* by

$$\hat{u}_{k,a_e|D_e} := u_{k,a_e|D_e,\hat{\boldsymbol{\theta}}(T_{1,\dots,i-1})} := C_{a_e|D_e}(u_{k,a_e} \mid \mathbf{u}_{k,D_e}; \hat{\boldsymbol{\theta}}(T_{1,\dots,i-1})),$$

$$\hat{u}_{k,b_e|D_e} := u_{k,b_e|D_e,\hat{\boldsymbol{\theta}}(T_{1,\dots,i-1})} := C_{b_e|D_e}(u_{k,b_e} \mid \mathbf{u}_{k,D_e}; \hat{\boldsymbol{\theta}}(T_{1,\dots,i-1})),$$

for $k = 1, \dots, n$. Using the pseudo-copula data $\mathbf{u}_e = \{\hat{u}_{k,a_e|D_e}, \hat{u}_{k,b_e|D_e}; k = 1, \dots, n\}$, the estimation procedure is done analogously to step 1. First, the likelihood

$$\ell^B(\boldsymbol{\theta}^B; \mathbf{u}_e) = \prod_{k=1}^{n} c^B(\hat{u}_{k,a_e|D_e}, \hat{u}_{k,b_e|D_e}; \boldsymbol{\theta}^B)$$

is maximized to find $\hat{\boldsymbol{\theta}}^B$ for each element in B^{all}. Then, the AIC

$$\mathrm{AIC}^B(\hat{\boldsymbol{\theta}}^B; \mathbf{u}_e) := -2 \sum_{k=1}^{n} \ln(c^B(\hat{u}_{k,a_e|D_e}, \hat{u}_{k,b_e|D_e}; \hat{\boldsymbol{\theta}}^B)) + 2k^B,$$

is minimized over B^{all} to finally determine the family and parameter(s) used for edge e.

3 Vine Copula Based Regression

In this chapter, two regression methods based on vine copulas will be presented. The first method is designed for univariate responses and employs D-vine tree structures. Using a D-vine structure where the response variable is positioned as the leaf node in the initial tree facilitates integration-free quantile computation. This capability will be leveraged for conditional simulation, and the sequential estimation procedure will be utilized for parameter learning in pair-copula Bayesian networks. The second method addresses bivariate responses, employing the Y-vine tree structure and a forward-looking algorithm to model the conditional distribution of two responses given a set of covariates. This method will introduce a new conditional independence test, which will be applied in subsequent chapters within the PC algorithm to learn the structure of Bayesian networks.

3.1 Univariate D-Vine-Based Regression

Kraus and Czado (2017) introduced a method for modeling the dependence between a univariate response and a set of covariates using D-vines. They developed an algorithm that sequentially fits a D-vine to the data by adding covariates one at a time, maximizing the conditional likelihood at each step. This process continues until no further improvement in the conditional likelihood is possible, effectively incorporating automatic variable selection. When applied to quantile regression, their method successfully addresses common issues of traditional approaches, such as quantile crossings.

To illustrate the concept, consider a univariate response Y with distribution function F_Y and a set of covariates $\mathbf{X} = (X_1, \ldots, X_p)^\top$, where each covariate X_j has a marginal distribution function F_j for $j = 1, \ldots, p$. The objectives of this section are the following:

i) Obtain an estimate of $F_{Y|X_1,\ldots,X_p}$, denoted by $\hat{F}_{Y|X_1,\ldots,X_p}$, assuming a D-vine structure.

ii) Use the inverse $\hat{F}^{-1}_{Y|X_1,\ldots,X_p}$ to simulate values of Y given the fixed values $(X_1, \ldots, X_p) = (x_1, \ldots, x_p)$.

Consider the probability integral transforms $V := F_Y(Y)$ and $U_j := F_j(X_j)$, with corresponding values $v := F_Y(y)$ and $u_j := F_j(u_j)$. Then, it follows that

$$\begin{aligned} F_{Y|X_1,\ldots,X_p}(y \mid x_1, \ldots, x_p) &= \mathbb{P}(Y \le y \mid X_1 = x_1, \ldots, X_p = x_p) \\ &= \mathbb{P}(V \le v \mid U_1 = u_1, \ldots, U_p = u_p) \\ &= C_{V|U_1,\ldots,U_p}(v \mid u_1, \ldots, u_p), \end{aligned}$$

where $C_{V|U_1,\ldots,U_p}$ denotes the conditional distribution of V given $U_1, \ldots, U_p$. Inverting this expression yields

$$F^{-1}_{Y|X_1,\ldots,X_p}(\alpha \mid x_1, \ldots, x_p) = F_Y^{-1}\Big(C^{-1}_{V|U_1,\ldots,U_p}(\alpha \mid u_1, \ldots, u_p)\Big),$$

for some $\alpha \in (0, 1)$. Thus, the estimation process can be split into two tasks:

i) Estimation of the marginal distributions F_Y, and $F_1, \ldots, F_p$.

ii) Estimation of $C_{V|U_1,\ldots,U_p}$, and $C^{-1}_{V|U_1,\ldots,U_p}$.

3.1.1 Kernel Density Estimation

A non-parametric and particularly flexible approach to estimate the marginal distributions was introduced by Parzen (1962) and is known as *kernel density estimation* when applied to probability density functions.

Definition 3.1.1 (Univariate local polynomial kernel density estimator)**.** Let X be a continuous random variable with distribution function F, and let $\mathbf{x} = (x_1, \dots, x_n)$ be a sample of X of size n. The **univariate local polynomial kernel density estimator** of X given the sample $\mathbf{x}$ is defined as

$$\hat{F}_X(x_0) = \frac{1}{nh} \sum_{i=1}^{n} K\Big(\frac{x_0 - x_i}{h}\Big), \quad x \in \mathbb{R},$$

where the **kernel** is given by $K(x) := \int_{-\infty}^{x} k(t)\mathrm{d}t$, with $k(\cdot)$ being a symmetric probability density function, and the **bandwidth** is given by $h > 0$.

While there are several choices for the kernel K, this thesis will exclusively use the *Gaussian kernel*, defined as

$$K(x) = \frac{1}{\sqrt{2\pi}} e^{-x^2/2}.$$

The bandwidth parameter $h > 0$ controls the smoothness of the estimated probability density function. Smaller bandwidths capture more detail but increase variance, while larger bandwidths produce smoother estimates at the cost of increased bias, highlighting the crucial bias-variance trade-off. Sheather and Jones (1991) developed a data-based method for selecting an optimal bandwidth h^*, which is used in all practical applications throughout this thesis. The described kernel density estimator is implemented in the R package `kde1d` (Nagler and Vatter 2024).

3.1.2 Sequential D-Vine Estimation

Let $\mathbf{y} = (y_i)_{i=1,\dots,n}$ and $\mathbf{x} = (\mathbf{x}_1^\top, \dots, \mathbf{x}_p^\top)$, where $\mathbf{x}_k = (x_{k,1}, \dots, x_{k,p})^\top$ for $k = 1, \dots, n$, be an i.i.d. sample of size n from the random vector $(Y, X_1, \dots, X_p)^\top$. The observed data is transformed to pseudo-copula data by setting $\hat{v}_i := \hat{F}_Y(y_i)$ and $\hat{u}_{i,j} = \hat{F}_j(x_{i,j})$ for $i = 1, \dots, n$ and $j = 1, \dots, p$, where kernel density estimation is used to estimate the marginals. The pseudo-copula data, $\hat{\mathbf{v}} = (v_i)_{i=1,\dots,n}$ and $\hat{\mathbf{u}} = (\hat{\mathbf{u}}_1^\top, \dots, \hat{\mathbf{u}}_p^\top)$, where $\hat{\mathbf{u}}_k = (\hat{u}_{k,1}, \dots, \hat{u}_{k,p})^\top$ for $k = 1, \dots, n$, is then approximately an i.i.d. sample from the random vector $(V, U_1, \dots, U_p)^\top$.

The objective is to fit a D-vine with order $V - U_{l_1} - \cdots - U_{l_p}$ to the pseudo-copula data. As will be demonstrated, choosing a D-vine with leaf node V is particularly advantageous, as it enables an integration-free computation of $c_{V|U_1,\dots,U_p}$.

Proposition 3.1.2. *For an estimated D-vine copula with order ℓ, parametric pair-copula families $\hat{\mathcal{F}}$, corresponding parameters $\hat{\boldsymbol{\theta}}$ given the pseudo-copula data $(\hat{\mathbf{v}}, \hat{\mathbf{u}})$, the conditional density $c_{V|U_1,\dots,U_p}$ can be expressed as the product over all pair-copula densities of the D-vine that contain V:*

$$\begin{aligned} c_{V|U_1,\dots,U_p}(\hat{v}_i \mid \hat{\mathbf{u}}_i; \ell, \hat{\mathcal{F}}, \hat{\boldsymbol{\theta}}) = {} & c_{VU_{l_1}}\big(\hat{v}_i, \hat{u}_{i,l_1}; \hat{\mathcal{F}}_{VU_{l_1}}, \hat{\theta}_{VU_{l_1}}\big) \\ & \cdot \prod_{j=2}^{p} c_{VU_{l_j};U_{l_1},\dots,U_{l_{j-1}}} \Big(\hat{C}_{V|U_{l_1},\dots,U_{l_{j-1}}}\big(\hat{v}_i \mid \hat{u}_{i,l_1}, \dots, \hat{u}_{i,l_{j-1}}\big), \\ & \hat{C}_{U_{l_j}|U_{l_1},\dots,U_{l_{j-1}}}\big(\hat{u}_{i,l_j} \mid \hat{u}_{i,l_1}, \dots, \hat{u}_{i,l_{j-1}}\big); \hat{\mathcal{F}}_{VU_{l_j};U_{l_1},\dots,U_{l_{j-1}}}, \hat{\theta}_{VU_{l_j};U_{l_1},\dots,U_{l_{j-1}}}\Big). \end{aligned}$$

A general proof of Proposition 3.1.2 can be found in Appendix 1 of Killiches et al. (2018). To illustrate the idea behind the proof, consider the following example:

Example 3.1.1 (Example 2.3.2 continued)**.** Suppose the response variable is U_1 and the covariates are $(U_2, U_3, U_4)^\top$. The conditional density of the response given the covariates is given by

$$c_{1|234}(u_1 \mid u_2, u_3, u_4) = \frac{c_{1234}(u_1, u_2, u_3, u_4)}{c_{234}(u_2, u_3, u_4)}.$$

An expression for c_{1234} is given by Equation (2.16) transformed to the u-level. According to Dißmann et al. (2013), Property 2.8 (ii), c_{234} can be expressed using the D-vine copula with the order $U_2 - U_3 - U_4$ as follows:

$$c_{234}(u_2, u_3, u_4) = c_{24;3}(C_{2|3}(u_2 \mid u_3), C_{4|3}(u_4 \mid u_3)) \cdot c_{34}(u_3, u_4) \cdot c_{23}(u_2, u_3).$$

Thus, the conditional density $c_{1|234}$ can be written as:

$$c_{1|234}(u_1 \mid u_2, u_3, u_4) = c_{14;23}(C_{1|23}(u_1 \mid u_2, u_3), C_{4|23}(u_4 \mid u_2, u_3)) \cdot c_{13;2}(C_{1|2}(u_1 \mid u_2), C_{3|2}(u_3 \mid u_2)) \cdot c_{12}(u_1, u_2).$$

This expression allows for an evaluation without integration, as all required components are derived recursively from the D-vine with order $U_1 - U_2 - U_3 - U_4$.

Instead of assuming a fixed order $\ell = (l_1, \dots, l_p)^\top$, let ℓ be a free parameter in the model. The goal is to select the order ℓ that maximizes the explanatory power of the D-vine copula model. To evaluate the model's fit, consider the *conditional log-likelihood (cll)*, defined as:

$$\text{cll}\,(\ell, \hat{\mathcal{F}}, \hat{\boldsymbol{\theta}}; \hat{\mathbf{v}}, \hat{\mathbf{u}}) := \sum_{i=1}^{n} \ln c_{V|U_1,\dots,U_p}(\hat{v}_i \mid \hat{\mathbf{u}}_i; \ell, \hat{\mathcal{F}}, \hat{\boldsymbol{\theta}}). \tag{3.1}$$

In total, there are $p!$ possible orders for the p covariates, making it impractical to fit and compare all of them. To address this, a step-wise approach is employed, where the D-vine is constructed incrementally by adding the most influential covariate at each step. The D-vine regression algorithm is outlined as follows:

i) **Step 1:** Start with the set of candidate covariates $\{U_{i_1}, \dots, U_{i_k}\}$ where $K_1 = \{i_1, \dots, i_k\}$ and initially $k = p$, meaning no variables have been selected yet. In this step, for any $c \in K_1$, adding the variable U_c to the D-vine introduces the structure $V - U_c$ and, consequently, the copula C_{VU_c}. The goal is to select $c \in K_1$ that maximizes the conditional log-likelihood given the pseudo-copula data. The cll for any $c \in K_1$ is given by

$$\text{cll}\,(\hat{\mathcal{F}}, \hat{\boldsymbol{\theta}}; \hat{\mathbf{v}}, \hat{\mathbf{u}}, c) = \sum_{i=1}^{n} \ln c_{VU_c}(\hat{v}_i, \hat{u}_{i,c}; \hat{F}_{VU_c}, \hat{\theta}_{VU_c}).$$

The maximum cll in the first step is denoted as $\text{cll}^1 := \max_{c\in K_1} \text{cll}\,(\hat{\mathcal{F}}, \hat{\boldsymbol{\theta}}; \hat{\mathbf{v}}, \hat{\mathbf{u}}, c)$, and the corresponding index is given by $l_1 := \arg\max_{c\in K_1} \text{cll}\,(\hat{\mathcal{F}}, \hat{\boldsymbol{\theta}}; \hat{\mathbf{v}}, \hat{\mathbf{u}}, c)$. Thus, U_{l_1} is added to the D-vine and removed from the set of candidate covariates.

ii) **Step r:** After completing the $(r-1)$-th step of the algorithm, the D-vine tree structure has the order $V - U_{l_1} - \dots - U_{l_{r-1}}$. At this stage, the set of candidate covariates is $\{U_{i_1}, \dots, U_{i_k}\}$ with $K_r = \{i_1, \dots, i_k\}$ and $k = p - (r-1)$, since $r-1$ variables have already been selected. The maximal conditional log-likelihood from the $(r-1)$-th step is denoted by cll^{r-1}, and the current order is $\ell^{r-1} = (l_1, \dots, l_{r-1})$. For any $c \in K_r$, adding the variable U_c to the D-vine introduces the structure $V - U_{l_1} - \dots - U_{l_{r-1}} - U_c$. The index $c \in K_r$ is determined in such a way that the conditional log-likelihood given the pseudo-copula data is maximized. The cll for any $c \in K_r$ is given by

$$\begin{aligned}\text{cll}\,(\ell^{r-1}, \hat{\mathcal{F}}, \hat{\boldsymbol{\theta}}; \hat{\mathbf{v}}, \hat{\mathbf{u}}, c) = \text{cll}^{r-1} + \sum_{i=1}^{n} \ln c_{VU_c;U_{l_1},\dots,U_{l_{r-1}}}\Big(&\hat{C}_{V|U_{l_1},\dots,U_{l_{r-1}}}(\hat{v}_i \mid \hat{u}_{i,l_1}, \dots, \hat{u}_{i,l_{r-1}}),\\ &\hat{C}_{U_c|U_{l_1},\dots,U_{l_{r-1}}}(\hat{u}_{i,c} \mid \hat{u}_{i,l_1}, \dots, \hat{u}_{i,l_{r-1}}); \hat{\mathcal{F}}_{VU_c;U_{l_1},\dots,U_{l_{r-1}}}, \hat{\theta}_{VU_c;U_{l_1},\dots,U_{l_{r-1}}}\Big).\end{aligned}$$

The maximum cll after the r-th step is denoted by $\text{cll}^r := \max_{c\in K_r} \text{cll}\,(\ell^{r-1}, \hat{\mathcal{F}}, \hat{\boldsymbol{\theta}}; \hat{\mathbf{v}}, \hat{\mathbf{u}}, c)$. The variable U_{l_r} is added to the order, where $l_r := \arg\max_{c\in K_r} \text{cll}\,(\ell^{r-1}, \hat{\mathcal{F}}, \hat{\boldsymbol{\theta}}; \hat{\mathbf{v}}, \hat{\mathbf{u}}, c)$.

It is important to note that the algorithm may terminate at the r-th step if adding any additional variables does not increase the conditional log-likelihood. This situation occurs when $C_{VU_{l_r};U_{l_1},\dots,U_{l_{r-1}}}$ is fitted as the independence copula, hence having a log-likelihood of zero. To potentially obtain even more parsimonious models, alternative approaches using AIC- or BIC-corrected versions of the conditional log-likelihood can be employed:

$$\mathrm{cll}^{\mathrm{AIC}}(\ell, \hat{\mathcal{F}}, \hat{\boldsymbol{\theta}}; \hat{\mathbf{v}}, \hat{\mathbf{u}}) := -2 \,\mathrm{cll}\,(\ell, \hat{\mathcal{F}}, \hat{\boldsymbol{\theta}}; \hat{\mathbf{v}}, \hat{\mathbf{u}}) + 2|\hat{\boldsymbol{\theta}}|,$$
$$\mathrm{cll}^{\mathrm{BIC}}(\ell, \hat{\mathcal{F}}, \hat{\boldsymbol{\theta}}; \hat{\mathbf{v}}, \hat{\mathbf{u}}) := -2 \,\mathrm{cll}\,(\ell, \hat{\mathcal{F}}, \hat{\boldsymbol{\theta}}; \hat{\mathbf{v}}, \hat{\mathbf{u}}) + \ln(n)|\hat{\boldsymbol{\theta}}|,$$

where $|\hat{\boldsymbol{\theta}}|$ denotes the number of estimated parameters, and n is the sample size. The one-step ahead algorithm for sequentially fitting a D-vine, as described in this section, is implemented in the R-package `vinereg` (Nagler and Kraus 2024). Additionally, Tepegjozova (2019) developed a less greedy selection procedure using a two-step ahead forward-looking algorithm and demonstrated in a case study that this approach can significantly improve the model fit compared to the one-step-ahead algorithm.

3.1.3 Conditional Simulation

Assume now that a D-vine copula model is fully specified, with the order ℓ, the families $\hat{\mathcal{F}}$, and the parameters $\hat{\boldsymbol{\theta}}$ all known. These may have been fixed or estimated, for example, using D-vine-based regression. Without loss of generality, assume that the order of the D-vine is given by $V - U_p - \cdots - U_1$. The objective is to simulate from the conditional distribution of V given the fixed values $U_1 = u_1^*, \ldots, U_p = u_p^*$. To generate samples from $C_{V|\mathbf{U}=\mathbf{u}^*}$, the following transformations, introduced by Rosenblatt (1952), are utilized:

Definition 3.1.3 (Rosenblatt transform and its inverse)**.** Let $\mathbf{X} = (X_1, \ldots, X_p)^\top$ be a random vector with distribution function F. The **Rosenblatt transform** $\mathbf{z} = T(\mathbf{x})$ is defined as:

$$z_1 = F_1(x_1), \quad z_2 = F_{2|1}(x_2 \mid x_1), \quad \ldots, \quad z_p = F_{p|1,\ldots,p-1}(x_p \mid x_1, \ldots, x_{p-1}).$$

The random variables $\mathbf{Z} = (Z_1, \ldots, Z_p)^\top = T(\mathbf{X})$ are mutually independent and follow a standard uniform distribution. Conversely, the **inverse Rosenblatt transform** $\mathbf{x} = T^{-1}(\mathbf{z})$ is given by

$$x_1 = F_1^{-1}(z_1), \quad x_2 = F_{2|1}^{-1}(z_2 \mid z_1), \quad \ldots, \quad x_p = F_{p|1,\ldots,p-1}^{-1}(z_p \mid z_1, \ldots, z_{p-1}).$$

For any joint distribution F, if $\mathbf{Z}$ is a vector of independent random variables each following a standard uniform distribution, then $\mathbf{X} = T^{-1}(\mathbf{Z})$ has distribution F.

The Rosenblatt transform and its inverse are implemented in the R-package `rvinecopulib` (Nagler and Vatter 2023). Following Aas et al. (2021), the procedure to generate the i-th sample of $C_{V|U_1=u_1^*,\ldots,U_p=u_p^*}$ is described as follows:

i) Let w be a sample from a standard uniform distribution. Set $\mathbf{v} = (u_1^*, \ldots, u_p^*, w)$.

ii) Apply the Rosenblatt transform to $\mathbf{v}$, i.e., set $\mathbf{u} = T(\mathbf{v})$:

$$\begin{aligned} u_1 &= C_{U_1}(u_1^*) = u_1^* \\ u_2 &= C_{U_2|U_1}(u_2^* \mid u_1^*) \\ &\cdots \\ u_p &= C_{U_p|U_{p-1},\ldots,U_1}(u_p^* \mid u_{p-1}^*, \ldots, u_1^*) \\ u_{p+1} &= C_{V|U_p,\ldots,U_1}(w \mid u_p^*, \ldots, u_1^*). \end{aligned}$$

iii) Obtain a sample z from the standard uniform distribution and set $\mathbf{u} = (u_1, \ldots, u_p, z)$.

iv) Apply the inverse Rosenblatt to $\mathbf{u}$ and set $\mathbf{v} = T^{-1}(\mathbf{u})$:

$$\begin{aligned} v_1 &= C_{U_1}^{-1}(u_1) = u_1 = u_1^* \\ v_2 &= C_{U_2|U_1}^{-1}(u_2 \mid u_1) = C_{U_2|U_1}^{-1}(C_{U_2|U_1}(u_2^* \mid u_1^*) \mid u_1^*) = u_2^* \\ &\cdots \\ v_p &= C_{U_p|U_{p-1},\ldots,U_1}^{-1}(u_p \mid u_{p-1}, \ldots, u_1) = C_{U_p|U_{p-1},\ldots,U_1}^{-1}(C_{U_p|U_{p-1},\ldots,U_1}(u_p^* \mid u_{p-1}^*, \ldots, u_1^*) \mid u_{p-1}, \ldots, u_1) = u_p^* \\ v_{p+1} &= C_{V|U_p,\ldots,U_1}^{-1}(z \mid u_p, \ldots, u_1). \end{aligned}$$

Thus, a sample from V given the fixed values $U_1 = u_1^*, \dots, U_p = u_p^*$ is given by v_{p+1}. Step ii) of the procedure is necessary to ensure that the values of the conditional variables are consistent in all samples.

3.2 Bivariate Y-Vine-Based Regression

Many data applications require joint modeling of bivariate responses given a set of covariates. The method of D-vine-based regression, as introduced in Section 3.1, can model the dependence between the covariates and between the response and the covariates. Tepegjozova and Czado (2023) extended the univariate case by introducing a novel vine tree structure, termed *Y-vine*, which is designed to model the dependence between two response variables. Using Y-vines, it has been demonstrated that the associated bivariate conditional density of the responses given the covariates can be expressed as a product of pair copulas involving at least one of the responses. Moreover, all pair copulas involved can be directly obtained from the vine structure, eliminating the need for integration. Additionally, Y-vine regression allows for symmetric treatment of the responses, which is crucial for ensuring consistent results across different response sets. Unless otherwise stated, the content of this section is based on Tepegjozova (2024).

To illustrate, consider the bivariate response vector $\mathbf{Y} = (Y_1, Y_2)^\top$ with corresponding marginal distribution functions F_{Y_i} for $i = 1, 2$, and the p-dimensional covariate vector $\mathbf{X} = (X_1, \dots, X_p)^\top$ with marginal distribution functions denoted as F_{X_i} for $i = 1, \dots, p$.

Proposition 3.2.1. *Using the probability integral transforms, let* $\mathbf{V} = (V_1, V_2)^\top := (F_{Y_1}(Y_1), F_{Y_2}(Y_2))^\top$ *and* $\mathbf{U} = (U_1, \dots, U_p)^\top := (F_{X_1}(X_1), \dots, F_{X_p}(X_p))^\top$. *The conditional distribution of* $\mathbf{Y}$ *given* $\mathbf{X}$ *can be expressed as*

$$F_{Y_1,Y_2|\mathbf{X}}(y_1, y_2 \mid \mathbf{x}) = C_{V_1,V_2|\mathbf{U}}(v_1, v_2 \mid \mathbf{u}),$$

where $\mathbf{v} = (v_1, v_2)^\top = (F_{Y_1}(y_1), F_{Y_2}(y_2))^\top$ *and* $\mathbf{u} = (u_1, \dots, u_p)^\top = (F_{X_1}(x_1), \dots, F_{X_p}(x_p))^\top$.

Proof.

$$\begin{aligned}
F_{Y_1,Y_2|\mathbf{X}}(y_1, y_2 \mid \mathbf{x}) &= \int_{-\infty}^{y_1}\int_{-\infty}^{y_2} f_{Y_1,Y_2|\mathbf{X}}(y_1', y_2' \mid \mathbf{x})\,\mathrm{d}y_2'\,\mathrm{d}y_1' \\
&= \int_{-\infty}^{y_1}\int_{-\infty}^{y_2} \frac{f_{Y_1,Y_2,\mathbf{X}}(y_1', y_2', \mathbf{x})}{f_{\mathbf{X}}(\mathbf{x})}\,\mathrm{d}y_2'\,\mathrm{d}y_1' \\
&= \frac{1}{f_{\mathbf{X}}(\mathbf{x})}\int_{-\infty}^{y_1}\int_{-\infty}^{y_2} \frac{\partial^{p+2}}{\partial y_1 \partial y_2 \partial x_1 \dots \partial x_p} F_{Y_1,Y_2,\mathbf{X}}(y_1, y_2, \mathbf{x})\Big|_{y_1=y_1', y_2=y_2'}\,\mathrm{d}y_2'\,\mathrm{d}y_1' \\
&= \frac{1}{f_{\mathbf{X}}(\mathbf{x})} \cdot \frac{\partial^p}{\partial x_1 \dots \partial x_p} F_{Y_1,Y_2,\mathbf{X}}(y_1, y_2, \mathbf{x}) \\
&= \frac{1}{f_{\mathbf{X}}(\mathbf{x})} \cdot \frac{\partial^p}{\partial x_1 \dots \partial x_p} C_{V_1,V_2,\mathbf{U}}\big(F_{Y_1}(y_1), F_{Y_2}(y_2), F_{X_1}(x_1), \dots, F_{X_p}(x_p)\big) \\
&= \frac{1}{f_{\mathbf{X}}(\mathbf{x})} \cdot \frac{\partial^p}{\partial u_1 \dots \partial u_p} C_{V_1,V_2,\mathbf{U}}(v_1, v_2, u_1, \dots, u_p)\Big|_{v_j=F_{Y_j}(y_j), u_i=F_{X_i}(x_i)} \cdot \frac{\partial u_1 \dots \partial u_p}{\partial x_1 \dots \partial x_p} \\
&= \frac{1}{f_{\mathbf{X}}(\mathbf{x})} \cdot \frac{\partial^p}{\partial u_1 \dots \partial u_p} C_{V_1,V_2,\mathbf{U}}(v_1, v_2, u_1, \dots, u_p)\Big|_{v_j=F_{Y_j}(y_j), u_i=F_{X_i}(x_i)} \cdot \prod_{i=1}^p f_{X_i}(x_i) \\
&= \frac{\partial^p}{\partial u_1 \dots \partial u_p} C_{V_1,V_2,\mathbf{U}}(v_1, v_2, u_1, \dots, u_p)\Big|_{v_j=F_{Y_j}(y_j), u_i=F_{X_i}(x_i)} \cdot \frac{1}{c_{\mathbf{U}}(\mathbf{u})} \qquad \text{(Sklar's Theorem)} \\
&= \frac{1}{c_{\mathbf{U}}(\mathbf{u})}\int_0^{v_1}\int_0^{v_2} \frac{\partial^{p+2}}{\partial v_1 \partial v_2 \partial u_1 \dots \partial u_p} C_{V_1,V_2,\mathbf{U}}(v_1, v_2, u_1, \dots, u_p)\Big|_{v_1=v_1', v_2=v_2'}\,\mathrm{d}v_2'\,\mathrm{d}v_1' \\
&= \int_0^{v_1}\int_0^{v_2} c_{V_1,V_2|\mathbf{U}}(v_1', v_2' \mid \mathbf{u})\,\mathrm{d}v_2'\,\mathrm{d}v_1' \\
&= C_{V_1,V_2|\mathbf{U}}\big(v_1, v_2 \mid \mathbf{u}\big).
\end{aligned}$$

□

From Proposition 3.2.1, it follows that to model the bivariate distribution $F_{Y_1,Y_2|\mathbf{X}}$, one needs to obtain the marginal distributions F_{Y_i} for $i = 1, 2$, F_{X_j} for $j = 1, \dots, p$, and the bivariate conditional distribution $C_{V_1,V_2|\mathbf{U}}$. Similar to the univariate case, the marginal distributions are estimated non-parametrically using kernel density estimation, as described in Section 3.1.1. Finally, it should be noted that the conditional distribution $C_{V_1,V_2|\mathbf{U}}$ is different from $C_{V_1,V_2;\mathbf{U}}$, as the latter denotes the copula associated with the bivariate conditional distribution of (Y_1, Y_2) given $\mathbf{X}$.

3.2.1 Y-Vine Copula Model

In the following, let $\mathbf{X}_{-i} := (X_1, \dots, X_{i-1}, X_{i+1}, \dots, X_p)^\top$ denote the $(p-1)$-dimensional vector obtained by removing the i-th variable from $\mathbf{X}$, for $i = 1, \dots, p$. Additionally, define the $(k+1)$-dimensional vector $\mathbf{X}_{i:i+k} := (X_i, \dots X_{i+k})^\top$, where $k = 0, 1, \dots, p-i$. This indexing convention also applies to all other random variables and observations used.

Definition 3.2.2 (Y-vine tree sequence)**.** Given the transformed response variables $\mathbf{V} = (V_1, V_2)^\top$ and the transformed covariates $\mathbf{U} = (U_1, \dots, U_p)^\top$, the **Y-vine tree sequence** consists of the following $p+1$ trees:

$$T_1 \text{ with } N_1 = \{V_1, V_2, U_1, \dots, U_p\} \text{ and } E_1 = \{\{V_1, U_1\}, \{V_2, U_1\}\} \cup \bigcup_{i=1}^{p-1} \{\{U_i, U_{i+1}\}\}.$$

$$T_2 \text{ with } N_2 = \{V_1U_1, V_2U_1, U_1U_2, \dots, U_{p-1}U_p\}$$

$$\text{and } E_2 = \{\{V_1U_1, U_1U_2\}, \{V_2U_1, U_1U_2\}\} \cup \bigcup_{i=1}^{p-2} \{\{U_iU_{i+1}, U_{i+1}U_{i+2}\}\}.$$

$$T_k \text{ for } 3 \le k \le p \text{ with } N_k = \bigcup_{j=1,2} \{V_jU_{k-1}; U_{1:k-2}\} \cup \bigcup_{i=1}^{p-k+1} \{U_iU_{i+k-1}; U_{i+1:i+k-2}\}$$

$$\text{and } E_k = \bigcup_{j=1,2} \{\{V_jU_{k-1}; U_{1:k-2}, U_1U_k; U_{2:k-1}\}\} \cup \bigcup_{i=1}^{p-k} \{\{U_iU_{i+k-1}; U_{i+1:i+k-2}, U_{i+1}U_{i+k}; U_{i+2:i+k-1}\}\}.$$

$$T_{p+1} \text{ with } N_{p+1} = \bigcup_{j=1,2} \{V_jU_p; U_{1:p-1}\} \text{ and } E_{p+1} = \{\{V_1U_p; U_{1:p-1}, V_2U_p; U_{1:p-1}\}\}.$$

Figure 3.1 depicts a Y-vine tree sequence with $p = 3$ covariates. The figure illustrates that the covariate nodes are structured similarly to those in a D-vine, while the nodes corresponding to the responses are leaf nodes at one end of the tree. Removing a response node, either V_1 or V_2, in the first tree results in the same structure as the univariate D-vine regression model. Importantly, the Y-vine tree sequence in Figure 3.1 is indeed an R-vine tree sequence. This property holds for any $p \in \mathbb{N}$:

Proposition 3.2.3. *A Y-vine tree sequence satisfies the conditions i)-iii) from Definition 2.3.11, and thus, it is a valid R-vine tree sequence.*

Proof. Condition i) is trivial and follows simply from the definition of T_1. Condition ii) states that $N_j = E_{j-1}$ for $j \ge 2$. For $j = 2$, it is evident from Definition 3.2.2 that $N_2 = E_1$. For $j = k > 2$:

$$N_k = \bigcup_{j=1,2} \{V_jU_{k-1}; U_{1:k-2}\} \cup \bigcup_{i=1}^{p-k+1} \{U_iU_{i+k-1}; U_{i+1:i+k-2}\}$$

$$E_{k-1} = \bigcup_{j=1,2} \{\{V_jU_{k-2}; U_{1:k-3}, U_1U_{k-1}; U_{2:k-2}\}\} \cup \bigcup_{i=1}^{p-(k-1)} \{\{U_iU_{i+k-2}; U_{i+1:i+k-3}, U_{i+1}U_{i+k-1}; U_{i+2:i+k-2}\}\}$$

The edges $\{V_jU_{k-2}; U_{1:k-3}, U_1U_{k-1}; U_{2:k-2}\}$ in T_{k-1} correspond to the nodes $\{V_jU_{k-1}; U_{1:k-2}\}$ in T_k, for $j = 1, 2$. Further, the edges $\{U_iU_{i+k-2}; U_{i+1:i+k-3}, U_{i+1}U_{i+k-1}; U_{i+2:i+k-2}\}$ in T_{k-1} are associated with the nodes $\{U_iU_{i+k-1}; U_{i+1:i+k-2}\}$ in T_k. Hence, condition ii) holds as well. Condition iii), known as the proximity

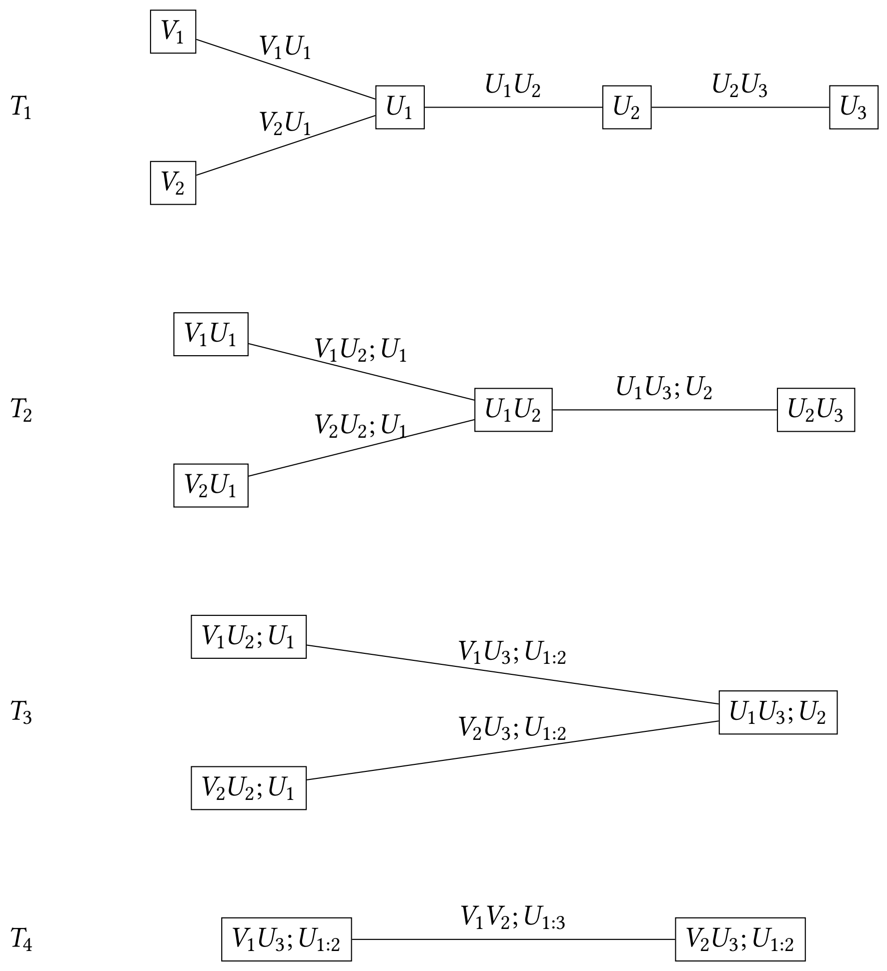


Figure 3.1 Y-vine tree sequence for $p = 3$ covariates

condition, states that for $j \geq 2$, any two nodes connected by an edge in T_j must share a common node in T_{j-1}. Considering only the nodes and edges corresponding to the covariates $(U_1, \dots, U_p)^\top$, the proximity condition is clearly fulfilled as these nodes and edges alone form a D-vine tree sequence. Considering the remaining nodes and edges, involving the covariates, first note that, for T_2, V_jU_1, $j = 1, 2$, is connected to U_1U_2. Therefore, they share the common node U_1 in T_1. For $j = k > 2$, the nodes $V_jU_{k-1}; \mathbf{U}_{1:k-2}$, $j = 1, 2$, are connected to $U_1U_k; \mathbf{U}_{2:k-1}$ in T_k, sharing the common node $U_1U_{k-1}; \mathbf{U}_{2:k-2}$ in T_{k-1}. □

A *Y-vine distribution* is a regular vine distribution associated with a Y-vine tree sequence. To express the joint density of a Y-vine copula, Equation (2.15) is applied. The joint density is given by:

$$\begin{aligned} c_{V_1,V_2,\mathbf{U}}(v_1, v_2, \mathbf{u}) = &\prod_{k=1}^{p-1} \Big(\prod_{i=1}^{p-k} c_{U_i,U_{i+k};\mathbf{U}_{i+1:i+k-1}} \left(C_{U_i|\mathbf{U}_{i+1:i+k-1}}(u_i|\mathbf{u}_{i+1:i+k-1}), C_{U_{i+k}|\mathbf{U}_{i+1:i+k-1}}(u_{i+k}|\mathbf{u}_{i+1:i+k-1}) \right) \Big) \\ &\cdot \prod_{i=1}^{p} \Big(\prod_{j=1,2} c_{V_j,U_i;\mathbf{U}_{1:i-1}} \left(C_{V_j|\mathbf{U}_{1:i-1}}(v_j|\mathbf{u}_{1:i-1}), C_{U_i|\mathbf{U}_{1:i-1}}(u_i|\mathbf{u}_{1:i-1}) \right) \Big) \qquad (3.2) \\ &\cdot c_{V_1,V_2;\mathbf{U}} \left(C_{V_1|\mathbf{U}}(v_1|\mathbf{u}), C_{V_2|\mathbf{U}}(v_2|\mathbf{u}) \right). \end{aligned}$$

In this formula, $U_{a:b} := \emptyset$ if $a > b$. For example,

$$c_{U_1,U_2;U_{2:1}} \left(C_{U_1|U_{2:1}}(u_1|u_{2:1}), C_{U_2|U_{2:1}}(u_2|u_{2:1}) \right) = c_{U_1,U_2} \left(C_{U_1}(u_1), C_{U_2}(u_2) \right) = c_{U_1,U_2}(u_1, u_2).$$

Note that the first line of Equation (3.2) represents the density of a D-vine copula with structure $U_1 - \cdots - U_p$, denoted by $c_{\mathbf{U}}$.

Theorem 3.2.4 (Bivariate conditional density of a Y-vine copula)**.** *The conditional density of* $\mathbf{V} = (V_1, V_2)^\top$ *given the covariates* $\mathbf{U} = (U_1, \dots, U_p)^\top$ *in a Y-vine copula is expressed as:*

$$\begin{aligned} c_{V_1,V_2|\mathbf{U}}(v_1, v_2 \mid \mathbf{u}) = &\prod_{i=1}^{p} \Big(\prod_{j=1,2} c_{V_j,U_i;\mathbf{U}_{1:i-1}} \left(C_{V_j|\mathbf{U}_{1:i-1}}(v_j|\mathbf{u}_{1:i-1}), C_{U_i|\mathbf{U}_{1:i-1}}(u_i|\mathbf{u}_{1:i-1}) \right) \Big) \\ &\cdot c_{V_1,V_2;\mathbf{U}} \left(C_{V_1|\mathbf{U}}(v_1|\mathbf{u}), C_{V_2|\mathbf{U}}(v_2|\mathbf{u}) \right) \end{aligned}$$

Proof. Using the definition of the conditional density, it holds that

$$c_{V_1,V_2|\mathbf{U}}(v_1, v_2 \mid \mathbf{u}) = \frac{c_{V_1,V_2,\mathbf{U}}(v_1, v_2, \mathbf{u})}{c_{\mathbf{U}}(\mathbf{u})}. \qquad (3.3)$$

Here, the joint density $c_{V_1,V_2,\mathbf{U}}(v_1, v_2, \mathbf{u})$ is given by Equation (3.2). Further, using Sklar's Theorem, the joint density of the covariates is given by $c_{\mathbf{U}}$ and corresponds to the density of the D-vine involving only the covariates. It is given by the first line of Equation (3.2), as stated before. By substituting both joint densities into Equation (3.3) and simplifying the term, the result is obtained. □

To derive the conditional density $c_{V_1,V_2|\mathbf{U}}$, Theorem 3.2.4 shows that the result mirrors the univariate case. Specifically, the conditional density of the responses given the covariates is expressed as a product of pair copulas involving at least one of the responses. Additionally, similar to the univariate scenario, the conditional densities of individual responses given the covariates can be obtained in closed form, without requiring integration.

Theorem 3.2.5 (Univariate conditional densities of a Y-vine copula)**.** *The conditional density of* V_j*, for* $j = 1, 2$*, given the covariates* $\mathbf{U} = (U_1, \dots, U_p)^\top$ *in a Y-vine copula is given by*

$$c_{V_j|\mathbf{U}}(v_j \mid \mathbf{u}) = \prod_{i=1}^{p} c_{V_j,U_i;\mathbf{U}_{1:i-1}} \left(C_{V_j|\mathbf{U}_{1:i-1}}(v_j|\mathbf{u}_{1:i-1}), C_{U_i|\mathbf{U}_{1:i-1}}(u_i|\mathbf{u}_{1:i-1}) \right).$$

Proof. Again, it clearly holds that

$$c_{V_j|\mathbf{U}}(v_j \mid \mathbf{u}) = \frac{c_{V_j,\mathbf{U}}(v_j, \mathbf{u})}{c_{\mathbf{U}}(\mathbf{u})}, \quad \text{for } j = 1, 2. \tag{3.4}$$

As before, $c_{\mathbf{U}}(\mathbf{u})$ is given by the first line in Equation (3.2). To obtain $c_{V_j,\mathbf{U}}$, consider the Y-vine where all edges and nodes involving V_k for $k \neq j$ are removed. This simplifies to a D-vine structure of the form $V_j - U_1 - \cdots - U_p$. Thus, the joint density is given by:

$$\begin{aligned} c_{V_j,\mathbf{U}}(v_j, \mathbf{u}) = &\prod_{k=1}^{p-1} \Big(\prod_{i=1}^{p-k} c_{U_i,U_{i+k};\mathrm{U_{i+1:i+k-1}}} \big(C_{U_i|\mathrm{U_{i+1:i+k-1}}}(u_i|\mathbf{u}_{\mathrm{i+1:i+k-1}}), C_{U_{i+k}|\mathrm{U_{i+1:i+k-1}}}(u_{i+k}|\mathbf{u}_{\mathrm{i+1:i+k-1}})\big) \Big) \\ &\cdot \prod_{i=1}^{p} c_{V_j,U_i;\mathrm{U_{1:i-1}}} \Big(C_{V_j|\mathrm{U_{1:i-1}}}(v_j|\mathbf{u}_{\mathrm{1:i-1}}), C_{U_i|\mathrm{U_{1:i-1}}}(u_i|\mathbf{u}_{\mathrm{1:i-1}})\Big). \end{aligned}$$

Substituting both joint densities into (3.4) and simplifying the term yields the result. □

It is important to note that the univariate conditional density of a single response given all covariates can be expressed as a product of pair copulas involving only the response variable of interest.

3.2.2 Sequential Y-Vine Estimation

The objective of this section is to fit a Y-vine with covariate order $U_{l_1} - \cdots - U_{l_p}$ to a given data set. The order itself is a free parameter that influences the model fit. Computing all $p!$ permutations of the order is computationally expensive. Therefore, a sequential algorithm is employed to add covariates step by step to the model. As in the univariate case, the conditional log-likelihood (cll) can be used as a criterion for model fitting.

To illustrate, let $\mathbf{y_j} = (y_{i,j})_{i=1,\dots,n}$ for $j = 1, 2$, and $\mathbf{x} = (\mathbf{x}_1^\top, \dots, \mathbf{x}_p^\top)$, where $\mathbf{x}_k = (x_{k,1}, \dots, x_{k,p})^\top$ for $k = 1, \dots, n$, be an i.i.d. sample of size n from the random vector $(Y_1, Y_2, X_1, \dots, X_p)^\top$. The observed data is transformed to pseudo-copula data by setting $\hat{v}_{i,j} := \hat{F}_{Y_j}(y_{i,j})$ and $\hat{u}_{i,j} = \hat{F}_{X_k}(x_{i,k})$ for $i = 1, \dots, n$, $j = 1, 2$, and $k = 1, \dots, p$, where kernel density estimation is used to estimate the marginals. The pseudo-copula data $\hat{\mathbf{v}}_j = (v_{i,j})_{i=1,\dots,n}$, $j = 1, 2$, and $\hat{\mathbf{u}} = (\hat{\mathbf{u}}_1^\top, \dots, \hat{\mathbf{u}}_p^\top)$, where $\hat{\mathbf{u}}_k = (\hat{u}_{k,1}, \dots, \hat{u}_{k,p})^\top$ for $k = 1, \dots, n$, is approximately an i.i.d. sample from the random vector $(V_1, V_2, U_1, \dots, U_p)^\top$. The conditional log-likelihood for a given covariate order ℓ, estimated copula families $\hat{\mathcal{F}}$, and parameters $\hat{\boldsymbol{\theta}}$ is defined as:

$$\begin{aligned} \mathrm{cll}\,(\ell, \hat{\mathcal{F}}, \hat{\boldsymbol{\theta}}; \hat{\mathbf{v}}, \hat{\mathbf{u}}) :=& \sum_{i=1}^{n} \ln c_{V_1,V_2|\mathrm{U}}(\hat{v}_{i,1}, \hat{v}_{i,2} \mid \hat{\mathbf{u}}_i; \ell, \hat{\mathcal{F}}, \hat{\boldsymbol{\theta}}) \\ =& \sum_{i=1}^{n} \ln c_{V_1,V_2;\mathrm{U}}(\hat{C}_{V_1|\mathrm{U}}(\hat{v}_{i,1} \mid \hat{\mathbf{u}}_i), \hat{C}_{V_2|\mathrm{U}}(\hat{v}_{i,2} \mid \hat{\mathbf{u}}_i); \hat{\mathcal{F}}_{V_1V_2;\mathrm{U}}, \hat{\theta}_{V_1V_2;\mathrm{U}}) \\ &+ \sum_{i=1}^{n} \ln c_{V_1|\mathrm{U}}(\hat{v}_{i,1} \mid \hat{\mathbf{u}}_i; \ell, \hat{\mathcal{F}}, \hat{\boldsymbol{\theta}}) + \sum_{i=1}^{n} \ln c_{V_2|\mathrm{U}}(\hat{v}_{i,2} \mid \hat{\mathbf{u}}_i; \ell, \hat{\mathcal{F}}, \hat{\boldsymbol{\theta}}) \\ =& \sum_{i=1}^{n} \ln c_{V_1,V_2;\mathrm{U}}(\hat{C}_{V_1|\mathrm{U}}(\hat{v}_{i,1} \mid \hat{\mathbf{u}}_i), \hat{C}_{V_2|\mathrm{U}}(\hat{v}_{i,2} \mid \hat{\mathbf{u}}_i); \hat{\mathcal{F}}_{V_1V_2;\mathrm{U}}, \hat{\theta}_{V_1V_2;\mathrm{U}}) \\ &+ \sum_{j=1,2} \Big(\sum_{i=1}^{n} \ln c_{V_j,U_{l_1}}(\hat{v}_{i,j}, \hat{u}_{i,l_1}; \hat{\mathcal{F}}_{V_jU_1}, \hat{\theta}_{V_jU_1}) \\ &+ \sum_{k=2}^{p} \sum_{i=1}^{n} \ln c_{V_j,U_{l_k};\mathrm{U}_{l_1:l_{k-1}}}(\hat{C}_{V_j|\mathrm{U}_{l_1:l_{k-1}}}(\hat{v}_{i,j} \mid \hat{\mathbf{u}}_{i,l_1:l_{k-1}}), \hat{C}_{U_{l_k}|\mathrm{U}_{l_1:l_{k-1}}}(\hat{u}_{l_k} \mid \hat{\mathbf{u}}_{i,l_1:l_{k-1}}); \\ &\qquad \hat{\mathcal{F}}_{V_j,U_{l_k};\mathrm{U}_{l_1:l_{k-1}}}, \hat{\theta}_{V_j,U_{l_k};\mathrm{U}_{l_1:l_{k-1}}}) \Big) \end{aligned} \tag{3.5}$$

The copula $c_{V_j,U_{l_k};\mathbf{U}_{l_1:l_{k-1}}}$ represents the distribution of (V_j, U_{l_k}) given that the effects of $U_{l_1}, \ldots, U_{l_{k-1}}$ are adjusted. Therefore, a large value of the corresponding log-likelihood indicates that U_{l_k} has an influence on the response V_j after accounting for $U_{l_1:l_{k-1}}$. In contrast, the pair copula $c_{V_1,V_2;\mathbf{U}}$ does not have a similar interpretation, as it represents the copula associated with the conditional distribution of (V_1, V_2) given the effects $U_{l_1}, \ldots, U_{l_k}$. Neither an increase nor a decrease in the corresponding log-likelihood can be interpreted as an increase in influence for a single covariate. Therefore, to use the cll as a measure of fit, appropriate adjustments need to be made.

