Virtual patient populations for quantitative pharmacology with its application in antibiotic treatment optimization

Guo, Y.

Chapter 2

Tutorial: Copulas for covariate simulation in R

Authors

Yuchen Guo

Tingjie Guo

J. G. Coen van Hasselt

Laura B. Zwep

CPT: Pharmacometrics & Systems Pharmacology15, no. 4 (2026): e70242

Abstract

Patient-specific covariates are commonly incorporated in pharmacometric and quantitative system pharmacology models to predict differences in pharmacokinetic or pharmacodynamic profiles between patients. When simulating new virtual populations of patients, generating realistic covariate sets that accurately reflect the correlation structures among covariates is essential to obtain reliable simulation outcomes. Copulas are joint distribution functions that characterize the dependence structures of patient covariates and enable the simulation of virtual populations. The current tutorial provides a step-by-step guide for understanding the concept of copulas and an overview of applications of copulas in pharmacometric research.

2.1 Introduction

In model-informed drug development and precision dosing, patient characteristics are key elements in predicting drug exposure (i.e., pharmacokinetics, PK) and treatment response (i.e., pharmacodynamics, PD). In population PK (PopPK) or physiological-based PK (PBPK) models, patient characteristics are often included as covariates or parameters, such as age, weight and creatinine concentrations. For PD or QSP models, treatment outcomes are often modeled in relation to patient-specific disease- or physiology-related characteristics, such as disease severity and baseline values of biomarker levels. In both scenarios, these models together with specified patient-specific properties can be used to derive individualized dosing schedules or optimized trial design which takes into account these patient-specific differences. For simplicity, covariates in population PK/PD and patient-specific parameterizations in PBPK and QSP models will be referred to as covariates.

Covariates are often correlated. Therefore, when patient-specific covariates are incorporated into pharmacometric simulations, particularly when existing models are reused for new simulation purposes, it is important to ensure realistic combinations of patient covariates are generated. Several approaches are currently used, such as bootstrapping, which requires the availability of underlying individual-level data. However, when individual level data are not available, alternative approaches are needed, like univariate or multivariate normal distributions (MVN) fitted on the covariate data from a relevant population [1, 2]. These methods can, however, fail to realistically capture the dependency structures among patient characteristics, and thus do not accurately reflect the real-world data [3, 4, 5].

Copulas offer a reliable way to model and simulate complex dependencies among patient characteristics, enabling more realistic and reproducible virtual populations for a variety of research and regulatory applications [3, 6]. In this tutorial, we introduce the theoretical concept of copulas in context of pharmacometrics (**Section 2.2**). Next, we provide a step-by-step practical guide with an example of covariate simulation in R, including evaluation of copula fits, allowing you to start using copulas right away (**Section 2.3**). Lastly, we discuss a number of specific issues and guidance for appli-

cations, including cases with discrete covariates (**Section 2.4**).

2.2 Theory

This section provides a thorough statistical understanding of copula concepts. It is however possible to first explore copula estimation yourself (**Section 2.3**) and refer back here as needed.

2.2.1 Statistical foundations

2.2.1.1 Probability density

Some covariate values are more likely to occur than other values, which is described by the probability distribution. For continuous covariates, this distribution can be expressed as a probability density function (PDF, $f(X)$). The PDF can be obtained by calculating the frequency of values occurring within certain value ranges, and is often visualized using a histogram. A PDF can also be defined by a parametric or non-parametric function from a certain distribution, such as a normal or log-normal distribution.

PDFs can describe the density of more than one covariate at once, which is called bivariate or multivariate density functions. If we consider a situation with two covariates (X_1 and X_2), their PDF ($f(X_1, X_2)$) is called the joint PDF and describes the relative likelihood of observing the combination of values for X_1 and X_2. The covariates can be correlated or dependent on each other, which is reflected in their joint distribution, also referred to as the dependency structure. In the case of two or multiple covariates, the marginal distribution of a covariate is obtained by considering its distribution regardless of other covariates, denoted by $f_1(X_1)$. This distribution can be viewed as the projection of a joint distribution onto a single dimension. Marginal distributions and the dependency structure are fundamental aspects when characterizing a covariate dataset.

A PDF can be estimated from data by choosing a function and estimating its parameters. The function can be either a parametric distribution, for example the (multivariate) normal distribution, or a non-parametric function, which does not assume a specific parametric distribution, like a kernel density function. Parameters are usually estimated by maximizing the likelihood. For a one-dimensional density or a marginal density, this optimization is relatively straightforward. However, the complexity of estimating a joint PDF increases with dimensionality.

2.2.1.2 Conditional probability density

The density of one covariate (e.g. X_1) conditioned on the value of another covariate (e.g. X_2) is called the conditional density ($f(X_1|X_2)$). Its relationship with joint density $f(X_1|X_2)$ and marginal density $f_2(X_2)$ is illustrated in **Eq. 1**.

$$f(X_1 \mid X_2) = \frac{f(X_1, X_2)}{f_2(X_2)} \qquad \textbf{(Eq. 1)}$$

Similarly, the joint density of two covariates conditioned on the third covariate is

the conditional joint density ($f(X_1, X_2|X_3)$). For example, the correlation between weight and height can vary by age, with a stronger correlation at a younger age. This can be denoted as $f(height, weight|age)$, the joint density of height and weight at a certain age.

2.2.1.3 Probability and cumulative density functions

A probability that a covariate falls within a certain range of covariate values is given by the integral of the PDF over that range. This integral is called the cumulative distribution function (CDFs, $F(x)$), which is defined as the probability of a covariate being a value less than (or equal to) a specific value of x ($P(X \leq x)$). Similarly, the CDF for two covariates ($F(X_1, X_2)$) represents the probability that two covariates are less than specific values of X_1 and X_2. The CDF can be estimated either nonparametrically, such as through data ranks or kernel density estimation, or parametrically by fitting a chosen CDF function (e.g., normal/lognormal distribution).

2.2.1.4 Uniform distribution

The last main concept required to understand copulas is the uniform distribution. A uniform distribution is when every value on the interval has the same density. Covariates are rarely uniformly distributed, but it is possible to transform the covariates to uniform distributions with probability integral transform (PIT, Figure 2.1) [7]. Briefly, for any covariate X with a PDF $f(X)$ and a CDF $F(X)$, the new random variable $U = F(X)$ follows a uniform distribution on $[0, 1]$ (derivation provided in **Supplementary material S1**). From uniform (U) scale, the data also can be easily mapped back to the original scale with the inverse of CDF ($F^{-1}(U)$).

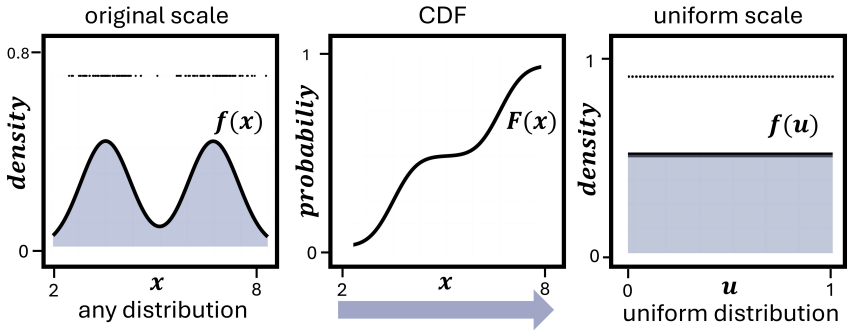


Figure 2.1: Probability integral transform. Transforming a covariate x with an original distribution (left) to a uniform distributed covariate u (right) using the CDF (middle). The total area under the PDF, represented by the shaded region, equals 1 for both x and u. CDF, cumulative distribution function; PDF, probability density function.

2.2.1.5 Copula

Copulas are functions describing multivariate distributions with marginal distributions that are uniform on $[0, 1]$ [8]. They are used to model the non-deterministic dependence between covariates. Therefore, derived covariates (e.g., body mass index) should not be included together with their components (e.g., height and weight). Throughout this section, we assume that all covariates are continuous and possess a joint density function. Under this assumption, Sklar's theorem implies that the joint density function of any pair of covariates ($f(X_1, X_2)$) could be expressed as the product of their marginal distributions and a copula function (**Eq. 2**). Here, $c(U_1, U_2)$ represents the copula.

$$\begin{aligned} f(X_1, X_2) &= c(U_1, U_2)\, f_1(X_1)\, f_2(X_2) \\ U_1 &= F_1(X_1), \quad U_2 = F_2(X_2) \end{aligned} \tag{Eq. 2}$$

The estimation of a copula can be done independently from the description of the marginal distribution. Since any marginal distribution can be transformed into a uniform distribution, copulas offer great flexibility for different marginal distributions that covariates may exhibit. Copulas include a wide variety of functions, including multivariate Gaussian (normal) distribution, but also any other (non-)parametric distributions. This flexibility allows copulas to capture diverse data structures. A bivariate copula describes the dependence structure between two covariates (Section Bivariate copula), and a vine copula extends the framework to multiple covariates (Section Vine copula). A more in-depth guide to the mathematical concepts of copulas is provided by Czado (2019) [9].

