Research Article **Open Access**
Special Issue: Salzburg Workshop on Dependence Models & Copulas

Thomas Nagler*, Christian Schellhase, and Claudia Czado

Nonparametric estimation of simplified vine copula models: comparison of methods

DOI 10.1515/demo-2017-0007
Received December 27, 2016; accepted May 16, 2017

Abstract: In the last decade, simplified vine copula models have been an active area of research. They build a high dimensional probability density from the product of marginals densities and bivariate copula densities. Besides parametric models, several approaches to nonparametric estimation of vine copulas have been proposed. In this article, we extend these approaches and compare them in an extensive simulation study and a real data application. We identify several factors driving the relative performance of the estimators. The most important one is the strength of dependence. No method was found to be uniformly better than all others. Overall, the kernel estimators performed best, but do worse than penalized B-spline estimators when there is weak dependence and no tail dependence.

Keywords: B-spline, Bernstein, copula, kernel, nonparametric, simulation, vine

1 Introduction

Simplified vine copulas, or pair-copula constructions (PCC), have become very popular over the last decade [1, 3, 4, 8, 37]. Vine copula models build a high-dimensional dependence structure by hierarchical modeling of bivariate copulas (the *pair-copulas*). Each pair-copula can be specified as a unique parametric copula function. Thus, simplified vine copula models give rise to very flexible models which are often found to be superior to other multivariate copula models [1, 12]. The models are also easily tractable because pair-copulas can be estimated sequentially. Parametric models for the pair-copulas are most common, but bear the risk of misspecification. In particular, most parametric families only allow for highly symmetric and monotone relationships between variables.

To remedy this issue, several nonparametric approaches have been proposed: penalized Bernstein polynomials and B-splines [21], kernel estimators [28], and a non-penalized Bernstein estimator [33]. A related contribution introduces the empirical pair-copula as an extension of the empirical copula [16], but does not aim at estimation of the vine copula density which is the focus of this article.

From a practitioner's point of view, the question arises: which method should I choose for a given data set? This question is difficult to answer theoretically because asymptotic approximations of nonparametric vine copula density estimators are prohibitively unwieldy, see Propositions 2 and 5 in [28]. In this article, we conduct an extensive simulation study to provide some guidance nevertheless. All estimation methods will be compared under several specifications of strength and type of dependence, sample size, and dimension, thereby covering a large range of practical scenarios.

***Corresponding Author: Thomas Nagler:** Department of Mathematics, Technische Universität München, Boltzmanstraße 3, 85748 Garching, E-mail: thomas.nagler@tum.de
Christian Schellhase: Centre for Statistics, Bielefeld University, Department of Business Administration and Economics, Germany, E-mail: cschellhase@wiwi.uni-bielefeld.de
Claudia Czado: Department of Mathematics, Technische Universität München, Boltzmanstraße 3, 85748 Garching, E-mail: cczado@ma.tum.de

Although our primary goal is to survey and compare existing methods, we extend the estimators proposed in [21, 28, 33] in several ways:

- The Bernstein and B-spline estimators of [21] and [33] are extended to allow for general R-vine structures (opposed to just D- and/or C-vine structures).
- Besides linear B-splines as in [21], we also consider quadratic B-splines.
- Beyond the classical kernel density estimator used in [28], we further consider local linear and local quadratic likelihood kernel estimators.
- All pair-copula estimators can be combined with structure selection algorithms using both Kendall's τ and a corrected AIC as target criterion.

The remainder of this article is organized as follows. Section 2 introduces simplified vine copula models. Section 3 presents and extends several existing nonparametric methods for pair-copula estimation, describes a step-wise algorithm for vine copula estimation, and discusses approaches for model selection. We describe the design of our simulation study in section 4 and summarize the results in section 5. In section 6, a real data set is used to illustrate the estimators' behavior and demonstrate the necessity for nonparametric estimators. Section 7 contains our conclusions.

2 Background on simplified vine copula models

This section gives a brief introduction to pair-copula constructions. For a more extensive treatment, we refer to [1, 8, 20].

By Sklar's theorem [36], any multivariate distribution function F can be split into its marginal distributions $F_1, \dots, F_d$ and a *copula* C:

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

The copula C describes the dependence structure of the random vector $\boldsymbol{X}$. It is, in fact, the joint distribution of the random vector $\boldsymbol{U} = (U_1, \dots, U_d) = \big(F_1(X_1), \dots, F_d(X_d)\big)$, if the distributions F_i are continuous for $i = 1, \dots, d$. Note that $U_1, \dots, U_d$ are uniformly distributed on the unit interval. If F admits a density with respect to the Lebesgue measure, we can differentiate the above equation to get

$$f(x_1, \dots, x_d) = c\big(F_1(x_1), \dots, F_d(x_d)\big) \times \prod_{k=1}^{d} f_k(x_k), \tag{1}$$

where $c, f_1, \dots, f_d$ are the probability density functions corresponding to C, $F_1, \dots, F_d$ respectively.

Any copula density c can be decomposed into a product of $d(d-1)/2$ bivariate (conditional) copula densities [2, 3, 19]. The decomposition is not unique, but all possible decomposition can be organized as graphical structure, called *regular vine (R-vine)*. It is a sequence of trees $T_m = (V_m, E_m)$, $m = 1, \dots, d-1$ satisfying the following conditions:

(i) T_1 is a tree with nodes $V_1 = \{1, \dots, d\}$ and edges E_1.
(ii) For $m \geq 2$, T_m is a tree with nodes $V_m = E_{m-1}$ and edges E_m.
(iii) (*Proximity condition*) Whenever two nodes in T_{m+1} are joined by an edge, the corresponding edges in T_m must share a common node.

Figure 1 shows an example of a regular vine with each edge e annotated by $(j_e, k_e; D_e)$. The notation for an edge e in T_i depends on the two shared edges in T_{i-1}, denoted by $a = (j_a, k_a; D_a)$ and $b = (j_b, k_b; D_b)$ with $\mathcal{V}_a = \{j_a, k_a, D_a\}$ and $\mathcal{V}_b = \{j_b, k_b, D_b\}$. Here D_e is a set of indices called conditioning set while $\{j_e, k_e\}$ is the conditioned set of an edge e. In Tree T_i, the nodes a and b are joined by edge $e = (j_e, k_e; D_e)$, with $j_e = \min\{l : l \in (\mathcal{V}_a \cup \mathcal{V}_b) \setminus D_e\}$, $k_e = \max\{l : l \in (\mathcal{V}_a \cup \mathcal{V}_b) \setminus D_e\}$ and $D_e = \mathcal{V}_a \cap \mathcal{V}_b$.

A vine copula is a graphical model describing the dependence of a d-variate random vector $\boldsymbol{U} = (U_1, \dots, U_d) \sim C$. The vine tree sequence is also called the *structure* of the vine copula model. This model

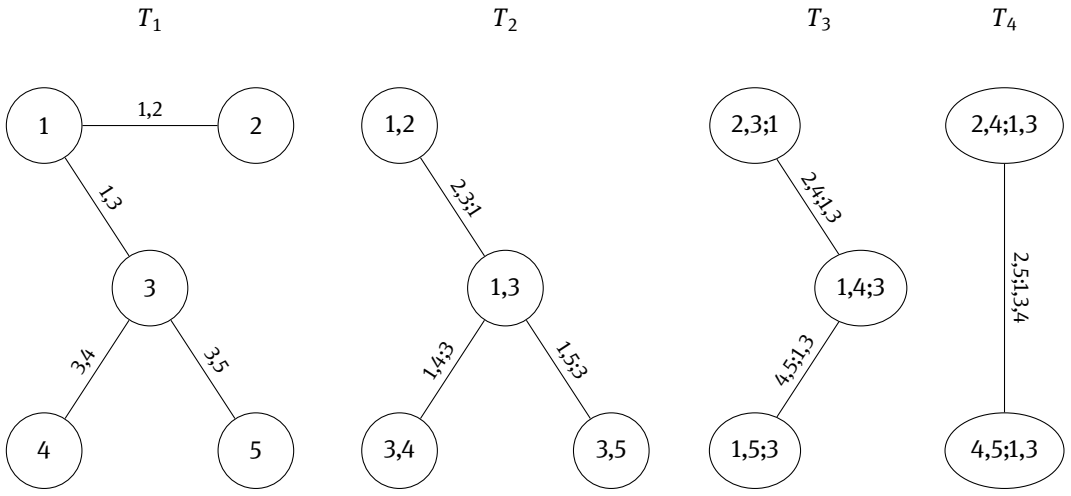

Figure 1: Example of a regular vine tree sequence.

identifies each edge e of the vine with a bivariate copula $c_{j_e,k_e;D_e}$ (called *pair-copula*). The joint density of the vine copula can then be written as the product of all pair-copula densities:

$$c(\boldsymbol{u}) = \prod_{m=1}^{d-1} \prod_{e \in E_m} c_{j_e,k_e;D_e}\big(C_{j_e|D_e}(u_{j_e}|\boldsymbol{u}_{D_e}), C_{k_e|D_e}(u_{k_e}|\boldsymbol{u}_{D_e}); \boldsymbol{u}_{D_e}\big), \tag{2}$$

where $\boldsymbol{u}_{D_e} := (u_\ell)_{\ell \in D_e}$ is a subvector of $\boldsymbol{u} = (u_1, \dots, u_d) \in [0,1]^d$ and $C_{j_e|D_e}$ is the conditional distribution of $U_{j_e}|\boldsymbol{U}_{D_e} = \boldsymbol{u}_{D_e}$. The pair-copula density $c_{j_e,k_e;D_e}$ is the copula density corresponding to the two variables U_{j_e} and U_{k_e}, conditional on $\boldsymbol{U}_{D_e} = \boldsymbol{u}_{D_e}$.

The density decomposition (2) holds for *any* copula density c. In this general form, the pair-copulas $c_{j_e,k_e;D_e}$ depend on the value of the conditioning vector $\boldsymbol{u}_{D_e}$. To make the model more tractable, one usually makes the *simplifying assumption* that the pair-copula densities do not change with $\boldsymbol{u}_{D_e}$. In this case, the model is called a simplified vine copula model and the corresponding density can be written as

$$c(\boldsymbol{u}) = \prod_{m=1}^{d-1} \prod_{e \in E_m} c_{j_e,k_e;D_e}\big(C_{j_e|D_e}(u_{j_e}|\boldsymbol{u}_{D_e}), C_{k_e|D_e}(u_{k_e}|\boldsymbol{u}_{D_e})\big).$$

Example 1. *The density of a simplified vine copula model corresponding to the tree sequence in Figure 1 is*

$$\begin{aligned} c(u_1, \dots, u_5) = {} & c_{1,2}(u_1, u_2) \times c_{1,3}(u_1, u_3) \times c_{3,4}(u_3, u_4) \times c_{3,5}(u_3, u_5) \\ & \times c_{2,3;1}(u_{2|1}, u_{3|1}) \times c_{1,4;3}(u_{1|3}, u_{4|3}) \times c_{1,5;3}(u_{1|3}, u_{5|3}) \\ & \times c_{2,4;1,3}(u_{2|1,3}, u_{4|1,3}) \times c_{4,5;1,3}(u_{4|1,3}, u_{5|1,3}) \\ & \times c_{2,5;1,3,4}(u_{2|1,3,4}, u_{5|1,3,4}), \end{aligned}$$

where we used the abbreviation $u_{j_e|D_e} := C_{j_e|D_e}(u_{j_e}|\boldsymbol{u}_{D_e})$.

