DEPARTMENT OF MATHEMATICS

TECHNISCHE UNIVERSITÄT MÜNCHEN

Master's Thesis

Analysis of Conditional Vine Copula Distributions Using Hamiltonian Monte Carlo

Petra Havlíčková

2022

DEPARTMENT OF MATHEMATICS

TECHNISCHE UNIVERSITÄT MÜNCHEN

Master's Thesis

Analysis of Conditional Vine Copula Distributions Using Hamiltonian Monte Carlo

Author:	Petra Havlíčková
Supervisor:	Prof. Claudia Czado, Ph.D.
Advisor:	Prof. Claudia Czado, Ph.D.
	M.Sc. Ariane Hanebeck
Submission Date:	30.11.2022

I confirm that this master's thesis is my own work and I have documented all sources and material used.

Munich, 30.11.2022 Petra Havlíčková

Acknowledgments

I would like to express my deepest appreciation to Prof. Claudia Czado and M.Sc. Ariane Hanebeck for guiding me in this thesis, for all their valuable advice and ideas. Despite the topic being very interesting, it was not always easy for me to work on the thesis, therefore I am extremely grateful for their patience during the whole time.

Moreover, I would like to extend my sincere gratitude to my family, Peter and friends for their unconditional love and support they provided me all these years.

Abstract

Vine Copulas are popular dependence models that provide flexible multivariate distribution classes by representing a joint distribution as univariate margins plus bivariate copulas characterizing the dependence structures. Sometimes, one is interested in sampling conditional values from those vine copula distributions. If, for any given R-vine, all required components for the conditional distribution are given directly in the representation of the vine, the conditional density can be determined easily. Other conditional densities, however, cannot be expressed because parts of the formula are not given and thus direct sampling from such conditional distribution can be hard. A feasible way to sample from conditional distributions like this is to use a Markov chain Monte Carlo (MCMC) approach, concretely an extension of the Hamiltonian Monte Carlo (HMC) algorithm – the No-U-Turn Sampler that is implemented in the probabilistic programming language Stan. By using that, we take advantage of the need for only proportional densities. By performing various simulation setups, we test whether the sampler proposed by M.Sc. Ariane Hanebeck is correctly sampling from any conditional vine copula distribution. Moreover, we apply the proposed sampler for the analysis of the Uranium data set.

Contents

1. Copula Theory

In this chapter, following Czado (2019), we introduce univariate and multivariate distributions, the concept of copulas and present several copula examples.

1.1. Univariate and Multivariate Distributions

A univariate distribution is a probability distribution of a single random variable. Random variables are generally denoted by capital letters and realizations of the random variables by small letters, i.e. we write $X = x$. We use the letter F for the distribution function and f for the corresponding density function. This density function always exists, since we consider only absolutely continuous distributions. We now introduce one example of a univariate distribution, the normal distribution. The illustration of its density for different parameters is given in Fig. 1.1.

Example 1.1 (Univariate normal distribution (Czado, 2019))**.** *The density of a univariate normal distribution with mean $\mu \in \mathbb{R}$ and variance $\sigma^2 > 0$ is given by*

$$f(x;\mu,\sigma^2) = \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left\{-\frac{1}{\sigma^2}(x-\mu)^2\right\}.$$

We denote a random variable X with a normal distribution with mean μ and variance σ^2 by $X \sim \mathcal{N}(\mu,\sigma^2)$. The distribution $\mathcal{N}(0,1)$ is called standard normal distribution.

One of the problems which can be considered is that the parameters of the distribution function of a random variable X are unknown and need to be estimated. One way to do the estimation is to use a parametric model for X with a parametric vector $\boldsymbol{\theta} \in \Theta$, i.e. $X \sim f(\cdot;\boldsymbol{\theta})$, where Θ is the corresponding parameter space. The estimation is based on a sample $x_1,\ldots,x_n$ of independent identically distributed (i.i.d.) observations of X and the parameter vector $\boldsymbol{\theta}$ is often estimated by the maximum likelihood method

$$\hat{\boldsymbol{\theta}} := \arg\max_{\boldsymbol{\theta}\in\Theta} \prod_{i=1}^{n} f(\cdot;\boldsymbol{\theta}).$$

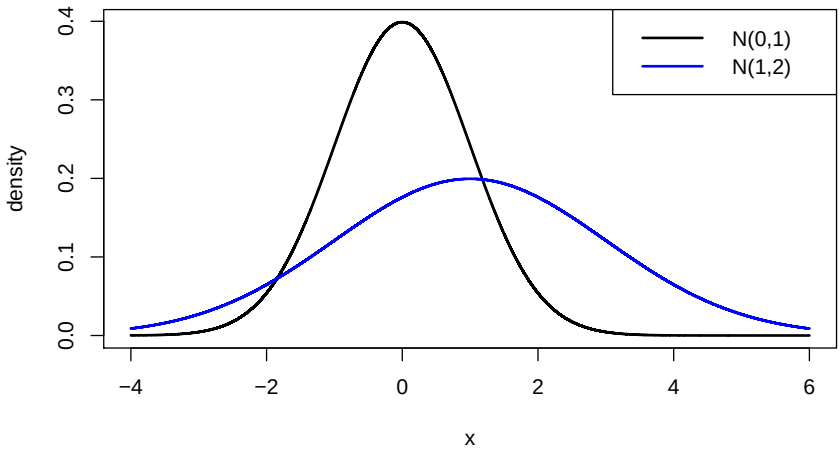

Figure 1.1.: Examples of univariate normal densities.

So, for the distribution function $F(\cdot;\boldsymbol{\theta})$ we have the estimation $F(\cdot;\hat{\boldsymbol{\theta}})$.

If one does not want to make the assumption of a parametric model, the empirical distribution function is often used.

Definition 1.2 (Univariate empirical distribution function (Czado, 2019))**.** *Let $x_1, \ldots, x_n$ be an i.i.d. sample from a distribution function F, then the empirical distribution function is defined as*

$$\hat{F}(x) := \frac{1}{n+1} \sum_{i=1}^{n} \mathbb{1}_{\{x_i \leq x\}},$$

for all x.

Remark 1.3. *Division by $n+1$ instead of n is used to avoid boundary problems of $\hat{F}(x)$.*

Another way to estimate the distribution without the assumption of a parametric model is by kernel density estimation (Parzen, 1962).

Definition 1.4 (Kernel density estimation (Parzen, 1962))**.** *Let $x_1, \ldots, x_n$ be i.i.d samples drawn from a univariate distribution F with a density f at any given point x. The kernel density estimator of f is given by*

$$\hat{f}_h(x) = \frac{1}{nh} \sum_{j=1}^{n} K\left(\frac{x - x_j}{h}\right),$$

where K is the kernel, a symmetric density function, and $h > 0$ is a smoothing parameter. Usually, $K(x) = \phi(x)$ is chosen, where ϕ is the standard normal density function, i.e. a Gaussian kernel.

Remark 1.5 (Role of the smoothing parameter h)**.** *The smoothing parameter h, or also called bandwidth, effects the shape of the corresponding estimator. A badly chosen value may lead to undesired change of the density, therefore a proper choice is a crucial problem. If the bandwidth is small, the estimator is under-smoothed with high variability. On the other hand, if the value is huge, the estimator will be over-smoothed and far from the true function.*

We demonstrate the effect in an example. We sampled $n = 5000$ i.i.d. samples from the $\mathcal{N}(0,1)$ distribution and compared the maximum likelihood estimation (MLE) with the kernel density estimation based on the sample with bandwidth $h = 50$ and $h = 0.1$. The effect on the density shape can be seen in Fig. 1.2.

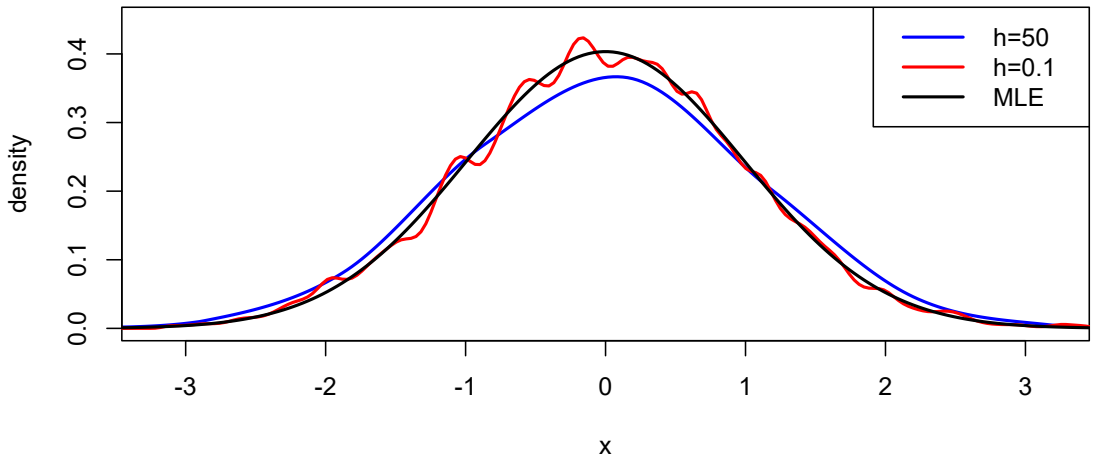

Figure 1.2.: Comparison of the maximum likelihood estimation (MLE) with the kernel density estimation with bandwidth $h = 50$ and bandwidth $h = 0.1$. The samples were drawn from $\mathcal{N}(0,1)$ distribution.

Now, we continue with multivariate distributions, which model the behaviour of several random variables. In this case, we distinguish between marginal, joint, and conditional distributions, which emerge from the multivariate distribution. For each of them, we use the following notation:

Definition 1.6 (Marginal, joint, and conditional distributions (Czado, 2019))**.** *For a d-variate random vector* $\mathbf{X} = (X_1, \ldots, X_d)^T$*, we define the following notations.*

- *Marginal distribution and density function of* X_j*:* $F_j(x_j)$ *and* $f_j(x_j)$*, for* $j = 1, \ldots, d$*.*
- *Joint distribution and density function of* $\mathbf{X}$*:* $F(x_1, \ldots, x_d)$ *and* $f_j(x_1, \ldots, x_d)$*.*
- *Conditional distribution and density function of* X_j *given* X_k*:* $F_{j|k}(x_j|x_k)$ *and* $f_{j|k}(x_j|x_k)$*, for* $j \neq k$*.*

Since we later use them, we introduce two examples of bivariate distributions, the bivariate normal and the bivariate Student's t distribution.

Example 1.7 (Bivariate normal distribution (Czado, 2019))**.** *The density of the bivariate normal distribution with mean vector* $\boldsymbol{\mu} = (\mu_1, \mu_2)^T \in \mathbb{R}^2$ *and positive definite covariance matrix* $\Sigma = (\sigma_{ij})_{i,j=1,2} \in \mathbb{R}^{2x2}$ *is given by*

$$f(\mathbf{x}; \boldsymbol{\mu}, \Sigma) = \frac{1}{2\pi} |\Sigma|^{-1/2} \exp\left\{ -\frac{1}{2} (\mathbf{x} - \boldsymbol{\mu})^T \Sigma^{-1} (\mathbf{x} - \boldsymbol{\mu}) \right\}.$$

In particular $E(X_i) = \mu_i$ *and* $Cov(X_i, X_j) = \sigma_{ij}$ *for all* $i, j = 1, 2$*, where* σ_{ij} *is the* (i, j)*th element of the matrix* Σ*, and* $|\Sigma|$ *denotes the determinant of the matrix* Σ*. We denote a random vector* $\mathbf{X} = (X_1, X_2)^T$ *with a bivariate normal distribution with mean vector* $\boldsymbol{\mu}$ *and covariance* Σ *by* $\mathbf{X} \sim \mathcal{N}_2(\boldsymbol{\mu}, \Sigma)$*.*

Example 1.8 (Bivariate Student's t distribution (Czado, 2019))**.** *The density of the bivariate Student's t distribution with* $\nu > 0$ *degrees of freedom (df), mean vector* $\boldsymbol{\mu} = (\mu_1, \mu_2)^T \in \mathbb{R}^2$ *and symmetric, positive definite scale parameter matrix* $\Sigma = (\rho_{ij})_{i,j=1,2} \in \mathbb{R}^{2x2}$ *is given as*

$$f(\mathbf{x}; \nu, \boldsymbol{\mu}, \Sigma) = \frac{\Gamma(\frac{\nu+2}{2})}{\Gamma(\frac{\nu}{2})(\nu\pi)} |\Sigma|^{-1/2} \left\{ 1 + \frac{1}{\nu} (\mathbf{x} - \boldsymbol{\mu})^T \Sigma^{-1} (\mathbf{x} - \boldsymbol{\mu}) \right\}^{-\frac{\nu+2}{2}}.$$

The matrix Σ *is the correlation matrix of X, i.e.* $Cor(X_i, X_j) = \rho_{ij}$*. We denote a random vector* $\mathbf{X} = (X_1, X_2)^T$ *with a bivariate Student's t distribution with* ν *degrees of freedom, mean vector* $\boldsymbol{\mu}$ *and scale parameter matrix* Σ *by* $\mathbf{X} \sim t_2(\nu, \boldsymbol{\mu}, \Sigma)$*.*

If parametric assumptions are to be avoided again, we can use a multivariate empirical distribution or a multivariate kernel density estimation.

Definition 1.9 (Multivariate empirical distribution (Czado, 2019)). *Let* $\mathbf{x_i} = (x_{i1}, \ldots, x_{id})$ *be an i.i.d. sample of size n from the d-dimensional distribution F, then the multivariate empirical distribution function is defined as*

$$\hat{F}(x_1, \ldots, x_d) := \frac{1}{n+1} \sum_{i=1}^{n} \mathbb{1}_{\{x_{i1} \leq x_1, \ldots, x_{id} \leq x_d\}},$$

for all $\mathbf{x} := (x_1, \ldots, x_d)^T \in \mathbb{R}^d$.

Definition 1.10 (Multivariate kernel density estimation (Duong, 2016)).
Let $\mathbf{x_i} = (x_{i1}, \ldots, x_{id})$ *be an i.i.d. sample of size n drawn from the d-dimensional distribution F with a density f, then the kernel density estimator is defined to be*

$$\hat{f}_{\mathbf{H}}(\mathbf{x}) = \frac{1}{n} \sum_{i=1}^{n} K_{\mathbf{H}}(\mathbf{x} - \mathbf{x}_i),$$

where again K is the kernel, a symmetric multivariate density function, and H is a smoothing $d * d$ *parameter matrix, which is symmetric and positive definite. The scaled kernel is defined as* $K_{\mathbf{H}}(\mathbf{x}) = |\mathbf{H}|^{-1/2} K(\mathbf{H}^{-1/2}\mathbf{x})$. *For the kernel function K the standard multivariate normal kernel is often used.*

In spite of the kernel density estimation being an important technique in multivariate data analysis, its performance worsens with high dimensional data as shown in Huber (1985). This phenomenon is called "curse of dimensionality".

Further, in order to characterize the dependence between multiple random variables, we need to standardize them. For this purpose, we use the probability integral transform for the margins.

Definition 1.11 (Probability integral transform (Czado, 2019)). *If* $X \sim F$ *is a continuous random variable and x is an observed value of X, then the transformation* $u := F(x)$ *is called probability integral transform (PIT) at x.*

Remark 1.12 (Distribution of the probability integral transform (Czado, 2019)). *If* $X \sim F$, *then* $U := F(X)$ *is uniformly distributed, since*

$$P(U \leq u) = P(F(X) \leq u) = P(X \leq F^{-1}(u)) = F(F^{-1}(u)) = u$$

holds for every $u \in [0, 1]$.

By using the probability integral transform, we can transform the set of random variables $(X_1, \ldots, X_d)$ from the original (x-) scale to $(U_1, \ldots, U_d) = (F_1(X_1), \ldots, F_d(X_d))$, the so-called copula scale (u-scale).

1.2. Concept of a Copula

The copula approach to multivariate data allows individual modelling of marginal distributions. The dependence between the components is thus separated from the margins. To see if there is any dependence between the random variables, we standardize them using the probability integral transform for each margin and obtain uniform marginal distributions. The dependence of these marginally standardized random variables is then modeled by a corresponding joint distribution function called copula.

Definition 1.13 (Copula (Czado, 2019))**.** *A d-dimensional copula C is a multivariate distribution function on the d dimensional hypercube* $[0,1]^d$ *with uniformly distributed marginals.*

Definition 1.14 (Copula density (Czado, 2019))**.** *The corresponding copula density for an absolutely continuous copula C, denoted by c, can be obtained by partial differentiation, i.e.:*

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

for all u in $[0,1]^d$.

One of the central results in the theory of copulas is Sklar's Theorem, which was first published in Sklar (1959). Sklar showed that any multivariate distribution can be represented in terms of their marginal distributions and a corresponding copula. The proof of this theorem can be found in Nelsen (2007).

Theorem 1.15 (Sklar's Theorem (Czado, 2019))**.** *Let* $\mathbf{X}$ *be a d-dimensional random vector with joint distribution function F and marginal distribution functions* F_i *for* $i = 1,\dots,d$*. Then the joint distribution function can be expressed as*

$$F(x_1,\dots,x_d) = C(F_1(x_1),\dots,F_d(x_d))$$

with associated density or probability mass function

$$f(x_1,\dots,x_d) = c(F_1(x_1),\dots,F_d(x_d))f_1(x_1)\dots f_d(x_d)$$

for some d-dimensional copula C with copula density c. For absolutely continuous distributions, the copula C is unique.

The inverse also holds: the copula corresponding to a multivariate distribution function F with marginal distribution functions F_i *for* $i = 1,\dots,d$ *can be expressed as*

$$C(u_1,\dots,u_d) = F(F_1^{-1}(u_1),\dots,F_d^{-1}(u_d))$$

and its copula density or probability mass function is determined by

$$c(u_1, \dots, u_d) = \frac{f(F_1^{-1}(u_1), \dots, F_d^{-1}(u_d))}{f_1(F_1^{-1}(u_1)) \dots f_d(F_d^{-1}(u_d))}.$$

Thanks to Sklar's Theorem, we are able to express conditional density and distribution functions of bivariate distributions by an associated bivariate copula and a marginal density.

Lemma 1.16 (Conditional densities and distributions functions of bivariate distributions in terms of their copula (Czado, 2019, p. 20, Lemma 1.15))**.** *The conditional density and distribution function can be rewritten as*

$$\begin{aligned} f_{1|2}(x_1|x_2) &= c_{12}(F_1(x_1), F_2(x_2)) f_1(x_1) \\ F_{1|2}(x_1|x_2) &= \frac{\partial}{\partial u_2} C_{12}(F_1(x_1), u_2)|_{u_2=F_2(x_2)} \\ &=: \frac{\partial}{\partial F_2(x_2)} C_{12}(F_1(x_1), F_2(x_2)) \end{aligned}$$

Proof. Using the definition of a conditional density and Sklar's Theorem 1.15, we have

$$\begin{aligned} f_{1|2}(x_1|x_2) &= \frac{f_{12(x_1,x_2)}}{f_2(x_2)} \\ &= \frac{c_{12}(F_1(x_1), F_2(x_2)) f_1(x_1) f_2(x_2)}{f_2(x_2)} \\ &= c_{12}(F_1(x_1), F_2(x_2)) f_1(x_1) \\ &= \frac{\partial^2 C_{12}(u_1, u_2)}{\partial u_1 \partial u_2}|_{u_1=F_1(x_1), u_2=F_2(x_2)} \frac{\partial u_1}{\partial x_1} \\ &= \frac{\partial}{\partial u_2} \left(\frac{\partial}{\partial x_1} C_{12}(F_1(x_1), u_2) \right) |_{u_2=F_2(x_2)}. \end{aligned}$$

Using the last expression for $f_{1|2}(x_1|x_2)$, we can prove the part about the conditional distribution function

$$\begin{aligned} F_{1|2}(x_1|x_2) &= \int_{-\infty}^{x_1} \frac{\partial}{\partial u_2} \left(\frac{\partial}{\partial z_1} C_{12}(F_1(z_1), u_2) \right) |_{u_2=F_2(x_2)} dz_1 \\ &= \frac{\partial}{\partial u_2} \left(\int_{-\infty}^{x_1} \frac{\partial}{\partial z_1} C_{12}(F_1(z_1), u_2) dz_1 \right) |_{u_2=F_2(x_2)} \\ &= \frac{\partial}{\partial u_2} C_{12}(F_1(x_1), u_2)|_{u_2=F_2(x_2)}. \end{aligned}$$

□

By applying Lemma 1.16 to the bivariate copula distribution C_{12}, we get

$$C_{1|2}(u_1|u_2) = \frac{\partial}{\partial u_2} C_{12}(u_1, u_2) \quad \forall u_1 \in [0,1]. \tag{1.1}$$

In this case, the conditional distribution and density are denoted by $C_{1|2}$ and $c_{1|2}$, respectively. It is now possible to obtain the relationship between $C_{1|2}$ and $F_{1|2}$ as

$$F_{1|2}(x_1|x_2) = \frac{\partial}{\partial u_2} C_{12}(F_1(x_1), u_2)|_{u_2=F_2(x_2)} = C_{1|2}(F_1(x_1)|F_2(x_2)). \tag{1.2}$$

Using Eq. (1.2), we obtain the link between the inverse functions of the conditional distribution functions:

$$F_{1|2}^{-1}(u_1|x_2) = F_1^{-1}(C_{1|2}^{-1}(u_1|F_2(x_2))) \text{ for a fixed value } x_2.$$

The conditional distribution function $C_{1|2}$ from Eq. (1.1) associated with a copula is also denoted as an *h*-function.

Definition 1.17 (h-functions of bivariate copulas (Czado, 2019))**.** *The h-functions corresponding to a bivariate copula C_{12} are defined for all $(u_1, u_2) \in [0,1]^2$ as*

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

1.3. Bivariate Copulas

There are three main classes of copulas according to how they were constructed, elliptical, Archimedean and extreme-value copulas. Before we present some of the bivariate copula examples, we introduce a dependence measure to evaluate the dependence between two random variables. There exist several measures, but we will mention one of those that are rank-based, can be expressed in terms of their associated copula, and therefore does not depend on the marginal distributions.

Kendall's tau is defined as the probability of concordance minus the probability of discordance of two random variables X_1 and X_2 and is denoted by τ.

Definition 1.18 (Kendall's tau (Czado, 2019))**.** *The Kendall's τ between the continuous random variables X_1 and X_2 is defined as*

$$\tau(X_1, X_2) = P((X_{11} - X_{21})(X_{12} - X_{22}) > 0) - P((X_{11} - X_{21})(X_{12} - X_{22}) < 0),$$

where (X_{11}, X_{12}) *and* (X_{21}, X_{22}) *are i.i.d. copies of* (X_1, X_2).

Definition 1.19 (Concordant, discordant, and extra pairs (Czado, 2019))**.** *The pair* (x_i, x_j) *is called*

- *concordant if the ordering in* $x^1 := (x_{i1}, x_{j1})$ *is the same as in* $x^2 := (x_{i2}, x_{j2})$*, i.e.* $x_{i1} < x_{j1}$ *and* $x_{i2} < x_{j2}$ *holds or* $x_{i1} > x_{j1}$ *and* $x_{i2} > x_{j2}$ *holds,*
- *discordant if the ordering in* x^1 *is opposite to the ordering of* x^2*, i.e.* $x_{i1} < x_{j1}$ *and* $x_{i2} > x_{j2}$ *holds or* $x_{i1} > x_{j1}$ *and* $x_{i2} < x_{j2}$ *holds,*
- *extra* x_1 *pair if* $x_{i1} = x_{j1}$ *holds,*
- *extra* x_2 *pair if* $x_{i2} = x_{j2}$ *holds.*

Since Kendall's τ is independent of the marginal distribution, it depends exclusively on the associated copula.

Definition 1.20 (Kendall's τ expressed in terms of the copula (Czado, 2019))**.** *Let* (X_1, X_2) *be continuous random variables, then Kendall's* τ *can be expressed as*

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

Now, we present several examples from the mentioned copula classes. They are characterized by the copula family and the corresponding parameter. First, we start with the independence copula, since it does not belong to any of the classes.

Example 1.21 (Bivariate independence copula)**.** *Consider two independent random variables* $U_j \sim U[0,1]$*,* $j = 1, 2$*. The joint distribution of* (U_1, U_2) *is the independence copula with distribution given by*

$$C(u_1, u_2) = u_1 u_2.$$

1.3.1. Elliptical Copulas

By applying the probability integral transform to the margins of elliptical distributions, elliptical copulas appear.

Example 1.22 (Bivariate Gaussian copula (Czado, 2019))**.** *The bivariate Gaussian copula can be constructed using a bivariate normal distribution with zero mean vector, unit variances, and correlation ρ and applying the inverse statement of Sklar's Theorem 1.15 to obtain*

$$C(u_1, u_2; \rho) = \Phi_2(\Phi^{-1}(u_1), \Phi^{-1}(u_2); \rho),$$

where $\Phi(\cdot)$ corresponds to the distribution function of a standard normal $\mathcal{N}(0,1)$ distribution and $\Phi_2(\cdot, \cdot; \rho)$ is the bivariate normal distribution function with zero means, unit variances, and correlation ρ.

Example 1.23 (Bivariate Student t copula (Czado, 2019))**.** *The bivariate Student's t copula can be constructed using a bivariate Student's t distribution and is given as*

$$C(u_1, u_2; R, \nu) = T_{R,\nu}(T_\nu^{-1}(u_1), T_\nu^{-1}(u_2)),$$

where $T_{R,\nu}$ denotes the distribution function of the bivariate standard Student's t distribution with scale parameter matrix $R \in [-1,1]^{2x2}$ and ν degrees of freedom. Further T_ν^{-1} denotes the inverse of the distribution function T_ν of the univariate standard Student's t distribution with ν degrees of freedom.

1.3.2. Archimedean Copulas

Another class of copulas is constructed using generator functions and is called Archimedean copulas.

Definition 1.24 (Bivariate Archimedean copulas (Czado, 2019))**.** *Let Ω be the set of all continuous, strictly monotone decreasing, and convex functions $\varphi : I \to [0, \infty]$ with $\varphi(1) = 0$. Let $\varphi \in \Omega$, then*

$$C(u_1, u_2) = \varphi^{[-1]}(\varphi(u_1) + \varphi(u_2))$$

is a copula. C is called a bivariate Archimedean copula with generator φ. Here $\varphi^{[-1]}$ is the pseudo-inverse of φ, which is defined as $\varphi^{[-1]} : [0, \infty] \to [0, 1]$ with

$$\varphi^{[-1]}(t) := \begin{cases} \varphi^{-1}(t), & 0 \leq t \leq \varphi(0) \\ 0, & \varphi(0) \leq t \leq \infty. \end{cases}$$

Now, we show examples of parametric bivariate Archimedean copulas with a single parameter.

Example 1.25 (Parametric bivariate Archimedean copulas with a single parameter (Czado, 2019))**.**

- *Clayton copula*

$$C(u_1, u_2) = (u_1^{-\delta} + u_2^{-\delta} - 1)^{-\frac{1}{\delta}},$$

where the parameter $\delta : 0 < \delta < \infty$ controls the degree of dependence. When $\delta \to \infty$, full dependence is obtained. On the other hand, independence is obtained, when $\delta \to 0$.

- *Gumbel copula*

$$C(u_1, u_2) = \exp\left[-\{(-\ln u_1)^{\delta} + (-\ln u_2)^{\delta}\}^{\frac{1}{\delta}}\right],$$

where $\delta : \delta \geq 1$ is the parameter of dependence. Full dependence is obtained when $\delta \to \infty$, while when $\delta = 1$, we have independence.

- *Frank copula*

$$C(u_1, u_2) = -\frac{1}{\delta} \ln\left(\frac{1}{1 - e^{-\delta}}[(1 - e^{-\delta}) - (1 - e^{-\delta u_1})(1 - e^{-\delta u_2})]\right),$$

where the parameter δ can take values $[-\infty, \infty] \backslash \{0\}$. Independence corresponds to $\delta \to 0^+$.

- *Joe copula*

$$C(u_1, u_2) = 1 - \left((1 - u_1)^{\delta} + (1 - u_2)^{\delta} - (1 - u_1)^{\delta}(1 - u_2)^{\delta}\right)^{\frac{1}{\delta}},$$

where $\delta \geq 1$. $\delta = 1$ corresponds to independence.

Moreover, there are also Archimedean copulas with two parameters, for example the following ones.

Example 1.26 (Parametric bivariate Archimedean copulas with two parameters (Czado, 2019))**.**

- *BB1 copula*

$$C(u_1, u_2; \theta, \delta) = \left(1 + [(u_1^{-\theta} - 1)^{\delta} + (u_2^{-\theta} - 1)^{\delta}]^{\frac{1}{\delta}}\right)^{-\frac{1}{\theta}},$$

where the parameters are $\delta \geq 1$, $\theta > 0$. For $\delta \to 1^+$ and $\theta \to 0^+$, the independence copula arises.

- *BB7 copula*

$$C(u_1, u_2; \theta, \delta) = 1 - \left(1 - [(1-(1-u_1)^\theta)^{-\delta} + (1-(1-u_2)^\theta)^{-\delta} - 1]^{-\frac{1}{\delta}}\right)^{\frac{1}{\theta}},$$

where $\delta > 0$ and $\theta \geq 1$. Independence is obtained when $\delta = 0$ and $\theta = 1$.

Remark 1.27 (Visualization and variable scales (Czado, 2019))**.** *A good visualization tool for bivariate copula density is the normalized bivariate copula contour plot. For that we consider the transformation to a bivariate distribution with a density $g(z_1, z_2)$ and normal $\mathcal{N}(0,1)$ margins. So, it gives us three variables scales:*

- *x-scale: original scale (X_1, X_2) with density $f(x_1, x_2)$,*
- *u-scale: copula scale (U_1, U_2), where $U_i := F_i(X_i)$ and copula density $c(u_1, u_2)$, and*
- *z-scale: marginal normalized scale (Z_1, Z_2), where $Z_i := \Phi^{-1}(U_i) = \Phi^{-1}(F_i(X_i))$ for $i = 1, 2$ with density*

$$g(z_1, z_2) = c(\Phi(z_1), \Phi(z_2))\phi(z_1)\phi(z_2).$$

Here $\Phi(\cdot)$ and $\phi(\cdot)$ are the distribution and density function of a $\mathcal{N}(0,1)$ variable.

To visualize the copula density, we use contours of the function $g(z_1, z_2)$, i.e. $g(z_1, z_2) = k$ for different values of k. In Fig. 1.3 we show samples of the copula of size $n = 500$ together with the normalized contour plots of their density.

1.3.3. Rotated Copulas

In order to extend the range of dependence, we can use the counterclockwise rotations of the copula density $c(\cdot, \cdot)$, as in Czado (2019), defined by

- $90^\circ : c_{90}(u_1, u_2) := c(1 - u_1, u_2)$,
- $180^\circ : c_{180}(u_1, u_2) := c(1 - u_1, 1 - u_2)$,
- $270^\circ : c_{270}(u_1, u_2) := c(u_1, 1 - u_2)$.

The term rotation is here used in the context of copula density and does not correspond to rotations of the random vector (U_1, U_2). As an example, all rotations of the Clayton copula density are illustrated in Fig. 1.4 using normalized contour plots.

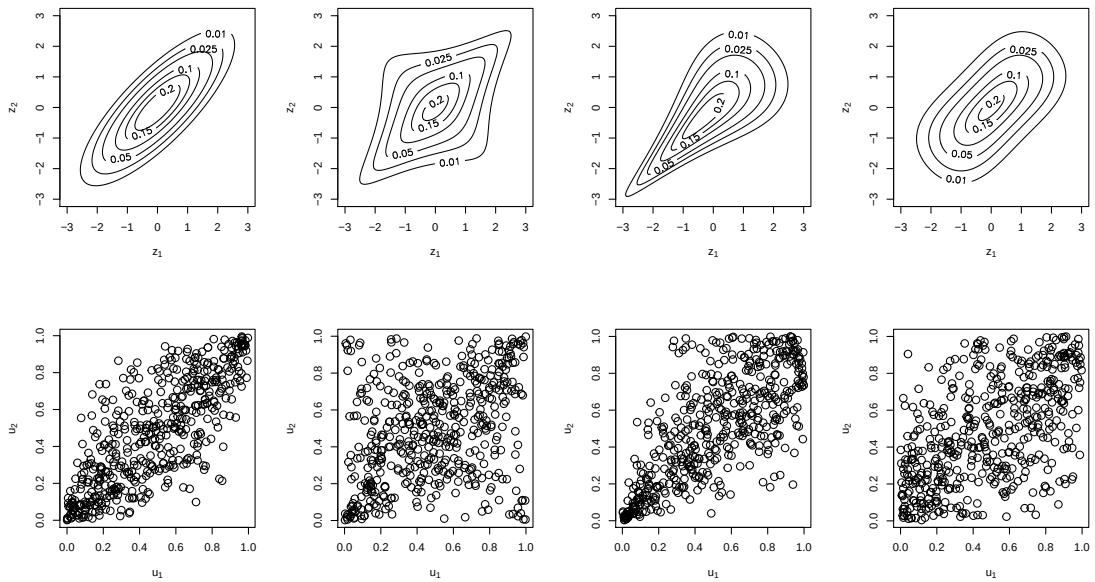

Figure 1.3.: Visualization: First column: Gaussian copula with $\tau = 0.6$, second column: Student's t copula with $\nu = 2$ and $\tau = 0.3$, third column: Clayton copula with $\tau = 0.5$, and fourth column: Frank copula with $\tau = 0.42$. Top row: normalized bivariate copula contours, bottom row: pairs plots of a random sample (u_1, u_2) on the copula scale.

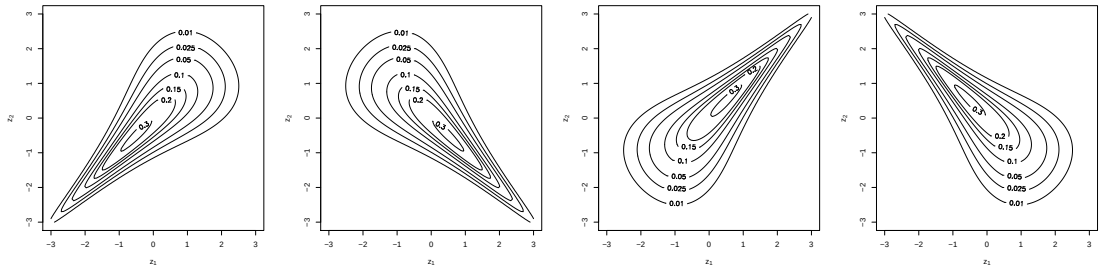

Figure 1.4.: Rotations: Normalized contour plots of Clayton rotations: First column: 0° rotation ($\tau = 0.6$), second column: 90° rotation ($\tau = -0.6$), third column: 180° rotation ($\tau = 0.6$), and fourth column: 270° rotation ($\tau = -0.6$).

2. Regular Vines

The aim of this chapter is to show the construction of multivariate distributions using the building blocks which we defined in the previous chapter, the bivariate pair copulas. To realize such constructions, conditioning is used. We present now the approach developed by Bedford and Cooke (2002), in which the pair copula constructions are expressed in terms of density functions.

2.1. Regular Vine Tree Sequence

First we present the necessary graph theory to then introduce the concept of a regular vine tree structure.

Definition 2.1 (Graph, path, cycle, tree (Czado, 2019))**.**

- *A graph is a pair* $G = (N, E)$ *of sets such that* $E \subseteq \{\{x, y\} : x, y \in N\}$*. The elements of* E *and* N *are called edges and nodes, respectively. The number of neighbors of a node* $v \in N$ *is the degree of* v*, denoted by* $d(v)$*.*
- *A path is a graph* $P = (N, E)$ *with node set* $N = \{v_0, v_1, \ldots, v_k\}$ *and edges* $E = \{\{v_0, v_1\}, \{v_1, v_2\}, \ldots, \{v_{k-1}, v_k\}\}$*. A graph* G *is called connected if any two of its nodes are linked by a path in* G*.*
- *A cycle is a path with* $v_0 = v_k$*.*
- *A tree is a graph* $T = (N, E)$ *for which holds: any two nodes of* T *are connected by a unique path in* T*,* T *is minimally connected (i.e.* T *is connected but* $T - e$ *is disconnected for every edge* $e \in E$*) and* T *is maximally acyclic (i.e.* T *contains no cycle but* $T + \{x, y\}$ *does for any two non-adjacent (not connected by an edge) nodes* $x, y \in N$*).*

Based on the graph theory, a regular vine tree structure can be defined.

Definition 2.2 (Regular (R-) vine tree sequence (Czado, 2019))**.** *The set of trees* $\mathcal{V} = (T_1, \ldots, T_{d-1})$ *is a regular vine tree sequence on d elements if:*

1. *Each tree* $T_j = (N_j, E_j)$ *is connected, i.e. for all nodes* $a, b \in T_j$, $j = 1, \ldots, d-1$, *there exists a path* $n_1, \ldots, n_k \subset N_j$ *with* $a = n_1, b = n_k$.
2. T_1 *is a tree with node set* $N_1 = \{1, \ldots, d\}$ *and edge set* E_1.
3. *For* $j \geq 2$, T_j *is a tree with node set* $N_j = E_{j-1}$ *and edge set* E_j.
4. *For* $j = 2, \ldots, d-1$ *and* $\{a, b\} \in E_j$ *it must hold that* $|a \cap b| = 1$.

Remark 2.3 (Proximity condition (Czado, 2019))**.** *Property* 4. *is called the proximity condition. It ensured that if there is an edge e connecting a and b in tree* T_j, $j \geq 2$, *then a and b (which are edges in* T_{j-1}*) must share a common node in* T_{j-1}.

Now, we introduce a simple edge notation for a regular vine tree sequence and for the later regular vine construction of multivariate distributions.

Definition 2.4 (Complete union, conditioning set and conditioned sets (Czado, 2019))**.** *For any edge* $e \in E_i$ *we define the sets*

- *complete union* A_e *of the edge e*

$$A_e := \{j \in N_1 | \exists e_1 \in E_1, \ldots, e_{i-1} \in E_{i-1} \text{ such that } j \in e_1 \in \cdots \in e_{i-1} \in e\},$$

- *conditioning set* D_e *of an edge* $e = \{a, b\}$

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

- *conditioned sets* $\mathcal{C}_{e,a}$ *and* $\mathcal{C}_{e,b}$

$$\begin{aligned} \mathcal{C}_{e,a} &:= A_a \backslash D_e, \\ \mathcal{C}_{e,b} &:= A_b \backslash D_e, \\ \mathcal{C}_e &:= \mathcal{C}_{e,a} \cup \mathcal{C}_{e,b}. \end{aligned}$$

We will abbreviate each edge $e = (\mathcal{C}_{e,a}, \mathcal{C}_{e,b}; D_e)$ *in the vine tree sequence by*

$$e = (a(e), b(e); D(e)).$$

Further, we consider two important sub-classes of R-vine tree sequences, drawable (D-) vine and canonical (C-) vine tree sequence.

Definition 2.5 (D-vine tree sequence, C-vine tree sequence (Czado, 2019))**.** *A regular vine tree sequence $\mathcal{V} = (T_1, \ldots, T_{d-1})$ is called*

- *D-vine tree sequence if for each node $n \in N_i$ we have $|\{e \in E_i | n \in e\}| \leq 2$.*
- *C-vine tree sequence if in each tree T_i there is one node $n \in N_i$ such that $|\{e \in E_i | n \in e\}| = d - i$. Such a node is called the root node of tree T_i.*

Remark 2.6 (Proximity condition on C- and D-vine tree sequences (Czado, 2019))**.** *For a D-vine tree sequence the proximity condition of Definition 2.2 induces that once tree T_1 is fixed all other trees T_2 to T_{d-1} are determined. For a C-vine tree sequence the proximity condition allows to choose $d - i + 1$ different root nodes in tree T_i for $i = 1 \ldots, d - 1$.*

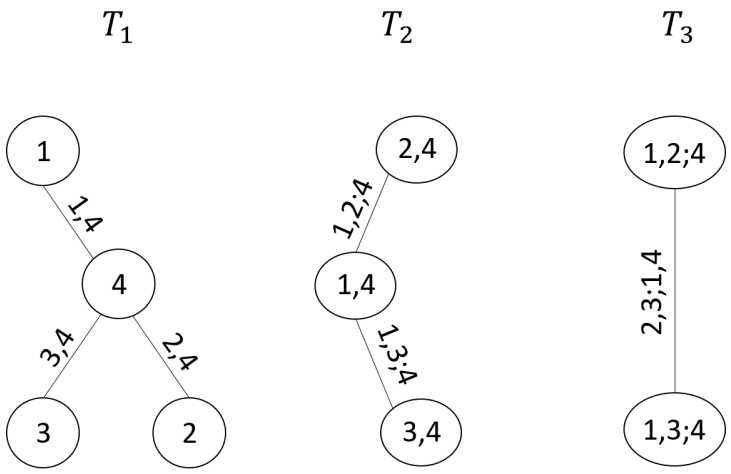

Figure 2.1.: 4-dimensional C-vine tree structure.

Example 2.7 (Example of C-vine tree sequence in 4 dimensions)**.** *An example of a C-vine tree sequence in 4 dimensions can be seen in Fig. 2.1. The specific root order in this case is $4, 14, 12; 4$. We could have also chosen $13; 4$ in tree T_3.*

Example 2.8 (Example of D-vine tree sequence in 4 dimensions)**.** *An example of a D-vine tree sequence in 4 dimensions can be seen in Fig. 2.2.*

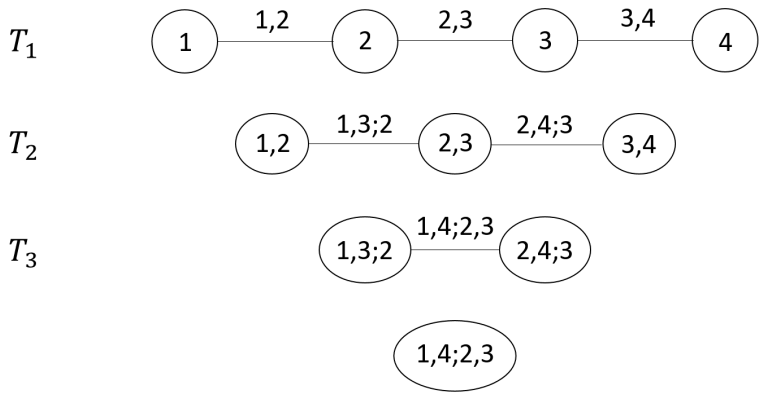

Figure 2.2.: 4-dimensional D-vine tree structure.

2.1.1. Representing Regular Vines Using Regular Vine Matrices

In order to work with arbitrary R-vines, we need a way to store the tree sequence in the computer. Assume the vine tree structure $\{a(e), b(e); D(e) \mid e \in T_j,\ j = 1, \dots, d\}$. Its associated indices are stored in an upper or lower triangular matrix.

Definition 2.9 (Regular vine matrix (Czado, 2019))**.** *Let M be an upper triangular matrix with entries $m_{i,j}$ for $i \leq j$. The elements $m_{i,j}$ can have values between 1 to d. A matrix M is called a regular vine matrix, if it satisfies the following conditions:*

1. $\{m_{1,i}, \dots, m_{i,i}\} \subset \{m_{1,j}, \dots, m_{j,j}\}$ *for* $1 \leq i < j \leq d$ *(The entries of a specific column are also contained in all columns right of this column.)*

2. $m_{i,i} \notin \{m_{1,i-1}, \dots, m_{i-1,i-1}\}$ *(The diagonal entry of a column does not appear in any column further to the left.)*

3. *For* $i = 3, \dots, d$ *and* $k = 1, \dots, i-1$ *there exist* (j, ℓ) *with* $j < i$ *and* $\ell < j$ *such that*

$$\{m_{k,i}, \{m_{1,i}, \dots, m_{k-1,i}\}\} = \{m_{j,j}, \{m_{1,j}, \dots, m_{\ell,j}\}\} \quad \text{or}$$
$$\{m_{k,i}, \{m_{1,i}, \dots, m_{k-1,i}\}\} = \{m_{\ell,j}, \{m_{1,j}, \dots, m_{\ell-1,j}, m_{j,j}\}\}.$$

There exists a bijection between regular vine trees and regular vine matrices, which is shown with this definition. The last assumption of the definition is the analogue of the proximity condition for regular vine trees. The algorithms for computing a regular vine matrix or constructing a tree sequence from an R-vine matrix can be found in

Stöber and Czado (2017). We present the algorithm for computing the R-vine matrix in Algorithm 1. Moreover, we illustrate the algorithm in an example.

Algorithm 1 Computing a regular vine matrix for a regular vine tree sequence $\mathcal{V}$ (Czado, 2019)

The input of the algorithm is a regular vine tree sequence $\mathcal{V} = (T_1, \dots, T_{d-1})$ and the output will be a regular vine matrix M.
$\mathcal{X} \leftarrow \{\}$
for $i = d, \dots, 3$ **do**
  Choose $x, \tilde{x}, D$ with $\tilde{x} \notin \mathcal{X}$ and $|D| = i - 2$ such that there is an edge e with $C_e = \{x, \tilde{x}\}$, $D_e = D$
  $m_{i,i} \leftarrow x$, $m_{i-1,i} \leftarrow \tilde{x}$
  for $k = i - 2, \dots, 1$ **do**
    Choose $\tilde{x}$ such that there is an edge e with $C_e = \{x, \tilde{x}\}$ and $|D| = k - 1$
    $m_{k,i} \leftarrow \tilde{x}$
  end for
  $\mathcal{X} \leftarrow \mathcal{X} \cup \{x\}$
end for
Choose $x, \tilde{x} \in \{1, \dots, d\} \backslash \mathcal{X}$
$m_{2,2} \leftarrow x$, $m_{1,2} \leftarrow \tilde{x}$, $m_{1,1} \leftarrow \tilde{x}$
$M \leftarrow (m_{k,i} | k = 1, \dots, d,\ k \leq i)$
return M

Example 2.10 (Construction of a regular vine matrix for the vine tree sequence from Example 2.7 shown in Fig. 2.1)**.** *In the first step we choose one of the entries of the conditioned set of the single edge in the last tree T_3, i.e.* 2 *or* 3 *from edge* $2, 3; 1, 4$*, and put it in the lower right corner of a d-dimensional matrix.*

Selecting for example the element 3 *we write down all indices that are in the conditioned set of an edge together with* 3 *(bolded in Fig. 2.3). These are the numbers* $2, 1, 4$*. We order them in this way, since* 2 *occurs in T_3,* 1 *in T_2 and* 4 *in T_1. Choosing* 3 *together with such a number and the numbers above this entry identifies a particular edge in the vine tree sequence. For instance the entries* 3 *and* 1 *and the entry* 4 *above in the last column of the matrix on the left panel in Fig. 2.3 identifies the edge* $1, 3; 4$*. You can see it highlighted with orange colour. In summary, by the last column the edge* $2, 3; 4$*, the edge* $1, 3; 4$ *and the edge* $3, 4$ *are identified. Generally, we*

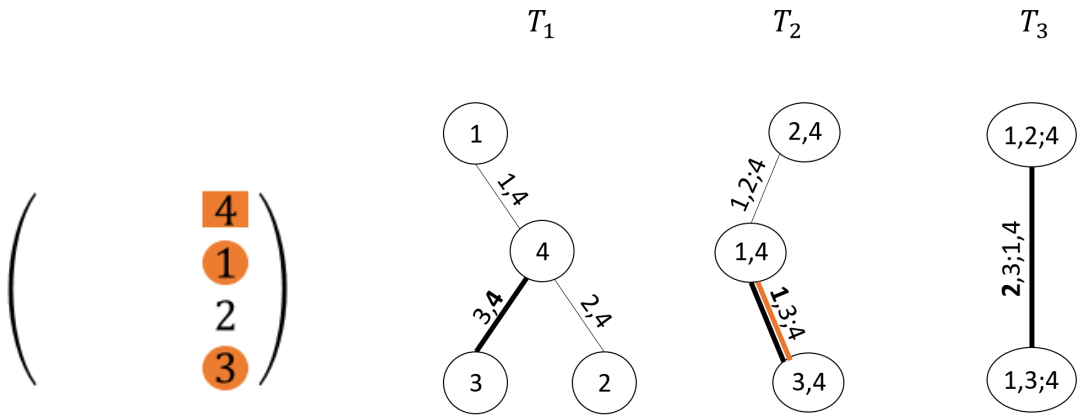

Figure 2.3.: R-vine matrix: Construction of the last column of the R-vine matrix corresponding to Example 2.7.

order the entries of conditioning and conditioned set in increasing order.

Therefore, in the last column of matrix M all pair copula terms characterizing the dependence of X_3 on X_1, X_2, X_4 are stored. Now, we remove all nodes and edges of the vine tree sequence that contain the index 3*. These are the ones we have just recorded into the matrix and we end up with the following reduced vine tree sequence given in Fig. 2.4.*

With this second vine tree sequence we redo the procedure above, selecting for instance 2 *in tree T_2 of Fig. 2.4. and putting it as a diagonal element of the second last column of the matrix. We add the entries which are in the conditioned sets with* 2 *ordered by the tree level they are occurring in, and fill the matrix as shown on the left panel of Fig. 2.4. The selected nodes in the second last column are then removed.*

These steps are repeated until all nodes of the original vine tree sequence have been removed, resulting in the final regular vine matrix, which is:

$$\begin{pmatrix} 4 & 4 & 4 & 4 \\ & 1 & 1 & 1 \\ & & 2 & 2 \\ & & & 3 \end{pmatrix}.$$

Once a regular vine tree sequence is given, we can always determine an R-vine matrix. It is possible also to consider the reverse problem, drawing the associated R-vine tree sequence from a given R-vine matrix. This algorithm inverts the procedure

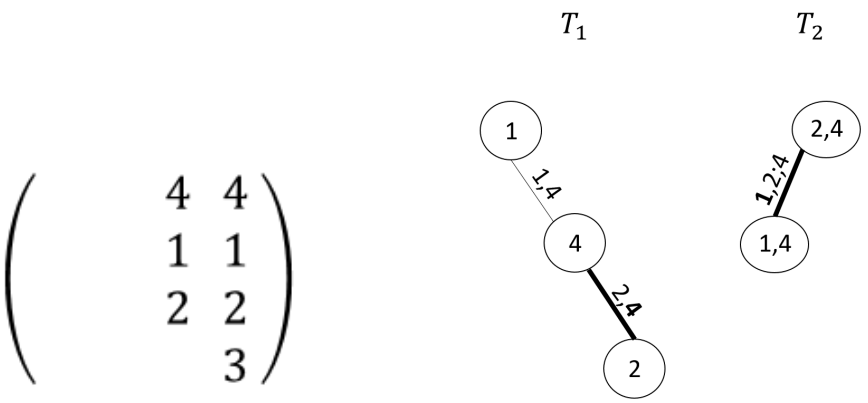

Figure 2.4.: R-vine matrix: Construction of the second last column (left panel) and the reduced vine sequence (right panel) after the first step of constructing the regular vine matrix.

in Algorithm 1 by adding for each identified edge from the R-vine matrix the associated nodes and edges, and can be found in Stöber and Czado (2017).

Remark 2.11 (Non-uniqueness of the R-vine matrix (Czado, 2019))**.** *Since at each step you can choose between the two elements of the conditioned set, the resulting R-vine matrix is not unique. However, it encodes all edges in a vine tree sequence and thus is highly useful for statistical programming with regular vines.*

2.2. Regular Vine Distributions and Copulas

In order to introduce the regular vine distributions and densities, we first present the notion of copulas associated with bivariate conditional distributions in contrast to bivariate conditional distributions on the copula scale. Moreover, we show the evaluation of conditional distribution functions, which will be required in R-vine densities.

Definition 2.12 (Copulas associated with bivariate conditional distributions (Czado, 2019))**.** *Let $(X_1, \ldots, X_d)$ be a set of random variables.*

- *Let D be a set of indices from $\{1, \ldots, d\}$ not including i and j. The copula associated*

with the bivariate conditional distribution of (X_i, X_j) *given that* $X_D = x_d$ *is denoted by* $C_{ij;D}(\cdot, \cdot; x_D)$.

- *In contrast, the conditional distribution function of* (U_i, U_j) *given* $U_D = u_D$ *is expressed as* $C_{ij|D}(\cdot, \cdot; u_D)$ *with bivariate density function* $c_{ij|D}(\cdot, \cdot; u_D)$.
- *For distinct indices* i, j *and* $D := \{i_1, \ldots, i_k\}$ *with* $i < j$ *and* $i_1 < \cdots < i_k$ *we use the abbreviation*

$$c_{i,j;D} := c_{i,j;D}(F_{i|D}(x_i|x_D), F_{j|D}(x_j|x_D); x_D). \tag{2.1}$$

Definition 2.13 (h-functions associated with the pair copula (Kraus and Czado, 2017a))**.** *Given the pair copula* $C_{ij;D}$ *corresponding to the specific edge in a regular vine tree sequence,* $i, j \notin D$*, the h-functions are given by*

$$h_{i|j;D}(u_i|u_j) = \frac{\partial C_{ij;D}(u_i, u_j)}{\partial u_j} = C_{i|j;D}(u_i|u_j),$$

$$h_{j|i;D}(u_j|u_i) = \frac{\partial C_{ij;D}(u_i, u_j)}{\partial u_i} = C_{j|i;D}(u_j|u_i).$$

Theorem 2.14 (Recursion for conditional distribution functions (Kraus and Czado, 2017a))**.** *Let* $l \in D$ *and* $D_{-l} := D \backslash \{l\}$*. Then*

$$C_{i|D}(u_i|\mathbf{u}_D) = h_{i|l;D_{-l}}(C_{i|D_{-l}}(u_i|\mathbf{u}_{D_{-l}})|C_{l|D_{-l}}(u_l|\mathbf{u}_{D_{-l}})),$$

where for $i, j \notin D$*,* $i < j$*,* $h_{i|j;D}(u|v)$ *and* $h_{j|i;D}(v|u)$ *are the h-functions associated with the pair-copula* $C_{ij;D}$*.*

Proof. This result follows directly from the chain rule of differentiation and was first stated in Joe (1996). □

Now, when all needed definitions and notations are introduced, we can move on and present the regular vine distributions and densities.