2.2.1.6 Bivariate copulas

Bivariate copulas, or pair copulas, are copulas for two covariates. According to Sklar's theorem, the joint distribution of two covariates X_1 and X_2 can be expressed as **Eq. 2**, where the $c(U_1, U_2)$ is the bivariate copula. Various bivariate copula functions are available to describe different shapes of correlations, including symmetric or asymmetric, linear or non-linear, monotonic or non-monotonic, and tail dependencies, where dependency differs over the values of the covariates [9, 10]. Bivariate copula functions include both parametric and non-parametric, depending on whether a specific form is assumed.

A variety of copula functions are available in statistical software, and these functions allow for different dependency patterns. For instance, the Gaussian copula is a frequently used function to handle symmetric dependencies, and Joe or Clayton copulas are suited for capturing tail dependency (Figure 2.2). Note that the multivariate Gaussian distribution is a specific type of joint distribution with Gaussian marginal distributions and a Gaussian copula. Non-parametric approaches, such as polynomials or kernel density estimators, approximate the actual dependency structure without assuming a parametric shape [11]. Compared to parametric copulas, non-parametric copulas offer more flexibility, making them more suitable for modeling complex dependencies, including non-monotonic dependencies. However, non-parametric estimation typically comes with higher computational costs and may encounter issues

due to data sparsity.

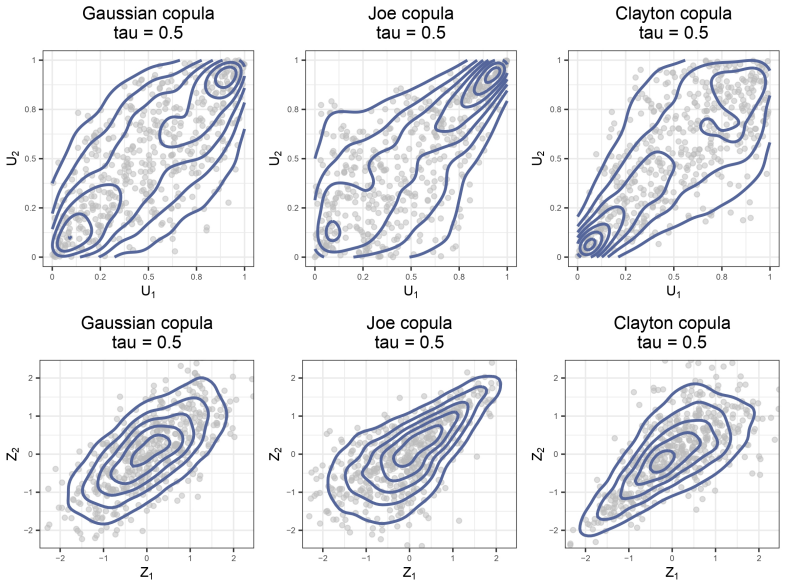


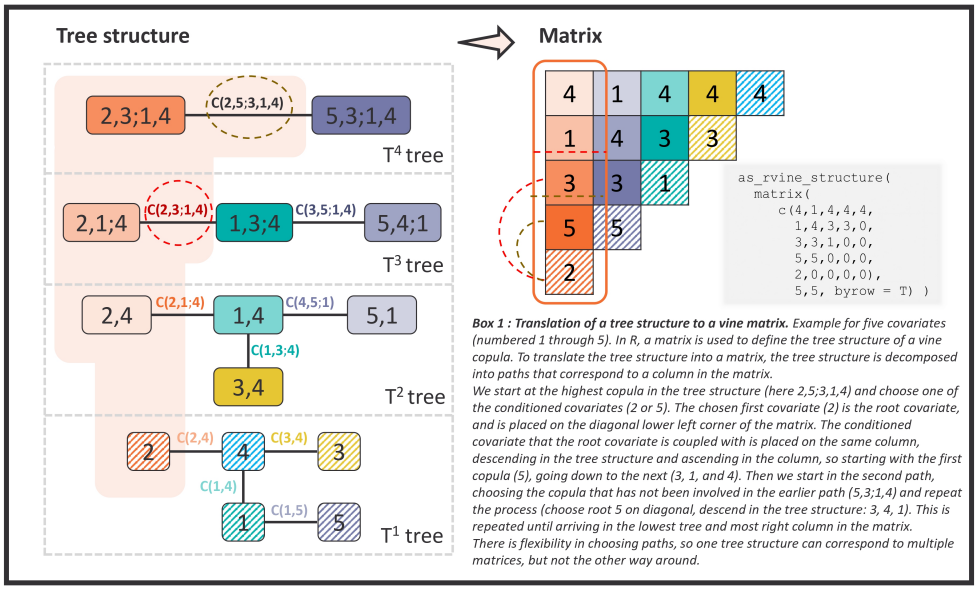
Figure 2.2: Two-dimensional density contour plots. Example bivariate datasets that exhibit different dependency patterns but share the same correlation (Kendall's tau). U_1 and U_2 represent the uniform scale covariate data, and Z_1, Z_2 represent the covariate data on a standard normal scale. The marginals of data were transformed from the uniform scale to the standard normal scale using standard normal quantile functions.

2.2.1.7 Vine copulas

While Sklar's theorem generalizes to 3 or more dimensions, the copula function for d dimensions ($c(F_1(X_1), F_2(X_2), \ldots, F_d(X_d))$) becomes exponentially more complex. Rather limited techniques are available to construct a multivariate copula [12]. To circumvent the problem, a *d*-dimensional copula can be constructed by decomposing the multivariate density into a set of bivariate copulas, called pair-copula construction [13]. Because the graphical representation of these copulas is similar to grape vine, copulas built via pair-copula constructions are called vine copulas.

Vine copula estimation involves an iterative process of structure learning and parameter estimation. The model structure of a vine copula, a tree structure that consists of a sequence of trees, specifies how covariates are associated or correlated with each other. A full tree structure for a d-dimension covariate dataset consists of $d-1$ trees, and the kth tree (T_k, k = 1,2,..,d-1) constitutes $d-k$ (conditional) bivariate copulas. For example, a full vine tree structure for five covariates consists of a sequence of four trees, starting with the first tree T_1 at the bottom and ending at the last tree T_4 at the top. (**Box 1**, left panel). T_1 tree defines the first set of pair copulas (edges), which captures the direct association between covariates (nodes). The second tree, T_2, captures the conditional copula between the copulas from T_1,

where the T_1 copulas become the nodes and conditional copulas are the edges. This procedure continues, where the edges of each lower-level tree become the nodes in the next higher-level tree, until all trees in the sequence are formed. Conditional copulas are denoted using a semicolon, e.g. c(1,3;4) for copulas $c(1,4)$ and $c(3,4)$.



Box 1 : Translation of a tree structure to a vine matrix. *Example for five covariates (numbered 1 through 5). In R, a matrix is used to define the tree structure of a vine copula. To translate the tree structure into a matrix, the tree structure is decomposed into paths that correspond to a column in the matrix.*
We start at the highest copula in the tree structure (here 2,5;3,1,4) and choose one of the conditioned covariates (2 or 5). The chosen first covariate (2) is the root covariate, and is placed on the diagonal lower left corner of the matrix. The conditioned covariate that the root covariate is coupled with is placed on the same column, descending in the tree structure and ascending in the column, so starting with the first copula (5), going down to the next (3, 1, and 4). Then we start in the second path, choosing the copula that has not been involved in the earlier path (5,3;1,4) and repeat the process (choose root 5 on diagonal, descend in the tree structure: 3, 4, 1). This is repeated until arriving in the lowest tree and most right column in the matrix.
There is flexibility in choosing paths, so one tree structure can correspond to multiple matrices, but not the other way around.

For each (conditional) pair copula, a bivariate copula function and its parameters are chosen and estimated independently, constituting a flexible framework for the estimation of a joint density function. Using the copula, the mathematical expression of joint density for the five covariates can be expressed as the product of their respective marginal distribution and all (conditional) bivariate copulas (**Supplementary material S2**).

2.3 Practical workflow for copula development

In this section, we will first explain the procedure to estimate a copula model with a dataset of an observed (real-world) population. Next, we will demonstrate how to evaluate a copula by assessing whether the covariates simulated from a copula (virtual population) are representative of the observed covariates (Figure 2.3). Lastly, we will briefly showcase the application of covariates simulated from a copula in PK simulations.