Definition 3.2.6 (Adjusted conditional log-likelihood)**.** The **adjusted conditional log-likelihood (acll)** of a Y-vine copula model is defined as

$$\begin{aligned}
\text{acll}\,(\ell, \hat{\mathcal{F}}, \hat{\boldsymbol{\theta}}; \hat{\mathbf{v}}, \hat{\mathbf{u}}) &:= \sum_{i=1}^{n} \ln c_{V_1,V_2|\mathbf{U}}(\hat{v}_{i,1}, \hat{v}_{i,2} \mid \hat{\mathbf{u}}_i; \ell, \hat{\mathcal{F}}, \hat{\boldsymbol{\theta}}) \\
&\quad - \sum_{i=1}^{n} \ln c_{V_1,V_2;\mathbf{U}}(\hat{C}_{V_1|\mathbf{U}}(\hat{v}_{i,1} \mid \hat{\mathbf{u}}_i), \hat{C}_{V_2|\mathbf{U}}(\hat{v}_{i,2} \mid \hat{\mathbf{u}}_i); \hat{\mathcal{F}}_{V_1V_2;\mathbf{U}}, \hat{\theta}_{V_1V_2;\mathbf{U}}) \\
&= \sum_{i=1}^{n} \ln c_{V_1|\mathbf{U}}(\hat{v}_{i,1} \mid \hat{\mathbf{u}}_i; \ell, \hat{\mathcal{F}}, \hat{\boldsymbol{\theta}}) + \sum_{i=1}^{n} \ln c_{V_2|\mathbf{U}}(\hat{v}_{i,2} \mid \hat{\mathbf{u}}_i; \ell, \hat{\mathcal{F}}, \hat{\boldsymbol{\theta}}) \\
&= \sum_{j=1,2} \Big(\sum_{i=1}^{n} \ln c_{V_j,U_{l_1}}(\hat{v}_{i,j}, \hat{u}_{i,l_1}; \hat{\mathcal{F}}_{V_jU_1}, \hat{\theta}_{V_jU_1}) \qquad (3.6) \\
&\quad + \sum_{k=2}^{p} \sum_{i=1}^{n} \ln c_{V_j,U_{l_k};\mathbf{U}_{l_1:l_{k-1}}}(\hat{C}_{V_j|\mathbf{U}_{l_1:l_{k-1}}}(\hat{v}_{i,j} \mid \hat{\mathbf{u}}_{i,l_1:l_{k-1}}), \hat{C}_{U_{l_k}|\mathbf{U}_{l_1:l_{k-1}}}(\hat{u}_{l_k} \mid \hat{\mathbf{u}}_{i,l_1:l_{k-1}}); \\
&\qquad \hat{\mathcal{F}}_{V_j,U_{l_k};\mathbf{U}_{l_1:l_{k-1}}}, \hat{\theta}_{V_j,U_{l_k};\mathbf{U}_{l_1:l_{k-1}}})\Big)
\end{aligned}$$

Assume that a Y-vine copula with covariate order $U_{l_1}, \ldots, U_{l_{k-1}}$ has already been fitted. To evaluate whether including a further variable U_{l_k} from the candidate set improves the model fit, let the adjusted conditional log-likelihood with covariate order given by $\ell^{k-1} = (l_1, \ldots, l_{k-1})$ be denoted by

$$\text{acll}\,(\ell^{k-1}, \hat{\mathcal{F}}^{k-1}, \hat{\boldsymbol{\theta}}^{k-1}; \hat{\mathbf{v}}, \hat{\mathbf{u}}_{l_1:l_{k-1}}).$$

The acll after adding U_{l_k} to the model is then given by

$$\begin{aligned}
\text{acll}\,(\ell^{k}, \hat{\mathcal{F}}^{k}, \hat{\boldsymbol{\theta}}^{k}; \hat{\mathbf{v}}, \hat{\mathbf{u}}_{l_1:l_k}) &= \text{acll}\,(\ell^{k-1}, \hat{\mathcal{F}}^{k-1}, \hat{\boldsymbol{\theta}}^{k-1}; \hat{\mathbf{v}}, \hat{\mathbf{u}}_{l_1:l_{k-1}}) \\
&\quad + \sum_{j=1,2} \sum_{i=1}^{n} \ln c_{V_j,U_{l_k};\mathbf{U}_{l_1:l_{k-1}}}\big(\hat{C}_{V_j|\mathbf{U}_{l_1:l_{k-1}}}(\hat{v}_{i,j} \mid \hat{\mathbf{u}}_{i,l_1:l_{k-1}}), \hat{C}_{U_{l_k}|\mathbf{U}_{l_1:l_{k-1}}}(\hat{u}_{l_k} \mid \hat{\mathbf{u}}_{i,l_1:l_{k-1}}); \\
&\qquad \hat{\mathcal{F}}_{V_j,U_{l_k};\mathbf{U}_{l_1:l_{k-1}}}, \hat{\theta}_{V_j,U_{l_k};\mathbf{U}_{l_1:l_{k-1}}}\big). \qquad (3.7)
\end{aligned}$$

The sequential estimation process closely resembles the method outlined in Section 3.1.2. At each step, the covariate from the candidate set that maximizes the increase in the adjusted conditional log-likelihood (acll) is added to the model. The algorithm terminates either when all covariates have been added or when no additional covariate from the candidate set improves the acll. The latter situation occurs when the densities $c_{V_j,U_{l_k};\mathbf{U}_{l_1:l_{k-1}}}$ (for $j = 1, 2$) from Equation (3.7) correspond to independence copulas, resulting in a log-likelihood of zero. This approach ensures an automatic forward selection process, including only those covariates that have an impact on at least one of the response variables. For increased model parsimony, one can further adjust the acll using AIC or BIC penalization. Additionally, the algorithm can be extended to a two-step-ahead forward-looking procedure for potentially improved results.

3.2.3 Y-Vine Conditional Independence Test

This section presents a new Y-vine-based conditional independence test. The goal is to test the hypothesis $H_0 : X_i \perp\!\!\!\perp X_j \mid \mathbf{X}_S$ against its alternative $H_1 : X_i \not\!\perp\!\!\!\perp X_j \mid \mathbf{X}_S$ by modeling a Y-vine with (X_i, X_j) as responses and $\mathbf{X}_{\mathrm{S}}$ as covariates. The approach is appealing because Y-vines provide a symmetric treatment of the responses and do not require the assumption of joint Gaussian distributions. Additionally, their flexibility eliminates the need for asymptotic results.

Later, the proposed conditional independence test will be employed in the PC algorithm to learn the structure of a Bayesian network. The performance of this method will be compared to that of the PC algorithm using Fisher's Z-test through simulations involving both Gaussian and non-Gaussian data.

Bauer and Czado (2016) introduced a R-vine-based conditional independence test. Their approach involved estimating a R-vine structure under the constraint that neither X_i nor X_j could be part of an inner node (a node with at least two neighbors) in any of the trees in the R-vine sequence. This construction ensures that the copula $C_{ij;\mathrm{S}}$ is always present in the last tree of the R-vine. They then applied ordinary independence tests based on the pseudo-observations $\hat{u}_{i|\mathrm{S}} := C_{i|\mathrm{S}}(u_i \mid \mathbf{u}_S; \hat{\boldsymbol{\theta}})$ and $\hat{u}_{j|\mathrm{S}} := C_{j|\mathrm{S}}(u_j \mid \mathbf{u}_S; \hat{\boldsymbol{\theta}})$ using a sample size of n.

Let $\mathbf{X} = (X_1, \dots, X_d)^\top$ be a d-dimensional random vector with marginal distribution functions F_j for X_j, $j = 1, \dots, d$. A sample of size n from $\mathbf{X}$ is given by

$$\mathbf{x} := (\mathbf{x}_1^\top, \dots, \mathbf{x}_n^\top), \text{ where } \mathbf{x}_k := (x_{k,1}, \dots, x_{k,d})^\top \text{ for } k = 1, \dots, n.$$

Proposition 3.2.7. *Two random variables X_i and X_j are conditionally independent given a random vector $\mathbf{X}_{\mathrm{S}}$, where $S \subseteq \{1, \dots, d\} \setminus \{i, j\}$, if and only if $C_{ij;\mathrm{S}}$ is the independence copula.*

Proof. Note that $C_{ij;\mathrm{S}}$ is the copula corresponding to the bivariate distribution of (X_i, X_j) given $\mathbf{X}_{\mathrm{S}}$. By Sklar's Theorem, it holds that

$$F_{i,j|\mathrm{S}}(x_i, x_j \mid \mathbf{x}_\mathrm{s}) = C_{ij;\mathrm{S}}(F_{i|\mathrm{S}}(x_i \mid \mathbf{x}_\mathrm{s}), F_{j|\mathrm{S}}(x_j \mid \mathbf{x}_\mathrm{s}); \mathbf{x}_\mathrm{s}).$$

"$\Longrightarrow$" Assume that $X_i \perp\!\!\!\perp X_j \mid \mathbf{X}_{\mathrm{S}}$ holds. Then,

$$F_{i,j|\mathrm{S}}(x_i, x_j \mid \mathbf{x}_\mathrm{s}) = F_{i|\mathrm{S}}(x_i \mid \mathbf{x}_\mathrm{s}) \cdot F_{j|\mathrm{S}}(x_j \mid \mathbf{x}_\mathrm{s}), \quad \text{for all } x_i \in \mathbb{D}_{F_i}, x_j \in \mathbb{D}_{F_j}, \tag{3.8}$$

where $\mathbb{D}_{F_k}$ denotes the domain of F_k, $k = i, j$. Since the range of $F_{k|\mathrm{S}}$ is $[0, 1]$, by Sklar's Theorem, it must hold that $C_{ij;\mathrm{S}}(u_1, u_2; \mathbf{x}_\mathrm{s}) = u_1 \cdot u_2$, for all $u_1, u_2 \in [0, 1]$, which is by definition the independence copula.
"$\Longleftarrow$" Conversely, assume that $C_{ij;\mathrm{S}}$ is the independence copula, i.e.,

$$C_{ij;\mathrm{S}}(u_1, u_2; \mathbf{x}_\mathrm{s}) = u_1 \cdot u_2, \quad \text{for all } u_1, u_2 \in [0, 1].$$

Setting $u_1 = F_{i|\mathrm{S}}(x_i \mid \mathbf{x}_\mathrm{s})$ and $u_2 = F_{j|\mathrm{S}}(x_j \mid \mathbf{x}_\mathrm{s})$ and using Sklar's Theorem, it is evident that Equation (3.8) holds. □

Note that the proof of Proposition 3.8 assumes the use of non-simplified conditional copulas for Sklar's Theorem to apply. However, throughout the applications in this thesis, a simplifying assumption is employed. Under this assumption, Proposition 3.8 is only approximately valid. Therefore, to assess whether $X_i \perp\!\!\!\perp X_j \mid \mathbf{X}_{\mathrm{S}}$ holds, it is still reasonable to evaluate if the simplified $C_{ij;\mathrm{S}}$ is close to the independence copula.

Fitting a Y-vine with responses (X_i, X_j) and covariates $\mathbf{X}_{\mathrm{S}}$ is particularly appealing because it allows for a symmetric treatment of the responses. Additionally, the sequential one-step ahead estimation method attempts to order the covariates to maximize the explanatory power of the model. Generally, the copula $C_{ij;\mathrm{S}}$ is obtained in the last tree of the Y-vine.

However, the algorithm also includes a variable selection. Consider the case where $k - 1$ variables from S have been added as covariates to the model. The candidate set $\mathrm{S}^{(k)}$ now contains $|\mathrm{S}| - (k - 1)$ remaining variables. Assume that for any $l_k \in \mathrm{S}^{(k)}$, the copulas $C_{V_j, U_{l_k}; \mathrm{U}_{l_1:l_{k-1}}}$, $j = 1, 2$, from Equation (3.7) are estimated as independence copulas, where $\{l_1, \dots, l_{k-1}\} = \mathrm{S} \setminus \mathrm{S}^{(k)}$. If this occurs, the algorithm stops and no additional variables are added. In this case, the copula in the last tree of the fitted Y-vine is given by $C_{i,j;\mathrm{S} \setminus \mathrm{S}^{(k)}}$, resulting in a reduced conditioning set. The question then arises: can it still be assumed that $X_i \perp\!\!\!\perp X_j \mid \mathbf{X}_{\mathrm{S}}$ holds if $C_{i,j;\mathrm{S} \setminus \mathrm{S}^{(k)}}$ is estimated as the independence copula? The answer is affirmative, as will be demonstrated now.

Proposition 3.2.8. *Let* $\mathrm{S} \subseteq \{1,\dots,d\} \setminus \{i,j\}$ *for any* $i,j \in \{1,\dots,d\}$. *Further, let* $\mathrm{S}^{(k)} \subseteq \mathrm{S}$, *where* $|\mathrm{S}^{(k)}| = |\mathrm{S}| - (k-1)$, *and* $1 \leq k \leq |\mathrm{S}| + 1$. *If, for all* $l_k \in \mathrm{S}^{(k)}$, *the copulas* $C_{i,l_k;\mathrm{S}\setminus\mathrm{S}^{(k)}}$, $C_{j,l_k;\mathrm{S}\setminus\mathrm{S}^{(k)}}$, *and* $C_{i,j;\mathrm{S}\setminus\mathrm{S}^{(k)}}$ *are independence copulas, then* $X_i \perp\!\!\!\perp X_j \mid \mathbf{X}_\mathrm{S}$ *holds.*

Proof. By Proposition 3.2.7 and the fact that $C_{i,l_k;\mathrm{S}\setminus\mathrm{S}^{(k)}}$ and $C_{j,l_k;\mathrm{S}\setminus\mathrm{S}^{(k)}}$ are independence copulas, it follows that $X_i \perp\!\!\!\perp X_{l_k} \mid \mathbf{X}_{\mathrm{S}\setminus\mathrm{S}^{(k)}}$ and $X_j \perp\!\!\!\perp X_{l_k} \mid \mathbf{X}_{\mathrm{S}\setminus\mathrm{S}^{(k)}}$ for all $l_k \in \mathrm{S}^{(k)}$. Therefore, it holds that

$$(X_i, X_j) \perp\!\!\!\perp \mathbf{X}_{\mathrm{S}^{(k)}} \mid \mathbf{X}_{\mathrm{S}\setminus\mathrm{S}^{(k)}}. \tag{3.9}$$

Additionally, by using Proposition 3.2.7 and the fact that $C_{i,j;\mathrm{S}\setminus\mathrm{S}^{(k)}}$ is the independence copula, it follows that

$$X_i \perp\!\!\!\perp X_j \mid \mathbf{X}_{\mathrm{S}\setminus\mathrm{S}^{(k)}}. \tag{3.10}$$

To show that $X_i \perp\!\!\!\perp X_j \mid \mathbf{X}_\mathrm{S}$ holds, consider:

$$\begin{aligned}
\mathbb{P}(X_i, X_j \mid \mathbf{X}_\mathrm{S}) &= \mathbb{P}(X_i, X_j \mid \mathbf{X}_{\mathrm{S}^{(\mathrm{k})}}, \mathbf{X}_{\mathrm{S}\setminus\mathrm{S}^{(k)}}) \\
&\overset{(3.9)}{=} \mathbb{P}(X_i, X_j \mid \mathbf{X}_{\mathrm{S}\setminus\mathrm{S}^{(k)}}) \\
&\overset{(3.10)}{=} \mathbb{P}(X_i \mid \mathbf{X}_{\mathrm{S}\setminus\mathrm{S}^{(k)}}) \cdot \mathbb{P}(X_j \mid \mathbf{X}_{\mathrm{S}\setminus\mathrm{S}^{(k)}}) \\
&\overset{(3.9)}{=} \mathbb{P}(X_i \mid \mathbf{X}_{\mathrm{S}\setminus\mathrm{S}^{(k)}}, \mathbf{X}_{\mathrm{S}^{(\mathrm{k})}}) \cdot \mathbb{P}(X_j \mid \mathbf{X}_{\mathrm{S}\setminus\mathrm{S}^{(k)}}, \mathbf{X}_{\mathrm{S}^{(\mathrm{k})}}) \\
&= \mathbb{P}(X_i \mid \mathbf{X}_\mathrm{S}) \cdot \mathbb{P}(X_j \mid \mathbf{X}_\mathrm{S})
\end{aligned}$$

□

Proposition 3.2.8 confirms that the variable selection process does not interfere with the method of testing for conditional independencies by examining whether the copula in the last tree corresponds to the independence copula. This is because the conditioning set is reduced only by excluding variables that are not found to influence any response variables. Furthermore, when integrated into the PC algorithm, a reduction of the conditioning set is uncommon. The PC algorithm tests for conditional independencies in increasing order, and when edges are removed, subsequent conditioning sets are automatically reduced. For example, if the PC algorithm accurately identifies the conditional independencies in Equation (3.9), then instead of testing $X_i \perp\!\!\!\perp X_j \mid \mathbf{X}_\mathrm{S}$ later, only $X_i \perp\!\!\!\perp X_j \mid \mathbf{X}_{\mathrm{S}\setminus\mathrm{S}^{(k)}}$ will be tested.

To formally define the Y-vine-based conditional independence test, assume there are n i.i.d. observations from $\mathbf{X} = (X_1,\dots,X_d)^\top$ available. For distinct $i,j \in \{1,\dots,d\}$, let $\mathbf{y}_1 = (x_{k,i})_{k=1,\dots,n}$ and $\mathbf{y}_2 = (x_{k,j})_{k=1,\dots,n}$ be realizations from (X_i, X_j). Additionally, $\mathbf{x}_\mathrm{S} = (\mathbf{x}_{l_1}^\top, \dots, \mathbf{x}_{l_p}^\top)$ represents the observations from $\mathbf{X}_\mathrm{S}$, where $\mathrm{S} = \{l_1,\dots,l_p\} \subseteq \{1,\dots,d\} \setminus \{i,j\}$. For now, assume that $|\mathrm{S}| \geq 1$ since Y-vines are undefined for the case of zero covariates. Using kernel density estimation, the observations are transformed to the u-level: $\hat{\mathbf{v}}_1 := \hat{F}_i(x_{k,i})_{k=1,\dots,n}$ and $\hat{\mathbf{v}}_2 := \hat{F}_j(x_{k,j})_{k=1,\dots,n}$, where $\hat{F}_i$ and $\hat{F}_j$ are the estimated marginal distributions of X_i and X_j. A similar transformation is performed for the covariates: $\hat{\mathbf{u}}_{l_m} := \hat{F}_{l_m}(x_{k,l_m})_{k=1,\dots,n}$ for $m = 1,\dots,p$, and $\hat{\mathbf{u}}_\mathrm{S} := (\hat{\mathbf{u}}_{l_1}^\top, \dots, \hat{\mathbf{u}}_{l_p}^\top)$, where $\hat{F}_{l_m}$ is the estimated marginal distribution of X_{l_m}. A Y-vine is then fitted to the pseudo-copula data, with response variables (V_1, V_2) and covariates $\mathbf{U}_\mathrm{S}$, by sequentially adding variables to the covariate order and maximizing the adjusted conditional log-likelihood as described in Section 3.2.2. Denote by $\tilde{\mathrm{S}} \subseteq \mathrm{S}$ the potentially reduced conditioning set after the algorithm terminates. The copula in the final tree is then given by

$$C_{V_1V_2;\mathbf{U}_{\tilde{\mathrm{S}}}}\big(\hat{C}_{V_1|\mathbf{U}_{\tilde{\mathrm{S}}}}(v_1 \mid \mathbf{u}_{\tilde{\mathrm{S}}}), \hat{C}_{V_2|\mathbf{U}_{\tilde{\mathrm{S}}}}(v_2 \mid \mathbf{u}_{\tilde{\mathrm{S}}}); \hat{\mathcal{F}}_{V_1V_2;\mathbf{U}_{\tilde{\mathrm{S}}}}, \hat{\theta}_{V_1V_2;\mathbf{U}_{\tilde{\mathrm{S}}}}\big).$$

The estimated Kendall's τ of the copula in the final tree depends on the estimated copula family and parameters and is denoted by

$$\hat{\tau}^{\text{last}} := \tau(\hat{\theta}_{V_1V_2;\mathbf{U}_{\tilde{\mathrm{S}}}}; \hat{\mathcal{F}}_{V_1V_2;\mathbf{U}_{\tilde{\mathrm{S}}}}). \tag{3.11}$$

Definition 3.2.9 (Y-vine-based conditional independence test)**.** Consider the hypotheses

$$H_0 : X_i \perp\!\!\!\perp X_j \mid \mathbf{X}_\mathrm{S} \quad \text{vs.} \quad H_1 : X_i \not\perp\!\!\!\perp X_j \mid \mathbf{X}_\mathrm{S},$$

where $\mathbf{X}_S \neq \emptyset$ (or, equivalently $|S| \geq 1$). The null hypothesis can be tested against its alternative using the following **Y-vine-based conditional independence test**:

Reject H_0 vs. H_1 at level $k \in (0, 1)$ if and only if

$$|\hat{\tau}^{\text{last}}| > k,$$

where $\hat{\tau}^{\text{last}}$ is the Kendall's τ of the last copula in a fitted Y-vine given by Equation (3.11), and k is a tuning parameter.

As outlined in the definition, the test decision does not rely on asymptotic results; instead, it uses a simple upper bound k on the absolute value of the estimated Kendall's τ. Nagler et al. (2019) used a similar threshold on Kendall's τ for the construction of sparse regular vine copulas which they termed *thresholded vine copulas*. Notably, only $\hat{\tau}^{\text{last}} = 0$ corresponds to the independence copula. However, focusing solely on independence copulas may be overly restrictive and could result in failing to detect many underlying conditional independencies. By increasing the upper bound k, the test is more likely to accept H_0, leading to more conditional independencies being identified. In the context of the PC algorithm, a larger k would lead to more edges being removed, resulting in sparser Bayesian network structures. Therefore, k serves as a tuning parameter for controlling the sparsity of the network structure.

In contrast to a simple upper bound, Bauer and Czado (2016) utilized an asymptotic result for Kendall's τ. Specifically, under the null hypothesis of independence between two random variables X and Y, it is known that

$$\sqrt{\frac{9n(n-1)}{2(2n+5)}}\hat{\tau}_n(X,Y) \xrightarrow{d} N(0,1) \quad \text{as } n \to \infty.$$

The variance of $\hat{\tau}_n(X,Y)$ is derived in Valz and McLeod (1990) and its asymptotic normality is established by the results in Hoeffding (1948). However, since the pseudo-copula data used to estimate $\hat{\tau}^{\text{last}}$ results from multiple estimation steps, the asymptotic normality may not hold even with large sample sizes. Additionally, due to the inherent flexibility of the Y-vine estimation procedure, reliance on asymptotic results becomes less critical, making a straightforward upper bound more appropriate. Therefore, independence testing based on the asymptotic properties of Kendall's τ will be applied only to ordinary independence cases, as Y-vines cannot be defined in such instances.

Definition 3.2.10 (Ordinary independence test based on Kendall's τ)**.** Let $\hat{\tau}_n$ be an estimate of Kendall's τ, assuming no ties, based on the observations $(\hat{F}_i(x_{k,i}), \hat{F}_j(x_{k,j}))$, $k = 1, \ldots, n$, as defined in Definition 2.3.5. Consider the hypotheses

$$H_0 : X_i \perp\!\!\!\perp X_j \quad \text{vs.} \quad H_1 : X_i \not\!\perp\!\!\!\perp X_j.$$

The null hypothesis can be tested against its alternative using the following **ordinary independence test based on Kendall's τ** (Hollander et al. 2014, p. 367):

Reject H_0 vs. H_1 at level $\alpha \in (0, 1)$ if and only if

$$\sqrt{\frac{9n(n-1)}{2(2n+5)}} \cdot \hat{\tau}_n > \Phi^{-1}(1-\alpha/2),$$

where Φ^{-1} denotes the quantile function of the standard normal distribution, α is the significance level, and n is the number of observations.

The estimation of Kendall's τ is based on the observations transformed to the u-level using kernel density estimation, which facilitates computations in R. Since Kendall's τ is rank-based and invariant to marginal transformations, this approach does not affect its validity. It is also crucial to distinguish that α represents the significance level and should not be confused with the tuning parameter k used in Y-vine-based conditional independence testing. Therefore, when using both tests within the PC algorithm, two separate parameters need to be chosen. For simplicity, the combination of the ordinary independence test based on Kendall's τ and the Y-vine-based conditional independence test for non-empty conditioning sets will be referred to as the *Y-test*.

4 Pair-Copula Bayesian Networks

Bayesian networks are powerful graphical models that are frequently applied in a broad range of areas. In the continuous case, modeling has mainly been limited to Gaussian Bayesian networks, as introduced in Section 2.2.3. In a Gaussian Bayesian network all conditional distributions can be derived analytically and the equivalence to a joint multivariate normal distribution facilitates tasks such as statistical inference and model simulation. However, restricting to the multivariate normal distribution lacks the flexibility to incorporate often observed features such as heavy-tailedness, tail dependence, or nonlinear and asymmetric dependence. Kurowicka and Cooke (2005) therefore first introduced a copula based approach to model Bayesian networks. They subsequently extended their method to allow for faster updating (Hanea et al. 2006) and mixed continuous and discrete data (Hanea and Kurowicka 2008). Their approach focuses on copula families with the property of zero rank correlation implying independence, and non-parametric inference. Extending the idea, Bauer et al. (2011) showed that every pdf corresponding to a continuous Bayesian network can be decomposed into univariate marginal distributions and pair-copulas. Hereby, each pair-copula is associated with a specific edge in the underlying DAG and no family restrictions are made. They termed this type of model *pair-copula Bayesian network* and showed it suitability to parametric likelihood inference.

4.1 Model Framework

Let $\mathbf{X} = (X_1, \dots, X_d)^\top$ be a d-dimensional random vector with probability distribution $\mathcal{P}$ and let $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ be a Bayesian network of $\mathcal{P}$. The goal is to derive a pair-copula decomposition of the pdf corresponding to $\mathcal{G}$, i.e., a pair-copula decomposition of

$$f(\mathbf{x}) = \prod_{v \in \mathcal{V}} f_{v|pa(v)}(x_v \mid \mathbf{x}_{pa(v)}).$$

Example 4.1.1 (Example 2.3.1 continued)**.** Consider the Bayesian network depicted in Figure 4.1. The corresponding pdf is given by

$$f(\mathbf{x}) = f_1(x_1) \cdot f_{2|1}(x_2 \mid x_1) \cdot f_{3|12}(x_3 \mid x_1, x_2). \tag{4.1}$$

A pair-copula decomposition of (4.1) was derived in Example 2.3.1, and is expressed as

$$f(\mathbf{x}) = c_{13;2}(F_{1|2}(x_1 \mid x_2), F_{3|2}(x_3 \mid x_2); x_2) \cdot c_{23}(F_2(x_2), F_3(x_3)) \cdot c_{12}(F_1(x_1), F_2(x_2)) \cdot \prod_{i=1}^{3} f_i(x_i).$$

The decomposition reveals that each edge in the Bayesian network can be associated with a specific pair-copula: c_{12} corresponds to the edge $1 \longrightarrow 2$, c_{23} to the edge $2 \longrightarrow 3$, and $c_{13;2}$ to the edge $1 \longrightarrow 3$. Figure 4.2a graphically illustrates this decomposition by labeling the copula subscripts on the corresponding edges.

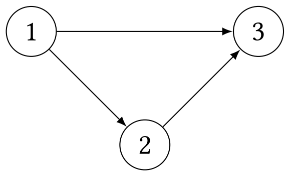


Figure 4.1 A three dimensional Bayesian network corresponding to a D-vine

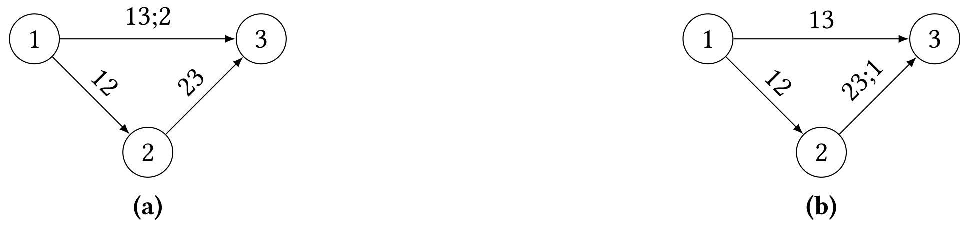


Figure 4.2 Two different pair-copula Bayesian networks associated with the DAG in Figure 4.1

As previously mentioned, the decomposition is not unique. For instance, by following the same reasoning as in Example 2.3.1, but choosing to derive $f_{3|12}$ by computing $f_{23|1}$ instead of $f_{13|2}$, the pdf can alternatively be decomposed as

$$f(\mathbf{x}) = c_{23;1}(F_{2|1}(x_2 \mid x_1), F_{3|1}(x_3 \mid x_1); x_1) \cdot c_{13}(F_1(x_1), F_3(x_3)) \cdot c_{12}(F_1(x_1), F_2(x_2)) \cdot \prod_{i=1}^{3} f_i(x_i).$$

This alternative decomposition is graphically represented in Figure 4.2b. While both decompositions are mathematically equivalent, they correspond to different D-vines: Figure 4.2a corresponds to a D-vine with order $3 - 2 - 1$ and Figure 4.2b corresponds to a D-vine with order $3 - 1 - 2$. Importantly, under the simplifying assumption, the estimators obtained from these decompositions through statistical inference can differ significantly.

Example 4.1.1 demonstrates that to fully specify a Bayesian network using pair-copulas, it is essential to define which copulas are employed in the decomposition of the pdf. Furthermore, it raises the question of whether every Bayesian network defined through its pair-copulas is equivalent to an R-vine. As will be shown later, this is generally not the case.

Notably, the decomposition of $f_{v|pa(v)}$ for any node $v \in \mathcal{V}$ is not unique if $|pa(v)| > 1$. To clarify which copulas are used in the decomposition, an ordering of the parents for each node with multiple parents will now be introduced.

Definition 4.1.1 (Parent order). Let $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ be a Bayesian network. The **parent order** of a node $v \in \mathcal{V}$ is a strict total order $<_v$ on the parents of v. The set $\mathcal{O} := \{<_v \mid v \in \mathcal{V}\}$ is called a **set of parent orderings of $\mathcal{G}$**. Moreover, for all $v \in \mathcal{V}$ and $w \in pa(v)$,

$$pa(v; w) := \{u \in pa(v) \mid u <_v w\}$$

is the **set of parents of v that have a lower order than w**.

The following theorem is adapted from Bauer (2013). It establishes the fundamental connection between the set of parent orderings and the decomposition of the pdf of a Bayesian network.

Theorem 4.1.2. *Let $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ be a Bayesian network of $\mathcal{P}$, and define $d := |\mathcal{V}|$. Then $\mathcal{P}$ is uniquely determined by its univariate margins and the pair copulas $C_{v,w;pa(v;w)}$, where $v \in \mathcal{V}$ and $w \in pa(v)$. The pdf of $\mathcal{P}$ is given by*

$$f(\mathbf{x}) = \prod_{v \in \mathcal{V}} f_v(x_v) \prod_{w \in pa(v)} c_{v,w;pa(v;w)}\big(F_{v|pa(v;w)}(x_v \mid \mathbf{x}_{pa(v;w)}), F_{w|pa(v;w)}(x_w \mid \mathbf{x}_{pa(v;w)}) \mid \mathbf{x}_{pa(v;w)}\big), \tag{4.2}$$

where $\mathbf{x} = (x_v)_{v \in \mathcal{V}} \in \mathbb{R}^d$.

Proof. The proof proceeds by induction and relies solely on graph-theoretical considerations. For $d = 1$, the claim is trivial. Now consider the case where $d > 1$. Since $\mathcal{G}$ is acyclic, there exists a *maximal vertex* in $\mathcal{G}$, i.e., a $m \in \mathcal{V}$ with $de(m) = \emptyset$. Define $\mathcal{V}' := \mathcal{V} \setminus \{m\}$ and $\mathcal{E}' := \mathcal{E} \cap (\mathcal{V}' \times \mathcal{V}')$. Since $\mathcal{G}$ is a Bayesian network of $\mathcal{P}$, it factorizes according to

$$\begin{aligned} f(\mathbf{x}) &= \prod_{v \in \mathcal{V}} f_{v|pa(v)}(x_v \mid \mathbf{x}_{pa(v)}) \\ &= f_{m|pa(m)}(x_m \mid \mathbf{x}_{pa(m)}) \prod_{v \in \mathcal{V}'} f_{v|pa(v)}(x_v \mid \mathbf{x}_{pa(v)}). \end{aligned}$$

Since m is a maximal vertex in $\mathcal{G}$, the sets $pa(v)$ and $nd(v)$ remain unchanged for all $v \in \mathcal{V}'$. Therefore, the probability distribution $\mathcal{P}' := \mathcal{P}\ |_{\mathcal{V}'}$ satisfies the local MP with respect to the DAG $\mathcal{G}' = (\mathcal{V}', \mathcal{E}')$. Consequently, $\mathcal{G}'$ factorizes as

$$f_{\mathcal{V}'}(\mathbf{x}_{\mathcal{V}'}) = \prod_{v \in \mathcal{V}'} f_{v|pa(v)}(x_v \mid \mathbf{x}_{pa(v)}).$$

Assume the decomposition (4.2) holds for the Bayesian network $\mathcal{G}'$. It remains to show that including the marginal density f_m and the pair-copula densities $c_{m,w\,;\,pa(m;w)}$, $w \in pa(m)$, yields a unique decomposition of the Bayesian network $\mathcal{G}$.

Let $k := |pa(m)|$. For $k = 0$, the claim is trivial. For $k > 0$, let $w_1 <_m \cdots <_m w_k$ denote the elements of $pa(m)$ and let $W := pa(m; w_k)$. By Sklar's Theorem and the fact that $W = pa(m) \setminus \{w_k\}$, it holds that:

$$\begin{aligned} f_{m|pa(m)}(x_m \mid \mathbf{x}_{pa(m)}) &= \frac{f_{\{m\}\cup pa(m)}(\mathbf{x}_{\{m\}\cup pa(m)})}{f_W(\mathbf{x}_W)} \cdot \frac{f_W(\mathbf{x}_W)}{f_{pa(m)}(\mathbf{x}_{pa(m)})} \\ &= \frac{f_{m,w_k|W}(x_m, x_{w_k} \mid \mathbf{x}_W)}{f_{w_k|W}(x_{w_k} \mid \mathbf{x}_W)} \\ &= c_{m,w_k;W}\big(F_{m|W}(x_m \mid \mathbf{x}_W), F_{w_k|W}(x_{w_k} \mid \mathbf{x}_W) \mid \mathbf{x}_W\big) \cdot f_{m|W}(x_m \mid \mathbf{x}_W). \end{aligned}$$

Since $pa(m) \subseteq \mathcal{V}'$, the conditional cdf $F_{w_k|W}(\cdot \mid \mathbf{x}_W)$ is completely determined by $\mathcal{P}'$. Furthermore, $f_{m|W}$ can be iteratively computed using the same reasoning. For instance, if $W_2 := pa(m; w_{k-1})$, it holds that

$$f_{m|W}(x_m \mid \mathbf{x}_W) = c_{m,w_{k-1};W_2}\big(F_{m|W_2}(x_m \mid \mathbf{x}_{W_2}), F_{w_{k-1}|W_2}(x_{w_{k-1}} \mid \mathbf{x}_{W_2}) \mid \mathbf{x}_{W_2}\big) \cdot f_{m|W_2}(x_m \mid \mathbf{x}_{W_2}).$$

Finally, for $W_k := pa(m; w_1) = \emptyset$,

$$f_{m|W_{k-1}}(x_m \mid \mathbf{x}_{W_{k-1}}) = c_{m,w_1}\big(F_m(x_m), F_{w_1}(x_{w_1})\big) \cdot f_m(x_m).$$

Combining the results yields

$$\begin{aligned} &f_{m|pa(m)}(x_m \mid \mathbf{x}_{pa(m)}) \\ &\quad = f_m(x_m) \cdot \prod_{w \in pa(m)} c_{m,w;pa(m;w)}\big(F_{m|pa(m;w)}(x_m \mid \mathbf{x}_{pa(m;w)}), F_{w|pa(m;w)}(x_w \mid \mathbf{x}_{pa(m;w)}) \mid \mathbf{x}_{pa(m;w)}\big), \end{aligned}$$

which establishes the claim. □

Definition 4.1.3 (Pair-copula Bayesian network)**.** Let $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ be a Bayesian network of the probability distribution $\mathcal{P}$, and let $\mathcal{O} := \{<_v \mid v \in \mathcal{V}\}$ denote the set of parent orderings of $\mathcal{G}$. For each $v \in \mathcal{V}$ and $w \in pa(v)$, let $pa(v; w) = \{u \in pa(v) \mid u <_v w\}$ be the set of parents of v with lower order than w. For each edge $(w, v) \in \mathcal{E}$, let $C_{v,w;pa(v;w)}$ be the copula associated with that edge. Then, the pair $(\mathcal{G}, \mathcal{O})$ is called a **pair-copula Bayesian network (PCBN)** of $\mathcal{P}$ and the pdf of $\mathcal{P}$ is given by Equation (4.2).

Pair-copula Bayesian networks combine the strengths of causal graphical models in capturing conditional independencies with the flexibility of copulas to model complex, non-linear dependencies independently of marginal distributions. Unlike R-vines, however, PCBNs allow for conditional cdfs in the decomposition of the pdf that cannot be obtained through the recursive computation of h-functions, as described in Lemma 2.3.15.

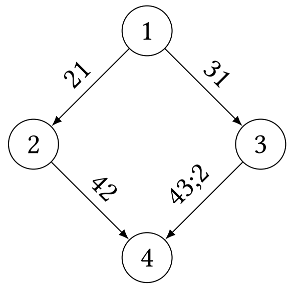


Figure 4.3 A four-dimensional PCBN with parent order $2 <_4 3$

Definition 4.1.4 (Specified pair-copulas). Let $(\mathcal{G}, \mathcal{O})$ be a pair-copula Bayesian network, where $\mathcal{G} = (\mathcal{V}, \mathcal{E})$. For any $v, w \in \mathcal{V}$ and $\mathrm{S} \subseteq \mathcal{V} \setminus \{v, w\}$, the copula $C_{v,w;\mathrm{S}}$ is **specified by the pair-copula Bayesian network** if one of the following conditions hold:

i) $w \in pa(v)$ and $\mathrm{S} = pa(v; w)$.

ii) $v \in pa(w)$ and $\mathrm{S} = pa(w; v)$.

iii) $< \{v\} \mid \mathrm{S} \mid \{w\} >_{\mathcal{G}}$.

It is important to note that Conditions i) and ii) in Definition 4.1.4 correspond to the pair-copulas that appear in the decomposition of the pdf. These copulas will be referred to as **specified by decomposition**. Condition iii) is related to the graphical structure of the underlying DAG. If this condition holds, then X_v and X_w are independent given $\mathbf{X}_\mathrm{S}$, and therefore the copula $C_{v,w;\mathrm{S}}$ is known to be the independence copula. These independence copulas will be referred to as **specified by d-separation**. The question of whether a conditional cdf can only be obtained via integration is directly related to which copulas are specified by a PCBN, as the following example, taken from Bauer (2013), will demonstrate.

Example 4.1.2. Consider the four-dimensional pair-copula Bayesian network shown in Figure 4.3. The implied conditional independence statements are $X_2 \perp\!\!\!\perp X_3 \mid X_1$ and $X_1 \perp\!\!\!\perp X_4 \mid X_2, X_3$. The only node with more than one parent is node 4, and its parent order is $2 <_4 3$. Consequently, the following sets are defined:

$$pa(1; \emptyset) = pa(2; 1) = pa(3; 1) = pa(4; 2) = \emptyset, \quad pa(4; 3) = \{2\}.$$

Applying Equation (4.2) yields the following decomposition of the pdf:

$$\begin{aligned} f(\mathbf{x}) =& \Big(\prod_{i=1}^{4} f_i(x_i) \Big) \cdot c_{21}(F_2(x_2), F_1(x_1)) \cdot c_{31}(F_3(x_3), F_1(x_1)) \cdot c_{42}(F_4(x_4), F_2(x_2)) \\ & \cdot c_{43;2}(F_{4|2}(x_4 \mid x_2), F_{3|2}(x_3 \mid x_2) \mid x_2), \quad \mathbf{x} \in \mathbb{R}^4. \end{aligned}$$

The conditional cdf $F_{4|2}$ can be directly obtained because the copula C_{42} is specified by the PCBN:

$$F_{4|2}(x_4 \mid x_2) = h_{4|2}(F_4(x_4) \mid F_2(x_2)).$$

The conditional cdf $F_{3|2}$, however, cannot be obtained in a similar manner because the copula C_{23} is not specified by the PCBN. This is due to the fact that C_{23} does not appear in the decomposition and node 3 is not d-separated from node 2 given the empty set. Nevertheless, $F_{3|2}$ can be derived through integration, utilizing the fact that $X_3 \perp\!\!\!\perp X_2 \mid X_1$ $(*)$:

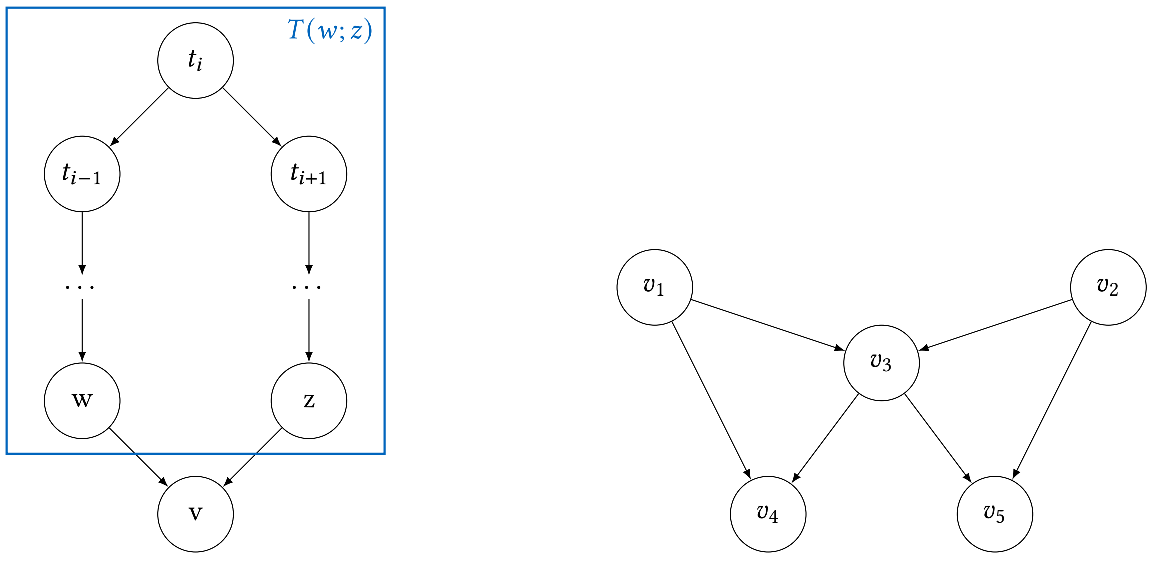
$$\begin{aligned}
f_2(x_2) \cdot f_3(x_3) \cdot c_{23}(F_2(x_2), F_3(x_3)) &= f_{23}(x_2, x_3) \\
&= \int_{\mathbb{R}} f_{123}(x_1, x_2, x_3) \, \mathrm{d}x_1 \\
&= \int_{\mathbb{R}} f_{23|1}(x_2, x_3 \mid x_1) \cdot f_1(x_1) \, \mathrm{d}x_1 \\
&= \int_{\mathbb{R}} f_{2|1}(x_2 \mid x_1) \cdot f_{3|1}(x_3 \mid x_1) \cdot f_1(x_1) \, \mathrm{d}x_1 \qquad (*) \\
&= \int_{\mathbb{R}} f_{21}(x_2, x_1) \cdot f_{31}(x_3, x_1) \cdot f_1^{-1}(x_1) \, \mathrm{d}x_1 \\
&= \int_{\mathbb{R}} f_1(x_1) \cdot f_2(x_2) \cdot f_3(x_3) \cdot c_{21}(F_2(x_2), F_1(x_1)) \cdot c_{31}(F_3(x_3), F_1(x_1)) \, \mathrm{d}x_1 \\
&= f_2(x_2) \cdot f_3(x_3) \cdot \underbrace{\int_{[0,1]} c_{21}(F_2(x_2), u_1) \cdot c_{31}(F_3(x_3), u_1) \, \mathrm{d}u_1}_{c_{23}(F_2(x_2), F_3(x_3))} .
\end{aligned}$$

Hence,

$$\begin{aligned}
F_{3|2}(x_3 \mid x_2) &= \frac{\partial C_{23}(F_2(x_2), F_3(x_3))}{\partial F_2(x_2)} \\
&= \frac{\partial\Big(\int_0^{F_2(x_2)} \int_0^{F_3(x_3)} c_{23}(u_2, u_3) \, \mathrm{d}u_3 \mathrm{d}u_2 \Big)}{\partial F_2(x_2)} \\
&= \int_0^{F_3(x_3)} c_{23}(F_2(x_2), u_3) \, \mathrm{d}u_3 \\
&= \int_0^{F_3(x_3)} \Big(\int_0^1 c_{21}(F_2(x_2), u_1) \cdot c_{31}(u_3, u_1) \, \mathrm{d}u_1 \Big) \mathrm{d}u_3 \\
&= \int_0^1 c_{21}(F_2(x_2), u_1) \cdot \Big(\int_0^{F_3(x_3)} c_{31}(u_3, u_1) \, \mathrm{d}u_3 \Big) \mathrm{d}u_1 \\
&= \int_0^1 c_{21}(F_2(x_2), u_1) \cdot \Big(\int_0^{F_3(x_3)} \frac{\partial C_{31}(u_3, u_1)}{\partial u_3 \partial u_1} \, \mathrm{d}u_3 \Big) \mathrm{d}u_1 \\
&= \int_0^1 c_{21}(F_2(x_2), u_1) \cdot \frac{\partial C_{31}(F_3(x_3), u_1)}{\partial u_1} \, \mathrm{d}u_1 \\
&= \int_0^1 c_{21}(F_2(x_2), u_1) \cdot h_{3|1}(F_3(x_3) \mid u_1) \, \mathrm{d}u_1 .
\end{aligned}$$

The last integral does not generally admit a closed-form solution. Selecting the parent order $3 <_4 2$ instead of $2 <_4 3$ results in a similar outcome, with the roles of nodes 2 and 3 swapped.

Example 4.1.2 demonstrates that not all PCBNs can be represented by R-vines. In this specific four-dimensional case, obtaining the conditional cdf $F_{3|2}$ involves evaluating a single one-dimensional integral, while all other conditional cdfs can be computed recursively using h-functions. Despite this particular case being manageable due to the limited number of integrals, the situation generally becomes more challenging as the dimension of the underlying DAG increases. This complexity can impede exact likelihood inference and simulation from such PCBNs. Moreover, deriving the integral representation of conditional cdfs in terms of specified pair-copulas can be complex, as it requires utilizing specific conditional independence statements implied by the Bayesian network. An algorithm developed by Bauer and Czado (2016) provides a solution by deriving a pair-copula decomposition, which may involve integration, for all conditional cdfs in a PCBN.



