Gaussian Process Vine Copulas for Multivariate Dependence

David Lopez-Paz DAVID.LOPEZ@TUE.MPG.DE
Max Planck Institute for Intelligent Systems

Jose Miguel Hernández-Lobato JMH233@ENG.CAM.AC.UK
Zoubin Ghahramani ZOUBIN@ENG.CAM.AC.UK
University of Cambridge

Abstract

Copulas allow to learn marginal distributions separately from the multivariate dependence structure (copula) that links them together into a density function. *Vine factorizations* ease the learning of high-dimensional copulas by constructing a hierarchy of conditional bivariate copulas. However, to simplify inference, it is common to assume that each of these conditional bivariate copulas is independent from its conditioning variables. In this paper, we relax this assumption by discovering the latent functions that specify the shape of a conditional copula given its conditioning variables. We learn these functions by following a Bayesian approach based on sparse Gaussian processes with expectation propagation for scalable, approximate inference. Experiments on real-world datasets show that, when modeling all conditional dependencies, we obtain better estimates of the underlying copula of the data.

1. Introduction

Copulas are becoming a popular approach in machine learning to describe multivariate data (Elidan, 2012; Kirshner, 2007; Elidan, 2010; Wilson & Ghahramani, 2010). Estimating multivariate densities is difficult due to possibly complicated forms of the data distribution and the curse of dimensionality. Copulas simplify this process by separating the learning of the marginal distributions from the learning of the multivariate dependence structure, or *copula*, that links them together into a density model (Joe, 2005). Learning the marginals is easy and can be done using standard univariate methods. However, learning the copula is more difficult and requires models that can represent a broad range of dependence patterns. For the two-dimensional case, there exists a large collection of parametric copula models (Nelsen, 2006). However, in higher dimensions, the number and expressiveness of families of parametric copulas is more limited. A solution to this problem is given by pair copula constructions, vine copulas or simply *vines* (Bedford & Cooke, 2002; Kurowicka & Cooke, 2006). These are graphical models that decompose any multivariate copula into a hierarchy of bivariate copulas, where some of them will be conditioned on a subset of the data variables. The deeper a bivariate copula is in the vine hierarchy, the more variables it will be conditioned on. If the conditional dependencies described above are ignored, vines are a straightforward approach to construct flexible high-dimensional dependence models using standard parametric bivariate copulas as building blocks.

Proceedings of the 30th International Conference on Machine Learning, Atlanta, Georgia, USA, 2013. JMLR: W&CP volume 28. Copyright 2013 by the author(s).

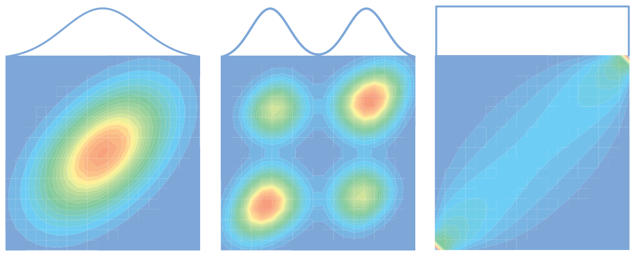


Figure 1. Two bidimensional densities (left, middle) that share the same underlying Gaussian copula, with correlation parameter $\theta = 0.8$ (right). The two distributions differ because of their marginal distributions, depicted at the top of each density plot.

The impact of ignoring conditional dependencies in the copula functions is likely to be problem specific. Hobaek et al. (2010) show thorough experiments with synthetic data that, in specific cases, ignoring conditional dependencies can lead to reasonably accurate approximations of the true copula. By contrast, Acar, Genest and Neslehova (2012) indicate that this *simplifying assumption* can be in other cases misleading, and develop a method to condition parametric bivariate copulas on a single scalar variable. In this paper, we extend the work of Acar et al. (2012) and propose a general technique to construct arbitrary vine models with full conditional parametric bivariate copulas. Our results on several real-world datasets show that it is often important to take into account conditional dependencies when constructing a vine model.

The proposed method is based on the fact that most parametric bivariate copulas can be specified in terms of Kendall's rank correlation parameter $\tau \in [-1, 1]$ (Joe, 1997). The dependence of the copula on a vector of conditioning variables $\mathbf{u} = (u_1, \ldots, u_d)^\mathrm{T}$ is then captured by specifying the relationship $\tau = \sigma(f(\mathbf{u}))$, where $f : \mathbb{R}^d \to \mathbb{R}$ is a non-linear function and $\sigma : \mathbb{R} \to [-1, 1]$ is a scaling operation. We follow a Bayesian approach to learn f from available data. In particular, we place a Gaussian process (GP) prior on f and use expectation propagation for approximate inference (Rasmussen & Williams, 2006; Minka, 2001). To make our method scalable, we use sparse GPs based on the generalized FITC approximation (Snelson & Ghahramani, 2006; Naish-Guzman & Holden, 2007).

2. Copulas and Vines

When the components of a d-dimensional random vector $\mathbf{x} = (x_1, \ldots, x_d)^\mathrm{T}$ are independent, their density function $p(\mathbf{x})$ can be factorized as

$$p(\mathbf{x}) = \prod_{i=1}^{d} p(x_i)\,. \tag{1}$$

The previous equality does not hold when $x_1, \ldots, x_d$ are not independent. Nevertheless, the differences can be corrected by multiplying the right hand side of (1) by a specific function that fully describes any possible form of dependence between the random variables $x_1, \ldots, x_d$. This function is called the *copula* of $p(\mathbf{x})$ (Nelsen, 2006), and satisfies:

$$p(\mathbf{x}) = \left[\prod_{i=1}^{d} p(x_i)\right] \underbrace{c(P(x_1), ..., P(x_d))}_{\text{copula}}, \tag{2}$$

where $P(x_i)$ is the marginal cumulative distribution function (cdf) of the random variable x_i. The copula c is the joint multivariate density of $P(x_1), \ldots, P(x_d)$ and it has uniform marginal distributions, since $P(x) \sim \mathcal{U}[0, 1]$ for any random variable x (Casella & Berger, 2001). This non-linear transformation from x to $P(x)$ is known as the Probability Integral Transform (PIT). The copula is the density of $\mathbf{x}$ after eliminating all the marginal information by applying the PIT to each individual component of $\mathbf{x}$. Therefore, c describes any dependence patterns which do not depend on the marginal distributions. If every $P(x_i)$ is continuous, then c is unique for any $p(\mathbf{x})$ (Sklar, 1959). However, infinitely many multivariate distributions share the same underlying copula (Figure 1).

