Universiteit Leiden

Leiden University

End-term project: Advanced Statistical Computing 2020

Copula Model and Statistical Analysis in Insurance Policy Management

Authors:

Huilin Li s2556057

February 13, 2023

1 Introduction

This project concerns applying copula model and some statistical analysis to solve one common issue that how an insurance company chooses reinsurance policy. That is an insurance company asks other insurance companies, named re-insurance company, for help, when meeting risks. In our case, our goal is to help the ANV company to choose the reinsurance policy.

There are two business lines, respectively named Professional liability insurance(PLI)[1] and Workers' compensation (WC)[2], in the ANV company, which were simultaneously affected due to a huge claim. For some different threshold $t = 100, 110, 120, \cdots, 200$, the reinsurance company requests the price $P(t) = 40000 \times e^{(-t/7)}$ (in million euros) for the reinsurance policy. However, whether the ANV company will accept this reinsurance suggestion is according to the expected value $V(t) = E[(PLI + WC)1(PLI + WC > t)]$. Therefore, this project will focus on approximating statistically the $V(t)$.

The Section 2 will illustrate how we choose statistical models and design simulation experiments. Then, the Section 3 will explain the detailed processes of the simulation study, as well as display the experiment results. After analysing the results, there is a conclusion discussion in the Section 4.

2 Methodology

We collected the actual PLI and WC values from 648 users, moreover, we defined the $X1 = PLI$ and $X2 = WC$. The Figure 1 shows the relationship between the PLI and the WC, as well as their respective density distributions. It is clear to find that both $X1$ and $X2$ follow a typical *lognormal* respectively. Therefore, we assumed the density distributions of $X1$ and $X2$ are described in the Equation 1.

$$\begin{aligned} &f_{X_j}(\cdot; \mu_j, \sigma_j) \sim \text{Lognormal}(\mu_j, \sigma_j), \mu_j \in \mathbb{R}, \sigma_j > 0 \\ &f_X(x) = \frac{1}{x\sigma\sqrt{2\pi}} \exp\left(-\frac{(\ln x - \mu)^2}{2\sigma^2}\right) \end{aligned} \tag{1}$$

where $j = 1, 2$ means f_{X_1} and f_{X_2} have their own density distribution. $f_X(x)$ is the density function of the *lognormal* distribution.

Moreover, we noticed that $X1$ and $X2$ have a clear dependency relationship. Therefore, we introduced the Copula model[3], which is widely used in exploring the joint density which focuses on the dependency relationship. The Copula model assumes the joint density f_{X_1,X_2}

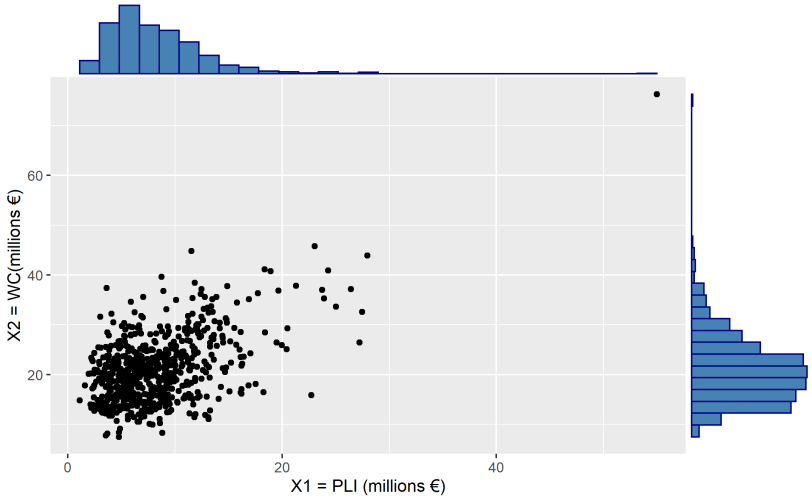


Figure 1: Actual PLI and WC values, with their respective density distributions

is as the Equation 2.

$$f_{X_1,X_2}(x_1, x_2) = f_{X_1}(x_1) f_{X_2}(x_2) c(F_{X_1}(x_1), F_{X_2}(x_2)) \tag{2}$$

where the $f_{X_1}(x)$ and f_{X_2} are the density distribution of X_1 and X_2 respectively, as shown in the Equation 1. The $F_{X_1}(x_1)$ and $F_{X_2}(x_2)$ are the cumulative distribution function(CDF) of X_1 and X_2 respectively.