Definition 2.15 (Regular vine distribution (Czado, 2019))**.** *The joint distribution* F *for the* d*-dimensional random vector* $\mathbf{X} = (X_1, \ldots, X_d)$ *has a regular vine distribution, if we can specify a triplet* $(\mathcal{F}, \mathcal{V}, \mathcal{B})$ *such that:*

1. ***Marginal distributions****:* $\mathcal{F} = (F_1, \ldots, F_d)$ *is a vector of continuous invertible marginal distribution functions, representing the marginal distribution functions of the random variable* X_i*,* $i = 1, \ldots, d$*.*

2. ***Regular vine tree sequence**: $\mathcal{V}$ is an R-vine tree sequence on d elements.*

3. ***Bivariate copulas**: The set $\mathcal{B} = \{C_e | e \in E_i; \quad i = 1, \dots, d-1\}$, where C_e is a symmetric bivariate copula with density. Here E_i is the edge set of tree T_i in the R-vine tree sequence $\mathcal{V}$.*

4. ***Relationship between R-vine tree sequence $\mathcal{V}$ and the set $\mathcal{B}$ of bivariate copulas**: For each $e \in E_i$, $i = 1, \dots, d-1$, $e = \{a, b\}$, C_e is the copula associated with the conditional distribution $X_{a(e)}$ and $X_{b(e)}$ given $\mathbf{X}_{D(e)} = \mathbf{x}_{D(e)}$. Further $C_e(\cdot, \cdot)$ does not depend on the specific value of $\mathbf{x}_{D(e)}$.*

Remark 2.16 (Simplifying assumption for regular vine distributions (Czado, 2019))**.** *The assumption in Definition 2.15 that the bivariate copulas $C_e(\cdot, \cdot)$ do not depend on the specific value of $\mathbf{x}_{D(e)}$ is called the simplifying assumption.*
The simplifying assumption is satisfied when for any $\mathbf{x}_D$

$$c_{i,j;D}(u_i, u_j; \mathbf{x}_D) = c_{i,j;D}(u_i, u_j) \text{ for } u_i \in [0,1], u_j \in [0,1]$$

holds. In the following we always assume that the simplifying assumption is satisfied.

Definition 2.17 (Pair copula and copula density associated with edge e (Czado, 2019))**.** *We will denote the copula C_e corresponding to the edge e by $C_{a(e),b(e);D(e)}$ and the corresponding density by $c_{a(e),b(e);D(e)}$, respectively.*

The R-vine triplet $(\mathcal{F}, \mathcal{V}, \mathcal{B})$ with properties $1. - 3.$ of Definition 2.15 can be uniquely linked to a d-dimensional distribution F, which was showed by Bedford and Cooke (2002). From these the associated joint density can always be constructed.

Definition 2.18 (Regular vine density (Czado, 2019))**.** *Let $(X_1, \dots, X_d)$ be a set of variables with d-dimensional regular vine distribution $F_{1,\dots,d}$. A joint density $f_{1,\dots,d}$ is constructed as*

$$\begin{aligned} f_{1,\dots,d}(x_1, \dots, x_d) =& f_1(x_1) \dots f_d(x_d) \\ & \prod_{i=1}^{d-1} \prod_{e \in E_i} c_{a(e)b(e);D(e)}(F_{a(e)|D(e)}(x_{a(e)}|\mathbf{x}_{D(e)}), F_{b(e)|D(e)}(x_{b(e)}|\mathbf{x}_{D(e)})) \end{aligned}$$

and for each $e \in E_i$, $i = 1, \dots, d-1$ with $e = \{a, b\}$ we have for the distribution of $X_{a(e)}$ and $X_{b(e)}$ given $\mathbf{X}_{D(e)} = \mathbf{x}_{D(e)}$

$$F_{a(e)b(e)|D(e)}(x_{a(e)}, x_{b(e)}|\mathbf{x}_{D(e)}) = C_e(F_{a(e)|D(e)}(x_{a(e)}|\mathbf{x}_{D(e)}), F_{b(e)|D(e)}(x_{b(e)}|\mathbf{x}_{D(e)})).$$

Further the one dimensional margins of F are given by $F_i(x_i)$, $i = 1, \dots, d$.

Remark 2.19 (Regular vine copula density)**.** *The joint density of the d-dimensional regular vine distribution on the copula scale is given by*

$$c(u_1,\dots,u_d)=\prod_{i=1}^{d-1}\prod_{e\in E_i} c_{a(e)b(e);D(e)}(C_{a(e)|D(e)}(u_{a(e)}|\mathbf{u}_{D(e)}),C_{b(e)|D(e)}(u_{b(e)}|\mathbf{u}_{D(e)})), \quad (2.2)$$

where all margins are uniformly distributed on $[0,1]$*. The term* $C_{k(e)|D(e)}$ *denotes the conditional distribution of* $U_{k(e)}$ *given* $\mathbf{U}_{D(e)}=\mathbf{u}_{D(e)}$*. This conditional function can also be identified by an h-function. For this assume that* $k(e)=i$*,* $D(e)=j\cup D$*, then*

$$C_{k(e)|D(e)}=C_{i|j\cup D}=h_{i|j;D}$$

holds. The density in Eq. (2.2) *is a d-variate copula and is called a regular vine copula.*

Definition 2.20 (Drawable (D-) vine density (Czado, 2019))**.** *Let* $(X_1,\dots,X_d)$ *be a set of variables with d-dimensional drawable vine distribution* $F_{1,\dots,d}$*. The joint density* $f_{1,\dots,d}$ *is decomposed as*

$$f_{1,\dots,d}(x_1,\dots,x_d)=\prod_{k=1}^{d} f_k(x_k)\prod_{j=1}^{d-1}\prod_{i=1}^{d-j} c_{i,(i+j);(i+1),\dots,(i+j-1)},$$

where we used the abbreviation introduced in Eq. (2.1)

Example 2.21 (Example of an R-vine distribution in 4 dimensions)**.** *The R-vine density corresponding to the R-vine tree structure given in Fig. 2.1 is*

$$\begin{aligned} f_{1234}(x_1,\dots,x_4) =& f_1(x_1)f_2(x_2)f_3(x_3)f_4(x_4)\\ &\cdot c_{1,4}(F_1(x_1),F_4(x_4))\cdot c_{2,4}(F_2(x_2),F_4(x_4))\cdot c_{3,4}(F_3(x_3),F_4(x_4))\\ &\cdot c_{1,2;4}(F_{1|4}(x_1|x_4),F_{2|4}(x_2|x_4))\cdot c_{1,3;4}(F_{1|4}(x_1|x_4),F_{3|4}(x_3|x_4))\\ &\cdot c_{2,3;1,4}(F_{2|14}(x_2|x_1,x_4),F_{3|14}(x_3|x_1,x_4)), \end{aligned}$$

where we assume that the simplifying assumption of Remark 2.16 holds.

The joint density of the corresponding regular vine copula is

$$\begin{aligned} c_{1234}(u_1,\dots,u_4) =& c_{1,4}(u_1,u_4)\cdot c_{2,4}(u_2,u_4)\cdot c_{3,4}(u_3,u_4)\\ &\cdot c_{1,2;4}(C_{1|4}(u_1|u_4),C_{2|4}(u_2|u_4))\cdot c_{1,3;4}(C_{1|4}(u_1|u_4),C_{3|4}(u_3|u_4))\\ &\cdot c_{2,3;1,4}(C_{2|14}(u_2|u_1,u_4),C_{3|14}(u_3|u_1,u_4)). \end{aligned}$$

Example 2.22 (Evaluation of $C_{2|14}$ and $C_{2|134}$)**.** *Considering the R-vine tree structure from Fig. 2.1 and assuming the simplifying assumption, thanks to the Theorem 2.14 we have*

$$\begin{aligned} C_{2|14}(u_2|u_1,u_4) &= h_{2|1;4}(C_{2|4}(u_2|u_4) \mid C_{1|4}(u_1|u_4)) \\ &= h_{2|1;4}(h_{2|4}(u_2|u_4) \mid h_{1|4}(u_1|u_4)), \end{aligned}$$

and

$$\begin{aligned} C_{2|134}(u_2|u_1,u_3,u_4) &= h_{2|3;14}(C_{2|14}(u_2|u_1,u_4) \mid C_{3|14}(u_3|u_1,u_4)) \\ &= h_{2|3;14}(h_{2|1;4}(C_{2|4}(u_2|u_4) \mid C_{1|4}(u_1|u_4)) \mid \\ &\qquad h_{3|1;4}(C_{3|4}(u_3|u_4) \mid C_{1|4}(u_1|u_4))) \\ &= h_{2|3;14}(h_{2|1;4}(h_{2|4}(u_2|u_4) \mid h_{1|4}(u_1|u_4)) \mid \\ &\qquad h_{3|1;4}(h_{3|4}(u_3|u_4) \mid h_{1|4}(u_1|u_4))). \end{aligned}$$

Note that we used only h-functions of pair copulas associated with the edges specified in the vine tree structure from Fig. 2.1.

2.3. Conditional Regular Vine Distributions

In the previous sections the construction of multivariate distributions using pair copulas is described. The aim of this thesis is to sample from and analyze conditional vine copula distributions and from now on we deal with distributions of variables on the copula scale.

Assume we have random variables $\mathbf{U} = (U_1, \dots U_d)^T$ with regular vine distribution, whose joint density is given in Eq. (2.2). Further assume subsets $\mathcal{C}_1 = \{\mathcal{C}_{1,1}, \dots, \mathcal{C}_{1,k}\}$ and $\mathcal{C}_2 = \{\mathcal{C}_{2,1}, \dots \mathcal{C}_{2,\ell}\}$ of $\mathcal{C} = \{1, \dots, d\}$ with $\mathcal{C}_1 \cap \mathcal{C}_2 = \emptyset$ and $\mathcal{C}_1 \cup \mathcal{C}_2 = \mathcal{C}$. The cardinalities of the subsets $\mathcal{C}_1$ and $\mathcal{C}_2$ are $|\mathcal{C}_1| = k$ and $|\mathcal{C}_2| = \ell$, respectively. We want to analyze the distribution of $(\mathbf{U}_{\mathcal{C}_1} | \mathbf{U}_{\mathcal{C}_2} = \mathbf{u}_{\mathcal{C}_2})$ where $\mathbf{U}_{\mathcal{C}_1} = (U_{\mathcal{C}_{1,1}}, \dots, U_{\mathcal{C}_{1,k}})^T$ and $\mathbf{U}_{\mathcal{C}_2} = (U_{\mathcal{C}_{2,1}}, \dots, U_{\mathcal{C}_{2,\ell}})^T$.

However, sometimes it is hard to determine the conditional density because not all required components in the representation of the vine are given directly. We demonstrate this problem in an example.

Example 2.23 (Determining conditional density from given vine tree structure)**.** *Looking for instance at the 3-dimensional vine given in Fig. 2.5, we can see that the density of* $(U_1|U_2 = u_2, U_3 = u_3)$ *is of the form*

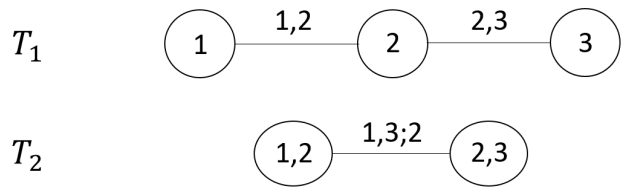

Figure 2.5.: 3-dimensional vine tree structure with copula density given by $c_{123}(u_1, u_2, u_3) = c_{13;2}(h_{1|2}(u_1|u_2), h_{3|2}(u_3|u_2))c_{12}(u_1, u_2)c_{23}(u_2, u_3)$.

$$c_{1|23}(u_1|u_2, u_3) = c_{13;2}(h_{1|2}(u_1|u_2), h_{3|2}(u_3|u_2))c_{12}(u_1, u_2). \tag{2.3}$$

When using the representation of the vine tree structure given in Fig. 2.5, all components of this equation are known. The density of $(U_3|U_1 = u_1, U_2 = u_2)$ *can be expressed in a similar form. On the other hand, when we try to write the density of* $(U_2|U_1 = u_1, U_3 = u_3)$ *we either get*

$$c_{2|13}(u_2|u_1, u_3) = c_{12;3}(h_{1|3}(u_1|u_3), h_{2|3}(u_2|u_3))c_{23}(u_2, u_3) \tag{2.4}$$

or

$$c_{2|13}(u_2|u_1, u_3) = c_{23;1}(h_{2|1}(u_2|u_1), h_{3|1}(u_3|u_1))c_{12}(u_1, u_2). \tag{2.5}$$

But this cannot be expressed with the representation given in Fig. 2.5 that is chosen for this copula since $c_{12;3}$ *and* $c_{23;1}$ *do not occur directly in the 3-dimensional copula representation.*

Generally, for any given R-vine, the density of $(\mathbf{U}_{\mathcal{C}_1}|\mathbf{U}_{\mathcal{C}_2} = \mathbf{u}_{\mathcal{C}_2})$ can be expressed directly provided we have all the components as seen by Eq. (2.3) in Example 2.23, or by using integration and thus as follows

$$c_{\mathcal{C}_1|\mathcal{C}_2}(\mathbf{u}_{\mathcal{C}_1}|\mathbf{u}_{\mathcal{C}_2}) = \frac{c(\mathbf{u}_{\mathcal{C}_1}, \mathbf{u}_{\mathcal{C}_2})}{c(\mathbf{u}_{\mathcal{C}_2})} = \frac{c(\mathbf{u})}{\int_{[0,1]^k} c(\mathbf{u}) du_{\mathcal{C}_{1,1}} \dots du_{\mathcal{C}_{1,k}}},$$

where $\mathbf{u}_{\mathcal{C}_1} \in [0,1]^k$, $\mathbf{u}_{\mathcal{C}_2} \in [0,1]^\ell$. This needs to be used for Eq. (2.4) and Eq. (2.5) in Example 2.23. The similar problem occurs with the distribution functions.

The idea is now to take the advantage of the fact that

$$c_{\mathcal{C}_1|\mathcal{C}_2}(\mathbf{u}_{\mathcal{C}_1}|\mathbf{u}_{\mathcal{C}_2}) = \frac{c(\mathbf{u}_{\mathcal{C}_1}, \mathbf{u}_{\mathcal{C}_2})}{c(\mathbf{u}_{\mathcal{C}_2})} \propto c(\mathbf{u}_{\mathcal{C}_1}, \mathbf{u}_{\mathcal{C}_2}) = c(\mathbf{u}). \tag{2.6}$$

Here we fix the values for $\mathbf{u}_{\mathcal{C}_2}$ and use the joint density $c(\mathbf{u})$ given in Eq. (2.2) as a function of $\mathbf{u}_{\mathcal{C}_1}$ alone to sample from the conditional density.

For this purpose, M.Sc. Ariane Hanebeck wrote a program in `Stan`'s probabilistic programming language (Stan Development Team, 2012), which uses a Markov chain Monte Carlo (MCMC) approach, concretely the extension of Hamiltonian Monte Carlo approach - the NO-U-Turn Sampler algorithm to sample from the conditional densities. Since `Stan` can deal with proportional densities, this joint density $c(\mathbf{u})$ can be used to sample from the density of $(\mathbf{U}_{\mathcal{C}_1}|\mathbf{U}_{\mathcal{C}_2} = \mathbf{u}_{\mathcal{C}_2})$. We talk about these concepts in the following chapters.

3. Hamiltonian Monte Carlo

One way how to sample from a probability distribution is to use Markov Chain Monte Carlo (MCMC) methods. For more details about Markov Chains and MCMC methods see Robert et al. (1999) and Meyn and Tweedie (2012). The desired distribution to sample from is in this case a stationary limiting distribution of the constructed Markov Chain. Two basic methods belong to this class, namely Gibbs sampling and the Metropolis-Hastings (MH) algorithm. We start with a brief description of the MH algorithm as the Hamiltonian Monte Carlo (HMC) algorithm is built on a similar concept. We continue with the HMC algorithm, as this one we use for sampling from conditional vine copula distributions. A useful extension to the HMC is the so called No-U-Turn sampler (NUTS), which is together with the HMC implemented in the programming language `Stan` (Stan Development Team, 2012). This chapter is based on Neal et al. (2011), Betancourt (2017) and Thomas and Tu (2021).

3.1. Metropolis-Hastings Algorithm

The Metropolis-Hastings (MH) algorithm produces a sequence of values that form a Markov Chain. Proposals are generated from a specified proposal function and accepted as a new value in the chain with some probability. This is done in such a way that it ensures that the stationary distribution of the chain is the desired one. The proposal density function is denoted by $q(\boldsymbol{\theta}|\boldsymbol{\theta}^{(r-1)})$ and there are variety of proposal types that can be used. Among them random walk proposals are the most common choice. This density is conditioned on the previous value $\boldsymbol{\theta}^{(r-1)}$. The algorithm is presented in Algorithm 2.

Algorithm 2 Metropolis-Hastings algorithm (Thomas and Tu, 2021)

Choose initial values $\boldsymbol{\theta}^{(0)}$
for $r = 1, \ldots, R$ **do**
  Generate a proposal $\boldsymbol{\theta}^{'}$ from the proposal density $q(\boldsymbol{\theta}|\boldsymbol{\theta}^{(r-1)})$
  $u \leftarrow U[0,1]$
  Set $\alpha = \min\left(1, \frac{\pi(\boldsymbol{\theta}^{'})q(\boldsymbol{\theta}^{(r-1)}|\boldsymbol{\theta}^{'})}{\pi(\boldsymbol{\theta}^{(r-1)})q(\boldsymbol{\theta}^{'}|\boldsymbol{\theta}^{(r-1)})}\right)$
  If $\alpha < u$, then $\boldsymbol{\theta}^{(r)} \leftarrow \boldsymbol{\theta}^{'}$. Otherwise $\boldsymbol{\theta}^{(r)} \leftarrow \boldsymbol{\theta}^{(r-1)}$
end for
return $\boldsymbol{\theta}^{(1)}, \ldots, \boldsymbol{\theta}^{(R)}$

3.2. Hamiltonian Monte Carlo

Hamiltonian Monte Carlo is an MCMC algorithm that improves the efficiency of MH by using the first order gradient information to guide better proposal generation.

Idea

The idea is to construct an artificial physical system that describes the movements of a particle following the Hamiltonian dynamics. Assume that the negative log-density of the desired function has a shape as the curve in Fig. 3.1. The particle moves on this friction-less curve. It often visits the bottom parts, the region of high density values, and occasionally visits the upper parts on both sides with lower density values. In mechanics, these movements are described by the Hamiltonian equations, where the location of the particle is given by the potential and kinetic energy.

The state of this system is composed of the position $\mathbf{q}(t)$ of the particle at time t and its momentum $\mathbf{p}(t)$, with potential energy $U(\mathbf{q}(t))$ and kinetic energy $K(\mathbf{p}(t))$. Both $\mathbf{q}(t)$ and $\mathbf{p}(t)$ are d-dimensional vectors and the system is described by the total energy, a function of both of them, called Hamiltonian: $H(\mathbf{q}(t), \mathbf{p}(t)) = U(\mathbf{q}(t)) + K(\mathbf{p}(t))$. The position $\mathbf{q}(t)$ corresponds to the variable of interest and the potential energy $U(\mathbf{q}(t))$ to minus the log probability density of these variables, i.e. $U(\mathbf{q}(t)) := -log(f(\mathbf{q}(t)))$. Momentum variables $\mathbf{p}(t)$ are introduced artificially, just in order to use it to simulate $\mathbf{q}(t)$. We typically assume $\mathbf{p}(t) \sim \mathcal{N}_d(0, \mathbf{M})$, where $\mathbf{M}$ is a user specified covariance matrix, assumed to be diagonal. Under this formulation we can express the Hamiltonian

as following

$$H(\mathbf{q}(t), \mathbf{p}(t)) = -log(f(\mathbf{q}(t))) + \frac{1}{2}\mathbf{p}(t)^T\mathbf{M}^{-1}\mathbf{p}(t). \tag{3.1}$$

Figure 3.1.: Movement of a particle on a friction-less curve. The picture is taken from Thomas and Tu (2021).

Hamiltonian Equations

To determine the trajectories we utilize the Hamiltonian equations. They determine how $\mathbf{q}(t)$ and $\mathbf{p}(t)$ change over time t using the partial derivatives of the Hamiltonian $H(\mathbf{q}(t), \mathbf{p}(t))$:

$$\frac{dq_i(t)}{dt} = \frac{\partial H(t)}{\partial p_i} = [\mathbf{M}^{-1}\mathbf{p}(t)]_i \tag{3.2}$$

$$\frac{dp_i(t)}{dt} = -\frac{\partial H(t)}{\partial q_i} = -\frac{\partial U(t)}{\partial q_i}, \tag{3.3}$$

for $i = 1, \ldots, d$. $\frac{\partial U}{\partial q_i}$ is the gradient of the log-density. A path of $(\mathbf{q}, \mathbf{p})$ is defined by the solution to this Hamiltonian equations. Then we can sample a value $\mathbf{q}$ from this path within an MCMC iteration.

Leapfrog Method

To solve the system of differential equations given in Eq. (3.2) and Eq. (3.3) is usually neither easy nor possible. By iterative approximation we can find a solution at each time t. For this we use the Leapfrog method. Assume that M is a diagonal matrix with diagonal entries $m_1, \ldots, m_d$ and ϵ a discrete step-size. The state at time $t + \epsilon$ is approximated by:

$$
\begin{aligned}
p_i(t+\epsilon/2) &= p_i(t) - \frac{\epsilon}{2}\frac{d}{dq_i}U(\mathbf{q}(t)) \\
q_i(t+\epsilon) &= q_i(t) + \epsilon\frac{p_i(t+\epsilon/2)}{m_i} \\
p_i(t+\epsilon) &= p_i(t+\epsilon/2) - \frac{\epsilon}{2}\frac{d}{dq_i}U(\mathbf{q}(t+\epsilon)),
\end{aligned}
$$

for $i = 1, \ldots, d$.

For HMC, to move a sufficient distance to the next proposal, several Leapfrog steps L are required. L is, therefore, a parameter representing the number of steps.

Algorithm

First, we specify the corresponding energy function $H(\mathbf{q}(t), \mathbf{p}(t))$, the Hamiltonian, given in Eq. (3.1). Then we specify the initial values $\mathbf{q}(0)$ and $\mathbf{p}(0)$. An MCMC update is obtained as following:

- We sample the parameter $\mathbf{p}$ from the multivariate normal distribution with zero mean vector and covariance matrix $\mathbf{M}$.
- We perform a Metropolis update: We simulate L steps of Hamiltonian dynamics from the current state $(\mathbf{q}, \mathbf{p})$ with the step-size ϵ using the Leapfrog method. We obtain a proposal $(\mathbf{q}', \mathbf{p}')$ and accept it with Metropolis acceptance probability

$$\min\left(1, exp(-H(\mathbf{q}', \mathbf{p}') + H(\mathbf{q}, \mathbf{p}))\right).$$

3.3. No-U-Turn Sampler

In standard HMC, the user needs to specify two tuning parameters, the step-size ϵ and the number of Leapfrog steps L. The performance of the algorithm is sensitive to the choice of this parameters and a bad choice may lead in poor efficiency.

If L is too small, we end up close to the previous iterate. If, on the other hand, L is too large, the algorithm may do too much work for one iteration and loop the trajectory back. If the step-size ϵ is too small, too many small steps will be taken resulting in long simulation times. On the contrary, if ϵ is too large, too many proposals may be rejected due to inaccuracy.

In order to eliminate the need to hand-tune these parameters, Hoffman et al. (2014) proposed an extension to the HMC algorithm, the No-U-Turn sampler (NUTS) that automatically and adaptively selects these tuning parameters.

To choose the number of L steps during the dynamics, the algorithm looks at whether the distance between the proposal $\boldsymbol{\theta}'$ and the initial value of $\boldsymbol{\theta}$ still increases or it starts to decrease, i.e. the proposal starts to make a U-turn and moves back towards $\boldsymbol{\theta}$. Therefore, in each iteration, the L steps are obtained until a U-turn and the new state is randomly chosen from the subset of the states visited during the Leapfrog method.

The step-size ϵ is updated during the burn-in period using stochastic optimization with an adaptation of the primal-dual algorithm.

In `Stan` (Stan Development Team, 2012), this No-U-Turn sampler is implemented, making it easier for the user to work with. During the Leapfrog steps, `Stan` uses automatic differentiation to compute gradients.

4. Statistical Testing

In this chapter, we explain the main concepts of statistical testing and present several statistical methods and tests we later use in a simulation study.

4.1. Brief Introduction to Hypotheses Testing

Following Casella and Berger (2021), we introduce concepts such as hypotheses, statistical testing and p-value. We start with the definition of a sample.

Definition 4.1 (Sample (Casella and Berger, 2021))**.** *The random variables $X_1, \ldots, X_n$ are called a random sample of size n from the population f if $X_1, \ldots, X_n$ are i.i.d. random variables with probability density function or probability mass function f.*

Suppose we have such a sample from a population. We formulate a hypothesis, a statement about the population, which we subsequently want to test. The goal of the hypothesis test (statistical test) is to decide which of the two complementary hypotheses are true, based on the sample from the population.

Definition 4.2 (Hypotheses (Casella and Berger, 2021))**.** *The two complementary hypotheses in a hypotheses testing problem are called null hypothesis and alternative hypothesis. They are denoted by H_0 and H_1, respectively.*

Example 4.3 (Format of the hypotheses (Casella and Berger, 2021))**.** *If θ denotes a population parameter, the general format of the null and alternative hypotheses is $H_0 : \theta \in \Theta_0$ and $H_1 : \theta \in \Theta_0^c$, where Θ_0 is some subset of the parameter space and Θ_0^c is its complement. For example, if θ denotes the average height of 30 year-old men and the current standpoint is 170 cm, an experimenter might be interested in testing $H_0 : \theta = 170$ versus $H_1 : \theta \neq 170$.*

After observing the sample, the experimenter must decide either to accept H_0 as true or to reject H_0 as false and decide H_1 is true. Typically, a hypothesis test is specified in terms of a test statistic $W(X_1, \ldots, X_n) = W(\mathbf{X})$, a function of the sample. And thus, the

test is performed using the test statistic, i.e. it is a base for the conclusion of the test to reject or fail to reject H_0. It can be, for instance, the sample mean and in that case $W(\mathbf{X}) = \bar{X}$ is the test statistic.

The subset of a sample space for which H_0 will be rejected is called rejection region. There are several ways how to choose test statistics and rejection regions, such as Likelihood ratio tests, Bayesian tests, and other. Generally, the tests and rejection rules are constructed that they control error probabilities.

There are two types of errors the hypothesis test can make, Type I Error and Type II Error. If $\theta \in \Theta_0$ and thus H_0 is true, but the test incorrectly decides to reject H_0, then the test makes a Type I Error. If, on the other hand, $\theta \in \Theta_0^c$ and thus H_1 is true, but the test decides to accept H_0, it makes a Type II Error. In Table 4.1 both situations are depicted.

		Decision	
		Accept H_0	Reject H_0
Truth	H_0	Correct decision	Type I Error
	H_1	Type II Error	Correct decision

Table 4.1.: Two types of errors in hypothesis testing, Type I and Type II Error.

Suppose R denotes the rejection region for a test. Then for $\theta \in \Theta_0$, the test makes a mistake when $\mathbf{X} \in R$, so the probability of a Type I Error is $P_\theta(\mathbf{X} \in R)$. For $\theta \in \Theta_0^c$, the probability of a Type II Error is $P_\theta(\mathbf{X} \in R^c) = 1 - P_\theta(\mathbf{X} \in R)$. The test with its rejection rule is designed to control the probability of Type I Error, so that $P_\theta(\mathbf{X} \in R) \leq \alpha$ holds. α is also called the level of significance of the test and is usually set to $\alpha = 0.05$. If α is small, the decision to reject H_0 is fairly convincing, on the other hand, if α is large, the decision to reject H_0 is not that convincing, since the test has a large probability of making that decision incorrectly.

Another way of evaluating and reporting the results of a test is by a p-value.

Definition 4.4 (Valid p-value (Casella and Berger, 2021))**.** *A p-value $p(\mathbf{X})$ is a test statistic satisfying $0 \leq p(\mathbf{x}) \leq 1$ for every sample point $\mathbf{x}$. Small values of $p(\mathbf{X})$ give evidence that H_1 is true. A p-value is valid if, for every $\theta \in \Theta_0$ and every $0 \leq \alpha \leq 1$,*

$$P_\theta(p(\mathbf{X}) \leq \alpha) \leq \alpha. \tag{4.1}$$

If $p(\mathbf{X})$ is a valid p-value, we can construct a level α test based on $p(\mathbf{X})$. Because of Eq. (4.1), the test that rejects H_0 if and only if $p(\mathbf{X}) \leq \alpha$ is a test with the level of significance α. The most common way to define a p-value can be seen in Theorem 4.5.

Theorem 4.5 (Casella and Berger, 2021, p. 397, Theorem 8.3.27)**.** *Let $W(\mathbf{X})$ be a test statistic such that large values of W give evidence that H_1 is true. For each sample point $\mathbf{x}$, define*

$$p(\mathbf{x}) = \sup_{\theta \in \Theta_0} P_\theta(W(\mathbf{X}) \geq W(\mathbf{x})).$$

Then, $p(\mathbf{X})$ is a valid p-value.

Proof. The proof can be found on p. 397 in Casella and Berger (2021). □

In other words, the p-value is the probability, assuming that H_0 is true, of obtaining a test statistic value at least as extreme as the result actually observed. If the p-value is lower than the chosen significance threshold α (equivalently, if the observed test statistic is in the rejection region), we say that the H_0 is rejected at the significance level α. To the contrary, if the p-value is greater then the chosen significance threshold (equivalently, if the observed test statistic is outside the rejection region), then the H_0 is not rejected.

4.2. Statistical Methods

First we present two specific statistical tests and procedure for solving the multiple comparisons problem and subsequently introduce two useful transformations.

In order to measure if a point sample $x_1, \ldots, x_n$ arises from a specified theoretical distribution, a **goodness-of-fit test** statistics is used. One of the most popular ones is a non-parametric test called **Kolmogorov-Smirnov test**. The test statistics together with its asymptotic distribution under the null hypothesis were published in Kolmogorov (1933), while a table of the distribution was published in Smirnov (1939). The test is implemented in R programming language as ks.test().

Definition 4.6 (Kolmogorov-Smirnov test (Dimitrova et al., 2020))**.** *Given a sample $x_1, \ldots, x_n$ of i.i.d random variables with empirical distribution function F_n defined in Definition 1.2, and a reference probability distribution function F, consider a problem of testing*

$$H_0 : F_n \textit{ can be approximated by } F$$

$$H_1 : F_n \text{ cannot be approximated by } F.$$

The test statistic quantifies a distance between empirical distribution function $F_n(x)$ and the cumulative distribution function of the reference distribution, and is defined as

$$T_n = \sup_x |F_n(x) - F(x)|.$$

Under the null hypothesis, if F is continuous, $\sqrt{n}T_n$ converges to the Kolmogorov distribution, which does not depend on the unknown F. The null hypothesis is rejected at significance level α if

$$T_n \geq K_\alpha.$$

The critical values K_α can be found in the table by Smirnov (1948).

Another necessary non-parametric test is a **correlation test**, available in R as cor.test(). With method set to "kendall", we test the hypothesis that the Kendall's Tau coefficient defined in Definition 1.18 is equal to zero. This statistical hypothesis test is used to establish whether two variables may be considered as statistically dependent.

Definition 4.7 (Bivariate dependence test based on Kendall's τ (Hollander et al., 2013))**.** *Let $(x_1, y_1), \ldots, (x_n, y_n)$ be a set of i.i.d. observations of the random variables X and Y. Consider a problem of testing*

$$H_0 : \tau = 0 \qquad vs. \qquad H_1 : \tau \neq 0,$$

where Kendall's τ is defined in Definition 1.18.

The Kendall sample correlation statistic K is computed as follows. Using Definition 1.19, let N_c be the number of concordant pairs, N_d be the number of discordant pairs and $\binom{n}{2}$ be the total number of pairs, in which n is the sample size. Then the test statistic is

$$K = \frac{N_c - N_d}{\binom{n}{2}}.$$

The null hypothesis is rejected at significance level α if

$$|K| \geq k_{1-\alpha/2}.$$

It is a two-sided symmetric test with $\alpha/2$ probability in each tail of the null distribution of K. The values of $k_{1-\alpha/2}$ are found using the R command qKendall. For less than 50 paired samples, an exact p-value of the test is computed this way, on the other hand, for larger samples, the test

statistic is the estimate scaled to zero mean and unit variance, and a normal approximation is utilized. Under the null, the expected value and variance of K are

$$E_0(K) = 0$$

and

$$var_0(K) = \frac{2(2n+5)}{9n(n-1)},$$

respectively. The standardized version of K is

$$K^* = \frac{K - E_0(K)}{\sqrt{var_0(K)}}.$$

When H_0 is true, as n goes to infinity, K^* has approximately $\mathcal{N}(0,1)$ distribution. The null hypothesis is rejected at significance level α if

$$\sqrt{\frac{9n(n-1)}{2(2n+5)}}\,|K^*| \geq z_{1-\alpha/2},$$

where z_β is the β quantile of a standard normal distribution.

These procedures are used when allowing for no ties in X or Y observations. When there are ties, i.e. extra pairs from Definition 1.19, the test statistics K needs to be adjusted to it, as well as its variance. Details on this modification can be found in Hollander et al. (2013).

In a case when we need to perform simultaneously multiple tests, which are not able to be combined into single test, we can run into a problem called **"multiple comparisons problem"** as explained in Lehmann et al. (2005). Let us consider the problem of simultaneously testing a finite number of hypothesis H_i $(i = 1, \ldots, m)$. If we test each hypothesis at level α, the probability of one or more false rejections quickly increases with m. Therefore we need to look at the requirement that when testing several hypothesis, the probability of one or more false rejections do not exceed a given threshold. And such probability is called **family-wise error rate (FWER)**. One of the procedures that control FWER is **Bonferroni procedure**, in which the cut-off for p-values is adjusted to α/m, where m is the number of simultaneous hypotheses.

Theorem 4.8 (Bonferroni procedure (Lehmann et al., 2005, p. 350, Theorem 9.1.1))**.** *If, for $i = 1, \ldots, m$, hypothesis H_i is rejected when p-value $\hat{p}_i \leq \alpha/m$, then the FWER for the simultaneous testing of $H_1, \ldots, H_m$ satisfies $FWER \leq \alpha$.*

Proof. Suppose hypotheses H_i with $i \in I$ are true and the remainder false, with $|I|$ denoting the cardinality of I. Further assume that $\hat{p}_i$ is a p-value for testing H_i. From the Bonferroni inequality it follows that:

$$\begin{aligned} FWER &= P\{\text{reject any } H_i \text{ with } i \in I \text{ at level } \alpha/m\} \leq \\ &\leq \sum_{i \in I} P\{\text{reject } H_i \text{ at level } \alpha/m\} = \\ &= \sum_{i \in I} P\{\hat{p}_i \leq \frac{\alpha}{m}\} \leq \sum_{i \in I} \frac{\alpha}{m} = |I| \frac{\alpha}{m} \leq \alpha. \end{aligned}$$

□

Now, we present two transformations, which help us to compare the sample with a reference distribution. In order to test how the sample agrees with the distribution we use the above mentioned Kolmogorov-Smirnov test. However, it is difficult to use the ks.test() for the vine copula distributions we are dealing with in this thesis. For this, we use the fact, that if a sample comes from the desired distribution, applying probability integral transform, shown in Definition 1.11, should result in uniformly distributed data. And therefore we can compare the transformed data with uniform distribution. However, in more than one dimension, a univariate probability integral transform is not sufficient and for this reason we present **Rosenblatt transformation** by Rosenblatt (1952).

Definition 4.9 (Rosenblatt transformation (Rosenblatt, 1952))**.** *Let $X = (X_1, \ldots, X_k)^T$ be a random vector with distribution function $F(x_1, \ldots, x_k)$. Let $\mathbf{z} = (z_1, \ldots, z_k) = T(x_1, \ldots, x_k)$, where T is the transformation considered. Then T is given by*

$$\begin{aligned} z_1 &= P\{X_1 \leq x_1\} = F_1(x_1), \\ z_2 &= P\{X_2 \leq x_2 | X_1 = x_1\} = F_2(x_2|x_1), \\ &\vdots \\ z_k &= P\{X_k \leq x_k | X_{k-1} = x_{k-1}, \ldots, X_1 = x_1\} = F_k(x_k|x_{k-1}, \ldots, x_1). \end{aligned}$$

Then, the random vector $\mathbf{Z} = T(\mathbf{X})$ is uniformly distributed on the k-dimensional hypercube, i.e. $Z_1, \ldots, Z_k$ are uniformly and independently distributed on $[0, 1]$.

Remark 4.10 (Inverse Rosenblatt transformation (Rosenblatt, 1952))**.** *The inverse operation*

$$X_1 = F_1^{-1}(Z_1), \quad X_2 = F_2^{-1}(Z_2|Z_1), \quad \ldots, \quad X_k = F_k^{-1}(Z_k|Z_{k-1}\ldots., Z_1)$$

can be used to simulate from a multivariate distribution. For any distribution F, if $\mathbf{Z}$ is a vector of independent random variables, $\mathbf{X} = T^{-1}(\mathbf{Z})$ has a distribution F.

The second transformation is the transformation from the copula scale (u-scale) to the z-scale, which we can use together with kernel density estimation. The reason for using it is that in spite of the kernel density estimation being a well-established non-parametric tool, it can face bias and consistency issues at the boundaries of the support, as mentioned for instance in Nagler et al. (2017). Therefore, we use a transformation trick. Assume we have a random variable on a u-scale $U \sim F_U$ and transform it to the z-scale: $Z = \Phi^{-1}(U)$, $(U = \Phi(Z))$. Since the function Φ^{-1} is increasing, the distribution of $Z = \Phi^{-1}(U)$ is given by

$$F_Z(z) = P(Z \leq z) = P(\Phi^{-1}(U) \leq z) = P(U \leq \Phi(z)) = F_U(\Phi(z))$$

and the density by

$$f_Z(z) = F'_Z(z) = \frac{d}{dz} F_U(\Phi(z)) = f_U(\Phi(z)) \frac{d}{dz} \Phi(z) = f_U(\Phi(z))\phi(z),$$

which implies

$$\frac{f_Z(z)}{\phi(z)} = f_U(\Phi(z)).$$

Here $\Phi(.)$ and $\phi(.)$ are the distribution and density functions of a $\mathcal{N}(0,1)$ variable. So it holds that

$$\frac{f_Z(\Phi^{-1}(u))}{\phi(\Phi^{-1}(u))} = f_U(u). \tag{4.2}$$

Consider a random sample $u^{(i)}$, $i = 1, \ldots, n$, whose density is to be estimated. Then the transformed $z^{(i)} = \Phi^{-1}(u^{(i)})$ is supported on the full $\mathbb{R}$. In this domain, kernel density estimators do not suffer from any boundary problems. Therefore we use the kernel density estimate $\hat{f_Z}$ based on $z_1, \ldots, z_n$ and scale it back according to Eq. (4.2) to get the density estimate $\hat{f_U}$. As a result we have

$$\hat{f_U}(u) := \frac{\hat{f_Z}(\Phi^{-1}(u))}{\phi(\Phi^{-1}(u))}. \tag{4.3}$$

In the bivariate case, it is known, that for $\mathbf{Z} = T(\mathbf{U})$, i.e. $Z_j = T_j(U_1, U_2)$, $j = 1, 2$, it holds that

$$f_{\mathbf{U}}(u_1, u_2) = f_{\mathbf{Z}}(T_1(u_1, u_2), T_2(u_1, u_2)) \, |det(dT)|,$$

where

$$dT = \begin{bmatrix} \frac{\partial T_1}{\partial u_1} & \frac{\partial T_1}{\partial u_2} \\ \frac{\partial T_2}{\partial u_1} & \frac{\partial T_2}{\partial u_2} \end{bmatrix}.$$

So, in our case $Z_j = \Phi^{-1}(U_j)$, $j = 1, 2$, we have

$$f_{\mathbf{U}}(u_1, u_2) = f_{\mathbf{Z}}(\Phi^{-1}(u_1), \Phi^{-1}(u_2))\, |det(dT)|,$$

with

$$dT = \begin{bmatrix} \frac{\partial \Phi^{-1}(u_1)}{\partial u_1} & \frac{\partial \Phi^{-1}(u_1)}{\partial u_2} \\ \frac{\partial \Phi^{-1}(u_2)}{\partial u_1} & \frac{\partial \Phi^{-1}(u_2)}{\partial u_2} \end{bmatrix} = \begin{bmatrix} \frac{1}{\phi(\Phi^{-1}(u_1))} & 0 \\ 0 & \frac{1}{\phi(\Phi^{-1}(u_2))} \end{bmatrix},$$

and

$$|det(dT)| = \left| \frac{1}{\phi(\Phi^{-1}(u_1))\phi(\Phi^{-1}(u_2))} \right|.$$

The resulting equation is therefore

$$f_{\mathbf{U}}(u_1, u_2) := \frac{f_{\mathbf{Z}}(\Phi^{-1}(u_1), \Phi^{-1}(u_2))}{|\phi(\Phi^{-1}(u_1))\phi(\Phi^{-1}(u_2))|}. \tag{4.4}$$

Since the $\phi(\cdot)$ function returns positive values we can omit the absolute value. Consider now a random sample $(u_1^{(i)}, u_2^{(i)})$, $i = 1, \ldots, n$, whose joint density is to be estimated. The transformed $(z_1^{(i)}, z_2^{(i)}) = (\Phi^{-1}(u_1^{(i)}), \Phi^{-1}(u_2^{(i)}))$ are supported on the full $\mathbb{R}^2$. We again use the kernel density estimate $\hat{f}_{\mathbf{Z}}$ based on $(z_1^{(i)}, z_2^{(i)})$, $i = 1, \ldots, n$ and scale it back according to Eq. (4.4) to get the density estimate $\hat{f}_{\mathbf{U}}$. As a result we have

$$\hat{f}_{\mathbf{U}}(u_1, u_2) := \frac{\hat{f}_{\mathbf{Z}}(\Phi^{-1}(u_1), \Phi^{-1}(u_2))}{\phi(\Phi^{-1}(u_1))\phi(\Phi^{-1}(u_2))}. \tag{4.5}$$

Note here that our $\mathbf{u}$ does not have a bivariate uniform distribution, therefore this $\mathbf{z}$ is not normally distributed. Nevertheless, we call it u-scale and z-scale, since $\mathbf{u} \in [0, 1]^2$ and $\mathbf{z} \in [-\infty, \infty]^2$, respectively. The same holds for the univariate case.

5. Developed STAN Program

Now, having all the required theory for this chapter presented, we can describe our implementation. First we noted that HMC only requires to know the desired limiting density up to a normalizing constant. In Eq. (2.6) we noted further that every conditional density is proportional to the joint density, i.e., we can use the joint copula density $c(\mathbf{u})$ for fixed values $\mathbf{u}_{\mathcal{C}_2}$ to sample from the density of $(\mathbf{U}_{\mathcal{C}_1}|\mathbf{U}_{\mathcal{C}_2} = \mathbf{u}_{\mathcal{C}_2})$ using `Stan` (Stan Development Team, 2012). Since the No-U-Turn sampler, which is implemented in `Stan`, is the extension of the Hamiltonian Monte Carlo algorithm, it possesses the same property of needing the density just up to the normalizing constant.

For this we used and adapted a prototype program written by M.Sc. Ariane Hanebeck. It is a mixture of R- and Stan-code. Since `Stan` (Stan Development Team, 2012) is a C++ library, the `R` package `rstan` provides RStan (Stan Development Team, 2020), the R interface to Stan, which enables one to write Stan code within RStudio, fit Stan models, sample and access the outputs conveniently from `R` (R Core Team, 2019). For further details about how to work with `Stan` and operate its interfaces see the `Stan`'s website `www.mc-stan.org`.

The following approach for using `Stan` via `rstan` is used. We represent the statistical model by writing its log density (up to an normalizing constant) using the Stan language in a separate file with a .stan extension. After that, the file with the Stan model is loaded into R as an instance of stanmodel class which can be passed to the `rstan` function `sampling`. Our sampler combines all these steps and the code is inspired and uses functions and items from `R` package `VineCopula` (Nagler et al., 2021).

In the following sections we present the documentation of the proposed sampler, together with its structure and detailed description of the R- and Stan- part of the code.

5.1. Documentation

Description

This function simulates from a given conditional vine-copula distribution using an extension of the Hamiltonian Monte Carlo algorithm - the No-U-Turn sampler.

Usage

```
sample_from_conditional(N, RVM, indexcon, ucon, burnin, thin, ...)
```

Arguments

`N`	Desired final number of simulated observations.
`RVM`	An RVineMatrix() object containing the information of the R-vine copula model.
`indexcon`	Vector $(1, \ldots, d)$, where the indices from the conditioned set $\mathcal{C}_1$ are set to FALSE (the ones desired to be sampled).
`ucon`	Vector of conditioning values. For the variables with the indices in the conditioning set $\mathcal{C}_2$, the values are set. For the variables with the indices in the conditioned set $\mathcal{C}_1$, FALSE is set.
`burnin`	The number of first samples/iterations that are discarded (not included in N). The default value is 1000.
`thin`	A positive integer specifying the period for saving samples. The default value is 10, since with this value the sampler is correctly sampling, which is also proved in the Simulation Study chapter. The argument N is the final number, so there is no need to adjust for thinning.
`...`	Possibility to set seed.

Value

The sampler returns an object of class stanfit, containing the simulated data from the given conditional vine-copula distribution and summary statistics (e.g. Rhat or effective sample size). Both can be extracted using the stanfit class's methods.

Details

We have a d-dimensional vine-copula distribution characterized by an R-vine structure and corresponding pair copula families and parameters and would like to sample from $(\mathbf{U}_{\mathcal{C}_1}|\mathbf{U}_{\mathcal{C}_2} = \mathbf{u}_{\mathcal{C}_2})$, where $\mathcal{C}_1$ and $\mathcal{C}_2$ are subsets of $\mathcal{C} = \{1, \ldots, d\}$ with $\mathcal{C}_1 \cap \mathcal{C}_2 = \emptyset$ and $\mathcal{C}_1 \cup \mathcal{C}_2 = \mathcal{C}$. $\mathcal{C}_1$ is the conditioned set of indices and $\mathcal{C}_2$ the conditioning set. First we define the desired R-vine with the help of the function `RVineMatrix` that belongs to the `VineCopula` package (Nagler et al., 2021) by combining R-vine matrix, family and parameter matrices into an RVineMatrix object. Second, the needed inputs indexcon and ucon are set. Indexcon represents a vector of indices in which the indices belonging to the conditioned set are replaced by FALSE, those are the indices of variables we would like to sample. On the other hand, ucon represents a vector of values for the conditioning variables in the conditioning set and similarly the indices from conditioned set are specified by FALSE, see *Example*. The returned object of the sampler is an object of class stanfit. By using the `rstan` function `extract`, we are able to obtain the samples.

References

Nagler et al. (2021).

Example

As an example consider a 5-dimensional vine-copula distribution and we would like to sample from $(U_2, U_4|U_1 = 0.7, U_3 = 0.2, U_5 = 0.35)$. The conditioned set of indices is $\{2, 4\}$, therefore replaced by FALSE in indexcon and ucon. For the conditioning set $\{1, 3, 5\}$, the values for corresponding conditioning variables are set in ucon. We specify all inputs as following and sample using the function. In Fig. 5.1 we can see the trees corresponding to the following R-vine structure as well as the scatter plot of the sampled u_2 and u_4 values.

```
#R-vine tree structure matrix
Matrix = c(5, 2, 3, 1, 4,
           0, 2, 3, 4, 1,
           0, 0, 3, 4, 1,
           0, 0, 0, 4, 1,
```

```
          0, 0, 0, 0, 1)
Matrix = matrix(Matrix, 5, 5)

#R-vine pair-copula family matrix
family = c(0, 10, 38, 4, 13,
           0, 0, 16, 7, 114,
           0, 0, 0, 4, 40,
           0, 0, 0, 0, 124,
           0, 0, 0, 0, 0)
family = matrix(family, 5, 5)

#R-vine pair-copula parameter matrices
par = c(0, 1.2, -6, 1.5, 2,
        0, 0, 1.1, 1.6, 1.9,
        0, 0, 0, 1.9, -5,
        0, 0, 0, 0, -4.8,
        0, 0, 0, 0, 0)
par = matrix(par, 5, 5)

par2 = c(0, 1, -8, 1.5, 3.9,
         0, 0, 1.1, 1.6, 0.9,
         0, 0, 0, 1.9, -0.5,
         0, 0, 0, 0, 0.8,
         0, 0, 0, 0, 0)
par2 = matrix(par2, 5, 5)

#RVineMatrix object
RVM = RVineMatrix(Matrix = Matrix, family = family, par = par,
                  par2 = par2, names = c("U1", "U2", "U3", "U4", "U5"))

#Conditioning variables and values
indexcon = c(1,FALSE,3,FALSE,5)
ucon = c(0.7,FALSE,0.2,FALSE,0.35)
```

```
#Simulate a 2-dimensional sample of size 200 from the conditional
vine-copula distribution
simdata = sample_from_conditional(2000, RVM, indexcon, ucon, burnin=1000,
                                  thin = 10, seed = 555)

#Extract samples for (U2, U4), U2 samples as the first dimension, and U4
the second
u2samples = extract(simdata,permute=FALSE)[,1,1]
u4samples = extract(simdata,permute=FALSE)[,1,2]
```

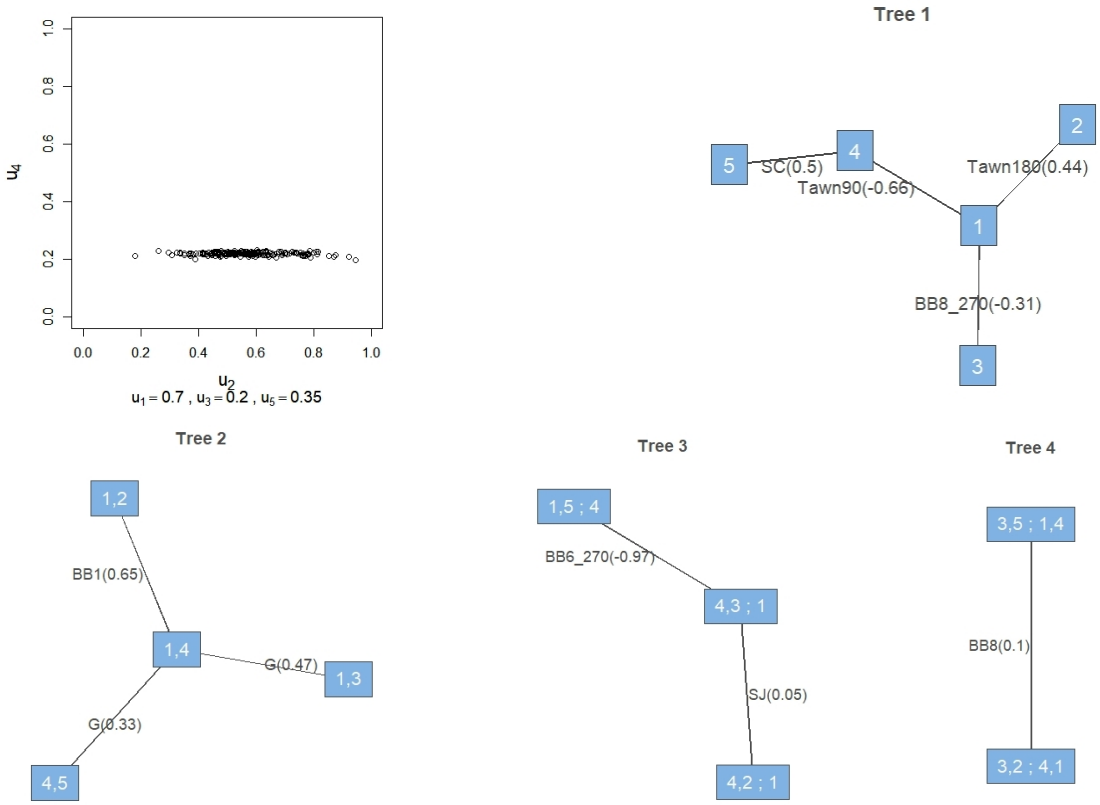

Figure 5.1.: The upper left panel shows the scatter plot of $(u_2, u_4|u_1 = 0.7, u_3 = 0.2, u_5 = 0.35)$ samples, with sample size 200. The remaining panels show the vine trees and pair copula families of the corresponding 5-dimensional R-vine structure.

