Projet QRM

```
library(copula)
library(VC2copula)
library(ggplot2)
library(latticeExtra)
```

```
## Loading required package: lattice
```

```
##
## Attaching package: 'latticeExtra'
```

```
## The following object is masked from 'package:ggplot2':
##
##     layer
```

```
library(VineCopula)
```

```
##
## Attaching package: 'VineCopula'
```

```
## The following objects are masked from 'package:VC2copula':
##
##     BB1Copula, BB6Copula, BB7Copula, BB8Copula, copulaFromFamilyIndex,
##     joeBiCopula, r270BB1Copula, r270BB6Copula, r270BB7Copula,
##     r270BB8Copula, r270ClaytonCopula, r270GumbelCopula,
##     r270JoeBiCopula, r270TawnT1Copula, r270TawnT2Copula, r90BB1Copula,
##     r90BB6Copula, r90BB7Copula, r90BB8Copula, r90ClaytonCopula,
##     r90GumbelCopula, r90JoeBiCopula, r90TawnT1Copula, r90TawnT2Copula,
##     surBB1Copula, surBB6Copula, surBB7Copula, surBB8Copula,
##     surClaytonCopula, surGumbelCopula, surJoeBiCopula, surTawnT1Copula,
##     surTawnT2Copula, tawnT1Copula, tawnT2Copula, vineCopula
```

```
## The following object is masked from 'package:copula':
##
##     pobs
```

```
set.seed(13112025)
```

load of data and cleaning

```r
#data <- read.csv("wearable_tech_sleep_quality_1.csv")
data <- read.csv("EnglandWeather.csv")
data$Date <- as.POSIXct(
  data$Formatted.Date,
  format = "%Y-%m-%d %H:%M:%OS %z",
  tz = "UTC"
)

data$Month <- format(data$Date, "%m")
data$Hour  <- format(data$Date, "%H")
data$Temp_anom <- data$Temperature..C. -
  ave(data$Temperature..C., data$Month,
      FUN = mean, na.rm = TRUE)
data$Hum_anom <- data$Humidity -
  ave(data$Humidity, data$Month,
      FUN = mean, na.rm = TRUE)
cor(data$Temperature..C., data$Humidity, use = "complete.obs")
```

```
## [1] -0.6322547
```

```r
cor(data$Temp_anom, data$Hum_anom, use = "complete.obs")
```

```
## [1] -0.6340224
```

```r
data_cl <- data[, c("Temperature..C.","Wind.Speed..km.h.","Pressure..millibars.","Humidity")]
data_cl <- data_cl[data_cl$Pressure..millibars. != 0, ] # on retire les lignes ou pression = 0 valeurs
data_cl <- data_cl[data_cl$Humidity != 0, ] #pareil pour l'humidite
data_cl$Humidity <- data_cl$Humidity + runif(nrow(data_cl), -0.0005, 0.0005)
str(data_cl)
```

```
## 'data.frame':    95143 obs. of  4 variables:
##  $ Temperature..C.     : num  9.47 9.36 9.38 8.29 8.76 ...
##  $ Wind.Speed..km.h.   : num  14.12 14.26 3.93 14.1 11.04 ...
##  $ Pressure..millibars.: num  1015 1016 1016 1016 1017 ...
##  $ Humidity            : num  0.89 0.86 0.89 0.83 0.83 ...
```