Software

The example code provided in this tutorial was developed in R 4.5.2. For updated code compatible with the latest version of R, refer to the *pmxcopula* package (`https://github.com/vanhasseltlab/pmxcopula`). Functions for kernel density integral

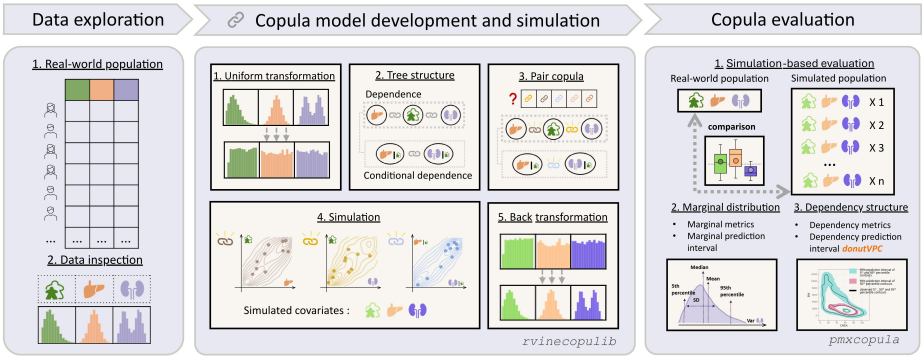


Figure 2.3: Flow chart of copula estimation and simulation using vine copulas. For each step, the corresponding R packages used in this tutorial are noted in the boxes.

transformation and copula estimation are from the R packages *kde1d* (v 1.1.1) and *rvinecopulib* (v 0.7.3.1.0), respectively [14, 15]. Example datasets `MIMIC_5cov` and `mix_data`, together with the evaluation tools are provided by the package *pmxcopula* (v 0.1.1). Pharmacokinetic simulation is done using *rxode2* (v 4.1.0) [16]. To navigate the package and its functionality, a summary table of key functions for copula development, simulation, and evaluation is provided in **Supplementary material S3**.

2.3.1 Data preparation

A copula can be estimated on data, which should represent the intended population and covariates of interest. The selection of relevant population and covariates depends on the research question. For example, the aim could be to simulate biomarkers in cancer patients, then the data need to include biomarker information of that specific population. For covariate selection it is important to consider the generalizability or specificity of the copula for future applications. By covering a relatively wide range of relevant covariates, the estimated copula can support the covariate simulation across various scenarios, however, more dimensions in the copula require more observations (**Section 2.4.2**). Lastly, secondary covariates that can be calculated from primary covariates should be removed from copula estimation to avoid adding deterministic elements in the stochastic model. For instance, if body weight and height are included, BMI should be excluded.

For the practical example, we consider `MIMIC_5cov` dataset from the *pmxcopula* package, which consists of five covariates from 1000 ICU patients, based on the MIMIC database [3]. Covariates are age, weight, blood urea nitrogen (BUN), serum creatinine and serum albumin. Heavily skewed marginal distribution can result in non-uniform transformed data with PIT. To improve uniformity, we use log-transformed BUN (logbun) and creatinine (logcrea) for copula model development.

```
library(kde1d)
library(pmxcopula)
```

```
library(dplyr)
data("MIMIC_5cov")
```


2.3.2 Copula estimation

2.3.2.1 Uniform transformation

Copula modeling typically begins with a uniform transformation of the margins, where the distribution of each covariate is rescaled to a uniform scale. Uniform distributed covariate data is obtained via PIT, which transforms any marginal distribution into a uniform distribution with the CDF of that covariate. CDF can be obtained with a fitted parametric distribution function, such as (log)normal distribution,

```
age_unif <- pnorm(MIMIC_5cov$age,
                  mean(MIMIC_5cov$age),
                  sd(MIMIC_5cov$age))
```

or a non-parametric distribution function, such as kernel density estimate (KDE) of the distribution, for example using the R package *kde1d* [15].

```
crea_kde <- kde1d(MIMIC_5cov$logcrea, mult = 1)
crea_uniform <- pkde1d(MIMIC_5cov$logcrea, crea_kde)
```

KDE-based distributions are constructed with a smooth function and can learn the shape of density from the data without specifying a particular parametric form [17]. Here, a bandwidth multiplier of `mult = 1` was used to ensure standard smoothing. Given its flexibility, KDE can be used for each covariate in a dataset, for example using a for-loop or `lapply` function. The kde1d estimates both the CDF used for PIT ($F(x)$), and the inverse of CDF ($F^{-1}(U)$) , which is essential for transforming simulated data from the uniform distribution back to the original data scale in the later section Simulation from the model developed with `vinecop()`.

```
kde_list <- lapply(MIMIC_5cov, kde1d)
MIMIC_unif <- mapply(function(x, kde) pkde1d(x, kde),
                     MIMIC_5cov, kde_list) %>% as.data.frame()
```

2.3.2.2 Copula fitting

A vine copula is fitted on the transformed data through pair copulas constructions, which capture the density between different pairs of covariates and copulas (see **Section 2.2.1.7**, **Box 1** left panel). This process consists of two steps: structure learning and parameter learning.

2.3.2.2.1 Structure learning The tree structure of a vine copula can be arbitrarily chosen, but the fit of a copula can be improved by placing pairs of covariates that are more correlated closer to each other in the vine.

The standard approach to determine a tree structure in *rvinecopulib* is the maximum spanning tree (MST) algorithm. In this algorithm, Kendall's tau correlations are

first calculated for all possible covariate pairs and covariates are paired in the tree so the sum of the correlations for $d-1$ covariate pairs is maximized [18]. Next, bivariate copula functions are fitted for the selected pairs and parameters are estimated (**Section 2.3.2.2.2**). For the second tree, the MST selects the copula pairs that yield the strongest conditional correlations given the shared covariate. This procedure is iterated for each subsequent tree until all trees are selected and estimated.

There can be reasons for specifying a different tree structure, such as previous knowledge about which covariates are expected to be exhibit stronger associations, or the interpretability of the copula between the covariates of interest in the first tree. This is possible within *rvinecopulib* by translating a part of, or a whole tree structure into a matrix that can be specified in R (**Box 1**).

2.3.2.2.2 Parameter learning For each selected covariate pair in the vine structure (**Box 1**), a bivariate copula function is fitted. A candidate bivariate copula function is first specified, followed by parameter estimation. The *rvinecopulib* package offers a variety of bivariate copula functions (Table 1). Instead of specifying a particular copula function for a covariate pair, a collection or family of candidate copula functions can be defined, from which the best fitting one is selected based on criteria such as the Akaike information criterion (AIC) or Bayesian information criterion (BIC). The choice of copula functions requires balancing flexibility and computational efficiency, which can also introduce overfitting (Section Data Size). Including a wider range of bivariate copulas increases flexibility but may lead to higher computational costs, especially with nonparametric copulas. To increase computational speed, nonparametric options can be excluded by using predefined sets, such as the ”parametric” copula family.

Table 2.1: Bivariate copula functions available for copula estimation in the package `rvinecopulib`.

Type	Name	Argument	Families
–	Independence	`indep`	`all, nonparametric, parametric, itau`
Elliptical	Gaussian	`gaussian`	`all, parametric, onepar, elliptical, itau`
Elliptical	Student t	`student`	`all, parametric, twopar, elliptical, itau`
Archimedean	Clayton	`clayton`	`all, parametric, onepar, archimedean, itau`
Archimedean	Gumbel	`gumbel`	`all, parametric, onepar, archimedean, itau`
Archimedean	Frank	`frank`	`all, parametric, onepar, archimedean, itau`
Archimedean	Joe	`joe`	`all, parametric, onepar, archimedean, itau`
Archimedean	Clayton–Gumbel (BB1)	`bb1`	`all, parametric, twopar, archimedean, BB`
Archimedean	Joe–Gumbel (BB6)	`bb6`	`all, parametric, twopar, archimedean, BB`
Archimedean	Joe–Clayton (BB7)	`bb7`	`all, parametric, twopar, archimedean, BB`
Archimedean	Joe–Frank (BB8)	`bb8`	`all, parametric, twopar, archimedean, BB`
Extreme-value	Tawn	`tawn`	`all, parametric, threepar, ev`
Nonparametric	Transformation kernel	`tll`	`all, nonparametric`

```
library(rvinecopulib)
MIMIC_copula_unif <- vinecop(MIMIC_unif, family_set = "parametric")
```

Once a copula is fitted, its tree structure can be obtained as matrix form and visualization.

```
MIMIC_copula_unif$structure
plot(MIMIC_copula_unif, tree = "ALL")
```

2.3.2.3 Single-function copula development

While following the above steps sequentially provides flexibility and helps build intuition, it is reassuring that this is not always necessary. The *rvinecopulib* package offers the powerful `vine()` function, which combines all prior steps into one function.

```
MIMIC_copula <- vine(MIMIC_5cov,
                     copula_controls = list(family_set =
                         "parametric"),
```

```
                    margins_controls = list(mult = 1))
```

The function `vine()` internally uses kernel density estimation (*kde1d*) for the transformation of covariates into uniform scale and fits a copula. The arguments `copula_controls` and `margins_controls` correspond to the arguments in `vinecop()` and `kde1d()`, respectively. For the copula model fitted with `vine()`, the tree structure is contained within the copula component and must be accessed via, for example, `MIMIC_copula$copula$structure`. Other functions can be applied following the same convention.

2.3.2.4 Copula simulation

Covariate simulation is often the ultimate goal of copula modeling. Once the copula has been estimated, covariates can be simulated by drawing random samples from the copula. This will produce a dataset with n rows and d covariates as columns. One point to keep in mind is that the scale of simulated outcomes differs depending on whether the model was fitted with `vinecop()` or `vine()`.