Here, $c(u_1, u_2)$ should be noted. That is defined as the copula density, where u_1 is the *cdf* of X_1 and u_2 is the *cdf* of X_2. In our case study, we assumed that the copula model is the *JoeCopula* model. The Equation 3 gives its parametric model expression.

$$c(\cdot; \theta) \sim \text{Joe}(\theta), \theta \geqslant 1 \tag{3}$$

The Figure 2 shows how we organized the simulation study. Firstly, Maximum likelihood estimation(MLE) is applied to estimate the parameters set$(\mu_1, \sigma_1, \mu_2, \sigma_2, \theta)$, which has been introduced in the Equation 1 and the Equation 3, and the Section 3.1 will explain more details. Then, we want to know if our estimations are correct, so there is an evaluation experiment in the Section 3.2 to check and explore our estimated joint model. Finally, the goal of this project to estimate the $V(t)$ will be achieved in the Section 3.3, by using Monte Carlo simulation(MC), Importance Sampling(IS) and Bootstrap Method.

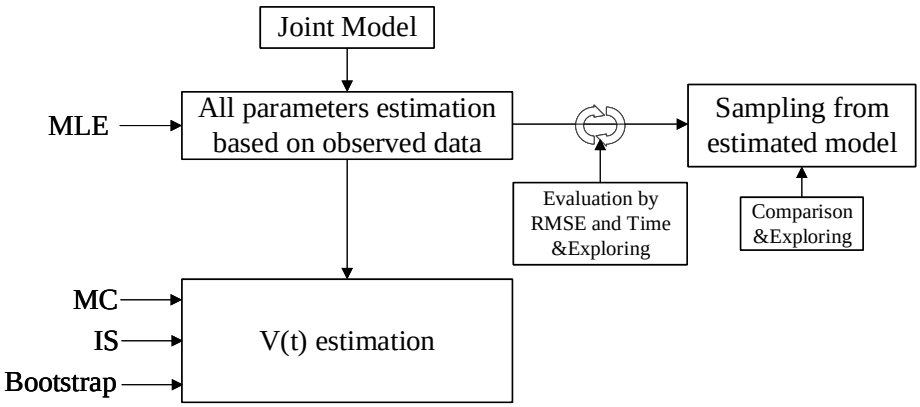


Figure 2: Steps and flows through the experiment study process

3 Simulation study

3.1 Maximum likelihood estimation

Maximum likelihood estimation(MLE) is a popular way to measure specific values of parameters in a model[4]. These parameters values are estimated by the collected data, and these parameters lead the maximum likelihood that the sampled 'fake' dataset produced by the model with these estimated parameters is similar with the actual observed dataset.

In our cases, we aim to figure out the parameters of the density distribution function, based on a series of observed data. At the same time, the joint probability density points out that when observations(x_1,x_2,x_3,$\cdots$,x_n) are independent with each other, the $P(x_1, x_2, x_3, \cdots, x_n; \alpha) = P(x_1; \alpha)P(x_2; \alpha) \cdots P(x_n; \alpha)$, where the α is the parameters set of the density function. Therefore, the goal of the MLE is to figure out the α which leads the maximum result of the above expression. Mathematically, the differentiation method is a common method helping to find the maxima or minima of the expression[4]. However, it seems impossible or troublesome to differentiate such an expression. Here, MLE introduces the taking the natural logarithm of the expression to simplify the process[4]. This is because the natural logarithm is a monotonically increasing function[5], which doesn't influence the maxima or minima of the original expression. Finally, our goal is changed from figuring out the maximum of $\prod_{i=1}^{n} P(x_i; \alpha)$ to figuring out the maximum fo $\sum_{i=1}^{n} ln(P(x_i; \alpha))$.

Here, we will use the help of the *optim*()[6] and the *nlm*()[7] in the R software[8], to fit the *lognormal* distribution and the *JoeCopula* distribution via the MLE.