(a) Active cycle **(b)** Interfering v-structure

Figure 4.4 Illustration of active cycles and interfering v-structures

Given that the primary difficulty with PCBNs lies in evaluating conditional cdfs, it is pertinent to explore whether certain conditions could eliminate the need for integration. Horsman (2023) identified two graphical structures associated with integration-free PCBNs.

Definition 4.1.5 (Active cycle)**.** Let $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ be a DAG. For $v \in \mathcal{V}$, let $w, z \in pa(v)$ be two distinct parents of v such that there is a trail $T(w; z)$ from w to z satisfying the following conditions:

i) $T(w; z)$ contains no v-structures.

ii) $T(w; z)$ contains no **chords**, meaning there is no edge between two non-consecutive nodes within the trail.

Then, the trail $v \longleftarrow T(w; z) \longrightarrow v$ forms an **active cycle**.

Definition 4.1.6 (Interfering v-structure)**.** Let $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ be a DAG with $|\mathcal{V}| \geq 5$, and let $(v_i)_{i=1,\ldots,5} \in \mathcal{V}$ be five nodes that satisfy the following conditions:

i) $v_3 \in pa(v_4) \cap pa(v_5)$.

ii) $v_1 \in pa(v_3) \cap pa(v_4)$ and $v_1 \notin pa(v_5)$.

iii) $v_2 \in pa(v_3) \cap pa(v_5)$ and $v_2 \notin pa(v_4)$.

Then, the nodes $(v_i)_{i=1,\ldots,5}$ are said to form an **interfering v-structure**.

Figure 4.4 illustrates the graphical structures of active cycles and interfering v-structures. According to Horsman (2023), if a DAG $\mathcal{G}$ does not contain any active cycles or interfering v-structures — referred to as a **restricted DAG** — then there exists a parent order $\mathcal{O}$ such that the joint density of the PCBN $(\mathcal{G}, \mathcal{O})$ can be computed without the need for integration. In contrast, if the DAG contains either an active cycle or an interfering v-structure, then it is impossible to compute the joint density without performing some integration, regardless of the chosen parent order. Additionally, the author provided an algorithm to determine a parent order that ensures integration-free computations for restricted DAGs.

In Example 4.1.2, the trail $4 \longleftarrow 2 \longleftarrow 1 \longrightarrow 3 \longrightarrow 4$ forms an active cycle. This is because $2 \longleftarrow 1 \longrightarrow 3$ is not a v-structure, and there is no edge between nodes 2 and 3. Therefore, the DAG is not restricted, and, as observed, integration is necessary to compute the joint density. It is worth noting that introducing an edge $2 \longrightarrow 3$ could break the active cycle. In this scenario, node 3 would have two parents, necessitating the definition of a parent order for node 3. Setting $1 <_3 2$ specifies the copulas C_{31} and $C_{32;1}$. However, $F_{3|2}$ or $F_{2|3}$ which appear as an argument in either $C_{43;2}$ or $C_{42;3}$, depending on the parent order of node 4, can still not be obtained without integration, as the copula C_{23} is not specified by the PCBN. Thus, only the parent order $2 <_3 1$ results in an integration-free decomposition, as it specifies the pair-copula C_{23}.

This illustrates that both the underlying graphical structure and the chosen parent order influence whether evaluating a joint density in a PCBN requires integration. Relying solely on restricted DAGs and specific parent orders can be problematic, especially in high-dimensional settings. If the true underlying DAG is sparse, it may include many active cycles. Therefore, restricting the search, such as through structure learning, to only restricted DAGs might not yield a DAG that closely approximates the true structure. Moreover, as will be discussed in the next section, in a regression context, the parent order is related to the strength of dependence between covariates and the response variable, as well as among the covariates themselves. Consequently, it may be undesirable to limit the set of possible parent orders in practical applications. For these reasons, this thesis does not assume that PCBNs are restricted in general and will introduce approximate methods for sampling and likelihood inference that do not require numerical integration.

4.2 Simulation Methods

This section will provide two methods for sampling from any pair-copula Bayesian network. The first is an exact method that uses the precise inverse of the conditional distribution for each node given its parents. The exact method may require numerical integration, as generally not all conditional cdfs can be derived from the pair-copulas specified by the PCBN. The second method is new and approximates the joint distribution of each node and its parents using a D-vine structural equation model, where the node is positioned as a leaf in the first tree, followed by its parents in their specified order. This approach is particularly appealing because the conditional distribution and its inverse of the node given its parents are easily obtained. However, as detailed in Section 4.2.2, this method may require the estimation of some copulas that involve only the parents of a node.

4.2.1 Exact Simulation

The aim of the exact simulation method is to generate samples from the exact probability distribution of a pair-copula Bayesian network. The results presented here are adapted from Bauer et al. (2011). Consider a d-dimensional PCBN $(\mathcal{G}, \mathcal{O})$ with node set $\mathcal{V} = \{1, \dots, d\}$, and let a topological order of the nodes be given by $\ell = (l_1, \dots, l_d)$. Define $\ell_{1:i} := (l_1, \dots, l_i)$ as the first i elements of the topological order, for $i = 1, \dots, d$. Assume that all copulas in the PCBN are parametric, with families and parameters denoted by $(\mathcal{F}, \boldsymbol{\theta}) = (\mathcal{F}_{v,w;pa(v;w)}, \theta_{v,w;pa(v;w)})_{v \in \mathcal{V}, w \in pa(v)}$. Although this parametric assumption is not strictly necessary for the simulation routine, in practical applications of this thesis only parametric copulas $C_{v,w;pa(v;w)}$ will be used. For simplicity, the equations will use only the parameter vector, while keeping in mind that the copulas also depend on the chosen family and rotation. To obtain a d-dimensional sample $\mathbf{u}^s = (u^s_{l_1}, \dots, u^s_{l_d}) \in [0,1]^d$ from the PCBN on the u-level, the following steps are performed:

i) Sample $w_i \overset{\text{i.i.d.}}{\sim} U[0,1], \quad i = 1, \dots, d.$

ii) Apply the inverse Rosenblatt transform (see Definition 3.1.3):

$$
\begin{aligned}
u^s_{l_1} &:= w_1 \\
u^s_{l_2} &:= C^{-1}_{l_2|l_1}(w_2 \mid u^s_{l_1}; \boldsymbol{\theta}) \\
u^s_{l_3} &:= C^{-1}_{l_3|l_{1:2}}(w_3 \mid \mathbf{u}^s_{l_{1:2}}; \boldsymbol{\theta}) \\
&\vdots \\
u^s_{l_d} &:= C^{-1}_{l_d|l_{1:(d-1)}}(w_d \mid \mathbf{u}^s_{l_{1:(d-1)}}; \boldsymbol{\theta}).
\end{aligned}
$$

Sampling according to the topological order of the nodes is necessary because, based on the Markov properties of a Bayesian network, the conditioning set for each node can be reduced to only its parents:

$$w_k = C_{l_k|l_{1:(k-1)}}(u^s_{l_k} \mid \mathbf{u}^s_{l_{1:(k-1)}}; \boldsymbol{\theta}) = C_{l_k|pa(l_k)}(u^s_{l_k} \mid \mathbf{u}^s_{pa(l_k)}; \boldsymbol{\theta}), \quad k = 2, \dots, d.$$

Further, assume that $pa(l_k) \neq \emptyset$ and let v be the last parent in the parent order of l_k, meaning $pa(l_k; v) = pa(l_k) \setminus \{v\}$. Then, by applying Lemma 2.3.15 it holds that

$$w_k = h_{l_k|v;pa(l_k;v)}\big(C_{l_k|pa(l_k;v)}(u^s_{l_k} \mid \mathbf{u}^s_{pa(l_k;v)}; \boldsymbol{\theta}) \mid C_{v|pa(l_k;v)}(u^s_v \mid \mathbf{u}^s_{pa(l_k;v)}; \boldsymbol{\theta}); \boldsymbol{\theta}\big), \tag{4.3}$$

where

$$h_{l_k|v;pa(l_k;v)}(u_1 \mid u_2) := \frac{\partial C_{l_k,v;pa(l_k;v)}(u_1, u_2)}{\partial u_2}.$$

In Equation (4.3), only the first argument, $C_{l_k|pa(l_k;v)}$, depends on $u^s_{l_k}$, and it can be determined using the same reasoning as before. Iteratively, the necessary inverse h-functions for sampling are $h^{-1}_{l_k|v^*;pa(l_k,v^*)}$, for $v^* \in pa(l_k)$. However, the second argument in Equation (4.3) may often require integration, as demonstrated in the subsequent example.

Example 4.2.1 (Example 4.1.2 continued)**.** Consider again the four-dimensional PCBN illustrated in Figure 4.3, with the parent order $2 <_4 3$. As shown in Example 4.1.2, the conditional distribution of X_3 given X_2 is expressed as:

$$F_{3|2}(x_3 \mid x_2) = \int_0^1 c_{21}(F_2(x_2), u_1) \cdot h_{3|1}(F_3(x_3) \mid u_1) \, \mathrm{d}u_1.$$

Therefore, the conditional distribution of $U_3 = F_3(X_3)$ given $U_2 = F_2(X_2)$ can be obtained by:

$$C_{3|2}(u_3 \mid u_2) = \int_0^1 c_{21}(u_2, u_1) \cdot h_{3|1}(u_3 \mid u_1) \, \mathrm{d}u_1.$$

Given that $w_1, \dots, w_4$ are independently drawn from the standard uniform distribution, the sampling equations are derived for the topological order $1 < 2 < 3 < 4$ as follows:

$$\begin{aligned} u^s_1 &:= w_1, \\ u^s_2 &:= h^{-1}_{2|1}(w_2 \mid u^s_1; \theta_{21}), \\ u^s_3 &:= h^{-1}_{3|1}(w_3 \mid u^s_1; \theta_{31}). \end{aligned}$$

To sample from U_4 given $U_2 = u^s_2$, $U_3 = u^s_3$, consider the following:

$$\begin{aligned} w_4 &= C_{4|23}(u^s_4 \mid u^s_2, u^s_3; \boldsymbol{\theta}) \\ &= h_{4|3;2}\big(C_{4|2}(u^s_4 \mid u^s_2; \boldsymbol{\theta}) \mid C_{3|2}(u^s_3 \mid u^s_2; \boldsymbol{\theta}); \theta_{43;2}\big) \\ &= h_{4|3;2}\left(h_{4|2}(u^s_4 \mid u^s_2; \theta_{42}) \;\middle|\; \int_0^1 c_{21}(u^s_2, u_1) \cdot h_{3|1}(u^s_3 \mid u_1; \theta_{31}) \, \mathrm{d}u_1; \theta_{43;2}\right). \end{aligned}$$

Inverting this expression leads to the following sampling equation:

$$u^s_4 := h^{-1}_{4|2}\left(h^{-1}_{4|3;2}\left(w_4 \;\middle|\; \int_0^1 c_{21}(u^s_2, u_1) \cdot h_{3|1}(u^s_3 \mid u_1; \theta_{31}) \, \mathrm{d}u_1; \theta_{43;2}\right) \;\middle|\; u^s_2; \theta_{42}\right).$$

Example 4.2.1 demonstrates that, due to the presence of an active cycle, numerical integration is required to simulate from the exact distribution of U_4 given its parents.

4.2.2 Approximate Simulation Using D-Vines

This section introduces a novel approach to approximately simulate from the distribution of any pair-copula Bayesian network will be presented. The core idea is to model the joint distribution of a node and its parents using a D-vine. The structure of the D-vine can be designed to specify the same copulas as those in the underlying PCBN, as demonstrated in the following proposition.

Proposition 4.2.1. *Let $(\mathcal{G}, \mathcal{O})$ be a PCBN with node set $\mathcal{V} = \{1, \dots, d\}$. For any node $v \in \mathcal{V}$ with $pa(v) \neq \emptyset$, let $\ell = (l_1, \dots, l_k)$, where $k < d$, present the tuple of the parents of v, with the elements arranged according the parent order $<_v$. Denote the first m elements of ℓ as $\ell_{1:m} := (l_1, \dots, l_m)$ for $1 \leq m \leq k$. Then, the simplified conditional pdf of $U_v = F_v(X_v)$ given $\mathbf{U}_{pa(v)}$ can be expressed as:*

$$\begin{aligned} c_{v|pa(v)}(u_v \mid \mathbf{u}_{pa(v)}) &= \prod_{w \in pa(v)} c_{v,w;pa(v;w)}\big(C_{v|pa(v;w)}(u_v \mid \mathbf{u}_{pa(v;w)}), C_{w|pa(v;w)}(u_w \mid \mathbf{u}_{pa(v;w)})\big) \\ &= c_{v,l_1}(u_v, u_{l_1}) \cdot \prod_{i=2}^{k} c_{v,l_i;\ell_{1:(i-1)}}\big(C_{v|\ell_{1:(i-1)}}(u_v \mid \mathbf{u}_{\ell_{1:(i-1)}}), C_{l_i|\ell_{1:(i-1)}}(u_{l_i} \mid \mathbf{u}_{\ell_{1:(i-1)}})\big) \end{aligned}$$

Specifying a D-vine with structure $v - l_1 - \cdots - l_k$ yields the same decomposition of $c_{v|pa(v)}$.

Proof. The first statement follows from Theorem 4.1.2 and the general factorization of Bayesian networks as expressed in Equation (2.3). To prove the second statement, consider a D-vine with structure $v - l_1 - \cdots - l_k$. In this D-vine, the edges in the first tree are $\{v, l_1\}, \{l_1, l_2\}, \dots, \{l_{k-1}, l_k\}$. The edges in tree j, for $1 < j < k$, are $\{v, l_j; \ell_{1:(j-1)}\}, \{l_1, l_{j+1}; \ell_{2:j}\}, \dots, \{l_{k-j}, l_k; \ell_{(k-j+1):(k-1)}\}$, and the edge in tree k is $\{v, l_k; \ell_{1:(k-1)}\}$. Therefore, the joint density of $(U_v, \mathbf{U}_\ell)$ is given by

$$\begin{aligned} c_{u_v,l_1,\dots,l_k}(v, \mathbf{u}_\ell) = c_{v,l_1}(u_v, u_{l_1}) &\cdot \Big(\prod_{j=2}^{k} c_{v,l_j;\ell_{1:(j-1)}}\big(C_{v|\ell_{1:(i-1)}}(u_v \mid \mathbf{u}_{\ell_{1:(i-1)}}), C_{l_i|\ell_{1:(i-1)}}(u_{l_i} \mid \mathbf{u}_{\ell_{1:(i-1)}})\big)\Big) \\ &\cdot \Big(\prod_{i=1}^{k-1} c_{l_i,l_{i+1}}(u_{l_i}, u_{l_{i+1}})\Big) \\ &\cdot \Big(\prod_{j=2}^{k-1}\prod_{i=1}^{k-j} c_{l_i,l_{i+j};\ell_{(i+1):(i+j-1)}}\big(C_{l_i|\ell_{(i+1):(i+j-1)}}(u_{l_i} \mid \mathbf{u}_{\ell_{(i+1):(i+j-1)}}), C_{l_{i+j}|\ell_{(i+1):(i+j-1)}}(u_{l_{i+j}} \mid \mathbf{u}_{\ell_{(i+1):(i+j-1)}})\big)\Big). \end{aligned}$$

Given that the joint distribution of $\mathbf{U}_\ell$ follows a D-vine with tree structure $l_1 - \cdots - l_k$, the joint pdf of the parents of v is given by

$$\begin{aligned} c_{l_1,\dots,l_k}(\mathbf{u}_\ell) = &\Big(\prod_{i=1}^{k-1} c_{l_i,l_{i+1}}(u_{l_i}, u_{l_{i+1}})\Big) \\ &\cdot \Big(\prod_{j=2}^{k-1}\prod_{i=1}^{k-j} c_{l_i,l_{i+j};\ell_{(i+1):(i+j-1)}}\big(C_{l_i|\ell_{(i+1):(i+j-1)}}(u_{l_i} \mid \mathbf{u}_{\ell_{(i+1):(i+j-1)}}), C_{l_{i+j}|\ell_{(i+1):(i+j-1)}}(u_{l_{i+j}} \mid \mathbf{u}_{\ell_{(i+1):(i+j-1)}})\big)\Big). \end{aligned}$$

Substituting these terms into the following equality yields the result:

$$\begin{aligned} c_{v|pa(v)}(u_v \mid \mathbf{u}_{pa(v)}) &= c_{v|l_1,\dots,l_k}(u_v \mid \mathbf{u}_\ell) \\ &= \frac{c_{v,l_1,\dots,l_k}(u_v, \mathbf{u}_\ell)}{c_{l_1,\dots,l_k}(\mathbf{u}_\ell)}. \end{aligned}$$

□

Proposition 4.2.1 establishes that the conditional distribution of each node given its parents in a PCBN can be modeled using a D-vine, where in the first tree, the node of interest serves as a leaf, followed by

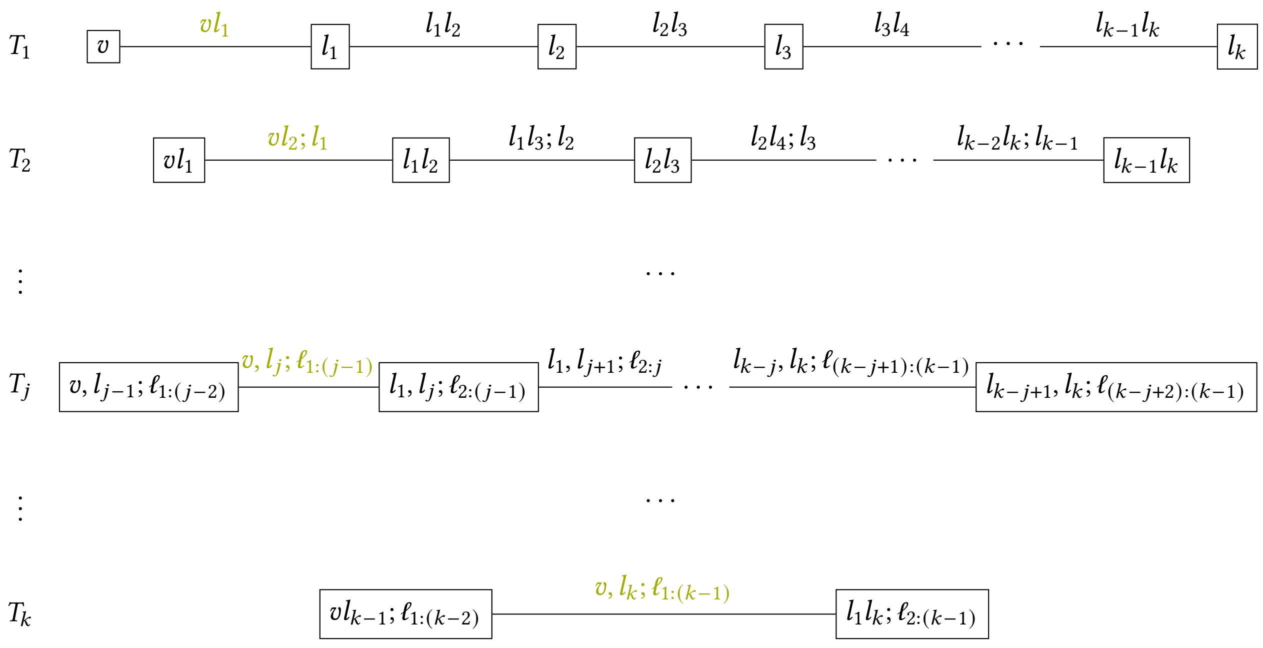


Figure 4.5 D-vine tree sequence with node v as a leaf, followed by its parents in their specified parent order. Edges in green represent copulas specified by the decomposition of the underlying PCBN

its parents in their specified order. The copulas specified by the PCBN's decomposition are represented in the first edge of each tree, as shown in Figure 4.5.

The D-vine formed solely by the parents of a node v, arranged in their parent order, is referred to as the **parental D-vine** of v. Additionally, a model in which the conditional distribution of each node given its parents is derived from a D-vine with this specific structure, combined with the factorization of the underlying Bayesian network as expressed in Equation 2.3, is called a **D-vine structural equation model (DV-SEM)**. If all copulas in the DV-SEM are directly obtained from the PCBN, the DV-SEM and the PCBN are distributionally equivalent. However, depending on the DAG structure and the chosen parent order, not all copulas in the parental D-vines may be specified by the PCBN. If a copula is not specified by the PCBN, it must be obtained through integration. Since samples are generated according to the topological order, a node's parents are always simulated before the node itself. Any unspecified copula will appear only in the parental D-vine and can be estimated from the parent samples, thus avoiding the need for integration.

The approximate simulation procedure can be outlined as follows: Let $(\mathcal{G}, \mathcal{O})$ be a PCBN with node set $\mathcal{V} = \{1, \dots, d\}$ and let $\mathbf{t} = (t_1, \dots, t_d)$ represent a topological order of the nodes. A d-dimensional *approximate* sample $\hat{\mathbf{u}}^s = (\hat{u}_1^s, \dots, \hat{u}_d^s)$ from the PCBN on the u-level is obtained through the following steps:

i) Sample $w_i \overset{\text{i.i.d.}}{\sim} U[0,1], \quad i = 1, \dots, d.$

ii) Utilizing the topological order $\mathbf{t}$:

a) If t_i has no parents, i.e., $pa(t_i) = \emptyset$, $i = 1, \dots, d$, set

$$\hat{u}_{t_i}^s := w_i.$$

b) If t_i does have parents, i.e., $pa(t_i) \neq \emptyset$, $i = 1, \dots, d$, proceed as follows:

1.) Construct a D-vine tree sequence $\mathcal{T}$, where t_i is the leaf node followed by its parents ordered according to their parent order $<_{t_i}$ denoted by $\ell = (l_1, \dots, l_k)$. According to the topological order, it holds that $l_j \in \{t_1, \dots, t_{i-1}\}$ for $j = 1, \dots, k$.

2.) Assign the pair-copulas that correspond to the first edge in each tree according to the decomposition of the PCBN. This set of specified pair-copulas is denoted by $\mathcal{B}^{(spec)}$.

3.) Estimate the families and parameters of the pair-copulas in the parental D-vine using the previously simulated data $(\hat{u}^s_{l_1}, \ldots, \hat{u}^s_{l_k})$. This estimation process follows the general procedure described in Section 2.3.5. The resulting set of estimated bivariate copulas is denoted by $\mathcal{B}^{(est)}$.

4.) Combine the specified and estimated copulas into a single set, denoted by $\mathcal{B} = \mathcal{B}^{(spec)} \cup \mathcal{B}^{(est)}$. Then, using the conditional simulation procedure described in Section 3.1.3, obtain the sample $\hat{u}^s_{t_i}$ from the D-vine copula $(\mathcal{T}, \mathcal{B})$.

Since the copulas in the parental D-vines are estimated rather than directly specified by the PCBN, the DV-SEM serves as an approximation of the underlying PCBN. However, unlike the exact simulation from the PCBN, sampling from the DV-SEM does not require numerical integration, as demonstrated in the following example.

Example 4.2.2 (Example 4.1.2 continued)**.** Consider again the four-dimensional PCBN depicted in Figure 4.3, with the parent order $2 <_4 3$. Let $w_{ij} \sim U[0, 1]$, for $i = 1, \ldots, n$ and $j = 1, \ldots, 4$ be an i.i.d. sample of size n from the standard uniform distribution. Using the topological order $1 < 2 < 3 < 4$, the approximate simulation proceeds with the following steps:

- **Node 1:** $pa(1) = \emptyset$. Therefore, $\hat{u}_{i1} = w_{i1}$ for $i = 1, \ldots, n$.
- **Node 2:** $pa(2) = \{1\}$. Define a D-vine tree sequence with the structure $2 - 1$. Since node 2 has only one parent, the D-vine consists of a single edge, specified by decomposition of the PCBN. No estimation is required, and the conditional simulation procedure described in Section 3.1.3 yields the following for $i = 1, \ldots, n$:
  i) Apply the Rosenblatt transform:
  $$\begin{aligned} u_{i1} &\coloneqq C_1(\hat{u}_{i1}) = \hat{u}_{i1}, \\ u_{i2} &\coloneqq h_{2|1}(w_{i2} \mid \hat{u}_{i1}; \theta_{21}). \end{aligned}$$
  ii) Independently sample z_{i2} from the standard uniform distribution.
  iii) Apply the inverse Rosenblatt transform:
  $$\begin{aligned} v_{i1} &\coloneqq C_1^{-1}(u_{i1}) = \hat{u}_{i1}, \\ v_{i2} &\coloneqq h^{-1}_{2|1}(z_{i2} \mid u_{i1}; \theta_{21}) = h^{-1}_{2|1}(z_{i2} \mid \hat{u}_{i1}; \theta_{21}). \end{aligned}$$
  iv) Set $\hat{u}_{i2} \coloneqq v_{i2}$.
- **Node 3:** $pa(3) = \{1\}$. Similarly to node 2, define the D-vine tree sequence $3 - 1$. For $i = 1, \ldots, n$:
  i) Apply the Rosenblatt transform:
  $$\begin{aligned} u_{i1} &\coloneqq C_1(\hat{u}_{i1}) = \hat{u}_{i1}, \\ u_{i2} &\coloneqq h_{3|1}(w_{i3} \mid \hat{u}_{i1}; \theta_{31}). \end{aligned}$$
  ii) Independently sample z_{i3} from the standard uniform distribution.
  iii) Apply the inverse Rosenblatt transform:
  $$\begin{aligned} v_{i1} &\coloneqq C_1^{-1}(u_{i1}) = \hat{u}_{i1}, \\ v_{i2} &\coloneqq h^{-1}_{3|1}(z_{i3} \mid u_{i1}; \theta_{31}) = h^{-1}_{3|1}(z_{i3} \mid \hat{u}_{i1}; \theta_{31}). \end{aligned}$$
  iv) Set $\hat{u}_{i3} \coloneqq v_{i2}$.

Edge	Family	Rotation	Parameters	τ
21	Gaussian	0°	-0.84	-0.64
31	Gumbel	90°	1.27	-0.21
42	Gumbel	0°	3.26	0.69
43;2	Joe	180°	2.03	0.36

Table 4.1 A selection of copulas specifying the PCBN displayed in Figure 4.3

Edge	Family	Rotation	Parameters	τ
32	BB1	180°	(0.07, 1.14)	0.16

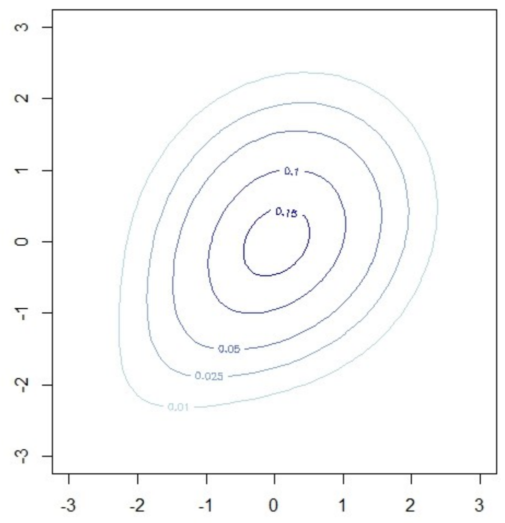
Table 4.2 Estimated $\hat{C}_{32}$ based on $n = 1000$ samples from the PCBN specified by the copulas in Table 4.1

- **Node 4:** $pa(4) = \{2, 3\}$. Since the parent order is given by $2 <_4 3$, define a D-vine tree sequence with the structure $4 - 2 - 3$. The edges in the first tree are $\{42, 23\}$, and in the second tree, $\{43; 2\}$. The copulas C_{42} and $C_{43;2}$ are specified by decomposition, while the copula C_{23} in the parental D-vine (with structure $2 - 3$) is estimated using the procedure described in Section 2.3.5. The conditional simulation method then yields the following:
  i) Apply the Rosenblatt transform:
$$\begin{aligned} u_{i1} &:= C_3(\hat{u}_{i3}) = \hat{u}_{i3}, \\ u_{i2} &:= h_{2|3}(\hat{u}_{i2} \mid \hat{u}_{i3}; \hat{\theta}_{23}) \\ u_{i3} &:= C_{4|23}(w_{i4} \mid \hat{u}_{i2}, \hat{u}_{i3}) \\ &= h_{4|3;2}(h_{4|2}(w_{i4} \mid \hat{u}_{i2}; \theta_{42}) \mid h_{3|2}(\hat{u}_{i3} \mid \hat{u}_{i2}; \hat{\theta}_{23}); \theta_{43;2}). \end{aligned}$$
  ii) Independently sample z_{i4} from the standard uniform distribution.
  iii) Apply the inverse Rosenblatt transform:
$$\begin{aligned} v_{i1} &:= C_3^{-1}(u_{i1}) = \hat{u}_{i3}, \\ v_{i2} &:= h_{2|3}^{-1}(u_{i2} \mid u_{i1}; \hat{\theta}_{23}) = h_{2|3}^{-1}(h_{2|3}(\hat{u}_{i2} \mid \hat{u}_{i3}; \hat{\theta}_{23}) \mid \hat{u}_{i3}; \hat{\theta}_{23}) = \hat{u}_{i2}, \\ v_{i3} &:= h_{4|2}^{-1}\Big(h_{4|3;2}^{-1}(z_{i4} \mid h_{3|2}(\hat{u}_{i3} \mid u_{i2}; \hat{\theta}_{23}); \theta_{43;2}) \Bigm| u_{i2}; \theta_{42}\Big) \end{aligned}$$
  iv) Set $\hat{u}_{i4} := v_{i3}$.

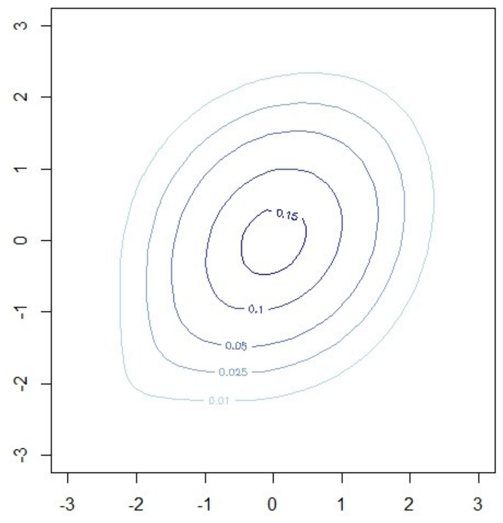
Comparing the approximate simulation in Example 4.2.2 to the exact method shown in Example 4.2.1 reveals that the sampling procedures differ only at node 4. In the approximate simulation, the random variables (U_1, U_2, U_3) are sampled from their exact joint distribution. If the same random seed is used in both procedures, then it holds that $\hat{u}^s_{ij} = u^s_{ij}$ for $i = 1, \ldots, n$ and $j = 1, 2, 3$. More generally, the approximate and exact simulation methods produce identical results for a random variable U_v if $|pa(v)| \leq 1$, as the parental D-vine is undefined in these cases, eliminating the need for estimation.

The approximate simulation method leverages the fact that samples of (U_2, U_3) are generated before sampling U_4. Based on a sample of size n, the copula C_{23} is estimated, and the resulting h-function $h_{3|2}(\cdot, \cdot; \hat{\theta}_{23})$ is used instead of the integral required in the exact simulation method. Consequently, in general, $\hat{u}^s_{i4} \neq u^s_{i4}$, even if the same seed is used in both procedures.

To illustrate, consider the specification provided in Table 4.1. Based on a sample of size $n = 1000$, an estimate of C_{32}, denoted by $\hat{C}_{32}$, is given in Table 4.2. The true copula C_{32} can only be obtained through integration, as described in Example 2.3.2. A comparison between the true C_{32} and the estimated $\hat{C}_{32}$ for the specific case considered is shown in Figure 4.6. The figure indicates that both copulas are highly similar, exhibiting only lower tail dependence.



(a) True C_{32} obtained via integration



(b) Estimated $\hat{C}_{32}$ as given in Table 4.2

Figure 4.6 Comparison between the normalized contour plots of the true C_{32} and the estimated $\hat{C}_{32}$ using a DV-SEM modeling approach

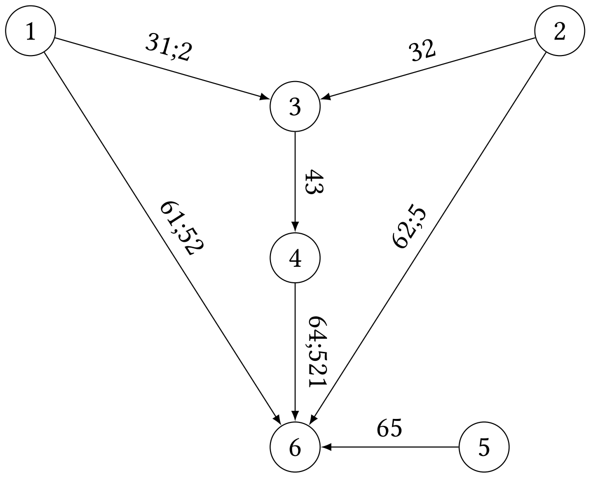
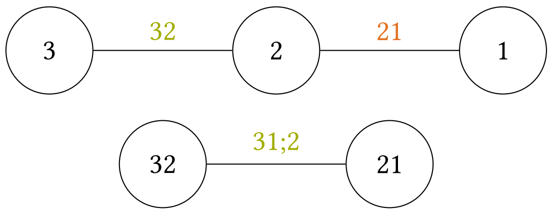


Figure 4.7 A six-dimensional PCBN with parent orders $2 <_3 1$ and $5 <_6 2 <_6 1 <_6 4$

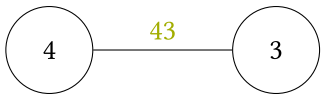
The approximate simulation method estimates all copulas involved in the parental D-vine of each node $v \in \mathcal{V}$ with $|pa(v)| > 1$. It is noteworthy that the number of parameters requiring estimation can be reduced since some of these pair-copulas might be specified by d-separation and are thus known to be independence copulas, as demonstrated in the following example:

Example 4.2.3. Consider the six-dimensional PCBN with the parent orders $2 <_3 1$ and $5 <_6 2 <_6 1 <_6 4$, as illustrated in Figure 4.7. The underlying DAG and its 45 implied conditional independencies were previously presented in Example 2.2.4 to demonstrate the PC algorithm's functionality. Due to the active cycles $6 \leftarrow 1 \rightarrow 3 \rightarrow 4 \rightarrow 6$ and $6 \leftarrow 2 \rightarrow 3 \rightarrow 4 \rightarrow 6$, the copula density of the PCBN cannot be obtained without integration. A topological order of the underlying DAG is $5 < 2 < 1 < 3 < 4 < 6$. The only nodes with no parents are nodes 1, 2, and 5. Consequently, the DV-SEM associated with this PCBN is represented by the following three D-vines:

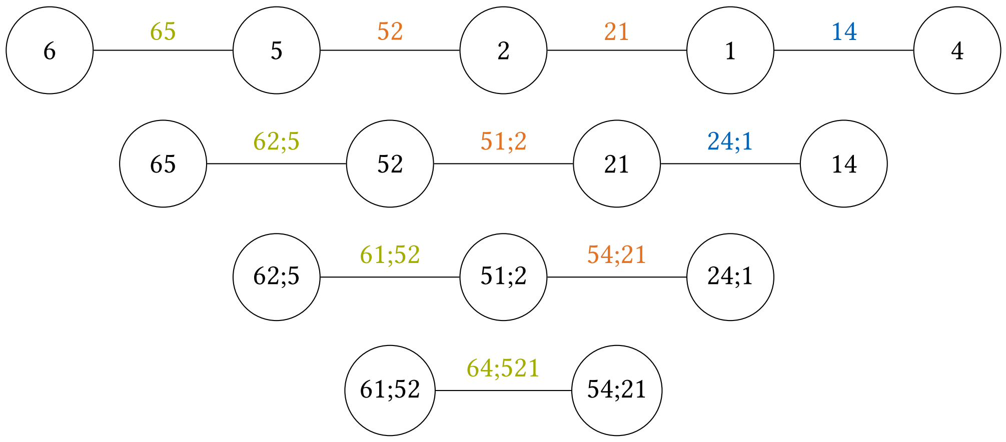
- **Node 3:**



- **Node 4:**



- **Node 6:**



The edges highlighted in green represent copulas that are directly specified by decomposition of the PCBN. Edges marked in orange correspond to copulas specified by d-separation, and thus they are associated with independence copulas. Only the pair-copulas linked to the blue edges remain unspecified.

Example 4.2.3 suggests a potential improvement for the approximate simulation procedure described in this section. Currently, the procedure estimates all parental D-vines in full, including copulas associated with edges marked in orange and blue. However, to avoid integration, it is sufficient to estimate only the unspecified copulas, corresponding to the edges marked in blue, while the others can be set to independence copulas based on d-separation. This approach can significantly reduce the number of parameters to be estimated. For instance, in the example provided, the number of copula parameters required would be reduced from seven to two.

At present, there is no available implementation in R that supports partial vine estimation — where some copulas in a vine tree sequence are specified by the user and others are estimated. Manually estimating the unspecified copulas is computationally challenging in higher dimensions, as it relies on pseudo-observations computed from preceding trees. Consequently, in this thesis, all samples based on the approximate method will involve estimating both, the copulas specified by d-separation and the unspecified pair-copulas. Future developments should aim to implement algorithms for partial D-vine estimation, as this could enhance the accuracy of approximation methods based on D-vine structural equation models.

To assess whether the estimation of the parental D-vines aligns with the implied conditional independencies, consider the orange-colored edges in Example 4.2.3, which correspond to the copulas $C_{21}, C_{52}, C_{51;2}$, and $C_{54;21}$. Note that C_{21} appears in both the D-vine of node 3 and the D-vine of node 6. Since the estimation of C_{21} in both cases is based on the same samples $(\hat{u}^s_{i2}, \hat{u}^s_{i1})$, $i = 1, \ldots, n$, the results will be identical. In general, if the same copula appears in different D-vines corresponding to different nodes, it will be estimated consistently across all D-vines, as the estimation relies on the same sample data.

Quantile	$\lvert\hat{\tau}_{21}\rvert$	$\lvert\hat{\tau}_{52}\rvert$	$\lvert\hat{\tau}_{51;2}\rvert$	$\lvert\hat{\tau}_{54;21}\rvert$
50%	0	0	0	0
62.5%	0	0.0156	0	0.0124
75%	0.0190	0.0277	0.0223	0.0261
87.5%	0.0314	0.0324	0.0307	0.0352
95%	0.0396	0.0394	0.0377	0.0396
Max	0.0558	0.0520	0.0578	0.0565

Table 4.3 Quantiles of the estimated absolute τ-values for copulas specified by d-separation in the DV-SEM from Example 4.2.3

To verify that the four copulas specified by d-separation are indeed independence copulas, the following simulation setup was used: For $N = 100$ repetitions and a sample size of $n = 1000$, the copulas specified by decomposition (colored in green) were randomly selected from the set of one-parameter copulas, with random rotations and τ-values independently drawn from the uniform distribution over the interval $(-0.9, -0.1) \cup (0.1, 0.9)$. For each of the $N = 100$ different specifications, samples were generated using the approximate simulation method, and all parental D-vines were estimated in full. Table 4.3 shows the quantiles of the estimated absolute τ-values for the copulas $C_{21}, C_{52}, C_{51;2}$, and $C_{54;21}$. The results reveal that the quantiles are highly consistent across all four copulas. Additionally, the independence copula was correctly identified in approximately 60% of the cases. In instances where the independence copula was not identified, the estimated τ-value was always less than 0.06 in absolute terms, which supports the robustness of the procedure.

4.3 Maximum Likelihood Estimation and Parameter Learning

Let $\mathbf{x} = (\mathbf{x}_1^\top, \ldots, \mathbf{x}_d^\top)$ be an i.i.d. sample of size n from the random vector $(X_1, \ldots, X_d)^\top$, where $\mathbf{x}_k = (x_{k,1}, \ldots, x_{k,d})^\top$ for $k = 1, \ldots, n$. Further, let $(\mathcal{G}, \mathcal{O})$ be a PCBN with node set $\mathcal{V} = \{1, \ldots, d\}$ specifying the pair-copulas $\{C_{v,w;pa(v;w)}(\cdot, \cdot\,; \boldsymbol{\theta}) \mid v \in \mathcal{V}, w \in pa(v), \boldsymbol{\theta} \in \Theta\}$ by decomposition.

Definition 4.3.1 (Exact log-likelihood)**.** Following Equation (4.2), the **exact log-likelihood** function takes the form

$$\begin{aligned}\ell(\boldsymbol{\theta}; \mathbf{x}) = &\sum_{i=1}^{n} \sum_{j=1}^{d} f_j(x_{ij}) \qquad (4.4)\\ &+ \sum_{i=1}^{n} \sum_{j=1}^{d} \sum_{w \in pa(j)} \ln\Big(c_{j,w;pa(j;w)}\big(F_{j|pa(j;w)}(x_{ij} \mid \mathbf{x}_{i,pa(j;w)}; \boldsymbol{\theta}), F_{w|pa(j;w)}(x_{iw} \mid \mathbf{x}_{i,pa(j;w)}; \boldsymbol{\theta}); \boldsymbol{\theta}\big)\Big).\end{aligned}$$

The first line on the right-hand side of Equation (4.4) is known as the **marginal log-likelihood** and the second part is referred to as the **copula log-likelihood**.

Similar to R-vines, parameter estimation for a PCBN can be performed using maximum likelihood estimation (MLE). The process involves first iterating over the nodes and their parents in both the topological and parental order to obtain sequential ML estimates. These preliminary estimates are then used to infer the joint ML estimates, denoted by $\hat{\boldsymbol{\theta}}_{v,w;pa(v;w)}$, as described by Bauer (2013). Unlike R-vines, the main challenge with PCBNs often lies in evaluating the conditional distribution functions, which frequently require numerical integration for their computation.

Example 4.3.1 (Example 4.1.2 continued)**.** The exact copula log-likelihood of the PCBN in Example 4.1.2 is given by

$$\ell(\boldsymbol{\theta};\mathbf{u}) = \sum_{i=1}^{n}\Bigg[\ln\big(c_{21}(u_{i2},u_{i1};\theta_{21})\big) + \ln\big(c_{31}(u_{i3},u_{i1};\theta_{31})\big) + \ln\big(c_{42}(u_{i4},u_{i2};\theta_{42})\big) + \ln\bigg(c_{43;2}\Big(h_{4|2}(u_{i4}\mid u_{i2};\theta_{42}), \int_0^1 c_{21}(u_{i2},w_1;\theta_{21})\cdot h_{3|1}(u_{i3}\mid w_1;\theta_{31})\,\mathrm{d}w_1;\theta_{43;2}\Big)\bigg)\Bigg].$$

In higher-dimensional settings, evaluating the exact log-likelihood function for each possible choice of bivariate copulas often requires computing several, potentially multi-dimensional integrals, which is computationally intensive. An alternative approach is needed.

As discussed in Section 4.2.2, one such alternative is to approximate the PCBN using a DV-SEM, which allows for an integration-free estimation of the log-likelihood function. Let $\mathcal{D} = \{(\mathcal{T}_v, \mathcal{B}_v)_{v\in\mathcal{V}}\,;\, \boldsymbol{\theta}^{(DV)} \in \Theta, \mathcal{O}\}$ represent a DV-SEM corresponding to the PCBN $(\mathcal{G}, \mathcal{O})$. Here, $\mathcal{D}$ is a collection of D-vines where, for each node $v \in \mathcal{V}$ with $|pa(v)| > 0$, the structure $\mathcal{T}_v$ is defined as the node v followed by its parents $pa(v)$ in their parent order as specified in $\mathcal{O}$. The bivariate copulas $\mathcal{B}_v$ in each D-vine are associated with a subset of the parameters $\boldsymbol{\theta}^{(DV)}$. These parameters differ from those of the pair-copulas specified by the decomposition of the PCBN, denoted by $\boldsymbol{\theta}$ in Equation (4.4), as $\boldsymbol{\theta}^{(DV)}$ also includes the parameters of the pair-copulas in the parental D-vines, denoted by $\hat{\boldsymbol{\theta}}^{(parDV)}$. Thus, it holds that

$$\boldsymbol{\theta}^{(DV)} = \boldsymbol{\theta} \cup \hat{\boldsymbol{\theta}}^{(parDV)}.$$

Therefore, the DV-SEM approximation introduces $|\hat{\boldsymbol{\theta}}^{(parDV)}|$ additional degrees of freedom to the model. This number of parameters can potentially be reduced by incorporating the independence information implied by d-separation and by employing partial D-vine estimation, which remains an area for further research.

Definition 4.3.2 (Approximate log-likelihood)**.** Following Equation (4.2), the **approximate log-likelihood** function takes the form

$$\hat{\ell}(\boldsymbol{\theta}^{(DV)};\mathbf{x}) = \sum_{i=1}^{n}\sum_{j=1}^{d} f_j(x_{ij}) + \sum_{i=1}^{n}\sum_{j=1}^{d}\sum_{w\in pa(j)} \ln\Big(c_{j,w;pa(j;w)}\big(F_{j|pa(j;w)}(x_{ij}\mid \mathbf{x}_{i,pa(j;w)};\boldsymbol{\theta}^{(DV)}), F_{w|pa(j;w)}(x_{iw}\mid \mathbf{x}_{i,pa(j;w)};\boldsymbol{\theta}^{(parDV)});\theta_{j,w;pa(j;w)}\big)\Big). \tag{4.5}$$

To differentiate between the true log-likelihood function ℓ and its approximate counterpart, the notation $\hat{\ell}$ is used. The DV-SEM facilitates integration-free computation of conditional distribution functions because $\boldsymbol{\theta}^{(DV)}$ includes parameter estimates for copulas that are not specified by the PCBN.

Example 4.3.2 (Example 4.1.2 continued)**.** The approximate copula log-likelihood of the PCBN in Example 4.1.2 is given by

$$\hat{\ell}(\boldsymbol{\theta}^{(DV)};\mathbf{u}) = \sum_{i=1}^{n}\Bigg[\ln\big(c_{21}(u_{i2},u_{i1};\theta_{21})\big) + \ln\big(c_{31}(u_{i3},u_{i1};\theta_{31})\big) + \ln\big(c_{42}(u_{i4},u_{i2};\theta_{42})\big) + \ln\bigg(c_{43;2}\Big(h_{4|2}(u_{i4}\mid u_{i2};\theta_{42}), h_{3|2}(u_{i3}\mid u_{i2};\hat{\theta}_{23});\theta_{43;2}\Big)\bigg)\Bigg].$$

In this case, one extra parameter $\hat{\boldsymbol{\theta}}^{(parD)} = \hat{\theta}_{23}$ needs to be estimated.

Assuming a DV-SEM, the estimation of the families and parameters for the pair-copulas specified by the decomposition of the PCBN becomes straightforward. For a given DAG $\mathcal{G} = (\mathcal{V}, \mathcal{O})$ and parent orders $\mathcal{O}$, one can fit a D-vine for each node $v \in \mathcal{V}$. In this D-vine, the node v is a leaf node in the first tree,

Edge	Family	Rotation	Parameters	τ
21	Gaussian	0°	-0.92	-0.74
31	Frank	0°	-5.14	-0.47
42	Frank	0°	1.03	0.11
43;2	Joe	0°	8.07	0.78

Table 4.4 A further selection of copulas specifying the PCBN displayed in Figure 4.3

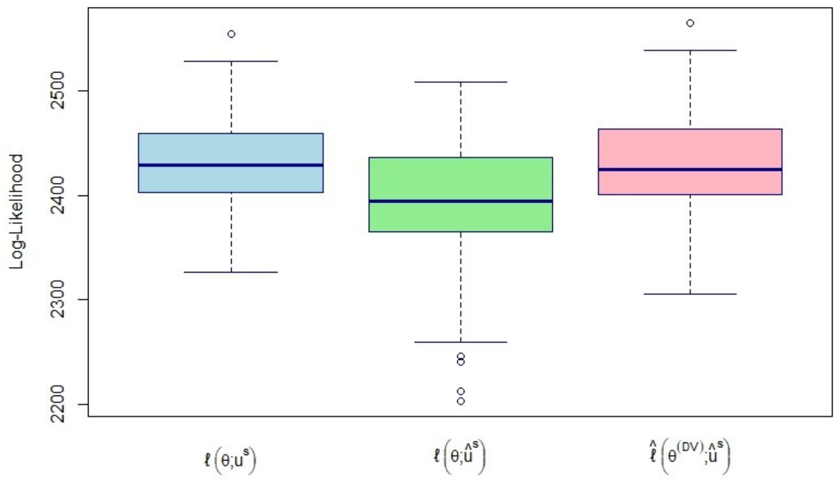


Figure 4.8 Comparison of the log-likelihood based on simulations from the PCBN defined by the pair-copulas in Table 4.4. Here, ℓ denotes the exact log-likelihood function, $\mathbf{u}^s$ is the exactly simulated data, $\hat{\ell}$ denotes the approximate log-likelihood function, and $\hat{\mathbf{u}}^s$ is the approximately simulated data

followed by its parents in their parent order as specified in $\mathcal{O}$. The families and parameters of the pair-copulas are estimated using the general procedure for R-vines as described in Section 2.3.5. The families and parameters of the pair-copulas specified by the PCBN decomposition are then represented by the first edge in each tree of each estimated D-vine.