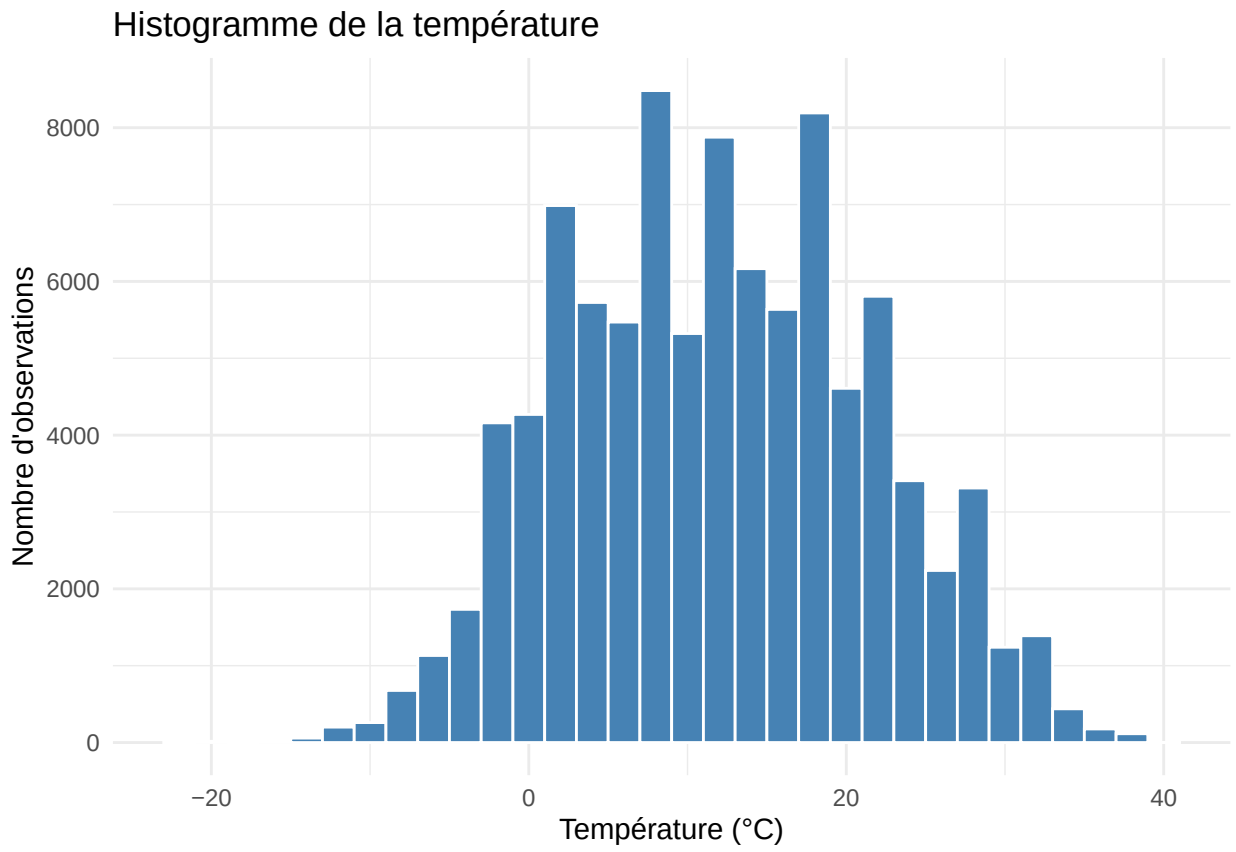
```r
summary(data_cl)
```

```
##  Temperature..C.   Wind.Speed..km.h. Pressure..millibars.    Humidity
##  Min.   :-21.822   Min.   : 0.000    Min.   : 973.8       Min.   :0.1198
##  1st Qu.:  4.744   1st Qu.: 5.812    1st Qu.:1012.1       1st Qu.:0.6004
##  Median : 12.039   Median : 9.918    Median :1016.5       Median :0.7804
##  Mean   : 11.955   Mean   :10.785    Mean   :1016.8       Mean   :0.7351
##  3rd Qu.: 18.839   3rd Qu.:14.120    3rd Qu.:1021.2       3rd Qu.:0.8905
##  Max.   : 39.906   Max.   :63.853    Max.   :1046.4       Max.   :1.0005
```