Vine copula densities involve conditional distributions $C_{j_e|D_e}$. We can express them in terms of conditional distributions corresponding to bivariate copulas in the previous tree as follows: Let $l_e \in D_e$ be another index such that $c_{j_e,l_e;D_e\setminus l_e}$ is a pair-copula in the previous tree, and define $D_e' = D_e \setminus l_e$. Then, we can express

$$C_{j_e|D_e}(u_{j_e}|\boldsymbol{u}_{D_e}) = h_{j_e|l_e;D_e'}\big(C_{j_e|D_e'}(u_{j_e}|\boldsymbol{u}_{D_e'}) \,\big|\, C_{l_e|D_e'}(u_{l_e}|\boldsymbol{u}_{D_e'})\big), \tag{3}$$

where the *h-function* is defined as

$$h_{j_e|l_e;D_e'}(u_{j_e}|u_{l_e}) := \int_0^{u_{j_e}} c_{j_e,l_e;D_e'}(v, u_{l_e})dv = \frac{\partial C_{j_e,l_e;D_e'}(u_{j_e}, u_{l_e}|\boldsymbol{u}_{D_e'})}{\partial u_{l_e}}. \tag{4}$$

The arguments $C_{j_e|D_e'}(u_{j_e}|\boldsymbol{u}_{D_e'})$ and $C_{l_e|D_e'}(u_{l_e}|\boldsymbol{u}_{D_e'})$ of the h-function in (3) can be rewritten in the same manner. In each step of this recursion the conditioning set D_e is reduced by one element. Eventually, this allows us to write any of the conditional distributions $C_{j_e|D_e}$ as a recursion over h-functions that are directly linked to the pair-copula densities in previous trees.

3 Nonparametric estimation of simplified vine copula models

We now discuss how simplified vine copula models can be estimated nonparametrically. First, we give an overview of nonparametric estimators of bivariate copula densities. Second, we outline a general step-wise estimation algorithm for the full vine copula density, which can be used with any bivariate copula density estimator. We also describe a data-driven structure selection algorithm that was initially proposed in [10].

3.1 Nonparametric estimation of bivariate copula densities

The classical approach to density estimation is to assume a parametric model and estimate its parameters by maximum likelihood. There is a large variety of bivariate parametric copula models. Special classes are the elliptical copulas (including the Gaussian and Student t families), and the Archimedean class (including the Clayton, Frank and Gumbel families); for more see [20]. However, parametric models notoriously lack flexibility and bear the risk of misspecification. Nonparametric density estimators are designed to remedy these issues. In the context of copula densities, these estimators have to take the bounded support into account.

In the following we summarize the state-of-the-art of the major strands of nonparametric copula density estimation. For simplicity, we only consider the bivariate case. We assume throughout that we are given n observations $(U_1^{(i)}, U_2^{(i)})$, $i = 1, \dots, n$, from a copula density c that we want to estimate.

3.1.1 Empirical Bernstein copula

A classical tool in function approximation are Bernstein polynomials [24]. The normalized Bernstein polynomial of degree K is defined as

$$B_{Kk}(u) = (K+1)\binom{K}{k}(u)^k(1-u)^{K-k}, \quad \text{for } k = 0, \dots, K.$$

The collection of all Bernstein polynomials form a basis of the space of all square-integrable functions on $[0, 1]$. A natural idea is to approximate an arbitrary function by a linear combination of a finite number of basis functions. Based on this idea, the Bernstein copula density was defined in [32]. It is an approximation of the true copula density, and can be expressed as

$$\widetilde{c}(u_1, u_2) = \sum_{k_1=0}^{K}\sum_{k_2=0}^{K} B_{Kk_1}(u_1)B_{Kk_2}(u_2)\nu_{k_1,k_2},$$

where

$$\nu_{k_1,k_2} = \int_{k_1/\bar{K}}^{(k_1+1)/\bar{K}}\int_{k_2/\bar{K}}^{(k_2+1)/\bar{K}} c(u_1, u_2)du_1du_2.$$

and $\bar{K} = (K+1)$. Note that the coefficient ν_{k_1,k_2} describes the probability that $(U_1^{(i)}, U_2^{(i)})$ is contained in the cell $[k_1/\bar{K}, (k_1+1)/\bar{K}] \times [k_2/\bar{K}, (k_2+1)/\bar{K}]$. The empirical Bernstein copula density estimator is defined by

$\widetilde{c}(u_1, u_2)$, but replacing v_{k_1,k_2} by the empirical frequencies obtained from a contingency table:

$$\widehat{c}(u_1, u_2) = \sum_{k_1=0}^{K} \sum_{k_2=0}^{K} B_{Kk_1}(u_1) B_{Kk_2}(u_2) \widehat{v}_{k_1,k_2},$$

where

$$\widehat{v}_{k_1,k_2} = \frac{1}{n} \times \#\left\{(U_1^{(i)}, U_2^{(i)}) \in [k_1/\bar{K}, (k_1+1)/\bar{K}] \times [k_2/\bar{K}, (k_2+1)/\bar{K}]\right\},$$

which is the maximum-likelihood estimator for v_{k_1,k_2}.

The Bernstein copula density estimator was used in the context of vine copulas in [33]. As the marginal distributions of the Bernstein copula density do not need to be uniform, the authors calculate an approximation to the contingency table by solving a quadratic program, imposing constraints for uniform marginal distributions. The smoothing parameter for the Bernstein copula density estimator is K, the number of knots. Selection rules for K that adapt to the sample size and strength of dependence were proposed in [30]. Our implementation is available in the `kdecopula` R package [26], and uses the rule

$$K^{opt} = \lfloor n^{1/3} \exp(|\widehat{\rho}|^{1/n})(|\widehat{\rho}| + 0.1) \rfloor,$$

where $\widehat{\rho}$ is the empirical Spearman's ρ.

3.1.2 Penalized Bernstein polynomials and B-splines

For fixed K, the Bernstein copula density estimator is a parametric model with $(K+1)^2$ parameters. As any parametric model with many parameters, it is prone to overfitting. To gain control of the smoothness of the fit, a penalized likelihood approach was proposed in [21].

Viewing the Bernstein copula density as a parametric model with parameter vector $\boldsymbol{v} = (v_{00}, \ldots, v_{0K}, \ldots, v_{KK})$, i.e.,

$$\widetilde{c}(u_1, u_2; \boldsymbol{v}) = \sum_{k_1=0}^{K} \sum_{k_2=0}^{K} B_{Kk_1}(u_1) B_{Kk_2}(u_2) v_{k_1,k_2}, \tag{5}$$

we can estimate the parameters by maximizing the log-likelihood,

$$\ell(\boldsymbol{v}) = \log \sum_{i=1}^{n} \widetilde{c}\left(U_1^{(i)}, U_2^{(i)}; \boldsymbol{v}\right). \tag{6}$$

As each of the normalized Bernstein polynomials is a density, the weighted sum of normalized Bernstein polynomials is a density, if we ensure that

$$\sum_{k_1,k_2} v_{k_1,k_2} = 1, \qquad v_{k_1,k_2} \geq 0. \tag{7}$$

We will need additional constraints to enforce uniform marginal distributions: for Bernstein polynomials $\int \tilde{c}(u_1, u_2)\, \mathrm{d}u_1 \equiv 1$ holds if the marginal coefficients fulfill

$$v_{k_1.} = \sum_{k_2} v_{k_1,k_2} = 1/(K+1), \quad \text{for all } k_1 = 0, \ldots, K. \tag{8}$$

The same constraints follow for $\int \tilde{c}(u_1, u_2)\, \mathrm{d}u_2 \equiv 1$. These constraints can be formulated in matrix notation yielding

$$A_K^T \boldsymbol{v} = \mathbf{1}/(K+1) \tag{9}$$

where A_K sums up the elements of v_{k_1,k_2} column-wise (i.e. over k_2) and row-wise (i.e. over k_1), i.e. $A_K^T = ((I_K \otimes \mathbf{1}_K^T), (\mathbf{1}_K^T \otimes I_K))$, where $\mathbf{1}_K$ is the column vector of dimension K with elements 1 and I_K is the K dimensional identity matrix.

The log-likelihood (6) can be maximized under the constraints (7), (8) and (9), using quadratic programming (e.g., with the `quadprog` R package [40]). But since this is a parametric model with many parameters, the fitted copula density may be wiggly, see e.g., [39]. This issue can be resolved by imposing an appropriate penalty on the basis coefficients. We postulate that the integrated squared second order derivatives are small and formulate the penalty as

$$\int \left(\frac{\partial^2 \widetilde{c}(u_1, u_2; \boldsymbol{\nu})}{(\partial u_1)^2} \right)^2 + \left(\frac{\partial^2 \widetilde{c}(u_1, u_2; \boldsymbol{\nu})}{(\partial u_2)^2} \right)^2 \, \mathrm{d}u_1 \, \mathrm{d}u_2,$$

see also [21, 41]. This can be written as a quadratic form of a penalty matrix $\mathbf{P}$, see the Appendix of [21]. The corresponding penalized log-likelihood is defined as

$$\ell^p(\boldsymbol{\nu}, \lambda) = \ell(\boldsymbol{\nu}) - \frac{1}{2} \lambda \boldsymbol{\nu}^T \mathbf{P} \boldsymbol{\nu}, \tag{10}$$

which is again maximized with respect to (7), (8) and (9). The penalty parameter λ needs to be selected adequately, that is data driven. In section 2.5 of [21], the authors propose a method that formulates the penalized likelihood approach as linear mixed model and comprehend the penalty as normal prior imposed on the coefficient vector. We apply this methodology, too.

One can further use B-spline basis functions instead of Bernstein polynomials [21]. They replace each B_{Kk} in (5) with a B-spline, located at equidistant knots $\kappa_k = k/K$ with $k = 0, \dots, K$, normalized so that it satisfies $\int_0^1 B_{Kk}(u) \, \mathrm{d}u = 1$ for $k = 0, \dots, K - 1 + q$. In [21], only normalized linear ($q = 1$) B-splines were used. To allow for more flexibility, we will also use normalized quadratic ($q = 2$) B-splines in our study.