To illustrate the difference between the approximate and exact log-likelihood function, consider the setup shown in Table 4.4. For each of the $N = 100$ repetitions, two datasets on the u-level were simulated: One using exact simulation and the other using approximate simulation, with a sample size of $n = 1000$. To ensure comparability, the same seed was used to generate both datasets, so the only difference between them lies in node 4; all other nodes were simulated exactly.

Figure 4.8 presents box plots of the following three log-likelihoods from left to right:

i) Exact log-likelihood ℓ computed from the exactly simulated data $\mathbf{u}^s$.

ii) Exact log-likelihood ℓ computed from the approximately simulated data $\hat{\mathbf{u}}^s$.

iii) Approximate log-likelihood $\hat{\ell}$ derived using a DV-SEM approach applied to the approximately simulated data $\hat{\mathbf{u}}^s$.

The figure demonstrates that the approximately simulated data $\hat{\mathbf{u}}^s$ has a lower true log-likelihood compared to the exactly simulated data $\mathbf{u}^s$. Specifically, the average exact log-likelihood for $\hat{\mathbf{u}}^s$ was approximately 2394.85, while for $\mathbf{u}^s$ it was 2433.72. This indicates that, on average, the approximate simulation reduced the log-likelihood by about 1.6%. Although the third case cannot be directly compared to the first two due to differences between $\hat{\ell}$ and ℓ, the figure shows that the quantiles of the log-likelihood are very close to each other. This suggests that estimating the log-likelihood function using a DV-SEM is a reasonable approach.

4.4 Selecting Parent Orders

In the previous section, it was assumed that the set of parent orders $\mathcal{O}$ was known, allowing for the straightforward construction of a DV-SEM by building D-vines according to this order. In practice, however, $\mathcal{O}$ is often an unknown parameter that needs to be estimated from the data. Bauer (2013) proposed a method for sequentially determining each parent order $<_v$, for $v \in \mathcal{V}$, by assigning weights to the edges $w \to v$, where $w \in pa(v)$. These weights could be, for example, the absolute values of Kendall's τ estimates or AIC/BIC values for selected pair-copula families. The parent order is then selected based on these weights in descending order.

This thesis introduces a new approach for sequentially estimating the parent orders and parameters of a PCBN by using a DV-SEM approximation and the sequential D-vine estimation method described in Section 3.1.2: Let $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ be a DAG. For each node $v \in \mathcal{V}$ with $|pa(v)| \geq 1$, perform a univariate D-vine regression with v as the response and $pa(v)$ as the covariates. Specifically, during the regression procedure, a D-vine is constructed sequentially, with v as a leaf node and the covariates from $pa(v)$ are added to the order by maximizing the conditional log-likelihood at each step as first proposed by Kraus and Czado (2017).

By applying this regression method to each node $v \in \mathcal{V}$ with $|pa(v)| \geq 1$, one obtains a DV-SEM that corresponds to the PCBN. The estimated parent orders are given by the orders derived from the D-vines. The parameters of the copulas specified by decomposition are represented by the estimated copulas in the first edge of each tree within each D-vine. Therefore, performing D-vine-based regression on each node of a fully specified DAG can be utilized to estimate the parameters of the PCBN. Additionally, incorporating a two-step ahead forward-looking procedure, as proposed by Tepegjozova et al. (2022), could potentially enhance the estimation method.

5 Simulation Study: Y-Vine-Based Structure Learning

In this chapter, simulation studies are conducted to assess the performance of the Y-vine-based PC algorithm in comparison to its benchmark, which utilizes Fisher's Z-test of partial correlation. These studies are carried out on simulated data from various PCBNs, including both non-Gaussian and fully Gaussian specifications. Bauer (2013) performed a similar study, employing R-vine-based conditional independence tests with different vine structures and asymptotic tests on pseudo-observations for structure learning within the PC algorithm. However, the analysis was confined to different specifications of a PCBN based on the same four-dimensional DAG shown in Figure 4.3, with comparisons of the fitted CPDAGs to the true CPDAG made only in a graph-theoretical sense.

This thesis extends existing research by incorporating six- and eleven-dimensional PCBNs. The newly introduced Y-test, which utilizes an upper bound on Kendall's τ, along with the Z-test for partial correlations, is applied within the PC algorithm to estimate the CPDAG based on simulated data. Additionally, rather than solely comparing the structure of the fitted CPDAGs, these models are extended with DV-SEM-based parameter estimation to fully specified PCBNs. Finally, the fitted PCBNs are evaluated against the true underlying PCBN using probabilistic measures such as the log-likelihood, AIC, and BIC.

5.1 Study Design

Figure 5.1 outlines the steps involved in performing the simulation studies. The procedure is as follows:

1) **Definition of the true PCBN:** A fixed PCBN $(\mathcal{G}^{(\text{True})}, \mathcal{O}^{(\text{True})})$, denoted as PCBN$^{(\text{True})}$, is defined by specifying:
   i) A DAG $\mathcal{G}^{(\text{True})} = (\mathcal{V}, \mathcal{E}^{(\text{True})})$, where $\mathcal{V} = \{1, \dots, d\}$. The cases considered include $d = 4, 6, 11$.
   ii) A set of parent orders $\mathcal{O}^{(\text{True})}$.
   iii) The copulas $C_{v,w;pa(v;w)}$ for $v \in \mathcal{V}$ and $w \in pa(v)$, specified by decomposition. If all pair-copulas are Gaussian, the PCBN$^{(\text{True})}$ is referred to as a **fully Gaussian setup**; otherwise, it is termed a **non-Gaussian setup**.

2) **DV-SEM Approximation:** The PCBN from Step 1 is approximated by a DV-SEM. This involves defining a separate D-vine for each $v \in \mathcal{V}$ with $pa(v) > 0$. Each D-vine structure is specified by the node v followed by its parents in their parent order as defined in $\mathcal{O}^{(\text{True})}$. The resulting DV-SEM is denoted as DV-SEM$^{(\text{True})}$.

3) **Simulation of u-data:** $n = 1000$ observations on the u-level are simulated from the DV-SEM$^{(\text{True})}$, corresponding to the approximate simulation method described in Section 4.2.2. Each parental D-vine is fully estimated based on the simulated data, with bivariate families and their parameters selected by the lowest AIC-value. In a non-Gaussian setup, the estimation procedure considers all parametric families defined in Appendix B; in a Gaussian setup, it is limited to the Gaussian family. The simulation is repeated $N = 100$ times (or $N = 50$ times in the 11-dimensional case), with the resulting approximately simulated data denoted by $\hat{\mathbf{u}}^s$. The superscript s indicates that the data was simulated. For simplicity, the notation does not explicitly show the dependence of the dataset on each specific repetition.

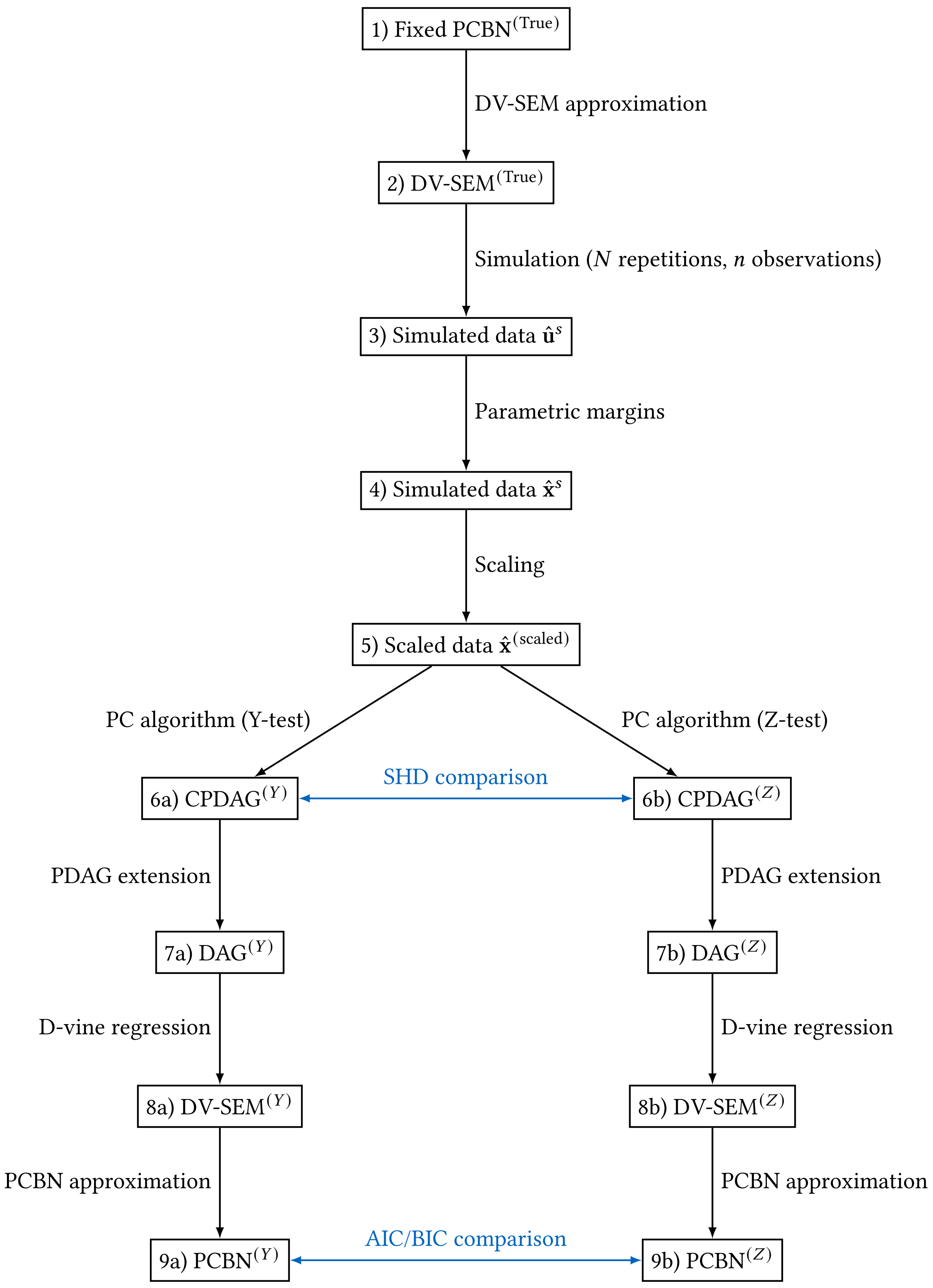


Figure 5.1 Simplified flowchart of the simulation study design

4) **Transformation to the x-level:** The simulated data is transformed to the x-level using the inverse probability integral transform, $\hat{x}_{ij}^{s} := F_j^{-1}(\hat{u}_{ij}^{s})$ for $i = 1, \dots, n$ and $j = 1, \dots, d$. For each $j = 1, \dots, d$, a parametric distribution F_j is randomly selected if the setup is non-Gaussian from the following:
   - Normal distribution with $\mu = \sigma = 1$.
   - Student's t-distribution with $\nu = 5$.
   - Beta distribution with shape parameters $\alpha = \beta = 0.5$.
   - Gamma distribution with shape $k = 2$ and scale $\theta = 2$.
   - Lognormal distribution with $\mu = 0$ and $\sigma = 1$.

   In the Gaussian setup, each parametric margin is set to the Normal distribution with $\mu = \sigma = 1$, making PCBN$^{(\text{True})}$ a Gaussian Bayesian network.

5) **Scaling the Data:** Before learning the structure, the data is scaled as follows:
$$\hat{\mathbf{x}}_{ij}^{(\text{scaled})} := \frac{\hat{x}_{ij}^{s} - \hat{\mu}_j}{\hat{\sigma}_j},$$
   for $i = 1, \dots, n$ and $j = 1, \dots, d$, where
$$\hat{\mu}_j := \frac{1}{n}\sum_{i=1}^{n} \hat{x}_{ij}^{s}, \quad \text{and} \quad \hat{\sigma}_j := \sqrt{\frac{1}{n-1}\sum_{i=1}^{n} (\hat{x}_{ij}^{s})^2}.$$

6) **Structure Learning:** Two versions of the PC algorithm, as stated in Algorithm 2 and 3, are applied to the observations $\hat{\mathbf{x}}^{(\text{scaled})}$. These versions differ only in the conditional independence test used:
   a) Y-vine-based conditional independence test (Y-test) as described in Definitions 3.2.9 and 3.2.10, using the adjusted conditional log-likelihood (acll) for selection. The value of the significance level α^Y, for ordinary independence testing, and the tuning parameter k^Y, which serves as an upper bound for the Kendall's τ in the last tree of the fitted Y-vine in cases involving non-empty conditioning sets, are set to 0.05. A detailed explanation for the parameter choice is given in Section 5.3.
   b) Fisher's Z-test of the partial correlation (Z-test) as described in Definition 2.2.30. The value of the significance level α^Z is set to 0.05. The PC algorithm using Fisher's Z-test serves as a benchmark.

   As a result, two (possibly identical) CPDAGs, denoted by CPDAG$^{(Y)}$ and CPDAG$^{(Z)}$, are obtained. The following steps are applied separately to each fitted CPDAG.

7) **PDAG Extension:** The CPDAGs from Step 6 are extended to DAGs using the PDAG extension described in Section 2.2.6. The two fitted DAGs are denoted by DAG$^{(Y)}$ and DAG$^{(Z)}$ and are given by $\mathcal{G}^{(Y)} = (\mathcal{V}, \mathcal{E}^{(Y)})$ and $\mathcal{G}^{(Z)} = (\mathcal{V}, \mathcal{E}^{(Z)})$.

8) **Construction of DV-SEMs:** Using DAG$^{(Y)}$ and DAG$^{(Z)}$, the DV-SEMs DV-SEM$^{(Y)}$ and DV-SEM$^{(Z)}$ are constructed by performing univariate D-vine-based regression as described in Section 3.1.2. For each DAG and each node $v \in \mathcal{V}$, a D-vine is built sequentially by setting v as the response and $pa(v)$ as the covariates, where $pa(\cdot)$ depends on the specific DAG. The conditional log-likelihood (cll) is used as the selection criterion. If the underlying DAGs are identical, i.e., $\mathcal{E}^{(Y)} = \mathcal{E}^{(Z)}$, only a single DV-SEM is estimated for computational efficiency.

9) **Translation into PCBNs:** The DV-SEMs from Step 8 are translated into PCBNs as follows: For each DV-SEM, the parent order of each node $v \in \mathcal{V}$ is given by the order of the parents in the respective D-vine. Thus, the sets of parent orders $\mathcal{O}^{(Y)}$ and $\mathcal{O}^{(Z)}$ are derived from the respective D-vine structures. Additionally, the copulas in the decomposition of the PCBN are defined by the first edge of each tree in each D-vine. These are denoted by $C_{v,w;pa(v;w)}^{(Y)}$ and $C_{v,w;pa(v;w)}^{(Z)}$, where $w \in pa(v)$, and $pa(\cdot)$ and $pa(\cdot;\cdot)$ are functions depending on the estimated edge set and parent order.

The simulation study is conducted across four different setups for each dimension $d \in \{4, 6, 11\}$. Thus, the procedure outlined in Figure 5.1 is repeated 12 times, with $N = 100$ repetitions (or $N = 50$ in the 11-dimensional case) and $n = 1000$ observations. The setups considered are as follows:

i) **Mixture of Elliptical and Archimedean Copulas:** The copulas selected in Step 1 of the procedure include the elliptical and Archimedean families listed in Appendix B. For the Student's t copula, which is the only copula with two parameters, the degrees of freedom are set to $\nu = 5$. The τ-values are chosen from the interval $(-0.9, -0.1) \cup (0.1, 0.9)$, and the corresponding parameters are computed using Kendall's τ inversion.

ii) **Only Archimedean Copulas:** The copulas selected in Step 1 of the procedure include only the Archimedean families listed in Appendix B. The τ-values and parameters are chosen as in Setup i).

iii) **Only Elliptical Copulas:** The copulas selected in Step 1 of the procedure include only the elliptical families listed in Appendix B. For the Student's t copula, the degrees of freedom are again set to $\nu = 5$. The τ-values and parameters are chosen as in Setup i).

iv) **Only Gaussian Copulas:** The copulas selected in Step 1 of the procedure include only the Gaussian family listed in Appendix B. The τ-values and parameters are chosen as in Setup i).

Specifications i) to iii) represent non-Gaussian setups, while Specification iv) represents a Gaussian setup. In the first three setups, the family set of copulas that are part of the parental D-vines and require estimation includes all copula families (including the BB copulas) listed in Appendix B. In the fourth setup, the family set is limited to Gaussian copulas. In all setups, the D-vines are fully parametric, but the marginal distributions are estimated non-parametrically using kernel density estimation.

5.2 Performance Measures

To assess the fit of the models generated by the two different conditional independence tests within the PC algorithm, two distinct measures of fit are employed. These measures are applied at the levels indicated by the blue edges in Figure 5.1. The first measure is purely graph-theoretical, used to compare the fitted CPDAGs to the true underlying CPDAG. This measure is adapted from Tsamardinos et al. (2006, p.53) and implemented in the R-package `pcalg` (Kalisch and Bühlmann 2007).

Definition 5.2.1 (Structural Hamming Distance (SHD))**.** Let $\mathcal{G}_1 = (\mathcal{V}, \mathcal{E}_1)$ and $\mathcal{G}_2 = (\mathcal{V}, \mathcal{E}_2)$ be two PDAGs with the same set of nodes $\mathcal{V} = \{1, \dots, d\}$. Denote their respective adjacency matrices as $M^{(1)}$ and $M^{(2)}$, defined by

$$M_{ij}^{(1)} := \mathbb{1}\{(i,j) \in \mathcal{E}_1\}, \quad \text{and} \quad M_{ij}^{(2)} := \mathbb{1}\{(i,j) \in \mathcal{E}_2\}$$

for $i, j = 1, \dots, d$, where $\mathbb{1}(\cdot)$ is the indicator function. The adjacency matrices for the skeletons of $\mathcal{G}_1$ and $\mathcal{G}_2$ are given by $S^{(1)}$ and $S^{(2)}$, respectively, where

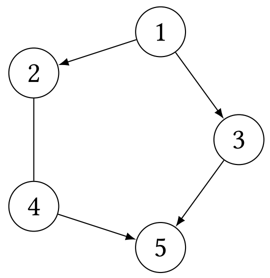
$$S_{ij}^{(1)} = \mathbb{1}\{M_{ij}^{(1)} + M_{ji}^{(1)} = 2\} + \mathbb{1}\{M_{ij}^{(1)} + M_{ji}^{(1)} \neq 2\}(M_{ij}^{(1)} + M_{ji}^{(1)}),$$
$$S_{ij}^{(2)} = \mathbb{1}\{M_{ij}^{(2)} + M_{ji}^{(2)} = 2\} + \mathbb{1}\{M_{ij}^{(2)} + M_{ji}^{(2)} \neq 2\}(M_{ij}^{(2)} + M_{ji}^{(2)}),$$

for $i, j = 1, \dots, d$. The difference between the skeletons of $\mathcal{G}_1$ and $\mathcal{G}_2$ is represented by the matrix $D^{(skel)}$, defined by the elements

$$D_{ij}^{(skel)} := S_{ij}^{(1)} - S_{ij}^{(2)},$$

for $i, j = 1, \dots, d$. Entries in $D^{(skel)}$ with a value of 1 correspond to edges that are present in the skeleton of $\mathcal{G}_1$ but missing in the skeleton of $\mathcal{G}_2$, while entries with a value of -1 indicate extra edges in the skeleton of $\mathcal{G}_2$ compared to the skeleton of $\mathcal{G}_1$. Define the $d \times d$-matrices $M^{(missing)}$ and $M^{(extra)}$ as

$$M_{ij}^{(missing)} := \mathbb{1}\{D_{ij}^{(skel)} = 1\}, \quad \text{and} \quad M_{ij}^{(extra)} := \mathbb{1}\{D_{ij}^{(skel)} = -1\},$$


(a) $\mathcal{G}_1 = (\mathcal{V}, \mathcal{E}_1)$

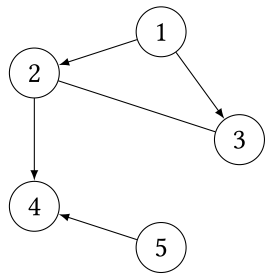

(b) $\mathcal{G}_2 = (\mathcal{V}, \mathcal{E}_2)$

Figure 5.2 Two different PDAGs with the same node set

for $i, j = 1, \ldots, d$. The total number of missing and extra edges are calculated as

$$N^{(missing)} = \frac{1}{2} \cdot \sum_{i=1}^{d} \sum_{j=1}^{d} M_{ij}^{(missing)}, \quad \text{and} \quad N^{(extra)} = \frac{1}{2} \cdot \sum_{i=1}^{d} \sum_{j=1}^{d} M_{ij}^{(extra)},$$

where the factor 1/2 prevents double counting of undirected edges. The adjacency matrix $M^{(combined)}$ is constructed to represent the DAG obtained by modifying $\mathcal{G}_1$ by removing missing edges and adding extra edges from $\mathcal{E}_2$:

$$M_{ij}^{(combined)} := M_{ij}^{(1)}(1 - M_{ij}^{(missing)}) + \mathbb{1}\{M_{ij}^{(2)} + M_{ij}^{(extra)} = 2\},$$

for $i, j = 1, \ldots, d$. The matrix $D^{(direction)}$ is then defined as the adjacency matrix of the DAG consisting of edges present in both $\mathcal{G}_1$ and $\mathcal{G}_2$ but with differing directions:

$$D_{ij}^{(direction)} := |M_{ij}^{(combined)} - M_{ij}^{(2)}|,$$

for $i, j = 1, \ldots, d$. The number of edges with different directions is given by

$$N^{(direction)} = \frac{1}{2} \cdot \sum_{i=1}^{d} \sum_{j=1}^{d} \mathbb{1}\{D_{ij}^{(direction)} + D_{ji}^{(direction)} > 0\}.$$

Finally, the **Structural Hamming Distance** between $\mathcal{G}_1$ and $\mathcal{G}_2$ is defined as

$$\text{SHD}(\mathcal{G}_1, \mathcal{G}_2) := N^{(missing)} + N^{(extra)} + N^{(direction)}.$$

In simple terms, the Structural Hamming Distance (SHD) quantifies the number of edge additions, deletions, or reversals required to transform one PDAG into another. The following example illustrates the computation of the SHD for two PDAGs with the same set of nodes.

Example 5.2.1. Consider the PDAGs depicted in Figure 5.2. Both PDAGs share the same node set, $\mathcal{V} = \{1, \ldots, 5\}$, with their respective edge sets given by:

$$\mathcal{E}_1 = \{(1,2), (1,3), (2,4), (4,2), (3,5), (4,5)\},$$
$$\mathcal{E}_2 = \{(1,2), (1,3), (2,3), (3,2), (2,4), (5,4)\}.$$

The corresponding adjacency matrices are:

$$M^{(1)} = \begin{pmatrix} 0 & 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 \end{pmatrix}, \quad M^{(2)} = \begin{pmatrix} 0 & 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \end{pmatrix}.$$

The adjaceny matrices for the skeletons of the PDAGs are:

$$S^{(1)} = \begin{pmatrix} 0 & 1 & 1 & 0 & 0 \\ 1 & 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 & 1 \\ 0 & 0 & 1 & 1 & 0 \end{pmatrix}, \quad S^{(2)} = \begin{pmatrix} 0 & 1 & 1 & 0 & 0 \\ 1 & 0 & 1 & 1 & 0 \\ 1 & 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 & 0 \end{pmatrix}.$$

The difference between the skeletons is:

$$D^{(skel)} = S^{(1)} - S^{(2)} = \begin{pmatrix} 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & -1 & 0 & 0 \\ 0 & -1 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \end{pmatrix}.$$

From $D^{(skel)}$, the matrices for missing and extra edges are:

$$M^{(missing)} = \begin{pmatrix} 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \end{pmatrix}, \quad M^{(extra)} = \begin{pmatrix} 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{pmatrix}.$$

The number of missing edges is:

$$N^{(missing)} = \frac{1}{2} \cdot \sum_{i=1}^{5} \sum_{j=1}^{5} M_{ij}^{(missing)} = 1.$$

This value corresponds to the directed edge $(3, 5) \in \mathcal{E}_1$. The number of extra edges is:

$$N^{(extra)} = \frac{1}{2} \cdot \sum_{i=1}^{5} \sum_{j=1}^{5} M_{ij}^{(extra)} = 1.$$

This value corresponds to the undirected edge $\{(2, 3), (3, 2)\} \in \mathcal{E}_2$. Note that undirected edges are counted only once, similar to directed edges.

Next, consider the adjacency matrix $M^{(combined)}$ for the DAG obtained by removing edges not present in $\mathcal{E}_2$ and adding the extra edges from $\mathcal{E}_2$:

$$M^{(combined)} = \begin{pmatrix} 0 & 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 \end{pmatrix}.$$

Comparing $M^{(combined)}$ to $M^{(1)}$ reveals that the directed edge (3,5) is removed and the undirected edge $\{(2, 3), (3, 2)\}$ is added. The comparison between $M^{(combined)}$ and $M^{(2)}$ is reflected in the matrix $D^{(direction)}$:

$$D^{(direction)} = \begin{pmatrix} 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 & 0 \end{pmatrix}.$$

The number of edges with different directions is:

$$N^{(direction)} = \frac{1}{2} \cdot \sum_{i=1}^{d} \sum_{j=1}^{d} \mathbb{1}\{D_{ij}^{(direction)} + D_{ji}^{(direction)} > 0\} = 2.$$

This includes the edge $(2,4)$, which is directed in $\mathcal{E}_2$ but undirected in $\mathcal{E}_1$, and the edge $(4,5)$, which exists in $\mathcal{E}_1$ but is reversed in $\mathcal{E}_2$.

Finally, the SHD between $\mathcal{G}_1$ and $\mathcal{G}_2$ is:

$$\mathrm{SHD}(\mathcal{G}_1, \mathcal{G}_2) = N^{(missing)} + N^{(extra)} + N^{(direction)} = 1 + 1 + 2 = 4.$$

Throughout the simulation studies, the SHD will be used for model comparison as follows: For each of the N repetitions, two fitted CPDAGs, denoted as CPDAG$^{(Y)}$ and CPDAG$^{(Z)}$, are obtained by applying the PC algorithm (see Step 6 in Figure 5.1). For each fitted CPDAG, the SHD between the fitted CPDAG and the true CPDAG will be calculated and compared:

$$\mathrm{SHD}_i^{(Y)} := \mathrm{SHD}([\mathcal{G}^{(\mathrm{true})}], [\mathcal{G}_i^{(Y)}]),$$
$$\mathrm{SHD}_i^{(Z)} := \mathrm{SHD}([\mathcal{G}^{(\mathrm{true})}], [\mathcal{G}_i^{(Z)}]),$$

for $i = 1, \ldots, N$, where $[\mathcal{G}^{(\mathrm{true})}]$ represents the true CPDAG, and $[\mathcal{G}_i^{(Y)}]$ and $[\mathcal{G}_i^{(Z)}]$ denote the fitted CPDAGs in the i-th repetition. Generally, a lower SHD for one of the fitted models compared to the other indicates that it is closer to the true model in terms of its structure. Therefore, the distributions of $(\mathrm{SHD}_i^{(Y)})_{i=1,\ldots,N}$ and $(\mathrm{SHD}_i^{(Z)})_{i=1,\ldots,N}$ will be compared. However, it is important to note that SHD alone does not provide a probabilistic interpretation. A lower SHD does not guarantee that one model fits the distribution better than the other. Consequently, an additional measure is needed to evaluate the models' performance more comprehensively.

A widely used measure for evaluating the goodness-of-fit of statistical models is the likelihood. In Section 4.3, it was shown that the true log-likelihood of a PCBN can often only be computed using numerical integration. However, an integration-free approximation of the log-likelihood can be derived from the estimated DV-SEM.

Consider the DV-SEM$^{(\mathrm{true})}$, which is obtained from the true, underlying PCBN $(\mathcal{G}^{(\mathrm{true})}, \mathcal{O}^{(\mathrm{true})})$. The approximated copula log-likelihood of the true model is given by:

$$\hat{\ell}(\boldsymbol{\theta}_k^{(DV,\mathrm{true})}; \mathbf{u}) = \sum_{i=1}^{n} \sum_{j=1}^{d} \sum_{w \in pa(j)} \ln \Big(c_{j,w;pa(j;w)} \big(C_{j|pa(j;w)}(u_{ij} \mid \mathbf{u}_{i,pa(j;w)}; \boldsymbol{\theta}_k^{(DV,\mathrm{true})}), \\ C_{w|pa(j;w)}(u_{iw} \mid \mathbf{u}_{i,pa(j;w)}; \hat{\boldsymbol{\theta}}_k^{(parDV,\mathrm{true})}); \theta_{j,w;pa(j;w)} \big) \Big),$$

where $k = 1, \ldots, N$, and

$$\boldsymbol{\theta}_k^{(DV,\mathrm{true})} = \boldsymbol{\theta}^{(\mathrm{true})} \cup \hat{\boldsymbol{\theta}}_k^{(parDV,\mathrm{true})},$$

denotes the parameters of DV-SEM$^{(\mathrm{true})}$. The parameters $\boldsymbol{\theta}^{(\mathrm{true})}$ are fixed values determined for the simulation and remain constant across all repetitions, whereas $\hat{\boldsymbol{\theta}}_k^{(parDV,\mathrm{true})}$ refers to the parameters estimated from the parental D-vines, which vary with each simulated sample. Consequently, the log-likelihood function also depends on the sample through the conditional distribution functions, as indicated by the subscript k.

For the fitted DV-SEMs, denoted as DV-SEM$^{(Y)}$ and DV-SEM$^{(Z)}$, the log-likelihood are defined as follows:

$$\hat{\ell}(\hat{\boldsymbol{\theta}}_k^{(DV,Y)}; \mathbf{u}) = \sum_{i=1}^{n} \sum_{j=1}^{d} \sum_{w \in pa^{(Y,k)}(j)} \ln \Big(c_{j,w;pa^{(Y,k)}(j;w)} \big(C_{j|pa^{(Y,k)}(j;w)}(u_{ij} \mid \mathbf{u}_{i,pa^{(Y,k)}(j;w)}; \hat{\boldsymbol{\theta}}_k^{(DV,Y)}), \\ C_{w|pa^{(Y,k)}(j;w)}(u_{iw} \mid \mathbf{u}_{i,pa^{(Y,k)}(j;w)}; \hat{\boldsymbol{\theta}}_k^{(parDV,Y)}); \hat{\theta}_{j,w;pa^{(Y,k)}(j;w)} \big) \Big),$$

and

$$\hat{\ell}(\hat{\boldsymbol{\theta}}_k^{(DV,Z)};\mathbf{u}) = \sum_{i=1}^{n}\sum_{j=1}^{d}\sum_{w\in pa^{(Z,k)}(j)} \ln\Big(c_{j,w;pa^{(Z,k)}(j;w)}\big(C_{j|pa^{(Z,k)}(j;w)}(u_{ij}\mid \mathbf{u}_{i,pa^{(Z,k)}(j;w)};\hat{\boldsymbol{\theta}}_k^{(DV,Z)}),$$
$$C_{w|pa^{(Z,k)}(j;w)}(u_{iw}\mid \mathbf{u}_{i,pa^{(Z,k)}(j;w)};\hat{\boldsymbol{\theta}}_k^{(parDV,Z)});\hat{\theta}_{j,w;pa^{(Z,k)}(j;w)}\big)\Big),$$

for $k = 1,\dots,N$. In contrast to $\hat{\ell}(\boldsymbol{\theta}_k^{(DV,\text{true})};\mathbf{u})$, where only the parameters in the parental D-vines are estimated, the fitted DV-SEMs estimate the entire D-vine structure. This distinction is denoted by the hat symbol on $\boldsymbol{\theta}$:

$$\hat{\boldsymbol{\theta}}_k^{(DV,Y)} := \hat{\boldsymbol{\theta}}_k^{(Y)} \cup \hat{\boldsymbol{\theta}}_k^{(parDV,Y)},$$
$$\hat{\boldsymbol{\theta}}_k^{(DV,Z)} := \hat{\boldsymbol{\theta}}_k^{(Z)} \cup \hat{\boldsymbol{\theta}}_k^{(parDV,Z)}.$$

Here, $\hat{\boldsymbol{\theta}}_k^{(Y)}$ (and similarly, $\hat{\boldsymbol{\theta}}_k^{(Z)}$) represents the parameters of the copulas that are specified by the decomposition of the fitted PCBN. These parameters correspond to the fitted copulas for the first edge in each tree of each D-vine. Additionally, $\hat{\boldsymbol{\theta}}_k^{(parDV,Y)}$ (and similarly, $\hat{\boldsymbol{\theta}}_k^{(parDV,Z)}$) denotes the parameters of the fitted parental D-vines. The functions $pa(\cdot)$ and $pa(\cdot;\cdot)$ depend not only on the conditional independence tests employed but also on the specific sample in each repetition, as indicated by the superscript k.

Example 5.2.2 (Example 4.2.3 continued)**.** Consider the six-dimensional PCBN shown in Figure 4.7 and its corresponding DV-SEM as presented in Example 4.2.3. Assume this model represents the true underlying structure from which n observations are simulated N times. The copula log-likelihood of the true model is calculated using the parameters:

$$\boldsymbol{\theta}_k^{(DV,\text{true})} = \boldsymbol{\theta}^{(\text{true})} \cup \hat{\boldsymbol{\theta}}_k^{(parDV,\text{true})},$$

where,

$$\boldsymbol{\theta}^{(\text{true})} := \{\theta_{32},\theta_{31;2},\theta_{43},\theta_{65},\theta_{62;5},\theta_{61;52},\theta_{64;521}\}$$

are the fixed copula parameters specified by the decomposition of the PCBN (highlighted in green in Example 4.2.3). Furthermore,

$$\hat{\boldsymbol{\theta}}_k^{(parDV,\text{true})} := \{\hat{\theta}_{21}^k,\hat{\theta}_{52}^k,\hat{\theta}_{14}^k,\hat{\theta}_{51;2}^k,\hat{\theta}_{24;1}^k,\hat{\theta}_{54;21}^k\}$$

represents the parameters of the parental D-vines, which may vary with each simulation run, $k = 1,\dots,N$, as these are estimated based on the simulated sample. The approximated log-likelihood for some simulation run $k \in \{1,\dots,N\}$ is then given by:

$$\begin{aligned}\hat{\ell}(\boldsymbol{\theta}_k^{(DV,\text{true})};\mathbf{u}) = \sum_{i=1}^{n}\Bigg[&\ln\Big(c_{32}(u_{i3},u_{i2};\theta_{32})\Big)\\
&+\ln\Big(c_{31;2}(C_{3|2}(u_{i3}\mid u_{i2};\theta_{32}),C_{1|2}(u_{i1}\mid u_{i2};\hat{\theta}_{21}^k);\theta_{31;2})\Big)\\
&+\ln\Big(c_{43}(u_{i4},u_{i3};\theta_{43})\Big)+\ln\Big(c_{65}(u_{i6},u_{i5};\theta_{65})\Big)\\
&+\ln\Big(c_{62;5}(C_{6|5}(u_{i6}\mid u_{i5};\theta_{65}),C_{2|5}(u_{i2}\mid u_{i5};\hat{\theta}_{52}^k);\theta_{62;5})\Big)\\
&+\ln\Big(c_{61;52}(C_{6|52}(u_{i6}\mid u_{i5},u_{i2};\theta_{62;5},\theta_{65},\hat{\theta}_{52}^k),C_{1|52}(u_{i1}\mid u_{i5},u_{i2};\hat{\theta}_{51;2}^k,\hat{\theta}_{52}^k,\hat{\theta}_{21}^k);\theta_{61;52})\Big)\\
&+\ln\Big(c_{64;521}(C_{6|521}(u_{i6}\mid u_{i5},u_{i2},u_{i1};\theta_{61;52},\theta_{62;5},\hat{\theta}_{51;2}^k,\theta_{65},\hat{\theta}_{52}^k,\hat{\theta}_{21}^k),\\
&\qquad C_{4|521}(u_{i4}\mid u_{i5},u_{i2},u_{i1};\hat{\theta}_{54;21}^k,\hat{\theta}_{51;2}^k,\hat{\theta}_{24;1}^k,\hat{\theta}_{52}^k,\hat{\theta}_{21}^k,\hat{\theta}_{14}^k);\theta_{64;521})\Big)\Bigg].\end{aligned}$$

For each repetition $k = 1, \dots, N$, the copula log-likelihoods for all three models will be computed using the simulated sample $\hat{\mathbf{u}}^s$ (see Step 3 in Figure 5.1). Directly comparing $\hat{\ell}(\boldsymbol{\theta}_k^{(DV,\text{true})}; \hat{\mathbf{u}}^s)$, $\hat{\ell}(\hat{\boldsymbol{\theta}}_k^{(DV,Y)}; \hat{\mathbf{u}}^s)$, and $\hat{\ell}(\hat{\boldsymbol{\theta}}_k^{(DV,Z)}; \hat{\mathbf{u}}^s)$ for each k might be inconclusive, as there is no guarantee that the models are nested. To address this, the log-likelihood is penalized according to the number of parameters in the copulas specified by each model's decomposition.

The penalization parameter, which varies with each repetition $k = 1, \dots, N$, is denoted by p_k. For each model, it is defined as follows:

$$\begin{aligned} p_k^{(\text{true})} &:= |\boldsymbol{\theta}^{(\text{true})}|, \\ p_k^{(Y)} &:= |\hat{\boldsymbol{\theta}}_k^{(Y)}|, \\ p_k^{(Z)} &:= |\hat{\boldsymbol{\theta}}_k^{(Z)}|, \end{aligned}$$

where $p_k^{(\text{true})}$ is constant across repetitions, as the true copulas are fixed.

The AIC and BIC for each model are computed as follows:

$$\begin{aligned} \text{AIC}_k^{(\text{true})} &:= \text{AIC}(\boldsymbol{\theta}_k^{(DV,\text{true})}; \hat{\mathbf{u}}^s) := 2 \cdot p_k^{(\text{true})} - 2 \cdot \hat{\ell}(\boldsymbol{\theta}_k^{(DV,\text{true})}; \hat{\mathbf{u}}^s), \\ \text{AIC}_k^{(Y)} &:= \text{AIC}(\hat{\boldsymbol{\theta}}_k^{(DV,Y)}; \hat{\mathbf{u}}^s) := 2 \cdot p_k^{(Y)} - 2 \cdot \hat{\ell}(\hat{\boldsymbol{\theta}}_k^{(DV,Y)}; \hat{\mathbf{u}}^s), \\ \text{AIC}_k^{(Z)} &:= \text{AIC}(\hat{\boldsymbol{\theta}}_k^{(DV,Z)}; \hat{\mathbf{u}}^s) := 2 \cdot p_k^{(Z)} - 2 \cdot \hat{\ell}(\hat{\boldsymbol{\theta}}_k^{(DV,Z)}; \hat{\mathbf{u}}^s), \end{aligned}$$

and

$$\begin{aligned} \text{BIC}_k^{(\text{true})} &:= \text{BIC}(\boldsymbol{\theta}_k^{(DV,\text{true})}; \hat{\mathbf{u}}^s) := \ln(n) \cdot p_k^{(\text{true})} - 2 \cdot \hat{\ell}(\boldsymbol{\theta}_k^{(DV,\text{true})}; \hat{\mathbf{u}}^s), \\ \text{BIC}_k^{(Y)} &:= \text{BIC}(\hat{\boldsymbol{\theta}}_k^{(DV,Y)}; \hat{\mathbf{u}}^s) := \ln(n) \cdot p_k^{(Y)} - 2 \cdot \hat{\ell}(\hat{\boldsymbol{\theta}}_k^{(DV,Y)}; \hat{\mathbf{u}}^s), \\ \text{BIC}_k^{(Z)} &:= \text{BIC}(\hat{\boldsymbol{\theta}}_k^{(DV,Z)}; \hat{\mathbf{u}}^s) := \ln(n) \cdot p_k^{(Z)} - 2 \cdot \hat{\ell}(\hat{\boldsymbol{\theta}}_k^{(DV,Z)}; \hat{\mathbf{u}}^s), \end{aligned}$$

where n is the number of simulated observations (typically $n = 1000$ in this study). Unlike the Structural Hamming Distance (SHD), which assesses model fit from a structural perspective, AIC and BIC offer a probabilistic evaluation of the goodness-of-fit.

5.3 Choice of Tuning Parameters

When estimating graphical structures using the PC algorithm, it is essential to set the significance level and tuning parameters, depending on the conditional independence test used. If Fisher's Z-test of partial correlation is applied, the only required parameter is α^Z, which defines the significance level for the null hypothesis of (conditional) independence. Alternatively, if the Y-vine-based conditional independence test is employed, two parameters must be set: α^Y, the significance level for testing ordinary independence, and k^Y, which acts as an upper bound on the estimated absolute τ-value of the last copula in a fitted Y-vine when the conditioning set is non-empty. If the estimated τ-value exceeds k^Y, the null hypothesis of conditional independence is rejected.

Both α^Z and α^Y are traditional significance levels, associated with asymptotic tests, allowing them to be interpreted in terms of type I error (the probability of falsely rejecting a true null hypothesis), typically set at $\alpha = 0.05$. Therefore, throughout the simulations, both α^Y and $\alpha^Z$are set to 0.05. In Chapter 7, other specific values will also be examined.

Unlike the significance levels, k^Y is a tuning parameter with no probabilistic interpretation. It serves as an upper threshold on the estimated absolute τ-value of the final copula in the Y-vine. Setting $k^Y \in [0, 1]$ involves a trade-off: higher values accept the null hypothesis more frequently, leading to more edge

(i,j)	S	$X_i \perp\!\!\!\perp X_j \mid X_S$?	$k^Y = 0.01$	$k^Y = 0.025$	$k^Y = 0.05$	$k^Y = 0.075$	$k^Y = 0.1$
(3,6)	1,2,4	TRUE	0.12	0.64	0.84	0.96	0.98
(1,4)	2,3,5	TRUE	0.26	0.58	0.94	0.96	0.96
(3,6)	4,5	FALSE	0.98	0.94	0.90	0.80	0.78
(2,3)	1,6	FALSE	0.98	0.98	0.98	0.98	0.98
(5,6)	1,2,4	FALSE	1.00	1.00	1.00	1.00	1.00
(1,3)	4	FALSE	1.00	1.00	0.96	0.88	0.78
(2,4)	3	TRUE	0.18	0.56	0.86	0.94	0.98
(2,3)	1,5,6	FALSE	1.00	1.00	0.98	0.98	0.98
(2,3)	6	FALSE	1.00	1.00	1.00	0.96	0.94
(3,4)	1,2,5,6	FALSE	0.98	0.94	0.92	0.82	0.76
(4,6)	2,3,5	FALSE	1.00	1.00	0.96	0.90	0.82
(1,4)	3	TRUE	0.16	0.66	0.94	1.00	1.00
(2,4)	1,3,5,6	FALSE	0.98	0.96	0.82	0.76	0.64
(3,5)	1,2	TRUE	0.14	0.58	0.98	0.98	1.00
(2,6)	1	FALSE	1.00	1.00	1.00	0.94	0.88
(3,6)	1,4	FALSE	0.98	0.90	0.78	0.70	0.60
(2,3)	1,4	FALSE	1.00	1.00	1.00	0.98	0.96
(4,5)	3	TRUE	0.22	0.60	0.98	1.00	1.00
(5,6)	4	FALSE	1.00	1.00	1.00	1.00	1.00
(4,5)	1,2,6	FALSE	1.00	1.00	0.92	0.88	0.84

Table 5.1 Comparison of the percentage of correctly estimated (in-)dependence statements over $N = 100$ repetitions using the Y-vine-based conditional independence test with varying k^Y values

Statement Type	$k^Y = 0.01$	$k^Y = 0.025$	$k^Y = 0.05$	$k^Y = 0.075$	$k^Y = 0.1$
All	0.748	0.867	0.938	0.921	0.894
Independence	0.180	0.603	0.923	0.973	0.987
Dependence	0.993	0.980	0.944	0.899	0.854

Table 5.2 Comparison of the average percentage of correctly estimated (in-)dependence statements over $N = 100$ repetitions based on the results illustrated in Table 5.1

removals but increasing the risk of type II errors (false negatives). Conversely, lower k^Y values lead to more edge retention, increasing the chance of type I errors (false positives). As there is no distributional result for the true τ-value of the last tree in a Y-vine, these errors cannot be quantified.

An experimental study is conducted to balance this trade-off. Simulating $n = 1000$ observations for $N = 100$ repetitions from the PCBN in Figure 4.7, pair-copulas are randomly selected from one-parametric families (Appendix B), including the t-copula with degrees of freedom set to 5. Kendall's τ-values are sampled from the interval $(-0.8, -0.1) \cup (0.1, 0.8)$ and transformed into copula parameters. The approximate simulation procedure from Section 4.2.2 is applied, followed by parametric margins and scaling, as described on page 69.

The Y-vine conditional independence test is applied with $k^Y \in \{0.01, 0.025, 0.05, 0.075, 0.1\}$. Out of the 240 statements associated with the DAG in Figure 4.7, 45 are independent. Given the computational demands of testing all statements across 100 repetitions, a random subset of 20 statements is analyzed, 6 being independent and 14 dependent.

Table 5.1 shows the frequencies of correctly estimated dependence relations for different k^Y values. As expected, low k^Y values accurately estimate dependence but struggle with conditional independence, while high k^Y values detect independence well but often miss dependencies.

Table 5.2 compares average correct frequencies across the varying k^Y values. The Y-test with $k^Y = 0.01$ achieves a high 99.3% accuracy in estimating dependence but only identifies 18% of independence statements correctly. In contrast, the test with $k^Y = 0.1$ correctly identifies 98.7% of independence but only 85.4% of dependence statements. The test with $k^Y = 0.05$ performs best overall, correctly estimating over 90% of both dependence and independence statements, with an average accuracy of 93.8%. It appears to best balance detecting both dependencies and independencies.

Therefore, the simulation study in the following section will use the Y-vine test with $\alpha^Y = k^Y = 0.05$. Other values of k^Y will be further explored in Chapter 7.

5.4 Case 1: Four Dimensions

Consider the four-dimensional PCBN depicted in Figure 4.3. Here, $d = 4$, and the graph is represented as $\mathcal{G}^{(\text{true})} = (\mathcal{V}, \mathcal{E}^{(\text{true})})$ where $\mathcal{V} := \{1, \ldots, 4\}$ and $\mathcal{E}^{(\text{true})} := \{(1,2), (1,3), (2,4), (3,4)\}$. Additionally, the ordering of the parents is $\mathcal{O}^{(\text{true})} := \{2 <_4 3\}$. The implied conditional independencies are $X_2 \perp\!\!\!\perp X_3 \mid X_1$ and $X_1 \perp\!\!\!\perp X_4 \mid X_2, X_3$. The joint density factorizes as follows:

$$f(\mathbf{x}) = \Big(\prod_{i=1}^{4} f_i \Big) \cdot c_{21} \cdot c_{31} \cdot c_{42} \cdot c_{43;2}, \quad \mathbf{x} \in \mathbb{R}^4,$$

where the arguments of the functions are omitted for simplicity.

Table 5.3 presents the specified pair-copulas for each setup described on page 70. The fixed parametric marginal distributions associated with these setups are detailed in Table 5.4. For each of the four setups, the procedure outlined in Figure 5.1 is executed $N = 100$ times, with each repetition involving $n = 1000$ observations. The tuning parameters and significance levels for each (conditional) independence test are set to $\alpha^Z = \alpha^Y = k^Y = 0.05$.

Figure 5.3 illustrates the observed relative frequencies of $\text{SHD}_i^{(Y)}$ and $\text{SHD}_i^{(Z)}$ for $i = 1, \ldots, N$. The results indicate that CPDAGs based on the Y-vine approach generally exhibit lower SHD values compared to those obtained using the Z-test within the PC algorithm. This trend is further supported by the lower average SHD values reported in the "Y" column of Table 5.5, particularly in Setups 1 to 3, which involve non-Gaussian copulas. Overall, this suggests that Y-vine-based models are structurally closer to the true model than their Z-test based counterparts. For instance, in Setup 2, Y-vine-based conditional independence tests recover the true CPDAG in 68% of cases, whereas the Z-test based PC algorithm fails to recover the correct CPDAG in any of the cases. The smallest difference between $\text{SHD}^{(Y)}$ and $\text{SHD}^{(Z)}$ is observed in Setup 4, which involves a Gaussian Bayesian network.