2.3.2.4.1 Simulation from the model developed with vinecop() The covariates simulated from the model developed with `vinecop()`, i.e., `MIMIC_copula_unif`, are on a uniform scale. The syntax for generating samples is similar to other random sampling functions in base R, such as `rnorm()`.

```
MIMIC_sim_unif <- rvinecop(100, MIMIC_copula_unif)
```

To transform the uniformly distributed data back to the original scale, we use the inverse of the CDF, the quantile function $F^{-1}(u)$. In case of a parametric distributions, the (base) R functions can be used to back-transform the data using `q<dist>()` (e.g., `qnorm()`). For kernel densities, the equivalent function `qkde1d()` can be used. Here, we need to enter the same distribution object as used in **section 2.3.2.1**. We apply the quantile function to each covariate using `mapply()`.

```
MIMIC_sim <- mapply(function(x, kde) qkde1d(x, kde),
        MIMIC_sim_unif, kde_list) %>% as.data.frame()
```

The outcome (`MIMIC_sim`) is a data frame with the simulated covariates on the original scale.

2.3.2.4.2 Simulation from the model developed with vine() Covariates at the original scale can be directly simulated from the copula object estimated with `vine()` function, e.g. `MIMIC_copula`. To prevent sampling out of the feasible space, it is possible to set boundaries (minimum and maximum) for marginal characterization during copula estimation.

```
MIMIC_sim <- rvine(100, MIMIC_copula)
```

In case of the situation where one specific subgroup population is relevant for further pharmacometric simulation, conditional sampling is supported by *pmxcopula*, with

the option to define the population of interest with constrained conditions for both continuous and discrete variables.

```
MIMIC_VP <- rscopula(copula = MIMIC_copula,
   n_sim = 100,
   xmin = c(18, 50, NA, NA, NA),
   xmax = c(50, 120, NA, NA, NA),
   var_types = c('c', 'c', 'c', 'c', 'c'))
```


2.3.2.5 Evaluation methods

The evaluation methods for copulas differ from those of pharmacometric models. Copula evaluation aims to not only assess how well the model fits the data, but also to examine whether the copula fulfills its intended purpose: simulating virtual populations that accurately represent the observed population. This section will cover how to critically examine the performance of a copula model.

2.3.2.5.1 Evaluation of model fit Different approaches can be used to evaluate the overall performance of a copula. Goodness-of-fit (GOF) measures, such as log-likelihood, AIC and BIC, can be used to select the most suitable copula. Overall model fit can be evaluated using the GOF measures, as well as the GOF tests based on the statistics of, for instance, Kolmogorov-Smirnov statistic, Cramér-von Mises statistic and Kullback-Leibler (KL) divergence [9, 19, 20]. These GOF tests, often summarized with bootstrapped *p*-values, can compare the performance of candidate copula models and facilitate the model selection. However, these summary metrics do not provide detailed and diagnostic insights into how well the copula is fit for purpose.

2.3.2.5.2 Evaluation for covariate simulation For covariate simulation, the purpose of copula evaluation is to determine whether simulated populations resemble the observed population. Therefore, we propose to evaluate a copula with simulation-based diagnostics. This involves comparing distribution properties between the observed population and simulated populations, considering two aspects: (1) the marginal distribution, which is the probability distribution of each individual covariate, such as age, and (2) the dependency structure, which is the relationships between covariates [6]. Both aspects can be evaluated visually and quantitatively. The simulation-based tools in *pmxcopula* also facilitate comparison of different covariate simulation methods, including but not limited to copulas. In this section, the performance of `MIMIC_copula` on the overall population is assessed.

2.3.2.5.3 VP simulation for evaluation A strategy involving multiple simulations of the original population is generally recommended to account for and assess the sampling variability, for example, 100 repeated simulations. In function `rcopula()`, `sim_nr` is specified to indicate the number of simulation replicates. Each replicate contains the same number of individuals as the original dataset. Next to the columns

representing each covariate, a column is included to indicate the simulation run in the simulated dataset. This column is required for subsequent evaluation functions.

```
set.seed(12345)
MIMIC_sim <- rcopula(MIMIC_copula, sim_nr = 100)
```


2.3.2.5.4 Visualization of distributions Density plots are a visual method to examine how the copula describes the observed data. The function `plot_1dist()` visualizes the density curves for each covariate from observed and multiple simulated datasets (Figure S2.1). To assess the dependent relations, the function `plot_mvdist()` displays two-dimensional density contours or scatter plots for all pair combinations of covariates, comparing the observed data with one of the simulated datasets (Figure S2.2). A density contour is a line or curve representing the region of equal probability density derived from the data. Together, these functions offer a quick overview of the model fit. Ideally, the density curves and density contours of the observed and simulated populations should closely align.

```
plot_1dist(sim_data = MIMIC_sim,
           obs_data = MIMIC_5cov,
           sim_nr = 100,
           pick_color = c("#F68E60","#747AA9"))

plot_mvdist(sim_data = MIMIC_sim,
            obs_data = MIMIC_5cov,
            pick_color = c("#F68E60","#747AA9"),
            plot_type = "density",
            bins = 8)
```

2.3.2.5.5 Quantitative evaluation metrics Quantitative evaluation metrics can assist copula evaluation in a quantitative manner. For this purpose, mean, median, standard deviation, 5^{th} and 95^{th} quantiles are included as marginal metrics in *pmxcopula* package. These metrics measure the centrality, limit behaviors, and variability of each covariate. The differences in marginal metrics between virtual and observed populations can be expressed as relative errors. Relative errors close to zero imply that the margins of virtual populations are in good agreement with those of the observed population.

```
calc_margin(sim_data = MIMIC_sim,
            obs_data = MIMIC_5cov,
            sim_nr = 100,
            var = NULL,
            aim_statistic = c("mean", "median"))
```

To quantify the difference between the dependency structure of observed and simulated distributions, two measures are available in the *pmxcopula* package. One

is Pearson's correlation, which quantifies the linear relationship. When the Pearson correlations between observed and simulated covariates are close, it suggests that the linear correlation between covariates has been well captured. Another measure is the overlap metric, which assesses the non-linear dependencies. Overlap metrics account for the shape of dependency pattern by calculating of overlaps between the bivariate density contours from observed covariates and those of simulated covariates (**Box 2**). The overlap metric quantifies the ratio of intersection area to union area of two density contours at the same percentile, also known as the Jaccard index. A high overlap relates to a good estimation of the dependency structure.


```
calc_dependency(sim_data = MIMIC_sim,
                obs_data = MIMIC_5cov,
                pairs_matrix = NULL,
                percentile = 95,
                sim_nr = 100,
                cores = 4)
```

GOF metrics are implemented in *VineCopula*, *gofCopula*, and *kldest* packages [19, 20, 21]. In *VineCopula* and *gofCopula*, GOF tests using Kolmogorov-Smirnov and Cramér-von Mises statistics could be used to test the difference between the copula and observed population using bootstrap-based *p*-values, where rejecting the null hypothesis implies a bad fit of the copula. In `kldest`, KL divergence measures the distance between the distributions of the observed and virtual population, with lower KL values indicating the copula more closely resembles the observed population.

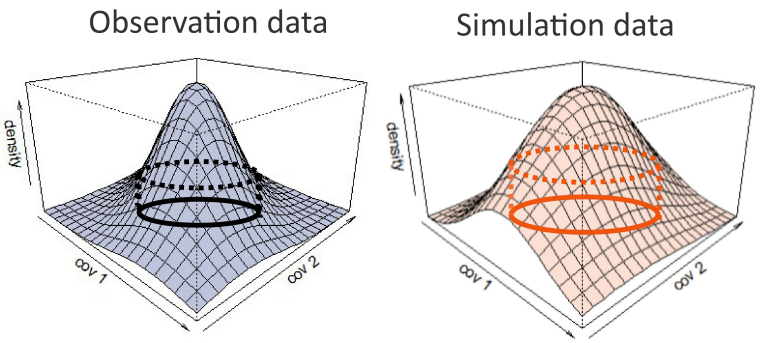
```
library(VineCopula)
copula_model <-  RVineStructureSelect(MIMIC_unif,family = c(0:4))
MIMIC_GOFtest <-  RVineGofTest(MIMIC_unif,
                               copula_model,
                               statistic = "CvM",
                               B = 100)
MIMIC_GOFtest[["p.value"]]

library(kldest)
kld_est_nn(MIMIC_sim %>% select(-simulation_nr), MIMIC_5cov)
```

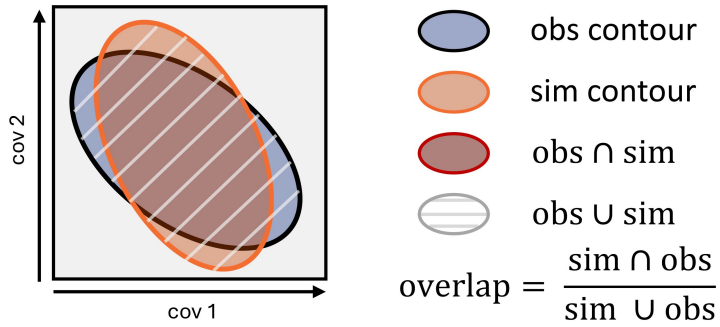
2.3.2.5.6 Visual predictive checks Visual predictive checks (VPCs), commonly used for the evaluation of pharmacometrics models, are also applicable to copulas. For marginal distributions, 95% prediction intervals across all quantiles (from 1st to 99th) from multiple simulations are plotted versus that of the observed population in quantile-quantile plot (Q-Q plot). A good fit corresponds to the ribbon tightly aligning the diagonal line (Figure S2.3).

```
plot_qq(sim_data = MIMIC_sim,
        obs_data = MIMIC_5cov,
        sim_nr = 100,
```