The main advantage of copulas is that they separate the learning of univariate marginal distributions from the learning of the multivariate dependence structure that describes how they are coupled (Joe, 2005). Learning the marginals is easy and can be done using standard univariate methods. However, learning the copula is more difficult and requires models that can represent a broad range of dependence patterns. For the two-dimensional case, a large collection of parametric copula models is available (Nelsen, 2006). Some examples are the Gaussian, Student, Clayton, Independent, Gumbel or Frank copulas. Each of these families describes a different dependence structure between two random variables. An intuitive example is the copula that describes independence, that is, the independent copula: it has density constant and equal to one, as one can infer from equations (1) and (2). The Appendix contains more on the bivariate Gaussian Copula, which is used extensively used throughout this paper.

Although there exist many parametric models for two-dimensional copulas, for more than two dimensions the number and expressiveness of families of parametric copulas is more limited. A solution to this problem is given by pair copula constructions, vine copulas or simply *vines* (Joe, 1996; Bedford & Cooke, 2002; Kurowicka & Cooke, 2006).

2.1. Regular Vines

Vine copulas are hierarchical graphical models that factorize a d-dimensional copula density into a product of $d(d-1)/2$ bivariate conditional copula densities. They offer great modeling flexibility, since each of the bivariate copulas in the factorization can belong to a different parametric family. Several types of vines have been proposed in the literature. Some examples are canonical vines (C-Vines), drawable vines (D-vines) or regular vines (R-Vines). In this paper we focus on regular vines, since they are a generalization of all the other types (Dissmann et al., 2012).

An R-vine $\mathcal{V}$ specifies a factorization of a copula density $c(u_1, \dots, u_d)$ into a product of bivariate conditional copulas. Such R-vine is constructed by forming a nested set of $d-1$ undirected trees, in which each of their edges corresponds to a conditional bivariate copula density. A particular nested set of trees identifies a particular valid factorization of $c(u_1, \dots, u_d)$. These trees can be sequentially constructed as follows:

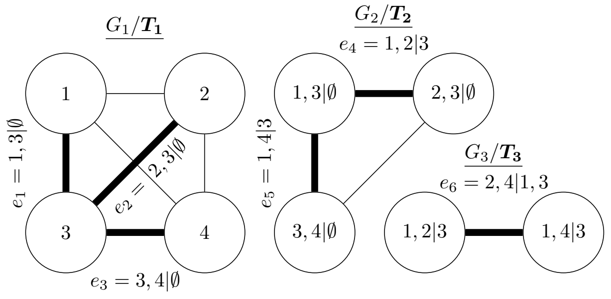
1. Let $T_1, \dots, T_{d-1}$ be the trees in a R-Vine $\mathcal{V}$, each of them with set of nodes V_i and set of edges E_i.
2. Every edge $e \in E_i$ has associated three sets of variable indexes $C(e), D(e), N(e) \subset \{1, \dots, d\}$ called the conditioned, conditioning and constraint sets of e, respectively.
3. The first tree in the hierarchy has set of nodes $V_1 = \{1, \dots, d\}$ and set of edges E_1, which is obtained by inferring a spanning tree from a complete graph G_1 over V_1.
4. For any edge $e \in E_1$ joining nodes $j, k \in V_1$, $C(e) = N(e) = \{j, k\}$ and $D(e) = \{\emptyset\}$.
5. The i-th tree has node set $V_i = E_{i-1}$ and edge set E_i, for $i = 2, \dots d-1$. E_i is obtained by inferring a spanning tree from a graph G_i; this graph has set of nodes V_i and edges $e = (e_1, e_2) \in E_i$, such that $e_1, e_2 \in E_{i-1}$ share a common node in V_{i-1}.
6. Edges $e = (e_1, e_2) \in E_i$ have conditioned, conditioning and constraint sets given by $C(e) = N(e_1)\Delta N(e_2)$, $D(e) = N(e_1) \cap N(e_1)$ and $N(e) = N(e_1) \cup N(e_2)$, where $A \,\Delta\, B = (A \setminus B) \cup (B \setminus A)$.

Each of the edges in the trees $T_1, \dots, T_{d-1}$ forming the vine $\mathcal{V}$ is a different factor in the factorization of $c(u_1, \dots, u_d)$, i.e. a different conditional bivariate copula density. Since there are a total of $d(d-1)/2$ edges, $\mathcal{V}$ factorizes $c(u_1, \dots, u_d)$ as the product of $d(d-1)/2$ factors. We now show how to obtain the form of each of these factors. For any edge $e(j,k) \in T_i$ with conditioned set $C(e) = \{j, k\}$ and conditioning set $D(e)$ we define $c_{jk|D(e)}$ to be the bivariate copula density for u_j and u_k given the value of the conditioning variables $\{u_i : i \in D(e)\}$, that is,

$$c_{jk|D(e)} := c(P_{j|D(e)}, P_{k|D(e)} | u_i : i \in D(e)), \qquad (3)$$

where $P_{j|D(e)} := P(u_j | u_i : i \in D(e))$ is the conditional cdf of u_j given the value of the conditioning variables $\{u_i : i \in D(e)\}$. Then, the vine $\mathcal{V}$ formed by the hierarchy of trees $T_1, \dots, T_{d-1}$ specifies the following factorization for the copula density:

$$c(u_1, \dots, u_d) = \prod_{i=1}^{d-1} \prod_{e(j,k) \in E_i} c_{jk|D(e)}\,, \qquad (4)$$



$$c_{1234} = \underbrace{\underbrace{c_{13|\emptyset}}_{e_1}\underbrace{c_{23|\emptyset}}_{e_2}\underbrace{c_{34|\emptyset}}_{e_3}}_{T_1}\underbrace{\underbrace{c_{12|3}}_{e_4}\underbrace{c_{14|3}}_{e_5}}_{T_2}\underbrace{\underbrace{c_{24|13}}_{e_6}}_{T_3}$$

Figure 2. Example of the hierarchical construction of an R-vine factorization of a copula density of four variables $c(u_1, u_2, u_3, u_4)$. The edges selected to form each tree are highlighted in bold. Conditioned and conditioning sets for each node and edge are shown as $C(e)|D(e)$.

as shown by Kurowicka & Cooke (2006).