Similarly, Figure 5.4 shows box plots of the AIC and BIC values, revealing that the goodness-of-fit measures for the true and fitted models differ more in the first three setups, with the Y-vine-based approach

Setup	Edge	Family	Rotation	Parameters	τ
1	21	Gaussian	0°	-0.57	-0.39
	31	Frank	0°	12.23	0.72
	42	t	0°	(0.4, 5)	0.27
	43;2	Joe	270°	2.85	-0.50
2	21	Joe	180°	1.83	0.31
	31	Frank	0°	-7.85	-0.60
	42	Gumbel	180°	2.32	0.57
	43;2	Joe	90°	1.74	-0.29
3	21	Gaussian	0°	-0.36	-0.24
	31	t	0°	(0.78, 5)	0.57
	42	t	0°	(-0.31, 5)	-0.20
	43;2	t	0°	(-0.85, 5)	-0.65
4	21	Gaussian	0°	-0.23	-0.15
	31	Gaussian	0°	-0.91	-0.73
	42	Gaussian	0°	0.78	0.57
	43;2	Gaussian	0°	0.97	0.83

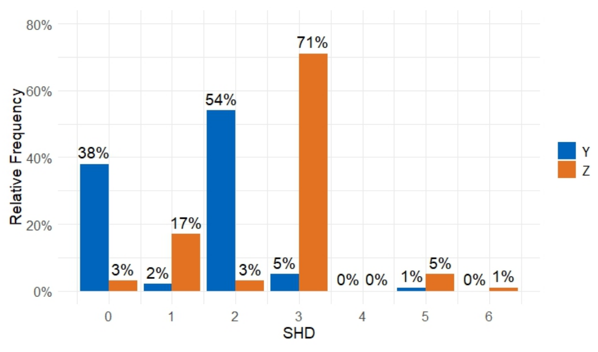
Table 5.3 Overview of pair-copulas for all setups in the case $d = 4$

Setup	Node	Marginal Distribution	Parameters
1	1	Normal	$\mu = \sigma = 1$
	2	Normal	$\mu = \sigma = 1$
	3	Beta	$\alpha = \beta = 0.5$
	4	Lognormal	$\mu = 0, \sigma = 1$
2	1	Lognormal	$\mu = 0, \sigma = 1$
	2	Normal	$\mu = \sigma = 1$
	3	Beta	$\alpha = \beta = 0.5$
	4	t	$\nu = 5$
3	1	Lognormal	$\mu = 0, \sigma = 1$
	2	Beta	$\alpha = \beta = 0.5$
	3	Beta	$\alpha = \beta = 0.5$
	4	Normal	$\mu = \sigma = 1$
4	1	Normal	$\mu = \sigma = 1$
	2	Normal	$\mu = \sigma = 1$
	3	Normal	$\mu = \sigma = 1$
	4	Normal	$\mu = \sigma = 1$

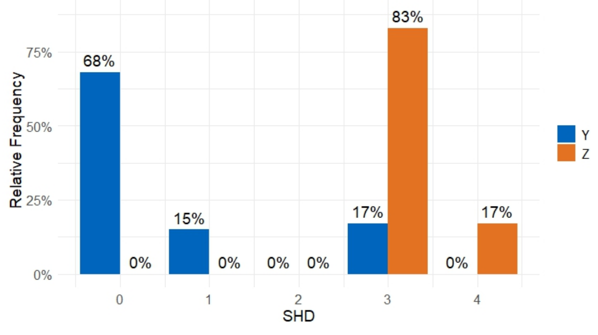
Table 5.4 Overview of marginal distributions for all setups in the case $d = 4$

Setup	SHD		Log-Likelihood			AIC			BIC		
	Y	Z	True	Y	Z	True	Y	Z	True	Y	Z
1	1.30	2.67	1535	1424	1418	-3060	-2838	-2823	-3035	-2813	-2791
2	0.66	3.17	1330	1227	1122	-2653	-2438	-2231	-2633	-2401	-2198
3	0.17	3.96	1294	1239	1217	-2575	-2462	-2412	-2541	-2422	-2360
4	1.06	1.08	2729	2689	2693	-5450	-5371	-5378	-5430	-5353	-5359

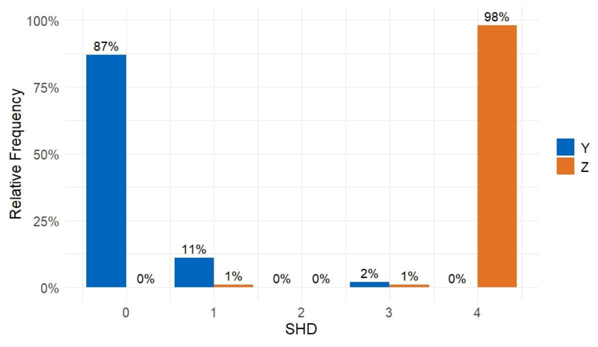
Table 5.5 Rounded average values of different performance measures for the case $d = 4$. Column "True" serves as a benchmark. Highest log-likelihood and lowest SHD, AIC, and BIC value between Y and Z is colored in blue



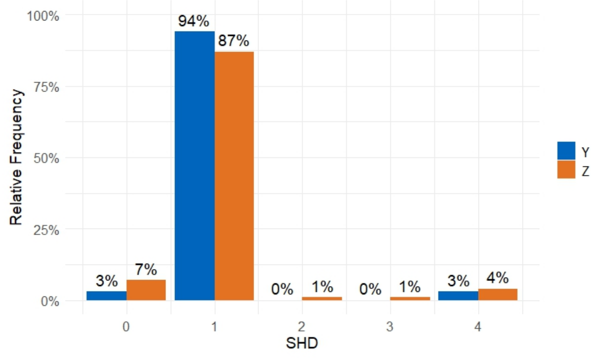
(a) Setup 1: Mixture of Elliptical and Archimedean



(b) Setup 2: Only Archimedean

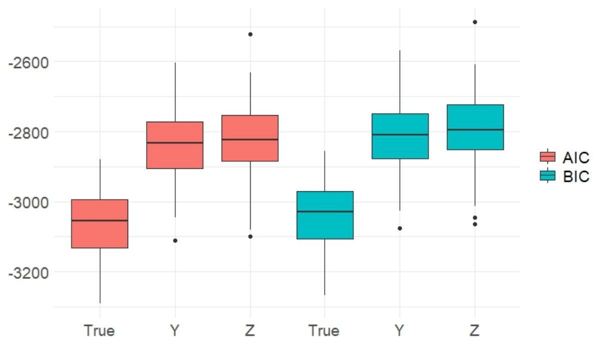


(c) Setup 3: Only Elliptical

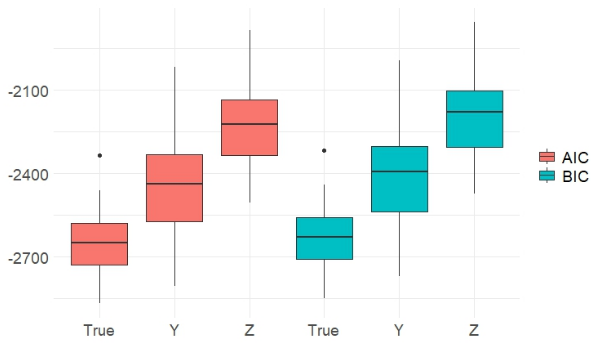


(d) Setup 4: Only Gaussian

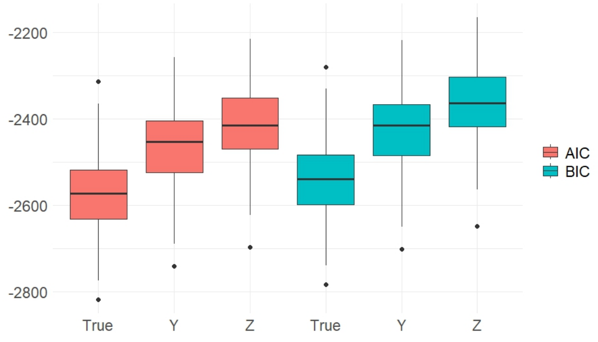
Figure 5.3 Relative frequency of observed SHD values between the Expert CPDAG and the fitted Y-CPDAG (blue), and between the Expert CPDAG and the fitted Z-CPDAG (orange) across different setups for $d = 4$



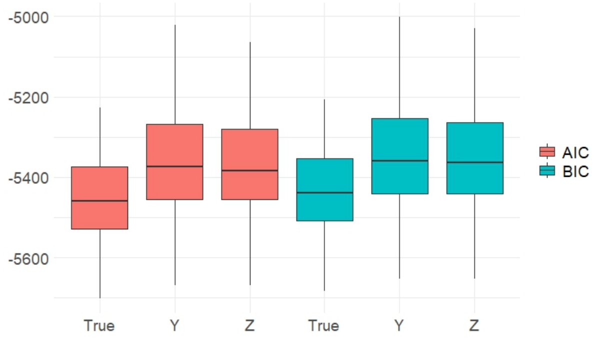
(a) Setup 1: Mixture of Elliptical and Archimedean



(b) Setup 2: Only Archimedean



(c) Setup 3: Only Elliptical



(d) Setup 4: Only Gaussian

Figure 5.4 Box plots of estimated AIC/BIC values for the different setups in the case $d = 4$

outperforming the Z-test benchmark, most notably in Setup 2 and 3. In Setup 4, the fitted models align more closely with the true model in terms of AIC and BIC. Here, the Z-test based model shows a slightly better fit compared to the Y-vine model. This is corroborated by Table 5.5, which shows a marginally higher average log-likelihood and lower average AIC and BIC values for the Z-test based PCBNs in Setup 4, and the opposite trend in Setups 1 to 3.

5.5 Case 2: Six Dimensions

Consider the six-dimensional PCBN depicted in Figure 4.7. Here, $d = 6$, and the graph is represented as $\mathcal{G}^{(\text{true})} = (\mathcal{V}, \mathcal{E}^{(\text{true})})$ where $\mathcal{V} := \{1, \dots, 6\}$ and $\mathcal{E}^{(\text{true})} := \{(1,3), (1,6), (2,3), (2,6), (3,4), (4,6), (5,6)\}$. Additionally, the ordering of the parents is given by

$$\begin{aligned} &2 <_3 1, \\ 5 <_6 \, &2 <_6 1 <_6 4. \end{aligned}$$

The implied conditional independencies are listed on page 14. The joint density factorizes as follows:

$$f(\mathbf{x}) = \Big(\prod_{i=1}^{4} f_i\Big) \cdot c_{32} \cdot c_{31;2} \cdot c_{43} \cdot c_{65} \cdot c_{62;5} \cdot c_{61;52} \cdot c_{64;521}, \quad \mathbf{x} \in \mathbb{R}^6,$$

where the arguments of the functions are omitted for simplicity.

Table 5.6 presents the specified pair-copulas for each setup described on page 70. The fixed parametric marginal distributions associated with these setups are detailed in Table 5.7. For each of the four setups, the procedure outlined in Figure 5.1 is executed $N = 100$ times, with each repetition involving $n = 1000$ observations. The tuning parameters and significance levels for each (conditional) independence test are set to $\alpha^Z = \alpha^Y = k^Y = 0.05$.

Figure 5.5 illustrates the observed relative frequencies of $\text{SHD}_i^{(Y)}$ and $\text{SHD}_i^{(Z)}$ for $i = 1, \dots, N$. The results indicate that CPDAGs based on the Y-vine approach generally exhibit lower SHD values compared to those obtained using the Z-test within the PC algorithm. This trend is further supported by the lower average SHD values reported in the "Y" column of Table 5.8, particularly in Setups 1 and 2, which involve non-elliptical copulas. Overall, this suggests that Y-vine-based models are structurally closer to the true model than their Z-test based counterparts as already observed in the four-dimensional case. For instance, in Setup 2, Y-vine-based conditional independence tests recover the true CPDAG up to a SHD of one in 56% of cases, whereas the Z-test based PC algorithm only yields a CPDAG with a SHD lower or equal to one in 4% of all cases. The smallest difference between $\text{SHD}^{(Y)}$ and $\text{SHD}^{(Z)}$ is observed in Setup 4, which involves a Gaussian Bayesian network. Here, the average SHD is practically identical for both methods. This result is expected, as Fisher's Z-test for partial correlation is asymptotically exact for testing conditional independence in the Gaussian setting.

Figure 5.6 shows box plots of the AIC and BIC values, revealing that the goodness-of-fit measures for the true and fitted models are highly similar across all setups, with the Y-vine-based approach outperforming the Z-test benchmark notably only in Setup 2. This is corroborated by Table 5.8, which shows similar average log-likelihoods and average AIC and BIC values for Z-test based PCBNs in all setups except for Setup 2.

Setup	Edge	Family	Rotation	Parameters	τ
1	32	t	0°	(0.62, 5)	0.43
	31;2	Clayton	90°	5.07	-0.72
	43	Joe	90°	2.63	-0.47
	65	Joe	90°	3.82	-0.60
	62;5	Frank	0°	-1.62	-0.18
	61;52	Gumbel	180°	1.24	0.19
	64;521	Gaussian	0°	0.55	0.37
2	32	Joe	180°	2.88	0.50
	31;2	Gumbel	180°	1.38	0.27
	43	Gumbel	180°	1.14	0.12
	65	Clayton	90°	0.61	-0.23
	62;5	Joe	270°	5.45	-0.70
	61;52	Frank	0°	-5.24	-0.47
	64;521	Gumbel	270°	1.18	-0.15
3	32	Gaussian	0°	0.39	0.26
	31;2	t	0°	(-0.98, 5)	-0.87
	43	t	0°	(-0.20, 5)	-0.13
	65	t	0°	(0.48, 5)	0.32
	62;5	t	0°	(0.45, 5)	0.30
	61;52	Gaussian	0°	0.80	0.59
	64;521	t	0°	(0.88, 5)	0.69
4	32	Gaussian	0°	-0.53	-0.35
	31;2	Gaussian	0°	0.36	0.23
	43	Gaussian	0°	0.90	0.72
	65	Gaussian	0°	0.73	0.52
	62;5	Gaussian	0°	0.38	0.25
	61;52	Gaussian	0°	-0.68	-0.48
	64;521	Gaussian	0°	0.57	0.39

Table 5.6 Overview of pair-copulas for all setups in the case $d = 6$

5.6 Case 3: Eleven Dimensions

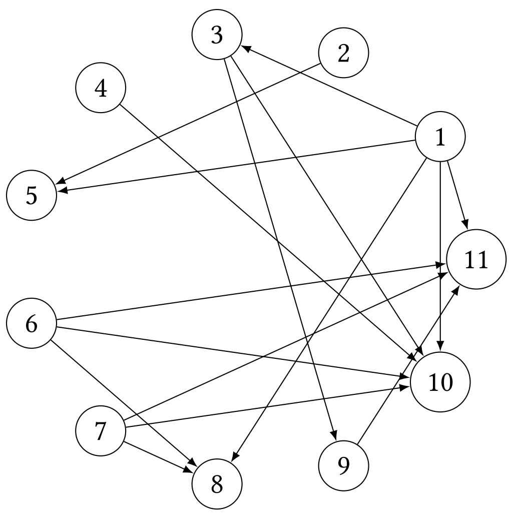


Figure 5.7 An 11-dimensional Bayesian network with 16 edges

Node $i \in \mathcal{V}$	Parent order of i
1	-
2	-
3	1
4	-
5	$2 <_5 1$
6	-
7	-
8	$7 <_8 6 <_8 1$
9	3
10	$7 <_{10} 6 <_{10} 4 <_{10} 1 <_{10} 3$
11	$7 <_{11} 6 <_{11} 1 <_{11} 9$

Table 5.9 Selection of the parent orders of the Bayesian network depicted in Figure 5.7

Setup	Node	Marginal Distribution	Parameters
1	1	Normal	$\mu = \sigma = 1$
	2	t	$\nu = 5$
	3	Lognormal	$\mu = 0, \sigma = 1$
	4	t	$\nu = 5$
	5	Beta	$\alpha = \beta = 0.5$
	6	Lognormal	$\mu = 0, \sigma = 1$
2	1	Lognormal	$\mu = 0, \sigma = 1$
	2	Beta	$\alpha = \beta = 0.5$
	3	Beta	$\alpha = \beta = 0.5$
	4	Beta	$\alpha = \beta = 0.5$
	5	Lognormal	$\mu = 0, \sigma = 1$
	6	Gamma	$k = \theta = 2$
3	1	t	$\nu = 5$
	2	Gamma	$k = \theta = 2$
	3	Gamma	$k = \theta = 2$
	4	t	$\nu = 5$
	5	Normal	$\mu = \sigma = 1$
	6	Normal	$\mu = \sigma = 1$
4	1	Normal	$\mu = \sigma = 1$
	2	Normal	$\mu = \sigma = 1$
	3	Normal	$\mu = \sigma = 1$
	4	Normal	$\mu = \sigma = 1$
	5	Normal	$\mu = \sigma = 1$
	6	Normal	$\mu = \sigma = 1$

Table 5.7 Overview of marginal distributions for all setups in the case $d = 6$

Setup	SHD		Log-Likelihood			AIC			BIC		
	Y	Z	True	Y	Z	True	Y	Z	True	Y	Z
1	2.27	4.33	2531	2209	2208	-5046	-4400	-4392	-5007	-4354	-4334
2	1.69	2.94	1919	1509	1447	-3824	-2996	-2868	-3790	-2941	-2805
3	1.14	1.66	3310	3094	3105	-6595	-6166	-6185	-6536	-6111	-6125
4	1.24	1.23	2059	1859	1864	-4104	-3703	-3711	-4069	-3664	-3672

Table 5.8 Rounded average values of different performance measures for the case $d = 6$. Column "True" serves as a benchmark. Highest log-likelihood and lowest SHD, AIC, and BIC value between Y and Z is colored in blue

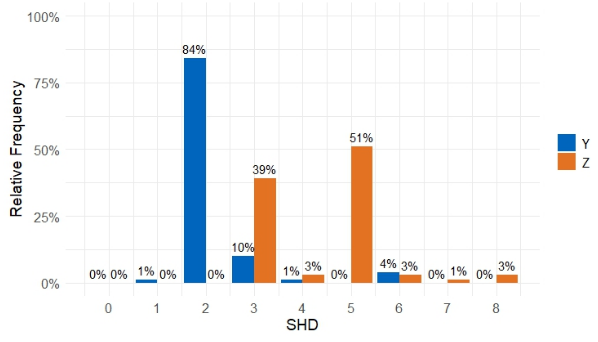
Consider the 11-dimensional Bayesian network depicted in Figure 5.7. Here, $d = 11$, and the graph is represented as $\mathcal{G}^{(\text{true})} = (\mathcal{V}, \mathcal{E}^{(\text{true})})$ where $\mathcal{V} := \{1, \dots, 11\}$ and

$$\begin{aligned}\mathcal{E}^{(\text{true})} :=&\{(1,3),(1,5),(1,8),(1,10),(1,11),(2,5),(3,9),(3,10),(4,10),(6,8),(6,10),(6,11),(7,8),\\ &(7,10),(7,11),(9,11)\}.\end{aligned}$$

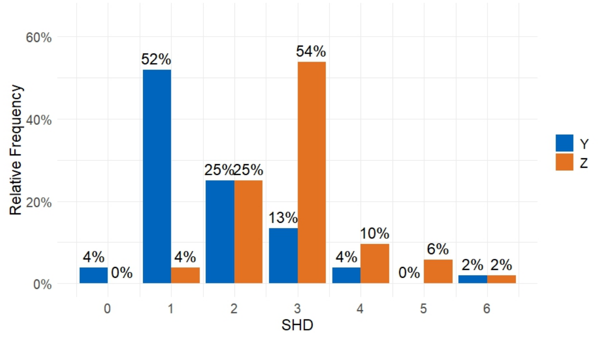
Additionally, a topological order of the nodes is given by

$$7 < 6 < 4 < 2 < 1 < 8 < 5 < 3 < 9 < 11 < 10$$

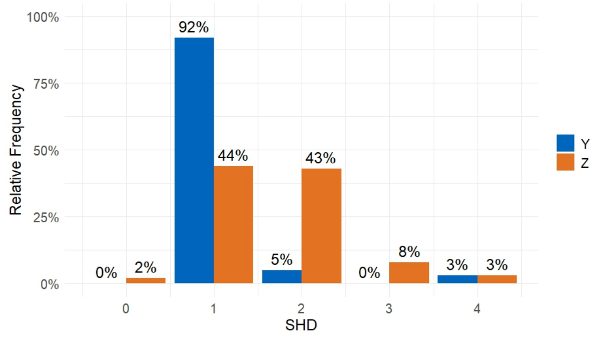
and the fixed parent orders $O^{(\text{true})}$ are illustrated in Table 5.9. In total, there are 9344 implied conditional independence statements. The following list depicts only the conditional independencies associated with missing edges in their most compact form, where the conditioning sets are as small as possible:



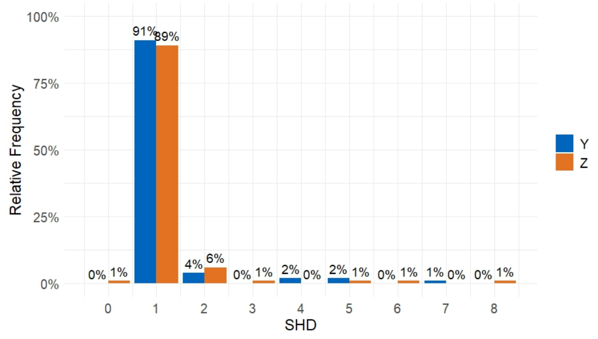
(a) Setup 1: Mixture of Elliptical and Archimedean



(b) Setup 2: Only Archimedean

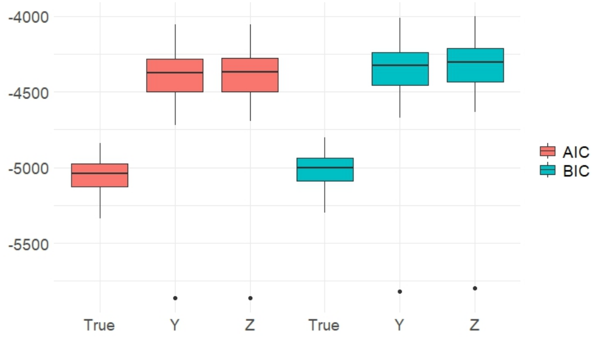


(c) Setup 3: Only Elliptical

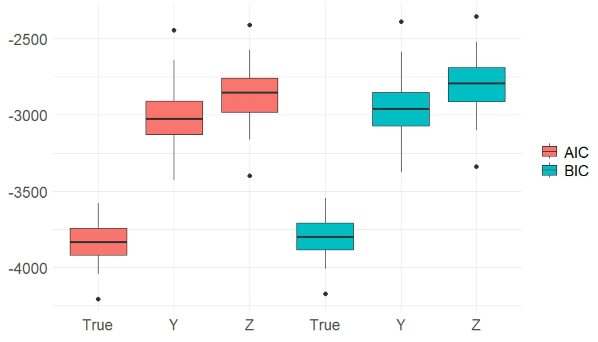


(d) Setup 4: Only Gaussian

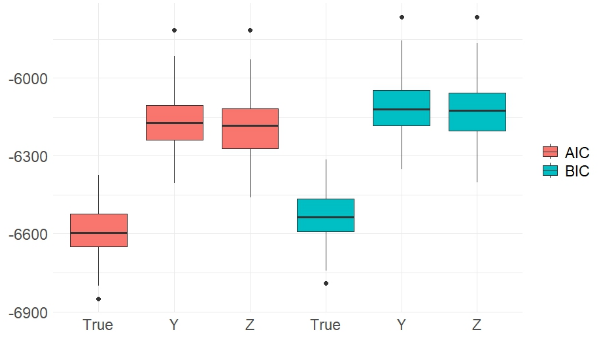
Figure 5.5 Relative frequency of observed SHD values between the Expert CPDAG and the fitted Y-CPDAG (blue), and between the Expert CPDAG and the fitted Z-CPDAG (orange) across different setups for $d = 6$



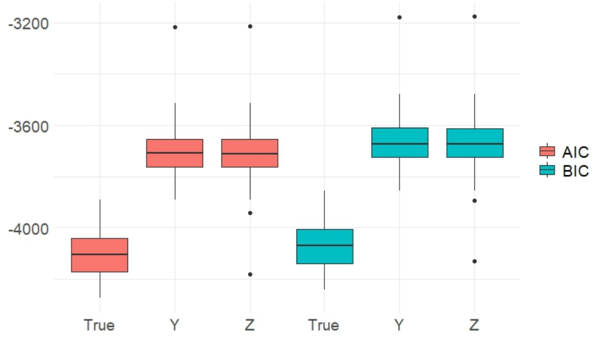
(a) Setup 1: Mixture of Elliptical and Archimedean



(b) Setup 2: Only Archimedean



(c) Setup 3: Only Elliptical



(d) Setup 4: Only Gaussian

Figure 5.6 Box plots of estimated AIC/BIC values for the different setups in the case $d = 6$

i) $X_1 \perp\!\!\!\perp X_j$, where $j \in \{2, 4, 6, 7\}$.

ii) $X_1 \perp\!\!\!\perp X_9 \mid X_3$.

iii) $X_2 \perp\!\!\!\perp X_j$, where $j \in \{3, 4, 6, 7, 8, 9, 10, 11\}$.

iv) $X_3 \perp\!\!\!\perp X_j$, where $j \in \{4, 6, 7\}$.

v) $X_3 \perp\!\!\!\perp X_j \mid X_1$, where $j \in \{5, 8\}$.

vi) $X_3 \perp\!\!\!\perp X_{11} \mid X_1, X_9$.

vii) $X_4 \perp\!\!\!\perp X_j$, where $j \in \{5, 6, 7, 8, 9, 11\}$.

viii) $X_5 \perp\!\!\!\perp X_j$, where $j \in \{6, 7\}$.

ix) $X_5 \perp\!\!\!\perp X_j \mid X_1$, where $j \in \{8, 9, 10, 11\}$.

x) $X_5 \perp\!\!\!\perp X_9 \mid X_3$.

xi) $X_6 \perp\!\!\!\perp X_j$, where $j \in \{7, 9\}$.

xii) $X_7 \perp\!\!\!\perp X_9$.

xiii) $X_8 \perp\!\!\!\perp X_9 \mid X_S$, where $S \in \{1, 3\}$.

xiv) $X_8 \perp\!\!\!\perp X_j \mid X_1, X_6, X_7$, where $j \in \{10, 11\}$.

xv) $X_9 \perp\!\!\!\perp X_{10} \mid X_3$.

xvi) $X_{10} \perp\!\!\!\perp X_{11} \mid X_1, X_6, X_7, X_9$.

xvii) $X_{10} \perp\!\!\!\perp X_{11} \mid X_1, X_3, X_6, X_7$.

The joint density factorizes as follows:

$$f(\mathbf{x}) = \Big(\prod_{i=1}^{11} f_i \Big) \cdot c_{31} \cdot c_{52} \cdot c_{51;2} \cdot c_{87} \cdot c_{86;7} \cdot c_{81;76} \cdot c_{93} \cdot c_{10,7} \cdot c_{10,6;7} \cdot c_{10,4;76} \cdot c_{10,1;764} \cdot c_{10,3;7641}$$
$$\cdot\, c_{11,7} \cdot c_{11,6;7} \cdot c_{11,1;76} \cdot c_{11,9;761}, \quad \mathbf{x} \in \mathbb{R}^{11},$$

where the arguments of the functions are omitted for simplicity.

Tables 5.10 and 5.11 present the specified pair-copulas for each setup described on page 70. The fixed parametric marginal distributions associated with these setups are detailed in Table 5.12. For each of the four setups, the procedure outlined in Figure 5.1 is executed only $N = 50$ instead of 100 times due to the increased computational demand. Each repetition involves $n = 1000$ observations. The tuning parameters and significance levels for each (conditional) independence test are set to $\alpha^Z = \alpha^Y = k^Y = 0.05$.

Figure 5.8 shows the observed relative frequencies of $\mathrm{SHD}_i^{(Y)}$ and $\mathrm{SHD}_i^{(Z)}$ for $i = 1, \dots, N$. Across all setups, the Y-vine-based CPDAGs are structurally closer to the true CPDAG than those fitted using the Z-test, as reflected by the lower average SHD values listed in Table 5.13. The smallest difference in average SHD occurs in Setup 4, corresponding to a Gaussian Bayesian network. Notably, unlike the lower-dimensional cases with $d = 4$ and $d = 6$, neither the Y-vine nor Z-test approaches successfully identify the correct CPDAG in any case, as indicated by the 0% relative frequency for SHD = 0.

Figure 5.9 presents box plots of the AIC and BIC values for each fitted PCBN, compared to the true AIC and BIC values. The averages are provided in Table 5.13. The Y-vine-based models generally yield better fit statistics, with higher average log-likelihoods and lower AIC and BIC values, except in Setup 3. In this setup, despite the Y-vine-based models being structurally closer to the true CPDAG, the Z-test-based PCBNs achieve lower AIC and BIC values. The Y-vine approach notably outperforms the Z-test approach in Setup 2, which involves only Archimedean copulas. However, the box plot in Figure 5.9 reveals that for this specific setup, the lower and upper quartiles of the AIC and BIC values are quite similar.

Setup	Edge	Family	Rotation	Parameters	τ
1	31	Gaussian	0°	-0.45	-0.29
	52	Joe	0°	2.82	0.50
	51;2	Gaussian	0°	0.85	0.65
	87	Clayton	90°	4.08	-0.67
	86;7	Clayton	270°	1.48	-0.42
	81;76	t	0°	(-0.90, 5)	-0.71
	93	t	0°	(0.31, 5)	0.20
	10,7	Clayton	0°	0.28	0.12
	10,6;7	Frank	0°	3.20	0.32
	10,4;76	Gumbel	0°	1.36	0.26
	10,1;764	Gaussian	0°	-0.79	-0.58
	10,3;7641	Clayton	270°	3.26	-0.62
	11,7	Frank	0°	10.99	0.69
	11,6;7	Frank	0°	2.39	0.25
	11,1;76	Gumbel	90°	1.44	-0.31
	11,9;761	Gumbel	0°	3.86	0.74
2	31	Gumbel	270°	1.15	-0.13
	52	Frank	0°	-15.70	-0.77
	51;2	Clayton	270°	2.40	-0.55
	87	Gumbel	90°	3.82	-0.74
	86;7	Frank	0°	4.45	0.42
	81;76	Joe	0°	3.65	0.58
	93	Gumbel	270°	1.74	-0.42
	10,7	Gumbel	180°	1.66	0.40
	10,6;7	Frank	0°	7.12	0.57
	10,4;76	Clayton	90°	1.13	-0.36
	10,1;764	Frank	0°	-2.11	-0.22
	10,3;7641	Gumbel	270°	3.02	-0.67
	11,7	Gumbel	180°	1.55	0.35
	11,6;7	Gumbel	180°	4.86	0.79
	11,1;76	Gumbel	0°	3.16	0.68
	11,9;761	Clayton	270°	7.30	-0.78

Table 5.10 Overview of pair-copulas for Setups 1 and 2 in the case $d = 11$

Setup	Edge	Family	Rotation	Parameters	τ
3	31	Gaussian	0°	-0.79	-0.58
	52	Gaussian	0°	0.32	0.20
	51;2	t	0°	(0.85, 5)	0.65
	87	Gaussian	0°	-0.61	-0.42
	86;7	Gaussian	0°	0.51	0.34
	81;76	t	0°	(0.57, 5)	0.39
	93	t	0°	(0.40, 5)	0.26
	10,7	Gaussian	0°	-0.50	-0.34
	10,6;7	t	0°	(-0.94, 5)	-0.77
	10,4;76	Gaussian	0°	0.41	0.27
	10,1;764	Gaussian	0°	-0.68	-0.47
	10,3;7641	Gaussian	0°	-0.20	-0.13
	11,7	t	0°	(0.50, 5)	0.33
	11,6;7	t	0°	(-0.43, 5)	-0.28
	11,1;76	t	0°	(-0.59, 5)	-0.40
	11,9;761	Gaussian	0°	-0.91	-0.72
4	31	Gaussian	0°	-0.66	-0.45
	52	Gaussian	0°	0.61	0.42
	51;2	Gaussian	0°	0.66	0.46
	87	Gaussian	0°	0.40	0.26
	86;7	Gaussian	0°	0.45	0.30
	81;76	Gaussian	0°	-0.54	-0.37
	93	Gaussian	0°	-0.33	-0.21
	10,7	Gaussian	0°	-0.87	-0.68
	10,6;7	Gaussian	0°	-0.74	-0.53
	10,4;76	Gaussian	0°	0.93	0.76
	10,1;764	Gaussian	0°	-0.74	-0.53
	10,3;7641	Gaussian	0°	0.87	0.67
	11,7	Gaussian	0°	-0.76	-0.55
	11,6;7	Gaussian	0°	-0.39	-0.26
	11,1;76	Gaussian	0°	0.80	0.59
	11,9;761	Gaussian	0°	-0.86	-0.66

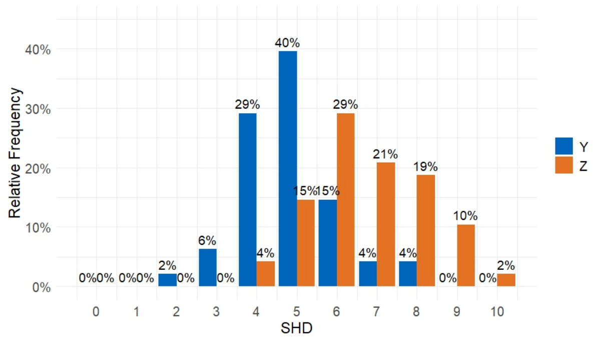
Table 5.11 Overview of pair-copulas for Setups 3 and 4 in the case $d = 11$

Setup	Node	Marginal Distribution	Parameters
1	1	t	$\nu = 5$
	2	Lognormal	$\mu = 0, \sigma = 1$
	3	Normal	$\mu = \sigma = 1$
	4	Lognormal	$\mu = 0, \sigma = 1$
	5	t	$\nu = 5$
	6	t	$\nu = 5$
	7	Gamma	$k = \theta = 2$
	8	Gamma	$k = \theta = 2$
	9	Lognormal	$\mu = 0, \sigma = 1$
	10	Gamma	$k = \theta = 2$
	11	Beta	$\alpha = \beta = 0.5$
2	1	Beta	$\alpha = \beta = 0.5$
	2	Gamma	$k = \theta = 2$
	3	Beta	$\alpha = \beta = 0.5$
	4	Lognormal	$\mu = 0, \sigma = 1$
	5	t	$\nu = 5$
	6	Lognormal	$\mu = 0, \sigma = 1$
	7	Lognormal	$\mu = 0, \sigma = 1$
	8	t	$\nu = 5$
	9	Lognormal	$\mu = 0, \sigma = 1$
	10	t	$\nu = 5$
	11	Normal	$\mu = \sigma = 1$
3	1	Normal	$\mu = \sigma = 1$
	2	Beta	$\alpha = \beta = 0.5$
	3	Lognormal	$\mu = 0, \sigma = 1$
	4	Beta	$\alpha = \beta = 0.5$
	5	Beta	$\alpha = \beta = 0.5$
	6	t	$\nu = 5$
	7	Normal	$\mu = \sigma = 1$
	8	Lognormal	$\mu = 0, \sigma = 1$
	9	Beta	$\alpha = \beta = 0.5$
	10	Normal	$\mu = \sigma = 1$
	11	Lognormal	$\mu = 0, \sigma = 1$
4	1	Normal	$\mu = \sigma = 1$
	2	Normal	$\mu = \sigma = 1$
	3	Normal	$\mu = \sigma = 1$
	4	Normal	$\mu = \sigma = 1$
	5	Normal	$\mu = \sigma = 1$
	6	Normal	$\mu = \sigma = 1$
	7	Normal	$\mu = \sigma = 1$
	8	Normal	$\mu = \sigma = 1$
	9	Normal	$\mu = \sigma = 1$
	10	Normal	$\mu = \sigma = 1$
	11	Normal	$\mu = \sigma = 1$

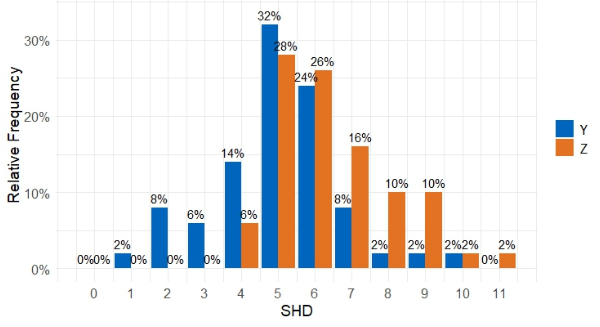
Table 5.12 Overview of marginal distributions for all setups in the case $d = 11$

Setup	SHD		Log-Likelihood			AIC			BIC		
	Y	Z	True	Y	Z	True	Y	Z	True	Y	Z
1	4.88	6.75	6544	5125	5103	-13052	-10206	-10157	-12964	-10095	-10036
2	5.06	6.44	8592	5107	5021	-17153	-10162	-9986	-17074	-10037	-9851
3	2.76	3.34	4901	4343	4451	-9756	-8636	-8848	-9643	-8515	-8714
4	4.75	5.00	6113	4604	4580	-12195	-9172	-9124	-12116	-9085	-9036

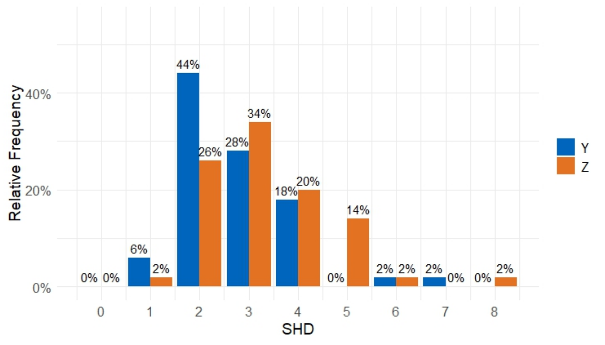
Table 5.13 Rounded average values of different performance measures for the case $d = 11$. Column "True" serves as a benchmark. Highest log-likelihood and lowest SHD, AIC, and BIC value between Y and Z is colored in blue



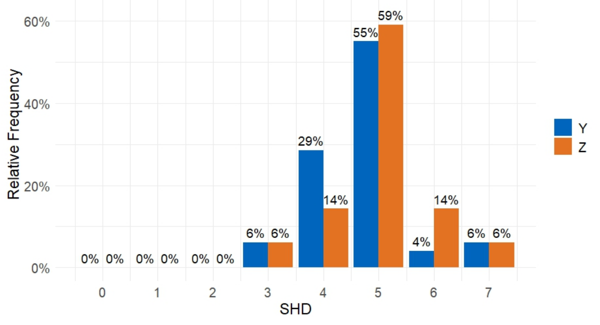
(a) Setup 1: Mixture of Elliptical and Archimedean



(b) Setup 2: Only Archimedean



(c) Setup 3: Only Elliptical



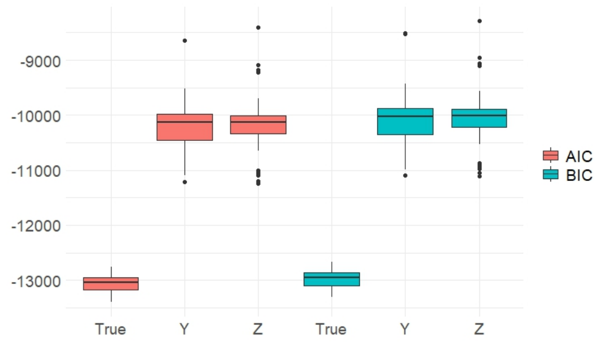
(d) Setup 4: Only Gaussian

Figure 5.8 Relative frequency of observed SHD values between the Expert CPDAG and the fitted Y-CPDAG (blue), and between the Expert CPDAG and the fitted Z-CPDAG (orange) across different setups for $d = 11$

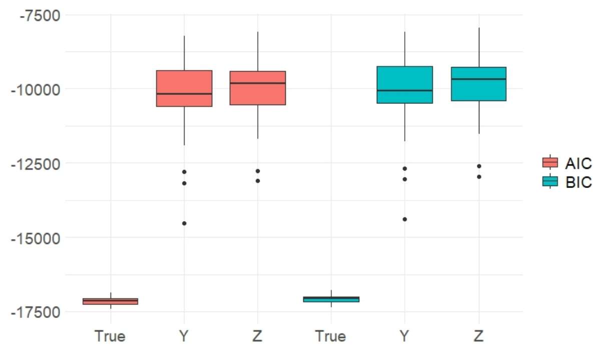
5.7 Computational Performance

Table 5.14 presents the average computational times in seconds for each dimension considered in the simulation studies, detailing the following procedures:

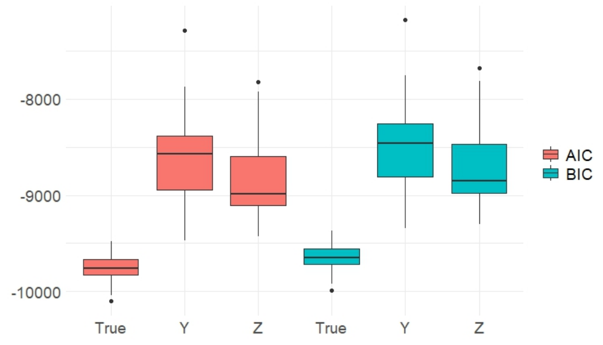
- **Simulation:** Generating observations from a PCBN at the u-level using the approximate simulation method.
- **PC algorithm:**
  - **Y:** A single run of the PC algorithm employing the Y-vine-based conditional independence test to estimate the CPDAG from the simulated data.
  - **Z:** A single run of the PC algorithm using Fisher's Z-test of the partial correlation to estimate the CPDAG from the simulated data.
- **DV-SEM Fitting:** Estimating a DV-SEM for parameter learning, based on a fitted DAG obtained from the PC algorithm and subsequent PDAG extension.



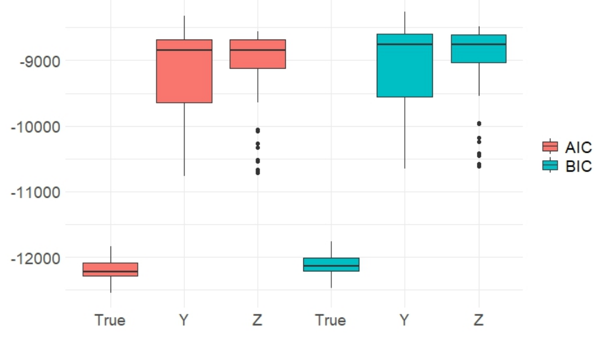
(a) Setup 1: Mixture of Elliptical and Archimedean



(b) Setup 2: Only Archimedean



(c) Setup 3: Only Elliptical



(d) Setup 4: Only Gaussian

Figure 5.9 Box plots of estimated AIC/BIC values for the different setups in the case $d = 11$

Dimension	**Simulation**	**PC algorithm**		**DV-SEM Fitting**
		Y	**Z**	
4	1.76	118.42	0.04	4.91
6	8.46	207.56	0.05	7.44
11	17.82	2528.97	0.50	35.93

Table 5.14 Average computational times in seconds for different steps involved in the simulation studies

The average times were computed for each dimension and averaged across all setups. All computations were performed on a system with the following specifications:

- **Processor (CPU):** Intel Core i5-7200U @ 2.50GHz (2 cores, up to 2.70 GHz).
- **Memory (RAM):** 8 GB.
- **Operating System:** Windows 10, 64-bit.
- **Graphics:** Intel HD Graphics 620 (Integrated GPU).
- **Software Environment:** R version 4.3.2.

The results in Table 5.14 show that the Y-vine-based method requires significantly more time, particularly as the dimensionality increases, with computation times rising from 118 seconds for dimension 4 to 2529 seconds for dimension 11. In contrast, the Z-test approach remains computationally efficient, with times increasing only slightly from 0.04 seconds to 0.50 seconds across the same dimensions. The parameter learning method employing D-vine-based regression demonstrates a moderate increase in computation time as dimensionality grows, but it remains much faster than the structure learning using Y-vines, particularly for larger datasets. Furthermore, the table highlights the computational efficiency of the approximate simulation method, even in higher dimensions, making it a preferable alternative to exact methods that often require computationally expensive multidimensional numerical integration.

6 Data Application: Flight Data Analysis

Aviation safety depends on the continuous monitoring and analysis of flight operations to detect and mitigate potential risks, such as mechanical failures, environmental hazards, and human errors. To enhance safety standards, the International Civil Aviation Organization (ICAO) Annex 6 mandates that airlines implement a flight data analysis program to systematically review flight performance and identify safety concerns. As part of this program, a Quick Access Recorder (QAR) records a comprehensive set of flight parameters—such as altitude, speed, heading, pilot control inputs, engine performance, and the status of critical onboard systems—during each flight (Drees 2016).

This thesis analyzes data from 711 Boeing 747-8 landings at a specific airport, integrating Quick Access Recorder (QAR) data with weather information from METeorological Aerodrome Reports (METAR). The focus is on 12 continuous variables, as detailed in Table 6.1. This dataset has previously been used in Drees (2016) and Wang et al. (2020), who applied physics-based deterministic models to assess runway overrun risks. Additionally, Alnasser and Czado (2022) employed D-vine quantile regression, as discussed in Section 3.1, to model the probability of *th80* exceeding a certain threshold based on other landing parameters. Although Zwirglmaier and Straub (2016) used a Bayesian network for runway overrun prediction, their approach required discretization of continuous data.

This thesis advances existing research by utilizing a pair-copula Bayesian network model with Y-vine structure estimation to identify critical factors affecting flight safety. The proposed model will be evaluated against one derived using Fisher's Z-test within the PC algorithm and another based on expert knowledge of the underlying structure.

Number	Variable	Definition
1	*th80*	Distance from the touchdown point to reach 80 knots in *m*.
2	*hws*	Headwind speed measured at *td* in *m/s*.
3	*temp*	Temperature in *Kelvin* provided by the METeorological Aerodrome Report (METAR).
4	*refAP*	Reference air pressure in *hPa*.
5	*asd*	Deviation in speed between target approach speed and the actual true airspeed at *td* in *m/s*.
6	*trd*	Time reversers deployed after *td* in seconds *s*.
7	*tsd*	Time spoilers deployed after *td* in *s*.
8	*lm*	Landing weight taken at *td* in *kg*.
9	*tbs*	Time brakes started after *td* in *s*.
10	*bd*	Brake duration until 80 *kts* in *s*.
11	*td*	Distance from touchdown zone to touchdown point in *m*.
12	*ea*	Constant deceleration from *td* to 80 *kts* in m/s^2.

Table 6.1 Variable definitions for landing parameters

Variable	Min	Max	Median	Mean	Standard Deviation	Skewness	Kurtosis
th80	-3.65	3.34	0.03	0	1	-0.29	3.27
hws	-3.57	4.18	-0.10	0	1	0.27	3.41
temp	-2.10	3.45	-0.02	0	1	0.41	2.79
refAP	-3.66	2.80	0.02	0	1	-0.29	3.37
asd	-3.47	2.73	-0.10	0	1	0.04	3.07
trd	-1.85	4.68	-0.20	0	1	1.46	5.86
tsd	-3.31	4.60	-0.15	0	1	0.33	5.03
lm	-3.00	1.35	0.09	0	1	-0.58	2.72
tbs	-0.88	5.27	-0.42	0	1	1.98	6.99
bd	-3.22	2.25	0.17	0	1	-0.79	3.42
td	-2.33	3.41	-0.02	0	1	0.28	2.98
ea	-4.27	3.14	-0.06	0	1	-0.29	4.22

Table 6.2 Descriptive statistics of the scaled flight variables. Skewness values outside the range of -0.5 to 0.5 and kurtosis values outside the range of 2 to 4 are highlighted in red

6.1 Exploratory Data Analysis

Let the flight data be presented by $\mathbf{x} = (\mathbf{x}_1^\top, \dots, \mathbf{x}_{711}^\top)$, where each $\mathbf{x}_k = (x_{k,1}, \dots, x_{k,12})^\top$ for $k = 1, \dots, 711$. Prior to fitting graphical models, the data is standardized by:

$$x_{ij}^{(\text{scaled})} := \frac{x_{ij} - \hat{\mu}_j}{\hat{\sigma}_j},$$

where $i = 1, \dots, 711$ and $j = 1, \dots, 12$. The mean $\hat{\mu}_j$ and standard deviation $\hat{\sigma}_j$ for each variable j are computed as:

$$\hat{\mu}_j := \frac{1}{711} \sum_{i=1}^{711} x_{ij}, \quad \text{and} \quad \hat{\sigma}_j := \sqrt{\frac{1}{710} \sum_{i=1}^{711} x_{ij}^2}.$$

Table 6.2 presents the descriptive statistics for the scaled flight variables. The variables *trd*, *lm*, *tbs*, and *bd* exhibit skewness that deviates from normality, while *trd*, *tsd*, *tbs*, and *ea* display kurtosis that is not consistent with a normal distribution.