Step 1: obtain the bivariate density contour



Step 2: calculate the Jaccard index



Box 2 : Illustration of the overlap measure. A specified density contour is a line going through the points that share a specific probability density, which was estimated from kernal density estimate. The percentiles of a contour is cumulative probability density, which indicates the proportions of data point covered. We usually use the 95th percentile contour for the calculation of overlap metrics, which encloses 95% of points. For this contour, the overlap metric can be calculated, which is the Jaccard index of the density contours of observed and simulated data. Jaccard index is calculated as the ratio of intersection area to union area.

```
        conf_band = 95,
        var = c("age","weight"),
        type = "ribbon")
```


A 2-dimensional VPC, named donutVPC, was developed to evaluate the model fitness for the pair-wise relations (Figure 2.4). Similar to the traditional pharmacometric VPC [22], donutVPCs are constructed with multiple simulations but focus on density contours. A density contour at the p^{th} percentile is estimated using kernel density estimation (KDE) at the region containing p% of the simulated individuals. The donutVPC is constructed by simulating new samples from the copula and estimating the KDE for each sample. Then the contour points are extracted for this p^{th} percentile. Around those contour points a new contour is estimated to get an approximation of the specified confidence band around the contours (e.g. 95% confidence, **Supplementary material S4**). The bands for the percentile contours are then plotted versus the observed data contours, producing a donut shaped plot. Traditional VPCs would include data points, however a scatter plot is not directly interpretable compared to density contours. A close overlap between the prediction bands and observed contours suggests that the copula accurately represents the observed population.

```
donutVPC(sim_data = MIMIC_sim,
         obs_data = MIMIC_5cov,
         percentiles = c(5, 50, 95),
         sim_nr = 100,
         pairs_matrix = matrix(c("age", "age", "weight",
             "logcrea"), 2, 2),
         conf_band = 95,
         colors_bands = c("#F68E60", "#FAC1A8"),
         cores = 4)
```

2.3.2.5.7 Improve copula fit Evaluation results could detect discrepancies between virtual populations and the observed population. It is possible that copula models fail to characterize subsets of covariates or subgroup populations. This section outlines three aspects relevant to improving a copula model.

Lack of fit may arise when the transformed data deviate from a uniform distribution. A proper choice of uniform transformation could improve the copula. In the case of a heavily skewed original marginal distribution, log transformation before PIT reduces skewness and improves the uniformity of the transformed covariates.

Inadequate model performance may result from the misspecification of the dependency structure. When data is insufficient to fit a variety of bivariate copula functions or estimate the full dependency structure, restricting bivariate copula options during vine estimation is recommended. However, this requires careful consideration, as the choice of bivariate copulas can have a significant impact on copula goodness-of-fit. For example, Gaussian vine copulas use only Gaussian bivariate copulas for all covariate pairs, which overlook asymmetric or tail dependencies. This limitation

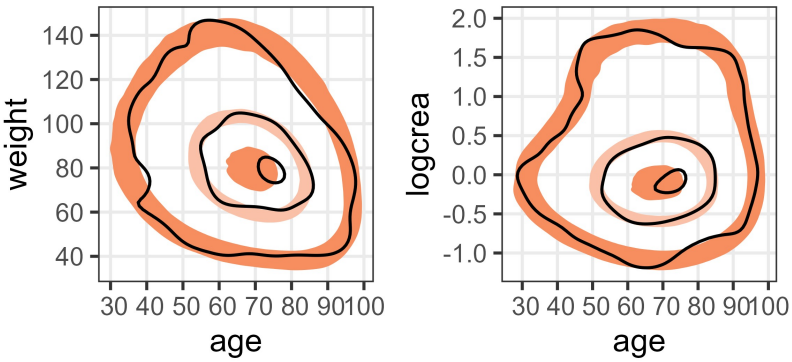


Figure 2.4: DonutVPC, visual predictive check for dependency structures. The shaded areas represent the 95% prediction interval of the 5^{th},50^{th} and 95^{th} percentile bivariate density contours of multiple simulations, and the black solid lines represent the density contours of the observed population.

can potentially lead to incorrect predictions, which contributed to the underestimation of risk in the 2008 financial crisis [23]. In most cases, a family of parametric bivariate copulas (parametric) is sufficient for modeling covariate relationships. However, if covariates exhibit unusual or complex dependency patterns that parametric copulas cannot capture, non-parametric bivariate copulas, such as kernel-based copulas, may provide a better fit.

Poor fits can also arise from outlier subpopulations. Such subpopulation can be difficult to detect without subgroup evaluation, since the overall fit of the copula model may still appear satisfactory. If the poor fits are identified within certain subpopulations, e.g. male Asian population, this likely indicates that the covariate dependency structure for this group differs from that of the overall population. In such cases, one can either exclude this subgroup and estimate a separate copula model, or proceed with caution using the existing copula, acknowledging its limitations if there is insufficient data to support a separate analysis.

2.3.2.6 Applications in pharmacometrics

Once a copula has been estimated and evaluated, it can be used to generate virtual populations with realistic covariate distributions. These simulated populations can be easily integrated into pharmacometric simulations, such as population PK, PD, PBPK, and QSP model-based simulations. By ensuring that the covariates or physiological parameters reflect realistic variability and correlations, copulas enhance the credibility and predictive performance of pharmacometric analyses. This approach enables the evaluation of dosing regimens in an untested population and supports individualized treatment strategies across diverse patient populations. Example code is included in the **Supplementary material S5**. In addition to virtual population generation, copulas have a wide set of potential uses in pharmacometrics. One of these

uses is to learn parameter uncertainty from bootstrap or sampling importance resampling results [24], which can then be used to simulate parameters, incorporating the parameter uncertainty.

2.4 Considerations for copula application

2.4.1 Discrete data

The first two sections focus on theory and practical examples within a continuous framework, however, copulas generalize to discrete covariates [25]. A common approach for modeling discrete covariates in the copula framework is to render discrete covariates continuous by assigning an order and adding small noise. The jittered data become continuous, and the original algorithm applies [26]. In rvinecopulib, this method is automatically applied if the covariate is specified as discrete (`"d"`).

As ordering is one of the intrinsic natures of continuous data, an order for the discrete covariate is required. For ordered categorical covariates, such as disease stage (stage I, II, III, IV) and pain scale (0 to 5), ordering can be set as the inherent order using the base R data type ordered. Then, the discrete covariate is converted into a factor variable with an order, and the copula can be fitted with the specification of covariate data type.

To illustrate copula estimation with both discrete and continuous covariates, we consider a mixed dataset `mix_data` from *pmxcopula*. The dataset includes sex, age, weight and albumin level. Although age is often treated as continuous, its integer nature allows it to be modelled as discrete. In this example, age is treated as discrete to demonstrate copula modeling with mixed data, with sex and age handled as ordered categorical covariates.

```
data("mix_data")
mix_data$Sex <- ordered(mix_data$Sex,
                        levels = c("male", "female"),
                        labels = c("1", "2"))
                        mix_data$Age <- ordered(mix_data$Age)
                        vine_discrete <- vine(dat = mix_data,
                        copula_controls = list(
                        family_set = "parametric",
                        var_types = c("d","d","c","c")))
```

In the case of dichotomous covariates, the order is arbitrary; for ordinal covariates, the order is known. There are, however, some limitations when dealing with unordered categorical covariates, also known as nominal covariates. If the order is unknown, for example for ethnicity and disease classification, the copula is not always able to provide a good fit, due to non-monotonicity. With a small number of unordered discrete covariates and categories, a good fit can be found by brute forcing, going through all order combinations and choosing the one with the best performance. An alternative approach is one-hot encoding, i.e., converting nominal variables into bi-

nary indicators. For example, a three-category variable can be encoded as $[1, 0, 0]$, $[0, 1, 0]$ and $[0, 0, 1]$ before fitting a copula. However, a key limitation is that a copula can simulate values that do not correspond to exactly one category (e.g., $[1, 1, 0]$). Although vine copulas provide useability for categorical covariates [14], more mathematical limitations have been discussed in literature as well [25, 27].

To evaluate the performance of copulas involving discrete covariates, the marginal distributions can be evaluated by comparing the number of simulated individuals in the category of interest (e.g., 0-10 years, 10-20 years) with the counts in the observed population. For both marginal and dependency structure, subgroup analysis is an option to detect any misfit within subgroup populations [6].

2.4.2 Data size

Copula estimation generally requires a large number of observations to obtain a reliable and robust density estimation. The required number of observations grows with the number of dimensions or covariates. For a dataset with d dimensions, the full vine copula structure includes $d(d-1)/2$ pair copulas. If using parametric bivariate copula functions, each bivariate copula requires the estimation of one or two parameters. For a 20-dimensional dataset, there are 190 bivariate copulas and 190-380 parameters to be estimated.