5.2. Structure

The HMC algorithm together with the No-U-Turn sampler need, in order to sample, the target density to be expressed in form of the log-likelihood function. The log-likelihood is expressed as

$$\ell := \ln\left(\prod_{i=1}^{N} c(\mathbf{u}_i)\right),$$

for $\mathbf{u}_1, \ldots, \mathbf{u}_N$ independent samples from $c(\cdot)$. However, in our program, we do not have any data samples, so here $N = 1$. Our data is, for instance, in the form $\mathbf{u} = (0.7, u_2, 0.2, u_4, 0.35)$ and `Stan` samples from u_2 and u_4, given that $u_1 = 0.7$, $u_3 = 0.2$ and $u_5 = 0.35$. Therefore, we just have the logarithm of its vine density as

$$\begin{aligned}\ell := \ln\left(c(\mathbf{u})\right) &= \ln\left(\prod_{i=1}^{d-1}\prod_{e\in E_i} c_{a(e)b(e);D(e)}\left(C_{a(e)|D(e)}(u_{a(e)}|\mathbf{u}_{D(e)}), C_{b(e)|D(e)}(u_{b(e)}|\mathbf{u}_{D(e)})\right)\right) \\ &= \sum_{i=1}^{d-1}\sum_{e\in E_i} \ln\left(c_{a(e)b(e);D(e)}\left(C_{a(e)|D(e)}(u_{a(e)}|\mathbf{u}_{D(e)}), C_{b(e)|D(e)}(u_{b(e)}|\mathbf{u}_{D(e)})\right)\right).\end{aligned}$$

This log-density can be calculated like in the function `RVineLogLik` from the package `VineCopula` (Nagler et al., 2021). This function is composed of two parts. First, the arguments are preprocessed and all necessary parameters are extracted from the RVineMatrix object. Then, these parameters are given to the function `VineLogLikRvine2`, which computes the log-likelihood. For detailed code, see the source of the two functions, for example, in the GitHub repository of the `VineCopula` package, which can be found on `https://github.com/tnagler/VineCopula`. We also follow this two-step approach, meaning the parameters that are given to `Stan` are extracted and processed in `R`, and then the log-likelihood is calculated in `Stan` in the form of the function `VineLogLikRvine2`. In the STAN file, this `VineLogLikRvine2` function together with other important functions needed to compute the log-likelihood are copied from the `VineCopula` package (Nagler et al., 2021) and adjusted to the `Stan` language and rules.

The STAN file is composed of 4 blocks, namely functions, data, parameters and model. In the function-definition block, all own functions can be defined. In our case, it consists of all important functions needed to compute the log-likelihood as well as the function `VineLogLikRvine2` itself, which computes it. The data block is used for declaration of variables for the model, that are read in as data. In our program, this data, such as pair copula families and parameters, dimensions and many more, are

sent to the `Stan` from `R`. In the parameters block, the declared variables are the ones being sampled by `Stan`'s sampler NUTS, and thus $\mathbf{U}_{\mathcal{C}_1}$. And finally, the model block is where the log probability function is defined.

The R file consists of needed libraries/packages, necessary pre-processing functions and the main function `sample_from_conditional`. In the function the different parameters/inputs are processed and modified in order to be inserted as data to the STAN. Likewise, the parameters for STAN are specified and finally the sampling from `Stan` is called.

5.3. R-part

In this section we take a closer look on the R-part of the code. At the beginning, the required packages need to be loaded, namely `rstan` (Stan Development Team, 2020) and `VineCopula` (Nagler et al., 2021).

Next, pre-processing functions need to be defined, from which we mention, for example, `normalizeRVineMatrix`. Since our sampler is based on the functions from `VineCopula` package and RVineMatrix needs to have a specific shape, we need to flip the matrix similarly as in the `VineCopula` package and we do it by using this function, which can be found, for instance, in the GitHub repository of the `VineCopula` package (`https://github.com/tnagler/VineCopula`). The R-vine matrix is permuted to achieve a natural ordering (i.e. `diag(RVM$Matrix) == d:1`)

The main part is the `sample_from_conditional` function, which is called by the user and returns the samples. The function is shown in Listing 5.1 and we now describe its parts.

First, we set default values for burn-in and thin. We use thin=10, since this thinning value is used and tested in simulation study. We extract the length of the $\mathbf{u}$ vector and the dimensions of conditioned set $\mathcal{C}_1$ (the ones we want to sample) and conditioning set $\mathcal{C}_2$ into d1 and d2, respectively. Then, our STAN file is loaded as an instance of stanmodel class. So far, we have two almost identical files, one for sampling from one-dimensional conditional distributions and one for multi-dimensional. The main difference lies in the parameters block, where it is not possible to distinguish between cases by the `if` function, and it is important for `Stan` whether the parameters of interest is a vector or a scalar. We talk about this more in the STAN part. And last but not least, the R-vine matrix is normalized if necessary, and whether the normalization is done is

saved in the *dataflip* variable.

Second, the data and parameters for STAN are defined, and `Stan` is called to sample. All important inputs such as pair copula families, parameters, structure and etc. are extracted from RVineMatrix object and processed. Besides that, additional matrices required internally for evaluating the density and etc., such as `MaxMat`, `CondDistr&direct` and `CondDistr&indirect` are prepared in the same way as in the `VineCopula` package. All of these inputs, together with indexcon and ucon vectors are inserted into STAN as the data list. We will see later that this data list matches with the variable declaration in data block in the STAN file. For the conditioned variables $\mathbf{U}_{\mathcal{C}_1}$, which we would like to sample and are called *ucalculate*, the initial values are specified. Furthermore, parameters for sampling are determined. These include number of iterations, burn-in, chains, cores and the control parameters of the sampler's behaviour. Having all of this done, we can insert everything into STAN and sample by calling the `sampling` method. The function returns an instance of stanfit class, from which the samples can be extracted.

Listing 5.1: Code for **sample_from_conditional** function.

```
sample_from_conditional=function(N,RVM,indexcon,ucon,burnin,thin,...)
{
  #default values for burnin and thin
  if(missing(burnin)){
     burnin = 1000
  }
  if(missing(thin)){
     thin = 10
  }

  #extracting the dimensions
  d=length(indexcon)
  d1=length(which(indexcon==FALSE))
  d2=d-d1

  #loading STAN
  if (d1==1){
```

```
    STAN = stan_model(file='STAN.stan')
}else{
    STAN = stan_model(file='STAN2.stan')
}

#normalizing RVineMatrix - needed in VineCopula package
dataflip=0
o <- diag(RVM$Matrix)
if (any(o != length(o):1)) {
    oldRVM <- RVM
    RVM <- normalizeRVineMatrix(RVM)
    dataflip=1
}

##############
##From the VineCopula package: the different parameters/inputs we
##need to compute log-lik in STAN - inserted into STAN as data
##############

T=1
w1 <- as.vector(RVM$family)
w1[is.na(w1)] <- 0
th <- as.vector(RVM$par)
th[is.na(th)] <- 0
th2 <- as.vector(RVM$par2)
th2[is.na(th2)] <- 0
condirect <- as.vector(as.numeric(RVM$CondDistr$direct))
conindirect <- as.vector(as.numeric(RVM$CondDistr$indirect))
maxmat <- as.vector(RVM$MaxMat)
matri <- as.vector(RVM$Matrix)
matri[is.na(matri)] <- 0
maxmat[is.na(maxmat)] <- 0
condirect[is.na(condirect)] <- 0
conindirect[is.na(conindirect)] <- 0
```

```
#insert all inputs into the list of data
data_stan_u_2=list(T=T, dataflip=as.integer(dataflip), d=d,
                   o=as.integer(o), d1=d1, d2=d2, family=as.integer(w1),
                   maxmat=as.integer(maxmat),matri=as.integer(matri),
                   condirect=as.integer(condirect),
                   conindirect=as.integer(conindirect),
                   par=as.double(th),par2=as.double(th2),
                   indexcon=indexcon, ucon=ucon)

#initial values for the conditioned variables
init_list_u_2=list(list(ucalculate=rep(0.5,d1)))

##############
## The parameters for the STAN-program
##############

It=N*thin
iter_u_2=It+burnin
burnin_u_2=burnin
chains_u_2=1
cores_u_2=1
adapt_delta_u_2=0.8 #0.85
max_treedepth_u_2=10 #15

##############
##Sample from STAN
##############

fit_u_2=sampling(STAN,iter=iter_u_2,warmup=burnin_u_2,chains=chains_u_2,
                 cores=cores_u_2,data=data_stan_u_2,init=init_list_u_2,
                 control=list(adapt_delta=adapt_delta_u_2,
                              max_treedepth=max_treedepth_u_2),
                 thin=thin, ...)
```

```
  return(fit_u_2)
}
```

5.4. Stan-part

As presented in the structure section, the STAN file consists of 4 blocks (functions, data, parameters and model) and we describe them more in this section. In Listing 5.2 we see the overall structure of our STAN file.

Listing 5.2: Structure of a general `Stan` file (Stan Development Team, 2012).

```
functions {
  // ... function declarations and definitions ...
}
data {
  // ... declarations ...
}
parameters {
   // ... declarations ...
}
model {
   // ... declarations ... statements ...
}
```

Function-definition Block

All important functions needed to compute the log-likelihood are placed here. The summary and description is presented in Table 5.1. The functions are copied from the `VineCopula` package and adjusted to the `Stan` language. For their source code see GitHub repository of the package, as mentioned before. As a reminder, the log-likelihood is expressed as

$$\ell := \sum_{i=1}^{d-1} \sum_{e \in E_i} \ln \left(c_{a(e)b(e);D(e)} \left(C_{a(e)|D(e)}(u_{a(e)}|\mathbf{u}_{D(e)}), C_{b(e)|D(e)}(u_{b(e)}|\mathbf{u}_{D(e)}) \right) \right) \tag{5.1}$$

Function	Description
`LL()`	Function to compute log-likelihood for bivariate copulas. For every type of copula, independent, Gaussian, Clayton etc., there is a definition how to compute its log-likelihood. Two cases are treated, namely 0° and 180° rotations.
`LL_mod2()`	Extension of the `LL()` function. In the case of 0° and 180° rotations, it calls `LL()` directly. On the other hand, if 90° and 270° rotations of copulas are needed, it changes the arguments and parameters accordingly and calls `LL()`.
`Hfunc()`	Function to compute h-function of corresponding copula (similar approach as `LL()` function). Again two cases are treated, namely 0° and 180° rotations. Since the h-functions are not symmetric for some copulas, both cases need to be implemented. In `Hfunc1()`, we condition on the second variable, i.e. $h(u_1\|u_2)$, and in `Hfunc2()` we condition on the first one, i.e. $h(u_2\|u_1)$.
`Hfunc1()`	$h(u_1\|u_2)$ - In the case of 0° and 180° rotations, it calls `Hfunc()` directly. On the other hand, if 90° and 270° rotations of copulas are needed, it changes the arguments and parameters accordingly and calls `Hfunc()`. All rotations of Tawn copulas are treated here, without using `Hfunc()`.
`Hfunc2()`	$h(u_2\|u_1)$ - Same as `Hfunc1()` with interchanged arguments.
`VineLogLikRvine2()`	Main function of the file. It takes as input all necessary data for computing log-density. The log-density is calculated by summing up all individual parts, namely the log-likelihoods of copulas as in Eq. (5.1). The copulas are stacked with corresponding h-functions. It calls `LL_mod2()` as well as `Hfunc1()` and `Hfunc2()` functions. It returns the final log-density.
helping functions	Various different functions that help to compute the log-likelihood for bivariate copulas and h-functions. They are used in `LL()` or `Hfunc()`, mainly for Tawn and BB copulas.

Table 5.1.: Summary and description of the different functions defined in the function-definition block.

So far, Student's t copula does not work in our sampler, i.e. is not implemented in the functions above. The reason for it is that t-copula is computed using quantile function, which is not defined in `Stan`. However, we are now in process of implementing solutions to this issue.

Data Block

In this block variables needed for the model are declared. These are the ones that are processed in `R` and forwarded into STAN as data. The data is then used by the function computing log-likelihood in the model block. The code for the data block is presented in Listing 5.3. The individual variables are explained in Table 5.2.

Listing 5.3: Code for the **data block**.

```
data{
  int T; //T=1
  int dataflip;
  int d;
  int o[d];
  int d1;
  int d2;
  int family[d*d];
  int maxmat[d*d];
  int matri[d*d];
  int condirect[d*d];
  int conindirect[d*d];
  vector[d*d] par;
  vector[d*d] par2;
  vector[d] indexcon;
  vector<lower=0, upper=1>[d] ucon;
  }
```

Variable	Description
T	Sample size in the log-likelihood formula. Since we have only log of the density and no samples, T=1.
dataflip	Value 1 or 0 meaning whether the R-vine matrix is normalized (1) or not (0)
d	dimension of the vine model, d=d1+d2
o	the original ordering of the R-vine matrix diagonal - before normalizing
d1	cardinality of conditioned set
d2	cardinality of the conditioning set
family	copulas' families
maxmat	internal helping matrix
matri	R-vine structure (R-vine matrix)
condirect	internal helping matrix
conindirect	internal helping matrix
par	copulas' parameters
par2	copulas' parameters 2
indexcon	vector of indices
ucon	vector of conditional values (values are constrained on [0,1])

Table 5.2.: Description of the variables declared in the data block.

Parameters Block

This block contains parameters of the model, the variables which are sampled by `Stan`. In our case, it is $\mathbf{u}_{\mathcal{C}_1}$ denoted in the code as *ucalculate*. In Listing 5.4, the code for the parameters block is presented. As mentioned before, in the parameters block no decision-making statements are allowed. Therefore, we declare 1-dimensional variable (in STAN.stan file) as written in the first line of the code for parameters block. In the multi-dimensional case, we declare d1-dimensional variable (in STAN2.stan file) as written in the note of the code. Values of both variables are constrained on [0,1].

Listing 5.4: Code for the **parameters block**.

```
parameters{
    real<lower=0, upper=1> ucalculate;
    //vector<lower=0, upper=1>[d1] ucalculate;
}
```

Model Block

The last block is a model where we define the log probability function. The function is assigned to **target**, and in our case it is the output from `VineLogLikRvine2` function. The code for this block is shown in Listing 5.5.

Listing 5.5: `Stan` code of the **model block** for conditional sampling from a vine copula.

```
model{
  // Declaration of variables
  vector[d] daten;
  vector[d] datennew;
  vector[d] check;
  int k;

  // Searching for the index i in indexcon and filling the daten variable
  // with conditioning values
  for(i in 1:d)
  {
    for(j in 1:d)
    {
      if(indexcon[j]==i)
      {
        daten[i]=ucon[j];
        check[i]=1;
      }
    }
  }
```

```
  // Searching for the empty parts of the daten variable and filling with
  // conditioned variables / their initial values

  // In a multidimensional conditional density case, we need to use vector
  // of ucalculate as in the note below
  k=1;
  for (i in 1:d)
  {
    if(check[i]!=1)
    {
      daten[i]=ucalculate; // daten[i]=ucalculate[k]; - multidim. case
      k=k+1;
    }
    else{
      k=k;
    }
  }

  // If the R-vine matrix is permuted, we need to permute the data as well
  if(dataflip==1){
      for(i in 1:d){
          datennew[i]=daten[o[d+1-i]];
      }
  }
  else{
      datennew=daten;
  }

  // log-density computed in VineLogLikRvine2 as the target
  target += VineLogLikRvine2(T, d, family, maxmat, matri, condirect,
                             conindirect, par, par2, datennew);
}
```

5.5. Computational Time

To sample from `Stan` requires some time, which depends on the dimension of the entire vine model, on the dimension of the distribution we are sampling from, as well as on the number of samples we wish to get. In this thesis, we sample from univariate and bivariate conditional distributions having three, five and seven dimensional vine copula models. The desired sample size is 1000, 5000 and 10000. The summary of the time required for sampling is presented in Table 5.3. The sampling is done using RStudio at LRZ server (specification: RStudio Server 2022.02.1 with R 4.2.0). For each of the 18 situations we sample 100 times and present the mean of the times required for sampling. Note that when sampling we use thinning, with thin=10. So the real number of sampled observations is 10 times higher.

		Dimension of vine model		
		d=3	d=5	d=7
univariate conditional distribution	$n = 1000$	1.10 s	1.84 s	4.65 s
	$n = 5000$	4.22 s	8.20 s	21.14 s
	$n = 10000$	9.59 s	15.80 s s	41.97 s
bivariate conditional distribution	$n = 1000$	1.84 s	2.59 s	7.25 s
	$n = 5000$	7.76 s	11.87 s	33.32 s
	$n = 10000$	15.94 s	24.26 s	65.76 s

Table 5.3.: The time required to sample from STAN using the RStudio at LRZ server. Sampling is done from univariate and bivariate distributions with sample sizes 1000, 5000 and 10000 having 3, 5 and 7-dimensional vine models. The table presents the average computation time over 100 iterations. The burn-in used is 1000 and is not included in the sample size.

6. Simulation Study

In this chapter, we test if our proposed STAN program correctly samples from the desired density by investigating its performance in simulation setups.

Assume that the random variables $\mathbf{U} = (U_1, \dots U_d)^T$ follow regular vine copula distribution in d-dimensions. Recall the subsets $\mathcal{C}_1 = \{\mathcal{C}_{1,1}, \dots, \mathcal{C}_{1,k}\}$ and $\mathcal{C}_2 = \{\mathcal{C}_{2,1}, \dots \mathcal{C}_{2,\ell}\}$ of $\mathcal{C} = \{1, \dots, d\}$ with $\mathcal{C}_1 \cap \mathcal{C}_2 = \emptyset$ and $\mathcal{C}_1 \cup \mathcal{C}_2 = \mathcal{C}$. The proposed program samples from the distribution of $(\mathbf{U}_{\mathcal{C}_1} | \mathbf{U}_{\mathcal{C}_2} = \mathbf{u}_{\mathcal{C}_2})$ where $\mathbf{U}_{\mathcal{C}_1} = (U_{\mathcal{C}_{1,1}}, \dots, U_{\mathcal{C}_{1,k}})^T$ and $\mathbf{U}_{\mathcal{C}_2} = (U_{\mathcal{C}_{2,1}}, \dots, U_{\mathcal{C}_{2,\ell}})^T$.

We start with sampling from D-vine copula distributions. For $d = 3$ and $d = 5$, the D-vine tree structures look as in Fig. 6.1 and Fig. 6.2, respectively. Later we perform sampling from one specific R-vine, whose tree structure looks as in Fig. 6.3.

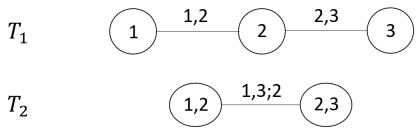

Figure 6.1.: 3-dimensional D-vine tree structure used in simulation study.

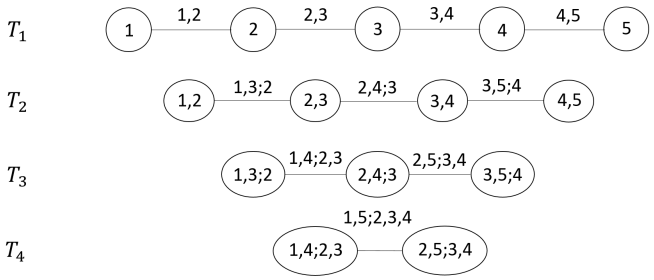

Figure 6.2.: 5-dimensional D-vine tree structure used in simulation study.

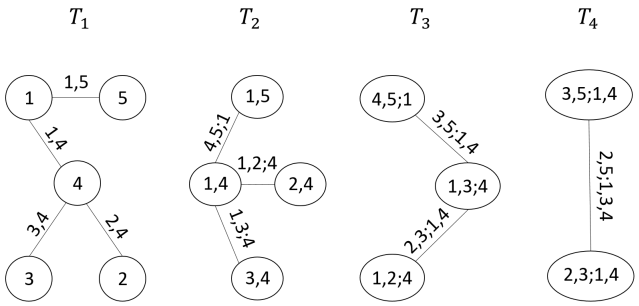

Figure 6.3.: 5-dimensional R-vine tree structure used in simulation study.

We divide this chapter into two sections. In the first one we present the sampling from univariate conditional distributions, i.e. $|\mathcal{C}_1| = k = 1$, $|\mathcal{C}_2| = \ell = d - 1$, and in the second one the bivariate conditional distributions, i.e. $|\mathcal{C}_1| = k = 2$ and $|\mathcal{C}_2| = \ell = d - 2$

6.1. Case I: Sampling from Univariate Cond. Distribution Functions Arising from a Vine Copula

Simulation Setup

Starting with the univariate conditional distributions, we choose the simulation setups such that we cover large number of possibilities. We distinguish between distributions, whose conditional density is easily expressed without integration and whose density cannot be expressed without integration as in Example 2.23. The dimension d is set here to 3 and 5, and we cover D-vine tree structure as well as one specific R-vine. The summary of the studied simulation setups is shown in Table 6.1. In every simulation setup we sample $n = 1000$, 5000 and 10000 samples with 3 different sets of conditioning values, namely their distance from a central vector $\mathbf{u_c} := (0.5, 0.5, \ldots, 0.5) \in \mathbb{R}^d$ being extremely low, medium and extremely large, resulting in 9 different scenarios summarized in Table 6.2. All computations are conducted using programming language R (R Core Team, 2019). In the following, we first introduce the approach and then present the results.

First we specify pair copula families and parameters for the vines associated with the specific vine tree structure.

Setup	Vine	d	Density availability	Conditioned, conditioning set
1	D	3	T	$\mathcal{C}_1 = \{1\}, \mathcal{C}_2 = \{2,3\}$
2	D	3	F	$\mathcal{C}_1 = \{2\}, \mathcal{C}_2 = \{1,3\}$
3	D	5	T	$\mathcal{C}_1 = \{1\}, \mathcal{C}_2 = \{2,3,4,5\}$
4	D	5	F	$\mathcal{C}_1 = \{2\}, \mathcal{C}_2 = \{1,3,4,5\}$
5	R	5	T	$\mathcal{C}_1 = \{5\}, \mathcal{C}_2 = \{1,2,3,4\}$
6	R	5	F	$\mathcal{C}_1 = \{2\}, \mathcal{C}_2 = \{1,3,4,5\}$

Table 6.1.: Selected simulation setups in sampling from univariate conditional distributions. Density availability means whether we can express the conditional density without integration, hence true or false. In conditioning and conditioned set we show the chosen conditioning and conditioned variables.

		Conditioning values		
		low	medium	large
Number of samples	$n = 1000$	9 conditioning value scenarios		
	$n = 5000$			
	$n = 10000$			

Table 6.2.: Choice of conditioning values for each simulation setup.

Choosing the Conditioning Values

For the selection of the conditioning values for all simulations we proceed as follows. Using the `rvinecopulib` package (Nagler and Vatter, 2021), we sample $\mathbf{u}_r := (u_{r1}, u_{r2}, \ldots, u_{rd})^T$, $r = 1, \ldots, R$, where $R = 1000$ from the specified vine copula distribution. We compute the Euclidean distance to the central vector $\mathbf{u_c}$, $e(\mathbf{u}_r) := ||(\mathbf{u}_r - \mathbf{u_c})||_2^2$ for each copula observation vector, so we get $\mathbf{e} = (e(\mathbf{u}_1), \ldots, e(\mathbf{u}_R))^T \in \mathbb{R}^R$. Let $q_\alpha(\mathbf{e})$ be α-quantile of $\mathbf{e}$, we find the sample iteration r_α such that $\mathbf{u}_{r_\alpha}$ has $e(\mathbf{u}_{r_\alpha}) = q_\alpha(\mathbf{e})$. In other words, we want to find the iteration number r_α such that the Euclidean distance of this observation from the central vector $\mathbf{u_c}$ is the α-quantile of the vector $\mathbf{e}$. In sampling from univariate conditional distribution, we choose $\alpha = 0.05, 0.5, 0.95$. In particular we call the conditioning values for $\alpha = 0.05$ **low**, the one for $\alpha = 0.5$ **medium** and $\alpha = 0.95$ **large**. We are now interested in sampling from $\mathbf{U}_{\mathcal{C}_1} | \mathbf{U}_{\mathcal{C}_2} = \mathbf{u}_{r_\alpha}^{\mathcal{C}_2}$, where $\mathbf{u}_{r_\alpha}^{\mathcal{C}_2} = (u_{r_\alpha, \mathcal{C}_{2,1}}, \ldots, u_{r_\alpha, \mathcal{C}_{2,\ell}})^T$.

Sampling

We can now proceed to sampling from the conditional distributions. Since the sample drawn from the developed STAN program is serially dependent we use thinning. That means, after obtaining the sampled values from the program we choose every x^{th} iteration as a valid sample and discard the rest. Our selected thinning is 10. Therefore we draw a sample of size $n = 10000, 50000, 100000$ for $\mathbf{U}_{\mathcal{C}_1}|\mathbf{U}_{\mathcal{C}_2} = \mathbf{u}_{r_\alpha}^{\mathcal{C}_2}$ and pick every 10^{th} iteration to obtain the samples of size $n = 1000, 5000, 10000$. We denote the sampled values by $\mathbf{u}_i(\mathbf{u}_{r_\alpha}^{\mathcal{C}_2})$, $i = 1, \ldots, n$.

Density Estimation

Further we examine first whether the samples are from the desired distribution visually and then we use statistical performance measures. In order to compare the true theoretical conditional density of $\mathbf{U}_{\mathcal{C}_1}|\mathbf{U}_{\mathcal{C}_2} = \mathbf{u}_{r_\alpha}^{\mathcal{C}_2}$ with an estimate based on the sample $\mathbf{u}_i(\mathbf{u}_{r_\alpha}^{\mathcal{C}_2})$, $i = 1, \ldots, n$, we use kernel density estimation. We can then plot the theoretical density together with the estimated densities for the samples with sizes $n = 1000, 5000, 10000$. For this we use the transformation trick, i.e. we transform the samples to the z-scale, estimate the density and scale it back as shown in Eq. (4.3).

Performance Measures

To measure the goodness-of-fit, we use the Kolmogorov-Smirnov test defined in Definition 4.6. In the univariate conditional case, we first apply the probability integral transform on the samples $u_i(\mathbf{u}_{r_\alpha}^{\mathcal{C}_2})$, $i = 1, \ldots, n$ and obtain transformed data $v_i(\mathbf{u}_{r_\alpha}^{\mathcal{C}_2}) := F(u_i(\mathbf{u}_{r_\alpha}^{\mathcal{C}_2})|\mathbf{U}_{\mathcal{C}_2} = \mathbf{u}_{r_\alpha}^{\mathcal{C}_2})$. We then test whether the transformed data $v_i(\mathbf{u}_{r_\alpha}^{\mathcal{C}_2})$ are uniformly distributed using Kolmogorov-Smirnov test and by examining the associated p-values.

6.2. Results - Case I

In this section we present sampling results from all 6 simulation setups shown in Table 6.1. To conduct the simulation study we perform the sampling for a specific distribution many times and check if in any event the samples are from the desired

distribution. The chosen number of simulations is $N = 100$ resulting in 100 goodness-of-fit tests.

Setup 1: D-vine, d=3, conditional density expressed without integration

In this setup we use the vine tree structure from Fig. 6.1. We are sampling from the distribution of $(U_1|U_2 = u_2, U_3 = u_3)$. The conditional density and distribution are

$$c_{1|23}(u_1|u_2, u_3) = \frac{c_{123}}{c_{23}} = \frac{c_{1,2}c_{2,3}c_{1,3;2}}{c_{2,3}} = c_{1,2}(u_1, u_2) \cdot c_{1,3;2}(C_{1|2}(u_1|u_2), C_{3|2}(u_3|u_2)),$$

$$C_{1|23}(u_1|u_2, u_3) = h_{1|3;2}(h_{1|2}(u_1|u_2) \mid h_{3|2}(u_3|u_2)).$$

The chosen pair copula families and parameters for this setup are shown in Fig. 6.4. Every row corresponds to a different example with four copula specifications being examined.

After performing 100 sampling simulations, we compare the true theoretical density with the estimated densities of the samples with sizes $n = 1000, 5000, 10000$. The comparison of densities for low conditioning values is shown in Fig. 6.5. The rows correspond to different specifications and columns to different iterations, i.e. we show 3 out of 100 iterations to compare the densities. Below each iteration plot, we present the values of conditioning variables. The comparison of densities for medium and large conditioning values are shown in Fig. 6.6 and Fig. 6.7, respectively.

For every sampling simulation we perform the goodness-of-fit test. The percentage of tests for uniform distributions that would be rejected by Kolmogorov-Smirnov test at significance level $\alpha = 5\%$, assuming low conditional values, is presented for every specification and sample size in the third column of Table 6.3. In the subsequent columns we present minimal and maximal value of various measures and that is the p-value of the Kolmogorov-Smirnov test, the effective sample size and $\hat{R}$. In the following we denote this kind of a table as a table of results. The table of results for medium and large conditioning values are presented in Table 6.4 and Table 6.5, respectively.

In Appendix A we present, in detail, further results and plots for the Specification 1 of this Simulation Setup.

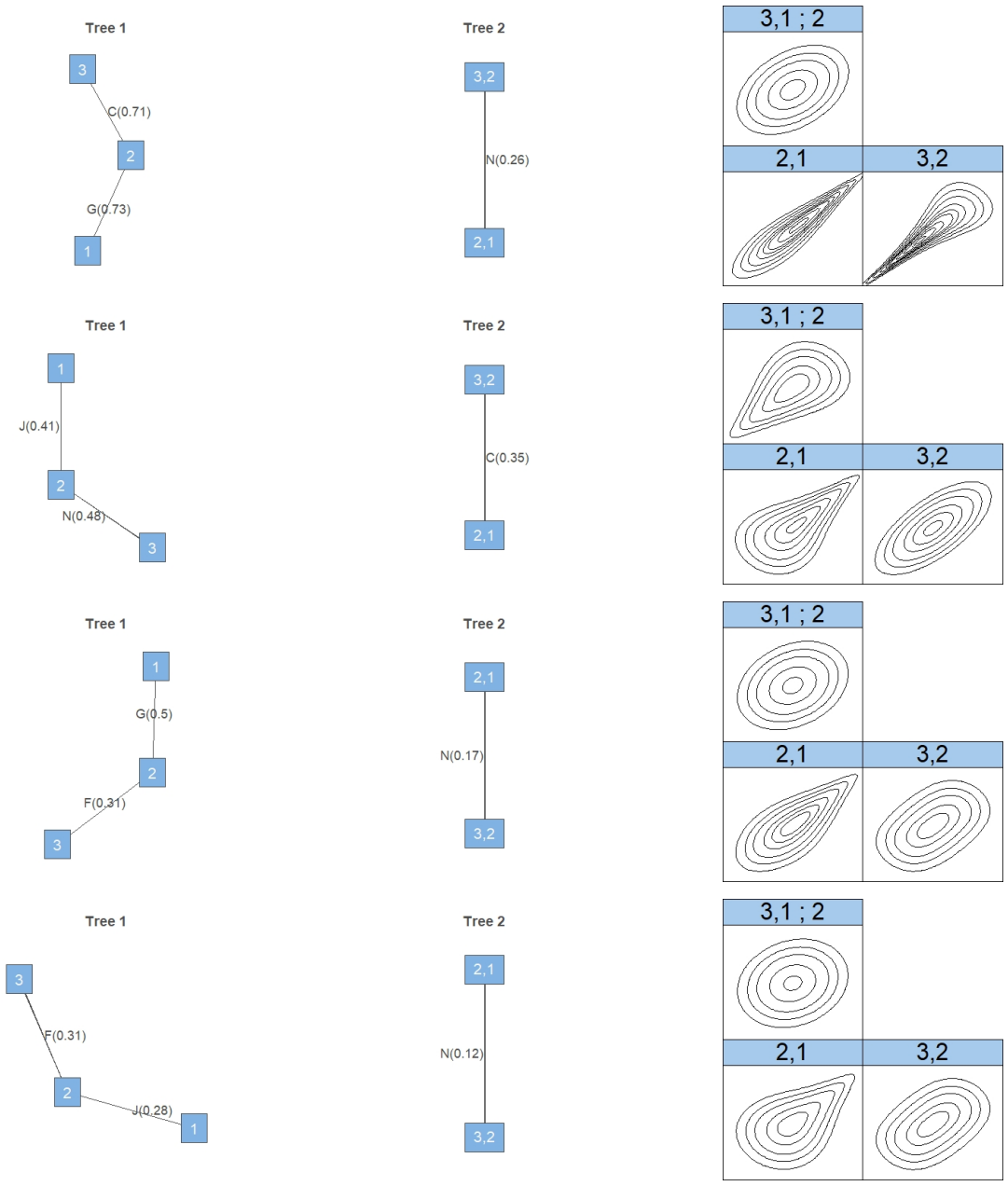

Figure 6.4.: Pair copula families and contour plots of the chosen distribution examples in the Simulation Setup 1. First two columns depict the copula families with corresponding Kendall's τ parameter and the third one the pair copula contour plots on the z-scale. Each row corresponds to a different D-vine copula specification.

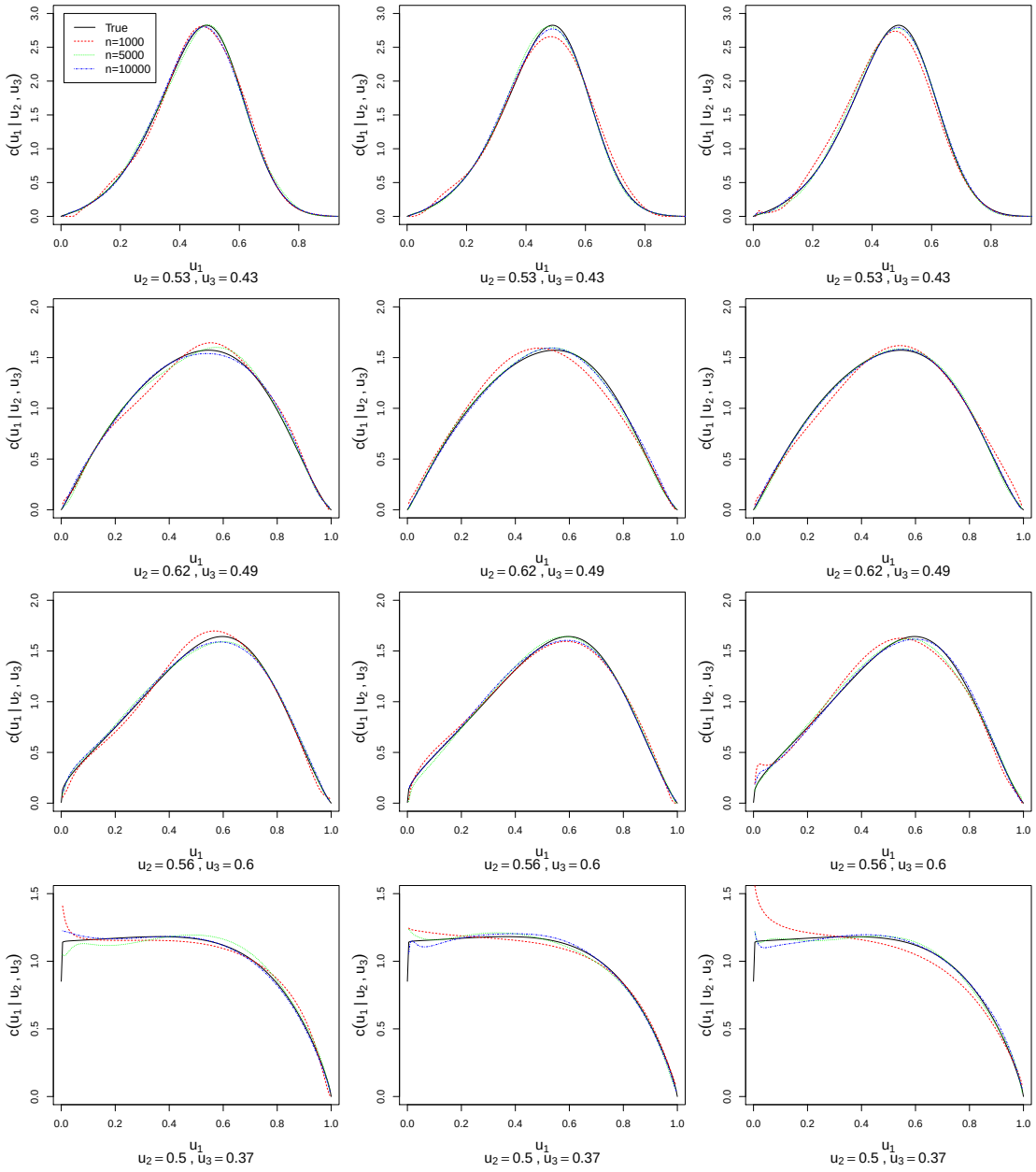

Figure 6.5.: Comparison of densities for Simulation Setup 1 with **low conditioning values**. Rows correspond to the 4 specifications shown in Fig. 6.4 and columns to 3 chosen simulation iterations out of 100. The true theoretical density is compared to kernel density estimates based on samples with sizes $n = 1000, 5000, 10000$ as seen in the legend in the upper left corner. Beneath each plot we show the values of conditioning variables.

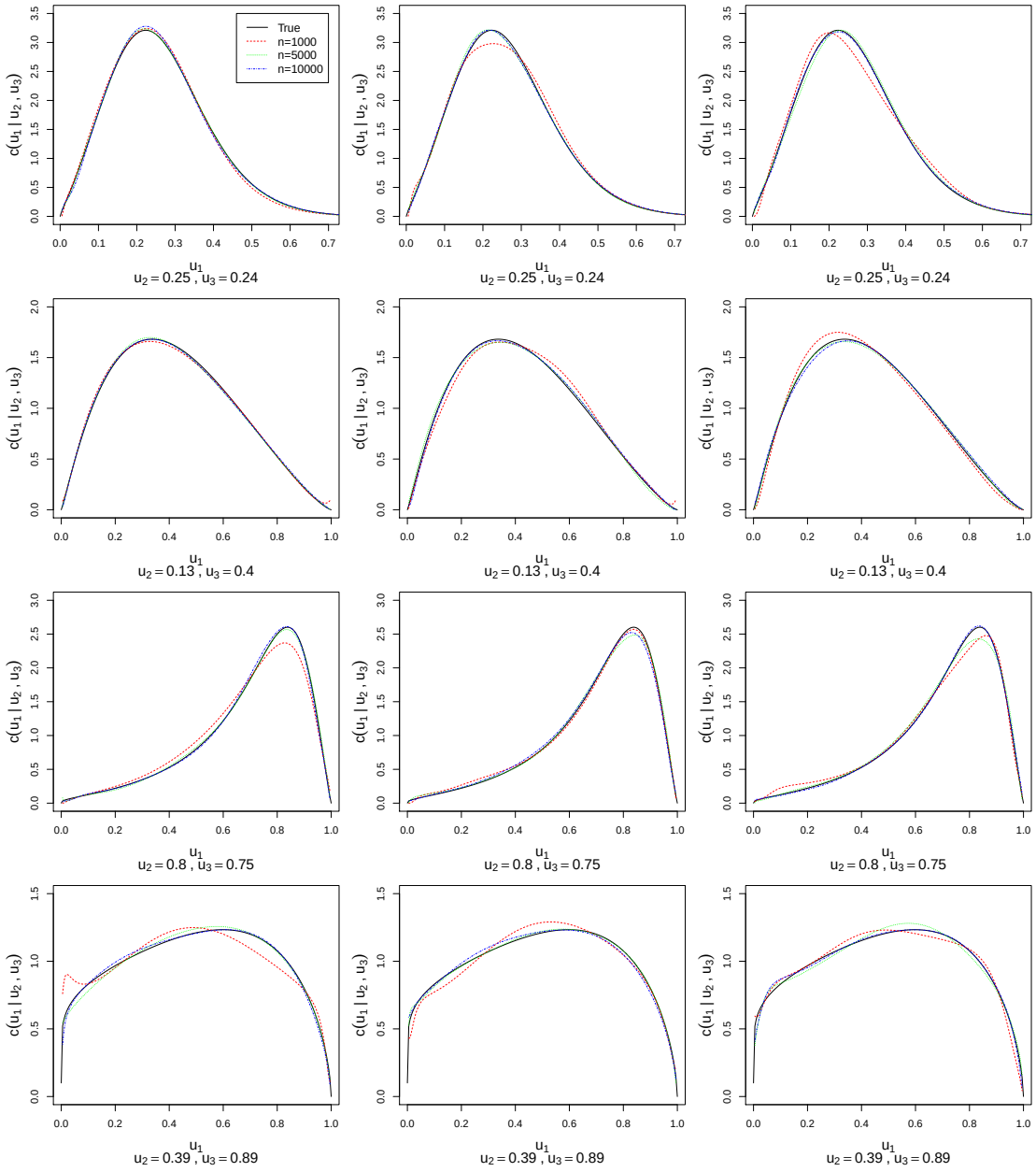

Figure 6.6.: Comparison of densities for Simulation Setup 1 with **medium conditioning values**. Rows correspond to the 4 specifications shown in Fig. 6.4 and columns to 3 chosen simulation iterations out of 100.

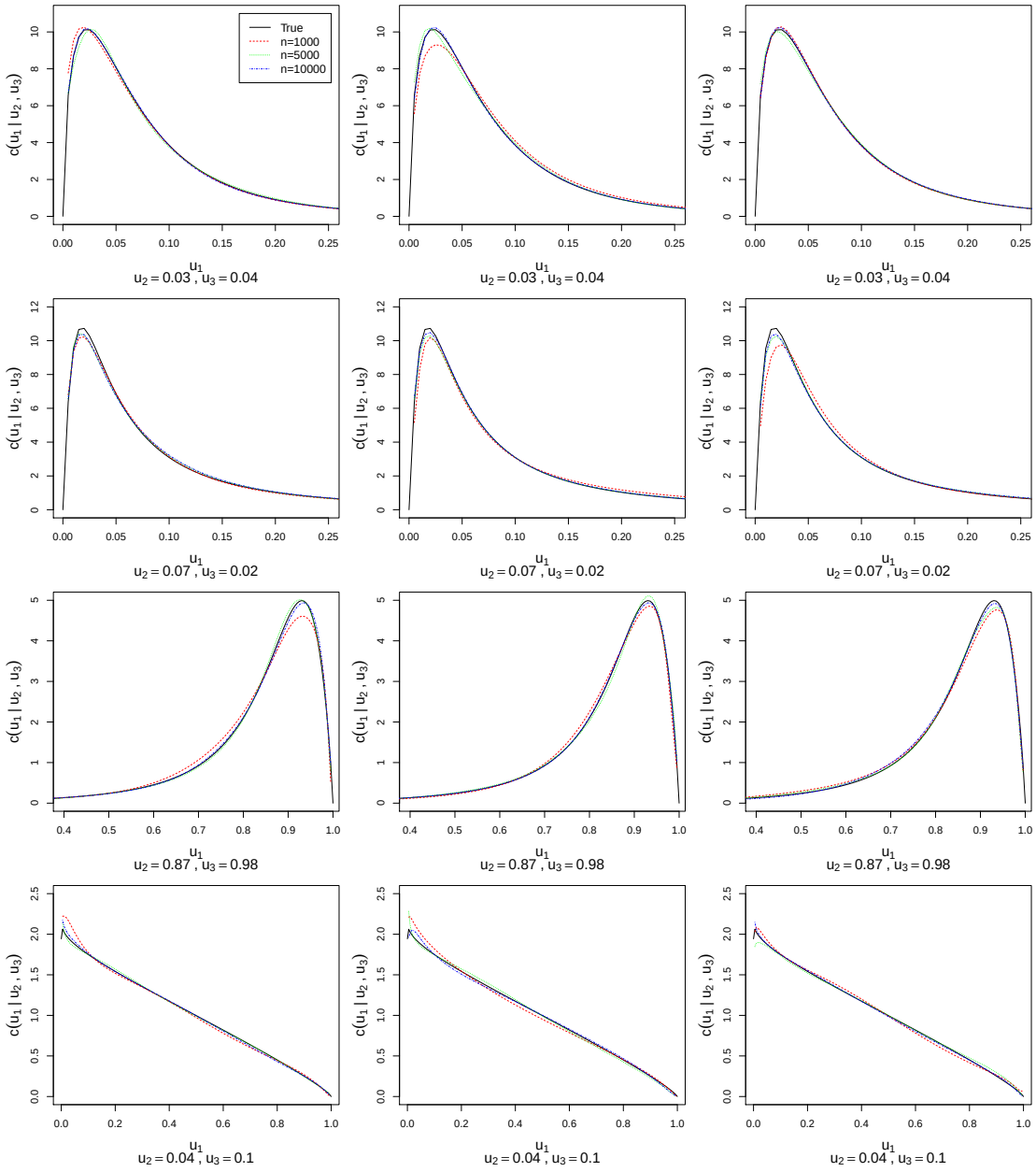

Figure 6.7.: Comparison of densities for Simulation Setup 1 with **large conditioning values**. Rows correspond to the 4 specifications shown in Fig. 6.4 and columns to 3 chosen simulation iterations out of 100.

		Rejected tests	p-value		Eff. s. size		R-hat	
			min	max	min	max	min	max
Specification 1	n=1000	4%	0.001	0.997	0.690	1.160	0.999	1.006
	n=5000	1%	0.049	0.997	0.844	1.083	1.000	1.002
	n=10000	5%	0.002	0.992	0.853	1.050	1.000	1.001
Specification 2	n=1000	9%	0.002	0.998	0.758	1.157	0.999	1.007
	n=5000	0%	0.068	0.997	0.845	1.106	1.000	1.001
	n=10000	7%	0.013	0.989	0.907	1.039	1.000	1.000
Specification 3	n=1000	6%	0.008	0.996	0.717	1.132	0.999	1.004
	n=5000	7%	0.001	0.998	0.834	1.072	1.000	1.001
	n=10000	3%	0.007	0.993	0.907	1.050	1.000	1.001
Specification 4	n=1000	2%	0.022	0.992	0.698	1.137	0.999	1.008
	n=5000	3%	0.019	0.996	0.800	1.084	1.000	1.002
	n=10000	2%	0.024	0.985	0.861	1.056	1.000	1.001

Table 6.3.: Table of results for Simulation Setup 1 with **low conditioning values**.

		Rejected tests	p-value		Eff. s. size		R-hat	
			min	max	min	max	min	max
Specification 1	n=1000	4%	0.008	0.986	0.688	1.127	0.999	1.005
	n=5000	3%	0.021	0.965	0.813	1.064	1.000	1.001
	n=10000	8%	0.000	0.993	0.888	1.048	1.000	1.000
Specification 2	n=1000	8%	0.000	1.000	0.605	1.181	0.999	1.003
	n=5000	3%	0.016	0.984	0.874	1.070	1.000	1.002
	n=10000	6%	0.014	0.990	0.902	1.069	1.000	1.001
Specification 3	n=1000	5%	0.029	0.988	0.749	1.221	0.999	1.008
	n=5000	4%	0.016	0.993	0.855	1.065	1.000	1.001
	n=10000	8%	0.011	0.997	0.853	1.054	1.000	1.001
Specification 4	n=1000	6%	0.011	0.991	0.666	1.176	0.999	1.004
	n=5000	3%	0.004	0.998	0.782	1.074	1.000	1.002
	n=10000	5%	0.017	0.947	0.900	1.028	1.000	1.000

Table 6.4.: Table of results for Simulation Setup 1 with **medium conditioning values**.

		Rejected tests	p-value		Eff. s. size		R-hat	
			min	max	min	max	min	max
Specification 1	n=1000	8%	0.006	1.000	0.644	1.210	0.999	1.015
	n=5000	7%	0.002	0.991	0.821	1.082	1.000	1.002
	n=10000	5%	0.017	0.976	0.890	1.057	1.000	1.001
Specification 2	n=1000	6%	0.021	0.993	0.711	1.154	0.999	1.007
	n=5000	2%	0.000	0.997	0.916	1.051	1.000	1.001
	n=10000	6%	0.028	0.972	0.891	1.053	1.000	1.001
Specification 3	n=1000	9%	0.000	0.984	0.533	1.110	0.999	1.007
	n=5000	6%	0.006	0.997	0.821	1.053	1.000	1.002
	n=10000	3%	0.000	0.994	0.854	1.030	1.000	1.001
Specification 4	n=1000	2%	0.016	0.995	0.749	1.148	0.999	1.007
	n=5000	7%	0.019	0.995	0.859	1.073	1.000	1.001
	n=10000	5%	0.000	1.000	0.848	1.045	1.000	1.001

Table 6.5.: Table of results for Simulation Setup 1 with **large conditioning values**.

Setup 2: D-vine, d=3, conditional density expressed with integration

In this setup we use the same vine tree structure as in the previous setup. We are sampling from the distribution of $(U_2|U_1 = u_1, U_3 = u_3)$, however in this case the density is not available without numerical integration. The conditional density and distribution are

$$c_{2|13}(u_2|u_1, u_3) = \frac{c_{123}}{c_{13}} = \frac{c_{123}(u_1, u_2, u_3)}{\int_0^1 c_{123}(u_1, t_2, u_3)dt_2},$$

$$C_{2|13}(u_2|u_1, u_3) = \int_0^{u_2} c_{2|13}dx_2 = \int_0^{u_2} \frac{c_{123}(u_1, x_2, u_3)}{\int_0^1 c_{123}(u_1, t_2, u_3)dt_2}dx_2.$$

For this setup, the chosen pair copula families and parameters are shown in Fig. 6.8. Similarly, every row corresponds to a different example with two D-vine copula specifications.

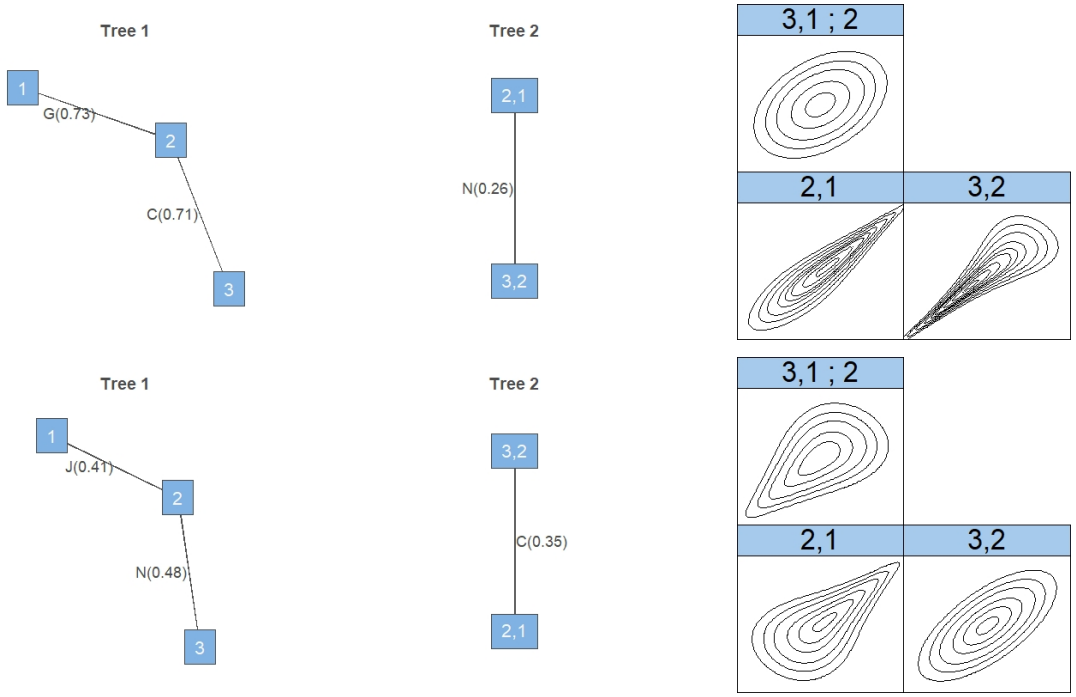

Figure 6.8.: Pair copula families and contour plots of the chosen D-vine copula specifications in the Simulation Setup 2.

The comparison of densities for low, medium and large conditioning values are shown in Fig. 6.9., Fig. 6.10 and Fig. 6.11, respectively. The table of results for low, medium and large conditioning values are presented in Table 6.6, Table 6.7 and Table 6.8, respectively.

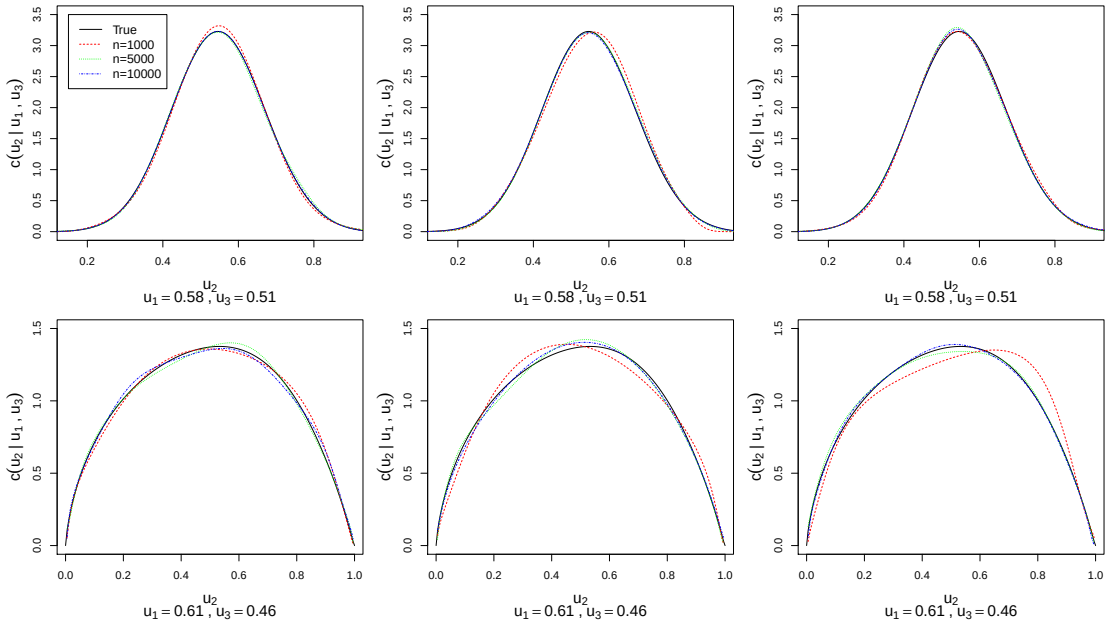

Figure 6.9.: Comparison of densities for Simulation Setup 2 with **low conditioning values**. Rows correspond to the 2 specifications shown in Fig. 6.8 and columns to 3 chosen simulation iterations out of 100.