Furthermore, Figure 6.1 shows the histograms for each variable, overlaid with the fitted kernel density estimation (KDE) and the standard normal distribution. Both the descriptive statistics and the margin plots indicate that the marginal distributions of these variables generally do not follow a normal distribution.

Consequently, non-parametric KDE margins, as described in Section 3.1.1, are fitted to the flight data. These margins are then used to transform the scaled data from the x-level to the u-level using the probability integral transform:

$$\hat{u}_{ij} := \hat{F}_j(x_{ij}^{(\text{scaled})}), \quad \text{for } i = 1, \dots, 711, \ j = 1, \dots, 12,$$

where $\hat{F}_j$ represents the fitted KDE margin for the j-th variable. The variable numbering is detailed in Table 6.1, and the fit statistics are provided in Table 6.3.

Figure 6.2 illustrates the pair plots of the transformed data $\hat{\mathbf{u}}$. Many of the contour plots show deviations from an elliptical shape, suggesting that a Gaussian dependence structure may not be appropriate. For instance, the dependence structure between *tbs* and *bd* exhibits asymmetric tail dependencies. Overall, the data do not appear to follow a multivariate Gaussian distribution, indicating that a pair-copula Bayesian network approach may be more suitable than a classical Gaussian Bayesian network.

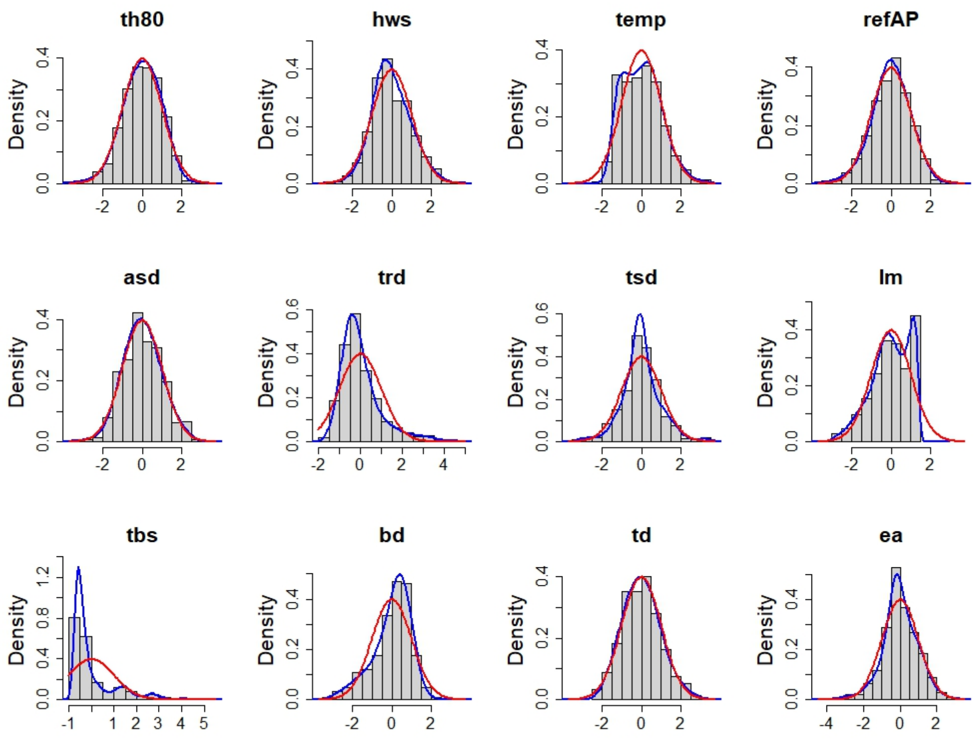


Figure 6.1 Histograms, fitted KDE margins (blue) and fitted Gaussian margins (red) of the scaled flight data

Variable	Log-Likelihood	df	AIC	BIC
th80	-999.98	8.14	2016.23	2053.38
hws	-998.01	7.71	2011.45	2046.65
temp	-971.47	6.52	1955.98	1985.78
refAP	-1000.84	5.09	2011.85	2035.10
asd	-1003.01	4.21	2014.43	2033.66
trd	-890.42	14.22	1809.27	1874.20
tsd	-955.26	10.85	1932.22	1981.77
lm	-893.52	6.34	1799.72	1828.67
tbs	-560.88	13.01	1147.78	1207.18
bd	-955.04	6.13	1922.34	1950.34
td	-1000.83	4.48	2010.62	2031.06
ea	-983.03	11.29	1988.65	2040.23
SUM	**-11212.29**	**97.99**	**22620.55**	**23068.02**

Table 6.3 Fit statistics of the KDE margins for the flight data. The column df shows the degrees of freedom

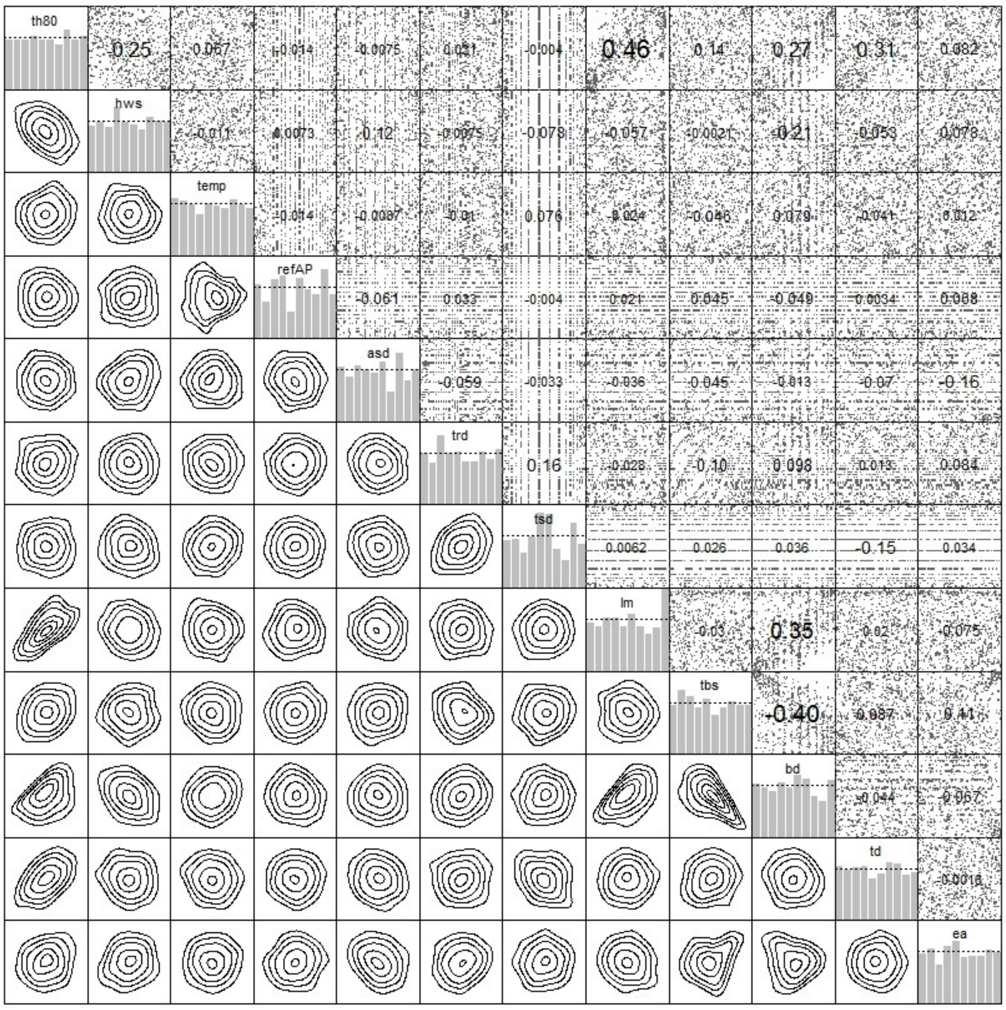


Figure 6.2 Pair plots of the flight data at the u-level using KDE-based marginal distributions

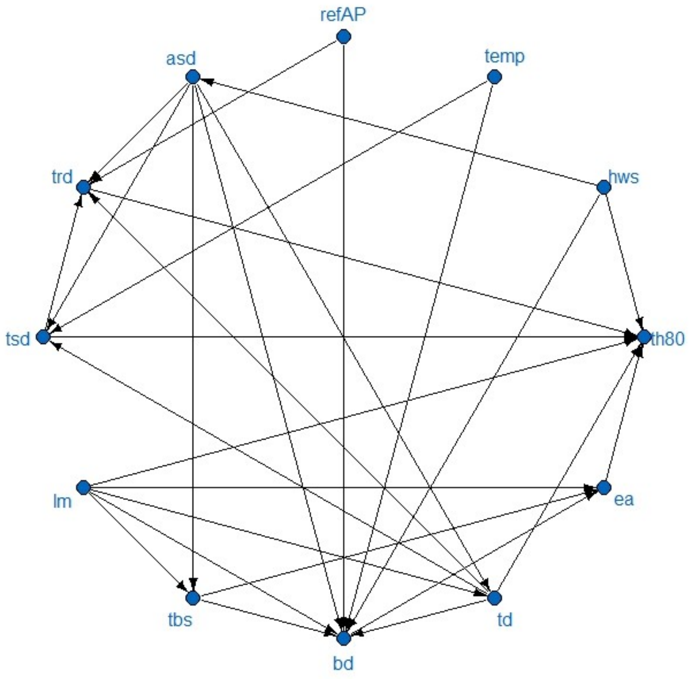


Figure 6.3 Expert DAG for the flight data

Number	Variable	Parent Order
1	*th80*	$lm <_1 td <_1 hws <_1 ea <_1 tsd <_1 trd$
2	*hws*	-
3	*temp*	-
4	*refAP*	-
5	*asd*	*hws*
6	*trd*	$tsd <_6 td <_6 asd <_6 refAP$
7	*tsd*	$td <_7 temp <_7 asd$
8	*lm*	-
9	*tbs*	$asd <_9 lm$
10	*bd*	$tbs <_{10} lm <_{10} hws <_{10} temp <_{10} asd <_{10} refAP <_{10} td$
11	*td*	$asd <_{11} lm$
12	*ea*	$tbs <_{12} lm <_{12} bd$

Table 6.4 Estimated parent orders for the PCBN fitted to the Expert DAG for the flight data

6.2 Model 1: Expert DAG

Consider the DAG shown in Figure 6.3. This graph was developed using expert knowledge from the Flight Dynamics Group of Prof. Holzapfel at the Technical University of Munich. It consists of 28 edges and will be referred to as the **Expert DAG**.

The objective is to fit a pair-copula Bayesian network to the scaled flight data using the Expert DAG. This requires estimating a parent order for each node with multiple parents, as well as determining a bivariate copula associated with each edge in the DAG. To accomplish this, a DV-SEM is constructed sequentially by performing D-vine-based regression for each node, given its parents, as introduced in Section 3.1. In this process, each node serves as the response variable, while its parents act as covariates. The margins are fitted using kernel density estimation, as shown in Figure 6.1.

The construction of the D-vines employs the conditional log-likelihood as the selection criterion, with the family set including all parametric bivariate copulas defined in Appendix B. After fitting each D-vine, the estimated parent order for a node v is determined by the order of its parents in the first tree of the D-vine where v is the response variable. The estimated parent orders are presented in Table 6.4.

Additionally, the pair-copulas used in the decomposition of the estimated PCBN are those found in the first edge of each tree of the fitted D-vines. These pair-copulas are listed in Table 6.5. The product of the 28 pair-copula densities corresponding to the edges in Table 6.5 forms the copula density of the estimated Bayesian network.

It is worth noting that many of the estimated bivariate copulas have a low estimated Kendall's τ-value and consequently a low log-likelihood. If model selection criteria such as AIC or BIC had been used instead of the conditional log-likelihood in the D-vine regression, some variables might not have been selected, suggesting that the parental sets could potentially be reduced, leading to sparser DAGs.

The overall model fit will be presented in Section 6.5 and compared, node-by-node, to the models obtained by applying structure learning algorithms using the Y-test and Z-test, respectively.

6.3 Model 2: Y-DAG

Instead of relying on expert knowledge, the objective is now to learn the structure of the Bayesian network directly from the data. In this section, the Y-vine conditional independence test is employed within the PC algorithm to perform structure learning based on the scaled data. As before, marginal distributions are fitted using kernel density estimation. The significance level, denoted as α^Y, for testing ordinary independence, and the tuning parameter k^Y, used for testing conditional independence with non-empty

Edge	$i, j; S$	Family	Rotation	Parameters	df	τ	Log-Likelihood
$lm \longrightarrow th80$	1,8	BB8	180°	(3.56, 0.84)	2	0.45	198.23
$td \longrightarrow th80$	1,11 ; 8	Frank	0°	4.35	1	0.41	149.12
$hws \longrightarrow th80$	1,2 ; 8,11	Gaussian	0°	-0.49	1	-0.33	96.12
$ea \longrightarrow th80$	1,12 ; 8,11,2	BB1	180°	(0.31, 1.20)	2	0.28	78.31
$tsd \longrightarrow th80$	1,7 ; 8,11,2,12	BB8	0°	(1.07, 1.00)	2	0.04	3.68
$trd \longrightarrow th80$	1,6 ; 8,11,2,12,7	Gaussian	0°	0.06	1	0.04	1.10
$hws \longrightarrow asd$	5,2	BB8	0°	(1.46, 0.87)	2	0.12	14.86
$tsd \longrightarrow trd$	6,7	BB1	180°	(0.05, 1.14)	2	0.15	23.03
$td \longrightarrow trd$	6,11 ; 7	t	0°	(0.08, 14.13)	2	0.05	3.56
$asd \longrightarrow trd$	6,5 ; 7,11	Gaussian	0°	-0.07	1	-0.05	1.93
$refAP \longrightarrow trd$	6,4 ; 7,11,5	BB8	0°	(1.60, 0.35)	2	0.03	0.83
$td \longrightarrow tsd$	7,11	BB7	270°	(1.14, 0.18)	2	-0.15	25.31
$temp \longrightarrow tsd$	7,3 ; 11	BB1	0°	(0.03, 1.04)	2	0.05	4.67
$asd \longrightarrow tsd$	7,5 ; 11,3	BB1	90°	(0.05, 1.02)	2	-0.04	2.03
$asd \longrightarrow tbs$	9,5	t	0°	(0.07, 17.28)	2	0.04	2.21
$lm \longrightarrow tbs$	9,8 ; 5	BB8	270°	(1.21, 0.68)	2	-0.03	0.96
$tbs \longrightarrow bd$	10,9	BB6	90°	(1.76, 1.16)	2	-0.40	192.19
$lm \longrightarrow bd$	10,8 ; 9	BB1	0°	(0.57, 1.29)	2	0.40	169.47
$hws \longrightarrow bd$	10,2 ; 9,8	BB1	270°	(0.06, 1.33)	2	-0.27	68.36
$temp \longrightarrow bd$	10,3 ; 9,8,2	BB8	0°	(1.49, 0.90)	2	0.14	17.27
$asd \longrightarrow bd$	10,5 ; 9,8,2,3	BB8	0°	(1.41, 0.93)	2	0.14	17.11
$refAP \longrightarrow bd$	10,4 ; 9,8,2,3,5	Gaussian	0°	-0.11	1	-0.07	4.19
$td \longrightarrow bd$	10,11 ; 9,8,2,3,5,4	Gaussian	0°	-0.09	1	-0.06	3.05
$asd \longrightarrow td$	11,5	BB7	90°	(1.02, 0.12)	2	-0.07	6.15
$lm \longrightarrow td$	11,8 ; 5	Joe	180°	1.03	1	0.02	1.11
$tbs \longrightarrow ea$	12,9	BB8	0°	(1.38, 1.00)	2	0.17	49.90
$lm \longrightarrow ea$	12,8 ; 9	BB8	90°	(1.14, 0.99)	2	-0.07	5.86
$bd \longrightarrow ea$	12,10 ; 9,8	t	0°	(0.20, 6.04)	2	0.13	21.95

Table 6.5 Summary of estimated pair-copulas associated with the 28 edges of the Expert DAG fitted to the flight data. The numbering of the variables is according to Table 6.1. τ-values outside the range of -0.1 and 0.1 are highlighted in blue

conditioning sets, are both set to 0.05. The selection criterion for fitting Y-vines is the adjusted conditional log-likelihood, and the family set includes all parametric bivariate copulas as defined in Appendix B.

Applying the PC algorithm directly to the data may yield edges or edge directions that are not plausible given the underlying reality represented by the data. Therefore, it is prudent to impose some constraints on the algorithm. This is accomplished by incorporating a blacklist — a set of edges that are prohibited from appearing in the final DAG due to their lack of interpretability. For the flight data, the blacklist consists of the following constraints:

i) *th80* cannot have outgoing edges.

ii) *hws*, *temp*, *refAP* and *lm* cannot have incoming edges.

The first constraint is justified because *th80* represents an outcome variable used to assess the risk of a runway overrun. The variables *hws* (headwind speed), *temp* (temperature), *refAP* (reference air pressure), and *lm* (landing weight) cannot have incoming edges as they represent external conditions or initial settings that influence the landing process but are not themselves influenced by other variables in the network. Consequently, there can also not be any edges between these four variables.

A blacklisted version of the PC algorithm is implemented in the R-package `pcalg` (Kalisch and Bühlmann 2007). However, this package only allows for blacklists that specify the complete absence of certain edges, not constraints on specific edge directions. When an edge is included in the blacklist, it is entirely removed from the initial graph, which typically corresponds to a fully connected undirected graph among all nodes. Other R-packages, such as `bnlearn` (Scutari and Silander 2024), permit directional blacklists within the PC algorithm but do not offer an interface for including user-specified conditional independence tests.

To address these limitations, the first step of the PC algorithm (skeleton estimation) is performed using the `pcalg` package, where the blacklist specifies that no edges are allowed between *hws*, *temp*, *refAP*, and *lm*. The second and third steps of the PC algorithm (identification of v-structures and remaining edge directions) are conducted manually to account for the blacklisted edge directions.

6.3.1 Determination of Edge Directions

The estimated skeleton obtained after applying the PC algorithm with the Y-vine conditional independence test is shown in Figure 6.4. It consists of 18 undirected edges. To manually determine all edge directions, the following heuristics are applied in sequence:

(H1) Edges constrained by the blacklist: Set the directions of edges that are limited to a single possible direction by the blacklist.

(H2) Alignment with the Expert DAG: For edges also present in the Expert DAG, assign the same direction as specified in the Expert DAG. If this introduces a new v-structure, verify whether the d-separation sets obtained from the skeleton estimation support the new v-structure. A v-structure $i \longrightarrow j \longleftarrow k$ is only plausible if $j \notin sepset(i,k)$.

(H3) Remaining edges: Direct any remaining edges, attempting to introduce new v-structures where feasible.

(H1) applies to 11 of the 18 edges. These edges are directed as follows:

- $hws \longrightarrow th80$.
- $hws \longrightarrow ea$.
- $hws \longrightarrow bd$.
- $hws \longrightarrow tsd$.
- $hws \longrightarrow asd$.

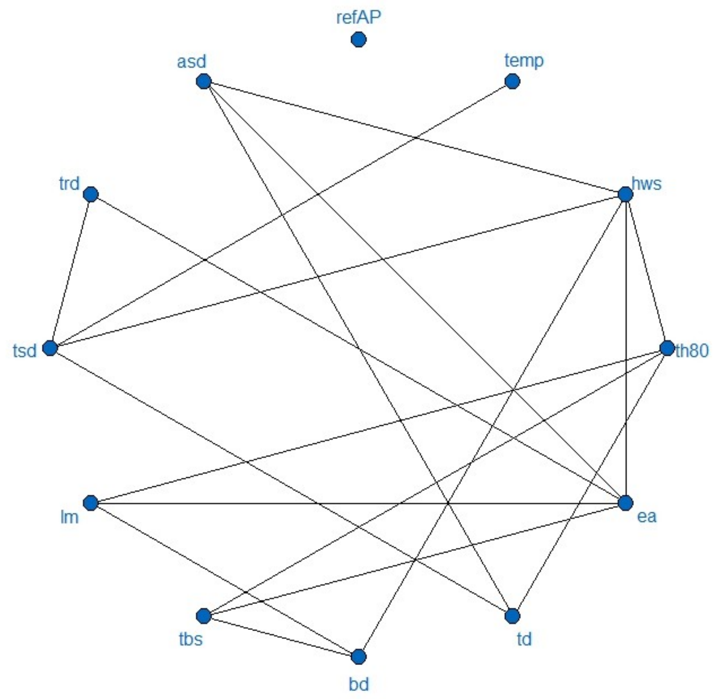


Figure 6.4 Estimated skeleton for the flight data using Y-vine-based conditional independence testing with $\alpha^Y = k^Y = 0.05$

- *temp* $\longrightarrow$ *tsd*.
- *lm* $\longrightarrow$ *th80*.
- *lm* $\longrightarrow$ *ea*.
- *lm* $\longrightarrow$ *bd*.
- *tbs* $\longrightarrow$ *th80*.
- *td* $\longrightarrow$ *th80*.

(H2) applies to 5 of the remaining 7 undirected edges:

- ***td*** $\longrightarrow$ ***tsd***:
  - New v-structure $td \longrightarrow tsd \longleftarrow hws$. It holds that $tsd \notin sepset(td, hws) = \{asd\} \Rightarrow$ **v-structure possible**.
  - New v-structure $td \longrightarrow tsd \longleftarrow temp$. It holds that $tsd \notin sepset(td, temp) = \emptyset \Rightarrow$ **v-structure possible**.

  $\Rightarrow$ Direction *td* $\longrightarrow$ *tsd* is set.
- ***tbs*** $\longrightarrow$ ***bd***:
  - New v-structure $tbs \longrightarrow bd \longleftarrow lm$. It holds that $bd \notin sepset(tbs, lm) = \emptyset \Rightarrow$ **v-structure possible**.
  - New v-structure $tbs \longrightarrow bd \longleftarrow hws$. It holds that $bd \notin sepset(tbs, hws) = \emptyset \Rightarrow$ **v-structure possible**.

  $\Rightarrow$ Direction *tbs* $\longrightarrow$ *bd* is set.
- ***asd*** $\longrightarrow$ ***td***: No new v-structure.

  $\Rightarrow$ Direction *asd* $\longrightarrow$ *td* is set.

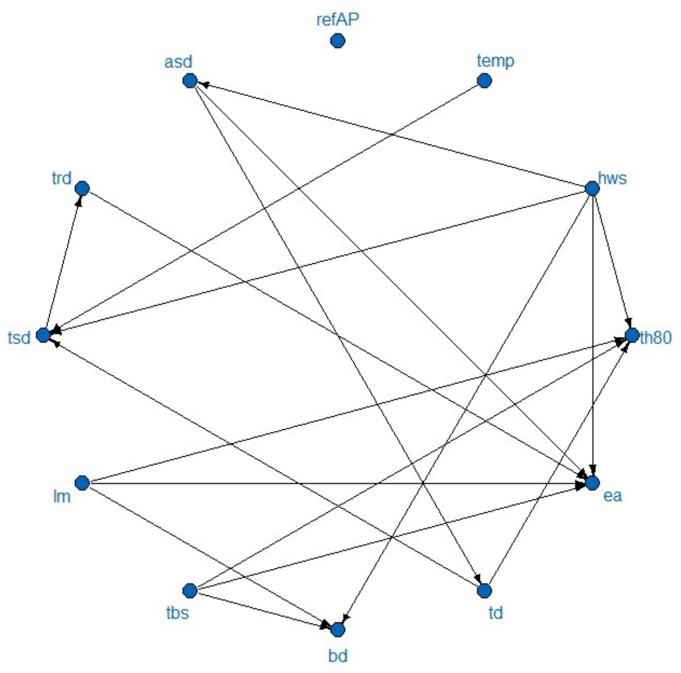


Figure 6.5 Fitted DAG based on the Y-vine conditional independence test for the flight data

- ***tbs* $\longrightarrow$ *ea*:**
  - New v-structure $tbs \longrightarrow ea \longleftarrow lm$. It holds that $ea \notin sepset(tbs, lm) = \emptyset \Rightarrow$ **v-structure possible**.
  - New v-structure $tbs \longrightarrow ea \longleftarrow hws$. It holds that $ea \notin sepset(tbs, hws) = \emptyset \Rightarrow$ **v-structure possible**.

  $\Rightarrow$ Direction $tbs \longrightarrow ea$ is set.

- ***tsd* $\longrightarrow$ *trd*:** No new v-structure.

  $\Rightarrow$ Direction $tsd \longrightarrow trd$ is set.

After applying (H2), the remaining undirected edges are $asd - ea$ and $trd - ea$. Considering the variable definitions, the only reasonable directions are $asd \longrightarrow ea$ and $trd \longrightarrow ea$. It remains to verify whether the new v-structure $asd \longrightarrow ea \longleftarrow trd$ is possible. Since $ea \notin sepset(asd, trd) = \{hws\}$, this is indeed the case, and these edges are directed accordingly.

Figure 6.5 illustrates the final DAG after all edges have been directed. This DAG will henceforth be referred to as the **Y-DAG**, as its skeleton estimation is based on Y-vine independence testing.

6.3.2 Parameter Learning

The next objective is to estimate the pair-copulas corresponding to the 18 edges of the fitted Y-DAG. The procedure follows the same steps as those used for the Expert DAG described in Section 6.2. Specifically, a DV-SEM will be fitted to the Y-DAG by applying D-vine-based regression for each node conditional on its parents.

The resulting estimated parent order for each node is provided in Table 6.6, while the estimated pair-copulas associated with each edge are detailed in Table 6.7. Compared to the expert model, the Y-vine-based model is more parsimonious, featuring fewer edges and fitting fewer copulas with small absolute τ-values. A comprehensive comparison of all models will be presented in Section 6.5.

Number	Variable	Parent Order
1	*th80*	$lm <_1 td <_1 hws <_1 tbs$
2	*hws*	-
3	*temp*	-
4	*refAP*	-
5	*asd*	*hws*
6	*trd*	*tsd*
7	*tsd*	$td <_7 hws <_7 temp$
8	*lm*	-
9	*tbs*	-
10	*bd*	$tbs <_{10} lm <_{10} hws$
11	*td*	*asd*
12	*ea*	$tbs <_{12} asd <_{12} trd <_{12} hws <_{12} lm$

Table 6.6 Estimated parent orders for the PCBN fitted to the estimated Y-DAG for the flight data

Edge	*i, j*; S	Family	Rotation	Parameters	df	τ	Log-Likelihood
$lm \longrightarrow th80$	1,8	BB8	180°	(3.56, 0.84)	2	0.45	198.23
$td \longrightarrow th80$	1,11 ; 8	Frank	0°	4.35	1	0.41	149.12
$hws \longrightarrow th80$	1,2 ; 8,11	Gaussian	0°	-0.49	1	-0.33	96.12
$tbs \longrightarrow th80$	1,9 ; 8,11,2	BB1	180°	(0.40, 1.08)	2	0.23	56.69
$hws \longrightarrow asd$	5,2	BB8	0°	(1.46, 0.87)	2	0.12	14.86
$tsd \longrightarrow trd$	6,7	BB1	180°	(0.05, 1.14)	2	0.15	23.03
$td \longrightarrow tsd$	7,11	BB7	270°	(1.14, 0.18)	2	-0.15	25.31
$hws \longrightarrow tsd$	7,2 ; 11	BB8	90°	(1.32, 0.90)	2	-0.09	9.16
$temp \longrightarrow tsd$	7,3 ; 11,2	BB1	0°	(0.03, 1.04)	2	0.05	4.45
$tbs \longrightarrow bd$	10,9	BB6	90°	(1.76, 1.16)	2	-0.40	192.19
$lm \longrightarrow bd$	10,8 ; 9	BB1	0°	(0.57, 1.29)	2	0.40	169.47
$hws \longrightarrow bd$	10,2 ; 9,8	BB1	270°	(0.06, 1.33)	2	-0.27	68.36
$asd \longrightarrow td$	11,5	BB7	90°	(1.02, 0.12)	2	-0.07	6.15
$tbs \longrightarrow ea$	12,9	BB8	0°	(1.38, 1.00)	2	0.17	49.90
$asd \longrightarrow ea$	12,5 ; 9	BB8	270°	(3.27, 0.48)	2	-0.18	24.28
$trd \longrightarrow ea$	12,6 ; 9,5	BB8	0°	(2.20, 0.56)	2	0.12	12.00
$hws \longrightarrow ea$	12,2 ; 9,5,6	Clayton	0°	0.19	1	0.09	9.93
$lm \longrightarrow ea$	12,8 ; 9,5,6,2	BB8	90°	(1.30, 0.83)	2	-0.07	4.98

Table 6.7 Summary of estimated pair-copulas associated with the 18 edges of the Y-DAG fitted to the flight data. The numbering of the variables is according to Table 6.1. τ-values outside the range of -0.1 and 0.1 are highlighted in blue

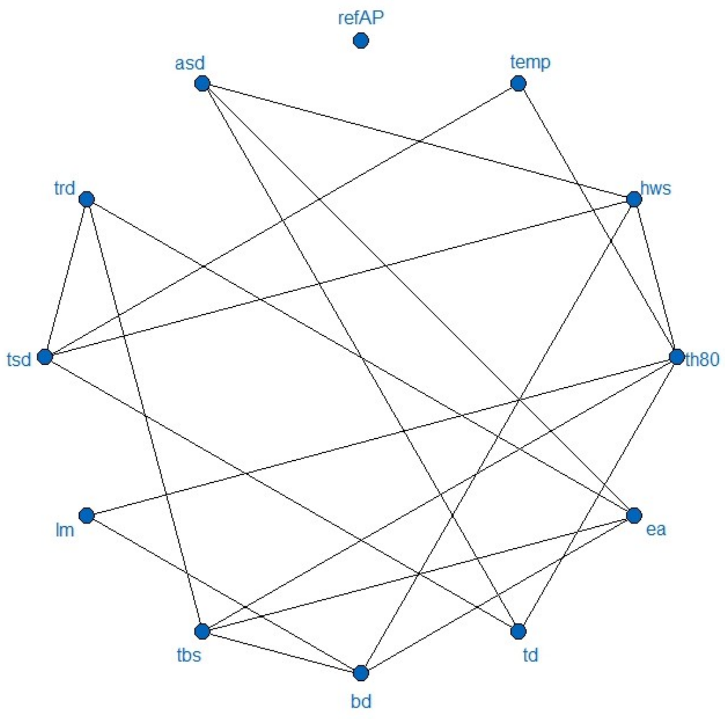


Figure 6.6 Estimated skeleton for the flight data using Fisher's Z-test with $\alpha^Z = 0.05$

6.4 Model 3: Z-DAG

The PC algorithm is applied again to learn the structure of the Bayesian network using the scaled flight data, but this time, Fisher's Z-test for partial correlation is used to assess conditional independence. The significance level for the Z-test is set to $\alpha^Z = 0.05$. The same blacklist as in the previous section is utilized. As before, the directions of the edges must be determined manually after estimating the skeleton.

6.4.1 Determination of Edge Directions

Figure 6.6 shows the estimated skeleton after applying the first step of the PC algorithm using Fisher's Z-test for conditional independence. The graph contains 19 undirected edges. The same heuristics outlined in Section 6.3.1 are now applied to determine the edge directions.

(H1) applies to 10 of the 19 edges, which are directed as follows:

- $hws \longrightarrow th80$.
- $hws \longrightarrow bd$.
- $hws \longrightarrow tsd$.
- $hws \longrightarrow asd$.
- $temp \longrightarrow th80$.
- $temp \longrightarrow tsd$.
- $lm \longrightarrow th80$.
- $lm \longrightarrow bd$.
- $tbs \longrightarrow th80$.
- $td \longrightarrow th80$.

(H2) is then applied to 6 of the remaining 9 undirected edges:

- ***tsd* → *trd*:** No new v-structure.
  ⇒ Direction $tsd \rightarrow trd$ is set.
- ***td* → *tsd*:**
  - New v-structure $td \rightarrow tsd \leftarrow hws$. It holds that $tsd \notin sepset(td, hws) = \{asd\}$ ⇒ **v-structure possible**.
  - New v-structure $td \rightarrow tsd \leftarrow temp$. It holds that $tsd \notin sepset(td, temp) = \emptyset$ ⇒ **v-structure possible**.

  ⇒ Direction $td \rightarrow tsd$ is set.
- ***tbs* → *bd*:**
  - New v-structure $tbs \rightarrow bd \leftarrow lm$. It holds that $bd \notin sepset(tbs, lm) = \emptyset$ ⇒ **v-structure possible**.
  - New v-structure $tbs \rightarrow bd \leftarrow hws$. It holds that $bd \notin sepset(tbs, hws) = \{ea\}$ ⇒ **v-structure possible**.

  ⇒ Direction $tbs \rightarrow bd$ is set.
- ***asd* → *td*:** No new v-structure.
  ⇒ Direction $asd \rightarrow td$ is set.
- ***tbs* → *ea*:** No new v-structure.
  ⇒ Direction $tbs \rightarrow ea$ is set.
- ***bd* → *ea*:** No new v-structure.
  ⇒ Direction $bd \rightarrow ea$ is set.

After applying (H2), the remaining undirected edges are $asd - ea$, $trd - ea$, and $trd - tbs$. First, note that the v-structure $trd \rightarrow ea \leftarrow asd$ is not possible since $sepset(trd, asd) = \{ea\}$. Additionally, setting $trd \rightarrow ea \rightarrow asd$ would create a directed cycle. Therefore, $trd \leftarrow ea$ must be assigned. This leaves two possible configurations: $ea \leftarrow asd$ or $ea \rightarrow asd$. In both scenarios, the direction $tbs \rightarrow trd$ must be set to avoid creating a directed cycle. The direction $ea \leftarrow asd$ appears more plausible. It introduces the v-structures $asd \rightarrow ea \leftarrow tbs$ and $asd \rightarrow ea \leftarrow bd$. These are valid because $ea \notin sepset(asd, tbs) = sepset(asd, bd) = \emptyset$.

Thus, the final directions for these edges are:

- $asd \rightarrow ea$.
- $ea \rightarrow trd$.
- $tbs \rightarrow trd$.

Figure 6.7 shows the final DAG after all edges have been assigned directions. This DAG will be referred to as the **Z-DAG**, reflecting that its skeleton was estimated using Fisher's Z-test for partial correlation.

6.4.2 Parameter Learning

As with the previous two models, the pair-copulas associated with the 19 edges of the fitted Z-DAG are estimated. The estimation procedure follows the same steps outlined for the Expert DAG in Section 6.2. In particular, a DV-SEM is fitted to the Z-DAG using D-vine-based regression for each node, given its parent nodes. The estimated parent order for each node is shown in Table 6.8, and the corresponding pair-copulas for each edge are listed in Table 6.9. Like the Y-vine-based model, the Z-test-based model is more parsimonious than the expert model, containing fewer edges and fitting fewer copulas with small absolute τ-values.

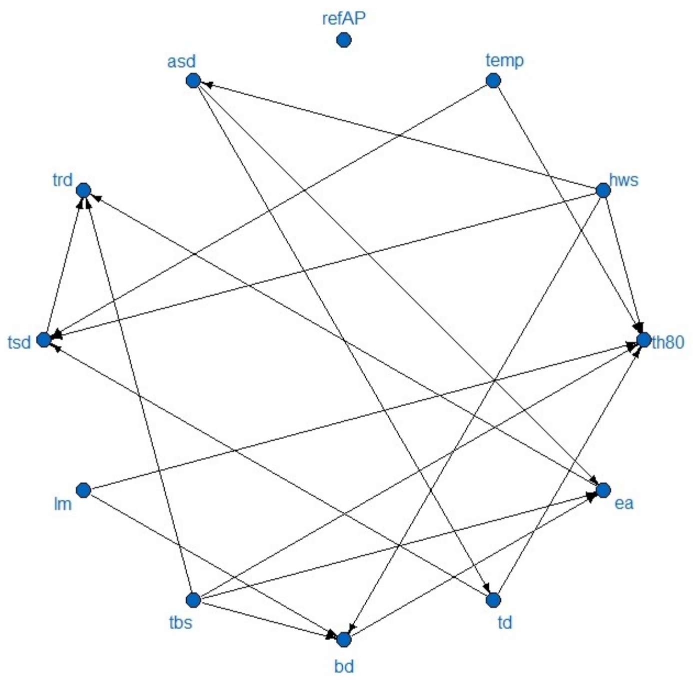


Figure 6.7 Fitted DAG based on Fisher's Z-test for the flight data

Number	Variable	Parent Order
1	*th80*	$lm <_1 td <_1 hws <_1 tbs <_1 temp$
2	*hws*	-
3	*temp*	-
4	*refAP*	-
5	*asd*	*hws*
6	*trd*	$tsd <_6 tbs <_6 ea$
7	*tsd*	$td <_7 hws <_7 temp$
8	*lm*	-
9	*tbs*	-
10	*bd*	$tbs <_{10} lm <_{10} hws$
11	*td*	*asd*
12	*ea*	$tbs <_{12} asd <_{12} bd$

Table 6.8 Estimated parent orders for the PCBN fitted to the estimated Z-DAG for the flight data

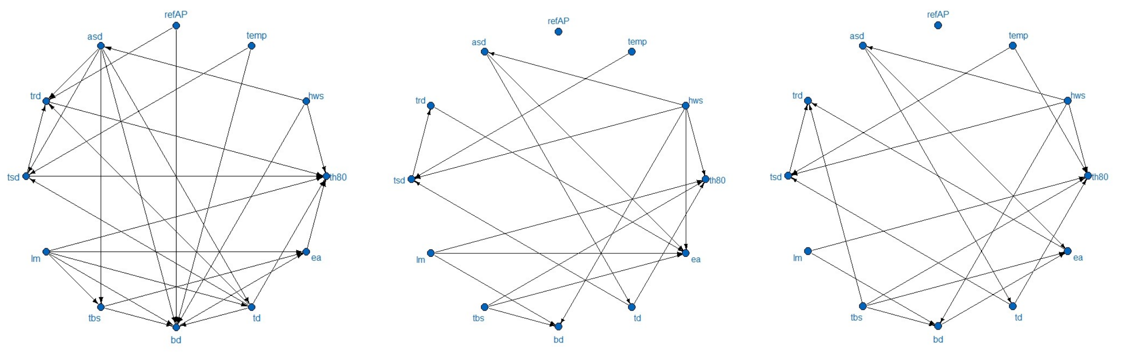
Edge	$i, j; S$	Family	Rotation	Parameters	df	τ	Log-Likelihood
$lm \longrightarrow th80$	1,8	BB8	180°	(3.56, 0.84)	2	0.45	198.23
$td \longrightarrow th80$	1,11 ; 8	Frank	0°	4.35	1	0.41	149.12
$hws \longrightarrow th80$	1,2 ; 8,11	Gaussian	0°	-0.49	1	-0.33	96.12
$tbs \longrightarrow th80$	1,9 ; 8,11,2	BB1	180°	(0.40, 1.08)	2	0.23	56.69
$temp \longrightarrow th80$	1,3 ; 8,11,2,9	Gaussian	0°	0.29	1	0.19	30.73
$hws \longrightarrow asd$	5,2	BB8	0°	(1.46, 0.87)	2	0.12	14.86
$tsd \longrightarrow trd$	6,7	BB1	180°	(0.05, 1.14)	2	0.15	23.03
$tbs \longrightarrow trd$	6,9 ; 7	BB8	90°	(1.56, 0.80)	2	-0.12	12.74
$ea \longrightarrow trd$	6,12 ; 7,9	BB8	0°	(2.89, 0.43)	2	0.13	12.63
$td \longrightarrow tsd$	7,11	BB7	270°	(1.14, 0.18)	2	-0.15	25.31
$hws \longrightarrow tsd$	7,2 ; 11	BB8	90°	(1.32, 0.90)	2	-0.09	9.16
$temp \longrightarrow tsd$	7,3 ; 11,2	BB1	0°	(0.03, 1.04)	2	0.05	4.45
$tbs \longrightarrow bd$	10,9	BB6	90°	(1.76, 1.16)	2	-0.40	192.19
$lm \longrightarrow bd$	10,8 ; 9	BB1	0°	(0.57, 1.29)	2	0.40	169.47
$hws \longrightarrow bd$	10,2 ; 9,8	BB1	270°	(0.06, 1.33)	2	-0.27	68.36
$asd \longrightarrow td$	11,5	BB7	90°	(1.02, 0.12)	2	-0.07	6.15
$tbs \longrightarrow ea$	12,9	BB8	0°	(1.38, 1.00)	2	0.17	49.90
$asd \longrightarrow ea$	12,5 ; 9	BB8	270°	(3.27, 0.48)	2	-0.18	24.28
$bd \longrightarrow ea$	12,10 ; 9,5	t	0°	(0.11, 8.99)	2	0.07	8.50

Table 6.9 Summary of estimated pair-copulas associated with the 19 edges of the Z-DAG fitted to the flight data. The numbering of the variables is according to Table 6.1. τ-values outside the range of -0.1 and 0.1 are highlighted in blue

6.5 Model Comparison

Finally, this section compares the expert, Y-, and Z-models. Unlike in the simulation studies, the true underlying DAG is unknown here, making it difficult to determine which structure is closest to the true one using standard measures like the Structural Hamming Distance. Therefore, the SHD can only be used to compare the models relative to each other. The SHD values are as follows:

- SHD(Expert DAG, Y-DAG)= 21.
- SHD(Expert DAG, Z-DAG)= 22.
- SHD(Y-DAG, Z-DAG)= 6.



(a) Expert DAG (28 edges) **(b)** Y-DAG (18 edges) **(c)** Z-DAG (19 edges)

Figure 6.8 Comparison of the Expert DAG, Y-DAG, and Z-DAG

Variable	Parent Order Expert	Parent Order Y	Parent Order Z
th80	*lm, td, hws, ea, tsd, trd*	*lm, td, hws, tbs*	*lm, td, hws, tbs, temp*
hws	-	-	-
temp	-	-	-
refAP	-	-	-
asd	*hws*	*hws*	*hws*
trd	*tsd, td, asd, refAP*	*tsd*	*tsd, tbs, ea*
tsd	*td, temp, asd*	*td, hws, temp*	*td, hws, temp*
lm	-	-	-
tbs	*asd, lm*	-	-
bd	*tbs, lm, hws, temp, asd, refAP, td*	*tbs, lm, hws*	*tbs, lm, hws*
td	*asd, lm*	*asd*	*asd*
ea	*tbs, lm, bd*	*tbs, asd, trd, hws, lm*	*tbs, asd, bd*

Table 6.10 Comparison of the parent orders for the Expert DAG, Y-DAG, and Z-DAG of the flight data

Variable	Conditional Log-Likelihood			df		
	Expert	Y	Z	Expert	Y	Z
th80	526.57	500.16	530.89	9	6	7
hws	-	-	-	-	-	-
temp	-	-	-	-	-	-
refAP	-	-	-	-	-	-
asd	14.86	14.86	14.86	2	2	2
trd	29.36	23.03	48.41	7	2	6
tsd	32.01	38.92	38.92	6	6	6
lm	-	-	-	-	-	-
tbs	3.17	-	-	4	-	-
bd	471.64	430.02	430.02	12	6	6
td	7.27	6.15	6.15	3	2	2
ea	77.70	101.09	82.68	6	9	6
SUM	1162.57	1114.24	1151.94	49	33	35

Table 6.11 Comparison of the conditional copula log-likelihood and degrees of freedom for each variable across the Expert, Y-, and Z-model of the flight data

These values indicate that the Y-DAG and Z-DAG are structurally more similar to each other than to the Expert DAG. Both Y- and Z-DAGs have significantly fewer edges than the Expert DAG, as illustrated in Figure 6.8. Notably, in both the Y-DAG and Z-DAG, the node *refAP* is isolated, suggesting that *refAP* is independent of all other nodes in the network. This outcome reflects the results of the conditional independence tests.

Table 6.10 compares the estimated parent orders across all three models. Overall, these orders exhibit some similarities, but there are notable exceptions, such as for the nodes *tbs* and *bd*. In the DAGs where the PC algorithm was applied, *tbs* has no parents, unlike in the Expert DAG. Additionally, in the Expert DAG, the node *bd* has seven incoming edges, whereas in the Y-DAG and Z-DAG, only the first three parents from the Expert DAG's parent order are retained.

Table 6.11 provides the conditional copula log-likelihood for each node given its parents in all three models, along with the respective degrees of freedom. Table 6.12 presents the conditional AIC and BIC values. Overall, the PCBN model fitted to the Expert DAG shows the highest estimated log-likelihood, while the PCBN model fitted to the Z-DAG exhibits the lowest AIC and BIC values. The Y-DAG model

Variable	Conditional AIC			Conditional BIC		
	Expert	Y	Z	Expert	Y	Z
th80	-1035.13	-988.33	-1047.79	-994.03	-960.93	-1015.82
hws	-	-	-	-	-	-
temp	-	-	-	-	-	-
refAP	-	-	-	-	-	-
asd	-25.72	-25.72	-25.72	-16.58	-16.58	-16.58
trd	-44.71	-42.07	-84.83	-12.74	-32.93	-57.43
tsd	-52.02	-65.85	-65.85	-24.62	-38.45	-38.45
lm	-	-	-	-	-	-
tbs	1.67	-	-	19.93	-	-
bd	-919.29	-848.04	-848.04	-864.49	-820.64	-820.64
td	-8.53	-8.30	-8.30	5.17	0.83	0.83
ea	-143.41	-184.19	-153.36	-116.00	-143.09	-125.96
SUM	-2227.14	-2162.49	-2233.87	-2003.37	-2011.79	-2074.04

Table 6.12 Comparison of the conditional AIC and BIC for each variable across the Expert, Y- and Z-model

shows a better conditional fit than the other two models only for the node *ea*, but it lags behind for the node *th80*.

Comparing the Z-DAG to the Y-DAG suggests that including the edge *temp* $\longrightarrow$ *th80* could improve the overall fit of the Y-vine-based model. Notably, the Expert DAG is the only model in which the node *tbs* has parents, but this does not appear to be supported by the data, as the conditional AIC and BIC for this node are positive. Overall, after accounting for the degrees of freedom, the more parsimonious models based on the Y- or Z-tests are favored over the expert model due to their simplicity and better penalized fit statistics.

7 Data Application: Sachs Data Analysis

The Sachs dataset is a benchmark dataset widely used in causal inference and graphical modeling. It originates from a study by Sachs et al. (2005), which aimed to explore the causal relationships between various molecular entities in human cells. This dataset contains observations on protein expression levels under different experimental conditions, offering insights into the interactions and causal dependencies among these proteins. One notable experiment, "CD3CD28+AktInhib", involves stimulating T cells with antibodies against CD3 and CD28 to mimic an activation signal, combined with the application of an Akt inhibitor to examine how inhibiting the Akt signaling pathway affects cellular responses in activated T cells (Sachs et al. 2005).

The dataset from this experiment comprises $n = 911$ observations (cells) and $d = 11$ variables (proteins). It was previously analyzed by Czado and Scharl (2021), who fitted a DV-SEM to an expert DAG using D-vine-based regression for each node conditional on its parents. Their approach utilized the BIC for model selection, which led to some parents not being included in the D-vines and resulted in fewer edges compared to the expert DAG.

This thesis builds on that research by applying the PC algorithm, utilizing the Y-vine conditional independence test and Fisher's Z-test for partial correlation, to determine the structure of the Bayesian network without relying on expert knowledge. Additionally, D-vine-based regression will be employed, as done with the flight data, for parameter learning. Various choices for tuning parameters and significance levels for conditional independence testing will be compared. The expert DAG from Sachs et al. (2005) will be used as a traditional benchmark to evaluate and compare the fitted structures.

7.1 Exploratory Data Analysis

Let the Sachs data be presented by $\mathbf{x} = (\mathbf{x}_1^\top, \dots, \mathbf{x}_{911}^\top)$, where each $\mathbf{x}_k = (x_{k,1}, \dots, x_{k,11})^\top$ for $k = 1, \dots, 911$. Prior to fitting graphical models, the data is log transformed and standardized by:

$$x_{ij}^{(\text{log,scaled})} := \frac{\ln(x_{ij}) - \hat{\mu}_j^{\log}}{\hat{\sigma}_j^{\log}},$$

where $i = 1, \dots, 911$ and $j = 1, \dots, 11$. The mean $\hat{\mu}_j^{\log}$ and standard deviation $\hat{\sigma}_j^{\log}$ for each variable j are computed as:

$$\hat{\mu}_j^{\log} := \frac{1}{911} \sum_{i=1}^{911} \ln(x_{ij}), \quad \text{and} \quad \hat{\sigma}_j^{\log} := \sqrt{\frac{1}{910} \sum_{i=1}^{911} \ln(x_{ij})^2}.$$

Figure 7.1 shows the histograms for each variable, overlaid with the fitted kernel density estimation (KDE) and the standard normal distribution. The margin plots indicate that the marginal distributions of these variables generally do not follow a normal distribution as first observed by Czado and Scharl (2021).