A rule of thumb is that an adequate dataset should have observations (n) more than the dimension (d) raised to the power of 4 (d^4) [28]. However, this can be different depending on the data and how much prior information or assumptions can be made. For instance, weak, medium or strong dependence relations results in different probability of successful identification of the copula model [29]. If prior knowledge, such as the (conditional) independence relationships for certain covariate pairs, is available, vine copula structure can be simplified to improve robustness of the estimation. This is particularly important for small datasets, where model simplification helps reduce the risks of overfitting and computational burden due to the large number of possible model parameters. The risk of overfitting can also be reduced by assuming a parametric copula instead of a non-parametric copula.

Copula truncation and pruning are often used to reduce model complexity by limiting the vine structure. Specifically, truncation sets bivariate copulas to be independent in the trees above a certain level, and pruning assumes independence for covariate pairs that have weak correlations. For datasets with a large number of covariates, it is possible to assume bivariate copulas in higher trees are independent [18]. To determine an optimal truncation level, criteria such as the BIC, modified BIC and Vuong test were proposed [30]. However, when all covariates are correlated, the correlation in higher level trees may remain significant. In such cases, truncating the entire tree might result in the loss of information. Instead, only pruning the covariate pairs that have weak correlations is a more effective approach to simplify the vine structure [31]. To this end, a threshold need to be defined in order to set a bivariate copula to independence. In *rvinecopulib* package, truncation level and pruning threshold can both be either selected by modified BIC or specified empirically [28].

2.4.3 Missing data

Missing data is common and can affect copula estimation and statistical inference. Missing data typically occurs in three mechanisms [32]: (1) missing at completely random (MACR): the probability of the observed missing data is independent of any covariate in the dataset; for example, covariates are lost because of random technical issue; (2) missing at random (MAR): the probability of observing the missing data is independent of the corresponding covariate but related to other covariates; for instance, in a dataset of male and female subjects, creatinine is only recorded for females, thus, the creatinine measurements for males are MAR; (3) missing not at random (MNAR): the probability of observing the missing data depends on the values of the corresponding covariate; for example, MNAR occurs when clinical variables are below the lower limit of quantification (BLQ) and recorded as missing.

The rvinecopulib package allows the estimation of vine copulas from incomplete datasets by using complete pairwise observations. However, this can lead to bias, depending on the missingness mechanism. A strategy to address this issue is to impute missing data with plausible values before estimation [33]. Many imputation methods are available, but the choice of imputation method requires careful consideration of the assumption made about the missing data mechanism. Different imputation methods have been previously described in literature [34].

The impact of missing data relies on the modeling purpose. If the aim is to approximate the underlying true joint distribution, expert knowledge on causal relationships in data generation process is crucial [35]. In contrast, if the aim is to generate a VP representative of the observed population, then the priority is to ensure simulated datasets resemble the observed datasets as closely as possible. Under MCAR, the missing data is a random subset of the real-world data, and standard copula estimation and simulation remain unbiased. When data are MAR or MNAR, the observed sample is not representative of the true distribution and can lead to bias in the generated virtual populations. For example, in the case of MAR, the unknown creatinine values for males can be inferred by assuming the same conditional distribution given males as observed in females. If the assumptions can be justified, simulations will not introduce bias from the missingness. If the assumption can not be justified, simulated data outside the observed space should be removed from further simulations or treated with care during application. In the case of MNAR, such as when values below BLQ or above the upper quantification limit are missing, discarding out-of-range data and setting boundaries for marginal distribution during copula estimation could ensure that generated virtual population remains within realistic and observed ranges.

The impact of missing data on copula-based covariate simulation also depends on the amount of missingness. When the proportion of missing data is relatively small, the effect on the estimated dependency structure is often negligible. As the proportion increases, the risk of bias correspondingly increases [36].

2.4.4 Compositional data

Compositional data represents a unique type of data in dependency modeling. A notable example is gut microbiome data, which has been increasing recognized as con-

tributors to variability in PK and clinical outcomes [37]. Microbiome data are typically reported as compositional data, with species abundances expressed as proportions (e.g., 0-100%) that sum to one. A key challenge in copula modeling of compositional data is the risk of generating virtual patients that fall outside bounds or violate the constant-sum constraint. Several approaches can be used to address this issue. If the absolute value is available (e.g., counts), compositional data can be mapped back to its original scale to eliminate the bounds, and copulas can be fitted on this scale, and relative abundances can be computed afterward from the simulated data. A second approach is to fit copulas directly on the compositional data and scale the simulated values by dividing by their sum to ensure the sum-to-one constraint. Alternative approaches include specifying bounds in margin characterization, or applying transformations (e.g., log-ratio transformation) [38] while modeling $d-1$ variables instead of fitting the copula on all (d) variables. However, these latter approaches still carry a risk of generating virtual patients that violate the constant-sum requirement.

2.4.5 Time-varying copulas

Covariates could vary with time, and the dependency structure between covariates may also be time dependent. Time-varying copulas remain an active research area to address the dynamic relationship between variables [39, 40, 41]. Several strategies have been proposed, such as building copulas based on parameters inferred from separate mixed-effects models of the covariates [3], or incorporating time-related parameters within copulas [42]. Although time-varying copulas are frequently used in econometrics, currently, no standard has been established in pharmacometrics yet.

2.5 Conclusion

This tutorial has provided an overview of the estimation, evaluation and application of copulas for covariate simulation in R. The copula is a versatile tool to describe the dependency between covariates. A robust copula could not only help reveal the complex relationships underlying the data but also facilitate the generation of realistic virtual populations for model-informed precision dosing and in silicon clinical trials. Once a valid copula is established, it serves as a vehicle to share sensitive patient data, enabling broader access to patient characteristics.

References

1. Tannenbaum SJ, Holford NH, Lee H, Peck CC, and Mould DR. Simulation of correlated continuous and categorical variables using a single multivariate distribution. Journal of pharmacokinetics and pharmacodynamics. 2006; 33:773–94
2. Teutonico D et al. Generating Virtual Patients by Multivariate and Discrete Re-Sampling Techniques. Pharmaceutical Research. 2015 Oct; 32:3228–37
3. Zwep LB, Guo T, Nagler T, Knibbe CA, Meulman JJ, and Hasselt JGC van. Virtual Patient Simulation Using Copula Modeling. Clinical Pharmacology & Therapeutics. 2024; 115:795–804
4. Smania G and Jonsson EN. Conditional distribution modeling as an alternative method for covariates simulation: Comparison with joint multivariate normal and bootstrap techniques. CPT: Pharmacometrics & Systems Pharmacology. 2021; 10:330–9
5. Hartung N and Khatova A. Information-theoretic evaluation of covariate distributions models. Journal of Pharmacokinetics and Pharmacodynamics. 2025; 52:21
6. Guo Y, Guo T, Knibbe CAJ, Zwep LB, and Hasselt JGC van. Generation of realistic virtual adult populations using a model-based copula approach. Journal of Pharmacokinetics and Pharmacodynamics. 2024 Dec; 51:735–46
7. Genest C and Rivest LP. On the multivariate probability integral transformation. Statistics & Probability Letters. 2001 Jul; 53:391–9
8. Sklar A. Random variables, joint distribution functions, and copulas. Kybernetika. 1973; 9:(449)–460
9. Czado, Claudia. Analyzing Dependent Data with Vine Copulas. 2019; 222
10. Marchione MM and Baione F. Non-monotone dependence modeling with copulas: an application to the volume-return relationship. 2024 Mar
11. Provost SB and Zang Y. Nonparametric Copula Density Estimation Methodologies. Mathematics. 2024 Jan; 12:398
12. Okhrin O, Ristig A, and Xu YF. Copulae in High Dimensions: An Introduction. *Applied Quantitative Finance*. Ed. by Härdle WK, Chen CYH, and Overbeck L. Berlin, Heidelberg: Springer, 2017 :247–77
13. Panagiotelis A, Czado C, and Joe H. Pair Copula Constructions for Multivariate Discrete Data. Journal of the American Statistical Association. 2012 Sep; 107:1063–72
14. Nagler T and Vatter T. rvinecopulib: High Performance Algorithms for Vine Copula Modeling. 2025
15. Nagler T and Vatter T. kde1d: Univariate Kernel Density Estimation. 2025
16. Fidler ML, Hallow M, Wilkins J, and Wang W. RxODE: Facilities for Simulating from ODE-Based Models. 2024
17. Chen YC. A tutorial on kernel density estimation and recent advances. Biostatistics & Epidemiology. 2017 Jan; 1:161–87
18. Dißmann J, Brechmann EC, Czado C, and Kurowicka D. Selecting and estimating regular vine copulae and application to financial returns. Computational Statistics & Data Analysis. 2013 Mar; 59:52–69
19. Okhrin O, Trimborn S, and Waltz M. gofCopula: Goodness-of-Fit Tests for Copulae. R Journal. 2021 Jun; 13:467–98
20. Hartung N. kldest: Sample-Based Estimation of Kullback-Leibler Divergence. 2024
21. Nagler T, Schepsmeier U, Stoeber J, Brechmann EC, Graeler B, and Erhardt T. VineCopula: Statistical Inference of Vine Copulas. 2023
22. Mats O. Karlsson. A Tutorial on Visual Predictive Checks. 2008
23. MacKenzie D and Spears T. 'The formula that killed Wall Street': The Gaussian copula and modelling practices in investment banking. Social Studies of Science. 2014 Jun; 44:393–417
24. Dosne AG, Bergstrand M, Harling K, and Karlsson MO. Improving the estimation of parameter uncertainty distributions in nonlinear mixed effects models using sampling importance resampling. Journal of Pharmacokinetics and Pharmacodynamics. 2016; 43:583–96