In order to guarantee that $\widetilde{c}(u_1, u_2; \boldsymbol{\nu})$ is a bivariate copula density, we impose similar constraints as the ones for the Bernstein polynomials. The linear constraints (7) will be the same for B-splines, but the uniform margins condition (8) has to be adapted. The condition takes the form $A_K \boldsymbol{\nu} = \mathbf{1}$ with $A_K = \boldsymbol{B}_K(\kappa)$, choosing

$$\kappa = \begin{cases} \kappa_0, \dots, \kappa_K, & \text{for linear B-splines,} \\ 0, \frac{\kappa_1 - \kappa_0}{2} + \kappa_0, \frac{\kappa_2 - \kappa_1}{2} + \kappa_1, \dots, \frac{\kappa_{K+1} - \kappa_K}{2} + \kappa_K, 1, & \text{for quadratic B-splines.} \end{cases}$$

For the penalization, we work with a penalty on the m-th order differences of the spline coefficients $\boldsymbol{\nu}$, as suggested for B-spline smoothing in [11], defining a penalty matrix $\mathbf{P^m}$, where we choose $m = q + 1$. Further details of this smoothing concept can be found in [31]. In the following, we define the difference based penalty matrix $\mathbf{P^m}$ for the m-order differences through

$$\mathbf{P^m} := (\mathbf{1}_{K+q} \otimes L_m)^T (L_m \otimes \mathbf{1}_{K+q}). \tag{11}$$

Let $L_m \in \mathbb{R}^{K+q-m \times K+q}$ be a difference matrix of order m, e.g., for $q = 1$ we get $m = 2$ and

$$L_2 = \begin{pmatrix} 1 & -2 & 1 & 0 & \cdots & 0 \\ 0 & 1 & -2 & 1 & \ddots & \vdots \\ \vdots & \ddots & \ddots & \ddots & \ddots & 0 \\ 0 & \cdots & 0 & 1 & -2 & 1 \end{pmatrix} \in \mathbb{R}^{K-1 \times K+1}.$$

Then for B-splines, the penalized log-likelihood becomes

$$l^p(\boldsymbol{\nu}, \lambda) = l(\boldsymbol{\nu}) - \frac{1}{2} \lambda \boldsymbol{\nu}^T \mathbf{P^m} \boldsymbol{\nu}. \tag{12}$$

Note, that we achieve an independence copula, if we set the penalty parameter λ to infinity in (10) or (12). The penalized Bernstein and B-splines estimators are implemented in the R package `penRvine` [34].

3.1.3 Kernel weighted local likelihood

Kernel estimators are well-established tools for nonparametric density estimation. Several kernel methods have been tailored to the problem of copula density estimation. Their main challenge is to avoid bias and

consistency issues at the boundaries of the support. The earliest contribution is the mirror-reflection method [15]. Later, the beta kernel density estimator of [7] was extended to the bivariate case in [6].

The more recent contributions all focus on a transformation trick. Assume we want to estimate a copula density c given a random sample $\left(U_1^{(i)}, U_2^{(i)}\right), i = 1, \dots, n$. Let Φ be the standard normal distribution function and ϕ its density. Then the random vectors $(Z_1^{(i)}, Z_2^{(i)}) = \left(\Phi^{-1}(U_1^{(i)}), \Phi^{-1}(U_2^{(i)})\right)$ have normally distributed margins and are supported on the full $\mathbb{R}^2$. In this domain, kernel density estimators work very well and do not suffer from any boundary problems. By Sklar's Theorem for densities (1), the density f of $(Z_1^{(i)}, Z_2^{(i)})$ decomposes to

$$f(z_1, z_2) = c\big(\Phi(z_1), \Phi(z_2)\big)\phi(z_1)\phi(z_2), \qquad \text{for all } (z_1, z_2) \in \mathbb{R}. \tag{13}$$

By isolating c in (13) and the change of variables $u_j = \Phi(z_j)$, $j = 1, 2$, we get

$$c(u_1, u_2) = \frac{f\big(\Phi^{-1}(u_1), \Phi^{-1}(u_2)\big)}{\phi\big(\Phi^{-1}(u_1)\big)\phi\big(\Phi^{-1}(u_2)\big)}. \tag{14}$$

We can use any kernel estimator $\widehat{f}$ of f to define a kernel estimator of the copula density c via (14):

$$\widehat{c}(u_1, u_2) = \frac{\widehat{f}\big(\Phi^{-1}(u_1), \Phi^{-1}(u_2)\big)}{\phi\big(\Phi^{-1}(u_1)\big)\phi\big(\Phi^{-1}(u_2)\big)}. \tag{15}$$

Estimators of this kind have an interesting feature. The denominator of (15) vanishes when u_1 or u_2 tend to zero or one. If the numerator vanishes at a slower rate, the estimated copula density explodes towards the corners of the unit square. This behavior is common for many popular parametric families, including the Gauss, Student, Gumbel, and Clayton families. The transformation estimator (15) is well suited to resemble such shapes. However, its variance will also explode towards the corners and the estimator will be numerically unstable. To accommodate for this, we restrict the estimator to $[0.001, 0.999]^2$ and set estimates outside of this region to the closest properly defined estimate.

To estimate the density f, the classical bivariate kernel density estimator was used in [28]. We will extend this approach by resorting to the more general class of local polynomial likelihood estimators; see [23] for a general account and [13] in the context of bivariate copula estimation.

Assume that the log-density $\log f(z_1, z_2)$ of the random vector $\boldsymbol{Z}^{(i)} = (Z_1^{(i)}, Z_2^{(i)})$ can be approximated locally by a polynomial of order q. For example, using a log-quadratic expansion, we get

$$\begin{aligned}\log f(z_1', z_2') &\approx P_{\boldsymbol{a}}(\boldsymbol{z}) \\ &= a_1 + a_2(z_1 - z_1') + a_3(z_2 - z_2') + a_4(z_1 - z_1')^2 + a_5(z_1 - z_1')(z_2 - z_2') + a_6(z_2 - z_2')^2\end{aligned}$$

for (z_1', z_2') in the neighborhood of $\boldsymbol{z} = (z_1, z_2)$. The polynomial coefficients $\boldsymbol{a}$ can be found by solving the weighted maximum likelihood problem

$$\widehat{\boldsymbol{a}} = \arg\max_{\boldsymbol{a} \in \mathbb{R}^6} \left\{ \sum_{i=1}^{n} \boldsymbol{K}\big(B^{-1}(\boldsymbol{z} - \boldsymbol{Z}^{(i)})\big) P_{\boldsymbol{a}}(\boldsymbol{z} - \boldsymbol{Z}^{(i)}) - n \int_{\mathbb{R}^2} \boldsymbol{K}\big(B^{-1}(\boldsymbol{z} - \boldsymbol{s})\big) \exp\big(P_{\boldsymbol{a}}(\boldsymbol{z} - \boldsymbol{s}) d\boldsymbol{s}\right\},$$

where the kernel K is a symmetric probability density function, $\boldsymbol{K}(\boldsymbol{z}) = K(z_1)K(z_2)$ is the product kernel, and $B \in \mathbb{R}^{2\times 2}$ is a matrix with $\det(B) > 0$. B is called the bandwidth matrix and controls the degree of smoothing. The kernel K serves as a weight function that localizes the above optimization problem around $\boldsymbol{z}$.

We obtain $\widehat{a}_1$ as an estimate for $\log f(z_1, z_2)$ and, consequently, $\exp(\widehat{a}_1)$ as an estimate for $f(z_1, z_2)$. An estimate of the copula density can be obtained by plugging this estimate in (14). For a detailed treatment of this estimator's asymptotic behavior we refer to [13]. In general, the estimator does not yield a *bona fide* copula density because the margins may not be uniform. This issue can be resolved by normalizing the density estimate, for details see [27].

Input: Observations $(U_1^{(i)}, \dots, U_d^{(i)})$, $i = 1, \dots, n$, vine structure $(E_1, \dots, E_{d-1})$.
Output: Estimates of pair-copula densities and h-functions required to evaluate the vine copula density (16).

---

for $m = 1, \dots, d-1$:
  for all $e \in E_m$:
  (i) Based on $\big(\widehat{U}^{(i)}_{j_e|D_e}, \widehat{U}^{(i)}_{k_e|D_e}\big)_{i=1,\dots,n}$, obtain an estimate of the copula density $c_{j_e,k_e;D_e}$ which we denote as $\widehat{c}_{j_e,k_e;D_e}$.
  (ii) Derive corresponding estimates of the h-functions $\widehat{h}_{j_e|k_e;D_e}$, $\widehat{h}_{k_e|j_e;D_e}$ by integration (eq. (4)).
  (iii) Set

$$\widehat{U}^{(i)}_{j_e|D_e\cup k_e} := \widehat{h}_{j_e|k_e;D_e}\big(\widehat{U}^{(i)}_{j_e|D_e}\big|\widehat{U}^{(i)}_{k_e|D_e}\big),$$
$$\widehat{U}^{(i)}_{k_e|D_e\cup j_e} := \widehat{h}_{k_e|j_e;D_e}\big(\widehat{U}^{(i)}_{k_e|D_e}\big|\widehat{U}^{(i)}_{j_e|D_e}\big), \quad i = 1, \dots, n.$$

  end for
end for

Algorithm 1: Sequential estimation of simplified vine copula densities

For applications of the estimator, an appropriate choice of the bandwidth matrix is crucial. For the local constant approximation, a simple rule of thumb was shown to perform well in [27]. We use an improved version of this rule that also adjusts to the degree of the polynomial q:

$$B_{\text{rot}} = \nu_q n^{-1/(4q^\star+2)}\widehat{\Sigma}_{\boldsymbol{Z}}^{1/2}, \quad q^\star = 1 + \lfloor q/2 \rfloor,$$

where $\widehat{\Sigma}_{\boldsymbol{Z}}$ is the empirical covariance matrix of $\boldsymbol{Z}^{(i)}$, $i = 1, \dots, n$, and $\nu_0 = 1.25$, $\nu_1 = \nu_2 = 5$. An implementation of the estimator is available in the R package `kdecopula` [26].

3.2 Step-wise estimation of vine copula densities

We now turn to the question how a simplified vine copula density can be estimated. Most commonly, this is done in a sequential procedure introduced in [1]. The procedure is generic in the sense that it can be used with any consistent estimator for a bivariate copula. It is summarized in algorithm 1.