Consequently, non-parametric KDE margins, as described in Section 3.1.1, are fitted to the log-transformed and scaled Sachs data. These margins are then used to transform the data from the x-level to the u-level using the probability integral transform:

$$\hat{u}_{ij} := \hat{F}_j(x_{ij}^{(\text{log,scaled})}), \quad \text{for } i = 1, \dots, 911,\ j = 1, \dots, 11,$$

where $\hat{F}_j$ represents the fitted KDE margin for the j-th variable. The variable numbering and the fit statistics are provided in Table 7.1.

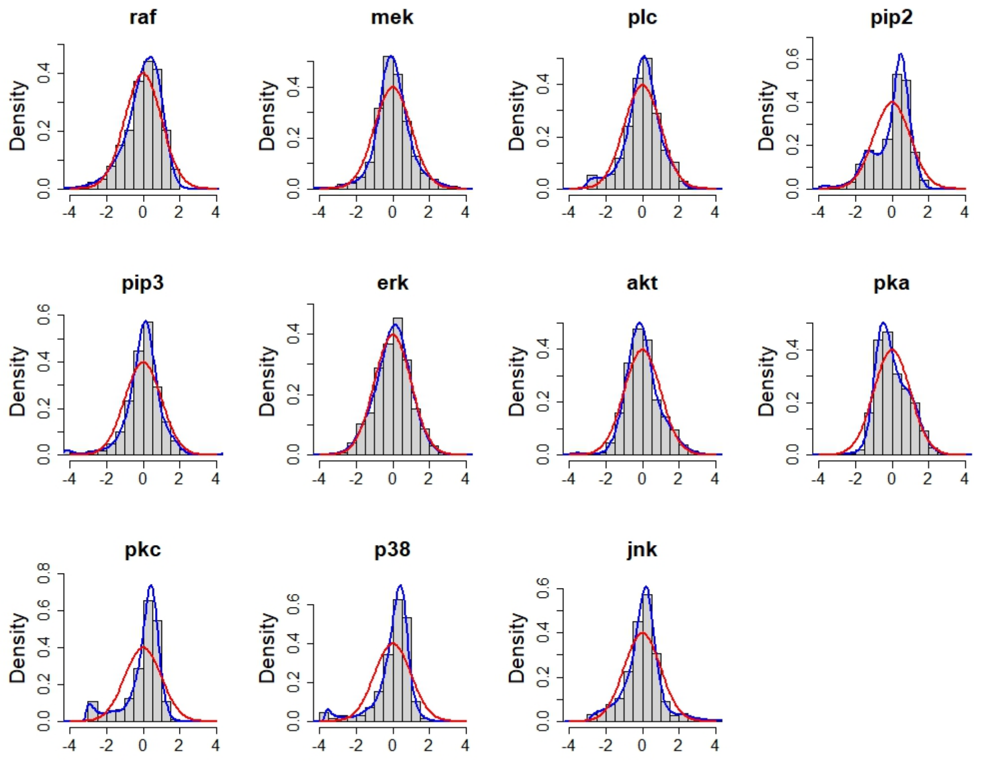


Figure 7.1 Histograms, fitted KDE margins (blue) and fitted Gaussian margins (red) of the log-transformed and scaled Sachs data

Number	Variable	Log-Likelihood	df	AIC	BIC
1	*raf*	-1228.56	7.23	2471.57	2506.38
2	*mek*	-1220.18	11.07	2462.50	2515.82
3	*plc*	-1236.08	6.70	2485.56	2517.79
4	*pip2*	-1155.21	8.65	2327.71	2369.35
5	*pip3*	-1179.64	11.69	2382.65	2438.93
6	*erk*	-1283.88	7.77	2583.29	2620.69
7	*akt*	-1228.90	12.88	2483.56	2545.55
8	*pka*	-1177.20	12.67	2379.74	2440.75
9	*pkc*	-1007.29	6.83	2028.25	2061.13
10	*p38*	-1032.90	8.51	2082.81	2123.79
11	*jnk*	-1192.28	10.68	2405.92	2457.34
Σ	**SUM**	**-12942.11**	**104.67**	**26093.56**	**26597.52**

Table 7.1 Variable numbering and fit statistics of the KDE margins for the Sachs data. The column df shows the degrees of freedom

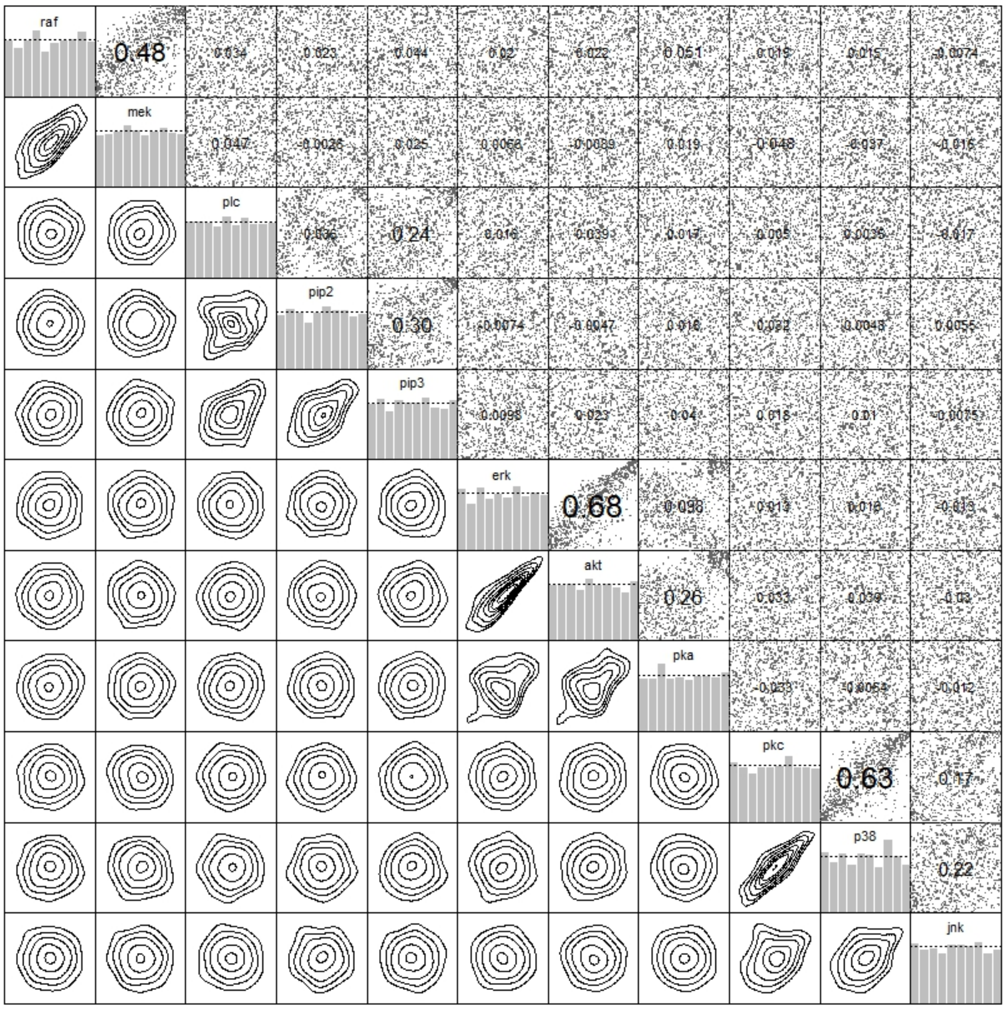


Figure 7.2 Pair plots of the Sachs data on the u-level based on KDE margins

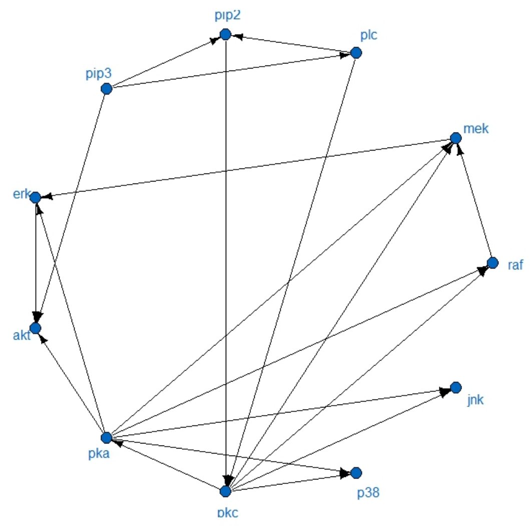


Figure 7.3 Expert DAG for the Sachs data

Figure 7.2 illustrates the pair plots of the transformed data $\hat{\mathbf{u}}$. Many of the contour plots show deviations from an elliptical shape, suggesting that a Gaussian dependence structure may not be appropriate. This was again first observed by Czado and Scharl (2021).

7.2 Model 1: Expert DAG

Consider the DAG shown in Figure 7.3. This graph relies on expert knowledge and was first introduced in Figure 3 in Sachs et al. (2005). It consists of 20 edges and will be referred to as the **Expert DAG** for the remainder of this chapter.

The objective is to fit a pair-copula Bayesian network to the Sachs data using the Expert DAG. This requires estimating a parent order for each node with multiple parents, as well as determining a bivariate copula associated with each edge in the DAG. To accomplish this, a DV-SEM is constructed sequentially by performing D-vine-based regression for each node, given its parents, as introduced in Section 3.1. In this process, each node serves as the response variable, while its parents act as covariates. The margins are fitted using kernel density estimation, as shown in Figure 7.1.

The construction of the D-vines employs the conditional log-likelihood as the selection criterion, with the family set including all parametric bivariate copulas defined in Appendix B. After fitting each D-vine, the estimated parent order for a node v is determined by the order of its parents in the first tree of the D-vine where v is the response variable. The estimated parent orders are presented in Table 7.2.

Additionally, the pair-copulas used in the decomposition of the estimated PCBN are those found in the first edge of each tree of the fitted D-vines. These pair-copulas are listed in Table 7.3. The product of the 20 pair-copula densities corresponding to the edges in Table 7.3 forms the copula density of the estimated Bayesian network.

It is worth noting that many of the estimated bivariate copulas have a low estimated Kendall's τ-value and consequently a low log-likelihood. If model selection criteria such as AIC or BIC had been used instead of the conditional log-likelihood in the D-vine regression, some variables might not have been selected. The overall copula log-likelihood of the PCBN fitted to the Expert DAG is 2042.86, which is the sum of the

Number	Variable	Parent Order
1	*raf*	$pka <_1 pkc$
2	*mek*	$raf <_2 pkc <_2 pka$
3	*plc*	*pip3*
4	*pip2*	$pip3 <_4 plc$
5	*pip3*	-
6	*erk*	$pka <_6 mek$
7	*akt*	$erk <_7 pka <_7 pip3$
8	*pka*	*pkc*
9	*pkc*	$pip2 <_9 plc$
10	*p38*	$pkc <_{10} pka$
11	*jnk*	$pkc <_{11} pka$

Table 7.2 Estimated Parent Orders for the Expert DAG of the Sachs data

log-likelihoods of each edge depicted in Table 7.3. It can be observed that the copulas associated with the five edges *raf* ⟶ *mek*, *pip3* ⟶ *pip2*, *erk* ⟶ *akt*, *pka* ⟶ *akt*, and *pkc* ⟶ *p38* contribute to

$$\frac{290.70 + 141.93 + 663.26 + 136.00 + 549.20}{2042.86} \approx 87.2\%$$

of the overall copula log-likelihood suggesting that the parental sets could potentially be reduced, leading to sparser DAGs. Czado and Scharl (2021) identified three edges that can be removed from the Expert DAG by performing D-vine-based regression with a penalized criterion. Namely, these are *mek* ⟶ *erk*, *plc* ⟶ *pkc*, *pip3* ⟶ *akt*. This is confirmed by Table 7.3 as the copulas associated with these edges show the lowest log-likelihood values.

The overall model fit will be presented in Section 7.5 and compared, node-by-node, to the models obtained by applying structure learning algorithms using the Y-test and Z-test, respectively.

7.3 Model 2: Y-DAG

In this section, the Y-vine conditional independence test is integrated into the PC algorithm to perform structure learning using log-transformed and scaled data. As before, marginal distributions are estimated using kernel density estimation. The adjusted conditional log-likelihood serves as the selection criterion for fitting Y-vines, with the family set encompassing all parametric bivariate copulas as outlined in Appendix B.

Unlike the analysis of the flight data, the significance level for testing ordinary independence, denoted as α^Y, and the tuning parameter k^Y, used for testing conditional independence with non-empty conditioning sets, are varied and set to different parameters. Additionally, no blacklists are specified for the Sachs data, and the PC algorithm estimates not only the skeletons but the complete CPDAGs. If the estimated CPDAG is not already a DAG, the remaining edge directions are manually assigned based on the Expert DAG where applicable.

When choosing different values for α^Y and k^Y, it is important to note that increasing or decreasing both values simultaneously has opposite effects on the estimated structure. Increasing k^Y, which serves as an upper bound for the copula in the last tree of the fitted Y-vine, results in higher fitted τ-values being interpreted as independence copulas, leading to less frequent rejection of the null hypothesis $H_0 : X_i \perp\!\!\!\perp X_j \mid \mathbf{X}_S$. Consequently, increasing k^Y leads to more edge removals and sparser fitted graphs. Conversely, α^Y serves as the significance level in testing the ordinary independence hypothesis $H_0 : X_i \perp\!\!\!\perp X_j$, where the null hypothesis is rejected if and only if:

$$\sqrt{\frac{9n(n-1)}{2(2n+5)}} \cdot \hat{\tau}_n > \Phi^{-1}(1 - \alpha^Y/2).$$

Edge	$i, j; S$	Family	Rotation	Parameters	df	τ	Log-Likelihood
$pka \longrightarrow raf$	1,8	Clayton	180°	0.09	1	0.04	3.16
$pkc \longrightarrow raf$	1,9 ; 8	BB8	180°	(1.09, 0.94)	2	0.03	1.73
$raf \longrightarrow mek$	2,1	Gaussian	0°	0.69	1	0.48	290.70
$pkc \longrightarrow mek$	2,9 ; 1	Gaussian	0°	-0.11	1	-0.07	5.16
$pka \longrightarrow mek$	2,8 ; 1,9	BB8	90°	(1.10, 0.90)	2	-0.03	1.37
$pip3 \longrightarrow plc$	3,5	BB8	0°	(1.70, 0.98)	2	0.25	99.93
$pip3 \longrightarrow pip2$	4,5	BB7	0°	(1.54, 0.30)	2	0.31	141.93
$plc \longrightarrow pip2$	4,3 ; 5	BB8	270°	(1.33, 1.00)	2	-0.16	52.82
$pka \longrightarrow erk$	6,8	BB7	180°	(1.07, 0.34)	2	0.17	52.84
$mek \longrightarrow erk$	6,2 ; 8	BB8	180°	(1.05, 0.66)	2	0.01	0.05
$erk \longrightarrow akt$	7,6	BB1	0°	(0.00, 3.00)	2	0.67	663.26
$pka \longrightarrow akt$	7,8 ; 6	BB8	0°	(2.62, 0.84)	2	0.33	136.00
$pip3 \longrightarrow akt$	7,5 ; 6,8	BB8	0°	(1.06, 0.83)	2	0.01	0.28
$pkc \longrightarrow pka$	8,9	BB8	90°	(1.19, 0.74)	2	-0.03	1.31
$pip2 \longrightarrow pkc$	9,4	Clayton	0°	0.07	1	0.03	2.14
$plc \longrightarrow pkc$	9,3 ; 4	BB7	90°	(1.00, 0.03)	2	-0.02	0.45
$pkc \longrightarrow p38$	10,9	BB1	0°	(0.29, 2.31)	2	0.62	549.20
$pka \longrightarrow p38$	10,8 ; 9	BB8	0°	(1.05, 1.00)	2	0.03	3.00
$pkc \longrightarrow jnk$	11,9	BB1	0°	(0.15, 1.11)	2	0.17	35.84
$pka \longrightarrow jnk$	11,8 ; 9	t	0°	(-0.01, 17.63)	2	-0.01	1.69

Table 7.3 Summary of estimated pair-copulas associated with the 20 edges of the Expert DAG fitted to the Sachs data. The numbering of the variables is according to Table 7.1. τ-values outside the range of -0.1 and 0.1 are highlighted in blue

If α^Y is increased, then both $1 - \alpha^Y/2$ and $\Phi^{-1}(1 - \alpha^Y/2)$ decrease due to the monotonicity of the quantile function. This leads to more frequent rejection of the null hypothesis, even for smaller values of $\hat{\tau}_n$, resulting in fewer edges being removed. Thus, to increase the sparsity of the resulting model, one should increase the tuning parameter k^Y while decreasing the significance level α^Y.

In the following, the values for the tuning parameter and significance level are chosen from

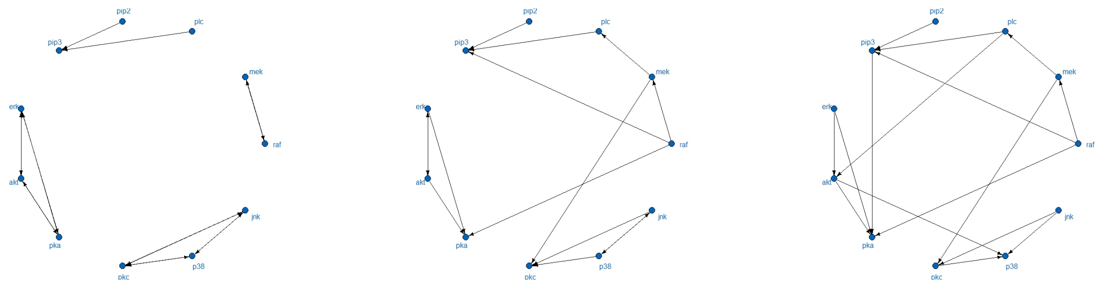
$$(k^Y, \alpha^Y) \in \{(0.05, 0.05), (0.025, 0.075), (0.01, 0.09)\}.$$

The fitted CPDAG for each configuration is referred to as a Y-CPDAG, indicating that the estimation procedure is based on Y-vines. The specific configuration is indicated by a superscript corresponding to the value of (k^Y, α^Y); for example, Y-CPDAG$^{0.025,0.075}$ represents the fitted CPDAG for the configuration $(k^Y, \alpha^Y) = (0.025, 0.075)$. The resulting Y-CPDAGs are shown in Figure 7.4, where bi-directed edges represent undirected edges. The Y-CPDAG$^{0.05,0.05}$ contains a total of 9 edges, 7 of which are undirected. In general, these undirected edges can be oriented in any direction, provided that no new v-structures or directed cycles are created. To determine the direction of the edges, the following heuristic will be applied:

(H1) Alignment with the Expert DAG: An undirected edge is oriented to match the direction of the corresponding edge in the Expert DAG, if such an edge exists, and provided that this orientation does not introduce any new v-structures or directed cycles.

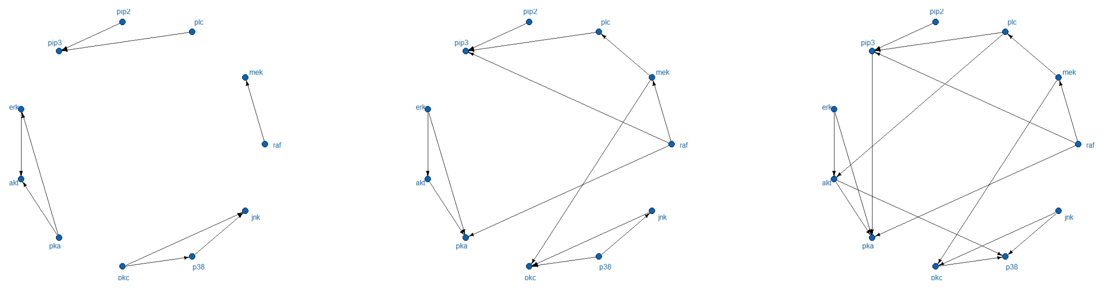
This heuristic (H1) applies to 6 of the 7 undirected edges:

- $mek - raf$: Set direction $raf \longrightarrow mek$.
- $pkc - p38$: Set direction $pkc \longrightarrow p38$.
- $pkc - jnk$: Set direction $pkc \longrightarrow jnk$.



(a) $k^Y = \alpha^Y = 0.05$ **(b)** $k^Y = 0.025, \alpha^Y = 0.075$ **(c)** $k^Y = 0.01, \alpha^Y = 0.09$

Figure 7.4 Comparison of the fitted Y-CPDAGs for the Sachs data and different values of the tuning parameter k^Y (higher order) and significance level α^Y (order zero)



(a) $k^Y = \alpha^Y = 0.05$ **(b)** $k^Y = 0.025, \alpha^Y = 0.075$ **(c)** $k^Y = 0.01, \alpha^Y = 0.09$

Figure 7.5 Comparison of the fitted Y-DAGs for the Sachs data and different values of the tuning parameter k^Y (higher order) and significance level α^Y (order zero)

- *pka – erk*: Set direction *pka* $\longrightarrow$ *erk*.
- *erk – akt*: Set direction *erk* $\longrightarrow$ *akt*.
- *pka – akt*: Set direction *pka* $\longrightarrow$ *akt*.

After applying heuristic (H1), Y-CPDAG$^{0.05,0.05}$ has only one remaining undirected edge, which is *p38 – jnk*. This edge can be oriented in either direction, and the chosen direction is *p38* $\longrightarrow$ *jnk*.

The Y-CPDAG$^{0.025,0.075}$ contains 13 edges, with 2 of them undirected. Heuristic (H1) is applied to the edge *erk – akt*, setting the direction to *erk* $\longrightarrow$ *akt*. The remaining undirected edge is again *p38 – jnk*, which is directed as *p38* $\longrightarrow$ *jnk*.

The Y-CPDAG$^{0.01,0.09}$ contains 16 edges, all of which are directed. Therefore, this CPDAG is already a DAG. The resulting Y-DAGs, after setting all edge directions, are shown in Figure 7.5. The SHDs between the Expert DAG and each of the fitted Y-DAGs are as follows:

$$\begin{aligned}
&\mathrm{SHD}(\text{Expert DAG}, \text{Y-DAG}^{0.05,0.05}) = 17,\\
&\mathrm{SHD}(\text{Expert DAG}, \text{Y-DAG}^{0.025,0.075}) = 22,\\
&\mathrm{SHD}(\text{Expert DAG}, \text{Y-DAG}^{0.01,0.09}) = 24.
\end{aligned}$$

Thus, although Y-DAG$^{0.05,0.05}$ has the fewest edges, it is the most structurally similar to the Expert DAG. Table 7.4 presents the conditional AIC and BIC values for each node across the fitted Y-DAGs, using different combinations of significance levels and tuning parameters. Among these, the Y-DAG fitted with parameters $\alpha^Y = 0.09$ and $k^Y = 0.01$ has the lowest AIC value, while the Y-DAG fitted with $\alpha^Y = 0.075$

Variable	Conditional AIC			Conditional BIC		
	$\alpha^Y = 0.05$ $k^Y = 0.05$	$\alpha^Y = 0.075$ $k^Y = 0.025$	$\alpha^Y = 0.09$ $k^Y = 0.01$	$\alpha^Y = 0.05$ $k^Y = 0.05$	$\alpha^Y = 0.075$ $k^Y = 0.025$	$\alpha^Y = 0.09$ $k^Y = 0.01$
raf	-	-	-	-	-	-
mek	-579.40	-579.40	-579.40	-574.59	-574.59	-574.59
plc	-	-1.30	-1.30	-	8.32	8.32
pip2	-	-	-	-	-	-
pip3	-452.55	-453.42	-453.42	-433.29	-424.54	-424.54
erk	-101.68	-	-	-92.05	-	-
akt	-1590.51	-1322.52	-1322.59	-1571.25	-1312.89	-1303.33
pka	-	-413.96	-410.72	-	-389.89	-377.01
pkc	-	-1131.00	-67.97	-	-1102.11	-48.71
p38	-1094.40	-	-1203.98	-1084.77	-	-1179.91
jnk	-170.11	-132.04	-	-150.85	-122.41	-
SUM	-3988.65	-4033.65	-4039.38	-3906.80	-3918.10	3899.76

Table 7.4 Comparison of the conditional AIC and BIC for each variable of the Sachs data across the fitted Y-models under different choices of α^Y and k^Y

and $k^Y = 0.025$ achieves the lowest BIC value. There are notable differences in the fit statistics for certain nodes due to variations in edge direction across the DAGs. For instance, the edge $pkc \longrightarrow p38$ is present in both Y-DAG$^{0.05,0.05}$ and Y-DAG$^{0.01,0.09}$, whereas in Y-DAG$^{0.025,0.075}$, the direction is reversed to $pkc \longleftarrow p38$. As a result, the first two DAGs have lower AIC and BIC values for the node *p38*, while Y-DAG$^{0.025,0.075}$ shows lower AIC and BIC values for the node *pkc*.

The fitted pair-copulas for each edge of the Y-DAG$^{0.025,0.075}$ are depicted in Table 7.5. Compared to the Expert DAG, less copulas with low absolute τ-values were fitted in this case. However, 5 of the 13 edges still exhibit low log-likelihood values indicating that a removal of these edges may be feasible.

7.4 Model 3: Z-DAG

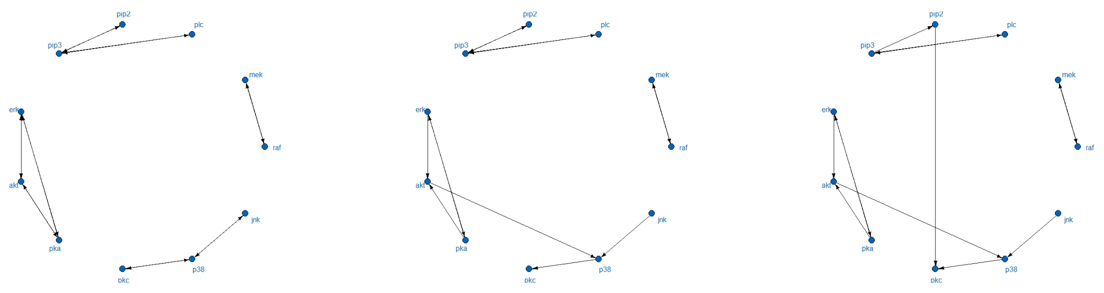
The PC algorithm is applied again to learn the structure of the Bayesian network using the log-transformed and scaled Sachs data, but this time, Fisher's Z-test for partial correlation is used to assess conditional independence. The significance level for the Z-test, α^Z, is varied across the values $\{0.05, 0.09, 0.13\}$. Similar to the significance level α^Y used for ordinary independence testing in the Y-test, increasing α^Z results in the null hypothesis of (conditional) independence being rejected more frequently, leading to less sparse graphical structures.

As before, the fitted CPDAG for each configuration is referred to as a Z-CPDAG, reflecting that the estimation procedure is based on Fisher's Z-test. The specific configuration is denoted by a superscript indicating the value of α^Z; for instance, Z-CPDAG$^{0.05}$ corresponds to the fitted CPDAG with $\alpha^Z = 0.05$. The resulting Z-CPDAGs are depicted in Figure 7.6, where bi-directed edges represent undirected edges. To determine the edge directions, the heuristic (H1) will be applied again. The Z-DAG$^{0.05}$ consists of 8 edges in total, all of which are initially undirected. The heuristic (H1) is applicable to 7 of these edges:

- *mek* – *raf*: Set direction $raf \longrightarrow mek$.
- *pkc* – *p38*: Set direction $pkc \longrightarrow p38$.
- *pka* – *erk*: Set direction $pka \longrightarrow erk$.
- *erk* – *akt*: Set direction $erk \longrightarrow akt$.
- *pka* – *akt*: Set direction $pka \longrightarrow akt$.

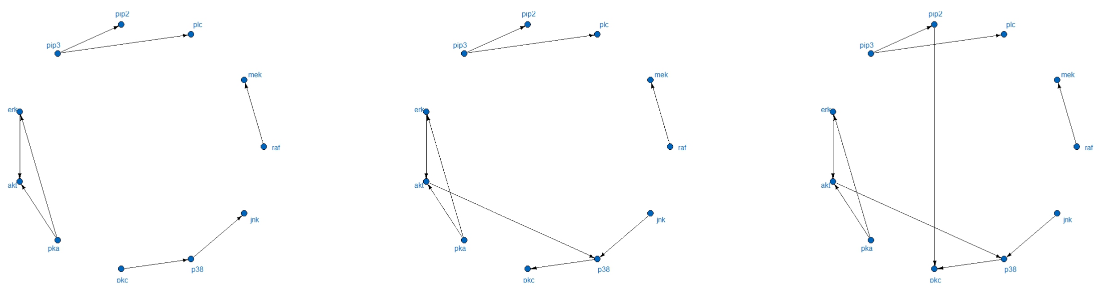
Edge	$i, j; S$	Family	Rotation	Parameters	df	τ	Log-Likelihood
$raf \longrightarrow mek$	2,1	Gaussian	0°	0.69	1	0.48	290.70
$mek \longrightarrow plc$	3,2	BB8	180°	(1.14, 0.91)	2	0.04	2.65
$pip2 \longrightarrow pip3$	5,4	BB7	0°	(1.54, 0.30)	2	0.31	141.93
$plc \longrightarrow pip3$	5,3 ; 4	BB8	0°	(1.87, 0.93)	2	0.26	88.35
$raf \longrightarrow pip3$	5,1 ; 4	t	0°	(0.04, 15.68)	2	0.03	2.44
$erk \longrightarrow akt$	7,6	BB1	0°	(0.00, 3.00)	2	0.67	663.26
$akt \longrightarrow pka$	8,7	BB7	180°	(1.13, 0.66)	2	0.28	129.61
$erk \longrightarrow pka$	8,6 ; 7	BB8	270°	(2.28, 0.81)	2	-0.25	78.48
$raf \longrightarrow pka$	8,1 ; 7,6	Clayton	180°	0.09	1	0.05	3.89
$p38 \longrightarrow pkc$	9,10	BB1	0°	(0.29, 2.31)	2	0.62	549.20
$jnk \longrightarrow pkc$	9,11 ; 10	BB8	90°	(1.14, 1.00)	2	-0.08	21.07
$mek \longrightarrow pkc$	9,2 ; 10,11	BB8	270°	(1.13, 0.83)	2	-0.03	1.23
$p38 \longrightarrow jnk$	11,10	BB1	0°	(0.05, 1.26)	2	0.22	68.02

Table 7.5 Summary of estimated pair-copulas associated with the 13 edges of the Y-DAG fitted to the Sachs data with $\alpha^Y = 0.075$ and $k^Y = 0.025$. The numbering of the variables is according to Table 7.1. τ-values outside the range of -0.1 and 0.1 are highlighted in blue



(a) $\alpha^Z = 0.05$ **(b)** $\alpha^Z = 0.09$ **(c)** $\alpha^Z = 0.13$

Figure 7.6 Comparison of the fitted Z-CPDAGs for the Sachs data and different values of the significance level α^Z



(a) $\alpha^Z = 0.05$ **(b)** $\alpha^Z = 0.09$ **(c)** $\alpha^Z = 0.13$

Figure 7.7 Comparison of the fitted Z-DAGs for the Sachs data and different values of the significance level α^Z

- *pip2 – pip3*: Set direction *pip3* $\longrightarrow$ *pip2*.
- *pip3 – plc*: Set direction *pip3* $\longrightarrow$ *plc*.

The remaining edge, *jnk–p38*, is oriented as *p38* $\longrightarrow$ *jnk* to prevent the formation of the v-structure *pkc* $\longrightarrow$ *p38* $\longleftarrow$ *jnk*.

The Z-DAG$^{0.09}$ consists of 9 edges in total, with 4 initially undirected. All of these edges are directed by applying heuristic (H1):

- *mek – raf*: Set direction *raf* $\longrightarrow$ *mek*.
- *pka – erk*: Set direction *pka* $\longrightarrow$ *erk*.
- *pip2 – pip3*: Set direction *pip3* $\longrightarrow$ *pip2*.
- *pip3 – plc*: Set direction *pip3* $\longrightarrow$ *plc*.

The Z-DAG$^{0.13}$ contains 10 edges in total, with 3 of them being undirected. All of these edges are directed after applying the heuristic:

- *mek – raf*: Set direction *raf* $\longrightarrow$ *mek*.
- *pka – erk*: Set direction *pka* $\longrightarrow$ *erk*.
- *pip3 – plc*: Set direction *pip3* $\longrightarrow$ *plc*.

The resulting Z-DAGs, after setting all edge directions, are displayed in Figure 7.7. The SHDs between the Expert DAG and each of the fitted Y-DAGs are as follows:

$$\begin{aligned}
\text{SHD}(\text{Expert DAG, Z-DAG}^{0.05}) &= 16,\\
\text{SHD}(\text{Expert DAG, Z-DAG}^{0.025}) &= 18,\\
\text{SHD}(\text{Expert DAG, Z-DAG}^{0.01}) &= 18.
\end{aligned}$$

The Z-DAGs are structurally closer to the Expert DAG than the fitted Y-DAGs. However, even with a significantly increased significance level α^Z, the Z-DAGs have fewer edges.

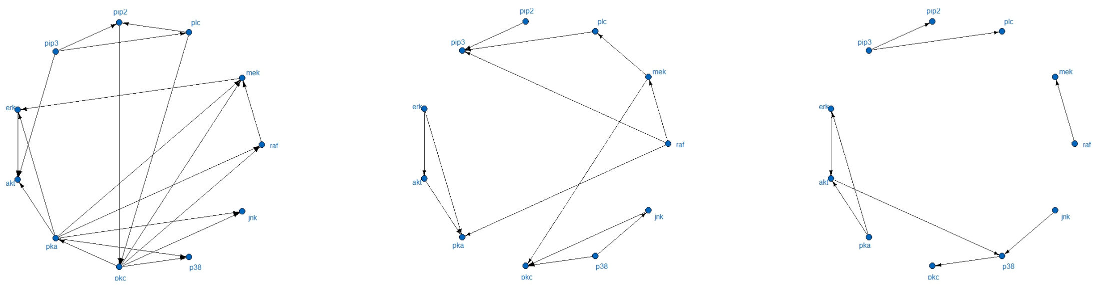
Table 7.6 presents the conditional AIC and BIC values for each node across the fitted Z-DAGs. Since the fitted models are structurally very similar, the differences between them are minimal. For nodes where differences are notable, they arise from varying edge directions across the models. For instance, the Z-DAG$^{0.05}$ features the structure *pkc* $\longrightarrow$ *p38* $\longrightarrow$ *jnk*, while the other two DAGs display the structure *pkc* $\longleftarrow$ *p38* $\longleftarrow$ *jnk*. Among the models fitted using Fisher's Z-test, Z-DAG$^{0.13}$ exhibits the lowest AIC value, whereas Z-DAG$^{0.09}$ has the lowest BIC value. The fitted pair-copulas for each edge of the Z-DAG$^{0.09}$ are shown in Table 7.7. Compared to the Expert DAG, only one copula, associated with the edge *akt* $\longrightarrow$ *p38*, has a low absolute τ-value in this case.

Variable	**Conditional AIC**			**Conditional BIC**		
	$\alpha^Z = 0.05$	$\alpha^Z = 0.09$	$\alpha^Z = 0.13$	$\alpha^Z = 0.05$	$\alpha^Z = 0.09$	$\alpha^Z = 0.13$
raf	-	-	-	-	-	-
mek	-579.40	-579.40	-579.40	-574.59	-574.59	-574.59
plc	-195.85	-195.85	-195.85	-186.22	-186.22	-186.22
pip2	-279.85	-279.85	-279.85	-270.22	-270.22	-270.22
pip3	-	-	-	-	-	-
erk	-101.68	-101.68	-101.68	-92.05	-92.05	-92.05
akt	-1590.51	-1590.51	-1590.51	-1571.25	-1571.25	-1571.25
pka	-	-	-	-	-	-
pkc	-	-1094.40	-1097.85	-	-1084.77	-1078.59
p38	-1094.40	-139.52	-139.52	-1084.77	-125.07	-125.07
jnk	-132.04	-	-	-122.42	-	-
SUM	-3973.74	-3981.22	-3984.66	-3901.53	-3904.18	-3898.00

Table 7.6 Comparison of the conditional AIC and BIC for each variable of the Sachs data across the fitted Z-models under different choices of α^Z

Edge	***i, j*; S**	**Family**	**Rotation**	**Parameters**	**df**	τ	**Log-Likelihood**
raf ⟶ *mek*	2,1	Gaussian	0°	0.69	1	0.48	290.70
pip3 ⟶ *plc*	3,5	BB8	0°	(1.70, 0.98)	2	0.25	99.93
pip3 ⟶ *pip2*	4,5	BB7	0°	(1.54, 0.30)	2	0.31	141.93
pka ⟶ *erk*	6,8	BB7	180°	(1.07, 0.34)	2	0.17	52.84
erk ⟶ *akt*	7,6	BB1	0°	(0.00, 3.00)	2	0.67	663.26
pka ⟶ *akt*	7,8 ; 6	BB8	0°	(2.62, 0.84)	2	0.33	136.00
jnk ⟶ *p38*	10,11	BB1	0°	(0.05, 1.26)	2	0.22	68.02
akt ⟶ *p38*	10,7 ; 11	Clayton	0°	0.11	1	0.05	4.74
p38 ⟶ *pkc*	9,10	BB1	0°	(0.29, 2.31)	2	0.62	549.20

Table 7.7 Summary of estimated pair-copulas associated with the 9 edges of the Z-DAG fitted to the Sachs data with $\alpha^Z = 0.09$. The numbering of the variables is according to Table 7.1. τ-values outside the range of -0.1 and 0.1 are highlighted in blue



(a) Expert DAG **(b)** Y-DAG with $\alpha^Y = 0.075$, $k^Y = 0.025$ **(c)** Z-DAG with $\alpha^Z = 0.09$

Figure 7.8 Comparison of the Expert DAG, the fitted Y-DAG with $\alpha^Y = 0.075$ and $k^Y = 0.025$ and the fitted Z-DAG with $\alpha^Z = 0.09$ for the Sachs data

Number	Variable	Expert	Y	Z
1	*raf*	$pka <_1 pkc$	-	-
2	*mek*	$raf <_2 pkc <_2 pka$	*raf*	*raf*
3	*plc*	*pip3*	*mek*	*pip3*
4	*pip2*	$pip3 <_4 plc$	-	*pip3*
5	*pip3*	-	$pip2 <_5 plc <_5 raf$	-
6	*erk*	$pka <_6 mek$	-	*pka*
7	*akt*	$erk <_7 pka <_7 pip3$	*erk*	$erk <_7 pka$
8	*pka*	*pkc*	$akt <_8 erk <_8 raf$	-
9	*pkc*	$pip2 <_9 plc$	$p38 <_9 jnk <_9 mek$	*p38*
10	*p38*	$pkc <_{10} pka$	-	$jnk <_{10} akt$
11	*jnk*	$pkc <_{11} pka$	*p38*	-

Table 7.8 Comparison of parent orders for the Expert DAG, the fitted Y-DAG with $\alpha^Y = 0.075$ and $k^Y = 0.025$ and the fitted Z-DAG with $\alpha^Z = 0.09$ for the Sachs data

7.5 Model Comparison

In this section, the PCBN fitted to the Expert DAG is compared node-by-node to the PCBNs fitted to estimated structures derived using the PC algorithm. The comparisons are based on the Y-vine conditional independence test with parameters $\alpha^Y = 0.075$ and $k^Y = 0.025$, and Fisher's Z-test for partial correlation with $\alpha^Z = 0.09$. These specific models were selected for the overall comparison as they exhibit the lowest BIC among the models fitted with varying significance levels and tuning parameters.

Figure 7.8 presents the three graphical structures side-by-side. Notably, the Expert DAG contains more edges than the estimated DAGs, with the Z-DAG$^{0.09}$ having the fewest (9 edges).

Table 7.8 compares the estimated parent sets and their order for each node across the different DAGs. It is clear that not only the parent sets but also the order of the parents differs significantly between the models. For instance, in the Expert DAG, *pka* is a child of *pkc*, while in the Y-DAG$^{0.025,0.075}$, *pka* has three parents, none of which is *pkc*. In contrast, the Z-DAG$^{0.09}$ shows *pka* as having no parents at all. These differences arise from the varying edge directions across the models. Both the Expert DAG and the Z-DAG contain the triangular structure $pka \longrightarrow erk \longrightarrow akt \longleftarrow pka$, while the Y-DAG represents a different structure: $pka \longleftarrow erk \longrightarrow akt \longrightarrow pka$. Without expert knowledge, it is challenging to assess whether the substructure proposed by the Y-DAG is a plausible alternative. This highlights the importance of incorporating expert-based constraints (e.g., blacklists) during structure estimation, as demonstrated in Chapter 6 with the flight data.

The differences in parent sets across the three models are further reflected in Tables 7.9 and 7.10, which show the conditional log-likelihoods, degrees of freedom, and the conditional AIC and BIC values for each

Variable	Conditional Log-Likelihood			df		
	Expert	Y	Z	Expert	Y	Z
raf	4.89	-	-	3	-	-
mek	297.23	290.70	290.70	4	1	1
plc	99.93	2.65	99.93	2	2	2
pip2	194.75	-	141.93	4	-	2
pip3	-	232.71	-	-	6	-
erk	52.89	-	52.84	4	-	2
akt	799.53	663.26	799.25	6	2	4
pka	1.31	211.98	-	2	5	-
pkc	2.59	571.50	549.20	3	6	2
p38	552.20	-	72.76	4	-	3
jnk	37.53	68.02	-	4	2	-
SUM	2042.83	2040.82	2006.61	36	24	16

Table 7.9 Comparison of the conditional copula log-likelihood and degrees of freedom for each variable across the Expert DAG, the fitted Y-DAG with $\alpha^Y = 0.075$ and $k^Y = 0.025$ and the fitted Z-DAG with $\alpha^Z = 0.09$ for the Sachs data

Variable	Conditional AIC			Conditional BIC		
	Expert	Y	Z	Expert	Y	Z
raf	-3.77	-	-	10.67	-	-
mek	-586.45	-579.40	-579.40	-567.20	-574.59	-574.59
plc	-195.85	-1.30	-195.85	-186.22	8.32	-186.22
pip2	-381.49	-	-279.85	-362.24	-	-270.22
pip3	-	-453.42	-	-	-424.54	-
erk	-97.78	-	-101.68	-78.52	-	-92.05
akt	-1587.07	-1322.59	-1590.51	-1558.18	-1312.89	-1571.25
pka	1.39	-413.96	-	11.02	-389.89	-
pkc	0.82	-1131.00	-1094.40	15.27	-1102.11	-1084.77
p38	-1096.39	-	-139.52	-1077.13	-	-125.07
jnk	-67.05	-132.04	-	-47.79	-122.42	-
SUM	-4013.66	-4033.65	-3981.22	-3840.33	-3918.10	-3904.18

Table 7.10 Comparison of the conditional AIC and BIC for each variable of the Sachs data across the Expert DAG, the fitted Y-DAG with $\alpha^Y = 0.075$ and $k^Y = 0.025$ and the fitted Z-DAG with $\alpha^Z = 0.09$

node at the copula level. In the Y-DAG$^{0.025,0.075}$, *pka* has multiple parents, resulting in the highest log-likelihood and the lowest AIC and BIC for that node. However, because these results are driven by the reversed edges *erk* $\longrightarrow$ *pka* and *akt* $\longrightarrow$ *pka*, the log-likelihoods for nodes *erk* and *akt* are significantly lower compared to the Expert DAG and the Z-DAG. A similar pattern emerges in other triangular structures, such as those involving *pip2*, *pip3*, and *plc*, or the trio of *pkc*, *p38*, and *jnk*, where edge directions vary between models.

Overall, the PCBN fitted to the Expert DAG yields the highest log-likelihood, closely followed by the PCBN fitted to the Y-DAG$^{0.025,0.075}$. After penalizing for the number of parameters, the best-fitting model in terms of both AIC and BIC is the PCBN fitted to the Y-DAG$^{0.025,0.075}$. Interestingly, both data-driven models based on the PC algorithm exhibit lower overall BIC values than the Expert Model, suggesting that the data supports a sparser graphical structure than the one proposed by experts.

8 Conclusion and Outlook

This thesis has shown that Y-vines can be effectively utilized for testing conditional independence statements of the form $X_i \perp\!\!\!\perp X_j \mid \mathbf{X}_S$ by treating X_i and X_j as response variables and $\mathbf{X}_S$ as covariates. In this framework, the response variables are treated symmetrically, and there is no need for integration to obtain the conditional copula density of the responses given the covariates. Furthermore, the flexibility inherent in the Y-vine estimation process reduces reliance on asymptotic results. As a result, a simple upper bound k^Y on the estimated Kendall's τ of the copula in the final Y-vine tree, $C_{ij;S}$, proves to be highly efficient.

Incorporating Y-vine-based conditional independence testing into the PC algorithm led to promising results, as demonstrated in simulation studies across both Gaussian and non-Gaussian settings with dimensions $d = 4, 6,$ and 11. These studies showed that the Y-vine-based PC algorithm more accurately recovers the true underlying structure of Bayesian networks compared to the standard PC algorithm, which uses Fisher's Z-test for partial correlations. This is evidenced by consistently lower SHD values across all setups and dimensions. However, in the case of Gaussian Bayesian networks, the Y-vine-based method does not significantly outperform its benchmark. This result is expected because Fisher's Z-test assumes a multivariate Gaussian distribution, where partial and conditional correlations coincide, and vanishing partial correlations indicate true independence.

A major limitation of the Y-vine-based PC algorithm is the significantly increased computational cost, especially in higher dimensions. The computational complexity of the PC algorithm is driven by the number of conditional independence queries, which can grow exponentially with dimensionality in the worst case. Using Y-vine-based regression for each query is computationally intensive, particularly as the conditioning sets grow larger, given that the regression method incorporates a one-step-ahead forward-selection algorithm for ordering covariates. Thus, reducing computational complexity is a key direction for future work. For example, the current PC algorithm implementation in the R package `pcalg` introduces inefficiencies by sometimes querying both $X_i \perp\!\!\!\perp X_j \mid \mathbf{X}_S$ and $X_j \perp\!\!\!\perp X_i \mid \mathbf{X}_S$ within the same iteration. As these statements are equivalent, reusing the result from the first test would eliminate the need to recompute the second.

Other promising avenues for future research include comparing the Y-vine-based PC algorithm with alternative structure learning methods used for continuous Bayesian networks, such as score-based or hybrid approaches. Additionally, exploring the use of a two-step-ahead forward-selection algorithm in Y-vine-based regression could potentially improve structure learning, although this would further increase computational costs, which are already a bottleneck.

Another key finding of this thesis is that D-vine structural equation models (DV-SEMs) can be employed to approximate pair-copula Bayesian networks. In this approach, a D-vine is constructed for each node with multiple parents, where the node of interest is set as a leaf node in the first tree, followed by its parents in their given order. While some copulas in the D-vine that involve only the parents of a node may require integration, in the DV-SEM approximation, these copulas are estimated directly from the data. This approach facilitates both sampling and likelihood inference for pair-copula Bayesian networks, as it avoids integration and imposes no restrictions on the structure or the parent ordering of a node.

In this thesis, the complete D-vine substructure involving only the parents of a node was estimated. While this approach is computationally efficient, it does not exploit the d-separation properties of the underlying Bayesian network. Specifically, if d-separation shows that two nodes i and j are d-separated given a set of nodes S, then the copula $C_{ij;S}$ is known to be the independence copula. When such a copula appears in a parental D-vine, it can be directly set as the independence copula, eliminating the need for estimation. The challenge, however, lies in identifying the position of these independence copulas within the D-vine structure, as there is currently no algorithm for partial D-vine estimation that allows only certain specific copulas to be estimated. Developing such an algorithm would likely enhance the accuracy of the pair-copula Bayesian network approximation through a DV-SEM.

Finally, this thesis has demonstrated that the DV-SEM approximation is also useful for learning the parameters of a PCBN. This was first noted by Czado and Scharl (2021). Specifically, combining Y-vine-based structure learning with D-vine-based parameter learning offers great flexibility in estimating the entire network, including its conditional independencies, parent orders, and the underlying probability distribution. In this context, penalized selection criteria could be applied in D-vine-based regression to further reduce the structure estimated by the PC algorithm. Additionally, while this thesis focused solely on parametric pair-copulas, incorporating non-parametric bivariate copulas could increase the model's flexibility. Moreover, a two-step-ahead forward-selection D-vine regression approach, as introduced by Tepegjozova et al. (2022), might improve model fit, albeit at the cost of increased computational effort.