25. Geenens G. Copula modeling for discrete random vectors. Dependence Modeling. 2020 Jan; 8:417–40

26. Nagler T. Asymptotic Analysis of the Jittering Kernel Density Estimator. Mathematical Methods of Statistics. 2018 Jan; 27:32–46

27. Faugeras OP. Inference for copula modeling of discrete data: a cautionary tale and some facts. Dependence Modeling. 2017 Jan; 5:121–32

28. Nagler T, Bumann C, and Czado C. Model selection in sparse high-dimensional vine copula models with an application to portfolio risk. Journal of Multivariate Analysis. Dependence Models 2019 Jul; 172:180–92

29. Huard D, Évin G, and Favre AC. Bayesian copula selection. Computational Statistics & Data Analysis. 2006 Nov; 51:809–22

30. Brechmann EC, Czado C, and Aas K. Truncated regular vines in high dimensions with application to financial data. Canadian Journal of Statistics. 2012; 40:68–85

31. Wan J and Li S. Modeling and application of industrial process fault detection based on pruning vine copula. Chemometrics and Intelligent Laboratory Systems. 2019 Jan; 184:1–13

32. Zhang Y, Zhang R, and Zhao B. A systematic review of generative adversarial imputation network in missing data imputation. Neural Computing and Applications. 2023 Sep; 35:19685–705

33. Liebscher E. Fitting copulas in the case of missing data. Statistical Papers. 2024 Aug; 65:3681–711

34. Kang H. The prevention and handling of the missing data. Korean Journal of Anesthesiology. 2013 May; 64:402–6

35. Kertel M and Pauly M. Estimating Gaussian copulas with missing data with and without expert knowledge. Entropy. An International and Interdisciplinary Journal of Entropy and Information Studies. 2022; 24:1849

36. Wang H, Fazayeli F, Chatterjee S, and Banerjee A. Gaussian copula precision estimation with missing values. *Artificial intelligence and statistics*. PMLR, 2014 :978–86

37. Saqr A, Cheng S, Al-Kofahi M, Staley C, and Jacobson PA. Microbiome-Informed Dosing: Exploring Gut Microbial Communities Impact on Mycophenolate Enterohepatic Circulation and Therapeutic Target Achievement. Clinical Pharmacology and Therapeutics. 2025 Dec; 118:1477–88

38. Calderín–Ojeda E, Lopez–Campos G, and Gómez–Déniz E. A Copula Type-Model for Examining the Role of Microbiome as a Potential Tool in Diagnosis. Mathematical Problems in Engineering. 2022; 2022:8033806

39. Liu H, Zhong X, Jiang Z, and Lai S. Dynamic correlation between hog futures and industry chain: an empirical study based on time-varying copula model. International Journal of System Assurance Engineering and Management. 2024 Nov; 15:5356–66

40. Hung NT. Green bonds and asset classes: new evidence from time-varying copula and transfer entropy models. Global Business Review. 2025; 26:1405–24

41. Brechmann EC and Czado C. COPAR—multivariate time series modeling using the copula autoregressive model. Applied Stochastic Models in Business and Industry. 2015 Jul; 31:495–514

42. Manner H and Reznikova O. A Survey on Time-Varying Copulas: Specification, Simulations, and Application. Econometric Reviews. 2012 Nov; 31:654–87

Supplementary material

Supplementary material S1

Derivation of uniform transformation via PIT

The cumulative distribution function (F(x) = P(X <x)) is intrinsically a monotonically increasing function (i.e., if $x_1 < x_2$, $F(x_1) \leq F(x_2)$). This means that the larger x value, the higher the probability of observing a covariate taking on a value less than the x. For simplicity, we assume F is strictly increasing (i.e., if $x_1 < x_2$, $F(x_1) < F(x_2)$), so that it is invertible and its inverse function (F^{-1}) is also monotonic. Under this assumption, the order of values keeps unchanged when applying $F^{-1}(x)$. Thus, for the new variable ($U = F(X)$) obtained via PIT, we can derive $P(U < u) = u$, as follows:

$$\begin{aligned} P(U < u) &= P\big(F(X) < u\big) \\ &= P\big(F^{-1}(F(X)) < F^{-1}(u)\big) \\ &= P(X < x) \\ &= u \end{aligned}$$

The variable U is distributed on the interval of $[0, 1]$, and its cumulative distribution equal to its value, which fulfills the definition of standard uniform distribution. In the settings where F is not strictly increasing (e.g., F being flat on some intervals), F is not invertible and the quantile function should be used as a generalized inverse of F, for which the PIT result still holds.

$$
\begin{aligned}
f(X_1, X_2, X_3, X_4, X_5) = {} & f_1(X_1)\, f_2(X_2)\, f_3(X_3)\, f_4(X_4)\, f_5(X_5) \\
& \times c_{24}\big(F_2(X_2), F_4(X_4)\big) \times c_{34}\big(F_3(X_3), F_4(X_4)\big) \times c_{14}\big(F_1(X_1), F_4(X_4)\big) \times c_{15}\big(F_1(X_1), F_5(X_5)\big) \\
& \times c_{21;4}\big(F_{2|4}(X_2 \mid X_4), F_{1|4}(X_1 \mid X_4); X_4\big) \times c_{13;4}\big(F_{1|4}(X_1 \mid X_4), F_{3|4}(X_3 \mid X_4); X_4\big) \\
& \times c_{45;1}\big(F_{4|1}(X_4 \mid X_1), F_{5|1}(X_5 \mid X_1); X_1\big) \\
& \times c_{23;14}\big(F_{2|14}(X_2 \mid X_1, X_4), F_{3|14}(X_3 \mid X_1, X_4); X_1, X_4\big) \\
& \times c_{35;14}\big(F_{3|14}(X_3 \mid X_1, X_4), F_{5|14}(X_5 \mid X_1, X_3, X_4); X_1, X_4\big) \\
& \times c_{25;134}\big(F_{2|134}(X_2 \mid X_1, X_3, X_4), F_{5|134}(X_5 \mid X_1, X_3, X_4); X_1, X_3, X_4\big).
\end{aligned}
$$

Note: In rvinecopulib and most existing work, R-vine copula estimation is performed under the simplifying assumption, i.e., conditional copula densities are assumed not to depend on the conditioning variables [1].

$$
c_{21;4}\big(F_{2|4}(X_2 \mid X_4), F_{1|4}(X_1 \mid X_4); X_4\big) = c_{21;4}\big(F_{2|4}(X_2 \mid X_4), F_{1|4}(X_1 \mid X_4)\big)
$$

[1] Nagler, Thomas. *Simplified vine copula models: state of science and affairs*. Risk Sciences (2025): 100022.

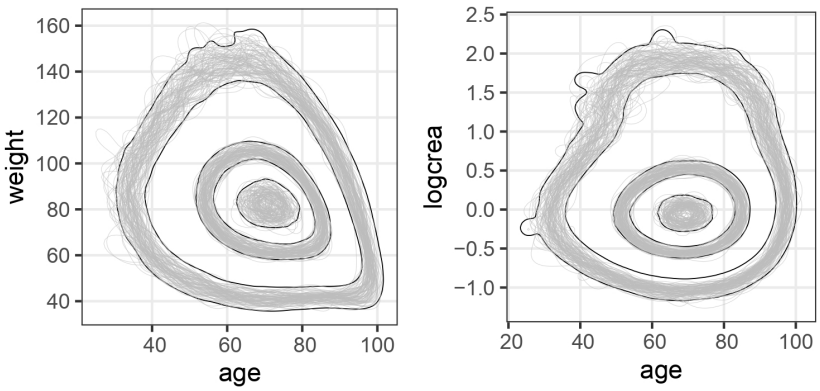
Supplementary material S3

Table S2.1: Key functions used in the tutorial together with the R package and usage.

Function	Package	Usage
`vinecop()`	`rvinecopulib`	Fit vine copula models on uniformly transformed data
`vine()`	`rvinecopulib`	Fit vine copula models on continuous or discrete data
`rvinecop()`	`rvinecopulib`	Simulate a virtual population on uniform scale from the `vinecop` object
`rvine()`	`rvinecopulib`	Simulate a virtual population from the vine object
`rscopula()`	`pmxcopula`	Simulate a subset of a virtual population from the vine object, based on a specified covariate range
`rcopula()`	`pmxcopula`	Simulate multiple virtual populations from the vine object for copula model evaluation
`plot_1dist()`	`pmxcopula`	Visualize and compare univariate density curves of observed and simulated data
`plot_mvdist()`	`pmxcopula`	Visualize and compare dependency structure of observed and simulated data
`calc_margin()`	`pmxcopula`	Calculate marginal metrics from observed and simulated data
`calc_dependency()`	`pmxcopula`	Calculate dependency metrics from observed and simulated data
`plot_qq()`	`pmxcopula`	Plot Q–Q plots comparing observed and simulated data
`donutVPC()`	`pmxcopula`	Plot donutVPCs comparing observed and simulated data

Supplementary material S4

The confidence bands in the donutVPC represent the confidence around the contour points of multiple copula simulations. Each contour has different number of points, so we use weighted KDE to give each simulation the same weight, providing a practical approximation for deriving confidence bands for density contours. The figure below demonstrates how the confidence band relates to the simulated contours.