		Rejected tests	p-value		Eff. s. size		R-hat	
			min	max	min	max	min	max
Specification 1	n=1000	6%	0.023	0.981	0.756	1.158	0.999	1.006
	n=5000	5%	0.006	0.999	0.870	1.062	1.000	1.002
	n=10000	6%	0.004	0.989	0.838	1.045	1.000	1.001
Specification 2	n=1000	6%	0.014	0.990	0.777	1.213	0.999	1.005
	n=5000	1%	0.028	0.992	0.883	1.067	1.000	1.001
	n=10000	6%	0.003	0.999	0.788	1.053	1.000	1.001

Table 6.6.: Table of results for Simulation Setup 2 with **low conditioning values**.

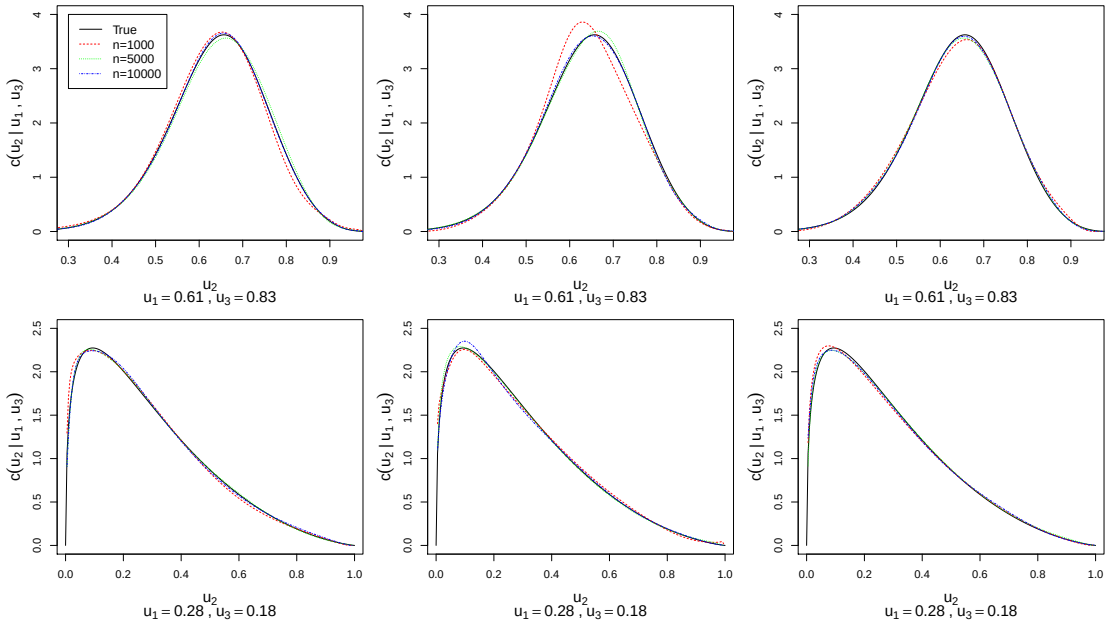

Figure 6.10.: Comparison of densities for Simulation Setup 2 with **medium conditioning values**. Rows correspond to the 2 specifications shown in Fig. 6.8 and columns to 3 chosen simulation iterations out of 100.

		Rejected tests	p-value		Eff. s. size		R-hat	
			min	max	min	max	min	max
Specification 1	n=1000	3%	0.040	0.984	0.768	1.121	0.999	1.008
	n=5000	7%	0.007	0.986	0.863	1.069	1.000	1.002
	n=10000	4%	0.001	0.988	0.879	1.031	1.000	1.001
Specification 2	n=1000	3%	0.025	1.000	0.695	1.141	0.999	1.007
	n=5000	7%	0.021	0.985	0.847	1.060	1.000	1.001
	n=10000	6%	0.006	0.996	0.814	1.045	1.000	1.001

Table 6.7.: Table of results for Simulation Setup 2 with **medium conditioning values**.

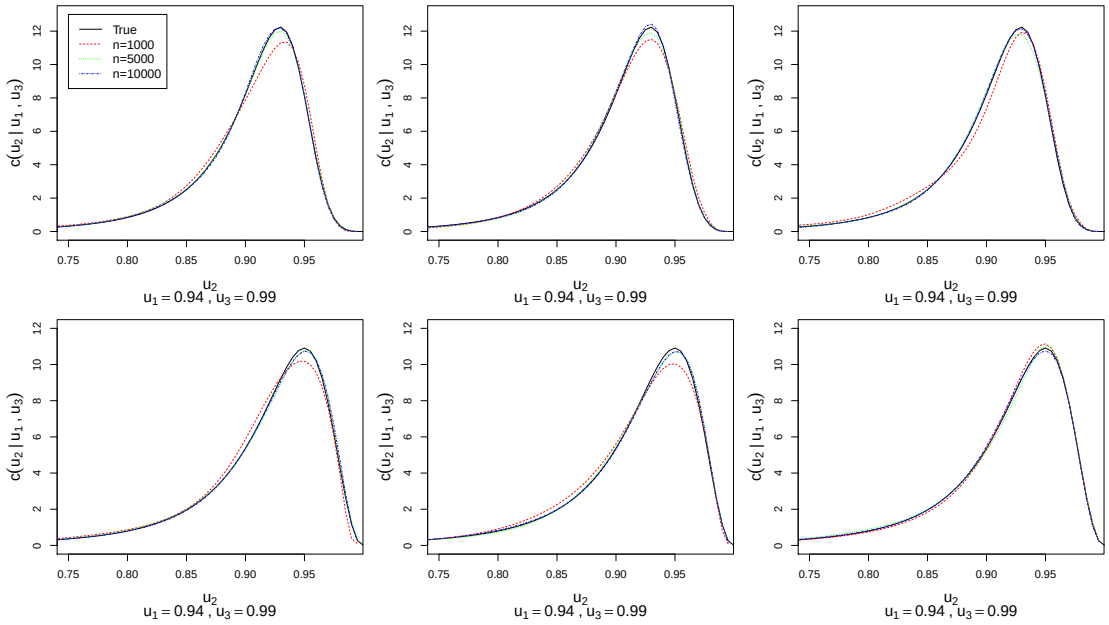

Figure 6.11.: Comparison of densities for Simulation Setup 2 with **large conditioning values**. Rows correspond to the 2 specifications shown in Fig. 6.8 and columns to 3 chosen simulation iterations out of 100.

		Rejected tests	p-value		Eff. s. size		R-hat	
			min	max	min	max	min	max
Specification 1	n=1000	4%	0.013	0.989	0.717	1.160	0.999	1.004
	n=5000	4%	0.012	0.959	0.786	1.057	1.000	1.002
	n=10000	7%	0.005	0.992	0.886	1.043	1.000	1.001
Specification 2	n=1000	6%	0.006	0.980	0.733	1.201	0.999	1.005
	n=5000	5%	0.003	0.983	0.840	1.075	1.000	1.001
	n=10000	0%	0.059	0.995	0.886	1.034	1.000	1.000

Table 6.8.: Table of results for Simulation Setup 2 with **large conditioning values**.

Setup 3: D-vine, d=5, conditional density expressed without integration

In this setup we use the vine tree structure from Fig. 6.2. We are sampling from the distribution of $(U_1|U_2 = u_2, \ldots, U_5 = u_5)$. The conditional density and distribution are

$$\begin{aligned} c_{1|2345}(u_1|u_2,u_3,u_4,u_5) = \frac{c_{12345}}{c_{2345}} = \frac{c_{1,2}c_{2,3}c_{3,4}c_{4,5}c_{1,3;2}c_{2,4;3}c_{3,5;4}c_{1,4;2,3}c_{2,5;3,4}c_{1,5;2,3,4}}{c_{2,3}c_{3,4}c_{4,5}c_{2,4;3}c_{3,5;4}c_{2,5;3,4}} = \\ = c_{1,2}(u_1,u_2)\cdot c_{1,3;2}(C_{1|2}(u_1|u_2), C_{3|2}(u_3|u_2)) \\ \cdot c_{1,4;2,3}(C_{1|23}(u_1|u_2,u_3), C_{4|23}(u_4|u_2,u_3)) \\ \cdot c_{1,5;2,3,4}(C_{1|234}(u_1|u_2,u_3,u_4), C_{5|234}(u_5|u_2,u_3,u_4)), \end{aligned}$$

$$C_{1|2345}(u_1|u_2,u_3,u_4,u_5) = h_{1|5;234}(C_{1|234} \mid C_{5|234}),$$

where

$$\begin{aligned} C_{1|234} &= h_{1|4;23}(C_{1|23} \mid C_{4|23}) \\ C_{5|234} &= h_{5|2;34}(C_{5|34} \mid C_{2|34}) \\ C_{1|23} &= h_{1|3;2}(h_{1|2}(u_1|u_2) \mid h_{3|2}(u_3|u_2)) \\ C_{4|23} &= h_{4|2;3}(h_{4|3}(u_4|u_3) \mid h_{2|3}(u_2|u_3)) \\ C_{5|34} &= h_{5|3;4}(h_{5|4}(u_5|u_4) \mid h_{3|4}(u_3|u_4)) \\ C_{2|34} &= h_{2|4;3}(h_{2|3}(u_2|u_3) \mid h_{4|3}(u_4|u_3)). \end{aligned}$$

In this setup we look at one D-vine specification. Its chosen pair copula families and parameters can be seen in Fig. 6.12.

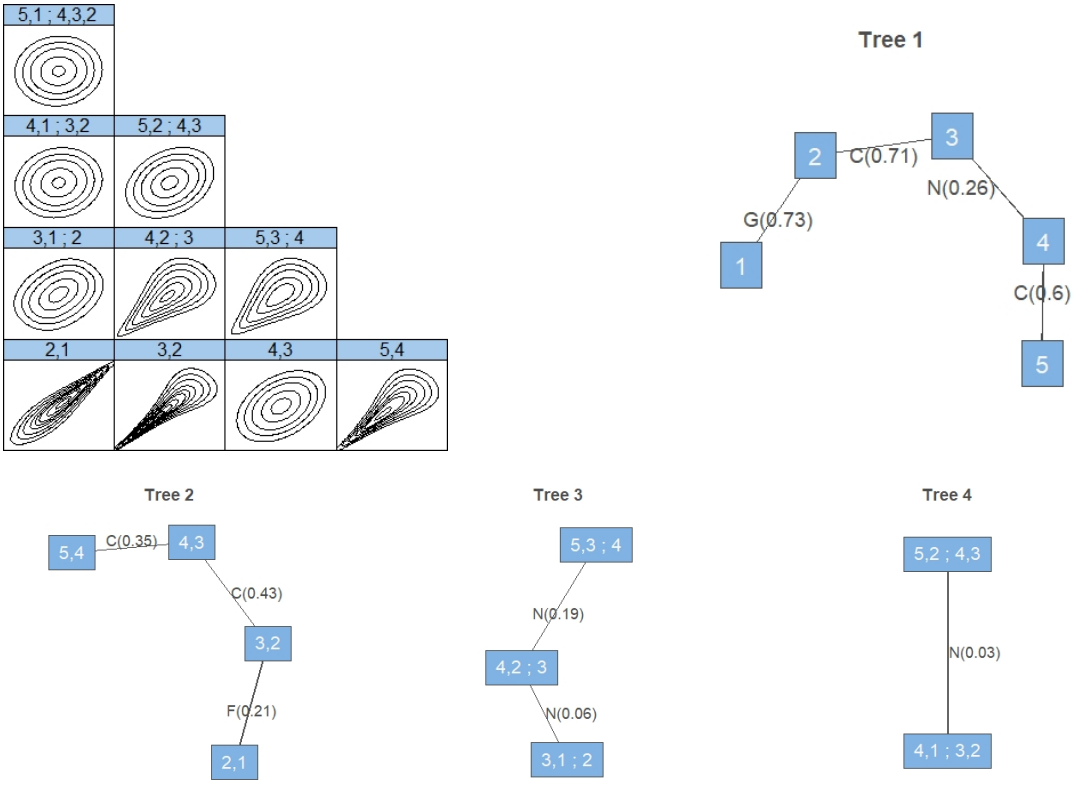

Figure 6.12.: Pair copula families and contour plots of the chosen D-vine specification in the Simulation Setup 3. First panel depicts the pair copula contour plots on the z-scale, and the remaining panels the D-vine tree structure with the copula families and corresponding Kendall's τ parameter.

The comparison of densities for low, medium and large conditioning values are shown together in Fig. 6.13. The table of results for low, medium and large conditioning values are presented together in Table 6.9.

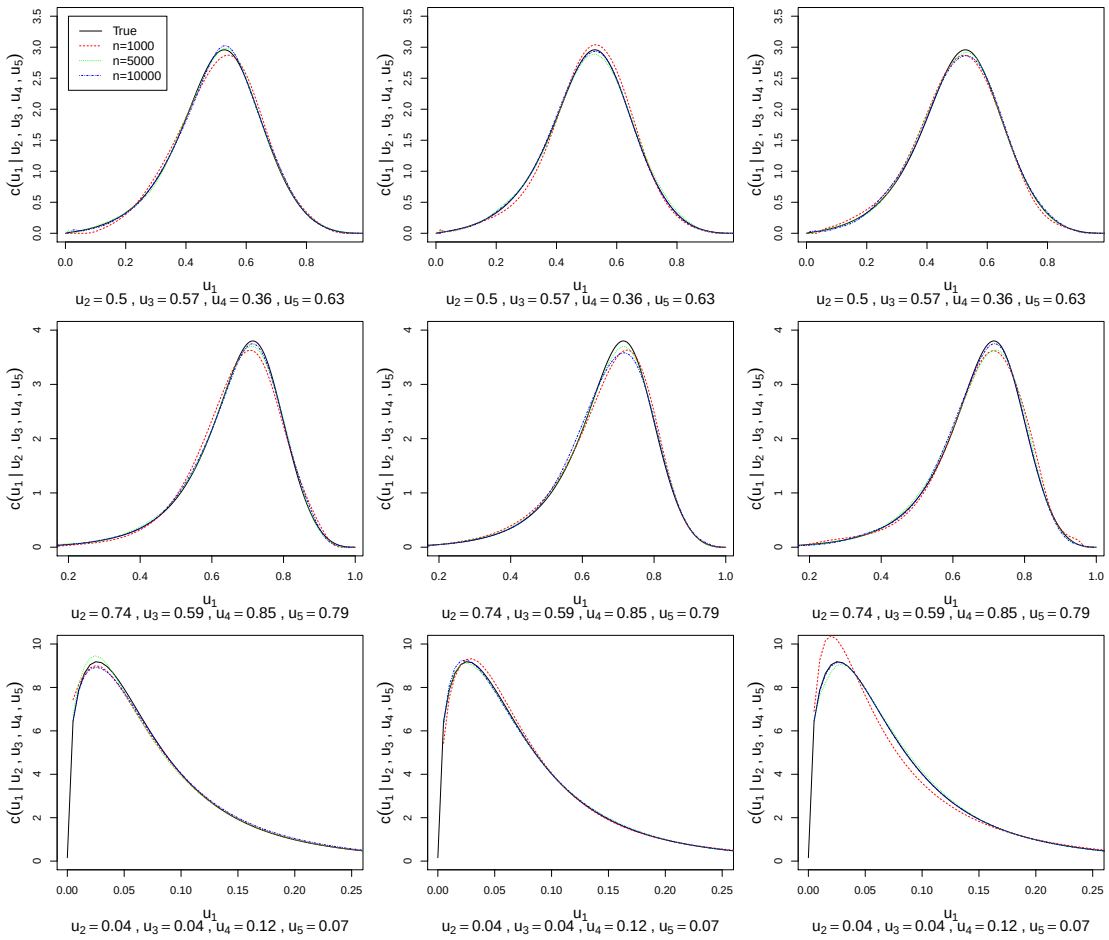

Figure 6.13.: Comparison of densities for Simulation Setup 3. For the chosen vine specification, each column shows the plot of one chosen iteration out of 100. The rows correspond to different conditioning values with order low, medium and large.

Conditioning values		Rejected tests	p-value		Eff. s. size		R-hat	
			min	max	min	max	min	max
low	n=1000	5%	0.009	0.998	0.727	1.195	0.999	1.007
	n=5000	2%	0.019	0.978	0.804	1.049	1.000	1.001
	n=10000	5%	0.009	0.999	0.874	1.053	1.000	1.000
medium	n=1000	4%	0.006	0.999	0.687	1.157	0.999	1.006
	n=5000	5%	0.004	0.984	0.816	1.063	1.000	1.001
	n=10000	2%	0.014	0.999	0.903	1.030	1.000	1.001
large	n=1000	2%	0.009	0.998	0.719	1.184	0.999	1.007
	n=5000	8%	0.013	0.990	0.846	1.071	1.000	1.001
	n=10000	4%	0.021	0.999	0.852	1.070	1.000	1.001

Table 6.9.: Table of results for Simulation Setup 3.

Setup 4: D-vine, d=5, conditional density expressed with integration

In this setup we use the same vine tree structure as in the previous one. We are sampling from the distribution of $(U_2|U_1 = u_1, U_3 = u_3, \ldots, U_5 = u_5)$, however in this case the density is not available without numerical integration. The conditional density and distribution are

$$c_{2|1345}(u_2|u_1, u_3, u_4, u_5) = \frac{c_{12345}}{c_{1345}} = \frac{c_{12345}(u_1, u_2, u_3, u_4, u_5)}{\int_0^1 c_{12345}(u_1, t_2, u_3, u_4, u_5)dt_2},$$

$$C_{2|1345}(u_2|u_1, u_3, u_4, u_5) = \int_0^{u_2} c_{2|1345}dx_2 = \int_0^{u_2} \frac{c_{12345}(u_1, x_2, u_3, u_4, u_5)}{\int_0^1 c_{12345}(u_1, t_2, u_3, u_4, u_5)dt_2}dx_2.$$

The pair copula families and parameters chosen for one D-vine specification are shown in Fig. 6.14.

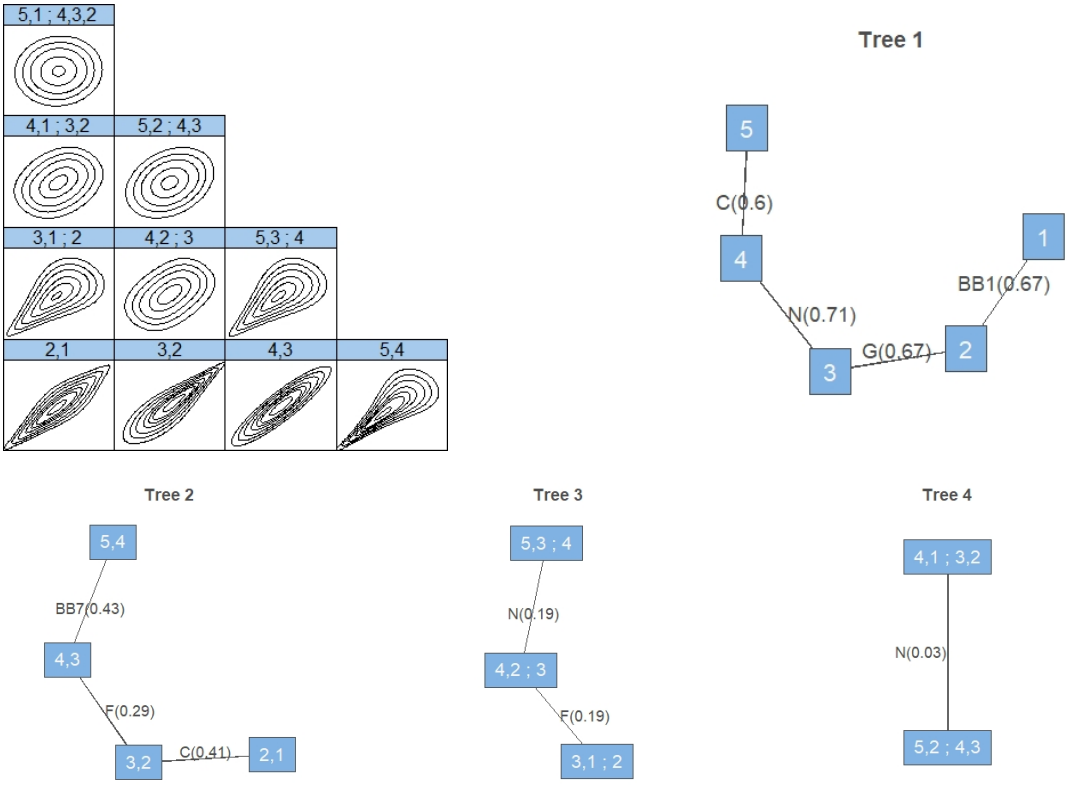

Figure 6.14.: Pair copula families and contour plots of the chosen D-vine specification in the Simulation Setup 4.

The comparison of densities for low, medium and large conditioning values are shown in Fig. 6.15. The table of results is presented in Table 6.10.

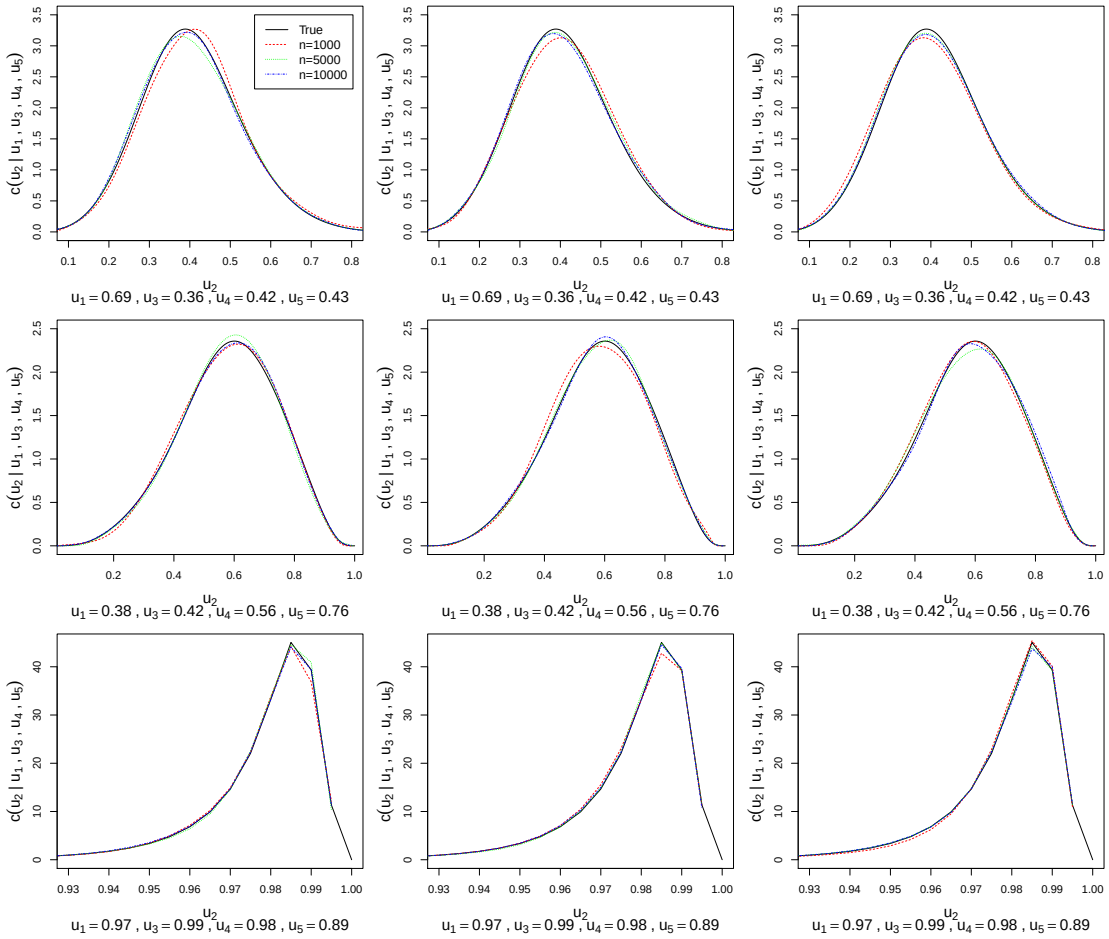

Figure 6.15.: Comparison of densities for Simulation Setup 4. For the chosen vine specification, each column shows the plot of one chosen iteration out of 100. The rows correspond to different conditioning values with order low, medium and large.

Conditioning values		Rejected tests	p-value		Eff. s. size		R-hat	
			min	max	min	max	min	max
low	n=1000	6%	0.000	0.998	0.733	1.136	0.999	1.006
	n=5000	4%	0.006	0.999	0.839	1.094	1.000	1.001
	n=10000	5%	0.022	0.992	0.888	1.040	1.000	1.001
medium	n=1000	4%	0.002	0.999	0.718	1.156	0.999	1.006
	n=5000	4%	0.010	0.999	0.904	1.070	1.000	1.001
	n=10000	6%	0.003	0.995	0.885	1.044	1.000	1.001
large	n=1000	5%	0.003	0.995	0.693	1.193	0.999	1.006
	n=5000	6%	0.006	0.976	0.851	1.073	1.000	1.002
	n=10000	4%	0.014	0.998	0.859	1.041	1.000	1.001

Table 6.10.: Table of results for Simulation Setup 4.

Setup 5: R-vine, d=5, conditional density expressed without integration

In this setup we use the vine tree structure from Fig. 6.3. We are sampling from the distribution of $(U_5|U_1 = u_1, \dots, U_4 = u_4)$. The conditional density and distribution are

$$\begin{aligned} c_{5|1234}(u_5|u_1,u_2,u_3,u_4) = \frac{c_{12345}}{c_{1234}} = \frac{c_{1,5}c_{1,4}c_{2,4}c_{3,4}c_{4,5;1}c_{1,2;4}c_{1,3;4}c_{3,5;1,4}c_{2,3;1,4}c_{2,5;1,3,4}}{c_{1,4}c_{2,4}c_{3,4}c_{1,2;4}c_{1,3;4}c_{2,3;1,4}} = \\ = c_{1,5}(u_1,u_5)\cdot c_{4,5;1}(C_{4|1}(u_4|u_1), C_{5|1}(u_5|u_1)) \\ \cdot c_{3,5;1,4}(C_{3|14}(u_3|u_1,u_4), C_{5|14}(u_5|u_1,u_4)) \\ \cdot c_{2,5;1,3,4}(C_{2|134}(u_2|u_1,u_3,u_4), C_{5|134}(u_5|u_1,u_3,u_4)), \end{aligned}$$

$$C_{5|1234}(u_5|u_1,u_2,u_3,u_4) = h_{5|2;134}(C_{5|134} \mid C_{2|134}),$$

where

$$\begin{aligned} C_{5|134} &= h_{5|3;14}(C_{5|14} \mid C_{3|14}) \\ C_{2|134} &= h_{2|3;14}(C_{2|14} \mid C_{3|14}) \\ C_{5|14} &= h_{5|4;1}(h_{5|1}(u_5|u_1) \mid h_{4|1}(u_4|u_1)) \\ C_{3|14} &= h_{3|1;4}(h_{3|4}(u_3|u_4) \mid h_{1|4}(u_1|u_4)) \\ C_{2|14} &= h_{2|1;4}(h_{2|4}(u_2|u_4) \mid h_{1|4}(u_1|u_4)). \end{aligned}$$

In this setup we look at one R-vine specification. Its chosen pair copula families and parameters can be seen in Fig. 6.16.

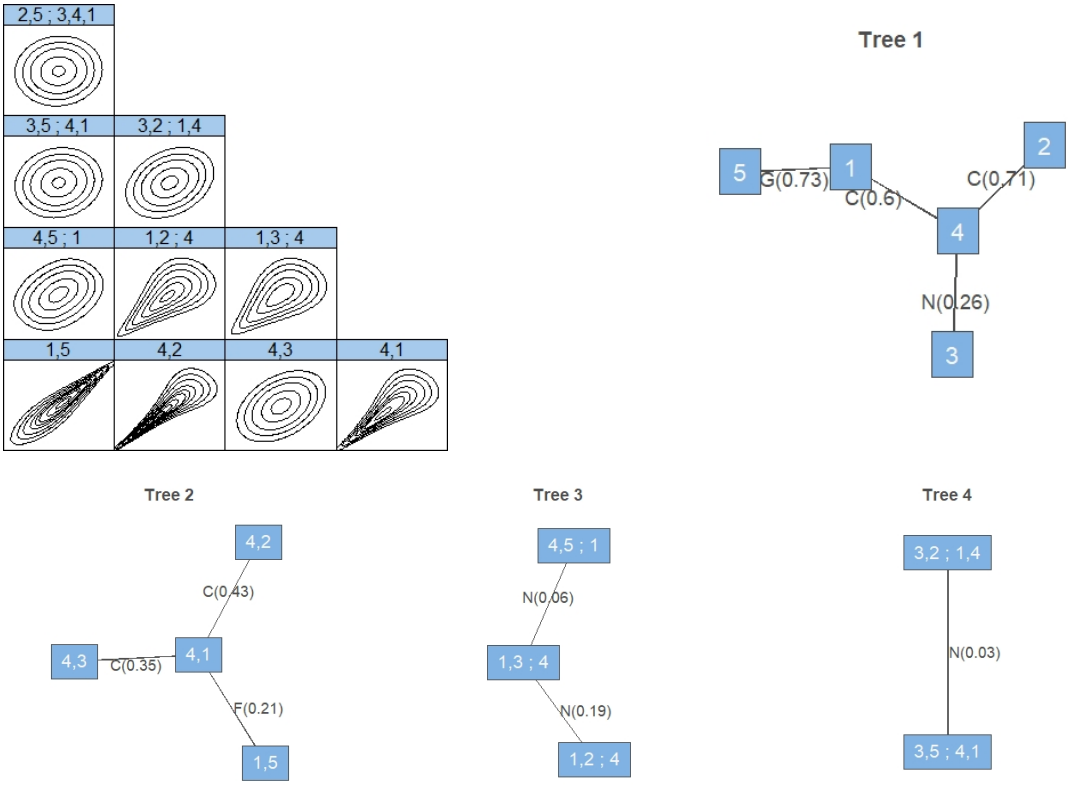

Figure 6.16.: Pair copula families and contour plots of the chosen R-vine specification in the Simulation Setup 5.

The comparison of densities for low, medium and large conditioning values are shown in Fig. 6.17. The table of results is presented in Table 6.11.

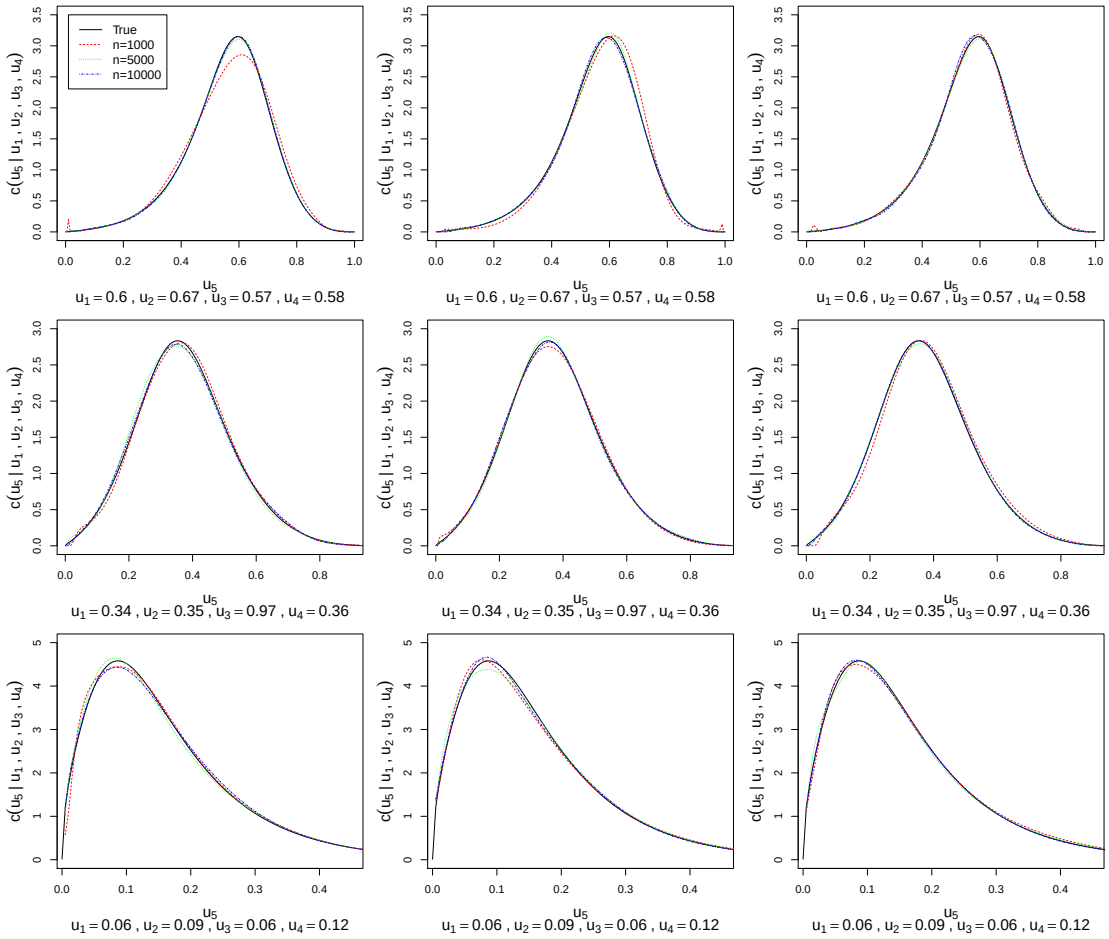

Figure 6.17.: Comparison of densities for Simulation Setup 5. For the chosen vine specification, each column shows the plot of one chosen iteration out of 100. The rows correspond to different conditioning values with order low, medium and large.

Conditioning values		Rejected tests	p-value		Eff. s. size		R-hat	
			min	max	min	max	min	max
low	n=1000	3%	0.019	0.997	0.656	1.178	0.999	1.004
	n=5000	5%	0.010	0.995	0.865	1.067	1.000	1.001
	n=10000	5%	0.004	0.992	0.888	1.042	1.000	1.001
medium	n=1000	10%	0.010	0.999	0.754	1.137	0.999	1.010
	n=5000	7%	0.003	1.000	0.865	1.069	1.000	1.001
	n=10000	8%	0.001	1.000	0.818	1.041	1.000	1.001
large	n=1000	3%	0.016	0.993	0.631	1.161	0.999	1.008
	n=5000	5%	0.002	0.997	0.886	1.053	1.000	1.001
	n=10000	3%	0.011	0.985	0.858	1.047	1.000	1.001

Table 6.11.: Table of results for Simulation Setup 5.

Setup 6: R-vine, d=5, conditional density expressed with integration

In this setup we use the same vine tree structure as in the previous one. We are sampling from the distribution of $(U_2|U_1 = u_1, U_3 = u_3, \ldots, U_5 = u_5)$, however in this case the density is not available without integration. The conditional density and distribution are

$$c_{2|1345}(u_2|u_1, u_3, u_4, u_5) = \frac{c_{12345}}{c_{1345}} = \frac{c_{12345}(u_1, u_2, u_3, u_4, u_5)}{\int_0^1 c_{12345}(u_1, t_2, u_3, u_4, u_5)dt_2},$$

$$C_{2|1345}(u_2|u_1, u_3, u_4, u_5) = \int_0^{u_2} c_{2|1345}dx_2 = \int_0^{u_2} \frac{c_{12345}(u_1, x_2, u_3, u_4, u_5)}{\int_0^1 c_{12345}(u_1, t_2, u_3, u_4, u_5)dt_2}dx_2.$$

The pair copula families and parameters chosen for one R-vine specification are shown in Fig. 6.18.

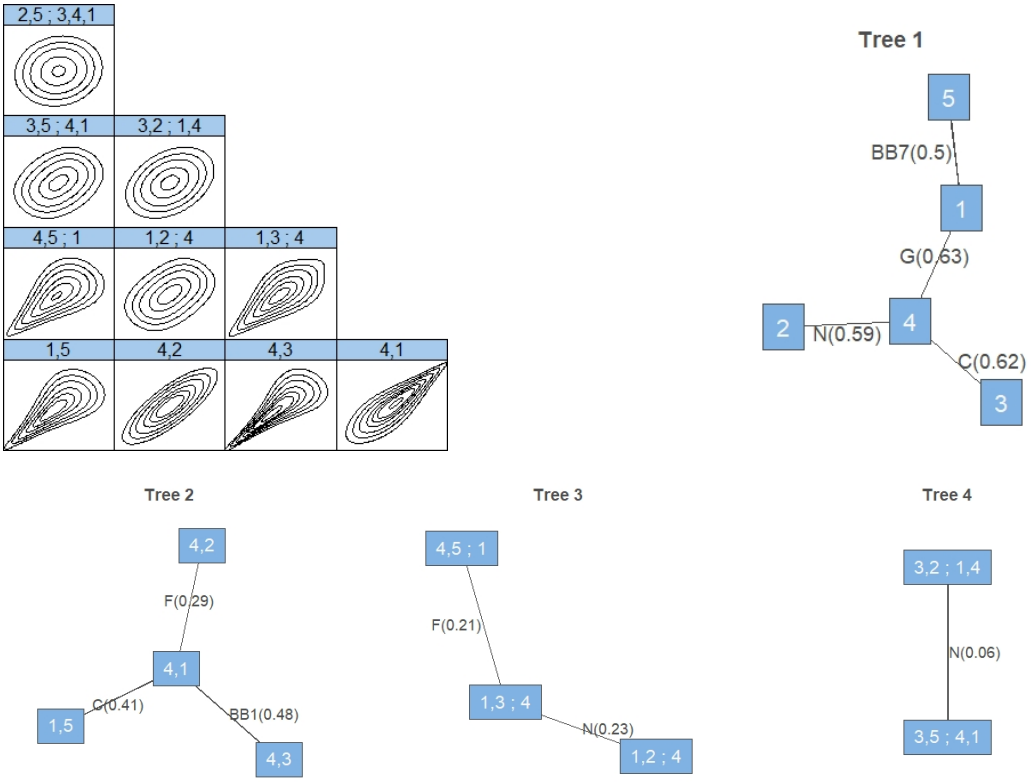

Figure 6.18.: Pair copula families and contour plots of the chosen R-vine specification in the Simulation Setup 6.

The comparison of densities for low, medium and large conditioning values are shown in Fig. 6.19. The table of results is presented in Table 6.12.

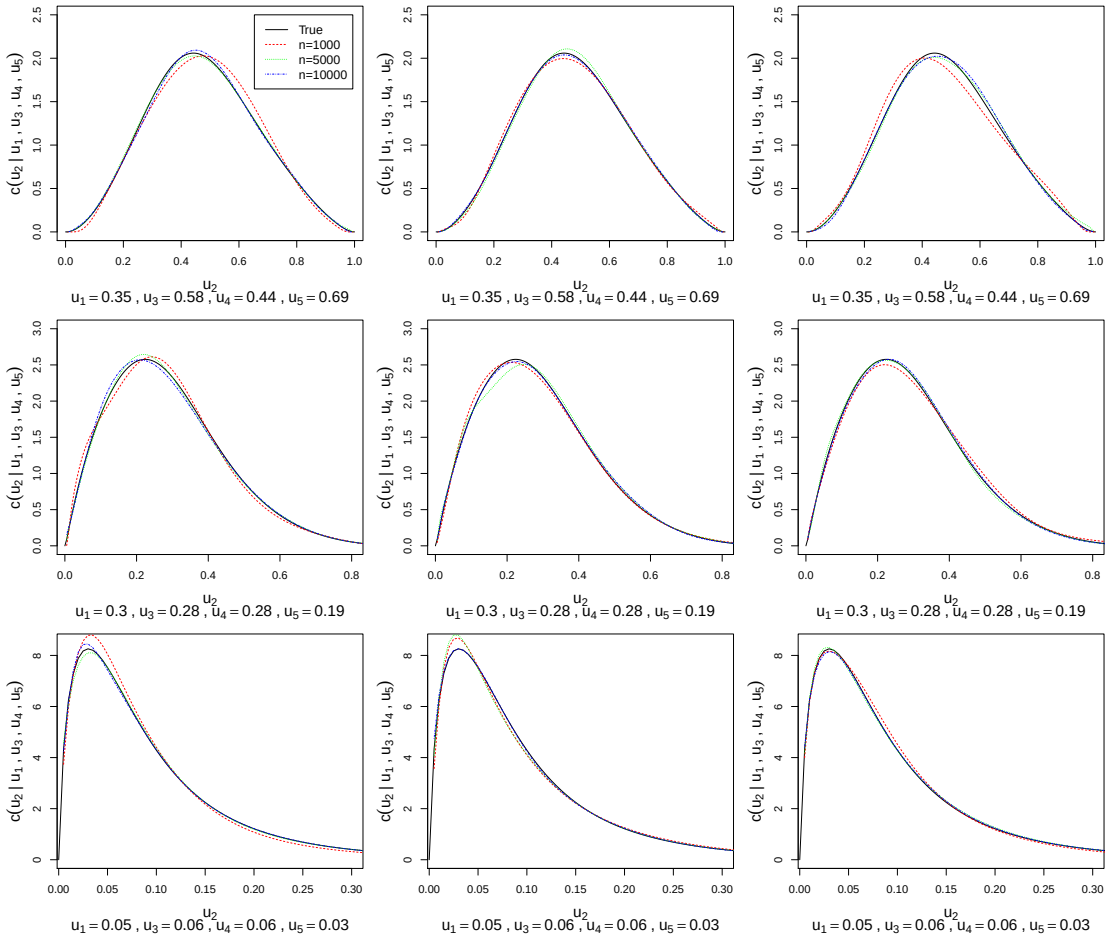

Figure 6.19.: Comparison of densities for Simulation Setup 6. For the chosen vine specification, each column shows the plot of one chosen iteration out of 100. The rows correspond to different conditioning values with order low, medium and large.

Conditioning values		Rejected tests	p-value		Eff. s. size		R-hat	
			min	max	min	max	min	max
low	n=1000	4%	0.001	0.991	0.784	1.184	0.999	1.006
	n=5000	3%	0.003	0.998	0.852	1.054	1.000	1.002
	n=10000	6%	0.002	0.997	0.901	1.034	1.000	1.000
medium	n=1000	1%	0.044	0.995	0.738	1.159	0.999	1.005
	n=5000	3%	0.015	0.981	0.869	1.077	1.000	1.002
	n=10000	3%	0.000	0.991	0.796	1.054	1.000	1.000
large	n=1000	4%	0.023	0.991	0.775	1.174	0.999	1.008
	n=5000	5%	0.005	1.000	0.835	1.078	1.000	1.001
	n=10000	7%	0.003	0.998	0.890	1.053	1.000	1.001

Table 6.12.: Table of results for Simulation Setup 6.

6.3. Case II: Sampling from Bivariate Cond. Distribution Functions Arising from a Vine Copula

Simulation Setup

We continue the simulation study with bivariate conditional distributions. Similarly, as in the univariate case, we look at D-vine tree structure as well as one specific R-vine, conditional densities easily expressed without integration as well as ones that cannot be expressed without integration. The dimension d is set here to 3 and 5, and in every simulation setup we sample $n = 1000, 5000$ and 10000 samples with 3 different sets of conditioning values, likewise their distance from a central vector $\mathbf{u_c} := (0.5, 0.5, \ldots, 0.5) \in \mathbb{R}^d$ being extremely low, medium and extremely large. The summary of the studied simulation setups is shown in Table 6.13. In the following, we first introduce the approach and then present the results.

First we specify pair copula families and parameters for the vines associated with the specific vine tree structure.

Setup	Vine	d	Density availability	Conditioned, conditioning set
1	D	3	T	$\mathcal{C}_1 = \{1,2\}, \mathcal{C}_2 = \{3\}$
2	D	5	F	$\mathcal{C}_1 = \{2,4\}, \mathcal{C}_2 = \{1,3,5\}$
3	R	5	F	$\mathcal{C}_1 = \{2,4\}, \mathcal{C}_2 = \{1,3,5\}$

Table 6.13.: Selected simulation setups in sampling from bivariate conditional distributions. Density availability means whether we can express the conditional density without integration, hence true or false. In conditioning and conditioned set we show the chosen conditioning and conditioned variables.

Choosing the Conditioning Values

The selection of the conditioning values for all simulations is the same as in the univariate case. In contrast with the univariate case, in sampling from bivariate conditional distribution we choose $\alpha = 0.05, 0.5, 0.85$. Likewise, we call the conditioning values for $\alpha = 0.05$ **low**, the one for $\alpha = 0.5$ **medium** and $\alpha = 0.85$ **large**. The reason for the alteration in this case is that the conditioning values from the $\alpha = 0.95$ quantile

are so extreme, that in the distribution there is extreme tail dependence. We are now interested in sampling from $\mathbf{U}_{\mathcal{C}_1}|\mathbf{U}_{\mathcal{C}_2} = \mathbf{u}_{r_\alpha}^{\mathcal{C}_2}$, where $\mathbf{u}_{r_\alpha}^{\mathcal{C}_2} = (u_{r_\alpha,\mathcal{C}_{2,1}}, \ldots, u_{r_\alpha,\mathcal{C}_{2,\ell}})^T$.

Sampling

The sampling is performed exactly the same way as in the previous case. We denote the sampled values by $\mathbf{u}_i(\mathbf{u}_{r_\alpha}^{\mathcal{C}_2})$, $i = 1, \ldots, n$.

Density Estimation

For examining visually whether the samples are from the desired distribution we use kernel density estimation and compare the plot of the estimated density with the plot of the true one. During the estimation, we use the transformation trick, i.e. we transform the samples to the z-scale, estimate the bivariate density and scale it back as shown in Eq. (4.5). We do it for all sample sizes $n = 1000$, 5000 and 10000.

Performance Measures

To measure the goodness-of-fit, we perform similar testing as in the univariate case. However, in the bivariate conditional case we cannot use the probability integral transform, therefore we use the Rosenblatt transformation defined in Definition 4.9. We apply the transformation on the samples $\mathbf{u}_i(\mathbf{u}_{r_\alpha}^{\mathcal{C}_2})$, $i = 1, \ldots, n$ and obtain

$$
\begin{aligned}
v_{i1}^m &= F_{\mathcal{C}_{11}|\mathcal{C}_2}(u_{i1}(\mathbf{u}_{r_\alpha}^{\mathcal{C}_2})|\mathbf{u}_{r_\alpha}^{\mathcal{C}_2}),\\
v_{i2}^c &= F_{\mathcal{C}_{12}|\mathcal{C}_{11},\mathcal{C}_2}(u_{i2}(\mathbf{u}_{r_\alpha}^{\mathcal{C}_2})|u_{i1}(\mathbf{u}_{r_\alpha}^{\mathcal{C}_2}), \mathbf{u}_{r_\alpha}^{\mathcal{C}_2}),
\end{aligned}
$$

or the other order of variables

$$
\begin{aligned}
v_{i2}^m &= F_{\mathcal{C}_{12}|\mathcal{C}_2}(u_{i2}(\mathbf{u}_{r_\alpha}^{\mathcal{C}_2})|\mathbf{u}_{r_\alpha}^{\mathcal{C}_2}),\\
v_{i1}^c &= F_{\mathcal{C}_{11}|\mathcal{C}_{12},\mathcal{C}_2}(u_{i1}(\mathbf{u}_{r_\alpha}^{\mathcal{C}_2})|u_{i2}(\mathbf{u}_{r_\alpha}^{\mathcal{C}_2}), \mathbf{u}_{r_\alpha}^{\mathcal{C}_2}).
\end{aligned}
$$

The pair $\{v_{i1}^m, v_{i2}^c\}$ or $\{v_{i1}^c, v_{i2}^m\}$ have then bivariate independent uniform distributions. After obtaining $v_{i1}^m, v_{i2}^c, v_{i1}^c$ and v_{i2}^m, we test if the pairs $\{v_{i1}^m, v_{i2}^c\}$ and $\{v_{i1}^c, v_{i2}^m\}$ are independent uniformly distributed. To test for uniform distributions we use Kolmogorov-Smirnov test. In order to test both pairs for independence, we use bivariate dependence test defined in Definition 4.7. In total, we perform 6 tests summarized in Table 6.14. Since we perform more than one test on one bivariate sample, we use Bonferroni

correction to avoid multiple testing problem. The significance cut-off is now set to $\alpha/6 = 0.05/6 = 0.0083$, where 6 is the number of tests performed. We reject that the sample is from the desired distribution if at least one p-value out of six is below this threshold.

Number	Variable(s)	Name of the test	Test for
1	v_{i1}^m	Kolmogorov-Smirnov test	uniform distribution
2	v_{i2}^c	Kolmogorov-Smirnov test	uniform distribution
3	v_{i1}^c	Kolmogorov-Smirnov test	uniform distribution
4	v_{i2}^m	Kolmogorov-Smirnov test	uniform distribution
5	$\{v_{i1}^m, v_{i2}^c\}$	Bivariate dependence test	independence
6	$\{v_{i1}^c, v_{i2}^m\}$	Bivariate dependence test	independence

Table 6.14.: List of the tests performed in order to measure the goodness-of-fit of the sample $\mathbf{u}_i(\mathbf{u}_{r_\alpha}^{\mathcal{C}_2})$, $i = 1, \ldots, n$.

6.4. Results - Case II

In this section we present sampling results for bivariate case for all 3 simulation setups shown in Table 6.13. The chosen number of simulations is $N = 100$ resulting in 100 performance measures each consisting of 6 tests summarized in Table 6.14.

Setup 1: D-vine, d=3

In this setup we use the vine tree structure from Fig. 6.1. We are sampling from the distribution of $(U_1, U_2|U_3 = u_3)$. The conditional density is

$$c_{12|3}(u_1, u_2|u_3) = \frac{c_{123}}{c_3} = \frac{c_{123}}{1} = c_{123}(u_1, u_2, u_3),$$

and the Rosenblatt transformation is expressed as

$$v_1^m = C(u_1|u_3) = \int_0^{u_1} c(a_1|u_3)da_1 = \int_0^{u_1} \frac{c(a_1,u_3)}{c(u_3)}da_1 = \int_0^{u_1} \frac{c(a_1,u_3)}{1}da_1 =$$
$$= \int_0^{u_1}\int_0^1 c(a_1,a_2,u_3)da_2da_1,$$
$$v_2^c = C(u_2|u_1,u_3) = \int_0^{u_2} c(a_2|u_1,u_3)da_2 = \int_0^{u_2} \frac{c(u_1,a_2,u_3)}{c(u_1,u_3)}da_2 = \frac{\int_0^{u_2} c(u_1,a_2,u_3)da_2}{\int_0^1 c(u_1,a_2,u_3)da_2}$$

and

$$v_2^m = C(u_2|u_3) = \int_0^{u_2} c(a_2|u_3)da_2 = \int_0^{u_2} \frac{c(a_2,u_3)}{c(u_3)}da_2 = \int_0^{u_2} \frac{c(a_2,u_3)}{1}da_2 =$$
$$= \int_0^{u_2}\int_0^1 c(a_1,a_2,u_3)da_1da_2,$$
$$v_1^c = C(u_1|u_2,u_3) = h_{1|3;2}(h_{1|2}(u_1|u_2)\,|\,h_{3|2}(u_3|u_2)).$$

In this setup we look at one D-vine specification. Its chosen pair copula families and parameters can be seen in Fig. 6.20.

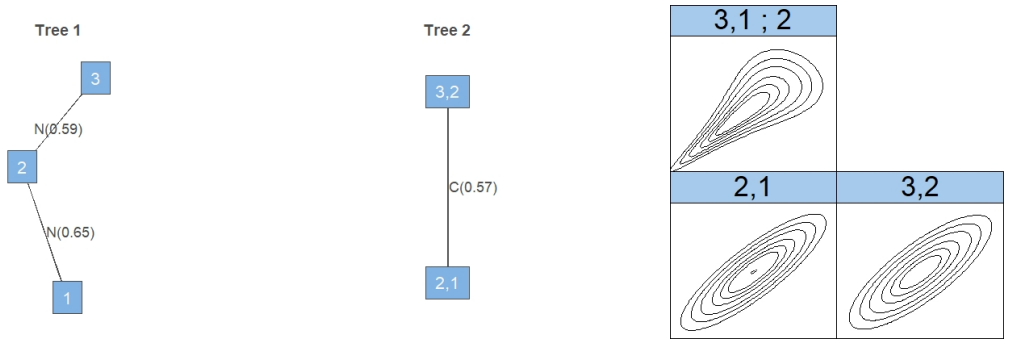

Figure 6.20.: Pair copula families and contour plots of the chosen D-vine specification in the bivariate Simulation Setup 1. First two columns depict the copula families with corresponding Kendall's τ parameter and the third one the pair copula contour plots on the z-scale.

Since in the bivariate case we cannot combine the estimated densities and the true one into one plot, we first present the true conditional density. We then continue with 3 simulation iterations of estimated conditional densities of samples of sizes $n = 1000$, 5000 and 10000. Finally, we present the comparisons of the contour plots of the true and estimated densities. We do this for all 3 types of conditioning values.

The true conditional density with **low** conditioning value is shown in Fig. 6.21, the estimated densities in Fig. 6.22 and the comparison of contour plots in Fig. 6.23.