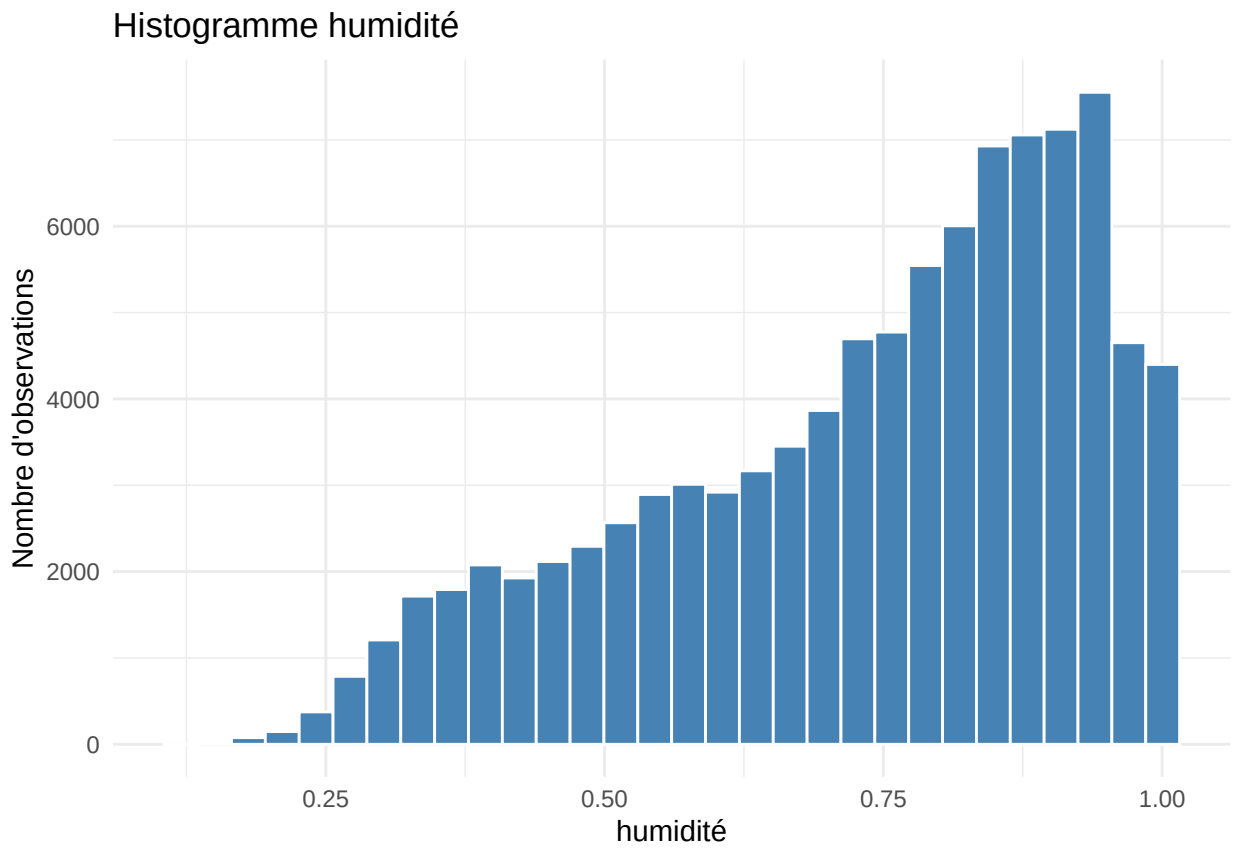
```r
colSums(is.na(data_cl))
```

```
##      Temperature..C.    Wind.Speed..km.h. Pressure..millibars.
##                    0                    0                    0
##             Humidity
##                    0
```