Figure 2 exemplifies how to construct a regular vine that factorizes the copula density $c(u_1, u_2, u_3, u_4)$ into the product of 6 bivariate conditional copula densities. The first tree T_1 has node set $V_1 = \{1, 2, 3, 4\}$. The edge set E_1 is obtained by selecting a spanning tree over G_1, the complete graph for the nodes in V_1. Our choice for E_1 is highlighted in bold in the left-most plot in Figure 2. Edges in $E_1 = \{e_1, e_2, e_3\}$ have conditioned and constraint sets $C(e_1) = N(e_1) = \{1, 3\}$, $C(e_2) = N(e_2) = \{2, 3\}$, $C(e_3) = N(e_3) = \{3, 4\}$ and conditioning sets $D(e_1) = D(e_2) = D(e_3) = \{\emptyset\}$. The second tree in the hierarchy has node set $V_2 = E_1$. In this case, we select a spanning tree over G_2, a graph with node set V_2 and edge set formed by pairs of edges $e_i, e_j \in E_1$ sharing some common node $v_k \in V_1$. We select $E_2 = \{e_4, e_5\}$ with conditioned sets $C(e_4) = N(e_1)\Delta N(e_2) = \{1, 2\}$ and $C(e_5) = N(e_2)\Delta N(e_3) = \{1, 4\}$, conditioning sets $D(e_4) = N(e_1) \cap N(e_2) = \{3\}$ and $D(e_5) = N(e_2) \cap N(e_3) = \{3\}$, and constraint sets $D(e_4) = N(e_1) \cup N(e_2) = \{1, 2, 3\}$ and $D(e_5) = N(e_1) \cap N(e_3) = \{1, 3, 4\}$. Finally, we build a third graph G_3 with node set $V_3 = E_2$ and only one edge e_6. This last edge is the only possible spanning tree and has node set V_3 and edge set $E_3 = \{e_6\}$. The edge e_6 has conditioned set $C(e_6) = N(e_4)\Delta N(e_5) = \{2, 4\}$, conditioning set $D(e_6) = N(e_4) \cap N(e_5) = \{1, 3\}$ and constraint set $N(e_6) = N(e_4) \cup N(e_5) = \{1, 2, 3, 4\}$. The resulting factorization of $c(u_1, u_2, u_3, u_4)$ given by the tree hierarchy is shown at the bottom of Figure 2.

There exist many factorizations of a copula density $c(u_1, \dots, u_d)$ in terms of bivariate copulas. Each factorization is determined by the specific choices of the spanning trees $T_1, \dots, T_d$ in the algorithm described above. In practice, the trees are selected by assigning a weight to each edge $e(j,k)$ (copula $c_{jk|D(e)}$) in the graphs $G_1, \dots, G_{d-1}$ and then selecting the maximum spanning tree at each iteration. A common practice is to directly relate the weight of the edge $e(j,k)$ to the amount of dependence described by the corresponding copula $c_{jk|D(e)}$. This amount of dependence can be measured as the absolute value of the empirical Kendall's τ correlation coefficient between the samples of $u_{j|D(e)}$ and $u_{k|D(e)}$. The maximum spanning tree can then be selected efficiently using Prim's algorithm (Prim, 1957; Dissmann et al., 2012).

On the first tree of a vine, only pairwise dependencies are described, and the corresponding copulas are not conditioned. The following trees describe dependencies between 3, 4, ... and $d-1$ variables by means of increasingly deeper conditioning, until completing a full description of the joint $d-$dimensional copula density. Since the cost of constructing the full tree hierarchy is quadratic in d, one may choose to prune the vine and construct only the first $d' < (d-1)$ trees, ignoring the remaining copula densities in the factorization. Since the independent copula has pdf constant and equal to one, this pruning assumes independence in the higher order interactions captured by the ignored copulas (Brechmann et al., 2012)

2.2. Conditional Dependencies in Vines

As shown in equations (3) and (4), vine distributions require to calculate marginal conditional cdfs and conditional bivariate copula densities. The number of variables to condition on increases as we move deeper in the vine hierarchy. In general, to obtain the factors corresponding to the i-th tree, we have to condition both copula densities and marginal cdfs to $i-1$ variables. The computation of the conditional cdfs appearing at tree T_i can be done using the copula functions from the previous tree T_{i-1}. In particular, the following recursive relationship holds

$$P_{j|D(e)} = \frac{\partial C_{jk|D(e)\setminus\{k\}}}{\partial P_{k|D(e)\setminus\{k\}}}, \qquad (5)$$

where $C_{jk|D(e)\setminus\{k\}} := C(P_{j|D(e)\setminus\{k\}}, P_{k|D(e)\setminus k}|u_i : i \in D(e) \setminus \{k\})$ is the cdf of the conditional copula density $c_{jk|D(e)\setminus\{k\}}$ and $D(e) \setminus \{k\}$ denotes the conditioning set $D(e)$ with the element k removed (Joe, 1996). This derivative has well-known, closed-forms for each parametric copula (refer to the Appendix for the Gaussian copula case). However, we still have to compute the conditional bivariate copula densities. A solution commonly found in the literature is to assume that the copulas $c_{jk|D(e)}$ in (4) are independent of their conditioning variables. This is known as the *simplifying assumption* for vine copulas (Hobaek et al., 2010). The main advantage is that we can construct vine models using standard unconditional parametric copulas. The disadvantage is that we may fail to capture some of the dependencies present in the data. As an alternative to the simplifying assumption, we now present a general technique to construct conditional parametric bivariate copulas.