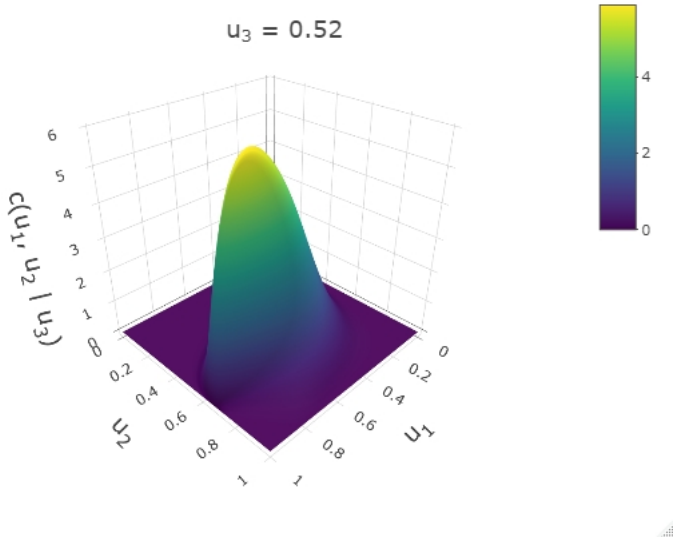

Figure 6.21.: True bivariate conditional density with **low conditioning value** in Simulation Setup 1.

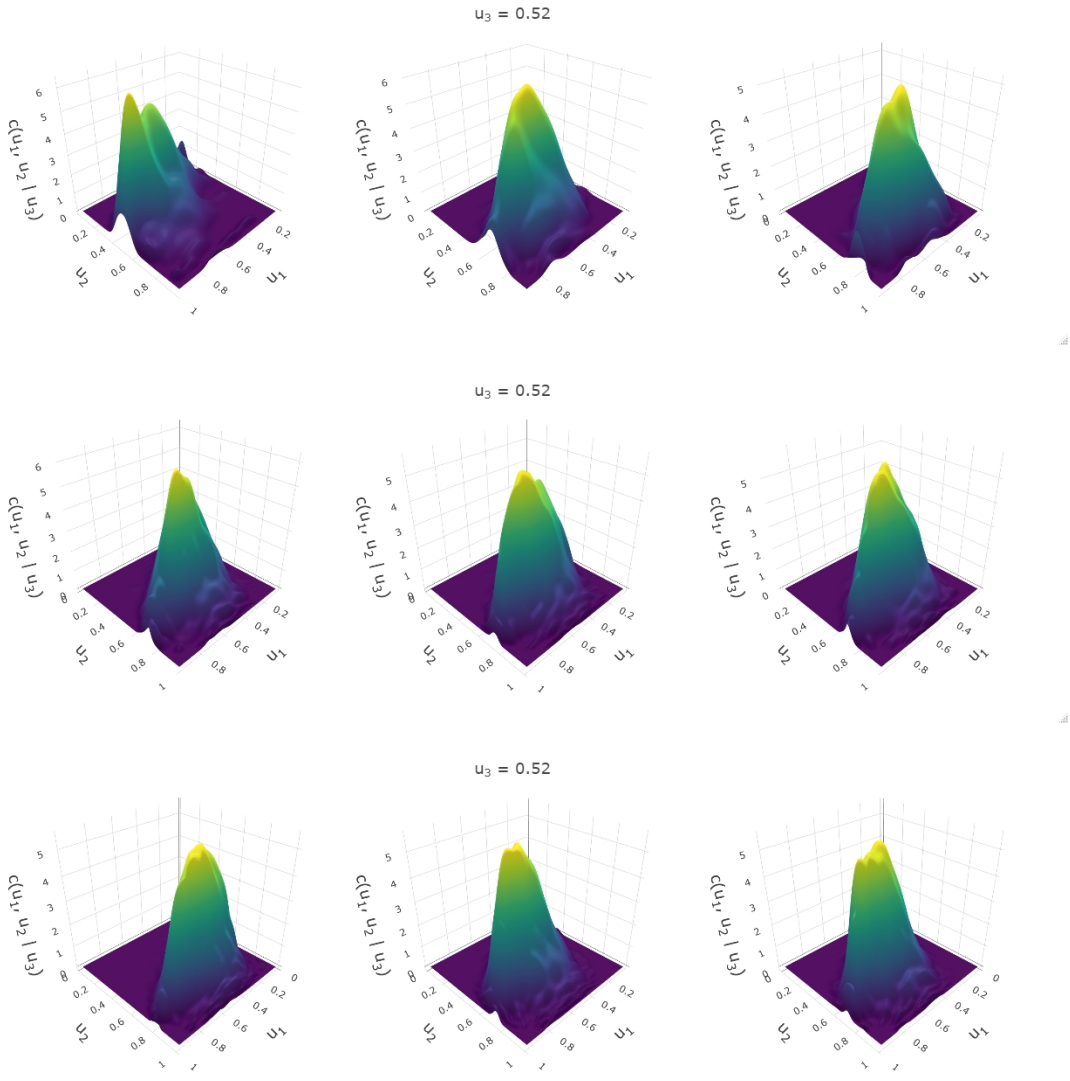

Figure 6.22.: Estimated densities with **low conditioning value** for Simulation Setup 1. For the chosen vine specification, each column shows the plot of one chosen iteration. The rows correspond to different sample sizes with order $n = 1000$, 5000 and 10000.

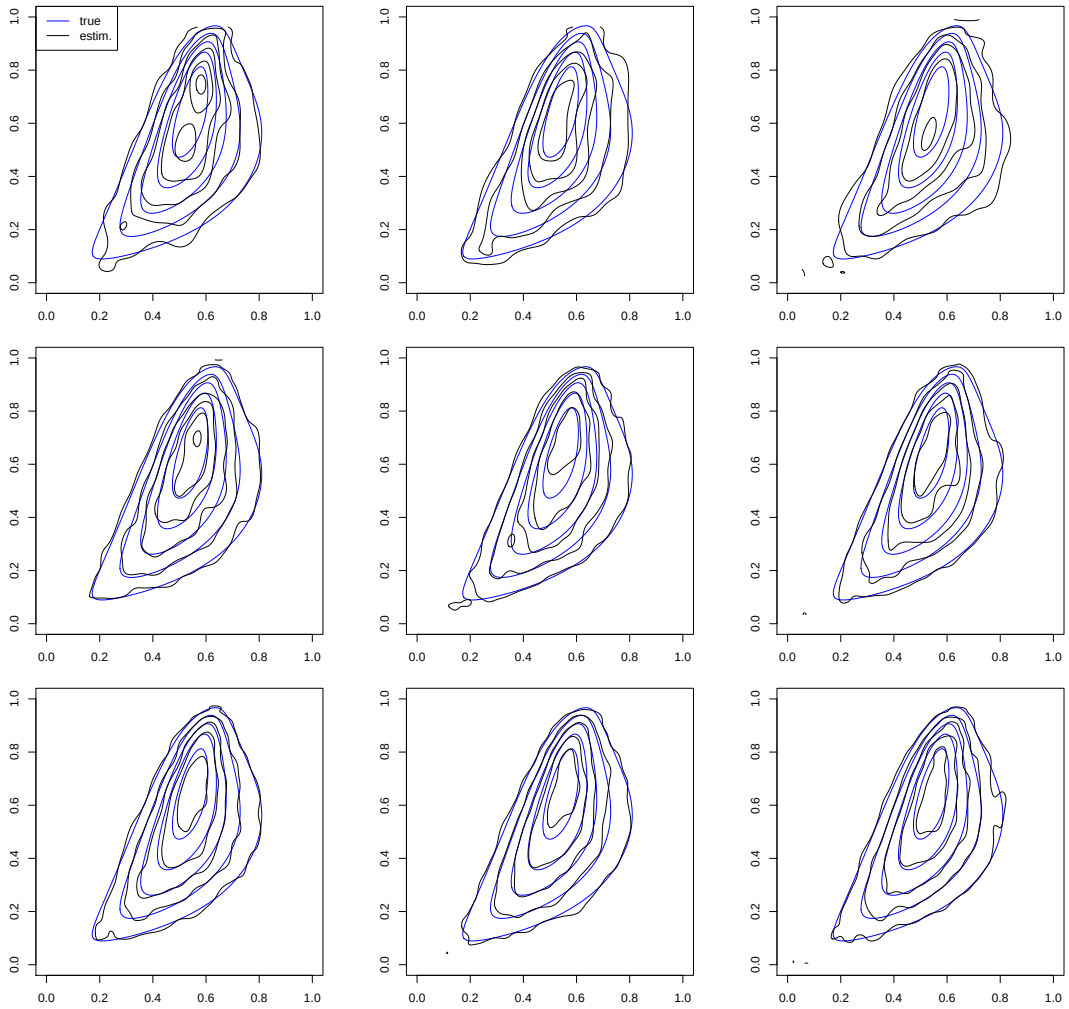

Figure 6.23.: Comparison of estimated densities and the true density with **low conditioning value** for Simulation Setup 1. Each column shows the plot for one chosen iteration. The rows correspond to different sample sizes with order $n = 1000$, 5000 and 10000.

The true conditional density with **medium** conditioning value is shown in Fig. 6.24, the estimated densities in Fig. 6.25 and the comparison of contour plots in Fig. 6.26.

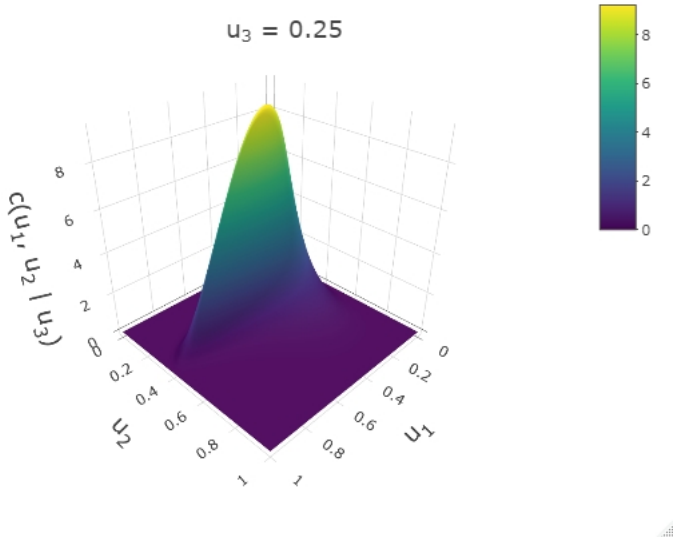

Figure 6.24.: True bivariate conditional density with **medium conditioning value** in Simulation Setup 1.

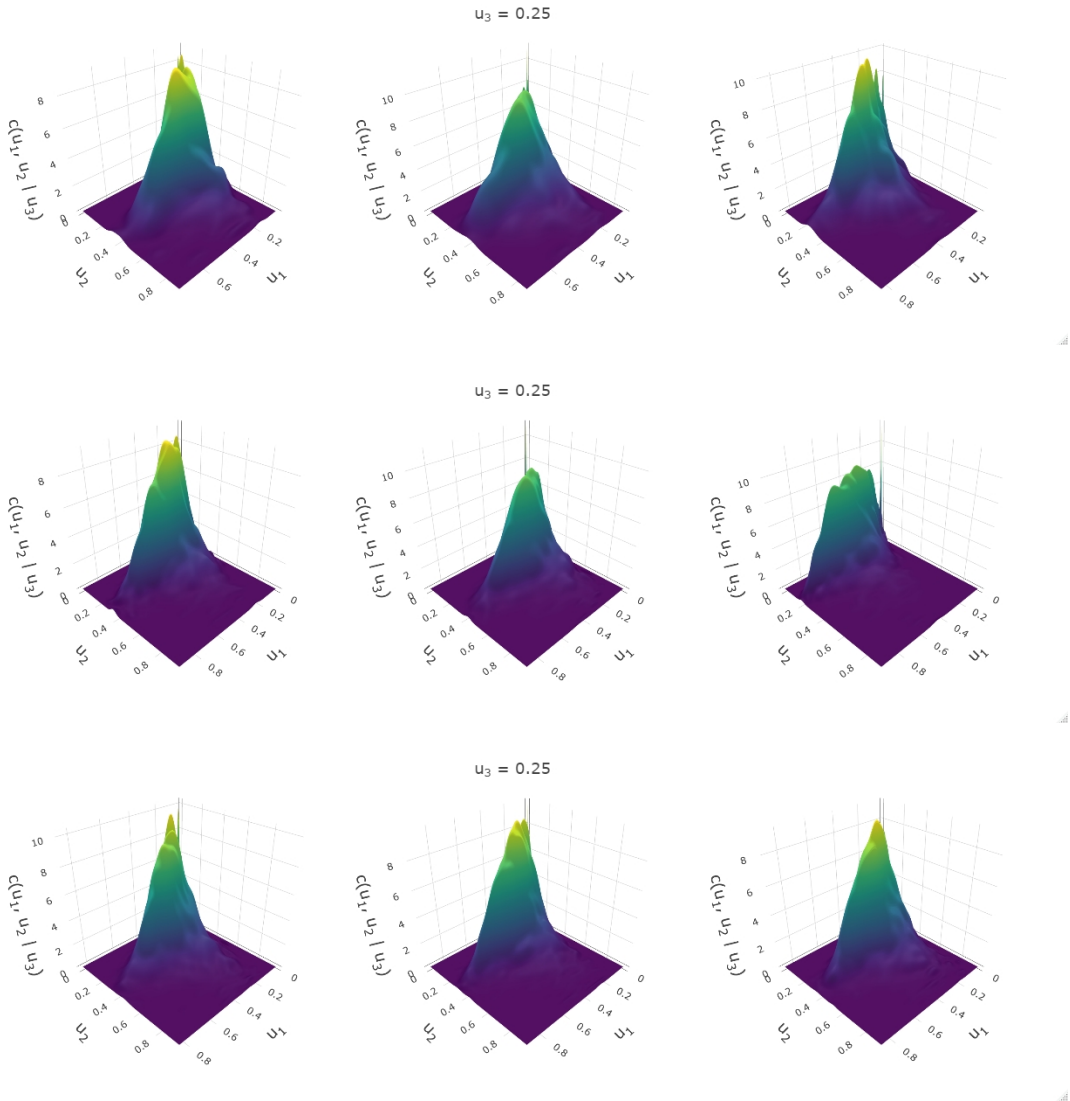

Figure 6.25.: Estimated densities with **medium conditioning value** for Simulation Setup 1. For the chosen vine specification, each column shows the plot of one chosen iteration. The rows correspond to different sample sizes with order $n = 1000$, 5000 and 10000.

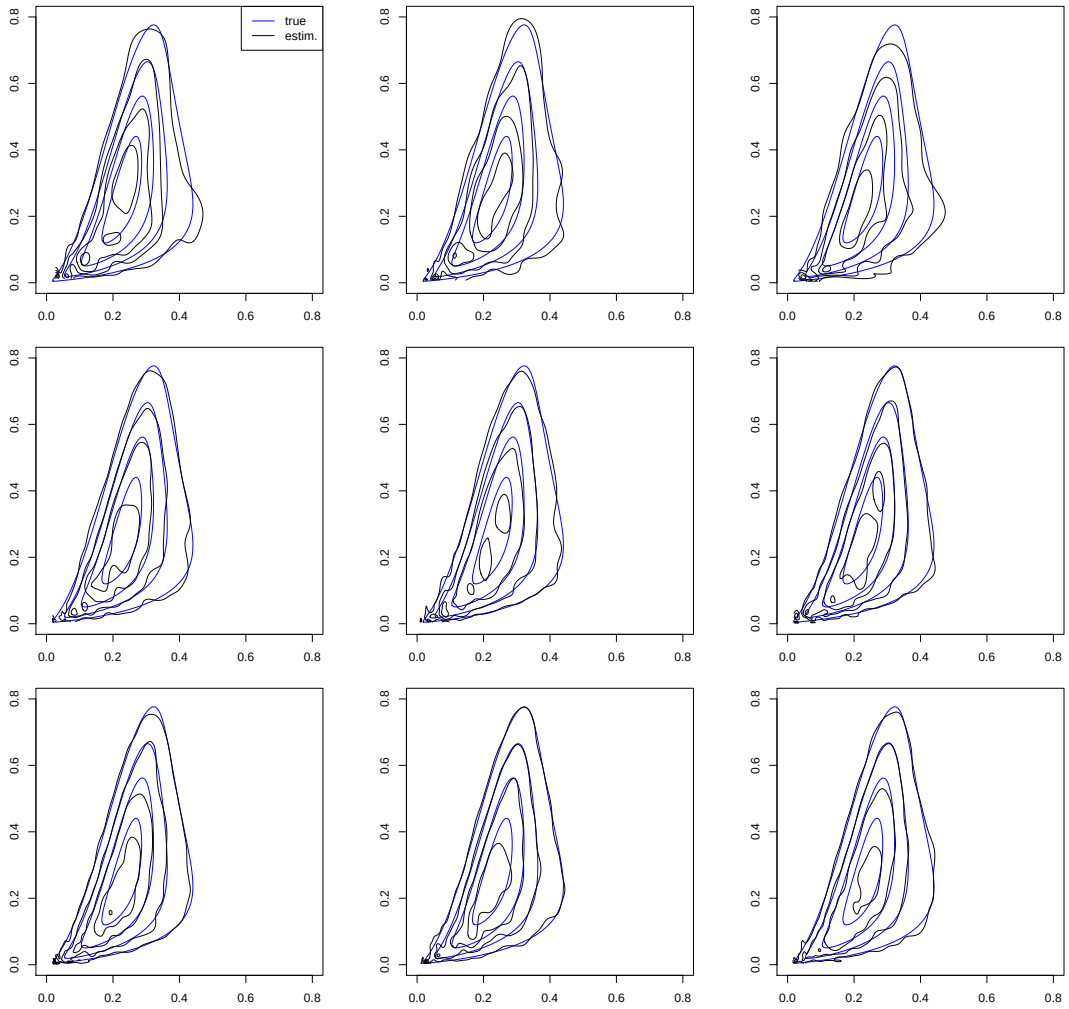

Figure 6.26.: Comparison of estimated densities and the true density with **medium conditioning value** for Simulation Setup 1. Each column shows the plot for one chosen iteration. The rows correspond to different sample sizes with order $n = 1000$, 5000 and 10000.

The true conditional density with **large** conditioning value is shown in Fig. 6.27, the estimated densities in Fig. 6.28 and the comparison of contour plots in Fig. 6.29.

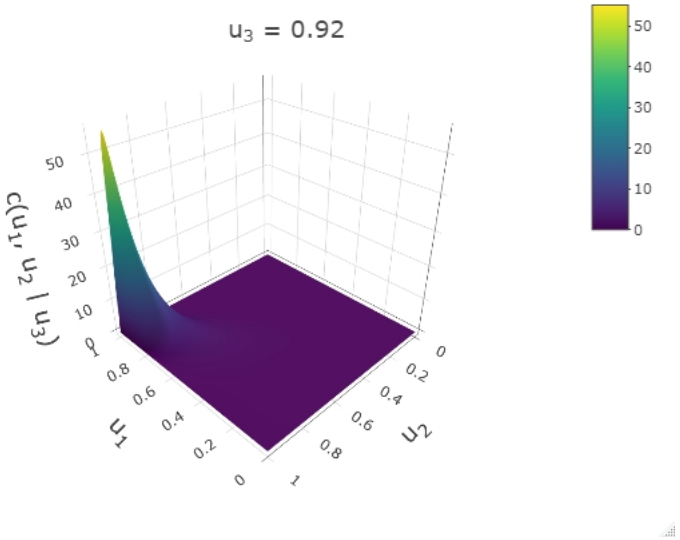

Figure 6.27.: True bivariate conditional density with **large conditioning value** in Simulation Setup 1.

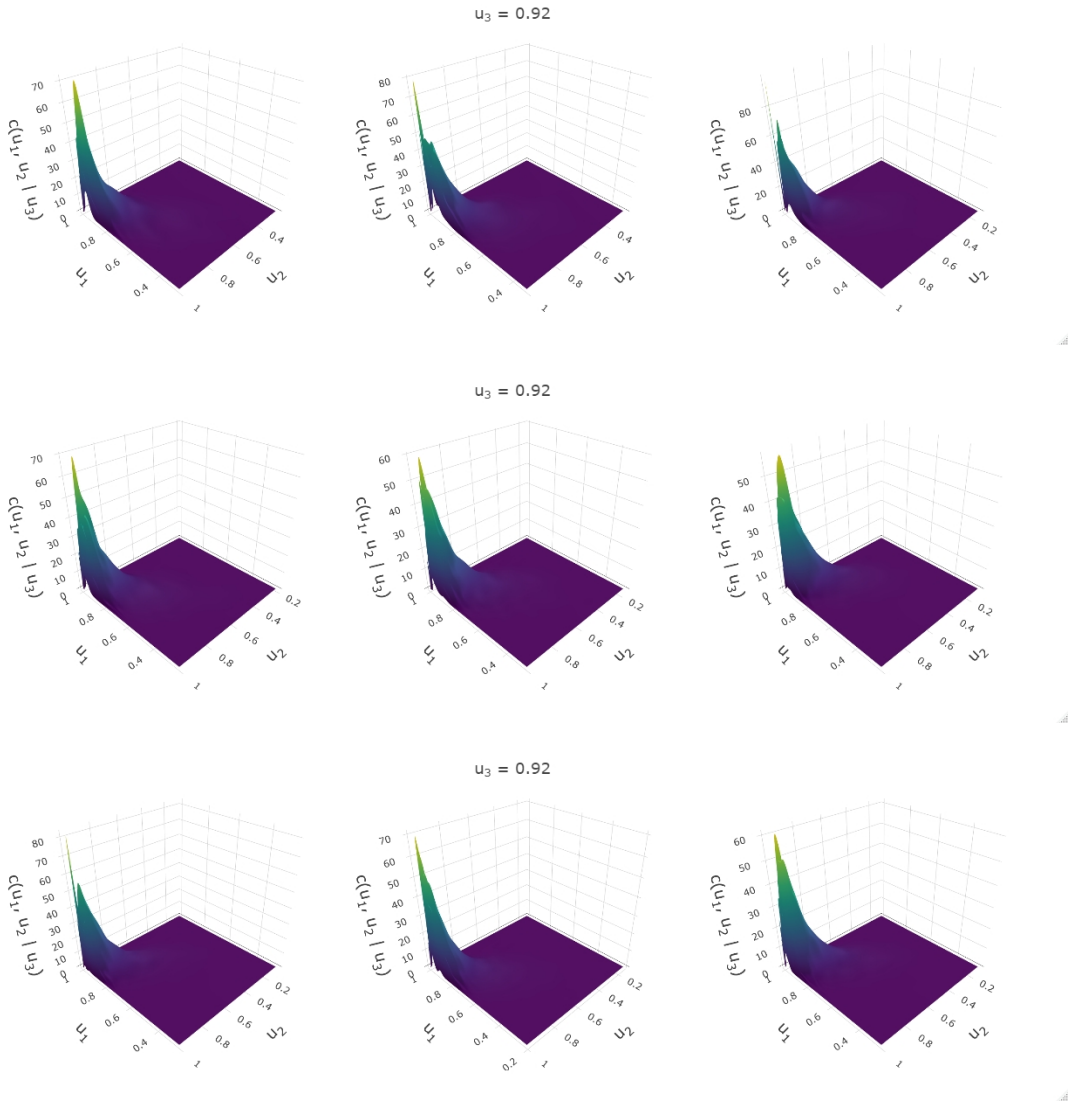

Figure 6.28.: Estimated densities with **large conditioning value** for Simulation Setup 1. For the chosen vine specification, each column shows the plot of one chosen iteration. The rows correspond to different sample sizes with order $n = 1000$, 5000 and 10000.

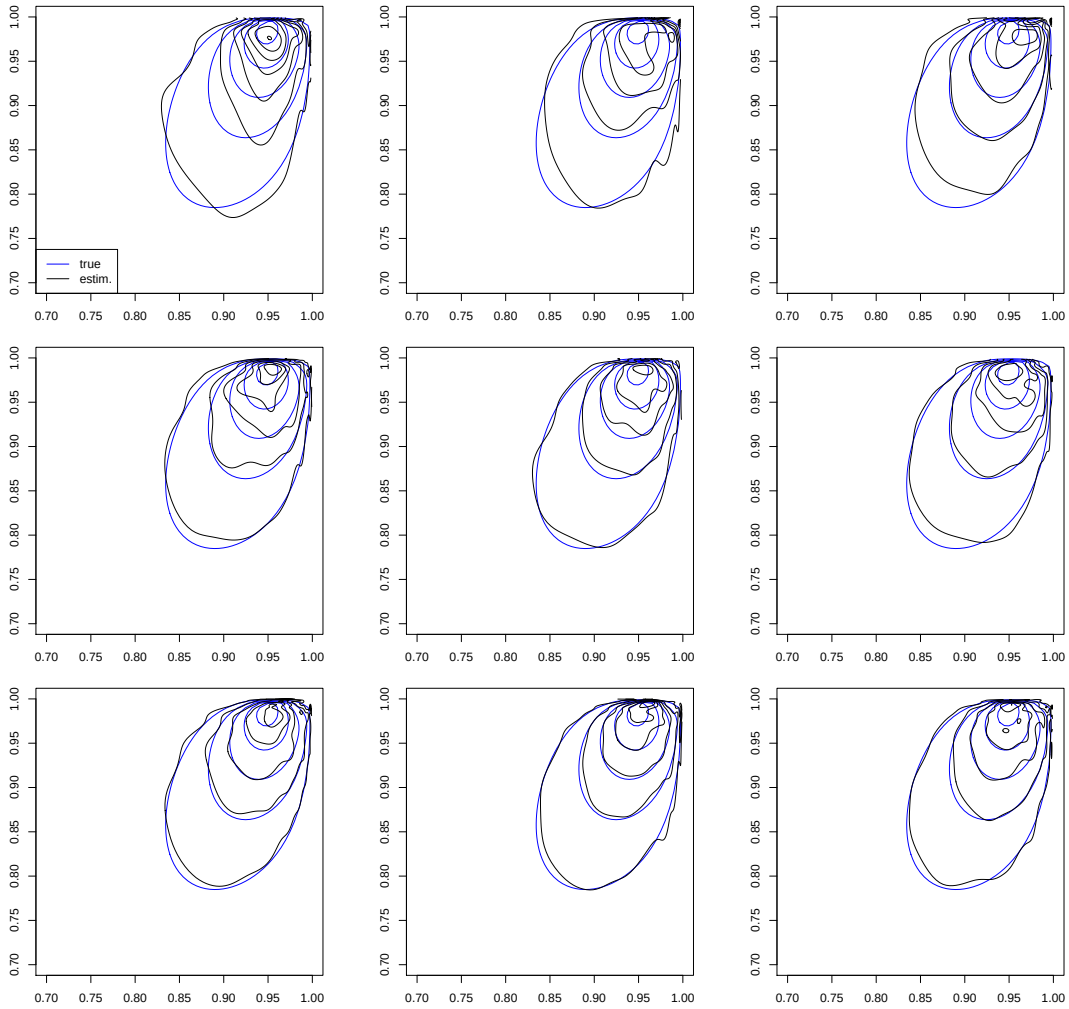

Figure 6.29.: Comparison of estimated densities and the true density with **large conditioning value** for Simulation Setup 1. Each column shows the plot for one chosen iteration. The rows correspond to different sample sizes with order $n = 1000$, 5000 and 10000.

The table of performance measure results for low, medium and large conditioning values are presented in Table 6.15. For every conditioning value and sample size, the percentage of iterations that would be rejected after the Bonferroni correction at 5% level is shown.

	Conditioning value		
	Low	Medium	Large
n=1000	3%	5%	5%
n=5000	3%	7%	6%
n=10000	6%	8%	5%

Table 6.15.: Table of results for bivariate Simulation Setup 1. The percentage of iterations that would be rejected after the Bonferroni correction at 5% level is shown for samples of sizes $n = 1000, 5000, 10000$, and low, medium and large conditioning values.

Setup 2: D-vine, d=5, conditional density expressed with integration

In this setup we use the vine tree structure from Fig. 6.2. We are sampling from the distribution of $(U_2, U_4|U_1 = u_1, U_3 = u_3, U_5 = u_5)$, however in this case the density is not available without numerical integration. The conditional density is

$$c_{24|135}(u_2, u_4|u_1, u_3, u_5) = \frac{c_{12345}}{c_{135}} = \frac{c_{12345}(u_1, u_2, u_3, u_4, u_5)}{\int_0^1 \int_0^1 c_{12345}(u_1, a_2, u_3, a_4, u_5) da_2 da_4},$$

and the Rosenblatt transformation is given as

$$\begin{aligned} v_2^m &= C(u_2|u_1, u_3, u_5) = \int_0^{u_2} c(a_2|u_1, u_3, u_5) da_2 = \int_0^{u_2} \frac{c(u_1, a_2, u_3, u_5)}{c(u_1, u_3, u_5)} da_2 = \\ &= \frac{\int_0^{u_2} \int_0^1 c(u_1, a_2, u_3, a_4, u_5) da_4 da_2}{\int_0^1 \int_0^1 c(u_1, a_2, u_3, a_4, u_5) da_2 da_4}, \\ v_4^c &= C(u_4|u_1, u_2, u_3, u_5) = \int_0^{u_4} c(a_4|u_1, u_2, u_3, u_5) da_4 = \\ &= \int_0^{u_4} \frac{c(u_1, u_2, u_3, a_4, u_5)}{c(u_1, u_2, u_3, u_5)} da_4 = \frac{\int_0^{u_4} c(u_1, u_2, u_3, a_4, u_5) da_4}{\int_0^1 c(u_1, u_2, u_3, a_4, u_5) da_4} \end{aligned}$$

and

$$\begin{aligned} v_4^m &= C(u_4|u_1, u_3, u_5) = \int_0^{u_4} c(a_4|u_1, u_3, u_5) da_4 = \int_0^{u_4} \frac{c(u_1, u_3, a_4, u_5)}{c(u_1, u_3, u_5)} da_4 = \\ &= \frac{\int_0^{u_4} \int_0^1 c(u_1, a_2, u_3, a_4, u_5) da_2 da_4}{\int_0^1 \int_0^1 c(u_1, a_2, u_3, a_4, u_5) da_2 da_4}, \end{aligned}$$

$$v_2^c = C(u_2|u_1,u_3,u_4,u_5) = \int_0^{u_2} c(a_2|u_1,u_3,u_4,u_5)da_2 =$$
$$= \int_0^{u_2} \frac{c(u_1,a_2,u_3,u_4,u_5)}{c(u_1,u_3,u_4,u_5)}da_2 = \frac{\int_0^{u_2} c(u_1,a_2,u_3,u_4,u_5)da_2}{\int_0^1 c(u_1,a_2,u_3,u_4,u_5)da_2}.$$

In this setup we look at one D-vine specification. Its chosen pair copula families and parameters can be seen in Fig. 6.30.

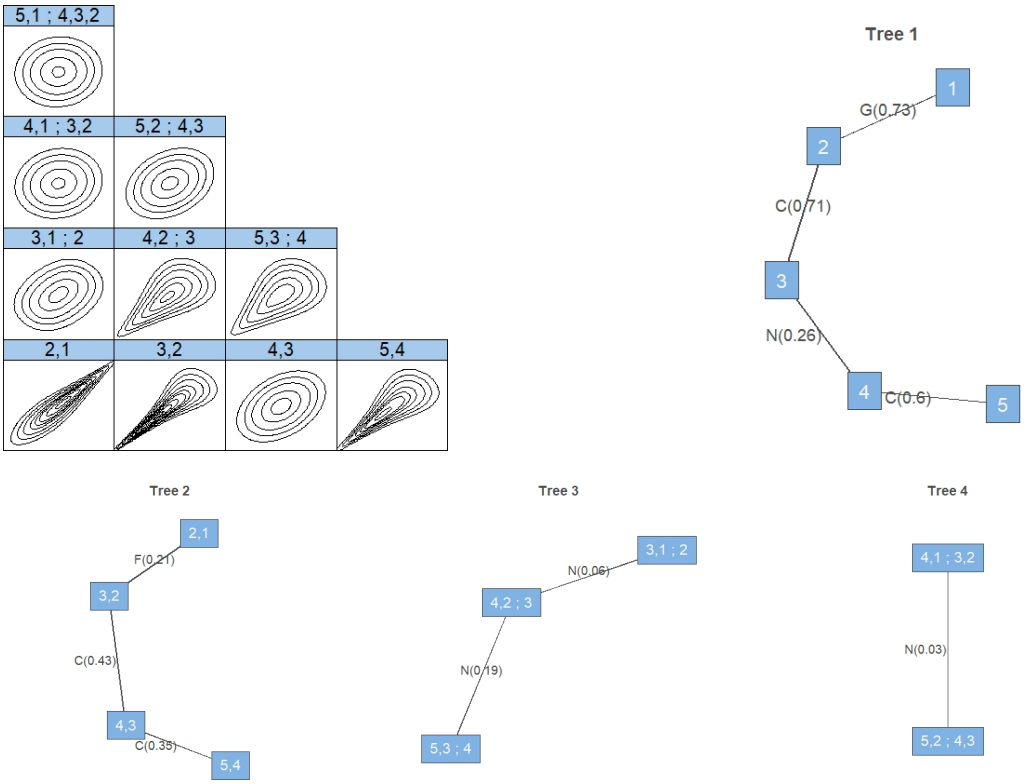

Figure 6.30.: Pair copula families and contour plots of the chosen D-vine specification in the bivariate Simulation Setup 2. First panel depicts the pair copula contour plots on the z-scale, and the remaining panels the D-vine tree structure with the copula families and corresponding Kendall's τ parameter.

The true conditional density with **low** conditioning values is shown in Fig. 6.31, the estimated densities in Fig. 6.32 and the comparison of contour plots in Fig. 6.33.

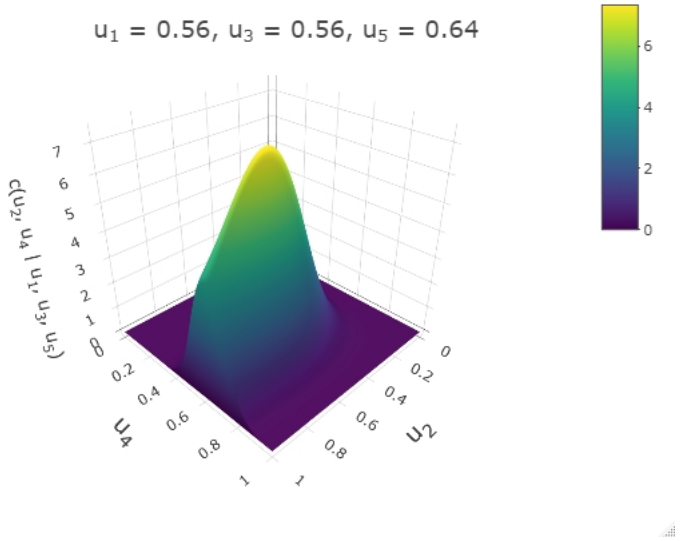

Figure 6.31.: True bivariate conditional density with **low conditioning values** in Simulation Setup 2.

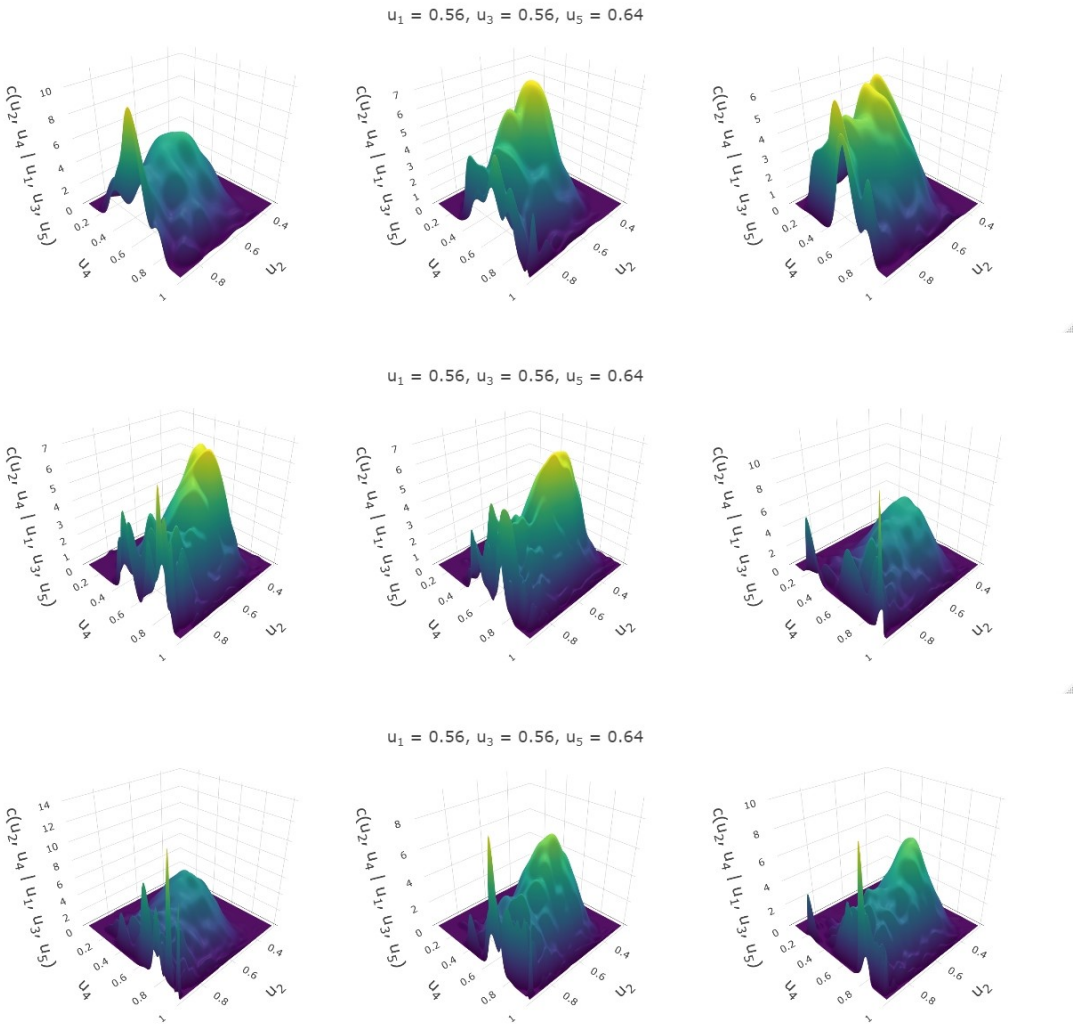

Figure 6.32.: Estimated densities with **low conditioning values** for Simulation Setup 2. For the chosen vine specification, each column shows the plot of one chosen iteration. The rows correspond to different sample sizes with order $n = 1000$, 5000 and 10000.

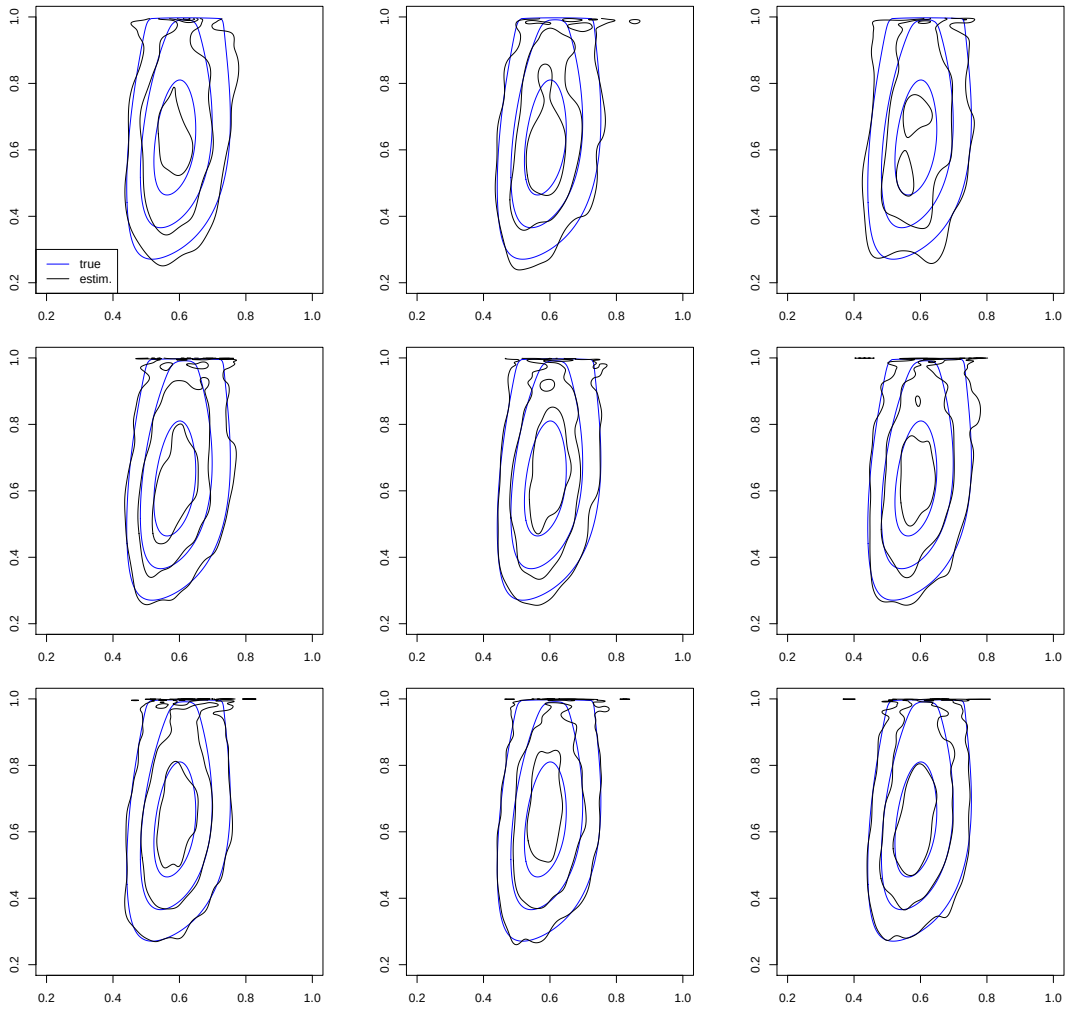

Figure 6.33.: Comparison of estimated densities and the true density with **low conditioning values** for Simulation Setup 2. Each column shows the plot for one chosen iteration. The rows correspond to different sample sizes with order $n = 1000$, 5000 and 10000.

The true conditional density with **medium** conditioning values is shown in Fig. 6.34, the estimated densities in Fig. 6.35 and the comparison of contour plots in Fig. 6.36.

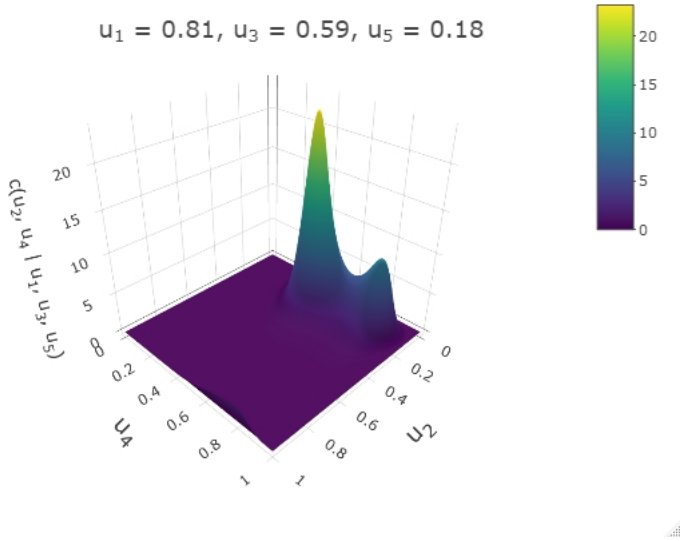

Figure 6.34.: True bivariate conditional density with **medium conditioning values** in Simulation Setup 2.

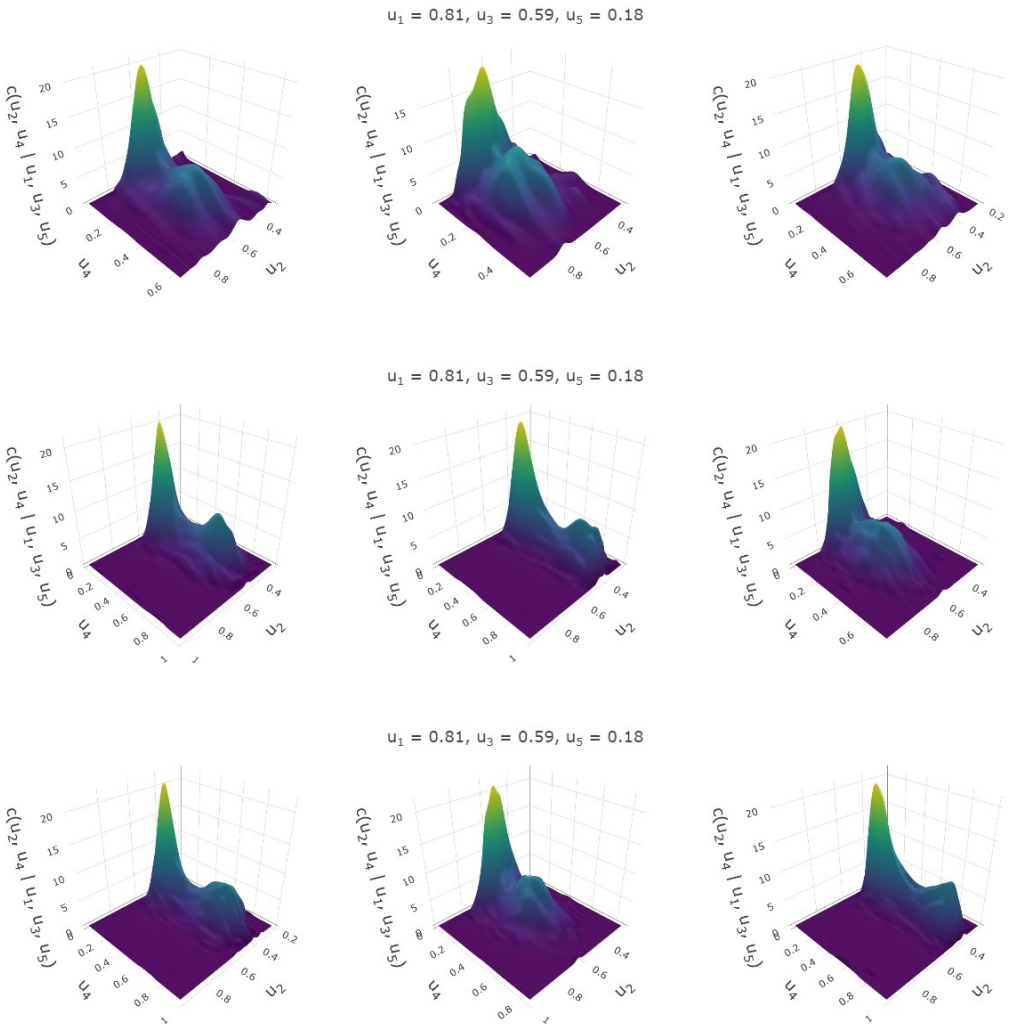

Figure 6.35.: Estimated densities with **medium conditioning values** for Simulation Setup 2. For the chosen vine specification, each column shows the plot of one chosen iteration. The rows correspond to different sample sizes with order $n = 1000$, 5000 and 10000.

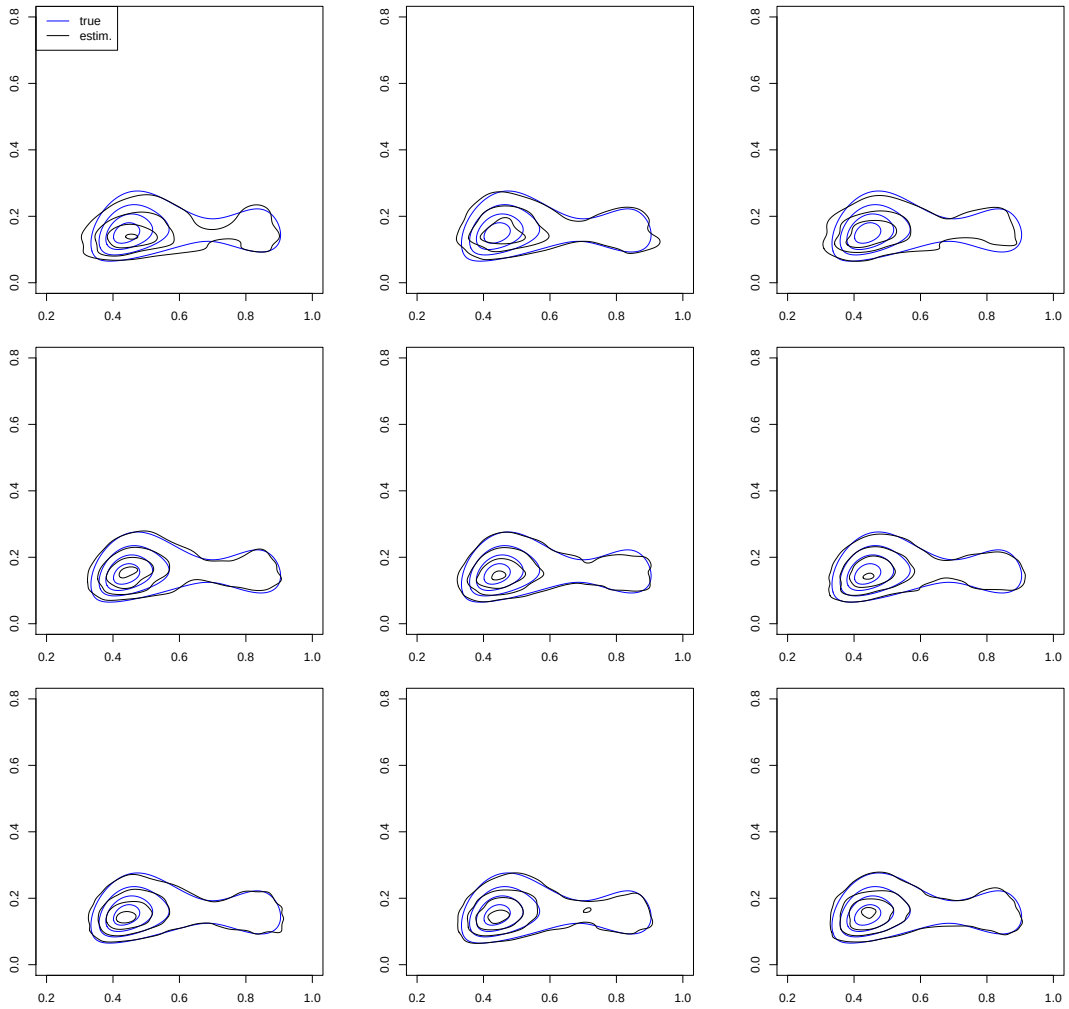

Figure 6.36.: Comparison of estimated densities and the true density with **medium conditioning values** for Simulation Setup 2. Each column shows the plot for one chosen iteration. The rows correspond to different sample sizes with order $n = 1000$, 5000 and 10000.

The true conditional density with **large** conditioning values is shown in Fig. 6.37, the estimated densities in Fig. 6.38 and the comparison of contour plots in Fig. 6.39.

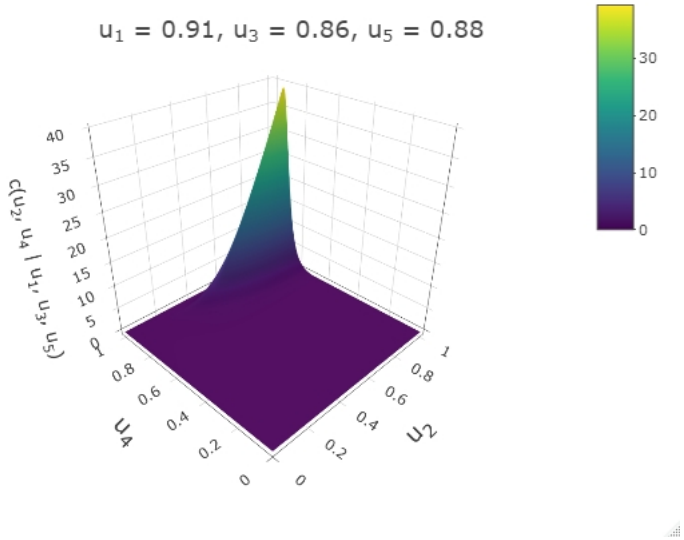

Figure 6.37.: True bivariate conditional density with **large conditioning values** in Simulation Setup 2.