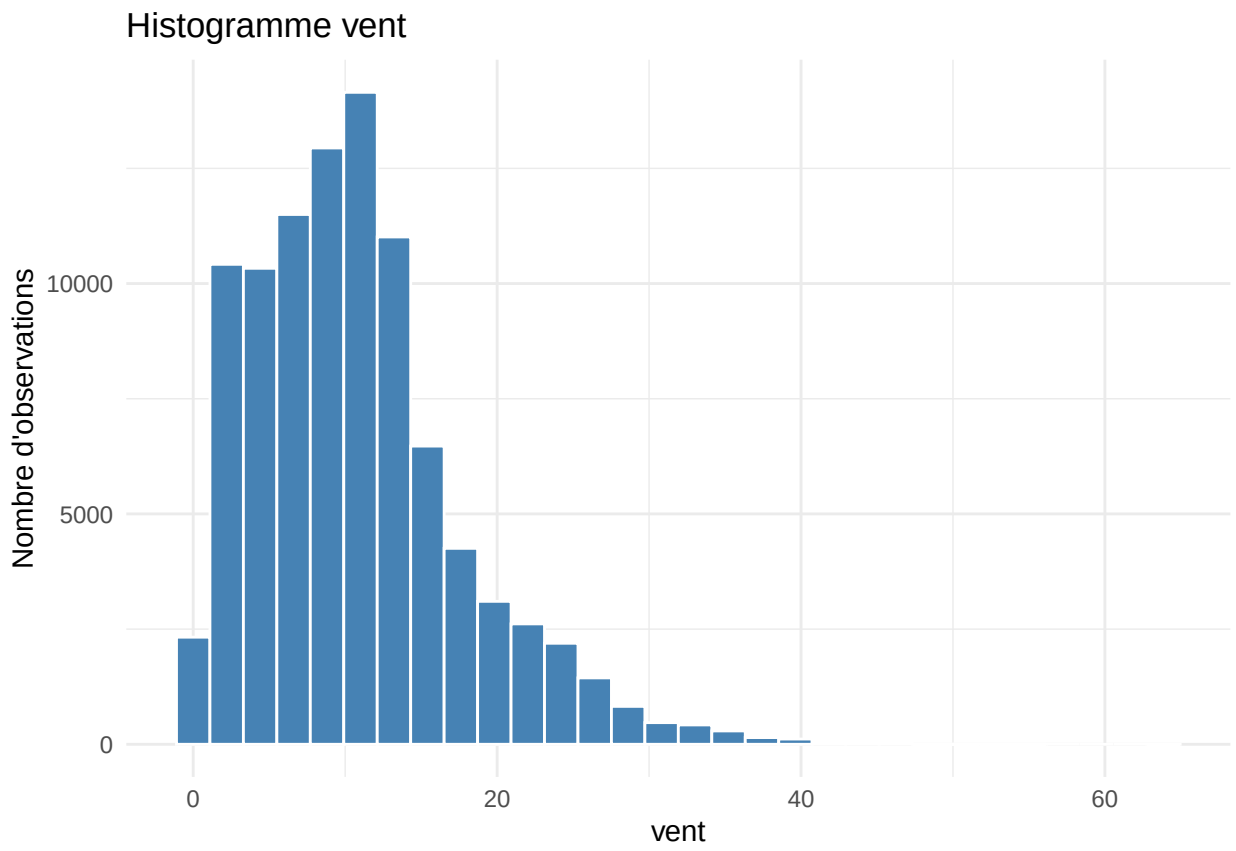
```r
ggplot(data_cl, aes(x = Temperature..C.)) +
  geom_histogram(fill = "steelblue", color = "white", binwidth = 2) +
  labs(
    title = "Histogramme de la température",
    x = "Température (°C)",
    y = "Nombre d'observations"
  ) +
  theme_minimal()
```