3. Proposed Approach to Estimate Conditional Bivariate Copulas

In this section we address the estimation of the conditional copula of two random variables X and Y given a vector of conditioning variables $\mathbf{Z} = (Z_1, \dots, Z_d)^{\mathrm{T}} \in \mathbb{R}^d$. Let $P_{X|\mathbf{Z}}$ and $P_{Y|\mathbf{Z}}$ be the conditional cdfs of X and Y given $\mathbf{Z}$. Patton (2006) shows that the conditional copula of X and Y given $\mathbf{Z}$ is the conditional distribution of the random variables $U = P_{X|\mathbf{Z}}(X|\mathbf{Z})$ and $V = P_{Y|\mathbf{Z}}(Y|\mathbf{Z})$ given $\mathbf{Z}$. We assume a parametric bivariate copula for the joint distribution of U and V. This type of copulas can often be fully specified in terms of Kendall's τ rank correlation coefficient (Joe, 1997). Table 1 shows, for some widely-used copula families, the domain of their parameter θ and the corresponding bijective expressions for θ as a function of Kendall's τ. To capture the dependence of the copula on $\mathbf{Z}$ we introduce a latent function $g : \mathbb{R}^d \to [-1,1]$ such that $\tau = g(\mathbf{Z})$. The task of interest is then to estimate g given observations of X, Y and $\mathbf{Z}$.

When $P_{X|\mathbf{Z}}$ and $P_{Y|\mathbf{Z}}$ are known, we can transform any sample of X, Y and $\mathbf{Z}$ into a corresponding sample of U, V and $\mathbf{Z}$. Let $\mathcal{D} = \{\mathcal{D}_{U,V} = \{(u_i, v_i)\}_{i=1}^n, \mathcal{D}_{\mathbf{Z}} = \{\mathbf{z}_i\}_{i=1}^n\}$ be such a sample, where u_i and v_i and $\mathbf{z}_i$ are paired. To guarantee that $g(x) \in [-1,1]$, we assume w.l.o.g. that $g(x) = 2\Phi(f(\mathbf{x})) - 1$, where $\Phi(x)$ is the standard Gaussian cdf and $f : \mathbb{R}^d \to \mathbb{R}$ is a non-linear function that uniquely specifies g. We can infer g by placing a Gaussian process prior on f and then computing the posterior for f given $\mathcal{D}$ (Rasmussen & Williams, 2006). For this, let $\mathbf{f}$ be the n-dimensional vector such that $\mathbf{f} = (f(\mathbf{z}_1), \dots, f(\mathbf{z}_n))^{\mathrm{T}}$. The prior for $\mathbf{f}$ given $\mathcal{D}_{\mathbf{Z}}$ is

$$p(\mathbf{f}|\mathcal{D}_{\mathbf{Z}}) = \mathcal{N}(\mathbf{f}|\mathbf{m}, \mathbf{K}), \qquad (6)$$

where $\mathbf{m}$ is a n-dimensional mean vector and $\mathbf{K}$ is an $n \times n$ covariance matrix generated by the covariance function or kernel

$$\begin{aligned} k_{ij} &\equiv \mathrm{Cov}[f(\mathbf{z}_i), f(\mathbf{z}_j)] \\ &\equiv \sigma \exp\left\{-(\mathbf{z}_i - \mathbf{z}_j)^{\mathrm{T}} \mathrm{diag}(\boldsymbol{\lambda})(\mathbf{z}_i - \mathbf{z}_j)\right\} + \sigma_0, \end{aligned} \qquad (7)$$

where $\boldsymbol{\lambda}$ is a vector of lengthscales and σ, σ_0 are amplitude and noise parameters. Then, the posterior distribution for $\mathbf{f}$ given $\mathcal{D}_{U,V}$ and $\mathcal{D}_{\mathbf{Z}}$ is

$$p(\mathbf{f}|\mathcal{D}_{U,V}, \mathcal{D}_{\mathbf{Z}}) = \frac{p(\mathcal{D}_{U,V}|\mathbf{f})p(\mathbf{f}|\mathcal{D}_{\mathbf{Z}})}{p(\mathcal{D}_{U,V}|\mathcal{D}_{\mathbf{Z}})}\,, \tag{8}$$

where $p(\mathcal{D}_{U,V}|\mathbf{f}) = \prod_{i=1}^{n} c(u_i, v_i|\tau = 2\Phi(f_i) - 1)$, $p(\mathcal{D}_{U,V}|\mathcal{D}_{\mathbf{Z}})$ is a normalization constant and $c(\cdot,\cdot|\tau)$ is the density of a parametric bivariate copula specified in terms of Kendall's τ. Given a particular value of $\mathbf{Z}$ such as $\mathbf{z}^\star$, we can make predictions about the conditional distribution of U and V given $\mathbf{z}^\star$ using

$$\begin{aligned} p(u^\star, v^\star|\mathbf{z}^\star) = \int & c(u^\star, v^\star|\tau = 2\Phi(f^\star) - 1) \\ & p(f^\star|\mathbf{f}, \mathbf{z}^\star, \mathcal{D}_{\mathbf{z}})p(\mathbf{f}|\mathcal{D}_{U,V}, \mathcal{D}_{\mathbf{Z}})\, d\mathbf{f}\, df^\star\,, \end{aligned} \tag{9}$$

$p(f^\star|\mathbf{f}, \mathbf{z}^\star, \mathcal{D}_{\mathbf{z}}) = \mathcal{N}(f^\star|\mathbf{k}^{\mathrm{T}}\mathbf{K}^{-1}\mathbf{f}, k - \mathbf{k}^{\mathrm{T}}\mathbf{K}^{-1}\mathbf{k})$, $\mathbf{k} = (\mathrm{Cov}(f(\mathbf{z}^\star), f(\mathbf{z}_1)), \ldots, \mathrm{Cov}(f(\mathbf{z}^\star), f(\mathbf{z}_n)))^{\mathrm{T}}$ and $k = \mathrm{Cov}(f(\mathbf{z}^\star), f(\mathbf{z}^\star))$. Unfortunately, (8) and (9) cannot be computed analytically, so we decide to approximate them using Expectation Propagation (EP) (Minka, 2001). This method approximates each of the n factors in $p(\mathcal{D}_{U,V}|\mathbf{f})$ with an unnormalized Gaussian distribution whose mean and variance parameters are updated iteratively by matching sufficient statistics. See Rasmussen & Williams (2006) for further details. To refine each of these univariate Gaussians, we have to compute three unidimensional integrals using quadrature methods. For prediction at $\mathbf{z}^\star$, we sample $f(\mathbf{z}^\star)$ from the Gaussian approximation found by EP and then average over copula models with $\tau = 2\Phi(f(\mathbf{z}^\star)) - 1$. The resulting conditional copula model is semi-parametric: The dependence between U and V given $\mathbf{Z}$ is parametric but the effect of $\mathbf{Z}$ on the copula is non-parametric.