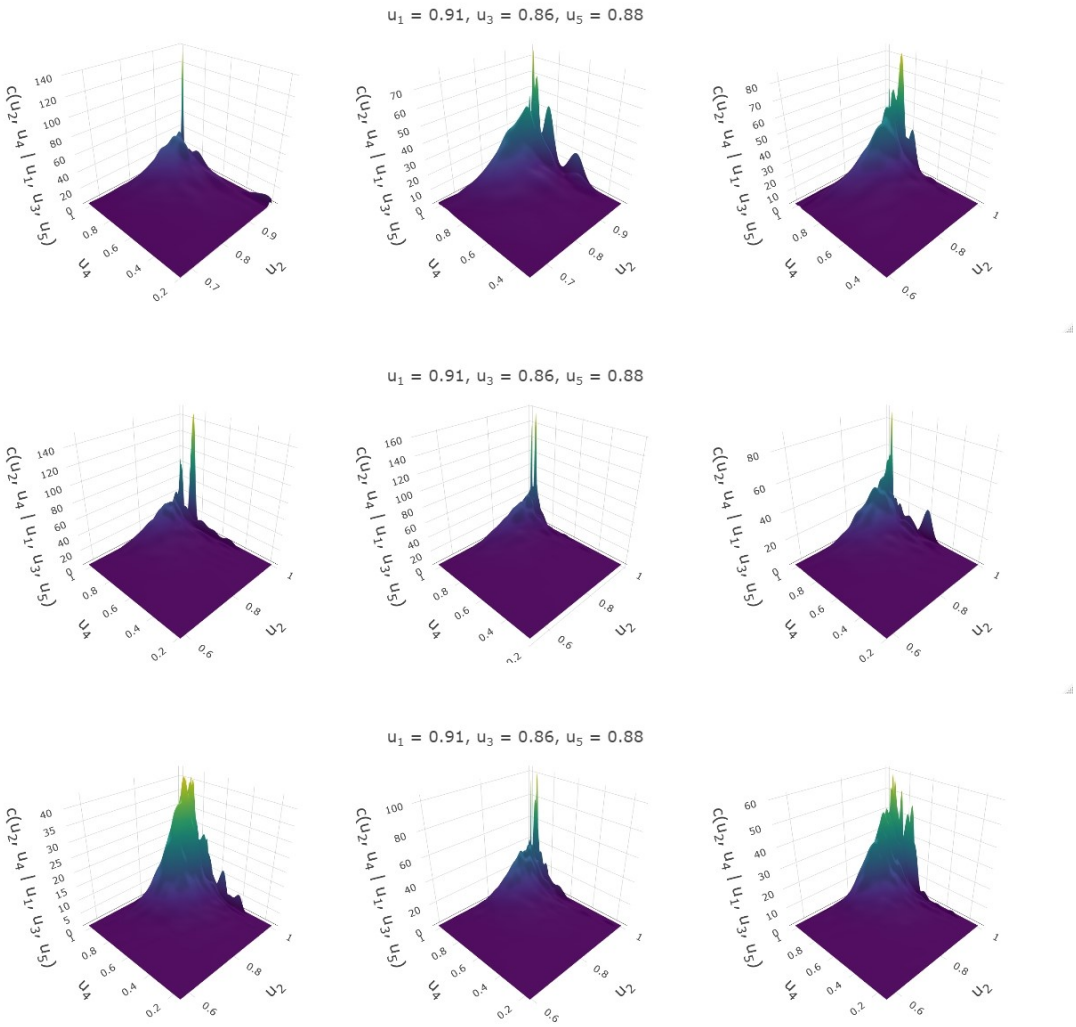

Figure 6.38.: Estimated densities with **large conditioning values** for Simulation Setup 2. For the chosen vine specification, each column shows the plot of one chosen iteration. The rows correspond to different sample sizes with order $n = 1000$, 5000 and 10000.

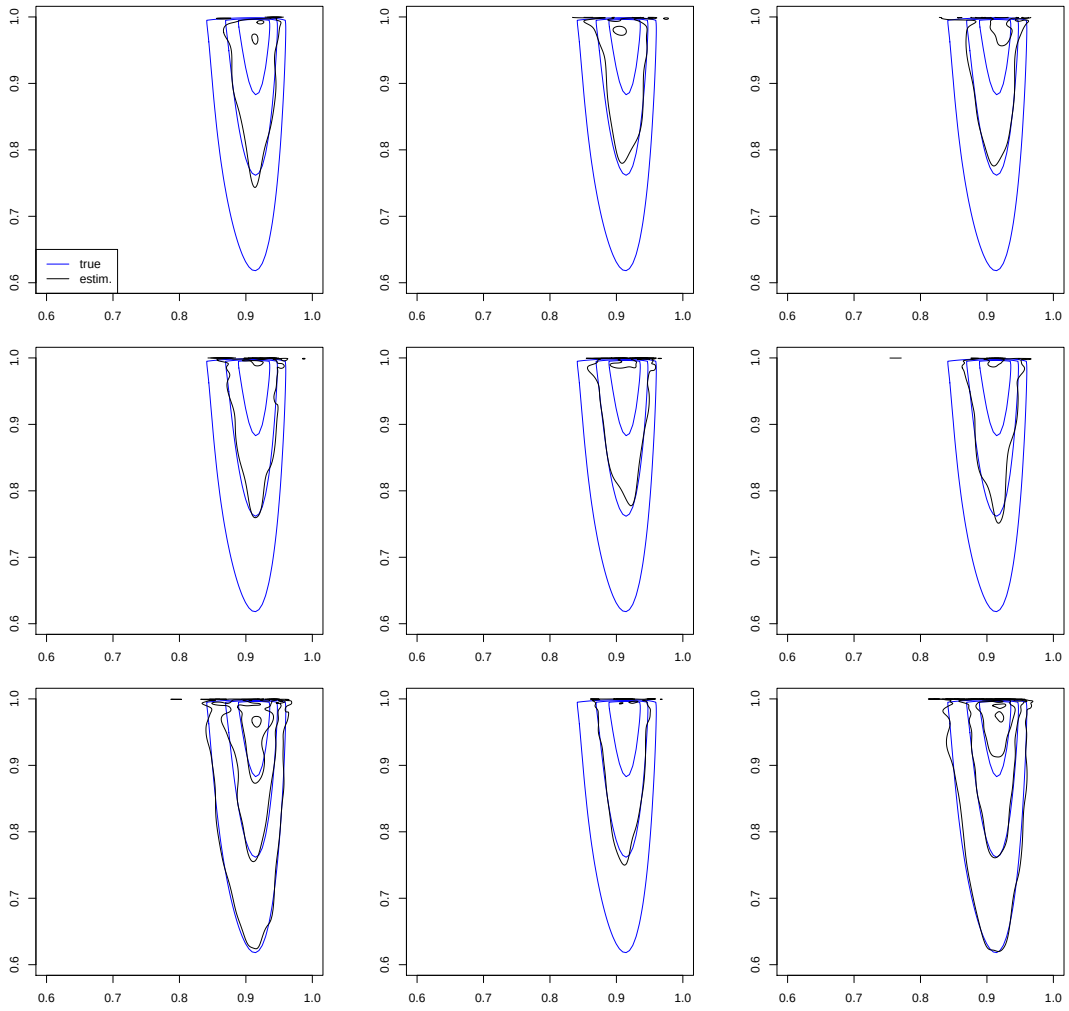

Figure 6.39.: Comparison of estimated densities and the true density with **large conditioning values** for Simulation Setup 2. Each column shows the plot for one chosen iteration. The rows correspond to different sample sizes with order $n = 1000$, 5000 and 10000.

The table of results for low, medium and large conditioning values are presented together in Table 6.16.

	Conditioning value		
	Low	Medium	Large
n=1000	2%	3 %	1%
n=5000	4%	2 %	3%
n=10000	3%	4%	2%

Table 6.16.: Table of results for bivariate Simulation Setup 2. The percentage of iterations that would be rejected after the Bonferroni correction at 5% level is shown for samples of sizes $n = 1000, 5000, 10000$, and low, medium and large conditioning values.

Setup 3: R-vine, d=5, conditional density expressed with integration

In this setup we use the vine tree structure from Fig. 6.3. We are sampling from the distribution of $(U_2, U_4|U_1 = u_1, U_3 = u_3, U_5 = u_5)$, in this case the density is again not available without numerical integration. The conditional density and the Rosenblatt transformation are expressed the same way as in the Setup 2.

Here we look at one R-vine specification. Its chosen pair copula families and parameters can be seen in Fig. 6.40.

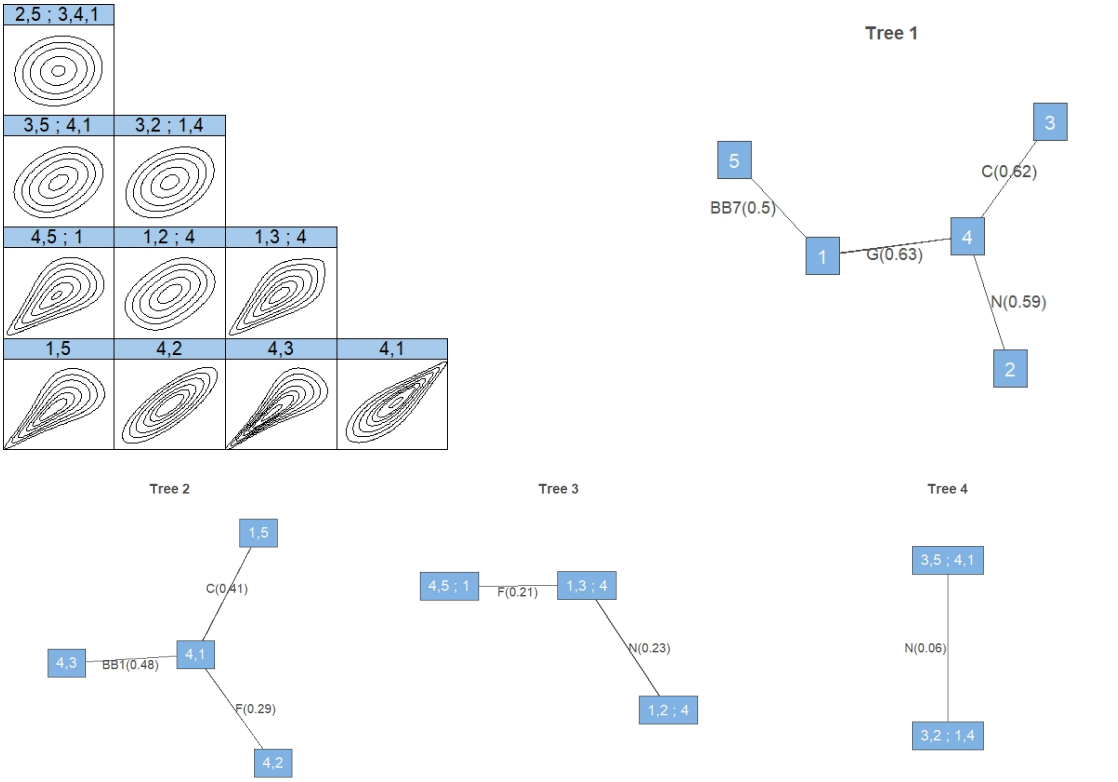

Figure 6.40.: Pair copula families and contour plots of the chosen R-vine specification in the bivariate Simulation Setup 3. First panel depicts the pair copula contour plots on the z-scale, and the remaining panels the R-vine tree structure with the copula families and corresponding Kendall's τ parameter.

The true conditional density with **low** conditioning values is shown in Fig. 6.41, the estimated densities in Fig. 6.42 and the comparison of contour plots in Fig. 6.43.

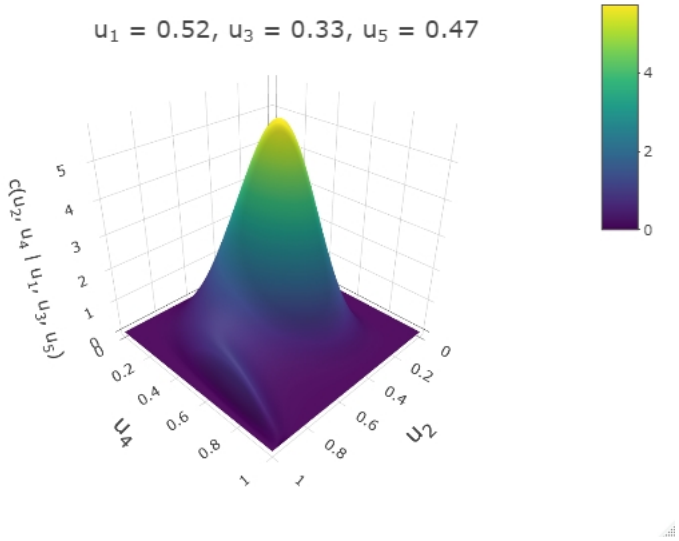

Figure 6.41.: True bivariate conditional density with **low conditioning values** in Simulation Setup 3.

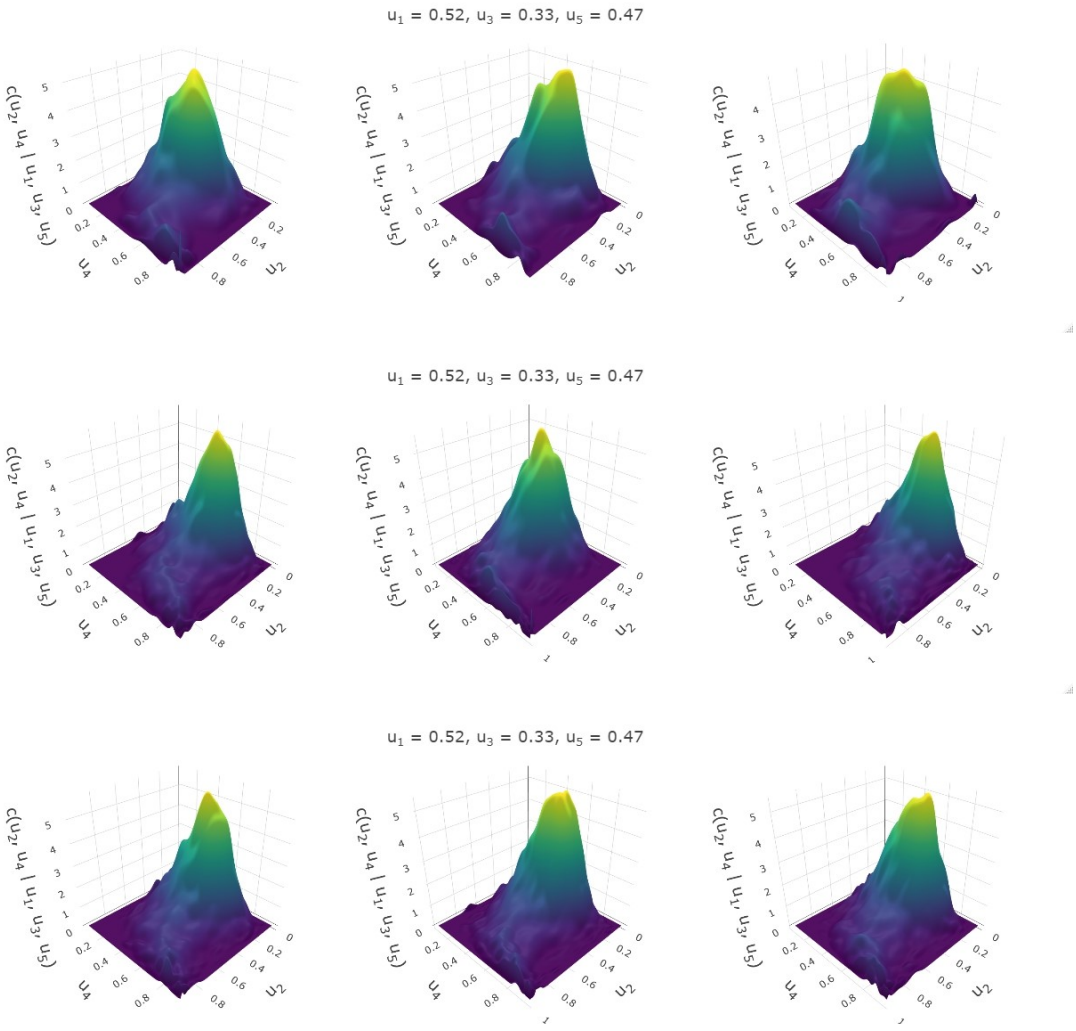

Figure 6.42.: Estimated densities with **low conditioning values** for Simulation Setup 3. For the chosen vine specification, each column shows the plot of one chosen iteration. The rows correspond to different sample sizes with order $n = 1000$, 5000 and 10000.

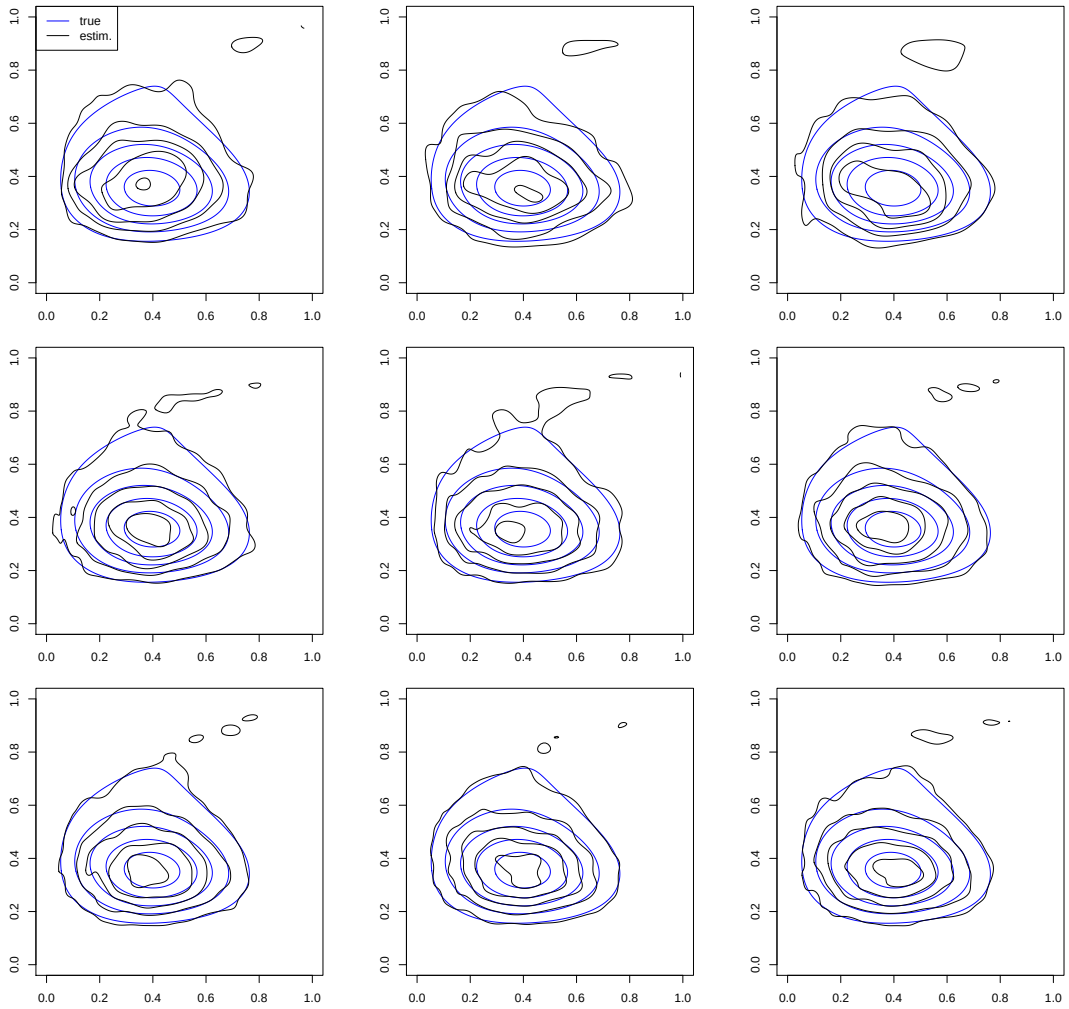

Figure 6.43.: Comparison of estimated densities and the true density with **low conditioning values** for Simulation Setup 3. Each column shows the plot for one chosen iteration. The rows correspond to different sample sizes with order $n = 1000$, 5000 and 10000.

The true conditional density with **medium** conditioning values is shown in Fig. 6.44, the estimated densities in Fig. 6.45 and the comparison of contour plots in Fig. 6.46.

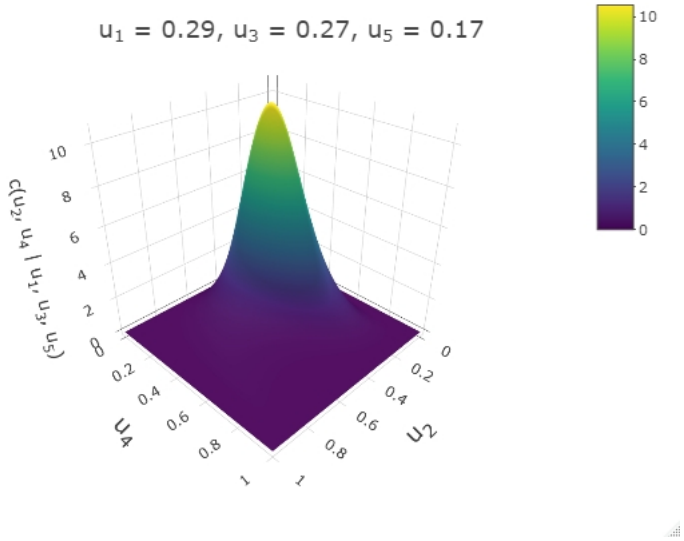

Figure 6.44.: True bivariate conditional density with **medium conditioning values** in Simulation Setup 3.

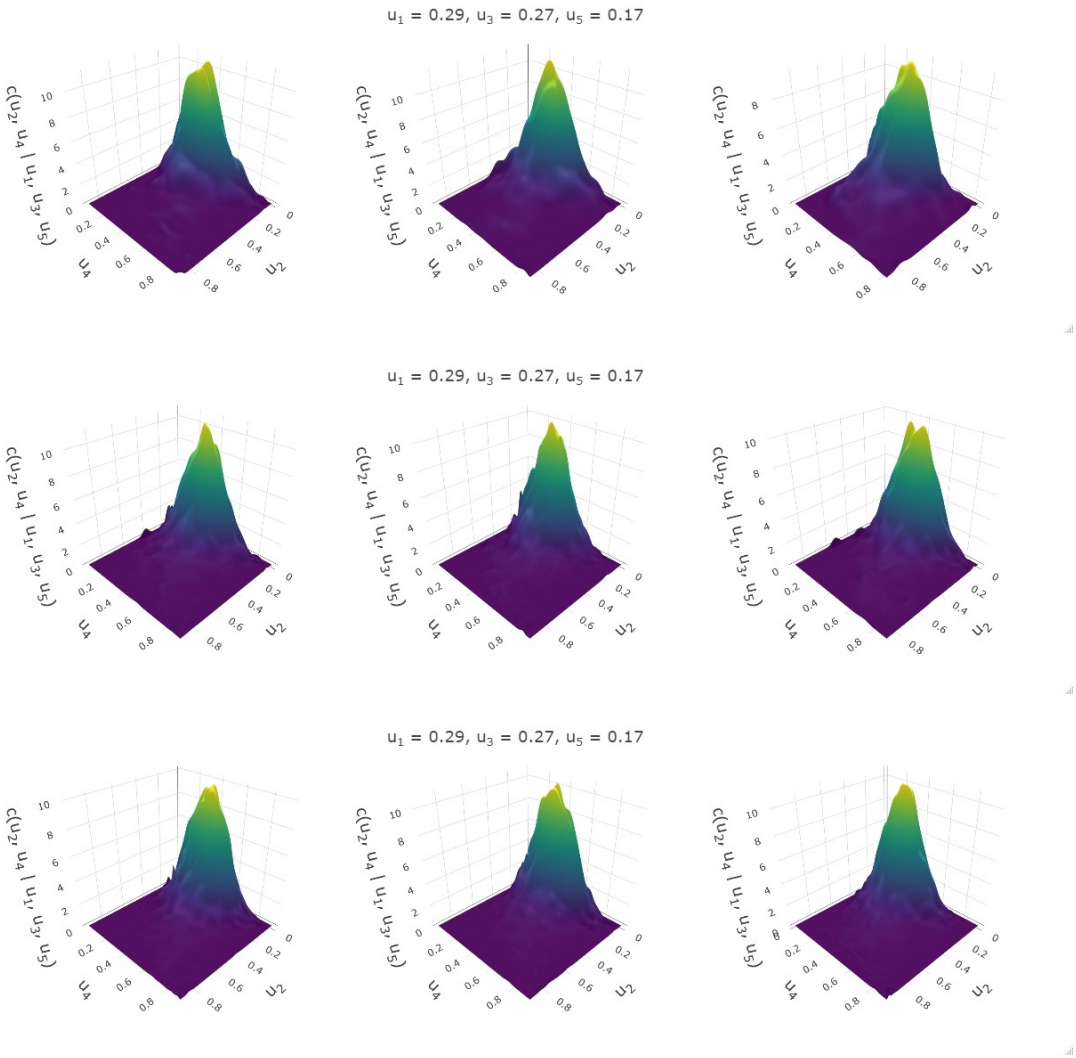

Figure 6.45.: Estimated densities with **medium conditioning values** for Simulation Setup 3. For the chosen vine specification, each column shows the plot of one chosen iteration. The rows correspond to different sample sizes with order $n = 1000$, 5000 and 10000.

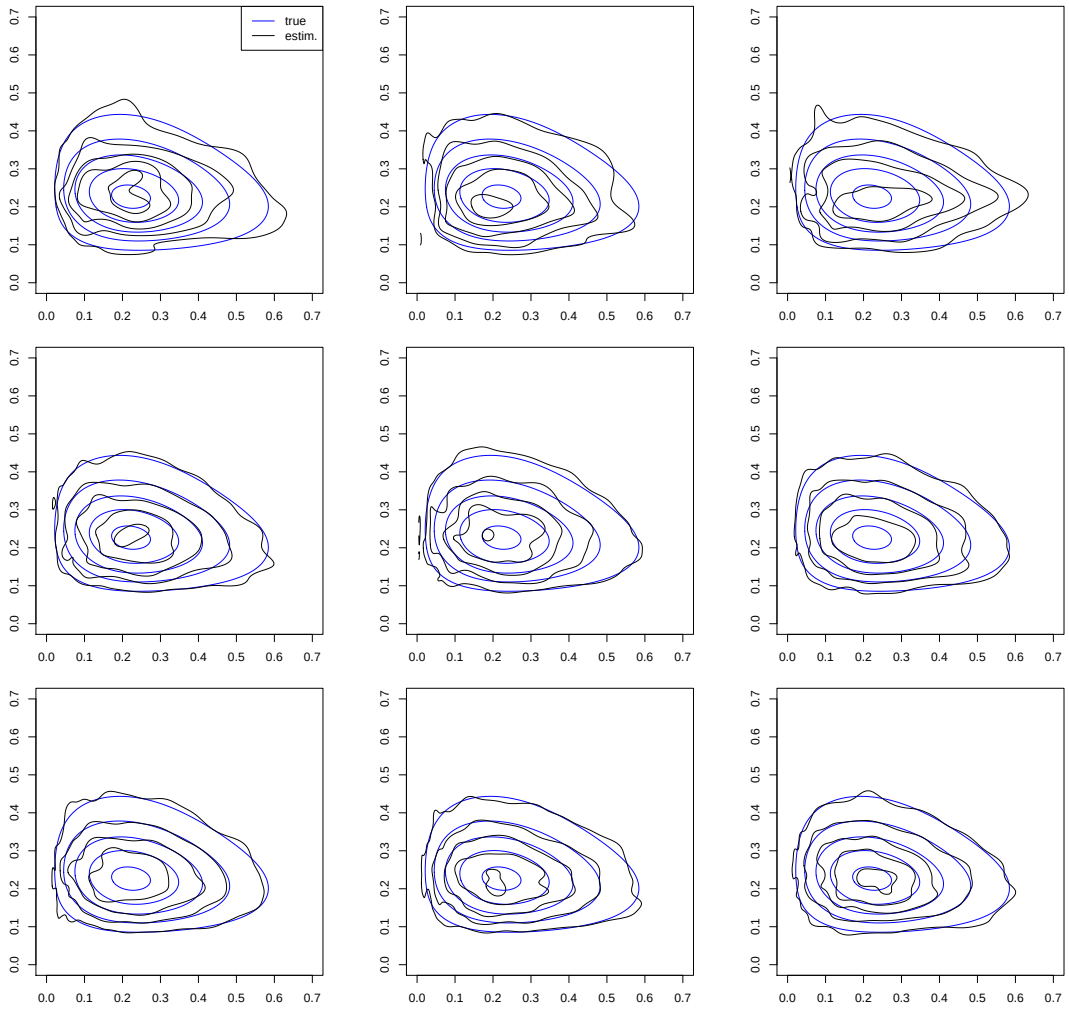

Figure 6.46.: Comparison of estimated densities and the true density with **medium conditioning values** for Simulation Setup 3. Each column shows the plot for one chosen iteration. The rows correspond to different sample sizes with order $n = 1000$, 5000 and 10000.

The true conditional density with **large** conditioning values is shown in Fig. 6.47, the estimated densities in Fig. 6.48 and the comparison of contour plots in Fig. 6.49.

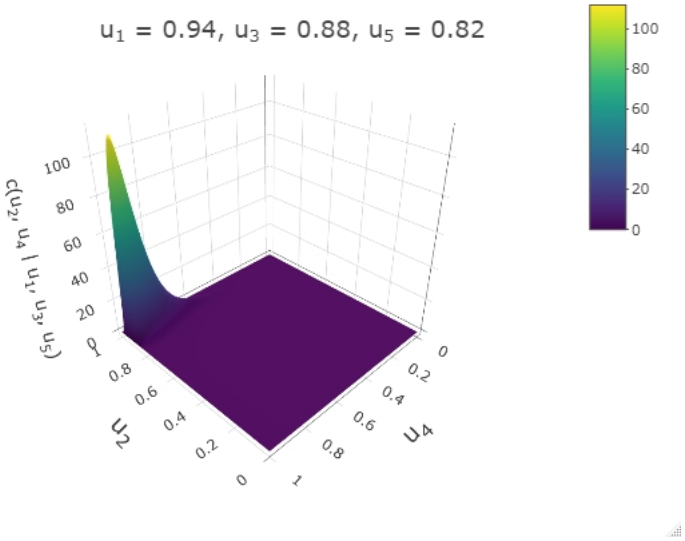

Figure 6.47.: True bivariate conditional density with **large conditioning values** in Simulation Setup 3.

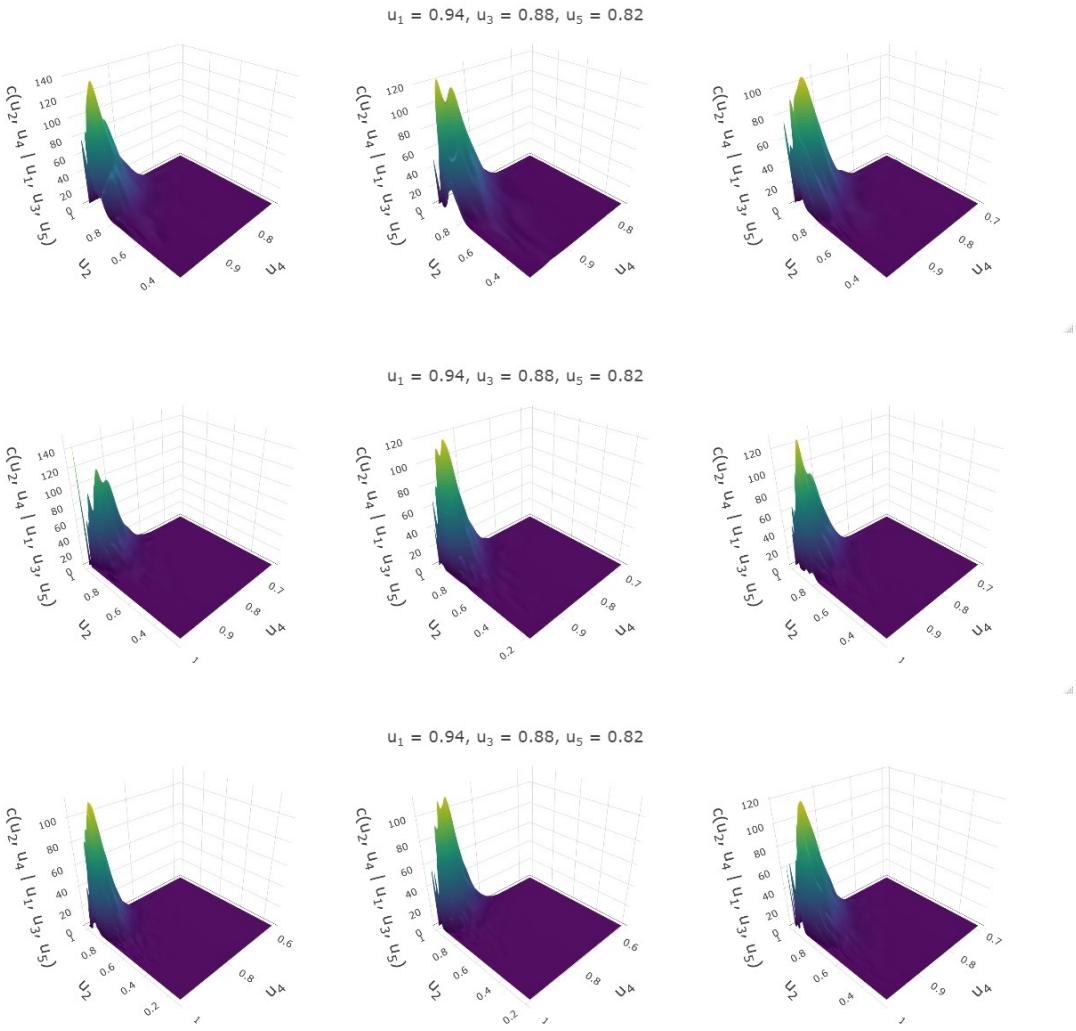

Figure 6.48.: Estimated densities with **large conditioning values** for Simulation Setup 3. For the chosen vine specification, each column shows the plot of one chosen iteration. The rows correspond to different sample sizes with order $n = 1000$, 5000 and 10000.

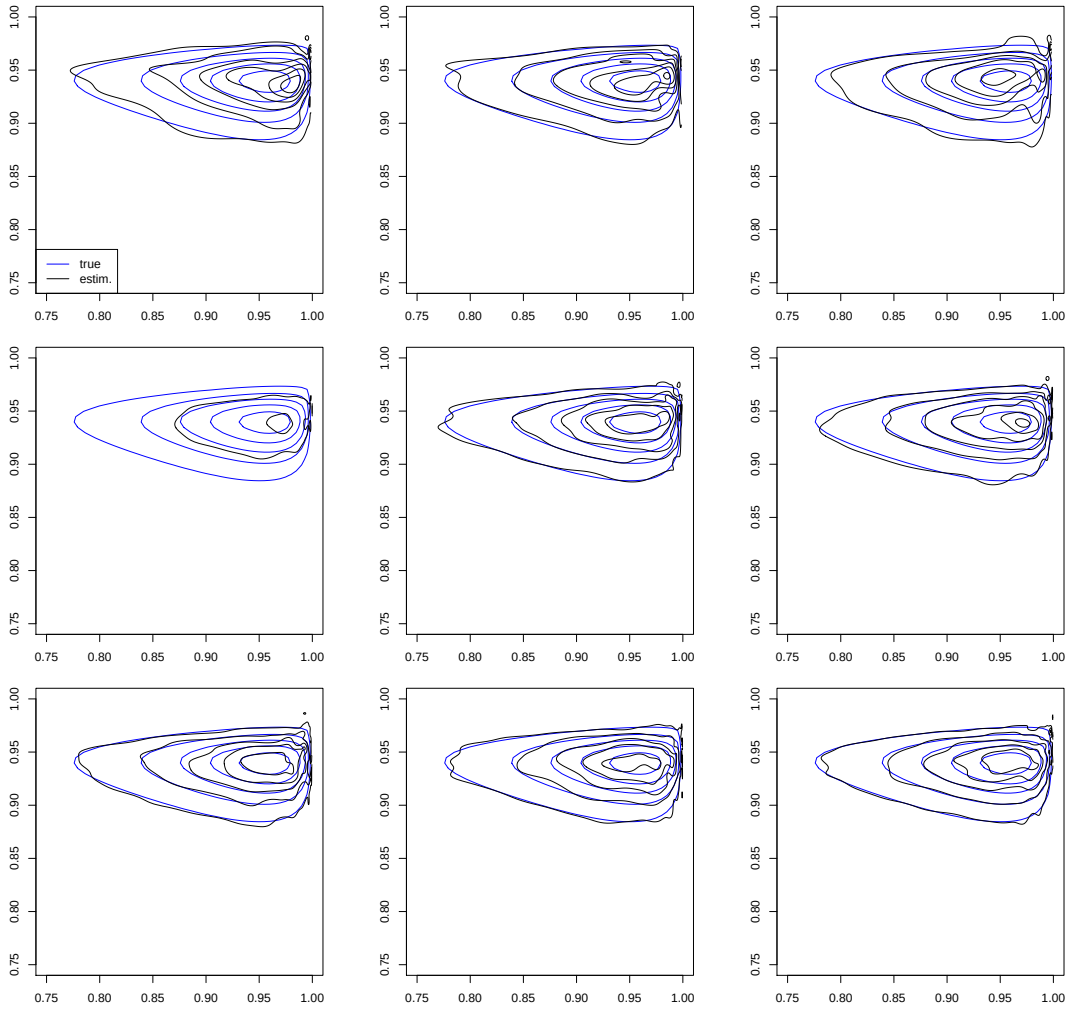

Figure 6.49.: Comparison of estimated densities and the true density with **large conditioning values** for Simulation Setup 3. Each column shows the plot for one chosen iteration. The rows correspond to different sample sizes with order $n = 1000$, 5000 and 10000.

The table of results for low, medium and large conditioning values are presented together in Table 6.17.

	Conditioning value		
	Low	Medium	Large
n=1000	2%	0%	1%
n=5000	3%	4%	5%
n=10000	5%	4%	3%

Table 6.17.: Table of results for bivariate Simulation Setup 3. The percentage of iterations that would be rejected after the Bonferroni correction at 5% level is shown for samples of sizes $n = 1000, 5000, 10000$, and low, medium and large conditioning values.

7. Application on Uranium Data Set

In this chapter, we are going to apply our proposed sampler for the analysis of a hydrogeochemical data set, which consists of $N = 655$ observations of log-concentrations of the seven chemicals **Uranium (U)**, **Lithium (Li)**, **Cobalt (Co)**, **Potassium (K)**, **Cesium (Cs)**, **Scandium (Sc)**, and **Titanium (Ti)** taken from river near Grand Junction, Colorado. The data set can be found in the `R` package `copula` (Hofert et al., 2020).

In order to assess the extent of uranium potential in the United States, the hydrogeochemical stream and sediment reconnaissance (HSSR) project has been created in the U.S Department of Energy, as stated in Cook and Johnson (1981). Water samples over a fairly fine grid (one per 4 km^2 in the contiguous states and one per 10 km^2 in Alaska) were collected and analysed to determine the concentrations of various chemicals. Since the data set consists of only samples taken from the quadrangle in Colorado, it is only a small subset of the whole data collected in the program. In order to get help with the analysis, the program funded the works Cook and Johnson (1981) and Cook and Johnson (1986), in which the data set is first mentioned and analysed. Their main interest was to find replacement to multivariate normal distributions when modelling multivariate data, since some data does not have to have either normal margins or the dependence between the correlated variables may deviate from the one in normal distributions. This was also the case for the uranium data set, even though logarithms of the concentrations were used to improve joint normality. The examination of scatter plots in the uranium data set suggested that the underlying contours are not elliptically symmetric, therefore the assumption of joint normality may not be appropriate and other joint distributions should be considered.

First, we look at the subset of the seven chemicals, namely the three chemicals Cobalt (Co), Scandium (Sc), Titanium (Ti) and use univariate and bivariate sampling. Later we analyze the whole set with all seven chemicals.

In order to use our proposed sampler, we need to transform the variables from x-scale to the u-scale, sample with our program and transform it back to the x-scale.

For this we use the expression of the conditional quantile function from Kraus and Czado (2017b). The conditional quantile function for $\alpha \in (0,1)$ is

$$F^{-1}_{Y|X_1,\dots,X_d}(\alpha|x_1,\dots,x_d) = F^{-1}_Y(C^{-1}_{V|U_1,\dots,U_d}(\alpha|u_1,\dots,u_d)), \tag{7.1}$$

where $Y \sim F_Y$ and $X_j \sim F_j$, $j = 1,\dots,d$ and the probability integral transform $V := F_Y(Y)$ and $U_j := F_j(X_j)$ for $j = 1,\dots,d$ is used.

Imagine we have variables X_1, X_2, X_3 and fit a vine copula model to it. We are interested in obtaining a sample $x_1^{(r)}(x_2,x_3) \sim F_{1|23}(\cdot|X_2 = x_2, X_3 = x_3)$. We use the probability integral transform $U_j = F_j(X_j)$, $j = 2,3$ to come to the u-scale. We sample with our proposed program from $U_1|U_2,U_3$ and obtain $u_1^{(r)}(u_2,u_3) \sim C_{1|23}(\cdot|U_2 = u_2, U_3 = u_3)$. We can obtain $x_1^{(r)}(x_2,x_3)$ using Eq. (7.1) as follows

$$x_1^{(r)}(x_2,x_3) = F^{-1}_{1|23}(v_1^{(r)}|x_2,x_3) = F_1^{-1}(C^{-1}_{1|23}(v_1^{(r)}|u_2,u_3)) = F_1^{-1}(u_1^{(r)}(u_2,u_3)),$$

where $v_1^{(r)} = C_{1|23}(u_1^{(r)}(u_2,u_3)|u_2,u_3) \sim U[0,1]$.

For the bivariate case we use the inverse Rosenblatt transformation. Now we are interested in obtaining $x_1^{(r)}, x_2^{(r)}|x_3 \sim F_{12|3}(\cdot,\cdot|X_3 = x_3)$. We use probability integral transform $U_j = F_j(X_j)$, $j = 3$ to transfer the conditioning value x_3 to the u-scale. We sample with our proposed program from $U_1,U_2|U_3$ and obtain $(u_1^{(r)}, u_2^{(r)}|u_3) \sim C_{12|3}(\cdot,\cdot|U_3 = u_3)$. We can obtain $(x_1^{(r)}, x_2^{(r)}|x_3)$ using the Eq. (7.1) and the inverse Rosenblatt as follows

$$\begin{aligned} x_2^{(r)} &= F^{-1}_{2|3}(v_2^{(r)}|x_3) = F_2^{-1}(C^{-1}_{2|3}(v_2^{(r)}|u_3)) = F_2^{-1}(u_2^{(r)}),\\ x_1^{(r)} &= F^{-1}_{1|23}(v_1^{(r)}|x_2^{(r)},x_3) = F_1^{-1}(C^{-1}_{1|23}(v_1^{(r)}|u_2^{(r)},u_3)) = F_1^{-1}(u_1^{(r)}), \end{aligned}$$

where $(v_1^{(r)}, v_2^{(r)}) \sim U[0,1]^2$, since $v_2^{(r)} = C_{2|3}(u_2^{(r)}|u_3)$ and $v_1^{(r)} = C_{1|23}(u_1^{(r)}|u_2^{(r)},u_3)$.

7.1. Three Dimensional Analysis

First we start with only three chemicals Cobalt (Co), Scandium (Sc) and Titanium (Ti). In Table 7.1 we can see the summary of the three variables. After transforming the data to the copula scale by applying the probability integral transform using the probability distribution function of kernel density estimation, we show the pairwise scatter plot in Fig. 7.1.

	Co	Sc	Ti
min	0.568	0.322	2.847
5% quantile	0.820	0.763	3.381
mean	1.028	1.022	3.673
95% quantile	1.243	1.320	3.993
max	1.450	1.459	4.368

Table 7.1.: Summary of the three chemicals Co, Sc and Ti.

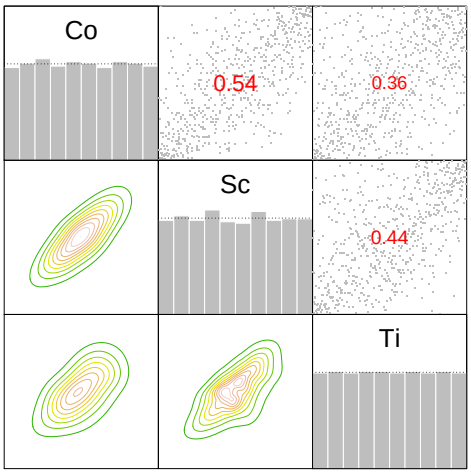

Figure 7.1.: Pairwise scatter plot of the three chemicals Co, Sc and Ti on the u-scale and normalized contour plots.

For the analysis we use the vine tree structure selected by an algorithm that was proposed by Kraus and Czado (2017c). The algorithm selects tree structures focused on producing simplified vine copulas for which the simplifying assumption is violated as little as possible. Chosen are edges (Co,Sc) and (Co,Ti) corresponding to pair copulas $c_{Co,Sc}$ and $c_{Co,Ti}$ for the first tree, and $c_{Sc,Ti;Co}$ in the second tree. The resulting structure is a D-vine: Sc-Co-Ti. The log-likelihood of this model is 433.06, AIC = -854.13 and BIC = -827.22. The fitted pair-copulas are displayed in Table 7.2.

edge	cop	par	par2	τ
Co,Sc	t	0.73	8.36	0.53
Co,Ti	Tawn2_180	1.82	0.56	0.31
Sc,Ti;Co	BB7	1.45	0.2	0.26

Table 7.2.: Fitted pair copulas for the selected vine tree structure in three dimensional case.

Since our program cannot sample from t-copulas yet, we use for $c_{Co,Sc}$ a Gaussian copula with the parameter 0.73. The final pair copula families used and its contour

plots are displayed in Fig. 7.2.

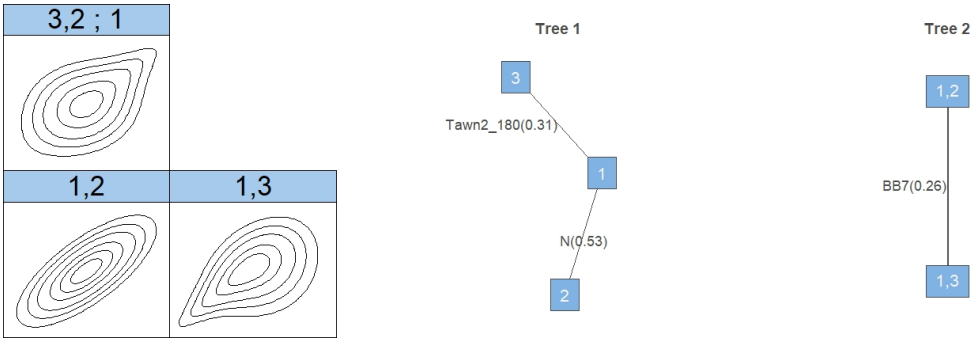

Figure 7.2.: Used vine tree structure and pair copula families, together with its contour plots. (1 = Co, 2 = Sc, 3 = Ti).

Univariate Conditional Sampling

Here we sample from univariate conditional distributions. We are interested in sampling from and expressing the distribution of $(X_{Co}|X_{Sc} = x_{Sc}, X_{Ti} = x_{Ti})$, $(X_{Sc}|X_{Co} = x_{Co}, X_{Ti} = x_{Ti})$ and $(X_{Ti}|X_{Co} = x_{Co}, X_{Sc} = x_{Sc})$, respectively. For the conditioning variables we choose low, medium and large values. Low values correspond to the 5% quantile, medium to the mean and large to the 95% quantile from the summary table shown in Table 7.1. We present the values in Table 7.3. Moreover, in Fig. 7.3 we show using the scatter plots, where the values are located.

We transform the variables from the x-scale to the u-scale with the probability integral transform, sample with our program, transform the variables back to the x-scale as described in the beginning of this chapter and plot the densities and distributions obtained by kernel density estimation. The densities together with the distributions are displayed in Fig. 7.4. We can see from the plot that with a change in the conditioning values, there is not only a change in the mean but also in the shape of the distribution, as seen for example, in the last row of the figure.

Conditioning values	Variables	Values used
low	(X_{Co}, X_{Sc})	$x_{Co} = 0.820, x_{Sc} = 0.763$
	(X_{Co}, X_{Ti})	$x_{Co} = 0.820, x_{Ti} = 3.381$
	(X_{Sc}, X_{Ti})	$x_{Sc} = 0.763, x_{Ti} = 3.381$
medium	(X_{Co}, X_{Sc})	$x_{Co} = 1.028, x_{Sc} = 1.022$
	(X_{Co}, X_{Ti})	$x_{Co} = 1.028, x_{Ti} = 3.673$
	(X_{Sc}, X_{Ti})	$x_{Sc} = 1.022, x_{Ti} = 3.673$
large	(X_{Co}, X_{Sc})	$x_{Co} = 1.243, x_{Sc} = 1.320$
	(X_{Co}, X_{Ti})	$x_{Co} = 1.243, x_{Ti} = 3.993$
	(X_{Sc}, X_{Ti})	$x_{Sc} = 1.320, x_{Ti} = 3.993$

Table 7.3.: The values used for conditioning variables in univariate conditional sampling in 3 dimensions.

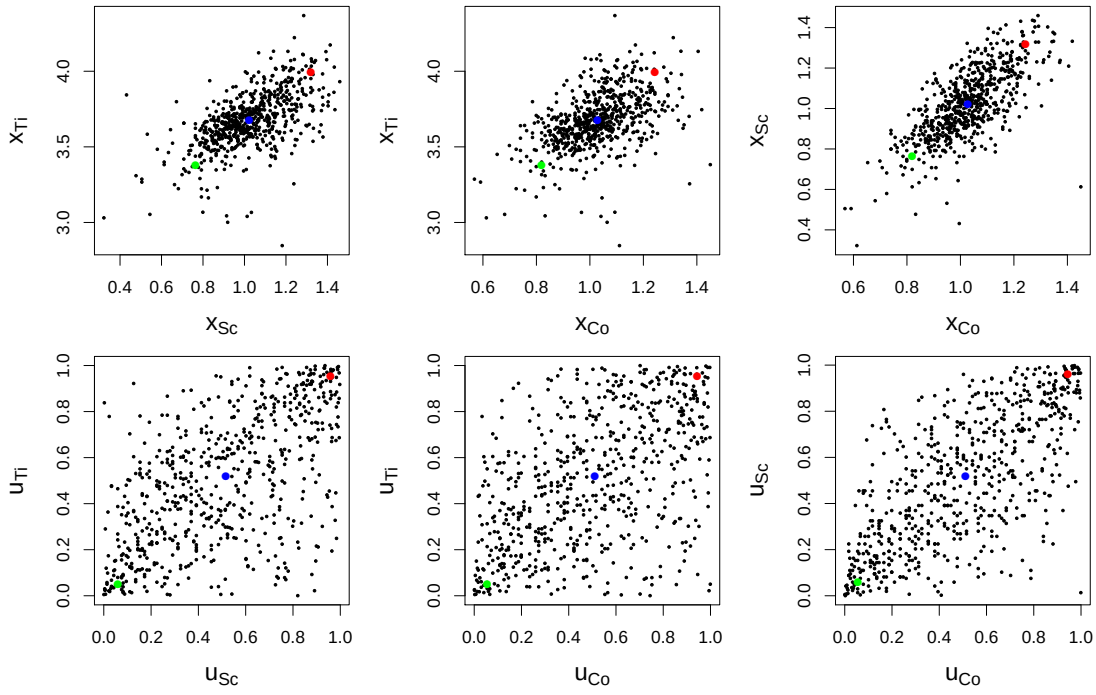

Figure 7.3.: Pair plots of the conditioning variables on the x-scale (first row) and on the u-scale (second row). Low conditioning values are depicted in green, medium in blue and large in red.

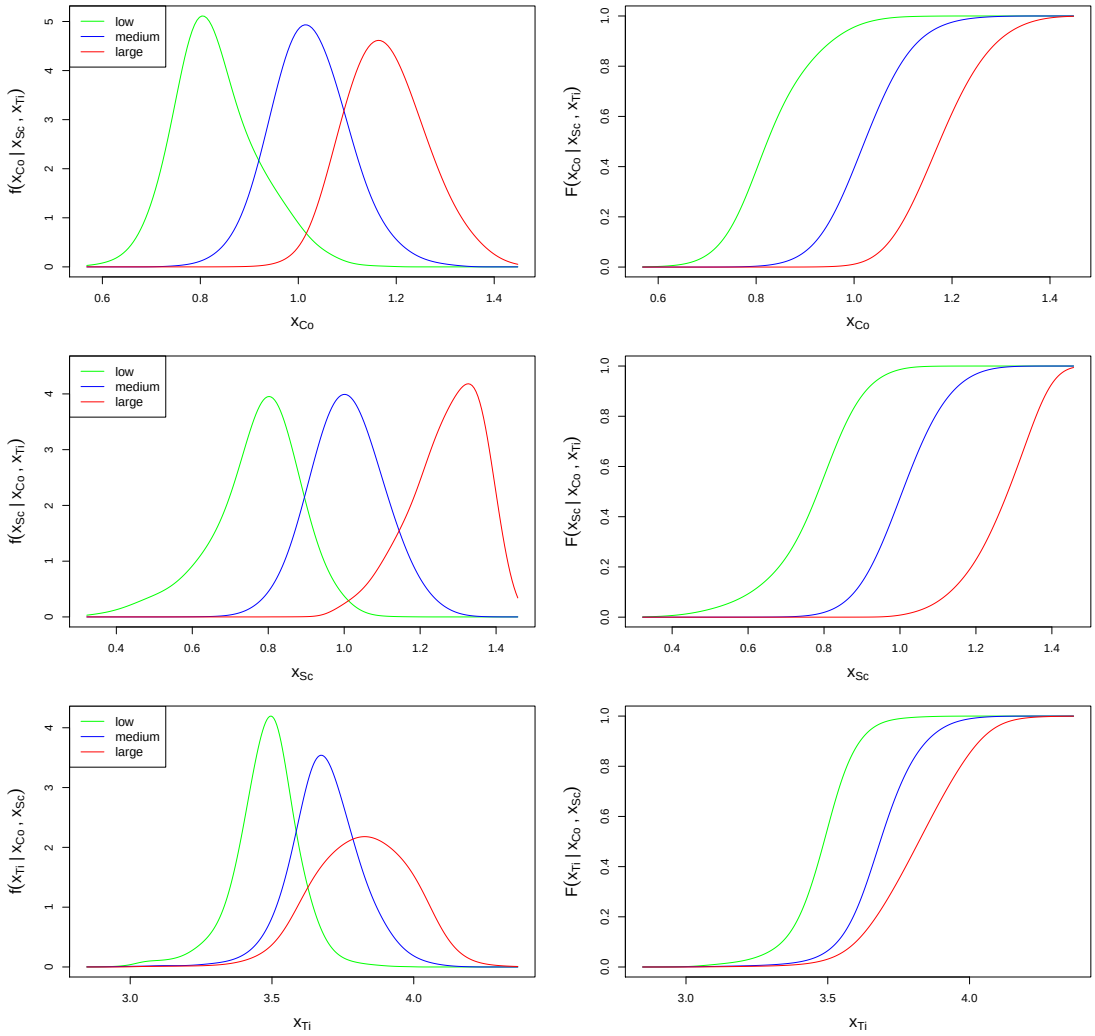

Figure 7.4.: Estimated densities and distribution functions in the three dimensional case. The rows correspond to different conditional density and distribution, in order $(X_{Co}|X_{Sc} = x_{Sc}, X_{Ti} = x_{Ti})$, $(X_{Sc}|X_{Co} = x_{Co}, X_{Ti} = x_{Ti})$ and $(X_{Ti}|X_{Co} = x_{Co}, X_{Sc} = x_{Sc})$. First column corresponds to density function, the second to distribution function. In each plot the densities and distributions for low, medium and large conditioning values are displayed together. The sample size is $n = 10000$, not including burn-in..