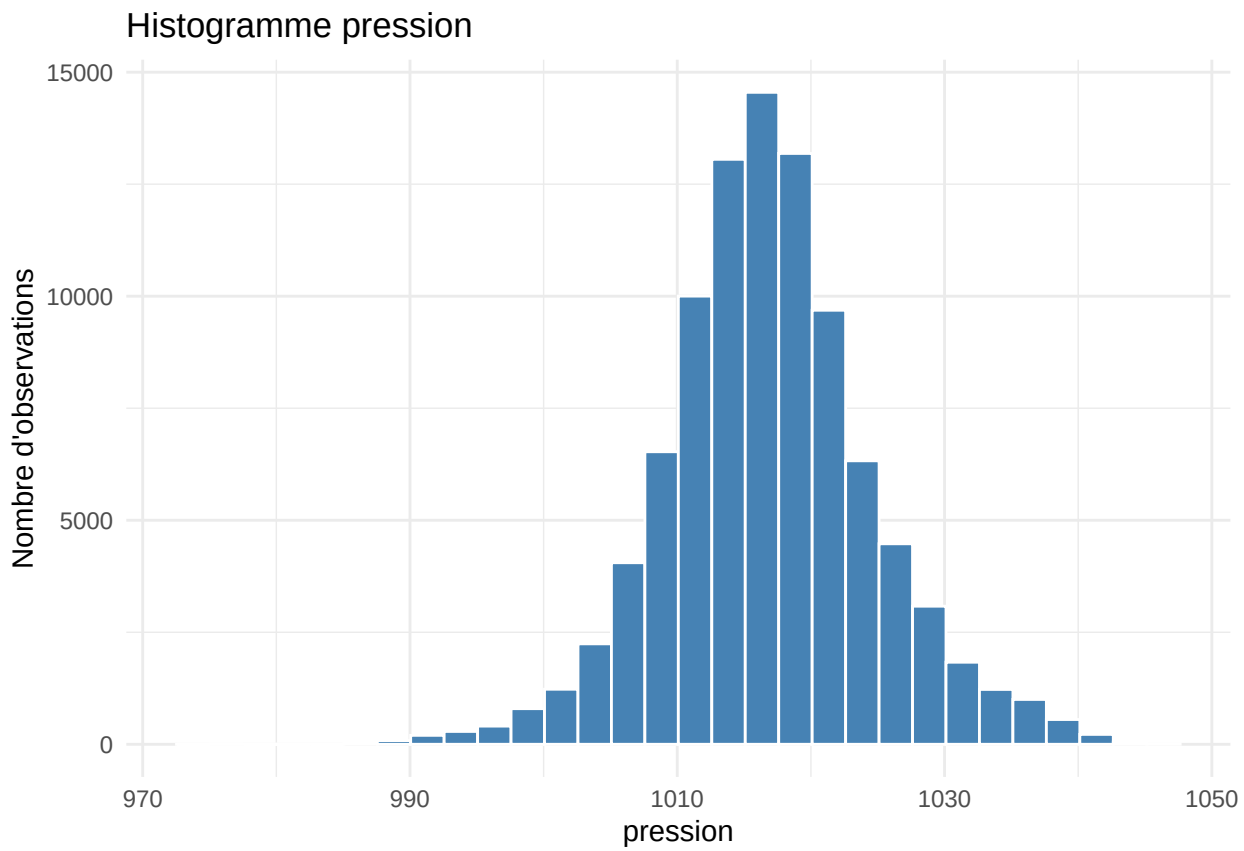

```r
ggplot(data_cl, aes(x = Humidity)) +
  geom_histogram(fill = "steelblue", color = "white", bins=30) +
  labs(
    title = "Histogramme humidité",
    x = "humidité",
    y = "Nombre d'observations"
  ) +
  theme_minimal()
```




```
ggplot(data_cl, aes(x = Wind.Speed..km.h.)) +
  geom_histogram(fill = "steelblue", color = "white", bins=30) +
  labs(
    title = "Histogramme vent",
    x = "vent",
    y = "Nombre d'observations"
  ) +
  theme_minimal()
```




```
ggplot(data_cl, aes(x = Pressure..millibars.)) +
  geom_histogram(fill = "steelblue", color = "white", bins=30) +
  labs(
    title = "Histogramme pression",
    x = "pression",
    y = "Nombre d'observations"
  ) +
  theme_minimal()
```