3.1. Sparse GPs to Speed up Computations

The total cost of EP is $O(n^3)$, since it is dominated by the computation of the Cholesky decomposition of an $n \times n$ matrix. To reduce this cost, we use the FITC approximation for Gaussian Processes (Snelson & Ghahramani, 2006; Naish-Guzman & Holden, 2007). Under this approximation, the $n \times n$ covariance matrix $\mathbf{K}$ is approximated by $\mathbf{K}' = \mathbf{Q} + \mathrm{diag}(\mathbf{K} - \mathbf{Q})$, where $\mathbf{Q} = \mathbf{K}_{nn_0}\mathbf{K}_{n_0n_0}^{-1}\mathbf{K}_{nn_0}^{\mathrm{T}}$, $\mathbf{K}_{n_0n_0}$ is the $n_0 \times n_0$ covariance matrix generated by evaluating (7) at all combinations of some $n_0 \ll n$ training points or *pseudo-inputs*, and $\mathbf{K}_{nn_0}$ is the $n \times n_0$ matrix with the covariances between all possible combinations of original training points and pseudo-inputs. These approximations allow us to run the EP method with cost $O(nn_0^2)$. The kernel hyper-parameters $\boldsymbol{\lambda}$, σ and σ_0 and the pseudo-inputs are optimized by maximizing the EP approximation of the model evidence (Rasmussen & Williams, 2006).

Table 1. Some copula families, with their parameter domains and expressions as Kendall's τ correlations. For some copulas like Frank or Joe, numerical approximations are used to compute $\theta(\tau)$.

Family	**Parameter**	$\theta(\tau) =$
Gaussian	$\theta \in [-1, 1]$	$sin\left(\frac{\pi}{2}\tau\right)$
Student	$\theta \in [-1, 1]$	
Clayton	$\theta \in (0, \infty)$	$2\tau/(1-\tau)$
Gumbel	$\theta \in [1, \infty)$	$1/(1-\tau)$
Frank	$\theta \in (0, \infty)$	No closed form
Joe	$\theta \in (1, \infty)$	

Learning a vine with our method scales linearly with the number of samples n, but quadratically with the number of pseudo-inputs n_0 and the number of variables d: thus, its complexity is $O(d^2 n_0^2 n)$. By contrast, learning a *simplified* vine has complexity $O(d^2)$.

3.2. Related Work

Acar, Genest and Neslehova (2012) first addressed the lack of conditional dependencies in the parametric copulas that form a vine model. They use a method similar in spirit to the one described above, and model τ as a non-linear function of a single conditioning variable Z (Acar et al., 2011). However, their method cannot handle multivariate conditional dependences and consequently, they only show results for trivariate vines. They use the *Maximum Local Likelihood* (MLL) method to infer a non-linear relationship between τ and Z. Acar et al. approximate linearly f at any point z where f needs to be evaluated. Then, they adjust the coefficients of the resulting linear form using the available observations in the neighborhood of z. In particular, given a sample $\mathcal{D} = \{\mathcal{D}_{U,V} = \{(u_i, v_i)\}_{i=1}^n, \mathcal{D}_Z = \{z_i\}_{i=1}^n\}$, they estimate $f(z)$ by solving the optimization problem

$$\begin{aligned} (b_{z,0}^\star, b_{z,1}^\star) = \underset{(b_0,b_1)}{\arg\max} \Bigg\{ \sum_{i=0}^{N} & k_h(z - z_i) \\ & \log c(u_i, v_i|\tau = 2\Phi(b_0 + b_1(z - z_i)) - 1)\Bigg\}\,, \end{aligned} \tag{10}$$

where the neighborhood of z is determined by the Epanechnikov kernel $k_h(x) = \frac{3}{4h}\max(0, 1 - (\frac{x}{h})^2)$ with bandwidth h. An estimate of $f(z)$ is then obtained as the intercept of the linear approximation at z, that is, $f(z) \approx b_{z,0}^\star$. Acar et al. adjust h by running a leave-one-out cross validation search on the training data. Some disadvantages of the MLL method are: (i) it can only condition on a single scalar variable, (ii) we have to solve the optimization problem (10) for each prediction that we want to make and more importantly (iii) since it is a local-based method (similarly as nearest neighbours) it can lead to poor predictive performance when the available data is sparsely distributed.

4. Experiments

We evaluate the performance of the proposed method for the estimation of vine copula densities with full conditional dependencies. Because GPs are an important part in this method we call it GPVINE. We compare with two benchmark methods: (i) a vine model based on the simplifying assumption (SVINE), which ignores any conditional dependencies in the bivariate copulas, and (ii) a vine model based on the MLL method of Acar et al. (2012) (MLLVINE). MLLVINE can only handle conditional dependencies with respect on a single scalar variable. Therefore, we can only evaluate its performance in the construction of vine models with two trees, since additional levels would require to account for multivariate conditional dependencies.

In all the experiments, we use 20 pseudo-inputs in the generalized FITC approximation. The Gaussian processes use a kernel function given by (7), whose hyperparameters and pseudo-inputs are tuned by maximizing the EP estimate of the marginal likelihood. The mean of the GP prior (6) is chosen to be constant and equal to $\Phi^{-1}((\hat{\tau}_{MLE}+1)/2)$, where $\hat{\tau}_{MLE}$ is the maximum likelihood estimate of τ for an unconditional Gaussian copula given the training data. In MLLVINE, the bandwidth of the Epanechnikov kernel is selected by running a leave-one-out cross validation search using a 30-dimensional log-spaced grid ranging from 0.05 to 10. To simplify the experiments, we focus on regular vines generated using bivariate Gaussian copulas (see Appendix A) as building blocks. The extension of the proposed approach to select among different parametric families of bivariate copulas is straightforward: the best family for a given pair of variables can be selected by Bayesian model selection, using the evidence approximation given by EP. All the data are preprocessed to have uniform marginal distributions: this is done by mapping each marginal observation to its empirical cumulative probability.

4.1. Synthetic Data

We first perform a series of experiments with synthetic three-dimensional data. In particular, we sample the scalar variables X, Y and Z according to the following generative process. First, Z is sampled uniformly from the interval $[-6, 6]$ and second, X and Y are sampled given Z from a bivariate Gaussian distribution with zero mean and covariance matrix given by

Table 2. Average test log-likelihood and standard deviations for SVINE and GPVINE on real-world datasets (higher is better). Asterisks denote results that are not statistically significant to a paired Wilcoxon test with p–value $= 10^{-3}$.