From now on we use c to denote a d-dimensional vine copula density. Assume we have a random sample $\boldsymbol{U}^{(i)} = \big(U_1^{(i)}, \dots, U_d^{(i)}\big)$, $i = 1, \dots, n$, from c. Recall that this density can be written as

$$c(\boldsymbol{u}) = \prod_{m=1}^{d-1}\prod_{e\in E_m} c_{j_e,k_e;D_e}(C_{j_e|D_e}(u_{j_e}|\boldsymbol{u}_{D_e}), C_{k_e|D_e}(u_{k_e}|\boldsymbol{u}_{D_e})). \tag{16}$$

In the first tree, the conditioning set D_e is empty. So for $e \in E_1$, estimation of the pair-copula densities $c_{j_e,k_e;D_e}$ is straightforward, since no conditioning is involved. We simply apply one of the estimators from subsection 3.1 to the bivariate random vectors $(U_{j_e}^{(i)}, U_{k_e}^{(i)})$. This gives us estimates $\widehat{c}_{j_e,k_e;D_e}$, $e \in E_1$. By one-dimensional integration (eq. (4)) we can derive estimates of the corresponding h-functions. They can be derived in closed form for Bernstein and B-spline estimators, see [21]. For kernel estimators, the h-functions have to be computed numerically.

In a next step, we transform the initial copula data by applying the estimated h-functions to obtain pseudo-observations from the pair-copulas in the second tree. Using these, we can estimate the pair-copula densities $c_{j_e,k_e;D_e}$, $e \in E_2$. We iterate through the trees in this manner until all pair-copula densities and h-functions have been estimated.

Theorem 1 in [28] shows that simplified vine copula density estimators defined by algorithm 1 are consistent under rather mild conditions. An appealing property of these estimators is the absence of curse of dimensionality: the convergence rate does not depend on the dimension. In fact, the convergence rate achieves the optimal rate for a two-dimensional nonparametric density estimator.

Input: Observations $(U_1^{(i)}, \dots, U_d^{(i)})$, $i = 1, \dots, n$.
Output: Vine structure $(E_1, \dots, E_{d-1})$ and estimates of pair-copula densities and h-functions required to evaluate the vine copula density (16).

---

for $m = 1, \dots, d-1$:

Calculate weights w_e for all possible edges $e = \{j_e, k_e; D_e\}$ that satisfy the proximity condition (see section 2) and select the edge set E_m as

$$E_m = \arg\max_{E_m^\star} \sum_{e \in E_m^\star} w_e,$$

under the constraint that $E_m^\star$ corresponds to a spanning tree.
for all $e \in E_m$:

(i) Based on $\big(\widehat{U}^{(i)}_{j_e|D_e}, \widehat{U}^{(i)}_{k_e|D_e}\big)_{i=1,\dots,n}$, obtain an estimate of the copula density $c_{j_e,k_e;D_e}$ which we denote as $\widehat{c}_{j_e,k_e;D_e}$.

(ii) Derive corresponding estimates of the h-functions $\widehat{h}_{j_e|k_e;D_e}$, $\widehat{h}_{k_e|j_e;D_e}$ by integration (eq. (4)).

(iii) Set

$$\widehat{U}^{(i)}_{j_e|D_e\cup k_e} := \widehat{h}_{j_e|k_e;D_e}\big(\widehat{U}^{(i)}_{j_e|D_e} \big| \widehat{U}^{(i)}_{k_e|D_e}\big),$$
$$\widehat{U}^{(i)}_{k_e|D_e\cup j_e} := \widehat{h}_{k_e|j_e;D_e}\big(\widehat{U}^{(i)}_{k_e|D_e} \big| \widehat{U}^{(i)}_{j_e|D_e}\big), \quad i = 1, \dots, n.$$

end for
end for

Algorithm 2: Sequential estimation and structure selection for simplified vine copula models

3.3 Selection strategies for the vine structure

So far we assumed that the structure of the vine (i.e., the edge sets $E_1, \dots, E_{d-1}$) is known. In practice, however, the structure has to be chosen by the statistician. This choice is very difficult, since there are $d!/2 \times d^{(d-2)(d-3)/2}$ possible vine structures [25], which grows excessively with d. When d is very small, it may still be practicable to estimate vine copula models for all possible structures and compare them by a suitable criterion (such as AIC). But already for a moderate number of dimensions one has to rely on heuristics.

A selection algorithm that seeks to capture most of the dependence in the first couple of trees was proposed in [10]. This is achieved by finding the maximum spanning tree using a dependence measure as edge weights, e.g., the absolute value of the empirical Kendall's τ. The resulting estimation and structure selection procedure is summarized in a general form in algorithm 2.

Several specifications of the edge weight were investigated in a fully parametric context in [9]. The most common edge weight w_e is the absolute value of the empirical Kendall's τ. It was proposed in [10] and used in a non-parametric context in [28]. On the other hand, a *corrected Akaike information criterion (cAIC)* [17] was used as edge weight w_e in [21]. When using the cAIC criterion in Algorithm 2, the weight w_e for edge e is

$$\mathrm{cAIC}_e = -2\ell_e + 2\mathrm{df}_e + \frac{2\mathrm{df}_e(\mathrm{df}_e + 1)}{n - \mathrm{df}_e - 1}, \tag{17}$$

where

$$\ell_e = \sum_{i=1}^{n} \ln \widehat{c}_{j_e,k_e;D_e}\big(\widehat{U}^{(i)}_{j_e|D_e}, \widehat{U}^{(i)}_{k_e|D_e}\big),$$

is the log-likelihood and df_e is the *effective degrees of freedom (EDF)* of the estimator $\widehat{c}_e$. For explicit formulas for the EDF we refer to [21] for the spline approach and to [23] for the kernel estimators. For parametric copula estimation, the EDF equals the number of estimated parameters for the chosen copula family.

From a computational point of view, the cAIC has a big disadvantage: before a tree can be selected, the pair-copulas of all possible edges in this tree have to be estimated. Just for the first tree, this amounts to estimating $\binom{d}{2}$ bivariate copula densities. The empirical Kendall's τ on the other hand can be computed rapidly

for all pairs. It allows to select the tree structure before any pair-copula has been estimated. Then, only $d-1$ pair-copulas have to be estimated in the first tree. The situation is similar for subsequent trees. Both approaches will be compared in our simulation study with regard to estimation accuracy and speed.

4 Description of the simulation study design

We compare the performance of the vine copula density estimators discussed in section 3 over a wide range of scenarios. We consider several specifications of sample size, dimension, strength of dependence, and tail dependence. We randomize the simulation models and characterize the scenarios by probability distributions for the pair-copula families and dependence parameters. A detailed description of the study design procedure will be given in the following sections.

4.1 Simulation scenarios based on model randomization

To investigate how various factors influence the estimators' performance, we create a number of scenarios. Each of these scenarios is characterized by a combination of the factors shown in Table 1.

To make the results for a particular dependence scenario as general as possible, we randomly generate a model in the following steps:

Step 1. **Draw R-vine structure:**
We do this in the following steps:

(i) Draw n samples for d independent uniform random variables, $\tilde{U}_{i,j}$, $i = 1, \dots, n$, $j = 1, \dots, d$.
(ii) On these samples, run the structure selection algorithm of [10] (only allowing for the independence family).
(iii) Set the model structure to the one selected by the algorithm.

Step 2. **Draw pair-copula families:**
- *only tail dependent copulas*: draw each of the $d(d-1)/2$ pair-copula families with equal probabilities from the Student t- ($df = 4$), Gumbel (with rotations) and Clayton (with rotations) copulas.
- *no tail dependence*: draw each of the $d(d-1)/2$ pair-copula families with equal probabilities from the Gaussian and Frank copulas.
- *both*: for each of $d(d-1)/2$ pair-copulas:
  (i)choose with equal probabilities whether the copula has tail dependence or not,
  (ii)proceed as above.

Step 3. **Draw pair-copula parameters:**
For each pair-copula:

(i) Randomly generate the absolute value of Kendall's τ from the following distributions:
  - *weak dependence:* $Beta(1, 4)$-distribution ($E[|\tau|] = 0.2$),
  - *strong dependence:* $Beta(5, 5)$-distribution ($E[|\tau|] = 0.5$).

  The densities are shown in Figure 2.
(ii) Randomly choose the sign of Kendall's τ as $Bernoulli(0.5)$ variable.
(iii) Usually, partial dependence is weaker than direct pair-wise dependence. To mimic this behavior we decrease the simulated absolute Kendall's τ by a factor of 0.8^m, where m is the tree level of the pair-copula.

Table 1: List of factors that determine the set of simulation scenarios.

Dimension d	Sample Size n	Type of dependence	Strength of dependence
5	400	only tail dependence	weak
10	2 000	no tail dependence	strong
		both types	

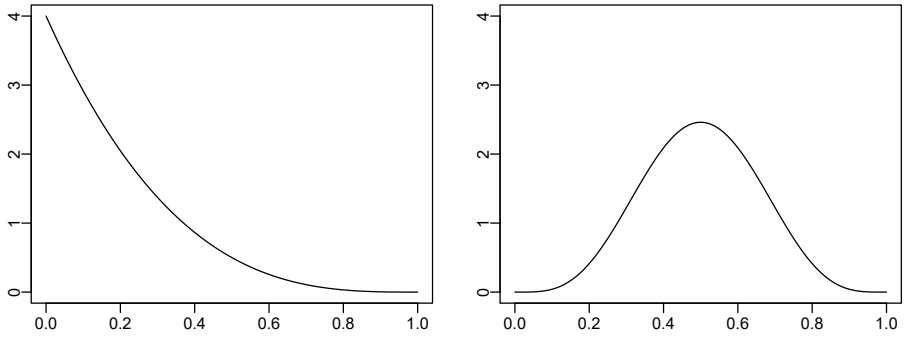

Figure 2: Densities for the simulation of absolute Kendall's τ in the scenarios with weak (left) and strong (right) dependence.

(iv) For all families under consideration there is a one-to-one relationship between the copula parameter and Kendall's τ, see e.g., Table 2 in [5]. Hence, we set the copula parameter by inversion of the reduced value of Kendall's τ.

Step 4. **Draw observations from the final model:**
With the selected structure, copula families and their parameters, the vine copula model is fully specified. We can draw random samples from this vine copula model using the algorithm of [38], as implemented in the `VineCopula` R library [35].

The stochastic model characterized by steps 1–4 can be interpreted as a whole. It is a mixture of vine copula models, mixed over its structure, families, and parameter. The mixing distribution for the families is uniform over sets determined by the 'type of dependence' hyper-parameter. The mixing distribution for the absolute Kendall's τ follows a Beta distribution with parameters characterized by the 'strength of dependence' hyper-parameter. Each scenario corresponds to a particular specification of the mixture's hyperparameters. The benefit of this construction is that it yields models that are representative for a wide range of scenarios encountered in practice. It also limits the degrees of freedom we would have when specifying all pair-copula families and parameters manually.