Préparation des données et pseudo-observations

Données brutes

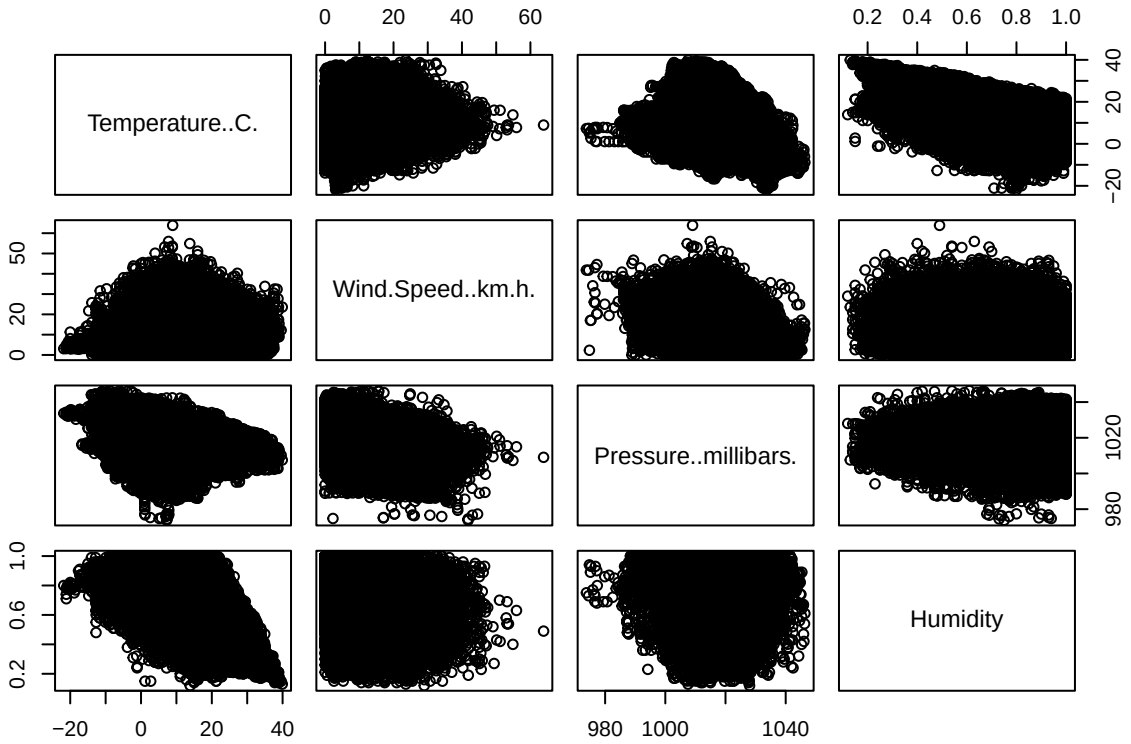
La première étape consiste toujours à regarder les données puis à tracer le nauge de points de données brutes.

```
head(data)
```

```
##                  Formatted.Date       Summary Precip.Type Temperature..C.
## 1 2006-04-01 00:00:00.000 +0200 Partly Cloudy        rain        9.472222
## 2 2006-04-01 01:00:00.000 +0200 Partly Cloudy        rain        9.355556
## 3 2006-04-01 02:00:00.000 +0200 Mostly Cloudy        rain        9.377778
## 4 2006-04-01 03:00:00.000 +0200 Partly Cloudy        rain        8.288889
## 5 2006-04-01 04:00:00.000 +0200 Mostly Cloudy        rain        8.755556
## 6 2006-04-01 05:00:00.000 +0200 Partly Cloudy        rain        9.222222
##   Wind.Speed..km.h. Pressure..millibars. Humidity                Date Month
## 1           14.1197              1015.13     0.89 2006-03-31 22:00:00    03
## 2           14.2646              1015.63     0.86 2006-03-31 23:00:00    03
## 3            3.9284              1015.94     0.89 2006-04-01 00:00:00    04
## 4           14.1036              1016.41     0.83 2006-04-01 01:00:00    04
## 5           11.0446              1016.51     0.83 2006-04-01 02:00:00    04
## 6           13.9587              1016.66     0.85 2006-04-01 03:00:00    04
##   Hour Temp_anom  Hum_anom
## 1   22  2.557246 0.1871505
```

```
## 2   23  2.440579 0.1571505
## 3   00 -3.391423 0.2489495
## 4   01 -4.480311 0.1889495
## 5   02 -4.013645 0.1889495
## 6   03 -3.546978 0.2089495
```

```
pairs(data_cl)
```




```
cor(data_cl, method = "pearson")
```

```
##                     Temperature..C. Wind.Speed..km.h. Pressure..millibars.
## Temperature..C.          1.00000000        0.01004164          -0.30988883
## Wind.Speed..km.h.        0.01004164        1.00000000          -0.25364912
## Pressure..millibars.    -0.30988883       -0.25364912           1.00000000
## Humidity                -0.63667418       -0.22482980           0.04062668
##                        Humidity
## Temperature..C.      -0.63667418
## Wind.Speed..km.h.    -0.22482980
## Pressure..millibars.  0.04062668
## Humidity              1.00000000
```