Data	Trees	SGVINE	GPVINE
Cloud	1	**7.860 ± 0.346**	**7.860 ± 0.346**
	2	8.899 ± 0.334	**9.335 ± 0.348**
	3	9.426 ± 0.363	**10.053 ± 0.397**
	4	9.570 ± 0.361	**10.207 ± 0.415**
	5	9.644 ± 0.357	**10.332 ± 0.440**
	6	9.716 ± 0.354	**10.389 ± 0.459**
	7	9.783 ± 0.361	**10.423 ± 0.463**
	8	9.790 ± 0.371	**10.416 ± 0.459**
	9	9.788 ± 0.373	**10.408 ± 0.460**
Glass	1	**0.827 ± 0.150**	**0.827 ± 0.150**
	2	1.206 ± 0.259	**1.264 ± 0.303**
	3	1.281 ± 0.251	**1.496 ± 0.289**
	4	1.417 ± 0.251	**1.740 ± 0.308**
	5	1.493 ± 0.291	**1.853 ± 0.318**
	6	1.591 ± 0.301	**1.936 ± 0.325**
	7	1.740 ± 0.282	**2.000 ± 0.345**
	8	1.818 ± 0.243	**2.034 ± 0.343**
Jura	1	**1.887 ± 0.153**	**1.887 ± 0.153**
	2	2.134 ± 0.164	**2.151 ± 0.173**
	3	2.199 ± 0.151	**2.222 ± 0.173**
	4*	2.213 ± 0.153	**2.233 ± 0.181**
	5*	2.209 ± 0.153	**2.215 ± 0.185**
	6*	**2.213 ± 0.155**	2.197 ± 0.189
Shuttle	1	**1.487 ± 0.256**	**1.487 ± 0.256**
	2	2.188 ± 0.314	**2.646 ± 0.349**
	3	2.552 ± 0.273	**3.645 ± 0.427**
	4	2.782 ± 0.284	**4.204 ± 0.551**
	5	3.092 ± 0.353	**4.572 ± 0.567**
	6	3.284 ± 0.325	**4.703 ± 0.492**
	7	3.378 ± 0.288	**4.763 ± 0.408**
	8	3.417 ± 0.257	**4.761 ± 0.393**
	9	3.426 ± 0.252	**4.755 ± 0.389**
Weather	1	**0.684 ± 0.128**	**0.684 ± 0.128**
	2	0.789 ± 0.159	**1.312 ± 0.227**
	3	0.911 ± 0.178	**2.081 ± 0.341**
	4	1.017 ± 0.184	**2.689 ± 0.368**
	5	1.089 ± 0.188	**3.078 ± 0.423**
	6	1.138 ± 0.181	**3.326 ± 0.477**
	7	1.170 ± 0.169	**3.473 ± 0.467**
	8	1.177 ± 0.170	**3.517 ± 0.465**
Stocks	1	**2.776 ± 0.142**	2.776 ± 0.142
	2*	**2.799 ± 0.142**	2.785 ± 0.146
	3	**2.801 ± 0.142**	2.764 ± 0.151
	4	**2.802 ± 0.143**	2.742 ± 0.158
	5	**2.802 ± 0.141**	2.721 ± 0.159
Housing	1	**3.409 ± 0.354**	**3.409 ± 0.354**
	3	4.128 ± 0.363	**4.953 ± 0.425**
	5	4.386 ± 0.380	**5.541 ± 0.498**
	7	4.576 ± 0.422	**5.831 ± 0.529**
	9	4.768 ± 0.399	**6.009 ± 0.516**
	11	4.949 ± 0.362	**6.113 ± 0.525**

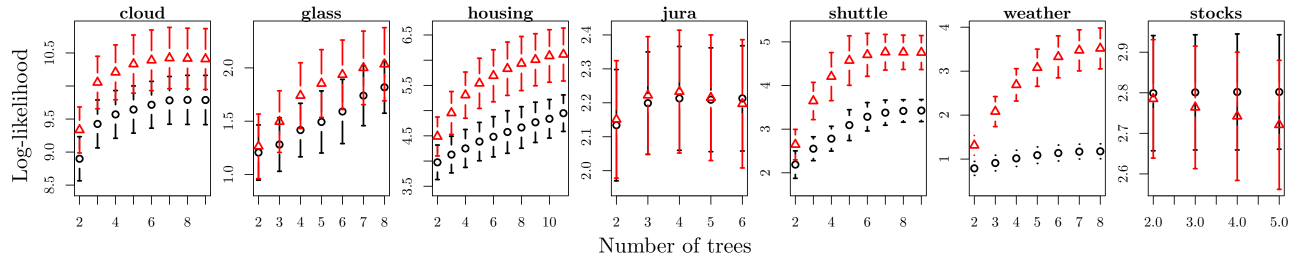


Figure 3. Average test log-likelihood and standard deviations for each dataset as the number of trees forming the vines increases, for GPVINE (red triangles) and SVINE (black dots) (higher is better).

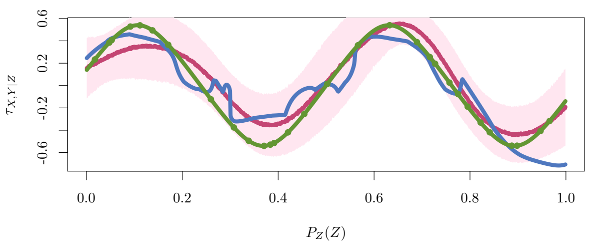


Figure 4. In green, the true function g that maps u_3 to τ. The approximations learned by GPVINE and MLLVINE are shown in red and blue, respectively. For GPVINE, the uncertainty of the prediction (± 1 standard deviation) is drawn as a red shaded area. Small dots are used to mark the available observations of u_3 for training.

$\text{Var}(X) = \text{Var}(Y) = 1$ and $\text{Cov}(X, Y|Z) = 3/4 \sin(Z)$. We sample a total of 1000 data points and then choose 50 subsamples of size 100 to infer a vine model for the data using SVINE, MLLVINE and GPVINE. Average test log-likelihoods on the remaining data points are shown in Table 3. GPVINE obtains the best results.