4.2 Estimation methods

We compare the following pair-copula estimators:

- `par`: parametric estimator as implemented in the function `BiCopSelect` of the R package `VineCopula` [35]. It estimates the parameters for several parametric families and selects the best model based on AIC. The implemented families are: Independence, Gaussian, Student t, Clayton, Frank, Gumbel, Joe, BB1, BB6, BB7, BB8, Tawn types I and II,
- `bern`: non-penalized Bernstein estimator (see subsubsection 3.1.1),
- `pbern`: penalized Bernstein estimator (see subsubsection 3.1.1) with $K = 14$ knots,
- `pspl1`: penalized linear B-spline estimator (see subsubsection 3.1.2) with $K = 14$,

- `pspl2`: penalized quadratic B-spline estimator (see subsubsection 3.1.2) with $K = 10$,
- `tll0`: transformation local likelihood kernel estimator of degree $q = 0$ (see subsubsection 3.1.3),
- `tll1`: transformation local likelihood kernel estimator of degree $q = 1$ (see subsubsection 3.1.3),
- `tll2`: transformation local likelihood kernel estimator of degree $q = 2$ (see subsubsection 3.1.3).

We further implemented two structure selection methods for each estimation pair-copula estimator (based on Kendall's τ and cAIC, see subsection 3.3); additionally we computed each estimator under the true structure.

4.3 Performance measurement

As a performance measure, we choose the *integrated absolute error (IAE)*

$$\text{IAE} = \int_{[0,1]^d} |\widehat{c}(\boldsymbol{u}) - c(\boldsymbol{u})| d\boldsymbol{u},$$

where $\widehat{c}$ is the estimated and c is the true copula density. The above expression requires us to calculate a d-dimensional integral, which can be difficult when d becomes large. To overcome this, we estimate this integral via importance sampling Monte Carlo, see e.g., section 5.2 in [29]. That is,

$$\widehat{\text{IAE}} = \frac{1}{N} \sum_{i=1,\dots,N} \frac{|\widehat{c}(\boldsymbol{U}_i) - c(\boldsymbol{U}_i)|}{c(\boldsymbol{U}_i)},$$

where $\boldsymbol{U}_i \overset{iid}{\sim} c$ is a random vector drawn from the true copula density c. This results in an unbiased estimator of the IAE with relatively small variance: usually the numerator is large/small when the denominator is large/small. Hence, the variance of the terms of the sum is small and, thereby, the variance of the sum is small. All results will be based on an importance sample of size $N = 1\,000$.

For each estimator and each possible simulation scenario emerging from Table 1, we record the $\widehat{\text{IAE}}$ on $R = 100$ simulated data sets.

5 Results

Figure 3 and Figure 4 present the results of the simulation study described in section 4. The analysis will be divided into several sections. The first takes a very broad view, whereas the remaining ones investigate the influence of individual factors. We acknowledge that the information density in the figures is extremely high. So we start with a detailed description of the figures' layout.

Figure 3 contains the results for all scenarios with weak dependence; Figure 4 with strong dependence. The left columns correspond to the smaller sample size ($n = 400$) and the right columns to the larger sample size ($n = 2\,000$). The figures are also partitioned row-wise with an alternating pattern of the dimensions $d = 5$ and $d = 10$. Two subsequent rows correspond to the same type of dependence (no tail dependence, both, only tail dependence). In total there are 32 panels, each representing one of the 32 possible combinations of the factors listed in Table 1.

Each panel contains 24 boxes in 8 groups. Each group corresponds to one estimation method for the pair-copulas. The three boxes in each group represent the three different methods for structure selection: known structure, maximum spanning trees with Kendall's τ, maximum spanning trees with cAIC (from left to right). The box spans the interquartile range, the median is indicated by a horizontal line, the whiskers represent the 10% and 90% percentiles.

5.1 Overall ranking of methods for pair-copula estimation

We begin our analysis with a broad view on the relative performance of the pair-copula estimators. We want to assess the performance of the estimation methods, averaged over all scenarios and structure selection

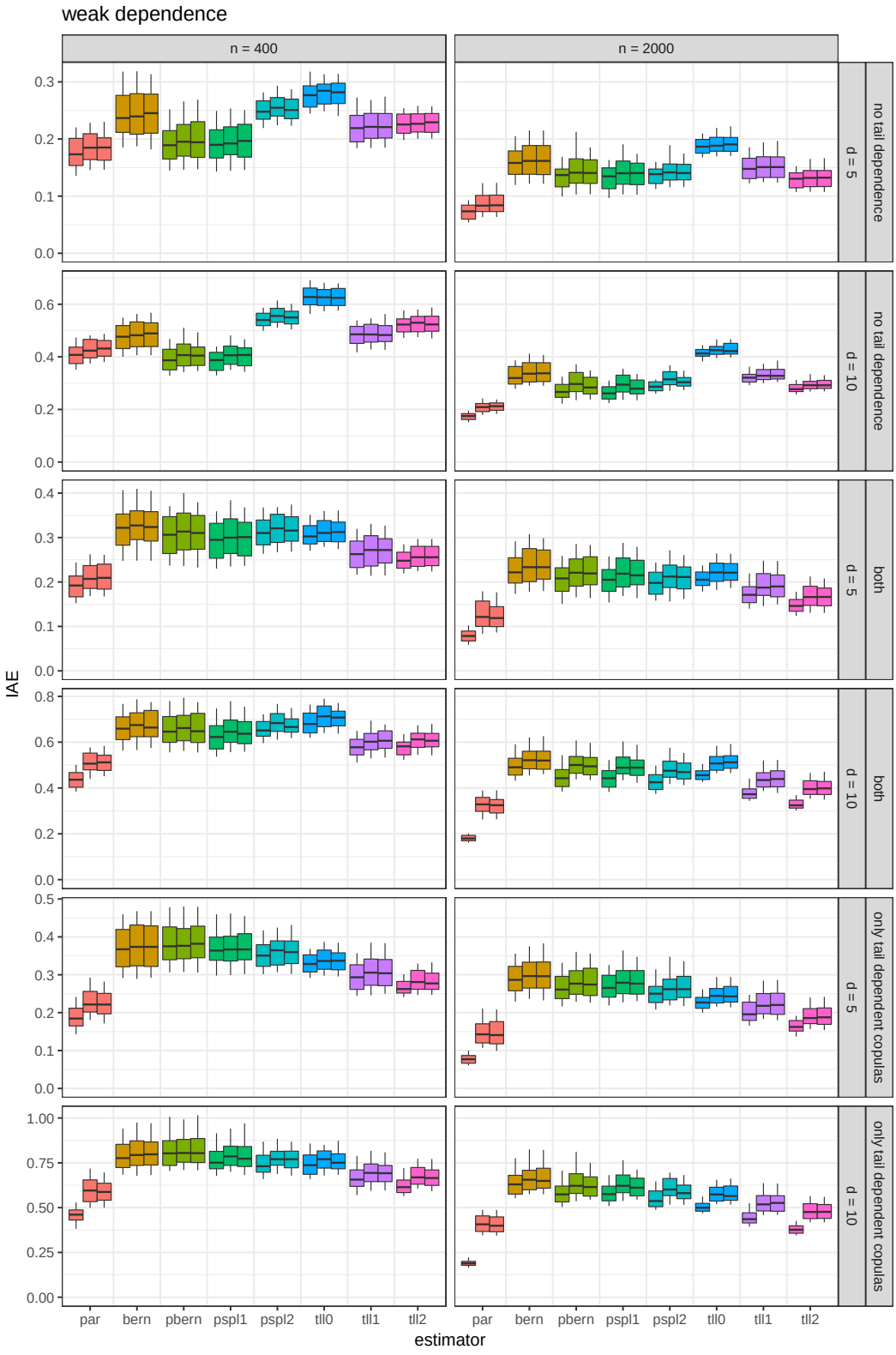

Figure 3: *weak dependence:* the box plots show the IAE achieved by each estimation method. Results are split by sample size, dimension, and type of dependence. Per estimator there are three boxes, corresponding to estimation under known structure, selection by Kendall's τ, and selection by cAIC (from left to right).

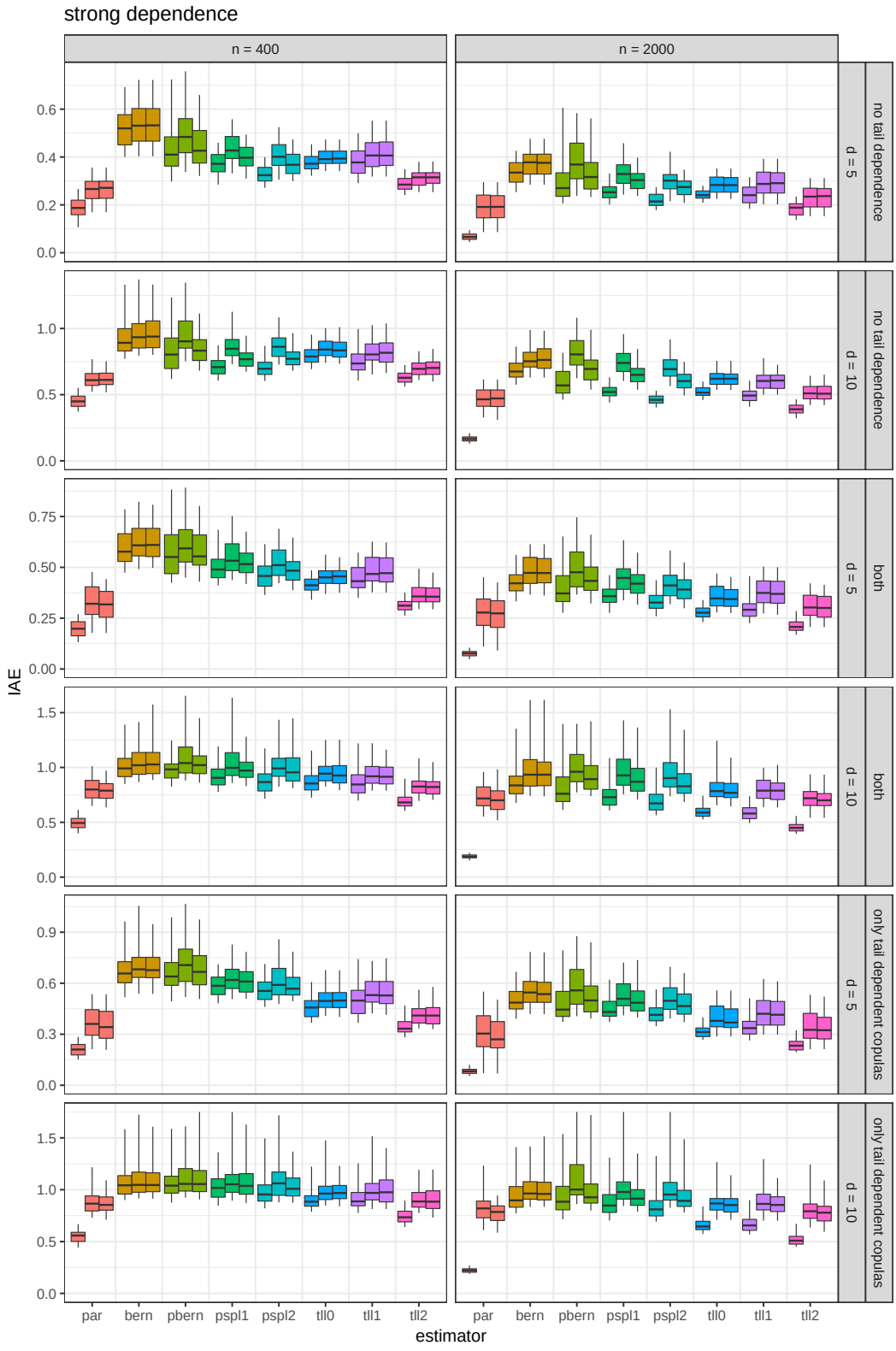

Figure 4: *strong dependence:* the box plots show the IAE achieved by each estimation method. Results are split by sample size, dimension, and type of dependence. Per estimator there are three boxes, corresponding to estimation under known structure, selection by Kendall's τ, and selection by cAIC (from left to right).