- Fitting *Lognormal* Distribution

First of all, we created the log_likelihood function of the original function. Although the density function of the *Lognormal* is visible, as shown in the Equation 1, the $dlnorm(x, \mu, \sigma)$[9] provides a convenient way to achieve it in the R coding. It is worth noting that the $optim()$ automatically computes the minimum of the input function, but $minf(x) = max(-f(x))$, therefore, our $lognormal\_lik()$ function returns $-y$. More details have been shown in the Code 1.

```
lognormal_lik <- function(pars, x){
  y <- sum(log(dlnorm(x, pars[1], pars[2])))
  return(-y)
}
lognormal_par1 = optim(par=c(0.5, 0.5 ), lognormal_lik, x=x1)
lognormal_par2 = optim(par=c(0.5,0.5 ), lognormal_lik, x=x2)
```

Code 1: Fitting *Lognormal* distribution via MLE

Moreover, we set up all initial parameters$(\mu_1, \sigma_1, \mu_2, \sigma_2)$ as 0.5, and $lognormal\_par1$ is the optimization results of the Lognormal$(X_1; \mu_1, \sigma_1)$, same as that the $lognormal\_par2$ is for the Lognormal$(X_2; \mu_2, \sigma_2)$. The Table 1 displays the results of the fitting *Lognormal* distribution parameters.

Table 1: Fitting *Lognormal* distribution results

$X_j, (j = 1, 2)$	μ	σ	Iterations
X_1	1.9820500	0.5133805	79
X_2	2.9969006	0.3107596	97

- Fitting *JoeCopula* Distribution

  Here, we also firstly created the log_likelihood function of the original density function. However, the density function of the *JoeCopula* is not visible, so we used the help of the $dCopula(u, joeCopula(theta))$[10]. Also, the $nlm()$ automatically computes the minimum of the input function, therefore, our $joe\_lik()$ function returns $-y$. Moreover, as we discussed in the Section 2, (u_1, u_2) are respectively the *cdf* of X_1 and the *cdf* of X_2, which could be achieved by the help of $plnorm(X, \mu, \sigma)$. More details have been shown in the Code 2.

```
joe_lik <- function(theta,u){
  y <- sum(log(dCopula(u, joeCopula(theta))))
```

```
  return(-y)
}
u1 <- plnorm(x1, mu1_hat, sigma1_hat )
u2 <- plnorm(x2, mu2_hat, sigma2_hat )
u12 = matrix(c(u1, u2),ncol = 2)
joe_par = nlm(joe_lik,c(1),u = u12)
```

Code 2: Fitting *JoeCopula* distribution via MLE

Moreover, we set up the initial parameters(θ) as 1, and u_1 is the $F_{\hat{\mu}_1,\hat{\sigma}_1}(X_1)$, as well as the u_2 is the $F_{\hat{\mu}_2,\hat{\sigma}_2}(X_2)$. The *joe_par* is the optimization results of the JoeCopula(u; θ). The Table 2 displays the results of the fitting *JoeCopula* distribution parameters.

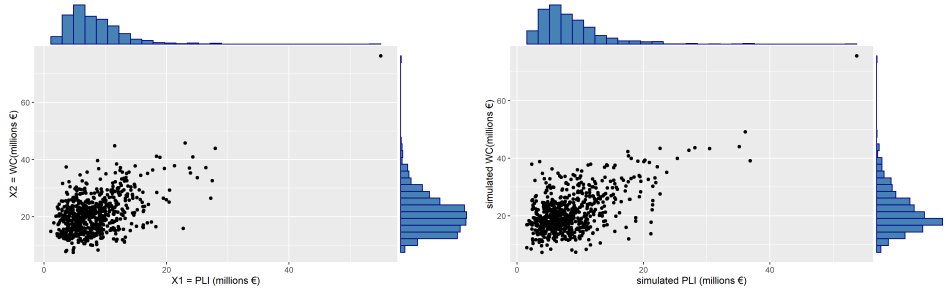
Table 2: Fitting *JoeCopula* distribution results

$u_j, (j = 1, 2)$	θ	Iterations
u	1.607826	7

3.2 Evaluation experiment

In this part, we check if our estimated joint model makes senses. Firstly, we applied $fitdistr(x, 'lognormal', method = 'mle')$[11] and $BiCopEst(U1, U2, family = 6, method = 'mle')$[12], which are R inner packages for *lognormal* and *JoeCopula* parameters estimation using MLE. Here, we collected that $\mu_1 = 1.9821374, \sigma_1 = 0.5134643, \mu_2 = 2.9968907, \sigma_2 = 0.3108413, \theta = 1.6081638$ which are very similar with the values calculated in the Section 3.1.