Bivariate Conditional Sampling

Here we sample from bivariate conditional distributions. We are interested in knowing the distribution and sampling from $(X_{Co}, X_{Sc}|X_{Ti} = x_{Ti})$. For the conditioning variable we choose low, medium and large value. Low value corresponds to the 5% quantile, medium to the mean and large to the 95% quantile from the summary table shown in Table 7.1, i.e. the values for X_{Ti} are $x_{Ti} = 3.381$ as low, $x_{Ti} = 3.673$ as medium and $x_{Ti} = 3.993$ as large. The densities together with the contour plots are displayed in Fig. 7.5.

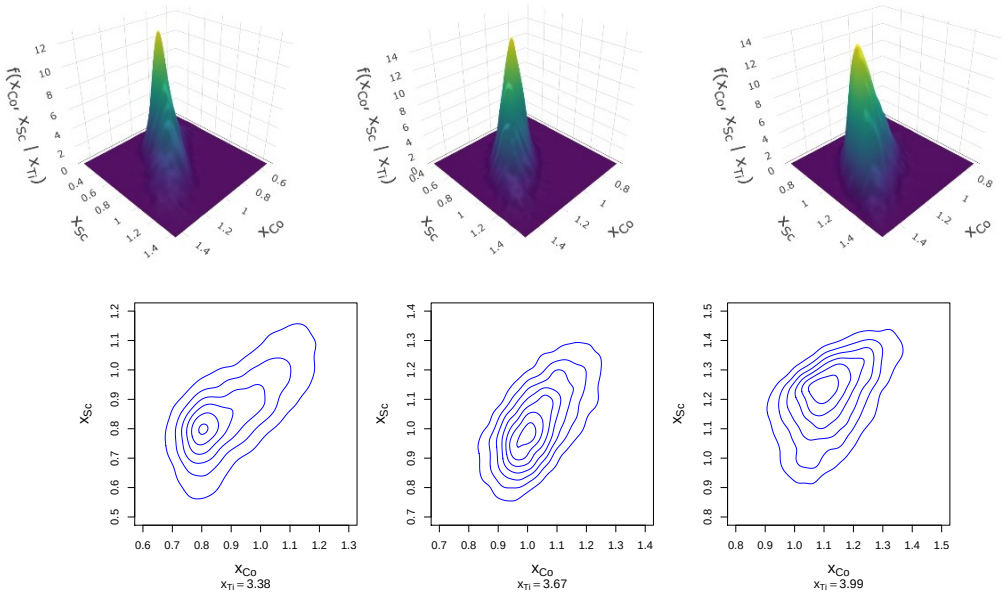

Figure 7.5.: Estimated densities and its contour plots for the distribution of $(X_{Co}, X_{Sc}|X_{Ti} = x_{Ti})$. The columns correspond to different conditioning value in order low, medium and large. First row corresponds to density function, the second to contour plot. The sample size is $n = 10000$.

Conditional Kendall's τ Associated with D-vine Sc-Co-Ti

We would like to see whether the chosen tree structure Sc-Co-Ti with its fitted pair copulas comply with the simplifying assumption for $c_{Sc,Ti;Co}$. We want to show that

the conditional copula $c_{Sc,Ti;Co}(C_{Sc|Co}(u_{Sc}|u_{Co}), C_{Ti|Co}(u_{Ti}|u_{Co}); u_{Co})$ does not depend on a value u_{Co}, i.e. we can write $c_{Sc,Ti;Co}(C_{Sc|Co}(u_{Sc}|u_{Co}), C_{Ti|Co}(u_{Ti}|u_{Co}))$ and thus the dependence between U_{Sc} and U_{Ti} is not changing with the value of U_{Co}. We show this by estimating the Kendall's Tau $\tau_{Sc,Ti|Co}$ for different conditioning value of Co.

Under the simplifying assumption $\hat{\tau}_{Sc,Ti|Co} = 0.26$, taken from Table 7.2. To observe the change, we compute the average of estimates of $\tau_{Sc,Ti|Co}$ together with its 90 % confidence intervals at 31 equally spaced grid points in the range of x_{Co} by sampling $n = 1000$ samples from $(U_{Sc}, U_{Ti}|U_{Co})$ $R = 100$ times using HMC.

First, we define 31 equally spaced grid points x^g_{Co}, $g = 1, \dots, 31$ in the range of x_{Co}. In order to estimate $\hat{F}_{Co}$ we use the kernel density estimation based on x_{Co} from the data. With its help, we transform the grid points x^g_{Co} by the probability integral transform to u^g_{Co}, i.e. $u^g_{Co} := \hat{F}_{Co}(x^g_{Co})$, $g = 1, \dots, 31$. For every grid point we sample $n = 1000$ samples from $(U_{Sc}, U_{Ti}|U_{Co} = u^g_{Co})$ $R = 100$ times using HMC and obtain 100 estimates of Kendall's Tau $\hat{\tau}_{Sc,Ti|Co}$, from which we compute the average and a 90 % confidence interval. The confidence interval is constructed by picking the 0.05 and the 0.95 empirical quantile as lower and upper limit among the 100 replications. The results are shown in the left panel of Fig. 7.6.

However, we would like to see the implication of using the simplified vine copula on the conditional $\tau_{Co,Ti|Sc}$ and $\tau_{Co,Sc|Ti}$. We estimate all Taus in the same way as described on $\tau_{Sc,Ti|Co}$ above. In order to obtain the estimates $\hat{\tau}_{Co,Ti|Sc}$ and $\hat{\tau}_{Co,Sc|Ti}$ under the simplifying assumption, we need to fit the model to the vine tree structure in which the edge in the last tree is $(Co, Ti; Sc)$ and $(Co, Sc; Ti)$, respectively. The resulting fits together with the estimated Kendall's Tau are presented in Table 7.4. Thus, under simplifying assumption $\hat{\tau}_{Co,Ti|Sc} = 0.08$ and $\hat{\tau}_{Co,Sc|Ti} = 0.42$. The results for $\tau_{Co,Ti|Sc}$ and $\tau_{Co,Sc|Ti}$ are plotted in the middle and right panel of Fig. 7.6, respectively.

edge	cop	par	par2	τ
Sc,Co	t	0.73	8.36	0.53
Sc,Ti	t	0.62	6.35	0.43
Co,Ti;Sc	t	0.12	6.22	0.08

edge	cop	par	par2	τ
Ti,Sc	t	0.62	6.35	0.43
Ti,Co	Tawn 180	1.82	0.56	0.31
Sc,Co;Ti	t	0.61	12.00	0.42

Table 7.4.: Fitted models in order to obtain the estimates of Kendall's Tau under the simplifying assumption. The model Co-Sc-Ti on the left shows $\hat{\tau}_{Co,Ti|Sc} = 0.08$ under the simplifying assumption and model Co-Ti-Sc on the right shows $\hat{\tau}_{Co,Sc|Ti} = 0.42$.

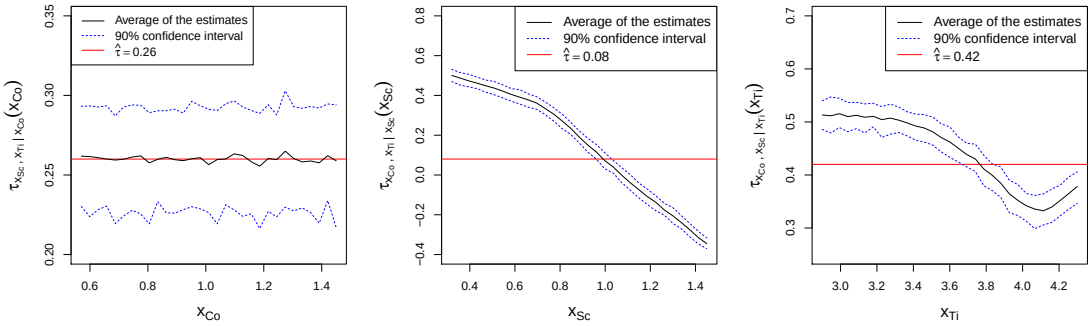

Figure 7.6.: Changes of Kendall's Tau in three dimensions. The 90% confidence interval together with the average of the estimates are shown in columns for $\tau_{Sc,Ti|Co}$, $\tau_{Co,Ti|Sc}$ and $\tau_{Co,Sc|Ti}$, respectively.

Comparison to Acar et al. (2012)

Acar et al. (2012) investigated if the simplifying assumption is appropriate in these 3-dimensional vine constructions using a local linear estimation together with a bootstrap method in order to get the confidence intervals. To build their models, they assumed that each pair of variables can be modeled by a Student's t copula, computed the conditional pseudo observations using h-functions of that t copula and fitted the pair copula in the last tree according to the joint behavior of those conditional pseudo observations. They studied two models, which are summarized in Table 7.5. They were interested in estimating $\tau_{Co,Ti|Sc}$ and $\tau_{Co,Sc|Ti}$ for different conditioning values of Sc and Ti, respectively.

edge	cop	par	par2	τ
Sc,Co	t	0.73	8.36	0.53
Sc,Ti	t	0.62	6.35	0.43
Co,Ti;Sc	F	0.72	-	0.08

edge	cop	par	par2	τ
Ti,Sc	t	0.62	6.35	0.43
Ti,Co	t	0.52	7.91	0.35
Co,Sc;Ti	Gum	1.65	-	0.39

Table 7.5.: Two fitted models Acar et al. (2012) used. They investigated the changes of $\tau_{Co,Ti|Sc}$ in the left model (model 1) and of $\tau_{Co,Sc|Ti}$ in the right model (model 2). The comparison was made with the estimates under the simplifying assumption depicted in the tables by blue colour.

Unfortunately, we cannot use the same models and sample from $U_{Co}, U_{Ti}|U_{Sc}$ and $U_{Co}, U_{Sc}|U_{Ti}$, since we would sample from the densities where the simplifying assumption is already assumed. Therefore, we use the vine tree structure we are interested in, the structure we use in this chapter and was proposed by Kraus and Czado (2017c). However, we fit different pair copulas as before so the model is as similar as possible to the ones by Acar et al. (2012). We likewise assume that each pair of variables can be modeled by a Student's t copula in the first tree and fit only the copula in the last tree. The resulting model is shown in Table 7.6. We again use Gaussian copulas instead of t-copulas.

edge	cop	par	par2	τ
Co,Sc	t	0.73	8.36	0.53
Co,Ti	t	0.52	7.91	0.35
Ti,Sc;Co	BB7	1.45	0.20	0.26

Table 7.6.: The most similar model with the vine tree structure from Kraus and Czado (2017c). The changes of Kendall's Tau in this model are used as a comparison to the changes of Kendall's Tau done in the work by Acar et al. (2012).

We can now compare the Kendall's Tau changes in the Acar et al. (2012) to the changes in Kendall's Tau estimates using the samples sampled by our proposed program. The comparisons are presented in Fig. 7.7. The first row corresponds to $\tau_{Co,Ti|Sc}$, where our estimates (left panel) are compared to Acar et al. (2012) (right panel) in which they used the model 1. The second row corresponds to $\tau_{Co,Sc|Ti}$, where our estimates (left panel) are compared to Acar et al. (2012) (right panel) in which they used the model 2.

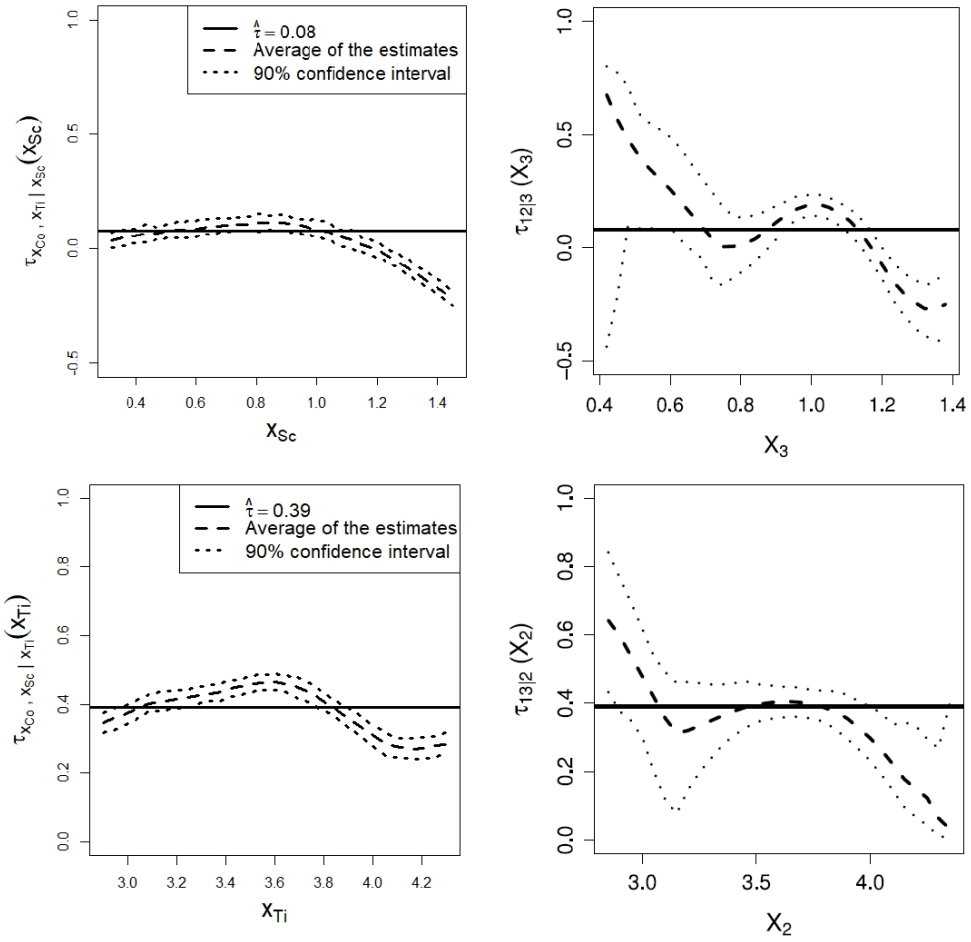

Figure 7.7.: Comparison of the estimates of $\tau_{Co,Ti|Sc}$ (top row) and $\tau_{Co,Sc|Ti}$ (bottom row) from Acar et al. (2012) (right) with the ones obtained by sampling from the proposed program (left).

7.2. Seven Dimensional Analysis

In this section we continue with all seven chemicals and their summary can be seen in Table 7.7. After transforming the data to the copula scale by applying the probability integral transform using the probability distribution function of kernel density estimation, we show the pairwise scatter plot in Fig. 7.8.

	U	Li	Co	K	Cs	Sc	Ti
min	0.146	0.602	0.568	3.263	1.114	0.322	2.847
5% quantile	0.431	1.079	0.820	3.998	1.713	0.763	3.381
mean	0.854	1.499	1.028	4.223	2.042	1.022	3.673
95% quantile	1.541	1.834	1.243	4.396	2.470	1.320	3.993
max	2.136	2.182	1.450	4.572	2.801	1.459	4.368

Table 7.7.: Summary measures for each of the seven chemicals (min, 5% quantile, mean, 95% quantile, max).

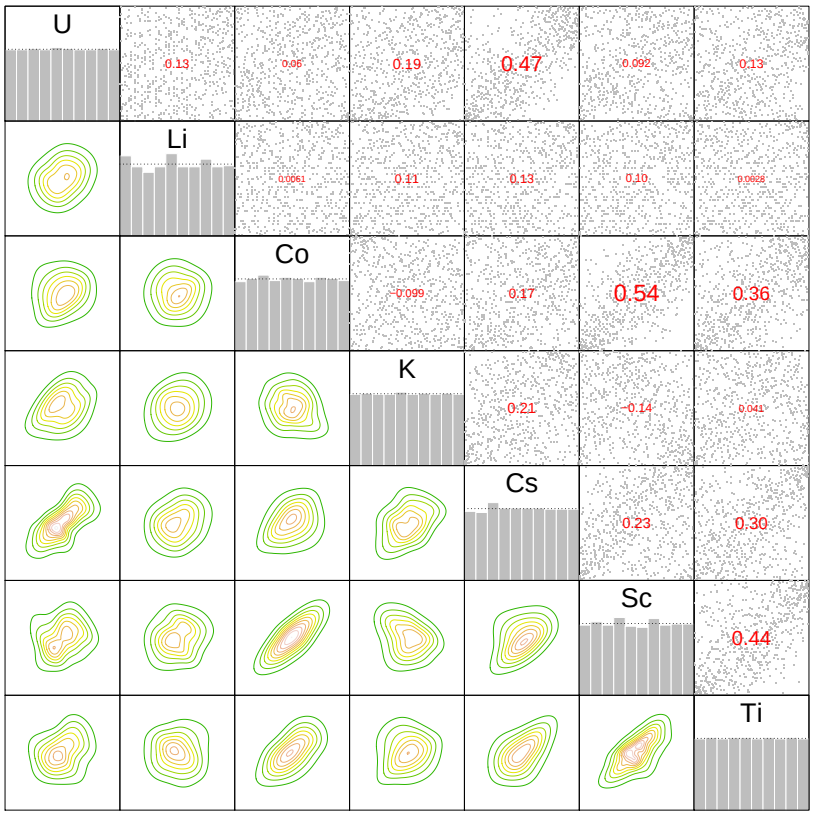

Figure 7.8.: Pairwise scatter plot of the seven chemicals on the u-scale and normalized contour plots.

For the analysis, we use three different vine tree structures, D-vine, C-vine and R-vine. The first one, denoted as **Vine 1**, is constructed from the 3-dimensional vine tree structure proposed by Kraus and Czado (2017c), which we mention in the previous section, and thus Sc-Co-Ti. We intend to build a D-vine, so we gradually add one other variable (chemical), that has the highest Kendall's Tau with the one of the variables already in the vine tree structure, with the position at the edge. After obtaining the whole D-vine structure we fit the pair copulas. Again, all fitted t-copulas have the second parameter above 5 and are replaced with Gaussian copulas. The resulting structure together with the pair copula families and their contour plots are shown in Fig. 7.9, Table 7.8 and Fig. 7.10. The variables Co, Sc, Ti have in 7-dimensional case numbers 3, 6, 7, respectively. So the starting structure Sc-Co-Ti is 6-3-7.

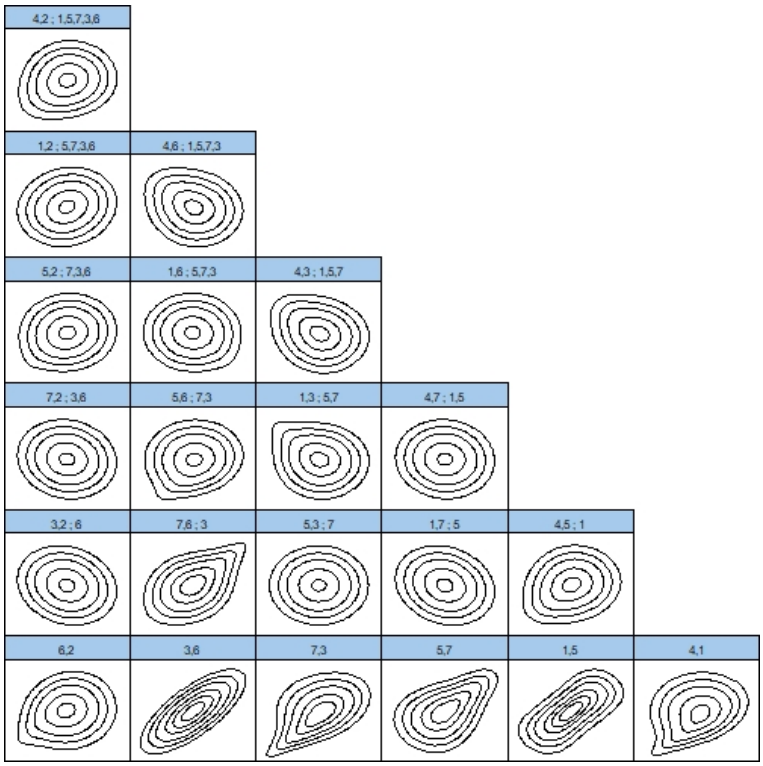

Figure 7.9.: Vine 1: Contour plots of the pair copulas used in 7-dimensional D-vine structure. (1 = U, 2 = Li, 3 = Co, 4 = K, 5 = Cs, 6 = Sc, 7 = Ti).

tree	edge	family	cop	par	par2	tau
1	4,1	114	Tawn180	1.75	0.27	0.17
1	1,5	5	F	5.23	0.00	0.47
1	5,7	10	BB8	1.90	0.97	0.30
1	7,3	114	Tawn180	1.82	0.56	0.31
1	3,6	2	t	0.73	8.36	0.53
1	6,2	214	Tawn2 180	1.28	0.30	0.10
2	4,5;1	20	SBB8	1.35	0.92	0.11
2	1,7;5	2	t	-0.17	6.62	-0.11
2	5,3;7	2	t	0.00	8.34	0.00
2	7,6;3	9	BB7	1.45	0.20	0.26
2	3,2;6	2	t	-0.15	17.36	-0.10
3	4,7;1,5	2	t	-0.05	5.75	-0.03
3	1,3;5,7	26	J90	-1.19	0.00	-0.10
3	5,6;7,3	114	Tawn180	1.33	0.16	0.08
3	7,2;3,6	2	t	-0.09	10.25	-0.06
4	4,3;1,5,7	30	BB8 90	-1.41	-0.91	-0.13
4	1,6;5,7,3	36	J270	-1.05	0.00	-0.03
4	5,2;7,3,6	14	SG	1.06	0.00	0.06
5	4,6;1,5,7,3	30	BB8 90	-1.39	-0.89	-0.11
5	1,2;5,7,3,6	5	F	0.72	0.00	0.08
6	4,2;1,5,7,3,6	3	C	0.20	0.00	0.09

Table 7.8.: Vine 1: Fitted pair copulas for the D-vine tree structure in seven dimensional case. (1 = U, 2 = Li, 3 = Co, 4 = K, 5 = Cs, 6 = Sc, 7 = Ti).

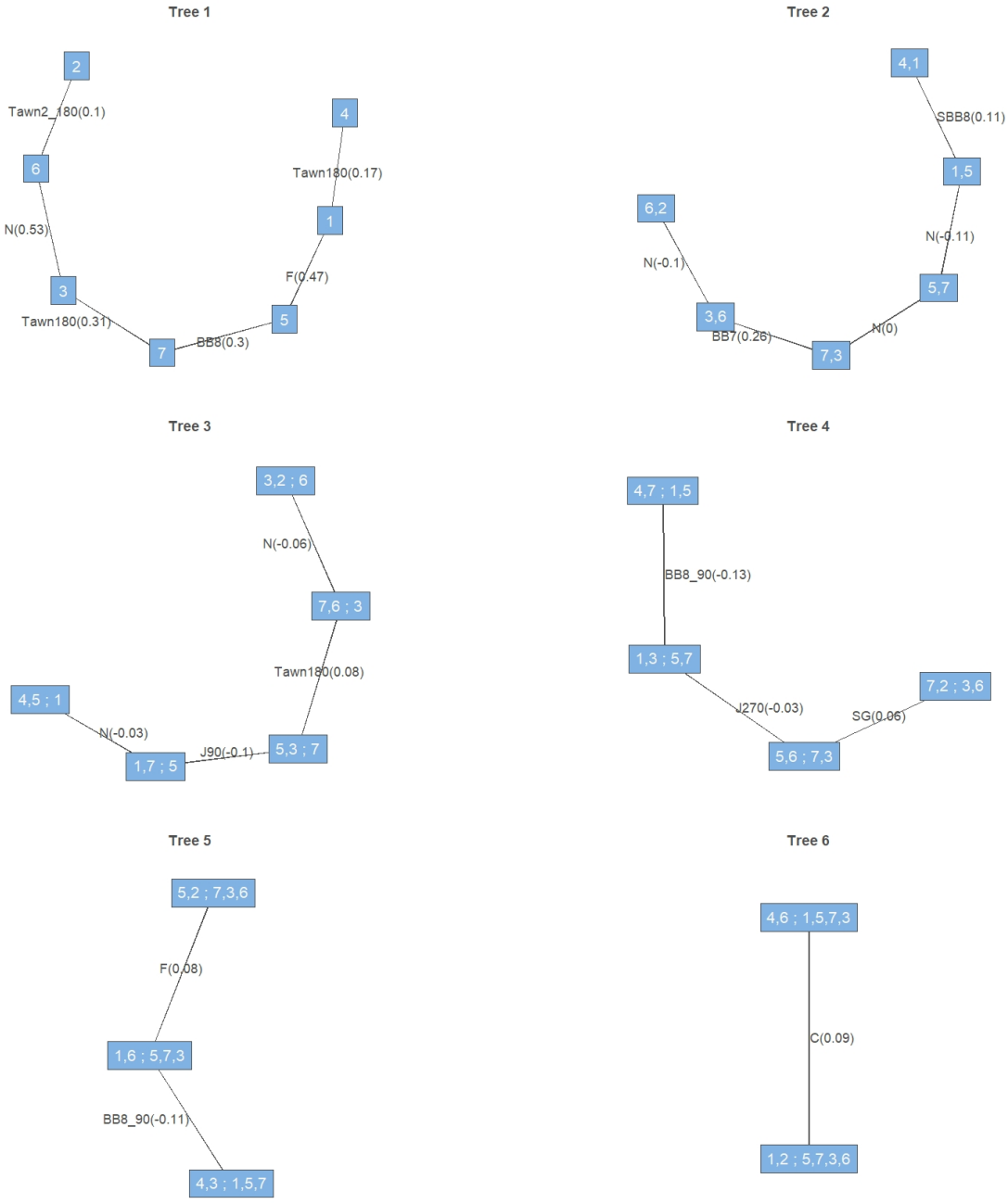

Figure 7.10.: Vine 1: 7-dimensional D-vine tree structure and pair copula families used. (1 = U, 2 = Li, 3 = Co, 4 = K, 5 = Cs, 6 = Sc, 7 = Ti).

The second one, denoted as **Vine 2**, is selected by algorithm that was proposed by Kraus and Czado (2017c) as the one, for which the simplifying assumption is violated as little as possible. The resulting vine tree structure is a C-vine. All fitted t-copulas have the second parameter above 5 and are replaced with Gaussian copulas. The structure together with the pair copula families and their contour plots are shown in Fig. 7.11, Table 7.9 and Fig. 7.12.

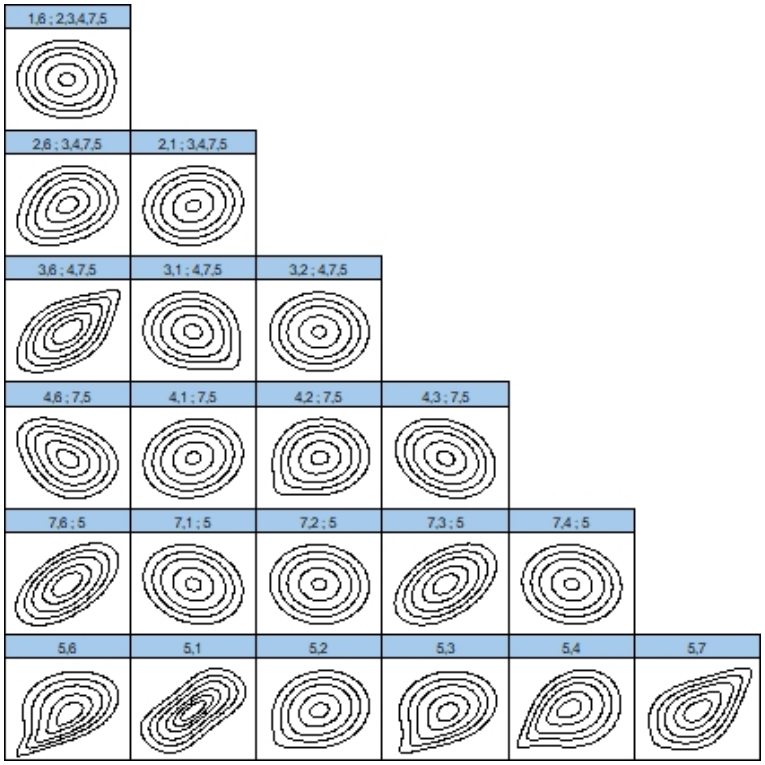

Figure 7.11.: Vine 2: Contour plots of the pair copulas used in 7-dimensional C-vine structure, selected by algorithm proposed by Kraus and Czado (2017c). (1 = U, 2 = Li, 3 = Co, 4 = K, 5 = Cs, 6 = Sc, 7 = Ti).

tree	edge	family	cop	par	par2	tau
1	5,7	19	SBB7	1.11	0.77	0.31
1	5,4	214	Tawn2_180	1.58	0.40	0.20
1	5,3	114	Tawn180	1.55	0.37	0.18
1	5,2	14	SG	1.13	0.00	0.11
1	5,1	5	F	5.24	0.00	0.47
1	5,6	114	Tawn180	1.88	0.43	0.26
2	7,4;5	2	t	-0.08	5.40	-0.05
2	7,3;5	2	t	0.47	6.87	0.31
2	7,2;5	2	t	-0.05	11.48	-0.03
2	7,1;5	2	t	-0.17	8.35	-0.11
2	7,6;5	2	t	0.56	5.41	0.38
3	4,3;7,5	2	t	-0.23	11.45	-0.15
3	4,2;7,5	16	SJ	1.15	0.00	0.08
3	4,1;7,5	1	N	0.13	0.00	0.09
3	4,6;7,5	30	BB8_90	-1.83	-0.87	-0.20
4	3,2;4,7,5	0	I	0.00	0.00	0.00
4	3,1;4,7,5	36	J270	-1.16	0.00	-0.09
4	3,6;4,7,5	7	BB1	0.19	1.42	0.36
5	2,1;3,4,7,5	5	F	0.59	0.00	0.07
5	2,6;3,4,7,5	20	SBB8	1.81	0.81	0.17
6	1,6;2,3,4,7,5	36	J270	-1.05	0.00	-0.03

Table 7.9.: Vine 2: Fitted pair copulas for the C-vine tree structure in seven dimensional case. (1 = U, 2 = Li, 3 = Co, 4 = K, 5 = Cs, 6 = Sc, 7 = Ti).

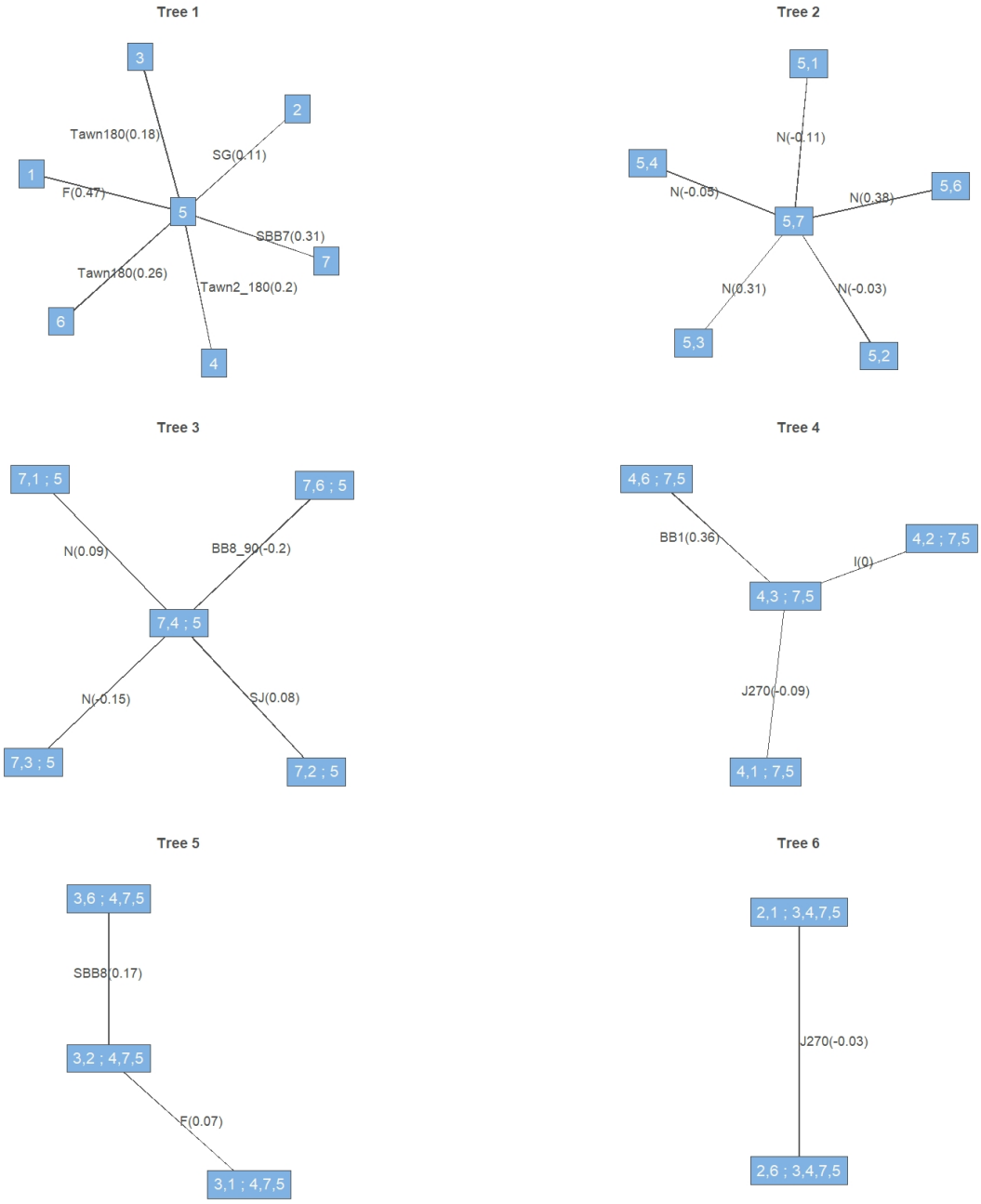

Figure 7.12.: Vine 2: 7-dimensional C-vine tree structure and pair copula families used. (1 = U, 2 = Li, 3 = Co, 4 = K, 5 = Cs, 6 = Sc, 7 = Ti).

The last one, denoted as **Vine 3**, is similarly as Vine 1 constructed from the 3-dimensional vine tree structure proposed by Kraus and Czado (2017c), the structure Sc-Co-Ti (6-3-7). We intend to build an R-vine, so we gradually add one other variable (chemical), that has the highest Kendall's Tau with the one of the variables already in the vine tree structure. However, this time we consider all variables already in the structure not only the ones at the edge. Since the first tree does not determine the following trees as in D-vine, we proceed as follows. After obtaining the first tree, we fit its pair copulas and compute the conditional pseudo observations using the h-functions of the respective copulas. We build the edge between the pairs with the highest conditional Kendall's Tau. We approach similarly after obtaining the second tree. The third tree is already a D-vine, which determines the following trees. Again, all fitted t-copulas have the second parameter above 5 and are replaced with Gaussian copulas. The resulting structure together with the pair copula families and their contour plots are shown in Fig. 7.13, Table 7.10 and Fig. 7.14.

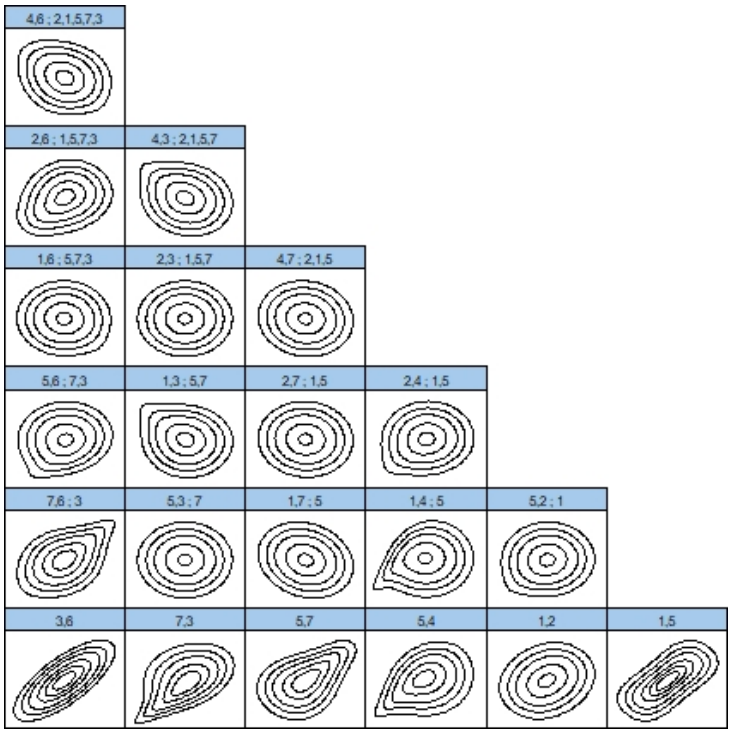

Figure 7.13.: Vine 3: Contour plots of the pair copulas used in 7-dimensional R-vine structure. (1 = U, 2 = Li, 3 = Co, 4 = K, 5 = Cs, 6 = Sc, 7 = Ti).

tree	edge	family	cop	par	par2	tau
1	1,5	5	F	5.23	0.00	0.47
1	1,2	5	F	1.19	0.00	0.13
1	5,4	214	Tawn2_180	1.59	0.40	0.20
1	5,7	10	BB8	1.90	0.97	0.30
1	7,3	114	Tawn180	1.82	0.56	0.31
1	3,6	2	t	0.73	8.36	0.53
2	5,2;1	16	SJ	1.08	0.00	0.04
2	1,4;5	214	Tawn2_180	1.85	0.13	0.10
2	1,7;5	2	t	-0.17	6.62	-0.11
2	5,3;7	2	t	0.00	8.34	0.00
2	7,6;3	9	BB7	1.45	0.20	0.26
3	2,4;1,5	16	SJ	1.12	0.00	0.06
3	2,7;1,5	2	t	-0.03	12.36	-0.02
3	1,3;5,7	26	J90	-1.19	0.00	-0.10
3	5,6;7,3	114	Tawn180	1.33	0.16	0.08
4	4,7;2,1,5	2	t	-0.06	6.21	-0.04
4	2,3;1,5,7	0	I	0.00	0.00	0.00
4	1,6;5,7,3	36	J270	-1.05	0.00	-0.03
5	4,3;2,1,5,7	24	G90	-1.15	0.00	-0.13
5	2,6;1,5,7,3	20	SBB8	1.58	0.89	0.16
6	4,6;2,1,5,7,3	37	BB1_270	-0.19	-1.06	-0.14

Table 7.10.: Vine 3: Fitted pair copulas for the R-vine tree structure in seven dimensional case. (1 = U, 2 = Li, 3 = Co, 4 = K, 5 = Cs, 6 = Sc, 7 = Ti).

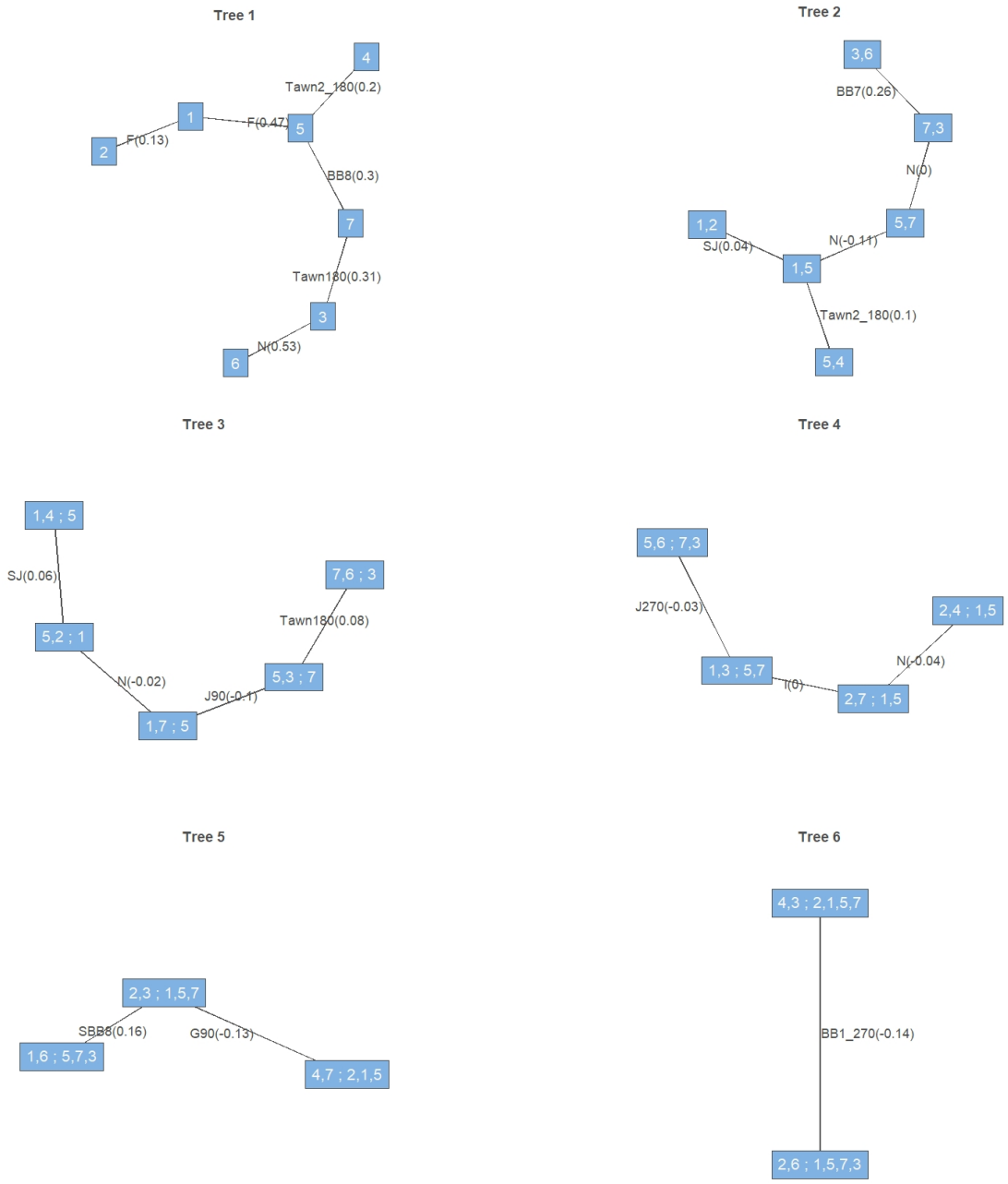

Figure 7.14.: Vine 3: 7-dimensional R-vine tree structure and pair copula families used. (1 = U, 2 = Li, 3 = Co, 4 = K, 5 = Cs, 6 = Sc, 7 = Ti).

In Table 7.11 we see the comparison of the three models, D-vine, C-vine and R-vine.

	type	AIC	BIC	log-lik
Vine 1	D	-1732.91	-1571.46	902.45
Vine 2	C	-1813.44	-1665.45	939.72
Vine 3	R	-1782.18	-1634.18	924.09

Table 7.11.: Comparison of three models, D-vine, C-vine, R-vine in terms of AIC, BIC and log-likelihood.

Univariate Conditional Sampling

In this part we sample from the univariate distributions using the 7-dimensional D-vine, C-vine and R-vine structure. We are interested in sampling from and expressing the distribution of

1. $(X_U|X_{Li} = x_{Li}, X_{Co} = x_{Co}, X_K = x_K, X_{Cs} = x_{Cs}, X_{Sc} = x_{Sc}, X_{Ti} = x_{Ti})$,
2. $(X_{Sc}|X_U = x_U, X_{Li} = x_{Li}, X_{Co} = x_{Co}, X_K = x_K, X_{Cs} = x_{Cs}, X_{Ti} = x_{Ti})$
3. $(X_{Ti}|X_U = x_U, X_{Li} = x_{Li}, X_{Co} = x_{Co}, X_K = x_K, X_{Cs} = x_{Cs}, X_{Sc} = x_{Sc})$.

These distributions are chosen, since none of them can be expressed directly when using Vine 1 (D-vine). On the other hand, in Vine 2 (C-vine) all of them can be expressed directly and in Vine 3 (R-vine) only the second one can be expressed directly.

For the conditioning variables we choose low, medium and large values. Low values correspond to 5% quantile, medium to mean and large to 95% quantile from the summary table shown in Table 7.7. We present the values in Table 7.12.

F		Values used
1	low	$x_{Li} = 1.079, x_{Co} = 0.820, x_K = 3.998, x_{Cs} = 1.713, x_{Sc} = 0.763, x_{Ti} = 3.381$
	med.	$x_{Li} = 1.499, x_{Co} = 1.028, x_K = 4.223, x_{Cs} = 2.042, x_{Sc} = 1.022, x_{Ti} = 3.673$
	lar.	$x_{Li} = 1.834, x_{Co} = 1.243, x_K = 4.396, x_{Cs} = 2.470, x_{Sc} = 1.320, x_{Ti} = 3.993$
2	low	$x_U = 0.431, x_{Li} = 1.079, x_{Co} = 0.820, x_K = 3.998, x_{Cs} = 1.713, x_{Ti} = 3.381$
	med.	$x_U = 0.854, x_{Li} = 1.499, x_{Co} = 1.028, x_K = 4.223, x_{Cs} = 2.042, x_{Ti} = 3.673$
	lar.	$x_U = 1.541, x_{Li} = 1.834, x_{Co} = 1.243, x_K = 4.396, x_{Cs} = 2.470, x_{Ti} = 3.993$
3	low	$x_U = 0.431, x_{Li} = 1.079, x_{Co} = 0.820, x_K = 3.998, x_{Cs} = 1.713, x_{Sc} = 0.763$
	med.	$x_U = 0.854, x_{Li} = 1.499, x_{Co} = 1.028, x_K = 4.223, x_{Cs} = 2.042, x_{Sc} = 1.022$
	lar.	$x_U = 1.541, x_{Li} = 1.834, x_{Co} = 1.243, x_K = 4.396, x_{Cs} = 2.470, x_{Sc} = 1.320$

Table 7.12.: The values used for conditioning variables in univariate conditional sampling in 7 dimensions.

As in the three dimensional case, we transform the variables from x-scale to the u-scale with probability integral transform, sample with our program, transform the variables back to the x-scale as explained in the beginning of this chapter and plot the densities and distributions obtained by kernel density estimation. The densities together with the distributions for all three types of vines are displayed in Fig. 7.15 for the first conditional distribution, in Fig. 7.16 for the second conditional distribution and in Fig. 7.17 for the third conditional distribution. We can again see from the plots that with a change in the conditioning values, there is not only a change in the mean but also in the shape of the distribution.

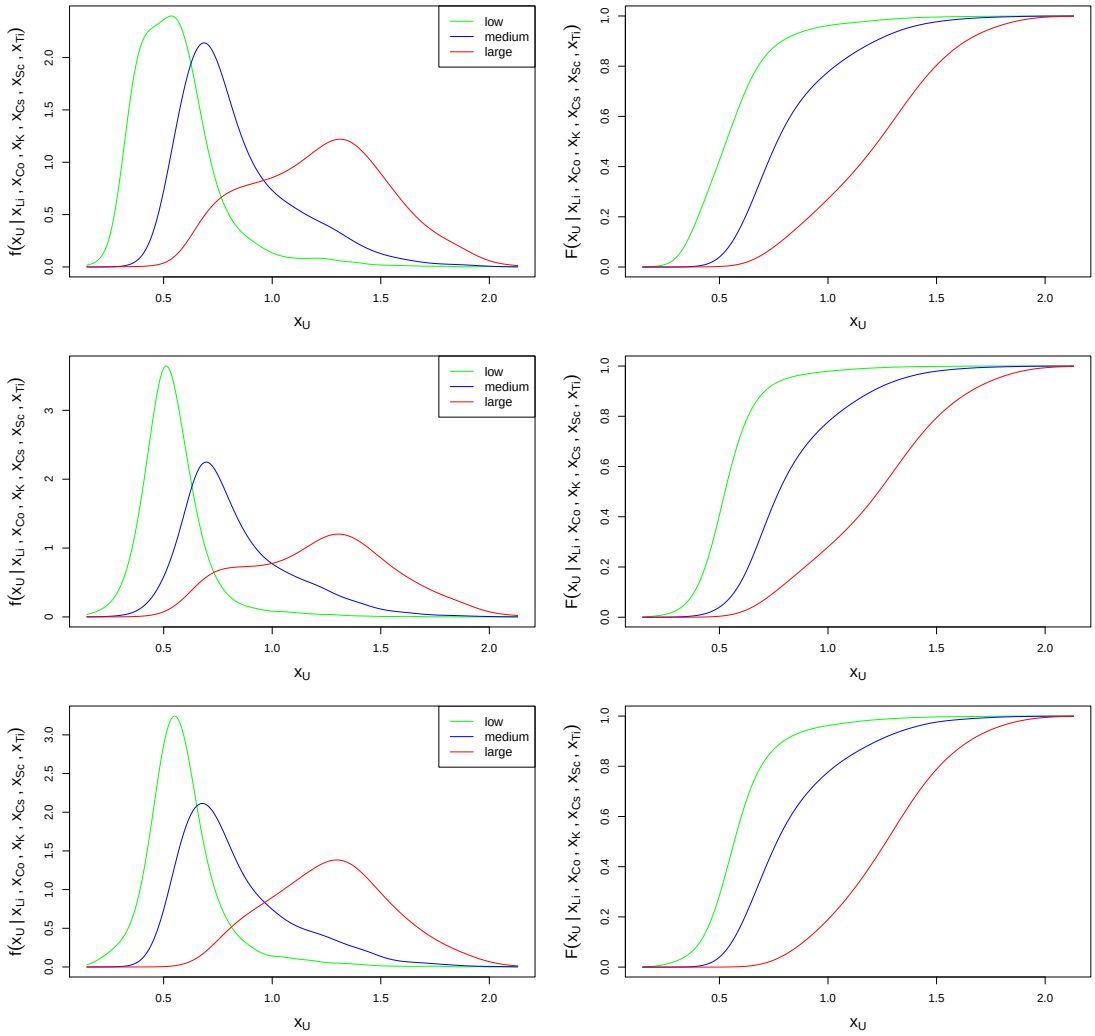

Figure 7.15.: Estimated densities and distribution functions in the seven dimensional case of $(X_U|X_{Li} = x_{Li}, X_{Co} = x_{Co}, X_K = x_K, X_{Cs} = x_{Cs}, X_{Sc} = x_{Sc}, X_{Ti} = x_{Ti})$. The rows correspond to different vine with order D-vine, C-vine and R-vine. First column corresponds to density function, the second to distribution function. In each plot the densities and distributions for low, medium and large conditioning values are displayed together. The sample size is $n = 10000$.

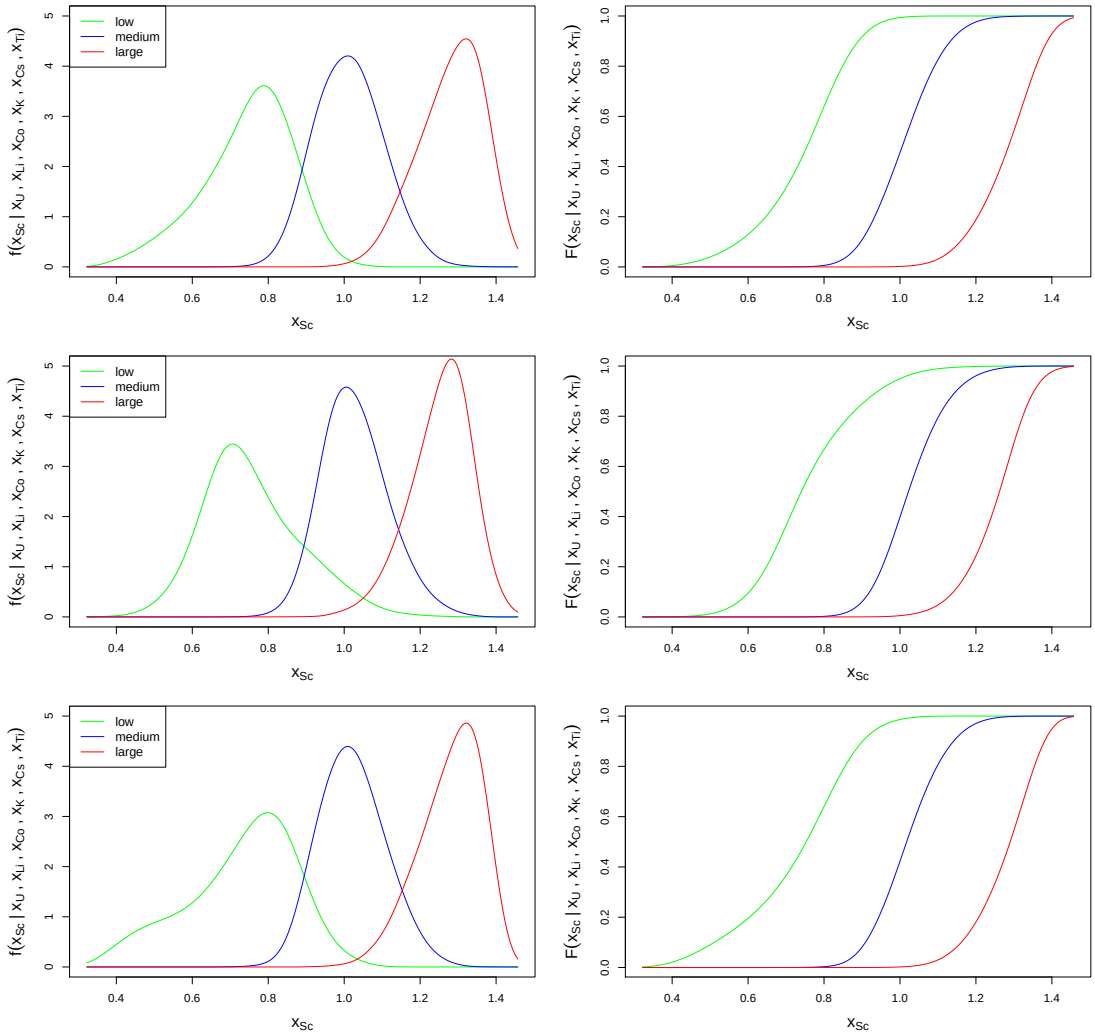

Figure 7.16.: Estimated densities and distribution functions in the seven dimensional case of $(X_{Sc}|X_U = x_U, X_{Li} = x_{Li}, X_{Co} = x_{Co}, X_K = x_K, X_{Cs} = x_{Cs}, X_{Ti} = x_{Ti})$. The rows correspond to different vine with order D-vine, C-vine and R-vine. First column corresponds to density function, the second to distribution function. In each plot the densities and distributions for low, medium and large conditioning values are displayed together. The sample size is $n = 10000$.