Table 2: The relative rank of estimators averaged over all scenarios.

	par	bern	pbern	pspl1	pspl2	tll0	tll1	tll2
average rank	1.28	7.17	6.35	5.00	5.01	5.01	3.83	2.35

strategies. But just taking the average IAE could be misleading. It is evident from Figures 3 and 4 that the scale of the IAE varies between scenarios. Averaging the bare IAE leads to an unbalanced few, laying more weight on particular scenarios. As a more robust alternative, we take the following approach: in each scenario, average the IAE over replications and structure selection strategies. Then rank the estimation methods by their relative performance. Ranks are comparable across scenarios, so our final criterion will be the average rank across all scenarios. These numbers are listed in Table 2.

The parametric estimator performs best, which is no surprise since our simulation models consist of only parametric copula families. We included it in this study mainly to get a sense of what is possible in each scenario. Remarkably, it is outperformed in very few cases by a nonparametric estimator. This is due to the need for structure selection which will be discussed in more detail later on.

Among the nonparametric estimators, the kernel estimators (`tll2`, `tll1`, `tll0`) perform best, followed by the spline methods (`pspl1`, `pspl2`) which perform as well as the worst kernel estimator `tll0`. The Bernstein estimators (`pbern`, `bern`) perform worst. Within these three classes, the accuracy improves mostly by how complex the estimation method is: going from regular Bernstein copulas to penalized ones; and going from local constant, to local linear, to local quadratic likelihood. It is the other way around for the B-spline methods, but the difference in the average rank is minuscule.

We will find that this relative ranking is fairly robust across scenarios. In the following analysis, we treat it as the benchmark ranking and focus on deviations from it.

5.2 Strength and type of dependence

By looking at the scale in each panel, we see that the performance of all estimators gets worse for increasing strength of dependence and increasing proportion of tail dependent families. This is explained by the behavior of the true densities. Many copula densities (and their derivatives) explode at a corner of the unit square. From the pair-copula families in our simulation model, only the Frank copula is bounded. Within each family, the tails explode faster when the strength of dependence increases. And tail dependence means that the tails explode particularly fast. Exploding curves are difficult to estimate for nonparametric estimators because their asymptotic bias and variance are usually proportional to the true densities' derivatives. Our results give evidence that this effect transfers to finite samples.

The estimators' response to these difficulties is the main driver behind their relative performance. In most scenarios, the ranking of estimators is similar to the benchmark rankings. But there are deviations. Let us walk through the scenarios one by one.

- *weak, no tail dependence*: `pbern1` and `pspl1` perform better than `pspl2`, the kernel estimators, and even the parametric estimator for $n = 400$. For $n = 2\,000$, the parametric estimator gets ahead and the penalized methods are on par with `tll1` and `tll2`.
- *weak, both*: `pbern1` and `pspl1` perform better than `pspl2` and `tll0` for $n = 400$, and comparable for $n = 2\,000$.
- *weak, only tail dependent copulas*: similar to the benchmark ranking.
- *strong, no tail dependence*: `bspl2` beats `tll0` and `tll1` for $n = 400$ and is on par for $n = 2\,000$.
- *strong, both*: similar to benchmark ranking.
- *strong, only tail dependent copulas*: similar to benchmark ranking.

Overall, the penalized estimators tend to do better under weak dependence and only little tail dependence, whereas the kernel estimators do better in the other scenarios. The method `tll2` is the top performer in all but a few cases.

5.3 Sample size and dimension

When the sample size increases, the estimators become more accurate. Any reasonable estimator should satisfy this property. The kernel estimators and the non-penalized Bernstein estimator seem to benefit more from an increased sample size. The effect is most obvious in the weak dependence, no tail dependence case. This has an explanation: theoretically, the number of knots used by the penalized estimators should increase with the sample size. But our implementation uses a fixed number of knots, as the computational burden is already substantial compared with the other methods (see subsection 5.5). All other methods adapt their smoothing parameterization to the sample size. It is very likely that the penalized methods improve when the number of knots is further increased.

Comparing a pair of panels corresponding to $d = 5$ and $d = 10$, we see very little differences. We conclude that the results are robust to changes in the dimensionality.

5.4 Structure selection

The first aspect we want to discuss is the loss in accuracy caused by the need to select the tree structure. Recall that the three subsequent boxes for each estimator correspond to: estimation under the true structure (in practice unknown), selection based on Kendall's τ, selection based on cAIC.

The IAEs for the two selection methods are always higher than the 'oracle' results with known structure. This makes sense: the true model is a simplified vine copula; if the true structure is known, the models are correctly specified and all estimators are consistent. In practice, the true structure is unknown, and a different structure will be selected most of the time. For the selected structure, there is no guarantee that the model is still simplified or that the estimators are consistent. For more details, we refer to [37] and section 8 in [28].

Overall, the average loss in accuracy when going from the true to a heuristically selected structure increases with strength of dependence and prevalence of tail dependence. But the extent of this effect varies between estimation methods. The parametric estimator suffers the most substantial losses. In fact, the parametric estimator's performance is often very close to that of the best nonparametric estimator when the structure is unknown. This is quite remarkable considering that we simulate from parametric models. Interestingly, the loss for the penalized Bernstein and B-spline methods (`pbern`, `pspl1`, `pspl2`) is negligible in most scenarios when cAIC is used—but not when Kendall's τ is used. This is a distinct property of these penalized methods. The non-penalized Bernstein and kernel methods perform similarly for the two structure selection criteria. In most scenarios, the relative performance ordering of the estimators is the same for each type of structure. But there are a few cases (strong dependence, n = 400) where the `bspl2` estimator is worse than `tll0` or `tll1` with Kendall's τ, but better with cAIC.

The results give evidence that the cAIC is the better criterion in terms of the estimators' accuracy. But it also makes the vine copula estimators more costly to fit (see subsection 3.3). So there is a trade-off between speed and accuracy. It usually depends on the problem at hand which to prioritize. We will investigate this issue further in the next section.

5.5 Computation time

Table 3 lists the average computation time[1] required to fit a vine copula and evaluate its density on 1 000 importance Monte-Carlo samples. The results are divided into the combinations of dimension d and sample size n.

Let us first focus on the selection criterion. We clearly see that the computation time increases substantially for all estimators when cAIC is used instead of Kendall's τ. This effect size differs, but is usually a factor of around two or three.

1 The time was recorded on a single thread of a 8-way Opteron (Dual-Core, 2.6 GHz) CPU with 64GB RAM.

Table 3: Average computation time (in seconds) required for estimation and selection of one vine copula model.

d	n	criterion	par	bern	pbern	pspl1	pspl2	tll0	tll1	tll2
5	400	τ	7	3	788	758	517	3	4	6
		cAIC	19	10	1 000	1 175	786	10	11	13
	2 000	τ	34	19	1 578	1 455	1 394	7	12	16
		cAIC	91	31	2 163	2 336	2 243	25	32	35
10	400	τ	33	17	2 983	3 183	2 205	14	19	29
		cAIC	98	49	5 292	6 110	4 156	48	55	65
	2 000	τ	159	65	6 553	6 694	6 515	35	56	71
		cAIC	472	139	11 992	13 514	12 394	127	158	173

The fastest two estimators are the simplest ones: `bern` and `tll0`. The other two kernel estimators are in the same ballpark, but the computation time increases slightly with the order of the polynomial. Only slightly slower is the parametric estimator. The reason is that the parametric estimator has to iterate through several different copula families before it can select the final model. The penalized estimators are roughly two orders of magnitude slower than their competitors. Take for example the case $d = 10$ and $n = 2\,000$, where most estimators take around one minute (using τ), but the penalized estimators take more than one and a half hours.

The large difference in computational demand is caused by the penalized estimation problem. One has to optimize over more than 100 parameters with more than 100 side constraints. Even worse, such a problem has to be solved multiple times until an optimal choice for the penalty parameter λ has been found. Reducing K (the number of knots) does significantly reduce this burden, but also limits the flexibility of the estimators. In the end, the statistician has to choose which K yields the best balance between speed and accuracy.

5.6 Limitations

The referees pointed out some limitations of our study which are addressed in the following.

5.6.1 Performance measure

All results focus on a single performance measure and therefore only provide a limited view on the estimators' performance. Although this is true, we considered several other measures in preliminary versions of this study and found the results to be quite robust with respect to the measure.

5.6.2 Estimation of marginal distributions

The study neglects the fact that observations from the copula are never observed and one has to rely on pseudo-observations that depend on estimated marginal distributions. An extensive simulation study in [22] revealed that this can be a problem in misspecified fully parametric models. But the issue is largely resolved when the margins are estimated nonparametrically. In this case, maximum likelihood estimators are unbiased and only slightly less efficient [14].

In a purely nonparametric context, it is even less of an issue. In fact, many authors have found that errors stemming from estimating the marginal distributions are asymptotically negligible when estimating the copula density, see e.g., [13, 18, 27]. This is explained by the fact that distribution functions can be estimated consistently at the parametric rate, whereas density estimators are bound to (slower) nonparametric rates. Accordingly, we can expect similar results to the ones presented even if margins were treated unknown.

5.6.3 Choice of smoothing parameters

It is a common theme in nonparametric estimation that the quality of estimators depends heavily on the choice of smoothing parameters. This is certainly also the case for the estimators considered in this study. However, we do not think it is feasible to assess the sensitivity of our results to this choice:

- Smoothing parameters are hardly comparable across estimation methods because they arrive at the density estimate in fundamentally different ways.
- There are too many smoothing parameters in a vine copula model: There are 10 ($d = 5$) resp. 45 ($d = 10$) pair-copulas, and for each pair-copula there are between one and three smoothing parameters (depending on the estimation method).
- Due to the sequential nature of the joint estimator, pair-copula estimators in later trees are affected by the estimates in earlier trees. This leads to significant interactions between smoothing parameters at different levels.