For further explanation, we designed a small experiment to check the parameters estimation values. In our $SAMP()$ function, we firstly used the $rCopula(n, joeCopula(theta))$[10] to sample $n \times 2$ pseudo-sampling data, next, each column of the sampling was transformed to the real sampling data, by using the $qlnorm()$[9]. Here, in order to compare the sampled data to the actual observed data, we set up the n as 648, the same shape of the observed data. The Figure 3 explains that the implementation is correct, because based on the same shape of data size, they look similar.



(a) Actual observed data (b) Simulated data

Figure 3: Compare the simulated data with the actual observed data

Moreover, we explored how the parameters influence the sampling data. The Figure 4 shows, the μ has slight influence in the sampling data, but higher σ_1 and σ_2 influence the sampling range of the X_1 and the X_2 respectively limit to a very narrow range which closes to zero. As well as, the higher θ makes a much more clear liner relationship between X_1 and X_2.

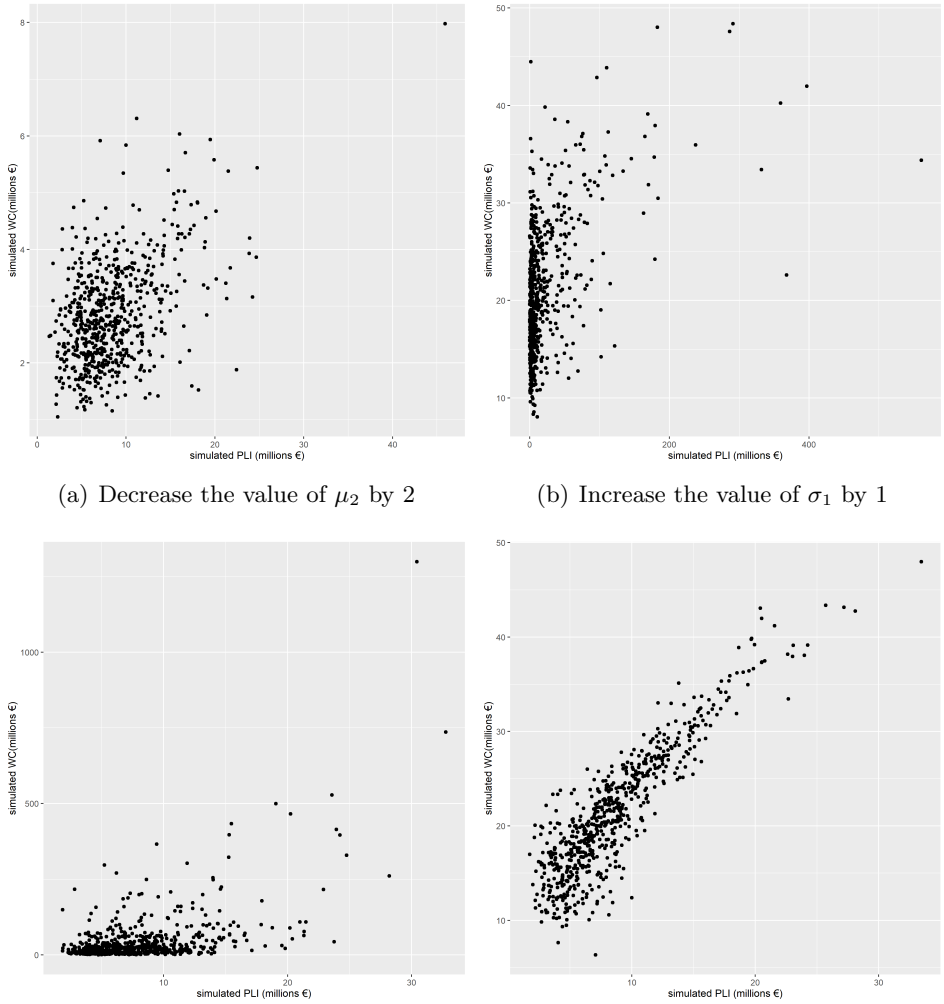
Next, we increased the number of samples to 100 rounds, and simultaneously fixed the $\mu_1 = 1, \sigma_1 = 2, \mu_2 = 3, \sigma_2 = 0.5, \theta = 2$. The Figure 5 shows that when the number of samples increases, the RMSE of each parameter will decrease. Also, we noticed that the parameter θ of the JoeCopula model is harder to estimate, which might is because the parameter θ has a narrower limitation, such as $\theta \geqslant 1$. Moreover, we calculated the running time of each sample, the Table 3 explained that when the size of samples becomes bigger, it has cost much time to finish sampling and estimating.