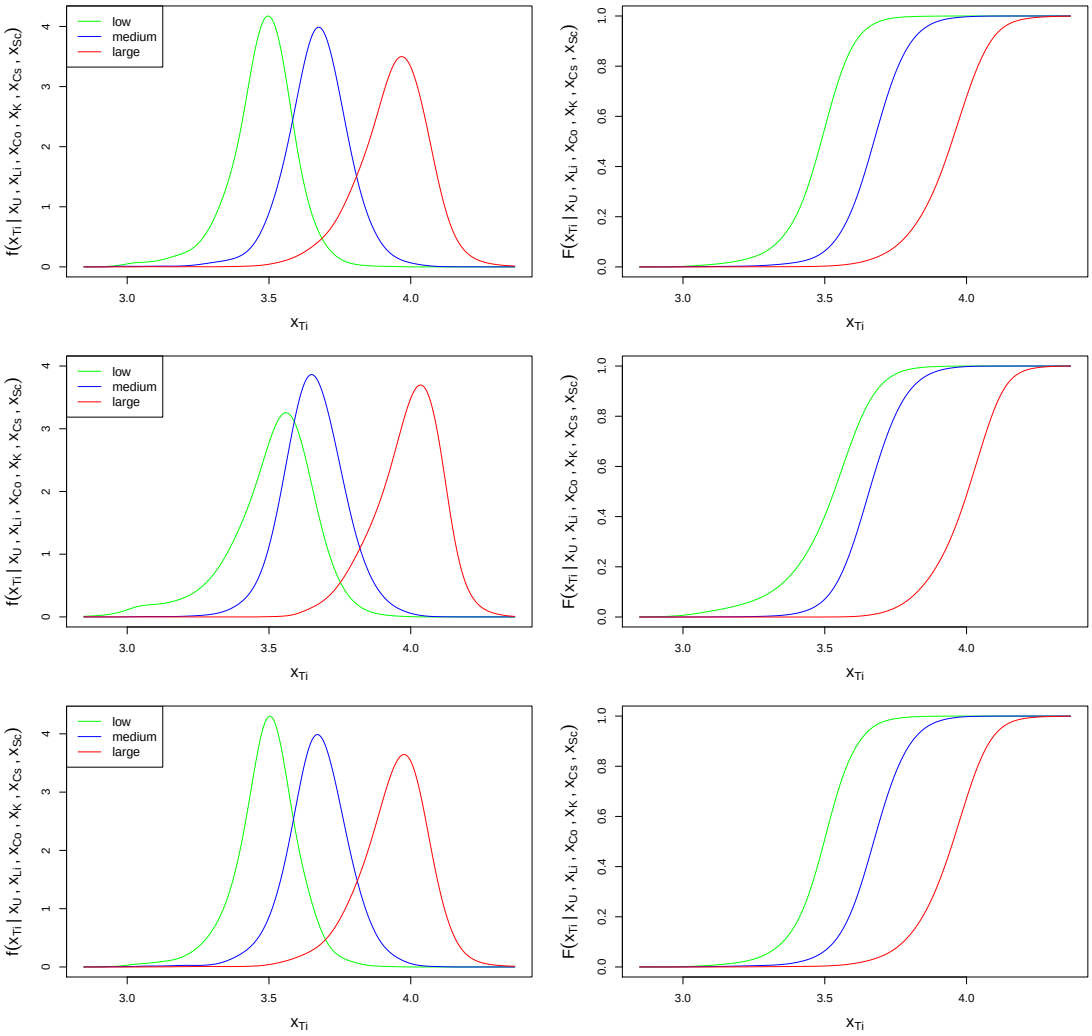

Figure 7.17.: Estimated densities and distribution functions in the seven dimensional case of $(X_{Ti}|X_U = x_U, X_{Li} = x_{Li}, X_{Co} = x_{Co}, X_K = x_K, X_{Cs} = x_{Cs}, X_{Sc} = x_{Sc})$. The rows correspond to different vine with order D-vine, C-vine and R-vine. First column corresponds to density function, the second to distribution function. In each plot the densities and distributions for low, medium and large conditioning values are displayed together. The sample size is $n = 10000$.

Bivariate Conditional Sampling

Here we sample from the bivariate distribution using the 7-dimensional D-vine, C-vine and R-vine structure. We are interested in knowing the distribution and sampling from $(X_{Co}, X_{Sc}|X_U = x_U, X_{Li} = x_{Li}, X_K = x_K, X_{Cs} = x_{Cs}, X_{Ti} = x_{Ti})$. For the conditioning variables we choose low, medium and large values as in the previous part. We present the values in Table 7.13.

Conditioning setup	Conditioning values used
low	$x_U = 0.431, x_{Li} = 1.079, x_K = 3.998, x_{Cs} = 1.713, x_{Ti} = 3.381$
medium	$x_U = 0.854, x_{Li} = 1.499, x_K = 4.223, x_{Cs} = 2.042, x_{Ti} = 3.673$
large	$x_U = 1.541, x_{Li} = 1.834, x_K = 4.396, x_{Cs} = 2.470, x_{Ti} = 3.993$

Table 7.13.: The values used for conditioning variables in bivariate conditional sampling in 7 dimensions.

The densities for all three types of vines are displayed in Fig. 7.18 and the contour plots in Fig. 7.19

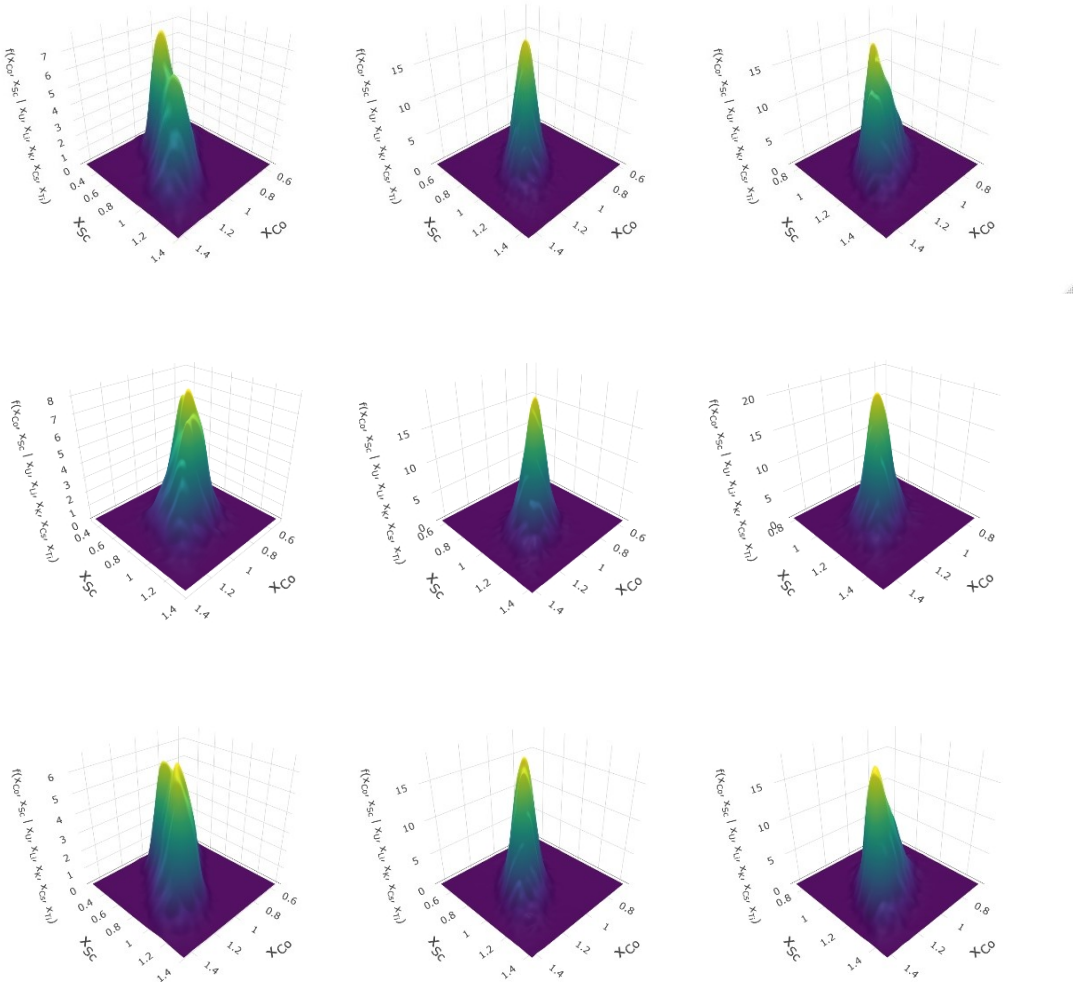

Figure 7.18.: Estimated densities for the distribution of $(X_{Co}, X_{Sc}|X_U = x_U, X_{Li} = x_{Li}, X_K = x_K, X_{Cs} = x_{Cs}, X_{Ti} = x_{Ti})$ for D-vine, C-vine and R-vine, in rows. The columns correspond to different conditioning values in order low, medium and large. The sample size is $n = 10000$.

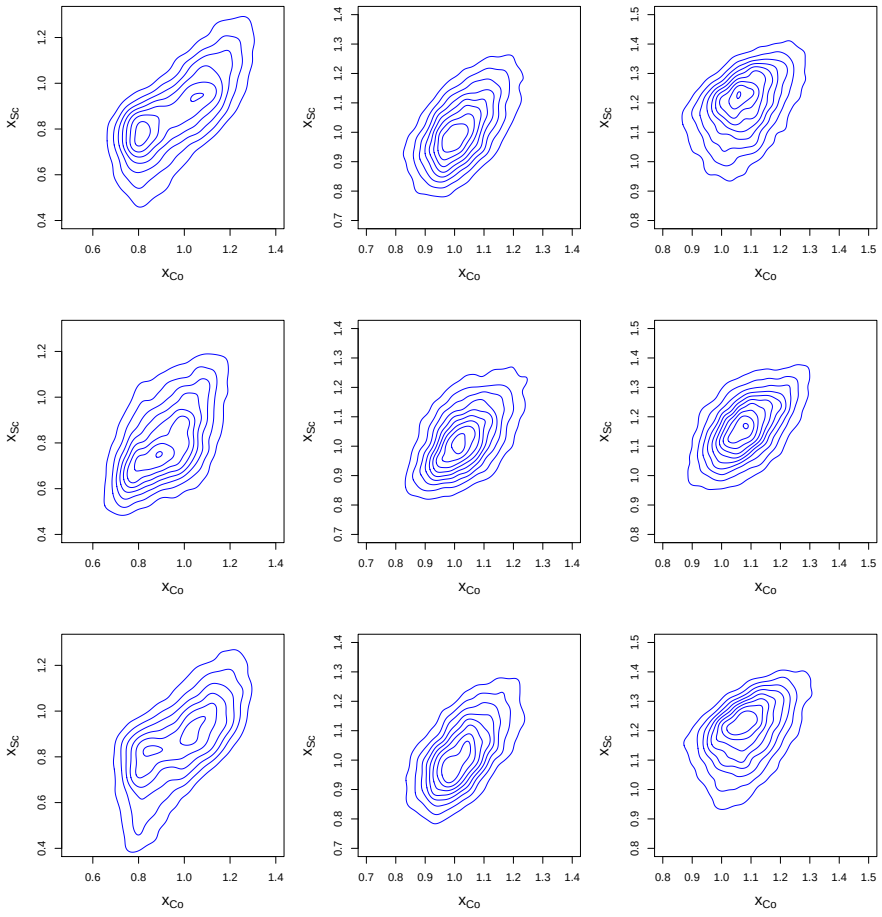

Figure 7.19.: Contour plots of estimated densities of $(X_{Co}, X_{Sc}|X_U = x_U, X_{Li} = x_{Li}, X_K = x_K, X_{Cs} = x_{Cs}, X_{Ti} = x_{Ti})$ for D-vine (top row), C-vine (middle row) and R-vine (bottom row). The columns correspond to different conditioning values: low (left column), medium (middle column) and large (right column). The sample size is $n = 10000$.

Effect of Conditioning Values of a Single Covariate on a Conditional Kendall's Tau, where all Other Covariates are Chosen to be Low, Medium and Large Values

Moreover, we would like to see how the conditional Kendall's Tau is changing when changing the values of different conditioning variables. We are interested in $\tau_{Co,Sc|U,Li,K,Cs,Ti}$. Under the simplifying assumption $\hat{\tau}_{Co,Sc|U,Li,K,Cs,Ti} = 0.37$, i.e. we first

choose an R-vine model in which the edge in the last tree is $(Co, Sc; U, Li, K, Cs, Ti)$ and then compute the estimation of the Kendall's Tau. The model is presented in Table 7.14 and its vine tree structure in Fig. 7.20. This estimate of Tau will be depicted in the plots by black line. We start with changing the values for X_U having the values for remaining conditioning variables low, medium and large. We repeat this for the other 4 variables (X_{Li}, X_K, X_{Cs}, X_{Ti}).

tree	edge	family	cop	par	par2	tau
1	6,7	2	t	0.62	6.35	0.43
1	7,5	10	BB8	1.90	0.97	0.30
1	5,4	214	Tawn2_180	1.59	0.40	0.20
1	4,2	16	SJ	1.19	0.00	0.10
1	2,1	5	F	1.19	0.00	0.13
1	1,3	2	t	0.09	3.87	0.06
2	6,5;7	214	Tawn2_180	1.54	0.21	0.12
2	7,4;5	2	t	-0.09	6.32	-0.06
2	5,2;4	14	SG	1.09	0.00	0.08
2	4,1;2	114	Tawn180	1.64	0.28	0.16
2	2,3;1	0	I	0.00	0.00	0.00
3	6,4;7,5	40	BB8_270	-1.82	-0.86	-0.20
3	7,2;5,4	2	t	-0.04	10.63	-0.03
3	5,1;4,2	5	F	4.54	0.00	0.42
3	4,3;2,1	30	BB8_90	-1.25	-0.97	-0.10
4	6,2;7,5,4	20	SBB8	1.96	0.74	0.16
4	7,1;5,4,2	2	t	-0.15	11.86	-0.10
4	5,3;4,2,1	4	G	1.23	0.00	0.19
5	6,1;7,5,4,2	36	J270	-1.12	0.00	-0.06
5	7,3;5,4,2,1	5	F	3.06	0.00	0.31
6	6,3;7,5,4,2,1	2	t	0.55	14.33	0.37

Table 7.14.: Fitted model in order to obtain the estimate of Kendall's Tau under the simplifying assumption. The model shows $\hat{\tau}_{Co,Sc|U,Li,K,Cs,Ti} = 0.37$ under the simplifying assumption. The value is used for comparison with the changes in Kendall's Tau when changing the conditioning variables. (1 = U, 2 = Li, 3 = Co, 4 = K, 5 = Cs, 6 = Sc, 7 = Ti).

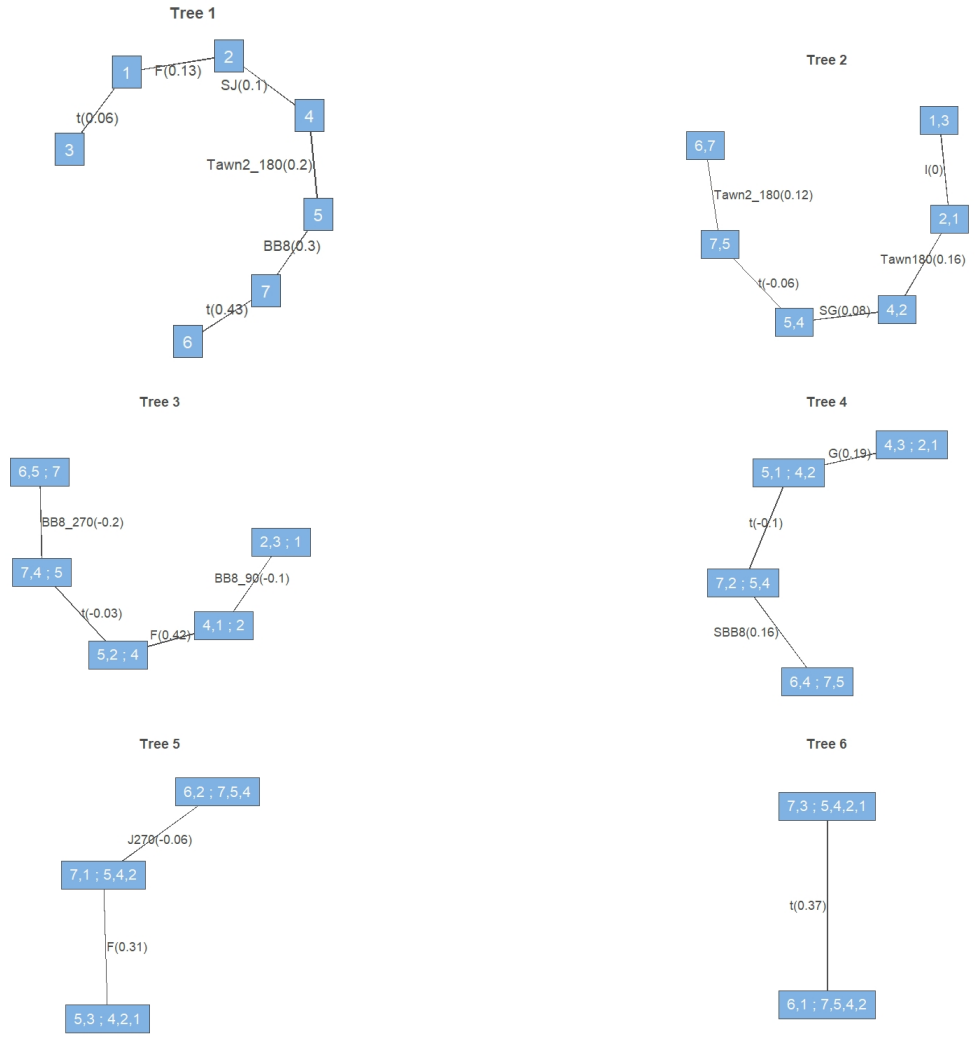

Figure 7.20.: Vine tree structure and pair copula families used in the model fitted in order to obtain the Kendall's Tau $\hat{\tau}_{Co,Sc|U,Li,K,Cs,Ti}$ under the simplifying assumption. (1 = U, 2 = Li, 3 = Co, 4 = K, 5 = Cs, 6 = Sc, 7 = Ti).

The approach is the same as in the 3-dimensional case. To assess how the conditional Kendall's Tau changes when changing the conditional variables, we compute the average of estimates of $\tau_{Co,Sc|U,Li,K,Cs,Ti}$ together with its 90 % confidence intervals at 31 equally spaced grid points in the range of e.g. x_U by sampling $n = 1000$ samples from $(U_{Co}, U_{Sc}|U_U, U_{Li}, U_K, U_{Cs}, U_{Ti})$ $R = 100$ times using HMC. The remaining conditioning

variables have low, medium or large values. We summarize the values in Table 7.15.

Changing		Values for remaining conditioning variables
X_U	low	$x_{Li} = 1.079, x_K = 3.998, x_{Cs} = 1.713, x_{Ti} = 3.381$
	medium	$x_{Li} = 1.499, x_K = 4.223, x_{Cs} = 2.042, x_{Ti} = 3.673$
	large	$x_{Li} = 1.834, x_K = 4.396, x_{Cs} = 2.470, x_{Ti} = 3.993$
X_{Li}	low	$x_U = 0.431, x_K = 3.998, x_{Cs} = 1.713, x_{Ti} = 3.381$
	medium	$x_U = 0.854, x_K = 4.223, x_{Cs} = 2.042, x_{Ti} = 3.673$
	large	$x_U = 1.541, x_K = 4.396, x_{Cs} = 2.470, x_{Ti} = 3.993$
X_K	low	$x_U = 0.431, x_{Li} = 1.079, x_{Cs} = 1.713, x_{Ti} = 3.381$
	medium	$x_U = 0.854, x_{Li} = 1.499, x_{Cs} = 2.042, x_{Ti} = 3.673$
	large	$x_U = 1.541, x_{Li} = 1.834, x_{Cs} = 2.470, x_{Ti} = 3.993$
X_{Cs}	low	$x_U = 0.431, x_{Li} = 1.079, x_K = 3.998, x_{Ti} = 3.381$
	medium	$x_U = 0.854, x_{Li} = 1.499, x_K = 4.223, x_{Ti} = 3.673$
	large	$x_U = 1.541, x_{Li} = 1.834, x_K = 4.396, x_{Ti} = 3.993$
X_{Ti}	low	$x_U = 0.431, x_{Li} = 1.079, x_K = 3.998, x_{Cs} = 1.713$
	medium	$x_U = 0.854, x_{Li} = 1.499, x_K = 4.223, x_{Cs} = 2.042$
	large	$x_U = 1.541, x_{Li} = 1.834, x_K = 4.396, x_{Cs} = 2.470$

Table 7.15.: Summary of the values for the conditioning variables when studying the change of conditional Kendall's Tau. In the first column the variable that is changing is shown. In the third one we present the values for the remaining variables in case of low, medium and large setup.

We present how the estimates $\hat{\tau}_{Co,Sc|U,Li,K,Cs,Ti}$ change when changing the conditioning variable X_U, X_{Li}, X_K, X_{Cs} in Fig. 7.21. When changing the conditioning variable X_{Ti}, we can compare the results in 7 dimensions with the 3-dimensional one from previous subsection. We present the results of $\hat{\tau}_{Co,Sc|Ti}$ together with the ones of $\hat{\tau}_{Co,Sc|U,Li,K,Cs,Ti}$ for D-vine, C-vine and R-vine together in Fig. 7.22. In the 3-dimensional case we do not have any other conditional variables than X_{Ti}, therefore there is not any low, medium and large case. From both figures it can be seen, that the C-vine structure was really constructed so that the simplifying assumption is violated as little as possible, since there are some Kendall's Taus that are not varying much with the change in the conditioning variables.

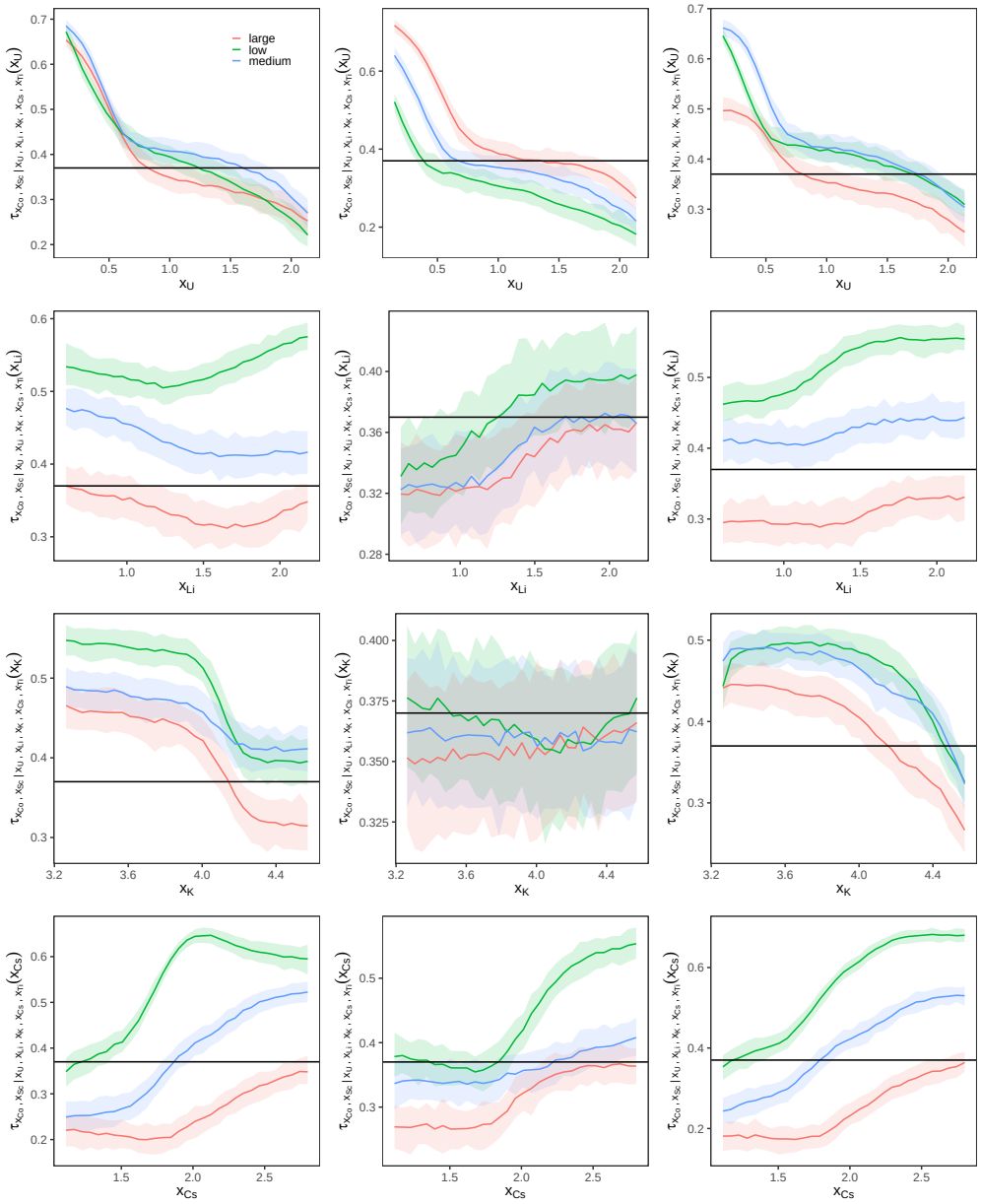

Figure 7.21.: Change of Kendall's Tau when changing a single variable with all other variables being set to low, medium and large values. The 90% confidence interval together with the average of the estimates are shown for $\tau_{Co,Sc|U,Li,K,Cs,Ti}$. In rows, different conditioning variable is changing, in order X_U, X_{Li}, X_K, X_{CS}. In columns, different vine is used, D-vine (left), C-vine (middle) and R-vine (right). In every plot the results for low, medium and large conditioning values is compared with the one under the simplifying assumption $\hat{\tau}_{Co,Sc|U,Li,K,Cs,Ti} = 0.37$ (black line).

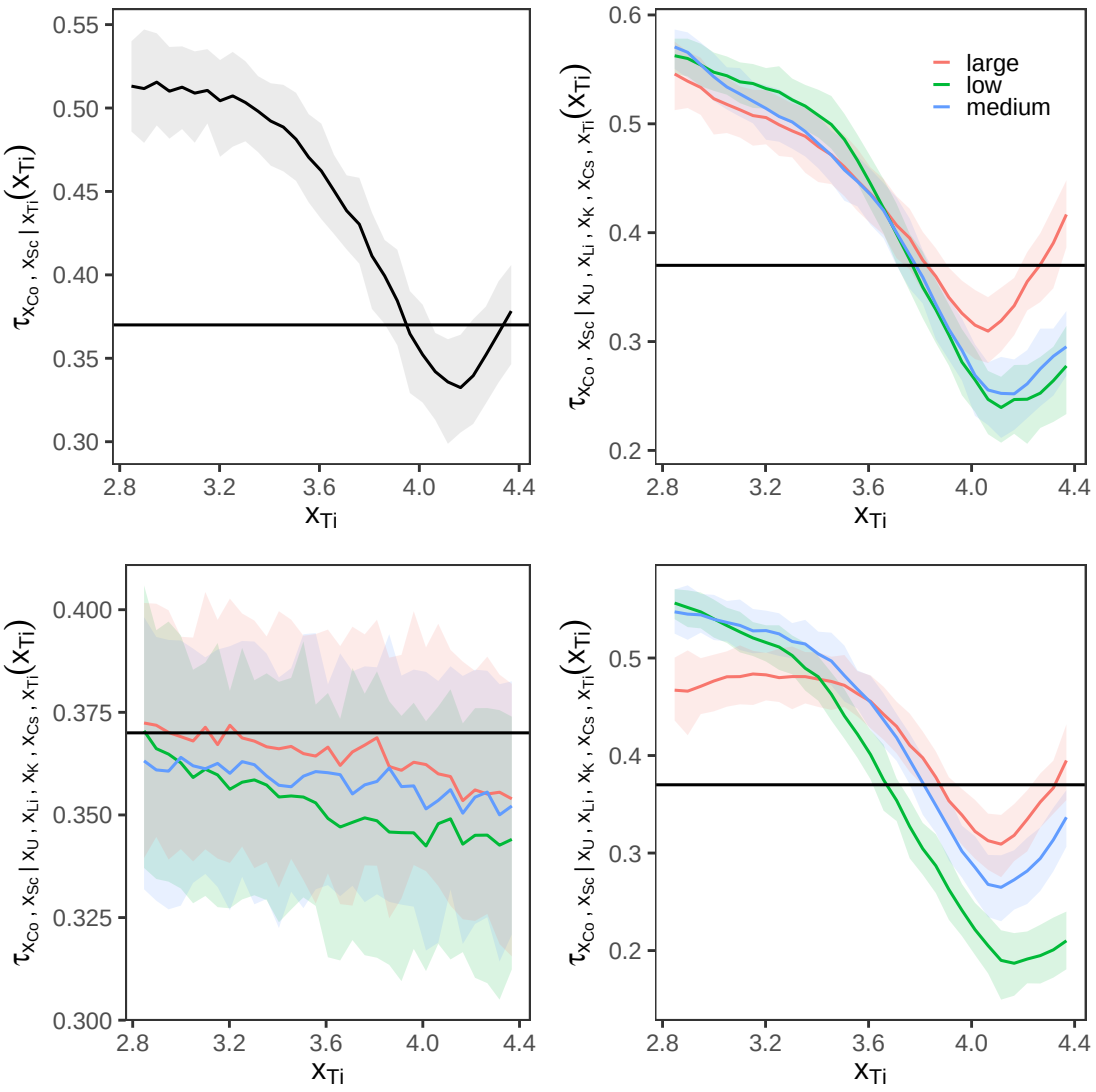

Figure 7.22.: Change of Kendall's Tau in seven dimensions compared to the three-dimensional one. The 90% confidence interval together with the average of the estimates are shown for $\tau_{Co,Sc|Ti}$ in the upper left corner and for $\tau_{Co,Sc|U,Li,K,Cs,Ti}$ in the remaining parts. Different vines are used in seven-dimensional case, D-vine (upper right), C-vine (lower left) and R-vine (lower right). In every plot the results for low, medium and large conditioning values are compared (except of the 3-dim. case in which there is not any low, medium, large case) with the one under the simplifying assumption $\hat{\tau}_{Co,Sc|U,Li,K,Cs,Ti} = 0.37$ (black line).

A. Further Plots from the Simulation Study

The following additional plots and results are related to the Specification 1 shown in the first row of Fig. 6.4 in the Simulation Setup 1.

A.1. Estimation of Cumulative Distribution Functions

Not only the conditional density function can be compared, but also cumulative distribution functions. In this section we compare the true theoretical cumulative distribution function of $\mathbf{U}_{\mathcal{C}_1}|\mathbf{U}_{\mathcal{C}_2} = \mathbf{u}_{r_\alpha}^{\mathcal{C}_2}$ with an estimate based on the sample $\mathbf{u}_i(\mathbf{u}_{r_\alpha}^{\mathcal{C}_2})$, $i = 1, \ldots, n$. We use again the kernel estimation together with the transformation trick as in the density estimation in Simulation Study, however, this time we estimate the probability distribution function. In the following figures, we, likewise, compare the estimates of samples with sizes $n = 1000$, 5000, 10000. The comparison of distributions are shown in Fig. A.1.

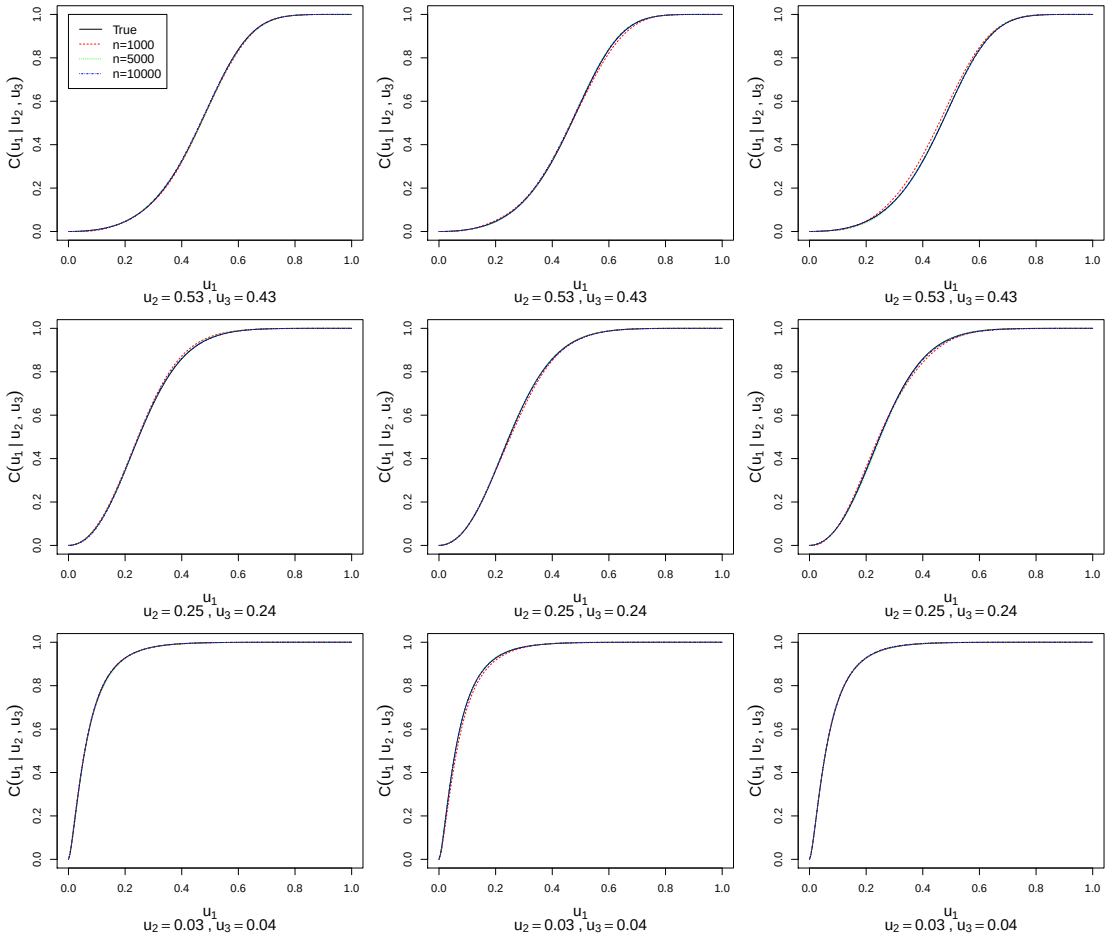

Figure A.1.: Comparison of distribution functions for the Specification 1 in Simulation Setup 1. Columns correspond to 3 chosen iterations out of 100 and rows to different conditioning values, in order low, medium and large. The true theoretical distribution is compared to kernel estimates based on samples with sizes $n = 1000, 5000, 10000$ as seen in the legend in the upper left corner. Beneath each plot we show the values of conditioning variables.

A.2. Histograms of Values Transformed by Probability Integral Transform

In the Simulation Study, to measure the goodness-of-fit, we use the Kolmogorov-Smirnov test to assess whether the transformed data $v_i(\mathbf{u}_{r_\alpha}^{\mathcal{C}_2}) := F(u_i(\mathbf{u}_{r_\alpha}^{\mathcal{C}_2})|\mathbf{U}_{\mathcal{C}_2} = \mathbf{u}_{r_\alpha}^{\mathcal{C}_2})$, in which $u_i(\mathbf{u}_{r_\alpha}^{\mathcal{C}_2})$, $i = 1, \ldots, n$ are the samples, are uniformly distributed. However, this can be assessed also visually by means of histograms. The histograms are approximate representations of distributions and the one for uniformly distributed data has the bars almost exactly the same height.

We present the histograms given different conditioning values as well as different sample sizes. The histograms of transformed values $v_i(\mathbf{u}_{r_\alpha}^{\mathcal{C}_2})$ of samples with sizes $n = 1000$, 5000 and 10000 are shown in Fig. A.2, Fig. A.3 and Fig. A.4, respectively.

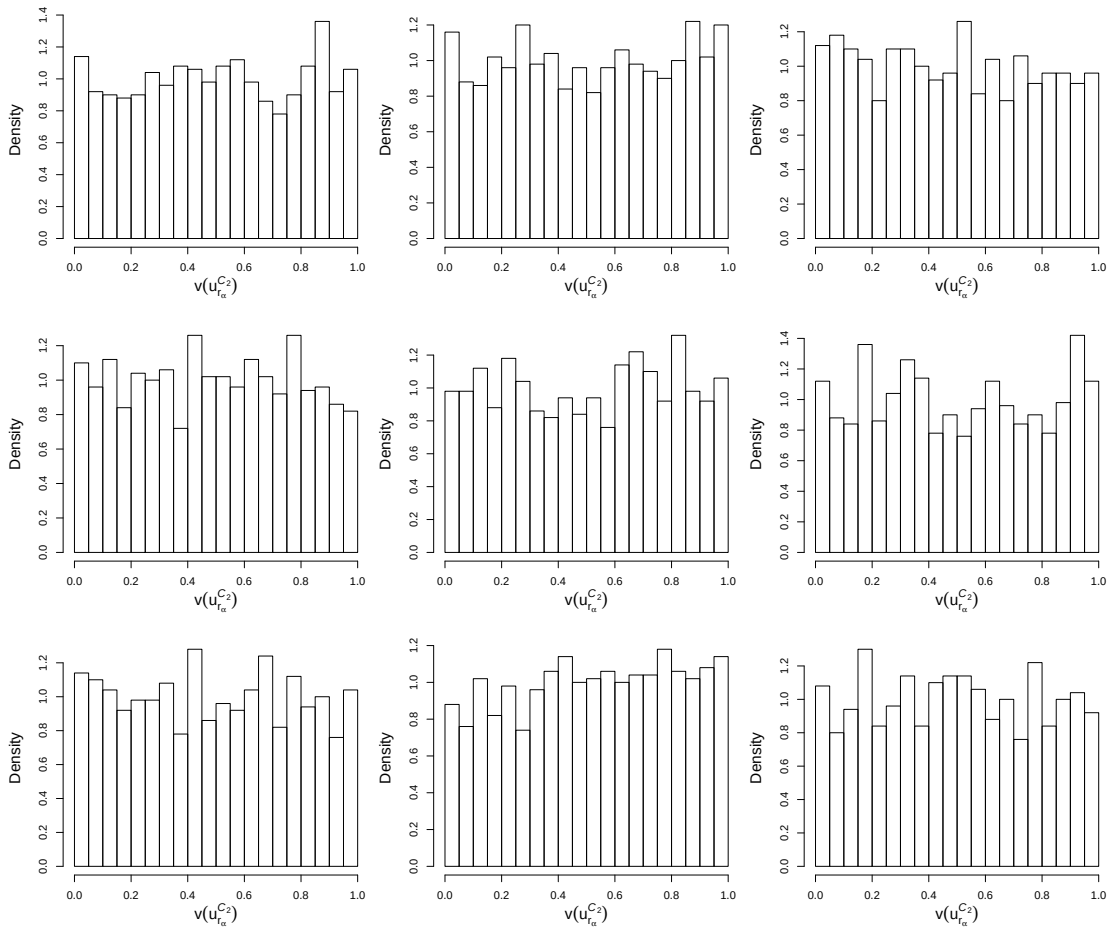

Figure A.2.: Histograms of transformed values $v_i(\mathbf{u}_{r_\alpha}^{\mathcal{C}_2})$ of samples with size $n = 1000$ for the Specification 1 in Simulation Setup 1. Columns correspond to 3 chosen iterations out of 100 and rows to different conditioning values, in order low, medium and large.

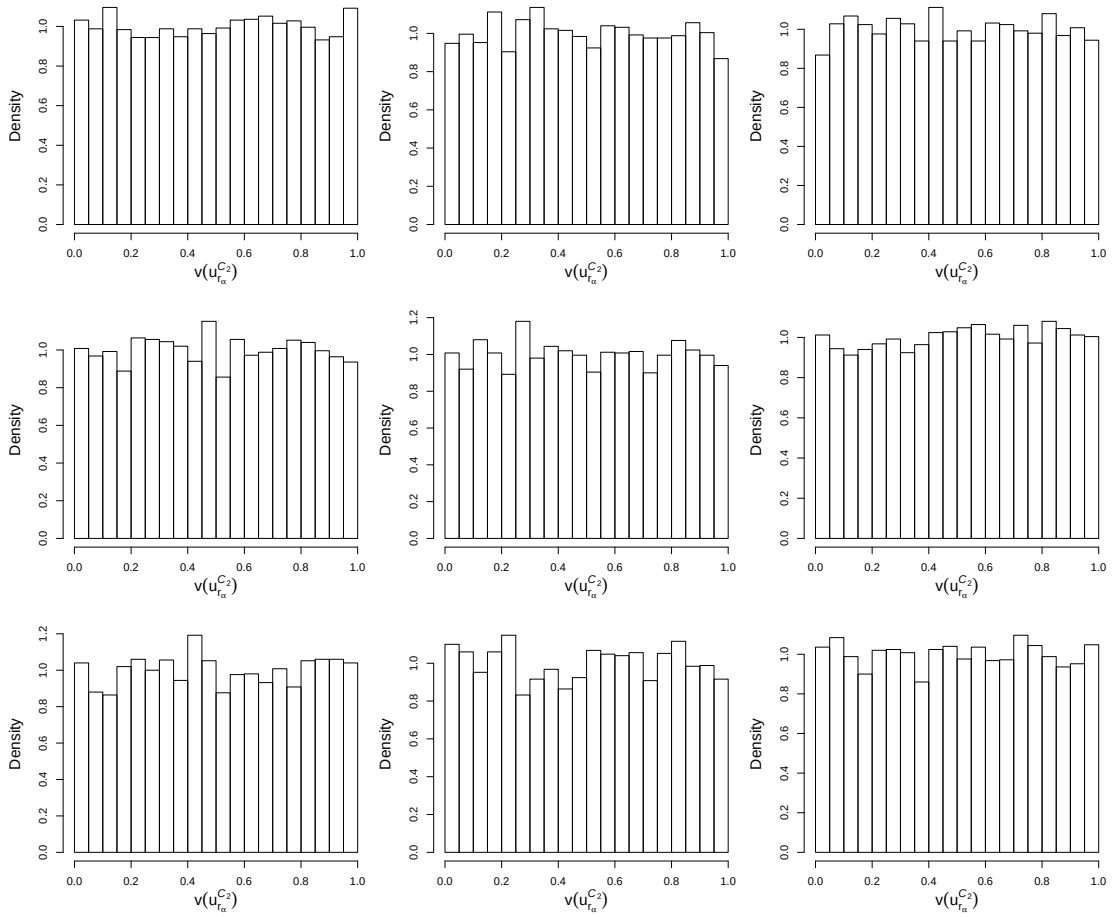

Figure A.3.: Histograms of transformed values $v_i(\mathbf{u}_{r_\alpha}^{\mathcal{C}_2})$ of samples with size $n = 5000$ for the Specification 1 in Simulation Setup 1. Columns correspond to 3 chosen iterations out of 100 and rows to different conditioning values, in order low, medium and large.

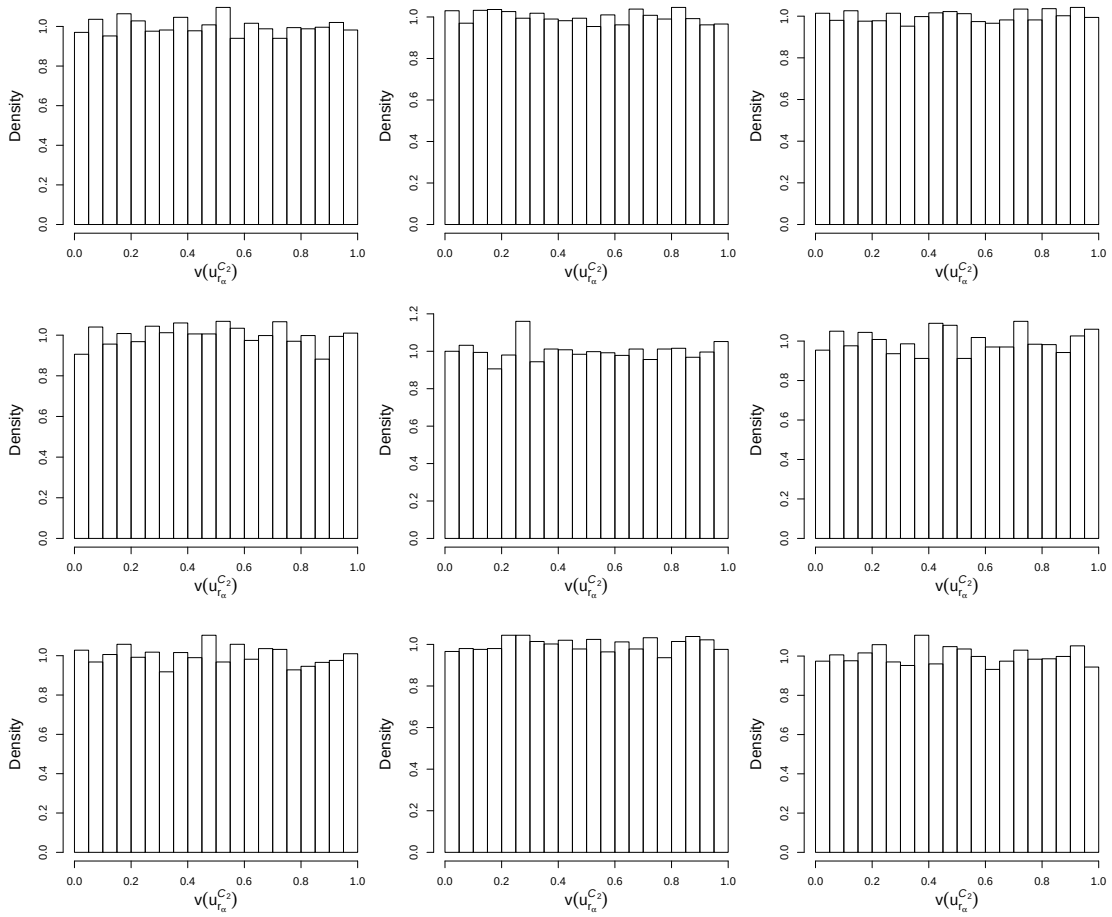

Figure A.4.: Histograms of transformed values $v_i(\mathbf{u}_{r_\alpha}^{\mathcal{C}_2})$ of samples with size $n = 10000$ for the Specification 1 in Simulation Setup 1. Columns correspond to 3 chosen iterations out of 100 and rows to different conditioning values, in order low, medium and large.

List of Figures

List of Tables

Listings

Bibliography

Acar, E. F., Genest, C., and Nešlehová, J. (2012). Beyond simplified pair-copula constructions. *Journal of Multivariate Analysis*, 110:74–90.

Bedford, T. and Cooke, R. M. (2002). Vines–a new graphical model for dependent random variables. *The Annals of Statistics*, 30(4):1031–1068.

Betancourt, M. (2017). A conceptual introduction to hamiltonian monte carlo.

Casella, G. and Berger, R. L. (2021). *Statistical inference*. Cengage Learning.

Cook, R. D. and Johnson, M. E. (1981). A family of distributions for modelling non-elliptically symmetric multivariate data. *Journal of the Royal Statistical Society: Series B (Methodological)*, 43(2):210–218.

Cook, R. D. and Johnson, M. E. (1986). Generalized burr-pareto-logistic distributions with applications to a uranium exploration data set. *Technometrics*, 28(2):123–131.

Czado, C. (2019). *Analyzing Dependent Data with Vine Copulas A Practical Guide With R*. Springer International Publishing.

Dimitrova, D. S., Kaishev, V. K., and Tan, S. (2020). Computing the kolmogorov-smirnov distribution when the underlying cdf is purely discrete, mixed, or continuous. *Journal of Statistical Software*, 95(1):1–42.

Duong, T. (2016). Non-parametric smoothed estimation of multivariate cumulative distribution and survival functions, and receiver operating characteristic curves. *Journal of the Korean Statistical Society*, 45(1):33–50.

Hofert, M., Kojadinovic, I., Maechler, M., and Yan, J. (2020). *copula: Multivariate Dependence with Copulas*. R package version 1.0-1.

Hoffman, M. D., Gelman, A., et al. (2014). The no-u-turn sampler: adaptively setting path lengths in hamiltonian monte carlo. *J. Mach. Learn. Res.*, 15(1):1593–1623.

Hollander, M., Wolfe, D. A., and Chicken, E. (2013). *Nonparametric statistical methods*, volume 751. John Wiley & Sons.

Huber, P. J. (1985). Projection pursuit. *The annals of Statistics*, pages 435–475.

Joe, H. (1996). Families of m-variate distributions with given margins and m (m-1)/2 bivariate dependence parameters. *Lecture Notes-Monograph Series*, pages 120–141.

Kolmogorov, A. (1933). Sulla determinazione empirica di una lgge di distribuzione. *Inst. Ital. Attuari, Giorn.*, 4:83–91.

Kraus, D. and Czado, C. (2017a). D-vine copula based quantile regression. *Computational Statistics & Data Analysis*, 110:1–18.

Kraus, D. and Czado, C. (2017b). D-vine copula based quantile regression. *Computational Statistics & Data Analysis*, 110:1–18.

Kraus, D. and Czado, C. (2017c). Growing simplified vine copula trees: improving dißmann's algorithm.

Lehmann, E. L., Romano, J. P., and Casella, G. (2005). *Testing statistical hypotheses*, volume 3. Springer.

Meyn, S. P. and Tweedie, R. L. (2012). *Markov chains and stochastic stability*. Springer Science & Business Media.

Nagler, T., Schellhase, C., and Czado, C. (2017). Nonparametric estimation of simplified vine copula models: comparison of methods. *Dependence Modeling*, 5(1):99–120.

Nagler, T., Schepsmeier, U., Stoeber, J., Brechmann, E. C., Graeler, B., and Erhardt, T. (2021). *VineCopula: Statistical Inference of Vine Copulas*. R package version 2.4.2.

Nagler, T. and Vatter, T. (2021). *rvinecopulib: High Performance Algorithms for Vine Copula Modeling*. R package version 0.5.5.1.1.

Neal, R. M. et al. (2011). Mcmc using hamiltonian dynamics. *Handbook of markov chain monte carlo*, 2(11):2.

Nelsen, R. B. (2007). *An introduction to copulas*. Springer Science & Business Media.

Parzen, E. (1962). On estimation of a probability density function and mode. *The annals of mathematical statistics*, 33(3):1065–1076.

R Core Team (2019). *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, Vienna, Austria.

Robert, C. P., Casella, G., and Casella, G. (1999). *Monte Carlo statistical methods*, volume 2. Springer.

Rosenblatt, M. (1952). Remarks on multivariate transformations. *Annals of Mathematical Statistics*, 23:470–472.

Sklar, A. (1959). Fonctions de répartition à n dimensions et leurs marges. *Publications de l'Institut de Statistique de l'Université de Paris*, 8:229–231.

Smirnov, N. (1948). Table for estimating the goodness of fit of empirical distributions. *The annals of mathematical statistics*, 19(2):279–281.

Smirnov, N. V. (1939). On the estimation of the discrepancy between empirical curves of distribution for two independent samples. *Bull. Math. Univ. Moscou*, 2(2):3–14.

Stan Development Team (2012). *Stan Modeling Language User's Guide and Reference Manual, Version 1.0*. http://mc-stan.org/.

Stan Development Team (2020). RStan: the R interface to Stan. R package version 2.21.2.

Stöber, J. and Czado, C. (2017). Pair copula constructions. In *Series in Quantitative Finance*, pages 185–230. WORLD SCIENTIFIC.

Thomas, S. and Tu, W. (2021). Learning hamiltonian monte carlo in r. *The American Statistician*, 75(4):403–413.