In our study, all smoothing parameters were selected by automatic procedures that are state-of-the-art. This realistically reflects statistical practice. But one should keep in mind that the performance of most estimators can likely be improved by advances in automatic selection procedures.

6 Illustration with real data

In the simulation study, the parametric estimator performed best in virtually all scenarios. But this is simply a consequence of simulating from parametric models. Real data do not always behave that nicely and nonparametric methods are required to appropriately capture the dependence. Such a case is illustrated in the following real-data example.

We consider a data set representative of measurements taken on images from the MAGIC (Major Atmospheric Gamma-ray Imaging Cherenkov) Telescopes (https://archive.ics.uci.edu/ml/datasets/MAGIC+Gamma+Telescope) with 19 020 observations for eleven different attributes, but focus only on gamma observations.

To show exemplary results of the different nonparametric copula density estimators, we select a random subset ($n = 2\,000$) from the MAGIC data with respect to to the three variables fConc1, fM3Long and fM3Trans. We compute pseudo-observations from the data by applying the marginal empirical distribution functions to each variable. Figure 5 shows scatter plots of the three pairs of the pseudo-observations. The shapes we see are different from what we know from popular parametric families. We fit several copula density estimators to each pair and show the results in Figure 6. The first column of Figure 6 shows the fitted pair-copula density between fConc1 (U_5) and fM3Long (U_7), the second column between fConc1 (U_5) and fM3Trans (U_8) and the third column contains the copula density between fM3Long (U_7) and fM3Trans (U_8).

The first pair of variables fConc1 (U_5) and fM3Long (U_7) a lot of pseudo-observations accumulate around the point $(0, 1)$, which is reflected as high density peaks in all fitted copula densities. But for the accumulation around the point $(1, 0.3)$, we observe a difference between the nonparametric estimators `bern`, `pspl2` and `ttl2` and the parametric copula density, which does not mirror this accumulation.

For the second pair, fConc1 (U_5) and fM3Trans (U_8), the estimated density varies considerably between methods. Estimates of `pspl2` and `ttl2` show peaks around the points $(0, 0)$ and $(0, 1)$, which reflects the large concentration of points in the scatter plot in Figure 5. The estimators `bern` and `par` do not contain these peaks. We observe similar differences for the estimated densities for the third data pair, presented in the right column of Figure 5. While `bern`, `pspl2` and `ttl2` show density peaks around the accumulation points $(1, 0)$ and $(1, 1)$, but the estimated parametric copula does not exhibit these structures of the data.

The previous examples have illustrated situations, in which parametric estimator fails because of its lack of flexibility. In such situations, nonparametric methods are required to adequately capture the true dependence structure. However, for illustrations, we merely looked at two unconditional pairs of variables, not a full dependence model.

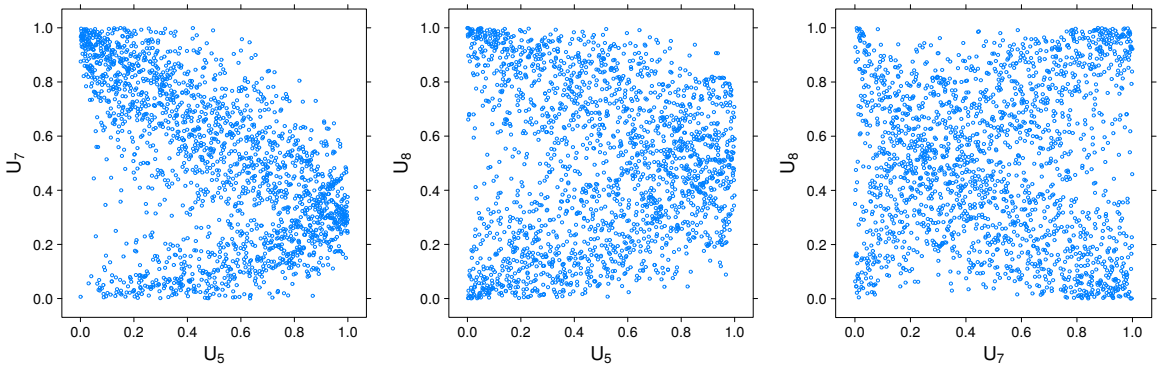

Figure 5: Scatterplots of pseudo observation ranks for pairs (left) fConc1 (U_5) and fM3Long (U_7), (middle) fConc1 (U_5) and fM3Trans (U_8) and (right) fM3Long (U_7) and fM3Trans (U_8) from the MAGIC data set ($n = 2\,000$).

To analyze the the performance of the estimators in an application, we estimate nonparametric vine copulas for the complete MAGIC data set with eleven attributes focusing on gamma observations. Because the true density is unknown in applications, we assess the performance of the estimators via cross-validation. Similar to our simulation study, we randomly draw a subset $\boldsymbol{U}_{\text{train}}$ of the data of sizes $n = 400, 2\,000$, apply the estimators, and calculate the mean out-of-sample log-likelihood on 1 000 randomly selected remaining observations $\boldsymbol{U}_{\text{test}}$, i.e.,

$$\ell(\boldsymbol{U}_{\text{test}}) = \frac{1}{1\,000} \sum_{i=1}^{1\,000} \ln \hat{c}(\boldsymbol{U}_{\text{test}}^{(i)}),$$

where $\hat{c}$ is a vine copula density estimator based on $\boldsymbol{U}_{\text{train}}$. This is repeated $N = 100$ times for sample sizes $n = 400$ and $n = 2\,000$. The results are summarized as box plots in Figure 7 for all estimators and structure selection based on Kendall's τ (left box) and cAIC (right box).

The parametric estimator performs unsatisfactory for $n = 400$ since it varies enormously for both data sets. But also for $n = 2\,000$, the parametric estimator is outperformed by most nonparametric alternatives. The performance of the nonparametric methods varies notably between methods. The methods `bern` and `tll1` do not perform well, but the other methods clearly outperform `par`. Furthermore, the performance differs significantly with respect to the structure selection criterion for `bern1` and `pspl1, pspl2`. For small sample size ($n = 400$) and using Kendall's τ as selection criterion, `tll0` results with highest mean, directly followed by `pspl2`. But choosing cAIC instead, the log-likelihood of `pspl2` increases, but not for `tll0`. The situation is similar for $n = 2\,000$. We conclude that the more sophisticated nonparametric methods adequately reflect the distribution of the data. In contrast, the dependence structure observed in Figure 5 can not be captured adequately with standard parametric models.

7 Conclusion

This articled compared existing methods for nonparametric estimation of simplified vine copula densities. The estimators considered are the non-penalized Bernstein estimator, the penalized Bernstein estimator, penalized B-spline estimators (linear and quadratic), and kernel weighted local likelihood estimators (local constant, linear, and quadratic). We compared these methods by an extensive simulation study and on two real data sets.

The simulation study comprises several scenarios for sample size, dimension, strength of dependence, and tail dependence. The simulation models are set up as parametric vine copulas with randomized vine structure, pair-copula families, and parameters. Overall, the kernel methods were found to perform best (especially the local quadratic version), followed by the penalized B-spline estimators. The Bernstein esti-

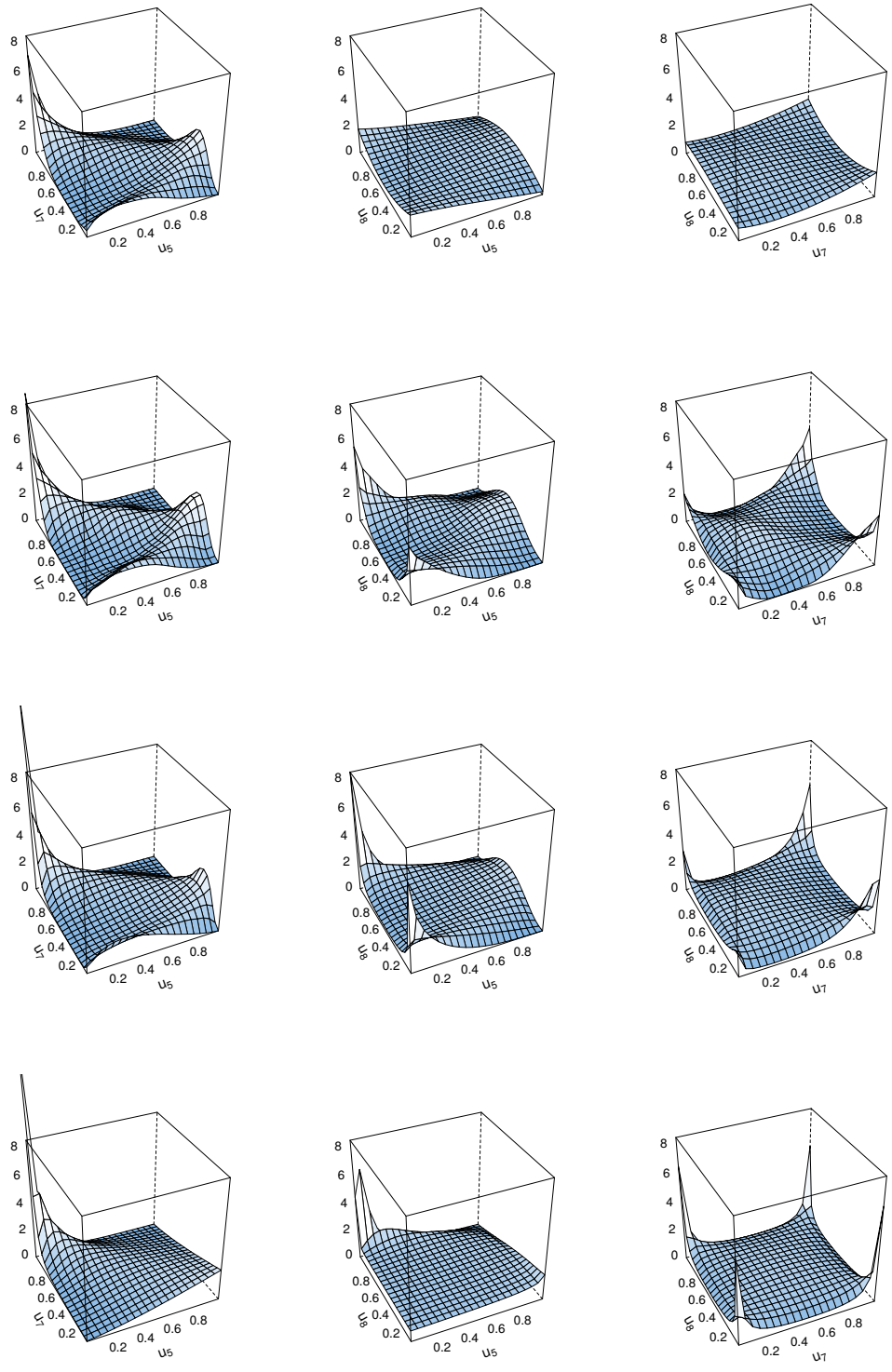

Figure 6: Exemplary density plots for MAGIC data ($n = 2\,000$). 1st row: Bernstein estimator `bern`, 2nd row: penalized quadratic B-splines estimator `pspl2`, 3rd row: kernel estimator `tll2`, 4th row: parametric estimator `par`.