Figure 4 displays the true value of the function g that maps u_3 to τ in the conditional copula $c(P(u_1|u_3), P(u_1|u_3)|u_3, \tau)$, where u_1, u_3 and u_3 are the empirical cumulative probability levels of the samples generated for X, Y and Z, respectively. We also show the approximations of g generated by GPVINE and MLLVINE. In this case, GPVINE is much better than MLLVINE at approximating the true g.

4.2. Real-world Datasets

We evaluate the performance of SVINE, MLLVINE and GPVINE on several real-world datasets. For each dataset, we generate 50 random partitions of the data into training and test sets, each containing half of the available data. The different methods are run on each training set and their log-likelihood is then evaluated on the corresponding test set (higher is better). The analyzed datasets are described in Section 4.2.1. Table 2 and Figure 3 show the test log-likelihood for SVINE and GPVINE, when using up to $1, \ldots, (d-1)$ trees in the vine, where d is the number of variables in the data. In general, taking into account possible dependencies in the conditional bivariate copulas leads to superior predictive performance. Also, we often find that the gains obtained get larger as we increase the number of trees in the vines. However, in a few of the datasets the simplifying assumption seems valid (Stocks and Jura datasets). Table 3 shows results for all methods (including MLLVINE) when only two trees are used in the vines. In these experiments, MLLVINE is most of the times outperformed by GPVINE. To better measure the percent improvement experienced when using GPVINE, one can subtract the achieved likelihood when using only the first tree of the vine from the all results. We also show how GPVINE can be used to discover scientifically interesting features through learning spatially varying correlations (Figure 5).

4.2.1. Description of the Datasets

Mineral Concentrations The *jura* dataset contains the concentration measurements of 7 chemical elements (Cd, Co, Cr, Cu, Ni, Pb, Cn) in 359 locations of the Swiss Jura Mountains (Goovaerts, 1997). The *uranium* dataset contains log-concentrations of 7 chemical elements (U, Li, Co, K, Cs, Sc, Ti) in a total of 655 water samples collected near Grand Junction, CO (Cook & Johnson, 1986). Acar et al. (2012) use the measurements for *Co*, *Ti* and *Sc* to evaluate the performance of MLLVINE. We replicated this task for the three analyzed methods (Table 3).

Barcelona Weather *OpenWeatherMap* (Extreme Electronics Ltd., 2012) provides access to meteorological stations around the world. We downloaded data for the 300 weather stations nearest to Barcelona, Spain (41.3857N, 2.1699E) on 11/19/2012 at 8pm (*weather* dataset). Each station returns values for *longitude*, *latitude*, *distance to Barcelona*, *temperature*, *atmospheric pressure*, *humidity*, *wind speed*, *wind direction* and *cloud cover percentage*. Figure 5 shows how the posterior mean of τ for the copula linking the variables *at-*

Table 3. Average test log-likelihood and standard deviations for all methods and datasets when limited to 2 trees in the vine (higher is better).

Data	SVINE	MLLVINE	GPVINE
Synthetic	-0.005 ± 0.012	0.101 ± 0.162	$\mathbf{0.298 \pm 0.031}$
Uranium	0.006 ± 0.006	0.016 ± 0.026	$\mathbf{0.022 \pm 0.012}$
Cloud	8.899 ± 0.334	9.013 ± 0.600	$\mathbf{9.335 \pm 0.348}$
Glass	1.206 ± 0.259	0.460 ± 1.996	$\mathbf{1.264 \pm 0.303}$
Housing	3.975 ± 0.342	4.246 ± 0.480	$\mathbf{4.487 \pm 0.386}$
Jura	2.134 ± 0.164	2.125 ± 0.177	$\mathbf{2.151 \pm 0.173}$
Shuttle	2.552 ± 0.273	2.256 ± 0.612	$\mathbf{3.645 \pm 0.427}$
Weather	0.789 ± 0.159	0.771 ± 0.890	$\mathbf{1.312 \pm 0.227}$
Stocks	$\mathbf{2.802 \pm 0.141}$	2.739 ± 0.155	2.785 ± 0.146

mospheric pressure and *cloud cover percentage* varies when conditioned on *latitude* and *longitude*.

World Stock Indices We apply the probability integral transform to the residuals of an ARMA(1,1)-GARCH(1,1) model with Student t innovations. The residuals are obtained after fitting this model to the daily log-returns of the major world stock indices in 2009 and 2010 (*stocks* dataset, 396 points in total) (Brechmann & Schepsmeier, 2013). The indices are the US American *S&P 500*, the Japanese *Nikkei 225*, the Chinese *SSE Composite Index*, the German *DAX*, the French *CAC 40* and the British *FTSE 100 Index*.

UCI Datasets We also include experimental results for the *Glass*, *Housing*, *Cloud* and *Shuttle* datasets from the UCI Dataset Repository (Frank & Asuncion, 2010).

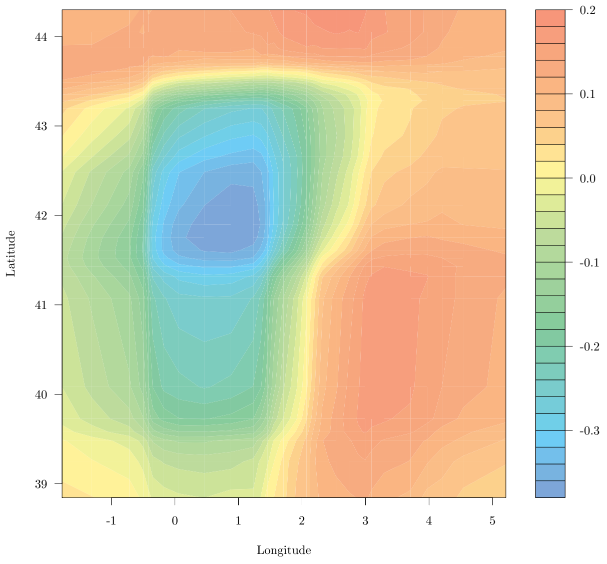


Figure 5. Kendall's τ correlation between *atmospheric pressure* and *cloud percentage cover* (color scale) when conditioned to *longitude* and *latitude*. The blue region in the plot corresponds to the Pyrenees mountains.