In practical applications, it is advisable to combine Y-vine-based structure learning and D-vine-based parameter learning for continuous Bayesian networks with expert knowledge. Specifically, expert knowledge could be used to create blacklists that specify infeasible edges before applying the learning algorithms. Currently, no R package exists that allows for both user-specified conditional independence tests and the creation of edge-direction blacklists. Future work should aim to integrate these functionalities into a single package to reduce manual effort.

A Continuous Parametric Distributions

The following appendix contains definitions for univariate and multivariate distributions used in this thesis. Each introduction to a distribution includes definitions of its parameters, probability density function, moments, and moment generating function. The content is taken from the appendix in Czado 2022.

A.1 Univariate Distributions

Definition A.1.1 (Uniform distribution)**.** A random variable X is said to follow a **uniform distribution** over $[a, b]$, denoted by $X \sim U(a, b)$, if its pdf is given by

$$f(x) = \frac{1}{b-a}, \quad x \in [a, b],$$

where $a, b \in \mathbb{R}$, $a < b$, are boundary parameters. The moments of X are given by

$$\mathbb{E}[X] = \frac{a+b}{2}, \quad \text{and} \quad \text{Var}[X] = \frac{(b-a)^2}{12}.$$

Further, the moment generating function of X is given by

$$m_X(t) = \frac{e^{tb} - e^{ta}}{t(b-a)}, \quad t \in \mathbb{R} \setminus \{0\}.$$

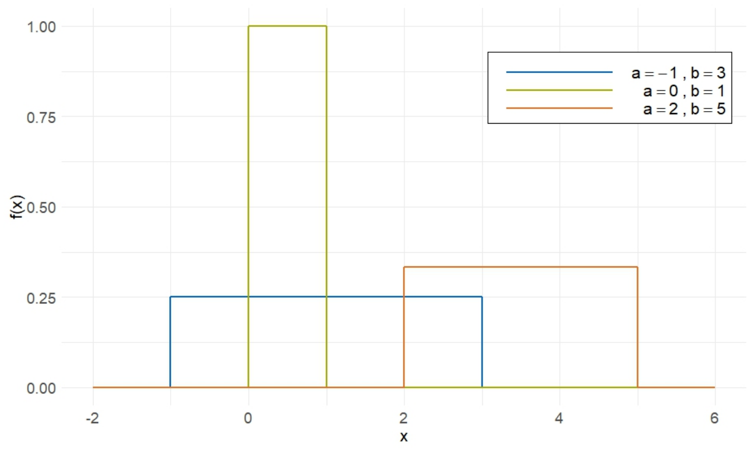


Figure A.1 Density of the uniform distribution for different values of a and b

Definition A.1.2 (Normal distribution)**.** A random variable X is said to follow a **normal distribution** with mean $\mu \in \mathbb{R}$ and variance $\sigma^2 > 0$, denoted by $X \sim N(\mu, \sigma^2)$, if its pdf is given by

$$f(x) = \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(-\frac{1}{2\sigma^2}(x-\mu)^2\right), \quad x \in \mathbb{R}.$$

The moments of X are given by

$$\mathbb{E}[X] = \mu, \quad \text{and} \quad \text{Var}[X] = \sigma^2.$$

Further, the moment generating function of X is given by

$$m_X(t) = \exp(\mu t + \frac{1}{2}\sigma^2 t^2), \quad t \in \mathbb{R}.$$

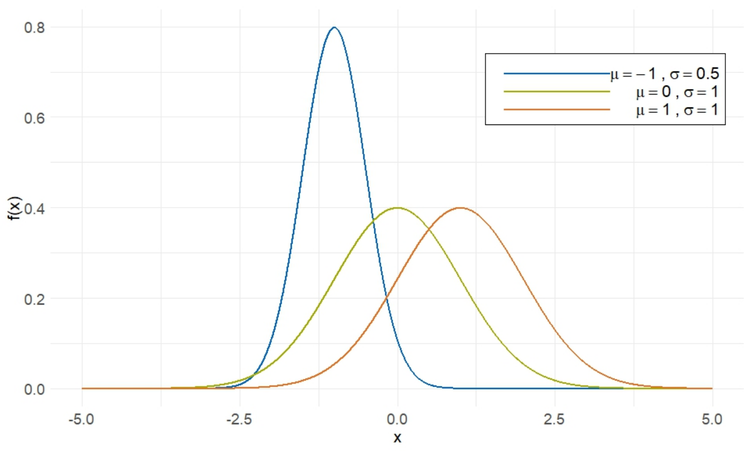


Figure A.2 Density of the normal distribution for different values of μ and σ

Definition A.1.3 (t-distribution)**.** A random variable X is said to follow a **t-distribution** with $\nu > 0$ degrees of freedom, denoted by $X \sim t(\nu)$, if its pdf is given by

$$f(x) = \frac{\Gamma\left(\frac{\nu+1}{2}\right)}{\sqrt{\nu\pi}\cdot\Gamma\left(\frac{\nu}{2}\right)}\left(1+\frac{x^2}{\nu}\right)^{-\frac{\nu+1}{2}}, \quad x \in \mathbb{R},$$

where $\Gamma(\cdot)$ denotes the gamma function. The moments of X are given by

$$\mathbb{E}[X] = 0 \quad \text{for} \quad \nu > 1, \quad \text{and} \quad \mathrm{Var}[X] = \begin{cases} \frac{\nu}{\nu-2}, & \text{for } \nu > 2, \\ \infty, & \text{for } 1 < \nu \leq 2. \end{cases}.$$

The moment generating function of X is not defined.

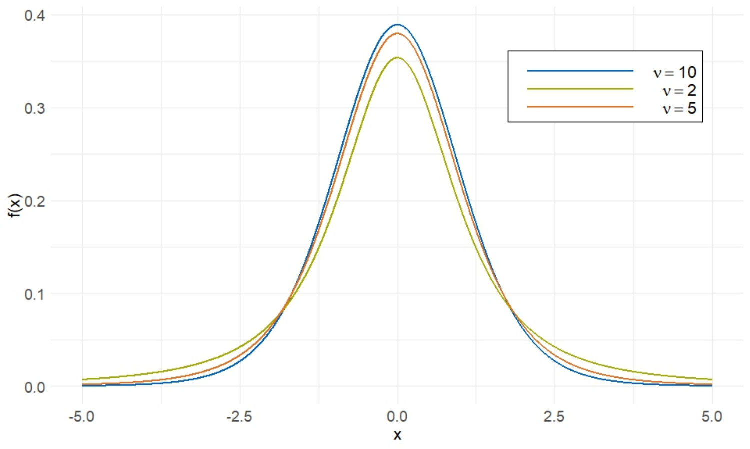


Figure A.3 Density of the t-distribution for different values of ν

Definition A.1.4 (Beta distribution)**.** A random variable X is said to follow a **beta distribution** with shape parameters $\alpha > 0$ and $\beta > 0$, denoted by $X \sim \mathrm{Beta}(\alpha, \beta)$, if its pdf is given by

$$f(x) = \frac{\Gamma(\alpha+\beta)}{\Gamma(\alpha)\Gamma(\beta)}x^{\alpha-1}(1-x)^{\beta-1}, \quad x \in [0,1],$$

where $\Gamma(\cdot)$ denotes the gamma function. The moments of X are given by

$$\mathbb{E}[X] = \frac{\alpha}{\alpha+\beta}, \quad \text{and} \quad \mathrm{Var}[X] = \frac{\alpha\beta}{(\alpha+\beta)^2(\alpha+\beta+1)}.$$

The moment generating function of X is not generally expressible in a simple form.

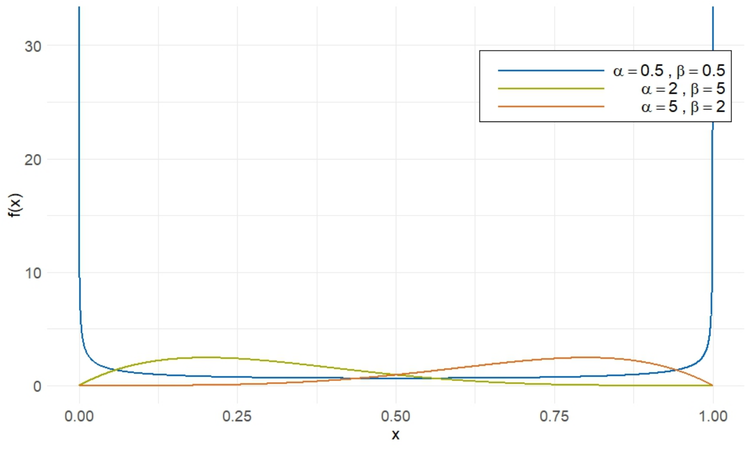


Figure A.4 Density of the beta distribution for different values of α and β

Definition A.1.5 (Gamma distribution). A random variable X is said to follow a **gamma distribution** with shape parameter $k > 0$ and scale parameter $\theta > 0$, denoted by $X \sim \text{Gamma}(k, \theta)$, if its pdf is given by

$$f(x) = \frac{1}{\Gamma(k)\theta^k} x^{k-1} e^{-\frac{x}{\theta}}, \quad x \geq 0,$$

where $\Gamma(\cdot)$ denotes the gamma function. The moments of X are given by

$$\mathbb{E}[X] = k\theta \quad \text{and} \quad \text{Var}[X] = k\theta^2.$$

The moment generating function of X is given by

$$m_X(t) = \left(\frac{1}{1-\theta t}\right)^k, \quad t < \frac{1}{\theta}.$$

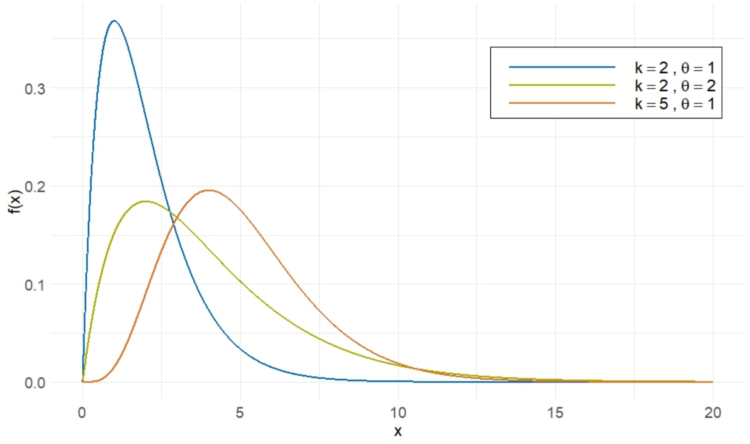


Figure A.5 Density of the gamma distribution for different values of k and θ

Definition A.1.6 (Lognormal distribution). A random variable X is said to follow a **lognormal distribution** with parameters $\mu \in \mathbb{R}$ and $\sigma > 0$, denoted by $X \sim LN(\mu, \sigma^2)$, if its pdf is given by

$$f(x) = \frac{1}{x\sigma\sqrt{2\pi}} \exp\left(-\frac{(\ln(x)-\mu)^2}{2\sigma^2}\right), \quad x > 0.$$

The moments of X are given by

$$\mathbb{E}[X] = e^{\mu+\frac{\sigma^2}{2}} \quad \text{and} \quad Var[X] = \left(e^{\sigma^2}-1\right) e^{2\mu+\sigma^2}.$$

The moment generating function of X is not defined.

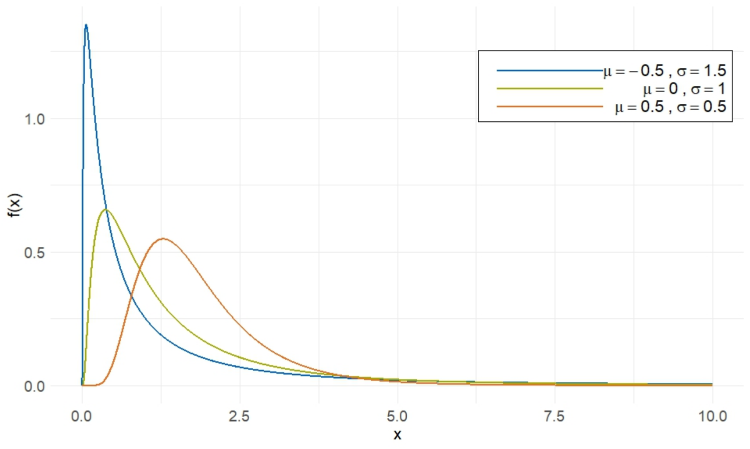


Figure A.6 Density of the lognormal distribution for different values of μ and σ

A.2 Multivariate Distributions

Definition A.2.1 (Multivariate normal distribution)**.** A random vector $\mathbf{X} = (X_1, \dots, X_d)^\top$ is said to follow a **multivariate normal distribution** with mean vector $\boldsymbol{\mu} \in \mathbb{R}^d$ and positive-definite covariance matrix $\Sigma \in \mathbb{R}^{d \times d}$, denoted by $\mathbf{X} \sim N(\boldsymbol{\mu}, \Sigma)$, if its pdf is given by

$$f(\mathbf{x}) = \frac{1}{(2\pi)^{d/2}|\Sigma|^{1/2}} \exp\left(-\frac{1}{2}(\mathbf{x}-\boldsymbol{\mu})^\top \Sigma^{-1}(\mathbf{x}-\boldsymbol{\mu})\right), \quad \mathbf{x} = (x_1, \dots, x_d)^\top \in \mathbb{R}^d,$$

where $|\Sigma|$ denotes the determinant of Σ, and Σ^{-1} denotes the inverse of Σ. The mean vector and covariance matrix are given by

$$\mathbb{E}[\mathbf{X}] = \boldsymbol{\mu} \quad \text{and} \quad \mathrm{Cov}[\mathbf{X}] = \Sigma.$$

The moment generating function of $\mathbf{X}$ is given by

$$m_{\mathbf{X}}(\mathbf{t}) = \exp\left(\boldsymbol{\mu}^\top \mathbf{t} + \frac{1}{2}\mathbf{t}^\top \Sigma \mathbf{t}\right), \quad \mathbf{t} \in \mathbb{R}^d.$$

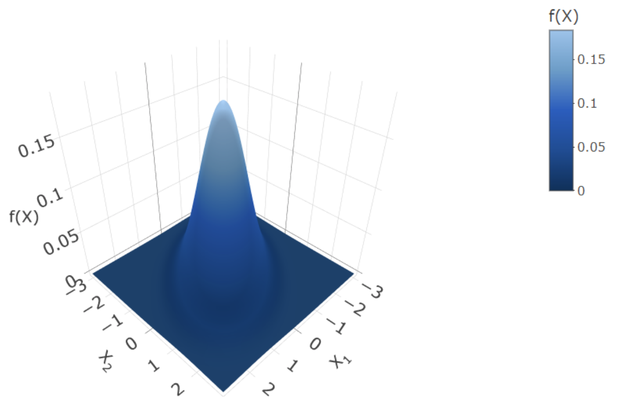


Figure A.7 Density of the bivariate normal distribution for $\boldsymbol{\mu} = \begin{pmatrix} 0 \\ 0 \end{pmatrix}$ and $\Sigma = \begin{pmatrix} 1 & 0.5 \\ 0.5 & 1 \end{pmatrix}$

B Bivariate Copulas

In the following section, all parametric bivariate copulas used in this thesis will be defined. For each copula, the parameter ranges, and different rotations will be provided, along with normalized contour plots. For further details, readers are referred to Joe (2014).

Definition B.0.1 (Independence copula)**.** The bivariate **independence copula** is given by

$$C(u_1, u_2) = u_1 \cdot u_2, \quad u_1, u_2 \in [0, 1]^2.$$

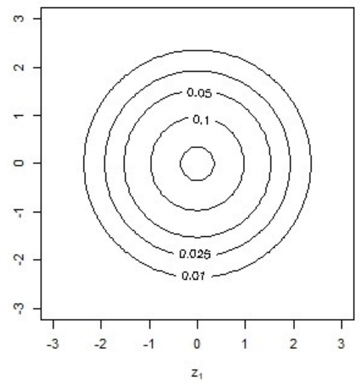


Figure B.1 Normalized contour plot of the independence copula density

B.1 Elliptical Copulas

Definition B.1.1 (Gaussian copula)**.** The bivariate **Gaussian copula** is given by

$$C(u_1, u_2; \rho) = \Phi_2(\Phi^{-1}(u_1), \Phi^{-1}(u_2)), \quad u_1, u_2 \in (0, 1),$$

where $\rho \in [-1, 1]$ is the correlation parameter, Φ_2 is the cdf of the bivariate standard normal distribution, and Φ^{-1} the quantile function of the univariate standard normal distribution.

Definition B.1.2 (Student's t copula)**.** The density of the bivariate **Student's t copula** is given by

$$c(u_1, u_2; \nu, \rho) = \frac{t(T_\nu^{-1}(u_1), T_\nu^{-1}(u_2); \nu, \rho)}{t_\nu(T_\nu^{-1}(u_1)) t_\nu(T_\nu^{-1}(u_2))}, \quad u_1, u_2 \in (0, 1),$$

where t_ν is the pdf of the univariate t-distribution with $\nu > 0$ degrees of freedom, T_ν^{-1} is the quantile function of the univariate t-distribution, and t is the pdf of the bivariate t-distribution given by

$$t(x_1, x_2; \nu, \rho) = \frac{\Gamma(\frac{\nu+2}{2})(1-\rho^2)^{-1/2}}{\Gamma(\frac{\nu}{2})\nu\pi} \left(1 + \frac{1}{\nu} \frac{x_1^2 - 2x_1x_2\rho + x_2^2}{1-\rho^2}\right)^{-\frac{\nu+2}{2}}$$

with scale parameter $\rho \in (-1, 1)$.

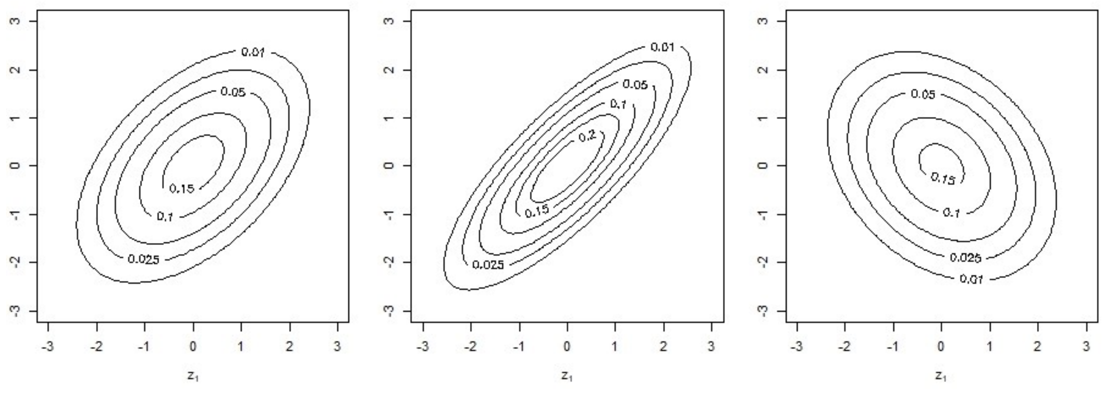


Figure B.2 Normalized contour plots of the Gaussian copula density for $\rho = 0.5, 0.8, -0.3$ yielding $\tau \approx 0.33, 0.59, -0.19$ (left to right)

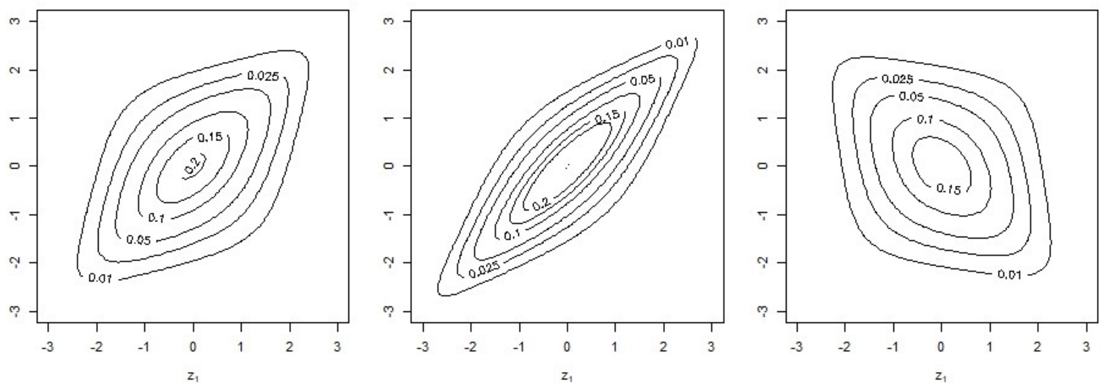


Figure B.3 Normalized contour plots of the Student's t copula density for $(\nu, \rho) = (4, 0.5), (4, 0.8), (4, -0.3)$ yielding $\tau \approx 0.33, 0.59, -0.19$ (left to right)

B.2 Archimedean Copulas

Definition B.2.1 (Clayton copula)**.** The bivariate **Clayton copula** is given by

$$C(u_1, u_2) = (u_1^{-\delta} + u_2^{-\delta} - 1)^{-\frac{1}{\delta}}, \quad u_1, u_2 \in [0, 1],$$

where $\delta \in (0, \infty)$ is a parameter controlling the degree of dependence.

Definition B.2.2 (Gumbel copula)**.** The bivariate **Gumbel copula** is given by

$$C(u_1, u_2) = \exp\left(-\left((-\ln u_1)^{\delta} + (-\ln u_2)^{\delta}\right)^{\frac{1}{\delta}}\right), \quad u_1, u_2 \in [0, 1],$$

where $\delta \in [1, \infty)$ is a parameter controlling the degree of dependence.

Definition B.2.3 (Frank copula)**.** The bivariate **Frank copula** is given by

$$C(u_1, u_2) = -\frac{1}{\delta}\ln\left(\frac{1 - e^{-\delta} - (1 - e^{-\delta u_1})(1 - e^{-\delta u_2})}{1 - e^{-\delta}}\right), \quad u_1, u_2 \in [0, 1],$$

where $\delta \in [-\infty, \infty] \setminus \{0\}$ is a parameter controlling the degree of dependence.

Definition B.2.4 (Joe copula)**.** The bivariate **Joe copula** is given by

$$C(u_1, u_2) = 1 - \left((1 - u_1)^{\delta} + (1 - u_2)^{\delta} - (1 - u_1)^{\delta}(1 - u_2)^{\delta}\right)^{\frac{1}{\delta}}, \quad u_1, u_2 \in [0, 1],$$

where $\delta \in [1, \infty)$ is a parameter controlling the degree of dependence.

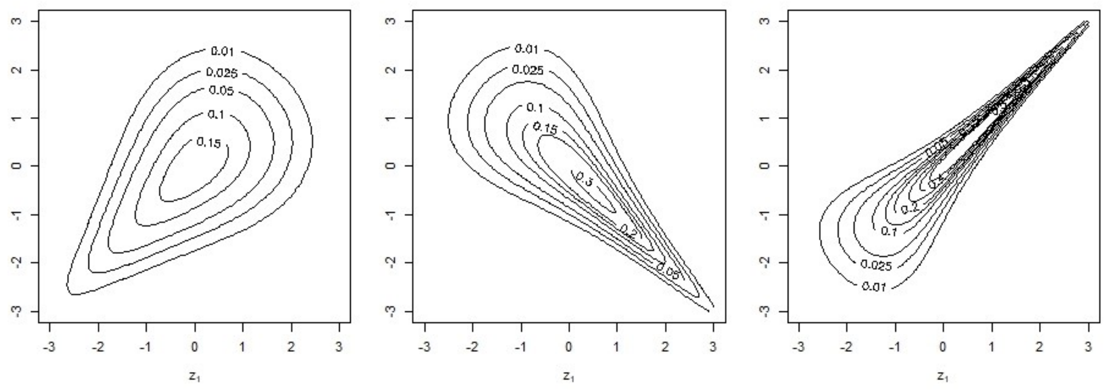


Figure B.4 Normalized contour plots of the Clayton copula density for $\delta = 1, 3, 8$ and rotations 0°, 90°, 180° yielding $\tau \approx 0.33, -0.6, 0.8$ (left to right)

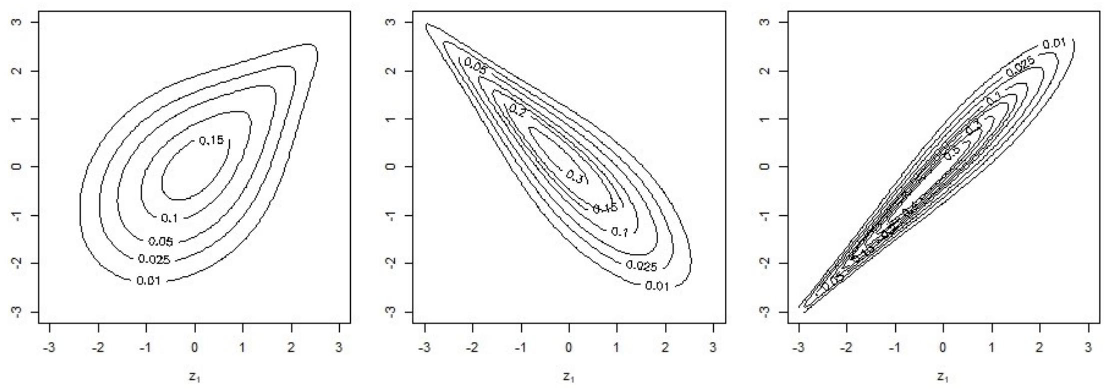


Figure B.5 Normalized contour plots of the Gumbel copula density for $\delta = 1.5, 3, 6$ and rotations 0°, 90°, 180° yielding $\tau \approx 0.33, -0.67, 0.83$ (left to right)

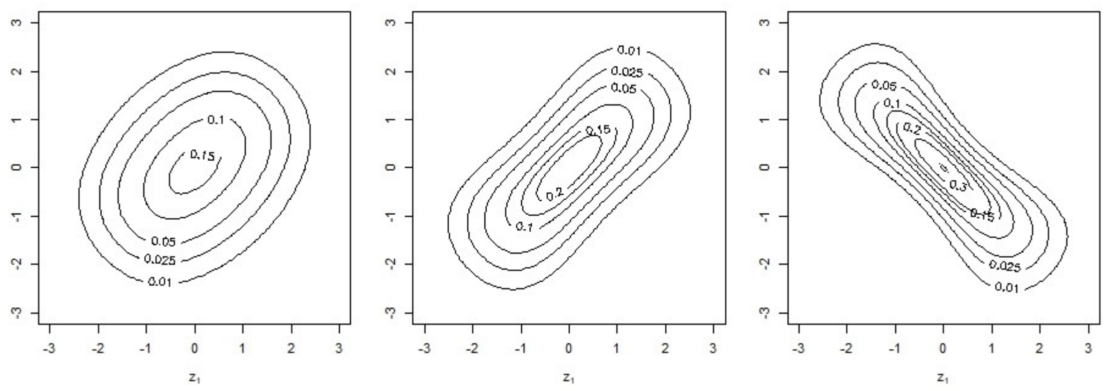


Figure B.6 Normalized contour plots of the Frank copula density for $\delta = 2, 6, -10$ and rotation 0° yielding $\tau \approx 0.21, 0.51, -0.67$ (left to right)

B.3 BB Copulas

Definition B.3.1 (BB1 copula)**.** The bivariate **BB1 copula** is given by

$$C(u_1, u_2) = \left(1 + \left((u_1^{-\theta} - 1)^{\delta} + (u_2^{-\theta} - 1)^{\delta}\right)^{\frac{1}{\delta}}\right)^{-1/\theta}, \quad u_1, u_2 \in [0, 1],$$

where $\delta \geq 1$ and $\theta > 0$.

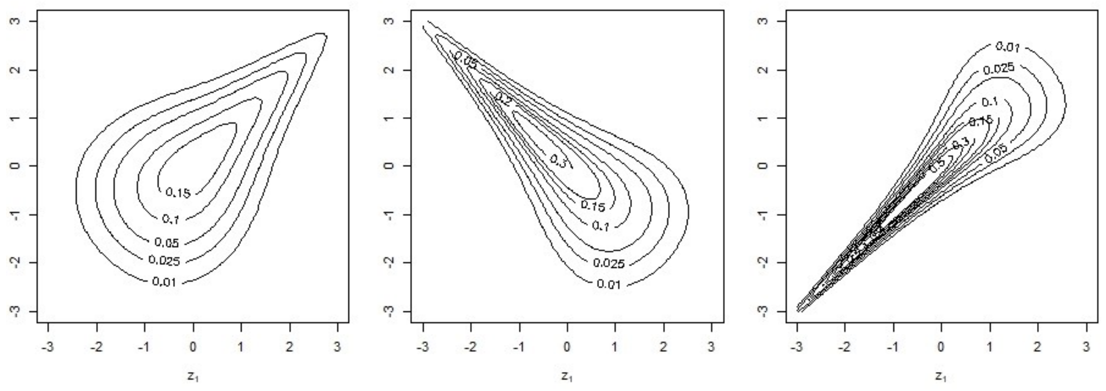


Figure B.7 Normalized contour plots of the Joe copula density for $\delta = 2, 4, 8$ and rotations 0°, 90°, 180° yielding $\tau \approx 0.36, -0.61, 0.78$ (left to right)

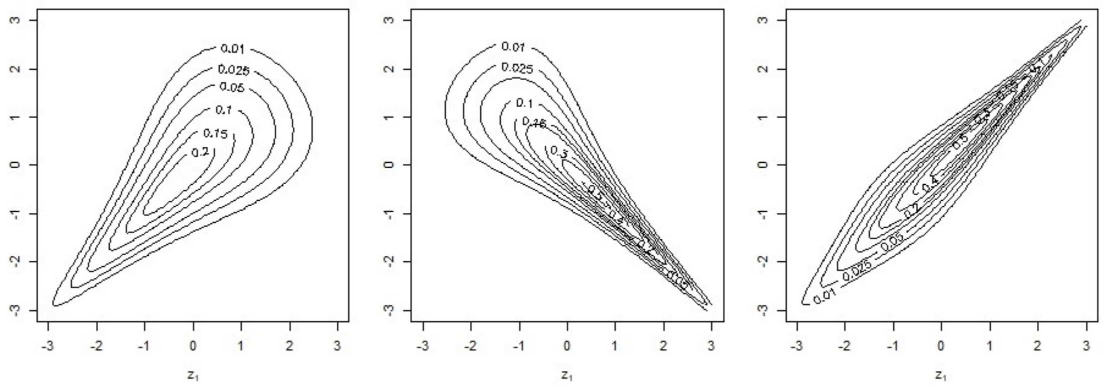


Figure B.8 Normalized contour plots of the BB1 copula density for $(\delta, \theta) = (1,2), (1,5), (2.5,2)$ and rotations 0°, 90°, 180° yielding $\tau \approx 0.5, -0.71, 0.8$ (left to right)

Definition B.3.2 (BB6 copula)**.** The bivariate **BB6 copula** is given by

$$C(u_1, u_2) = 1 - \left(1 - \exp\Big(-\big((-\ln(1 - \overline{u}_1^{\theta}))^{\delta} + (-\ln(1 - \overline{u}_2^{\theta}))^{\delta}\big)^{\frac{1}{\delta}}\Big)\right)^{1/\theta}, \quad u_1, u_2 \in [0,1],$$

where $\delta, \theta \geq 1$ and $\overline{u}_i = 1 - u_i$, for $i = 1, 2$.

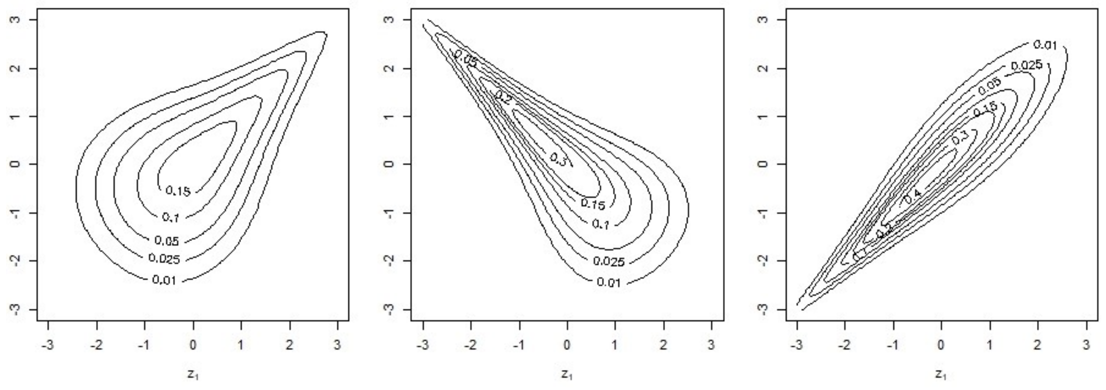


Figure B.9 Normalized contour plots of the BB6 copula density for $(\delta, \theta) = (1,2), (1,4), (4,1)$ and rotations 0°, 90°, 180° yielding $\tau \approx 0.36, -0.61, 0.75$ (left to right)

Definition B.3.3 (BB7 copula)**.** The bivariate **BB7 copula** is given by

$$C(u_1, u_2) = 1 - \left(1 - \left((1 - \overline{u}_1^{\theta})^{-\delta} + (1 - \overline{u}_2^{\theta})^{-\delta} - 1\right)^{-\frac{1}{\delta}}\right)^{1/\theta}, \quad u_1, u_2 \in [0, 1],$$

where $\delta > 0$, $\theta \geq 1$, and $\overline{u}_i = 1 - u_i$, for $i = 1, 2$.

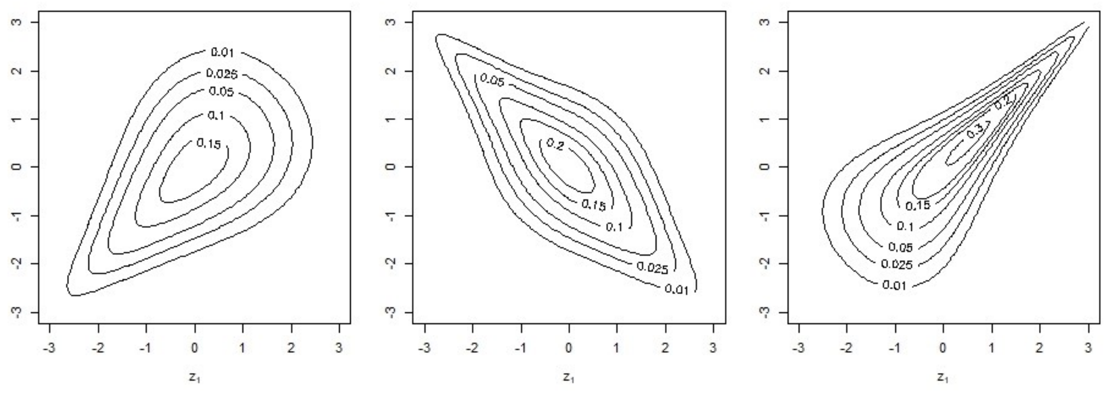


Figure B.10 Normalized contour plots of the BB7 copula density for $(\delta, \theta) = (1, 1), (1, 2), (3, 1)$ and rotations 0°, 90°, 180° yielding $\tau \approx 0.33, -0.5, 0.6$ (left to right)

Definition B.3.4 (BB8 copula)**.** The bivariate **BB8 copula** is given by

$$C(u_1, u_2) = \frac{1}{\delta}\left(1 - \left(1 - \eta^{-1}\left(1 - (1 - \delta u_1)^{\vartheta}\right)\left(1 - (1 - \delta u_2)^{\vartheta}\right)\right)^{1/\vartheta}\right), \quad u_1, u_2 \in [0, 1],$$

where $\delta \in (0, 1]$, $\vartheta \geq 1$, and $\eta = 1 - (1 - \delta)^{\vartheta}$.

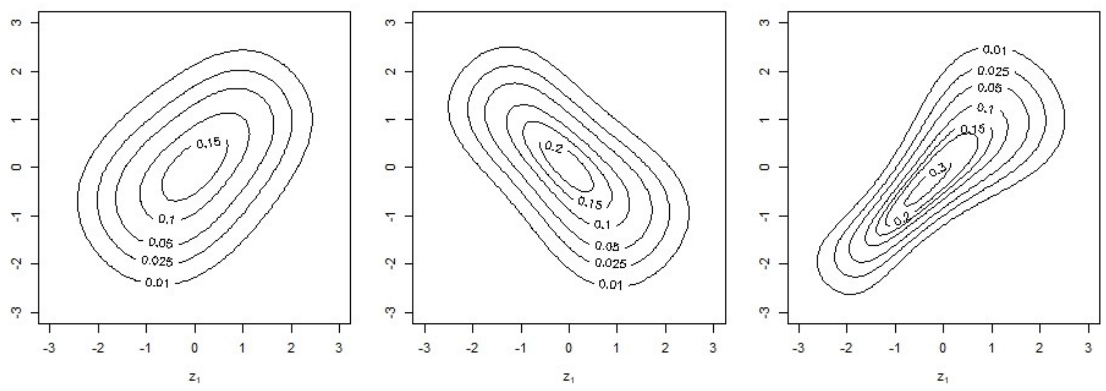


Figure B.11 Normalized contour plots of the BB8 copula density for $(\delta, \vartheta) = (0.5, 5), (0.5, 8), (0.9, 5)$ and rotations 0°, 90°, 180° yielding $\tau \approx 0.31, -0.46, 0.61$ (left to right)

List of Figures

List of Tables

Bibliography

Aas, K. et al. (2021). “Explaining Predictive Models Using Shapley Values and Non-Parametric Vine Copulas”. In: *Dependence Modeling* 9, pp. 62–81.

Aas, K. et al. (2009). “Pair-copula constructions of multiple dependence”. In: *Insurance: Mathematics and Economics* 44.2, pp. 182–198.

Akaike, H. (1998). “Information Theory and an Extension of the Maximum Likelihood Principle”. In: *Selected Papers of Hirotugu Akaike*. Ed. by E. Parzen, K. Tanabe, and G. Kitagawa. Berlin: Springer, pp. 199–213.

Alnasser, H. H. and C. Czado (2022). *An Application of D-Vine Regression for the Identification of Risky Flights in Runway Overrun*. Preprint. Munich, Germany: Faculty of Mathematics, Technical University of Munich.

Anderson, T. (2003). *An Introduction to Multivariate Statistical Analysis*. 3rd ed. John Wiley & Sons.

Baba, K., R. Shibata, and M. Sibuya (2004). “Partial correlation and conditional correlation as measures of conditional independence”. In: *Australian & New Zealand Journal of Statistics* 46.4, pp. 657–664.

Bang-Jensen, J. and G. Z. Gutin (2008). *Digraphs: Theory, Algorithms and Applications*. 2nd. Springer.

Bauer, A. and C. Czado (2016). “Pair-Copula Bayesian Networks”. In: *Journal of Computational and Graphical Statistics* 25.4, pp. 1248–1271.

Bauer, A., C. Czado, and T. Klein (2011). “Pair-copula constructions for non-Gaussian DAG models”. In: *Proceedings of the 58th World Statistical Congress, 2011, Dublin (Session CPS057)*. International Statistical Institute. Dublin.

Bauer, A. X. (2013). “Pair-copula constructions for non-Gaussian Bayesian networks”. Doktorarbeit. Munich, Germany: Technische Universität München.

Bedford, T. and R. M. Cooke (2002). “Vines: A new graphical model for dependent random variables”. In: *Annals of Statistics* 30.4, pp. 1031–1068.

Colombo, D. and M. H. Maathuis (2014). “Order-Independent Constraint-Based Causal Structure Learning”. In: *Journal of Machine Learning Research* 15. Ed. by P. Spirtes. Submitted 9/13; Revised 7/14; Published 11/14, pp. 3921–3962.

Czado, C. (2019). *Analyzing Dependent Data with Vine Copulas: A Practical Guide with R*. Springer Series in Statistics. Cham, Switzerland: Springer. ISBN: 978-3-030-13785-6.

– (2022). “Generalized Linear Models with Applications”. Lecture Notes.

Czado, C. and S. Scharl (2021). “Analysis of an interventional protein experiment using a vine copula based structural equation model”. Version v1. In: *arXiv preprint arXiv:2111.10113*.

Dißmann, J. et al. (2013). “Selecting and estimating regular vine copulae and application to financial returns”. In: *Computational Statistics and Data Analysis* 59, pp. 52–69.

Dor, D. and M. Tarsi (1992). *A simple algorithm to construct a consistent extension of a partially oriented graph*. Technical Report R-185. Tel-Aviv, Israel and Los Angeles, CA: Tel-Aviv University and University of California, Los Angeles.

Drees, L.-W. A. (2016). “Predictive Analysis: Quantifying Operational Airline Risks”. Dissertation. Munich, Germany: Technische Universität München, Institute of Flight System Dynamics.

Edwards, D. (2000). *Introduction to Graphical Modelling*. Second Edition. Springer. ISBN: 978-0-387-98976-4.

Fisher, R. A. (1924). “The distribution of the partial correlation coefficient”. In: *Metron* 3, pp. 329–332.

Fisher, R. A. (1915). “Frequency distribution of the values of the correlation coefficient in samples from an indefinitely large population”. In: *Biometrika* 10.4, pp. 507–521.

Hanea, A. M., D. Kurowicka, and R. M. Cooke (2006). “Hybrid Method for Quantifying and Analyzing Bayesian Belief Nets”. In: *Quality and Reliability Engineering International* 22.6, pp. 709–729.

Hanea, A. and D. Kurowicka (2008). “Mixed Non-Parametric Continuous and Discrete Bayesian Belief Nets”. In: *Advances in Mathematical Modeling for Reliability.* Ed. by T. Bedford et al. Amsterdam, Netherlands: IOS Press, pp. 9–16.

Hoeffding, W. (1948). “A Class of Statistics with Asymptotically Normal Distribution”. In: *The Annals of Mathematical Statistics* 19.3, pp. 293–325.

Hollander, M., D. A. Wolfe, and E. Chicken (2014). *Nonparametric Statistical Methods.* Third. Hoboken, NJ: Wiley.

Horsman, N. (2023). “On the Restrictions of Pair-Copula Bayesian Networks for Integration-Free Computations”. Master’s Thesis. Delft, The Netherlands: Delft University of Technology.

Joe, H. (2014). *Dependence Modeling with Copulas.* Chapman and Hall/CRC.

– (1996). “Families of m-Variate Distributions with Given Margins and m(m-1)/2 Bivariate Dependence Parameters”. In: *Distributions with Fixed Marginals and Related Topics.* Vol. 28. Lecture Notes-Monograph Series. Institute of Mathematical Statistics, pp. 120–141.

Kalisch, M. and P. Bühlmann (2007). “Estimating High-Dimensional Directed Acyclic Graphs with the PC-Algorithm”. In: *Journal of Machine Learning Research* 8. Submitted 9/06; Revised 1/07; Published 3/07, pp. 613–636.

Kalisch, M. et al. (2024). *Package ’pcalg’: Methods for Graphical Models and Causal Inference.* R package version 2.7-11.

Killiches, M., D. Kraus, and C. Czado (2018). “Model distances for vine copulas in high dimensions”. In: *Statistics and Computing* 28, pp. 323–341.

Koller, D. and N. Friedman (2009). *Probabilistic Graphical Models: Principles and Techniques.* MIT Press.

Kraus, D. and C. Czado (2017). “D-vine copula based quantile regression”. In: *Computational Statistics and Data Analysis* 110, pp. 1–18.

Kurowicka, D. (2006). *Uncertainty Analysis with High Dimensional Dependence Modelling.* Chichester: Wiley, pp. VIII, 284. ISBN: 978-0-470-01477-2.

Kurowicka, D. and R. M. Cooke (2005). “Distribution-Free Continuous Bayesian Belief Nets”. In: *Modern Statistical and Mathematical Methods in Reliability.* Ed. by A. Wilson et al. Vol. 28. Series on Quality, Reliability and Engineering Statistics. Singapore: World Scientific, pp. 309–323.

Kurowicka, D. and H. Joe (2011). *Dependence Modeling: Vine Copula Handbook.* Singapore: World Scientific. ISBN: 978-981-4366-94-4.

Lauritzen, S. L. (1996). *Graphical Models.* Oxford: Clarendon Press.

Meek, C. (1995). *Strong completeness and faithfulness in Bayesian networks.* Technical Report. Pittsburgh, PA: Department of Philosophy, Carnegie Mellon University.

Nagarajan, R., M. Scutari, and S. Lèbre (2014). *Bayesian Networks in R: with Applications in Systems Biology.* Use R! Springer. ISBN: 978-1-4614-6445-2.

Nagler, T., C. Bumann, and C. Czado (2019). “Model selection in sparse high-dimensional vine copula models with an application to portfolio risk”. In: *Journal of Multivariate Analysis* 172, pp. 180–192.

Nagler, T. and D. Kraus (2024). *D-Vine Quantile Regression.* R package version 0.10.0.

Nagler, T. and T. Vatter (2024). *Package ‘kde1d‘.* R package version 1.0.7.

– (2023). *Package ’rvinecopulib’: High Performance Algorithms for Vine Copula Modeling.* R package version 0.6.3.1.1.

Parzen, E. (1962). “On Estimation of a Probability Density Function and Mode”. In: *Annals of Mathematical Statistics* 33.3, pp. 1065–1076.

Pearl, J. (1988). *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference.* Morgan Kaufmann.

Rosenblatt, M. (1952). “Remarks on a Multivariate Transformation”. In: *The Annals of Mathematical Statistics* 23.3, pp. 470–472.

Sachs, K. et al. (2005). “Causal protein signaling networks derived from multiparameter single-cell data”. In: *Science* 308, pp. 523–529.

Schwarz, G. (1978). “Estimating the Dimension of a Model”. In: *Annals of Statistics* 6.2, pp. 461–464.

Scutari, M. and T. Silander (2024). *Package 'bnlearn': Bayesian Network Structure Learning, Parameter Learning and Inference.* R package version 5.0.1.

Sheather, S. J. and M. C. Jones (1991). "A Reliable Data-Based Bandwidth Selection Method for Kernel Density Estimation". In: *Journal of the Royal Statistical Society: Series B (Methodological)* 53.3, pp. 683–690.

Sklar, A. (1959). "Fonctions de répartition à n dimensions et leurs marges". In: *Publications de l'Institut de Statistique de l'Université de Paris* 8, pp. 229–231.

Spirtes, P., C. Glymour, and R. Scheines (1993). *Causation, Prediction, and Search.* Vol. 81. Lecture Notes in Statistics. Springer-Verlag.

Tepegjozova, M. (2019). "D- and C-vine Quantile Regression for Large Data Sets". Master's Thesis. Munich, Germany: Technische Universität München, Department of Mathematics.

– (2024). "Statistical learning with vine copulas in regression settings". Doktorarbeit. Munich, Germany: TUM School of Computation, Information and Technology, Technische Universität München.

Tepegjozova, M. and C. Czado (2023). "Bivariate vine copula based regression, bivariate level and quantile curves". Version v2. In: *arXiv preprint arXiv:2205.02557.*

Tepegjozova, M. et al. (2022). "Nonparametric C- and D-vine-based quantile regression". In: *Dependence Modeling* 10.1, pp. 1–21.

Textor, J., B. van der Zander, and A. Ankan (2023). *Package 'dagitty'.* R package version 0.3-4.

Tsamardinos, I., L. E. Brown, and C. F. Aliferis (2006). "The max-min hill-climbing Bayesian network structure learning algorithm". In: *Machine Learning* 65.1, pp. 31–78.

Valz, P. D. and A. I. McLeod (1990). "A Simplified Derivation of the Variance of Kendall's Rank Correlation Coefficient". In: *The American Statistician* 44.1, pp. 39–40.

Verma, T. S. and J. Pearl (1990). "Equivalence and Synthesis of Causal Models". In: *Proceedings of the 6th Conference on Uncertainty in Artificial Intelligence (UAI '90).* Elsevier, pp. 255–270.

Wang, X. et al. (2020). "Calibration of Contributing Factors for Model-Based Predictive Analysis Algorithm Using Polynomial Chaos Expansion Methods". In: *Proceedings of the 30th European Safety and Reliability Conference and the 15th Probabilistic Safety Assessment and Management Conference.* Ed. by P. Baraldi, F. D. Maio, and E. Zio. Singapore: Research Publishing, pp. 2968–2975.

Whittaker, J. (1990). *Graphical Models in Applied Multivariate Statistics.* John Wiley & Sons.

Yule, G. U. (1917). *An Introduction to the Theory of Statistics.* 4th. Charles Griffin & Company, Limited.

Zwirglmaier, K. and D. Straub (2016). "A Discretization Procedure for Rare Events in Bayesian Networks". In: *Reliability Engineering & System Safety* 153, pp. 96–109.