Contours across 100 simulations in $MIMIC_{sim}$ (gray lines) with the corresponding confidence band at 95% confidence (black outlines). Contours for the 5^{th}, 50^{th}, and 95^{th} percentiles were calculated.

Supplementary material S5

Copula-based PopPK simulation code

```
library(rxode2)
# create a population PK model
mod <- function() {
ini({
    p_CL  = 10; # typical value of clearance
    p_V  = 10; # typical value of volume of distribution
    eta_CL ~ 0.3^2;
    eta_V ~ 0.3^2;
    conc.sd = 0.1;
  })
model({
    CL = p_CL *(weight/60)^0.75* (creatinine/125)^0.3 *
        exp(eta_CL);  # individual clearance
    V = p_V * (weight /60) * exp(eta_V);  # individual volume of
        distribution
    C = centr/V ;
    d/dt(centr)   = -CL*C;
    C ~ prop(conc.sd)
  })
}

# define the dosing regimen and sampling time in the event table
ev <- eventTable() %>%
        add.dosing(dose = 500,
                   dosing.interval = 24,
                   rate = 200) %>%
        add.sampling(seq(0, 24, by = 0.1)) %>%
        et(id = 1:100) # ID identifier for 1000 individuals

# format virtual population data to ensure compatibility with the
    model and event table
MIMIC_VP <- MIMIC_VP %>%
            mutate(id = 1:100, creatinine = exp(logcrea))

# simulate the PK profiles with virtual populations
PK_sim <- rxSolve(mod, ev, iCov = MIMIC_VP)
```

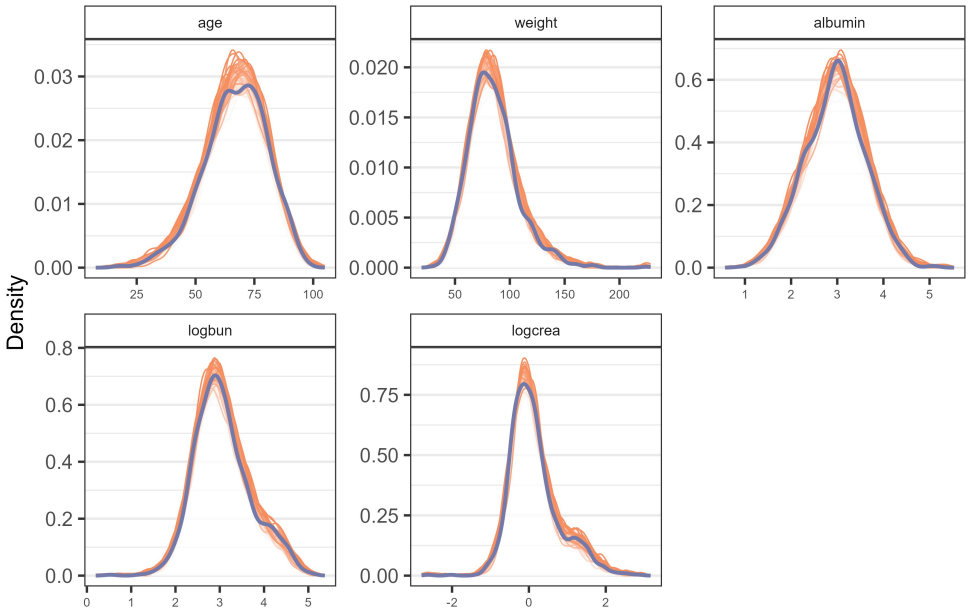


Figure S2.1: Visual inspection of the performance of copula on capturing marginal distributions. The lines represent the univariate density curves of each covariate from multiple simulated (orange) and observed (purple) populations.

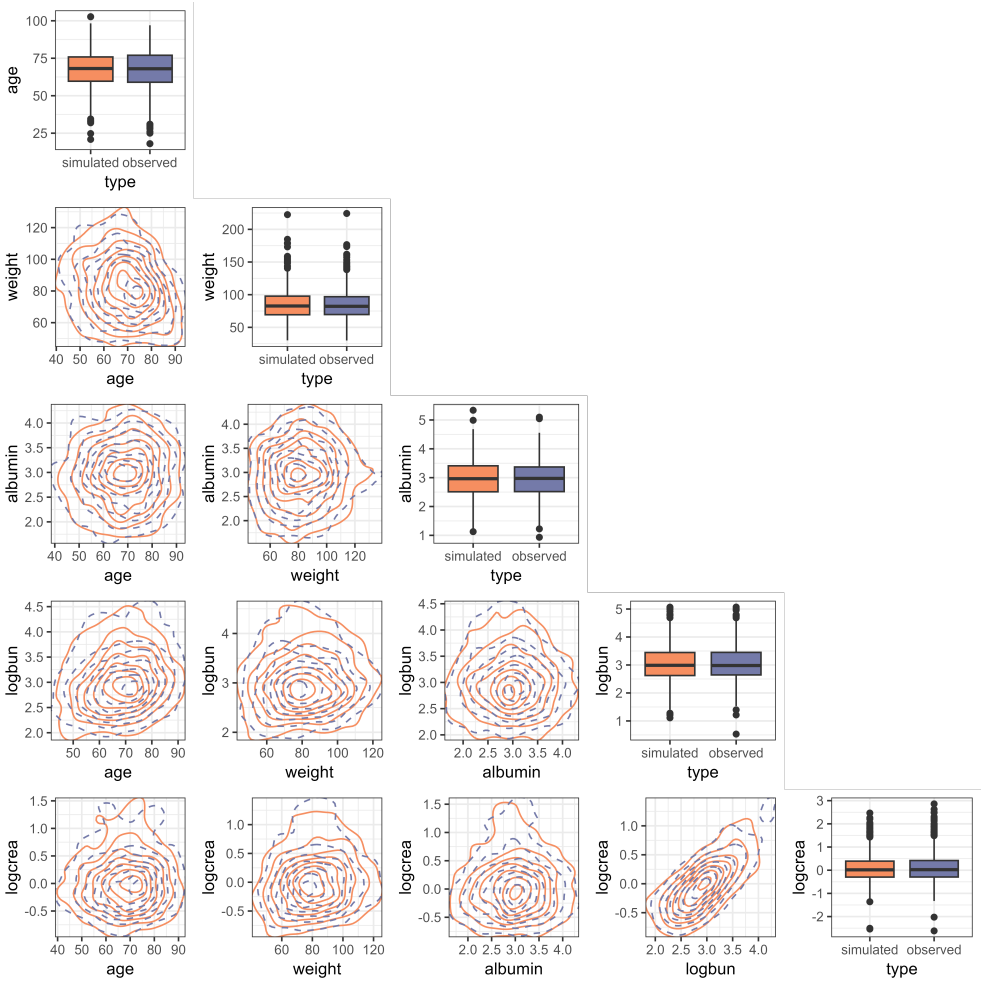


Figure S2.2: Visual inspection of the performance of copula on capturing the joint distribution. The lines represent the bivariate density contours of each covariate from simulated (solid orange) and observed (dashed purple) populations. A density contour is a curve that connects the points of the same probability density.

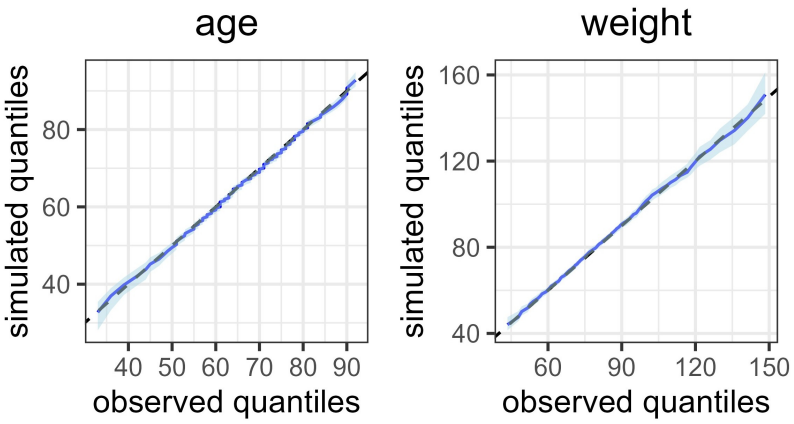


Figure S2.3: A Q-Q plot as visual predictive check for marginal distribution. The blue solid line and shaded area represent the median and the 95% prediction interval for each quantile (from 1st to 99th) of a covariate, respectively.