Table 3: Running time of samples based on different sample size

Sample Size n	$n = 200$	$n = 500$	$n = 1000$
Running Time (in Seconds)	0.5	0.92	1.63

3.3 $V(t)$ estimation

In this part, first of all, Monte Carlo simulation and Importance sampling are used to estimate the $V(t)$. The Figure 6 shows that as t increases, both $P(t)$ and $V(t)$ are decreasing, but the $V(t)$ sampled by the Importance sampling is much closer to the $P(t)$. Moreover, in the Figure 6(b), we could conclude that when $t = 110, 120, 130.140$, the ANV company will pay for the reinsurance policy.



(a) Decrease the value of μ_2 by 2

(b) Increase the value of σ_1 by 1

(c) Increase the value of σ_2 by 1

(d) Increase the value of θ by 1

Figure 4: Exploring the changes of sampled data, when increase/decrease parameters

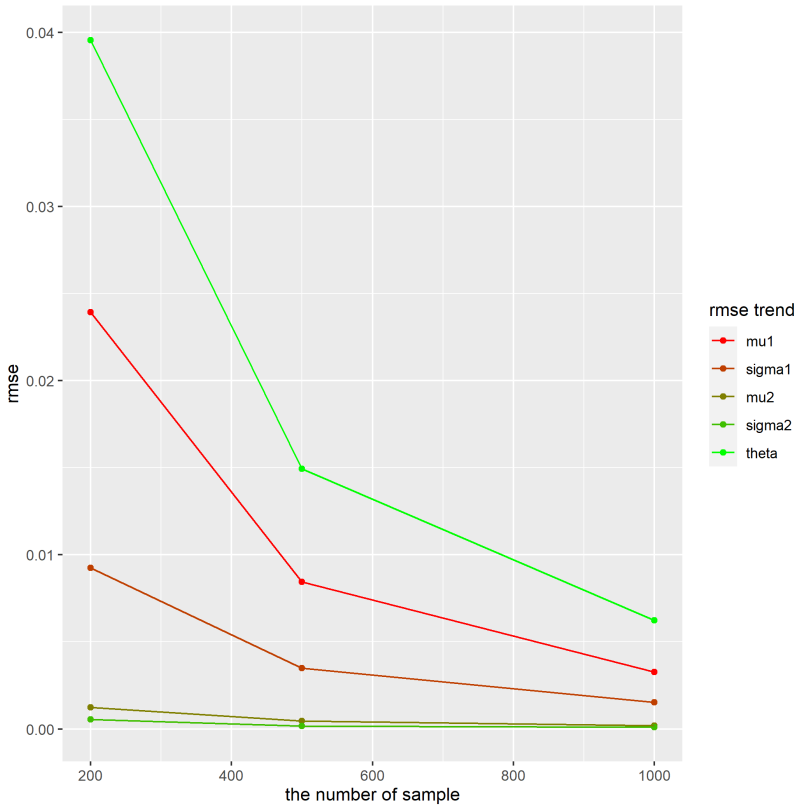
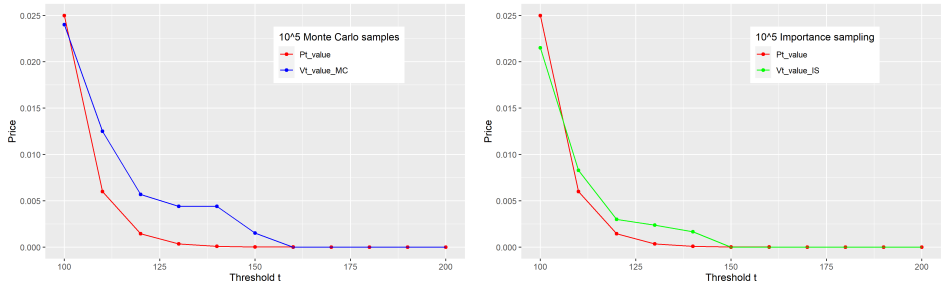