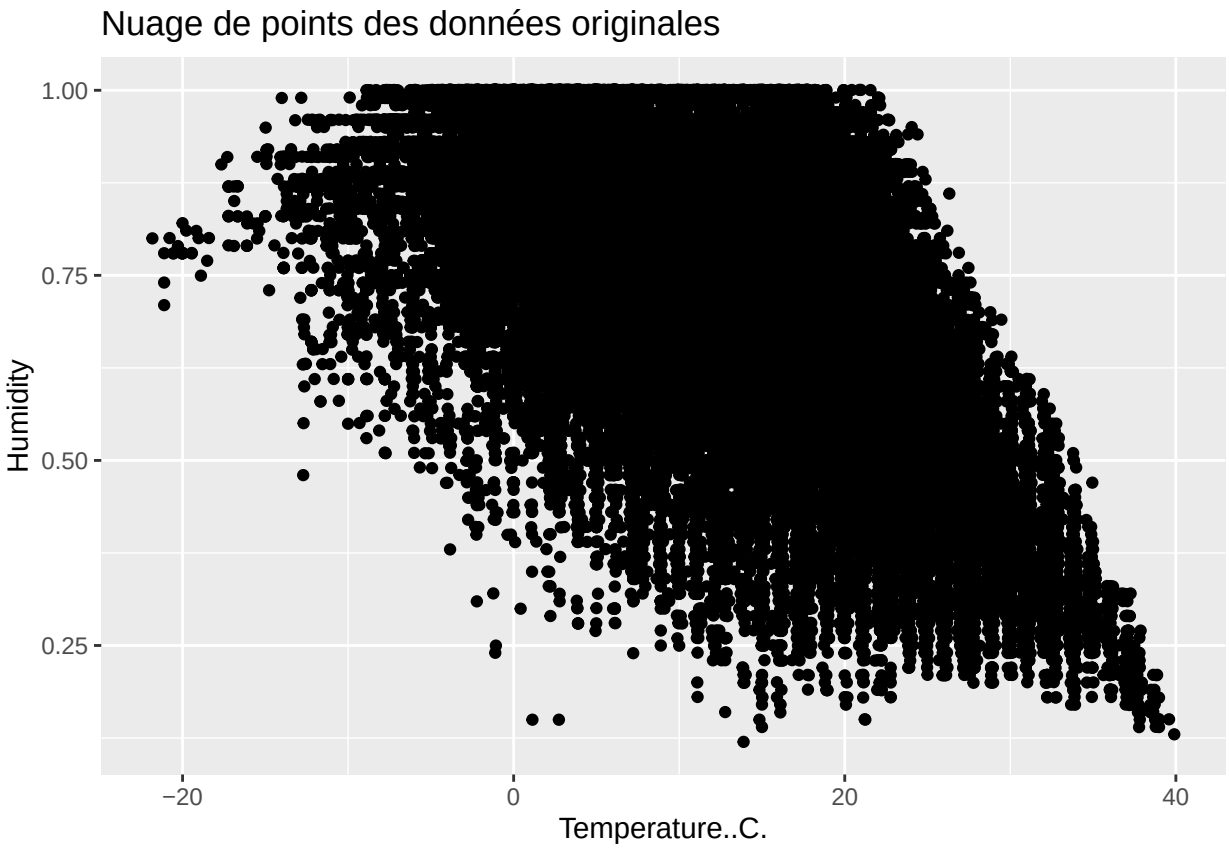
```
cor(data_cl, method = "spearman")
```

```
##                     Temperature..C. Wind.Speed..km.h. Pressure..millibars.
```

```
## Temperature..C.           1.00000000        0.01802144        -0.31445614
## Wind.Speed..km.h.         0.01802144        1.00000000        -0.22792373
## Pressure..millibars.     -0.31445614       -0.22792373         1.00000000
## Humidity                 -0.59005848       -0.26111971         0.04345611
##                         Humidity
## Temperature..C.      -0.59005848
## Wind.Speed..km.h.    -0.26111971
## Pressure..millibars.  0.04345611
## Humidity              1.00000000
```

```
#cor(data_cl, method = "kendall")
```

```
ggplot(data_cl, aes(x = Temperature..C., y = Humidity)) + geom_point() + labs(title = "Nuage de points
```



Pseudo-observations

La seconde étape (cruciale) consiste à transférer les données en pseudo-observations sur $(0, 1)$, pour ne pas être perturbé par les marginales. Cela peut se faire avec la fonction `pobs`, puis on trace le nuage de poitns des pseudo-observations.

```
#U <- pobs(as.matrix(data))
U_brut <- pobs(data_cl)
colnames(U_brut) <- c("U_temp", "U_wind", "U_pressure", "U_humidity")
```

```r
eps <- 1e-4  #on limite les extremum  pour eviter des problemes avec la log vraisemblance
U<- pmin(pmax(U_brut, eps), 1 - eps)

n <- nrow(U)
n_sub <- min(3000, n)   # 2000-3000 suffisent largement
idx_sub <- sample(seq_len(n), n_sub)

U_sub <- U[idx_sub, ]


idx_light <- sample(seq_len(n), size = floor(0.5 * n))

U_light <- U[idx_light, ]
```