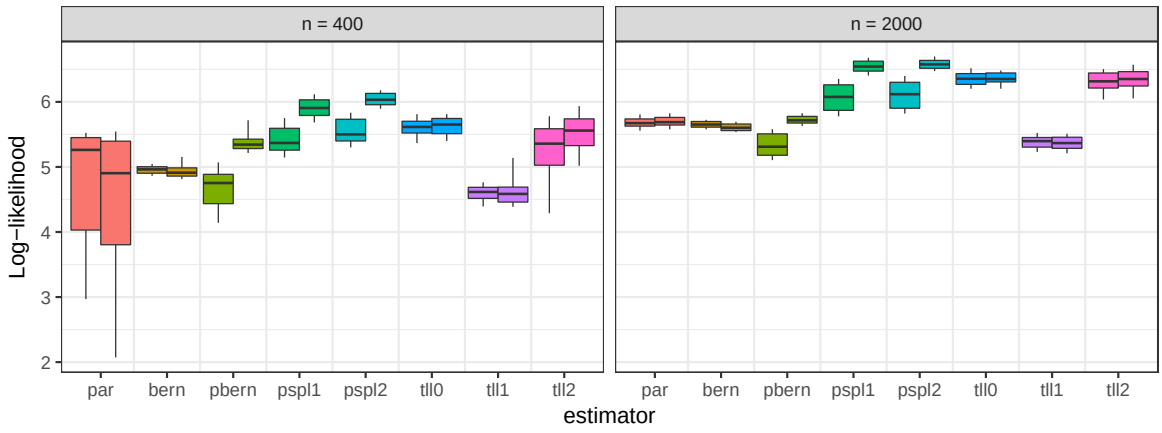

Figure 7: The box plots show the mean log-likelihood values attained by the different estimation methods. Each boxplot on the left hand side: structure selection based on Kendell's τ and each boxplot on the right hand side: structure selection based on cAIC.

mators performed worst. An exception to this pattern was found in scenarios with small sample size, weak dependence, and no tail dependence. Here, the penalized B-spline and Bernstein estimators outperformed the kernel methods. Additionally, we demonstrated the need for nonparametric methods on real data whose dependence structure cannot be adequately captured by a the parametric estimator.

Overall, we found that no estimator is uniformly better than the others; it depends on the data which is to be preferred. Our analysis highlighted which factors drive the performance of the various methods, and which methods should be preferred for certain scenarios. In applications, statisticians can determine the characteristics of their data by an exploratory analysis, and make a well-informed choice based on these results.

Acknowledgement: The first and third author were partially supported by the German Research Foundation (DFG grants CZ 86/5-1 and CZ 86/4-1). The study was executed on a computing cluster to which the Leibniz Supercomputing Centre (www.lrz.de) kindly provided access. The authors thank the guest editors and two anonymous referees whose remarks considerably improved our contribution.

References

[1] Aas, K., C. Czado, A. Frigessi, and H. Bakken (2009). Pair-copula constructions of multiple dependence. *Insurance Math. Econom. 44*(2), 182–198.

[2] Bedford, T. and R. M. Cooke (2001). Probability density decomposition for conditionally dependent random variables modeled by vines. *Ann. Math. Artif. Intell. 32*(1-4), 245–268.

[3] Bedford, T. and R. M. Cooke (2002). Vines - A new graphical model for dependent random variables. *Ann. Statist. 30*(4), 1031–1068.

[4] Brechmann, E. C., C. Czado, and K. Aas (2012). Truncated regular vines in high dimensions with application to financial data. *Canad. J. Statist. 40*(1), 68–85.

[5] Brechmann, E. C. and U. Schepsmeier (2013). Modeling dependence with C- and D-vine copulas: The R package CDVine. *J. Stat. Softw. 52*(3), 1–27.

[6] Charpentier, A., J.-D. Fermanian, and O. Scaillet (2006). The estimation of copulas: Theory and practice. In J. Rank (Ed.), *Copulas: From Theory to Application in Finance*, pp. 35–62. Risk Books, London.

[7] Chen, S. X. (1999). Beta kernel estimators for density functions. *Comput. Statist. Data Anal. 31*(2), 131–145.

[8] Czado, C. (2010). Pair-copula constructions of multivariate copulas. In P. Jaworski, F. Durante, W. K. Härdle, and T. Rychlik (Eds.), *Copula Theory and its Applications*, pp. 93–109. Springer, Heidelberg.

[9] Czado, C., S. Jeske, and M. Hofmann (2013). Selection strategies for regular vine copulae. *J. SFdS 154*(1), 174–191.

[10] Dißmann, J., E. C. Brechmann, C. Czado, and D. Kurowicka (2013). Selecting and estimating regular vine copulae and application to financial returns. *Comput. Statist. Data Anal. 59*(1), 52–69.
[11] Eilers, P. H. C. and B. D. Marx (1996). Flexible smoothing with B-splines and penalties. *Statist. Sci. 11*(2), 89–121. With comments and a rejoinder by the authors.
[12] Fischer, M., C. Köck, S. Schlüter, and F. Weigert (2009). An empirical analysis of multivariate copula models. *Quant. Finance 9*(7), 839–854.
[13] Geenens, G., A. Charpentier, and D. Paindaveine (2017). Probit transformation for nonparametric kernel estimation of the copula density. *Bernoulli 23*(3), 1848–1873.
[14] Genest, C., K. Ghoudi, and L.-P. Rivest (1995). A semiparametric estimation procedure of dependence parameters in multivariate families of distributions. *Biometrika 82*(3), 543–552.
[15] Gijbels, I. and J. Mielniczuk (1990). Estimating the density of a copula function. *Comm. Statist. - Theory and Methods 19*(2), 445–464.
[16] Hobæk Haff, I. and J. Segers (2015). Nonparametric estimation of pair-copula constructions with the empirical pair-copula. *Comput. Statist. Data Anal. 84*, 1–13.
[17] Hurvich, C. M., J. S. Simonoff, and C.-L. Tsai (1998). Smoothing parameter selection in nonparametric regression using an improved Akaike information criterion. *J. R. Stat. Soc. Ser. B Stat. Methodol. 60*(2), 271–293.
[18] Janssen, P., J. Swanepoel, and N. Veraverbeke (2014). A note on the asymptotic behavior of the Bernstein estimator of the copula density. *J. Multivariate Anal. 124*, 480–487.
[19] Joe, H. (1996). Families of m-variate distributions with given margins and $m(m-1)/2$ bivariate dependence parameters. In L. Rüschendorf, B. Schweizer, and M. D. Taylor (Eds.), *Distributions with Fixed Marginals and Related Topics*, pp. 120–141. Inst. Math. Statist., Hayward CA.
[20] Joe, H. (2015). *Dependence Modeling with Copulas*. CRC Press, Boca Raton FL.
[21] Kauermann, G. and C. Schellhase (2014). Flexible pair-copula estimation in D-vines using bivariate penalized splines. *Stat. Comput. 24*(6), 1081–1100.
[22] Kim, G., M. J. Silvapulle, and P. Silvapulle (2007). Comparison of semiparametric and parametric methods for estimating copulas. *Comput. Statist. Data Anal. 51*(6), 2836–2850.
[23] Loader, C. (1999). *Local Regression and Likelihood*. Springer-Verlag, New York.
[24] Lorentz, G. G. (1953). *Bernstein Polynomials*. Univ. of Toronto Press, Toronto.
[25] Morales-Nápoles, O., R. Cooke, and D. Kurowicka (2011). Counting vines. In D. Kurowicka and H. Joe (Eds.), *Dependence Modeling: Vine Copula Handbook*, pp. 189-218. World Scientific Publishing, Singapore.
[26] Nagler, T. (2016). *kdecopula: Kernel Smoothing for Bivariate Copula Densities*. R package version 0.8.0. Available on CRAN.
[27] Nagler, T. (2017). kdecopula: An R Package for the Kernel Estimation of Bivariate Copula Densities. Available at https://arxiv.org/abs/1603.04229.
[28] Nagler, T. and C. Czado (2016). Evading the curse of dimensionality in nonparametric density estimation with simplified vine copulas. *J. Multivariate Anal. 151*, 69–89.
[29] Ripley, B. D. (1987). *Stochastic Simulation*. John Wiley & Sons, Chichester.
[30] Rose, D. (2015). *Modeling and Estimating Multivariate Dependence Structures with the Bernstein Copula*. Ph. D. thesis, Ludwig-Maximilians Universität München.
[31] Ruppert, D., M. P. Wand, and R. J. Carroll (2003). *Semiparametric Regression*. Cambridge University Press, Cambridge.
[32] Sancetta, A. and S. Satchell (2004). The Bernstein copula and its applications to modeling and approximations of multivariate distributions. *Econometric Theory 20*(3), 535–562.
[33] Scheffer, M. and G. N. F. Weiß(2017). Smooth nonparametric Bernstein vine copulas. *Quant. Finance 17*(1), 139–156.
[34] Schellhase, C. (2016). *penRvine: Pair-Copula Estimation in R-Vines using Bivariate Penalized Splines*. R package version 0.2. Available on CRAN.
[35] Schepsmeier, U., J. Stoeber, E. C. Brechmann, B. Graeler, T. Nagler, and T. Erhardt et al. (2017). *VineCopula: Statistical Inference of Vine Copulas*. R package version 2.1.2. Available on CRAN.
[36] Sklar, A. (1959). Fonctions de répartition à n dimensions et leurs marges. *Publ. Inst. Statist. Univ. Paris 8*, 229–231.
[37] Spanhel, F. and M. S. Kurz (2015). Simplified vine copula models: Approximations based on the simplifying assumption. Available at https://arxiv.org/abs/1510.06971.
[38] Stöber, J. and C. Czado (2012). Sampling pair copula constructions with applications to mathematical finance. In J.-F. Mai and M. Scherer (Eds.), *Simulating Copulas: Stochastic Models, Sampling Algorithms, and Applications*. World Scientific Publishing, Singapore.
[39] Wahba, G. (1990). *Spline Models for Observational Data*. SIAM, Philadelphia, PA.
[40] Weingessel, A. (2013). *quadprog: Functions to solve Quadratic Programming Problems*. R package version 1.5-5. Available on CRAN.
[41] Wood, S. N. (2006). *Generalized Additive Models: An Introduction with R*. Chapman and Hall/CRC, Boca Raton FL.