Figure 5: RMSE changes based on different number of sample



(a) Monte Carlo simulation (b) Importance sampling

Figure 6: Compare the $V(t)$ with $P(t)$, using two sampling methods

Finally, we come to the final step - Bootstrap sampling. The method samples data from the observed original data by replacement. In this case, we are curious about the sum of two means from two samples $(x_1^{'}, x_2^{'})$, where the $x_1^{'}$ is the samples related to the X_1, and the $x_2^{'}$ is the samples related to the X_2. This is because our final goal $V(t)$ is an expression related to the sum of c. First, we created two matrices filled with 10^5 samples from the observed original data with replacement. And the shape of these two matrices is $(648, 10^5)$, where 648 is the number of the actual observed data. After calculating each column mean of each matrix,

Boot.sum is our final object which computes the sum of each $x_1^{'}, x_2^{'}$ pairs. Here, we studied the range of $(X_1 + X_2)$ based on confidence 80% by using the 'PERCENTILE' bootstrap method, such as $quantile(Boot.sum, prob = 0.15)$, and $quantile(Boot.sum, prob = 0.95)$.

Table 4: Percentile Bootstrap with confidence 80%

Object	15%	95%
$(X_1 + X_2)$	28.96363	29.8324

The Table 4 shows that we are 80% confident that the sum of $(X_1 + X_2)$ is somewhere between 28.96363 in millions euros and 29.8324 in millions euros. This is to say, mostly the $V(t)$ is below the $min(t) = 100$. Next, we discussed the estimate error and the MC approximation error. We repeated the simple sampling 1000 times, based on the estimated model,and sampled 648 data with the same size as the observed data, then rmse between the estimated $V(t)$ and the actual $V(t)$. We found the rmse equals zero, which could say that the Bootstrap can account for the estimated error.

Also, we repeated the MC sampling 1000 times, and calculated the rmse which is around 0.009041239, which is also very small. But compared to the previous simple sampling, it is hard to accept that the Bootstrap can account for the MC approximation error.

Finally, we come up with that no matter which the threshold is, in most cases, the ANV company has to pays the claim themselves. Therefore, the ANV company doesn't need to pay for the reinsurance policy.

4 Analysis and Conclusion

Our research demonstrated that simulation study based on one *JoeCopula* joint model, and gave detailed statistical analysis to help the ANV company making decision related to if it is necessary to pay for the reinsurance policy. Finally, we conclude that the ANV doesn't need to pay for the reinsurance policy. Moreover, by observing the actual observed data, there is only one x_1, x_2 pairs whose sum is higher than the $min(t) = 100$, which further explained the possibility of $PLI + WC > t$ is very small, and the ANV company doesn't need to request a reinsurance policy.

References

[1] Wikipedia contributors. Professional liability insurance — Wikipedia, the free encyclopedia, 2020. [Online; accessed 24-October-2020].

[2] Wikipedia contributors. Workers' compensation — Wikipedia, the free encyclopedia, 2020. [Online; accessed 24-October-2020].

[3] Wikipedia contributors. Copula (probability theory) — Wikipedia, the free encyclopedia, 2020. [Online; accessed 24-October-2020].

[4] Jonny Brooks-Bartlett. Probability concepts explained: Maximum likelihood estimation, 2018.

[5] Wikipedia contributors. Monotonic function — Wikipedia, the free encyclopedia, 2020. [Online; accessed 25-October-2020].

[6] R-core R-core@R-project.org. optim: General-purpose optimization, 2020.

[7] Non-Linear Minimization. Non-linear minimization, 2020.

[8] R Core Team. *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, Vienna, Austria, 2020.

[9] Package stats version 4.1.0. *The Log Normal Distribution*, 2020.

[10] Martin Maechler. *Copula Distribution Functions*, 2020.

[11] W. N. Venables and B. D. Ripley. *Modern Applied Statistics with S*. Springer, New York, fourth edition, 2002. ISBN 0-387-95457-0.

[12] Thomas Nagler, Ulf Schepsmeier, Jakob Stoeber, Eike Christian Brechmann, Benedikt Graeler, and Tobias Erhardt. *VineCopula: Statistical Inference of Vine Copulas*, 2020. R package version 2.4.0.