5. Conclusion

Vine copulas are increasingly popular models for multivariate data. They specify a factorization of any high-dimensional copula density into a product of conditional bivariate copulas. However, some of the conditional dependencies in these bivariate copulas are usually ignored when constructing the vine. This can produce overly simplistic estimates when dealing with real-world data. To avoid this, we presented a method for the estimation of fully conditional vines using Gaussian processes (GPVINE). A series of experiments with synthetic and real-world data show that, often, GPVINE obtains better predictive performance than a baseline method that ignores conditional dependencies. Additionally, GPVINE performs favorably with respect to state-of-the-art alternatives based on maximum local-likelihood methods (MLLVINE).

Acknowledgements

DLP and JMLH contributed equally to this work. We thank P. Hennig, A. W. Wilson, C. Czado, N. Krämer and E. C. Brechmann for their helpful feedback. DLP was funded by Fundación Caja Madrid and the PASCAL2 Network of Excellence. JMHL was funded by Infosys Labs, Infosys Limited.

A. The Bivariate Gaussian Copula

The bivariate Gaussian copula with correlation parameter θ represents the dependence structure found in a bivariate Gaussian distribution of two random variables with correlation θ. The Gaussian copula has cdf

$$C(u,v|\theta) = \Phi_2(\Phi^{-1}(u), \Phi^{-1}(v))|\theta), \qquad (11)$$

where $\Phi_2(\cdot,\cdot|\theta)$ is the cdf of a bivariate Gaussian with marginal variances equal to one and correlation θ, and Φ^{-1} is the quantile function of the standard Gaussian distribution. The corresponding pdf is

$$c(u,v|\theta) = \frac{\phi_2(\Phi^{-1}(u), \Phi^{-1}(v)|\theta)}{\phi(\Phi^{-1}(u))\phi(\Phi^{-1}(v))}, \qquad (12)$$

where ϕ_2 is the derivative (pdf) of Φ_2. The conditional cdfs are given by

$$\frac{\partial C(u,v|\theta)}{\partial u} = \Phi\left(\frac{\Phi^{-1}(v) - \theta\,\Phi^{-1}(u)}{\sqrt{1-\theta^2}}\right), \qquad (13)$$

$$\frac{\partial C(u,v|\theta)}{\partial v} = \Phi\left(\frac{\Phi^{-1}(u) - \theta\,\Phi^{-1}(v)}{\sqrt{1-\theta^2}}\right), \qquad (14)$$

where Φ is the standard Gaussian cdf.

References

Acar, E. F., Craiu, R. V., and Yao, F. Dependence calibration in conditional copulas: A nonparametric approach. *Biometrics*, 67(2):445–453, 2011.

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

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

Brechmann, E.C. and Schepsmeier, U. Modeling dependence with C- and D-vine copulas: The R package CDVine. *Journal of Statistical Software*, 52(3): 1–27, 2013.

Brechmann, E.C., Czado, C., and Aas, K. Truncated regular vines in high dimensions with applications to financial data. *Canadian Journal of Statistics*, 40 (1):68–85, 2012.

Casella, George and Berger, Roger. *Statistical Inference*. Duxbury Resource Center, 2001.

Cook, R. D. and Johnson, M. E. Generalized burrparetologistic distributions with applications to a uranium exploration data set. *Technometrics*, 28: 123–131, 1986.

Dissmann, J., Brechmann, E. C., Czado, C., and Kurowicka, D. Selecting and estimating regular vine copulae and application to nancial returns. *arXiv preprint*, 2012.

Elidan, G. Copula Bayesian networks. In *Advances in Neural Information Processing Systems 23*, pp. 559–567, 2010.

Elidan, G. Copulas and machine learning. *Invited survey to appear in the proceedings of the Copulae in Mathematical and Quantitative Finance workshop*, 2012.

Extreme Electronics Ltd. OpenWeatherMap, 2012. URL http://openweathermap.org/.

Frank, A. and Asuncion, A. UCI machine learning repository, 2010. URL http://archive.ics.uci.edu/ml.

Goovaerts, P. *Geostatistics for natural resources evaluation*. Oxford University Press, 1st edition, 1997.

Hobaek, I., Aas, K., and Frigessi, A. On the simplified pair-copula construction. simply useful or too simplistic? *Journal of Multivariate Analysis*, 101(5): 1296–1310, 2010.

Joe, H. Families of m-variate distributions with given margins and $m(m-1)/2$ bivariate dependence parameters. *Distributions with Fixed Marginals and Related Topics*, 1996.

Joe, H. *Multivariate Models and Dependence Concepts*. CRC Press, 1997.

Joe, H. Asymptotic efficiency of the two-stage estimation method for copula-based models. *Journal of Multivariate Analysis*, 94(2):401–419, 2005.

Kirshner, S. Learning with tree-averaged densities and distributions. In *Advances in Neural Information Processing Systems 20*, 2007.

Kurowicka, D. and Cooke, R. *Uncertainty Analysis with High Dimensional Dependence Modelling*. Wiley Series in Probability and Statistics, 1st edition, 2006.

Minka, T. P. Expectation Propagation for approximate Bayesian inference. *Proceedings of the 17th Conference in Uncertainty in Artificial Intelligence*, pp. 362–369, 2001.

Naish-Guzman, Andrew and Holden, Sean B. The generalized FITC approximation. In *Advances in Neural Information Processing Systems 20*, 2007.

Nelsen, R. *An Introduction to Copulas*. Springer Series in Statistics, 2006.

Patton, A. J. Modelling asymmetric exchange rate dependence. *International Economic Review*, 47(2): 527–556, 2006.

Prim, R. C. Shortest connection networks and some generalizations. *Bell System Technology Journal*, 36: 1389–1401, 1957.

Rasmussen, C. E. and Williams, C. K. I. *Gaussian Processes for Machine Learning*. The MIT Press, 1st edition, 2006.

Sklar, A. Fonctions de repartition à n dimension set leurs marges. *Publ. Inst. Statis. Univ. Paris*, 8(1): 229–231, 1959.

Snelson, E. and Ghahramani, Z. Sparse Gaussian processes using pseudo-inputs. *Proceedings of the 20th Conference in Advances in Neural Information Processing Systems*, pp. 1257–1264, 2006.

Wilson, A. G. and Ghahramani, Z. Copula processes. In *Advances in Neural Information Processing Systems 23*, pp. 2460–2468, 2010.