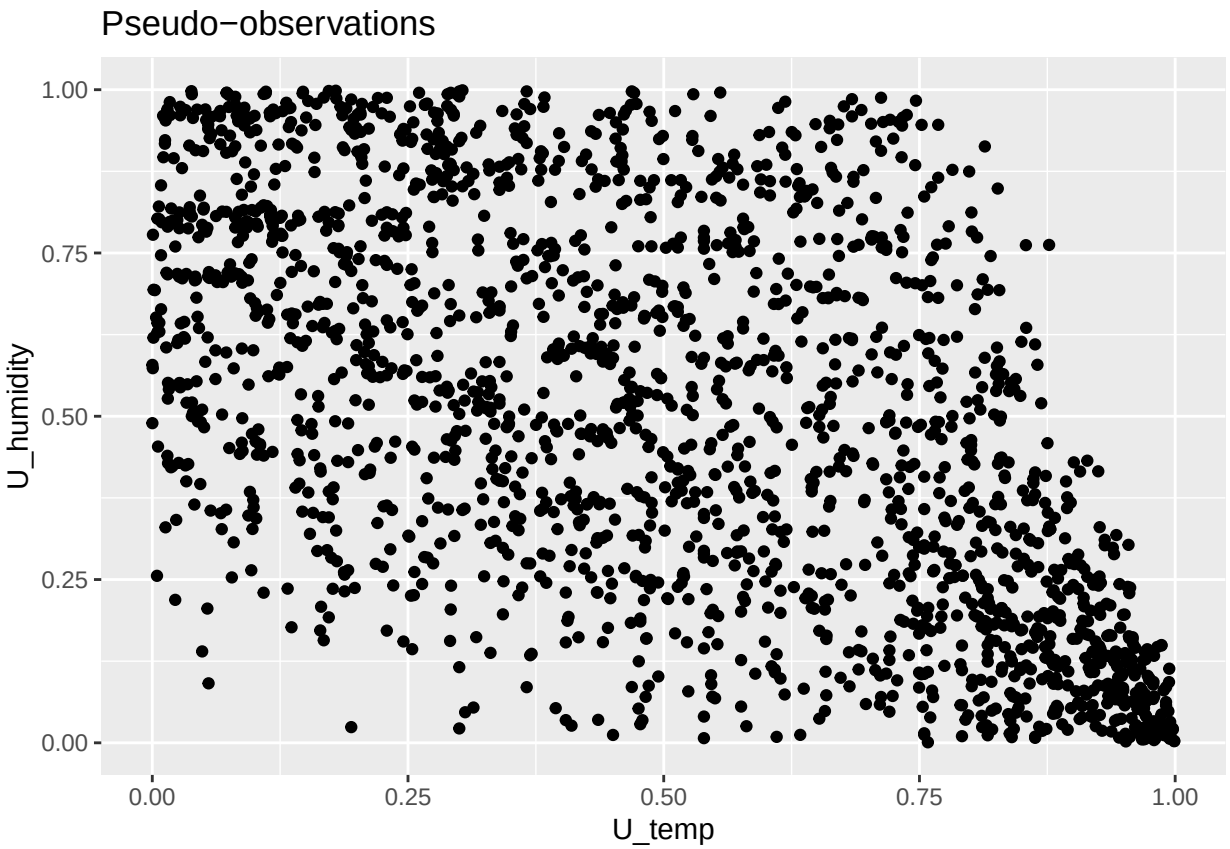
```r
summary(U[, c(1,4)])
```

```
##      U_temp         U_humidity
##  Min.   :0.0001   Min.   :0.0001
##  1st Qu.:0.2500   1st Qu.:0.2500
##  Median :0.4999   Median :0.5000
##  Mean   :0.5000   Mean   :0.5000
##  3rd Qu.:0.7496   3rd Qu.:0.7500
##  Max.   :0.9999   Max.   :0.9999
```

```r
#idx <- sample(1:nrow(U), size = 1000)
#pairs(U[idx, c(1,4) ])


#U_sub <- U[, c(1, 4)]
#U_sub[, 2] <- 1 - U_sub[, 2]
#pairs(U[idx, c(1,4)])
```

```r
idx <- sample(1:nrow(U), size = 2000)
ggplot(as.data.frame(U[idx, ]), aes(x = U_temp, y = U_humidity)) +
  geom_point() +
  labs(title = "Pseudo-observations",
       x = "U_temp", y = "U_humidity")
```



Sur ce deuxième nuage de points, on observe uniquement la dépendance. Nous pouvons confirmer que les données semblent présenter une dépendance et potentiellement une dépendance dans la queue de distributions inférieure droite.

Nous pourrions donc penser à une copule de clayton (tournée à 270°) ou de gumbel (tournée à 90°). Il est aussi possible de faire des transformations sur les données : par exemple U_temp = 1 - U_temp pour avoir quelquechose qui ressemble directement à la copule de clayton standard.

```
#cor(U, method = "pearson")
#cor(data_cl, method = "spearman")
cor(U_sub, method = "kendall")
```

```
##                  U_temp       U_wind U_pressure U_humidity
## U_temp       1.000000000  0.002446052 -0.2315644 -0.4177824
## U_wind       0.002446052  1.000000000 -0.1415108 -0.1670792
## U_pressure -0.231564379 -0.141510752  1.0000000  0.0406565
## U_humidity -0.417782445 -0.167079174  0.0406565  1.0000000
```

Test d'indépendance

Avant toute étude de dépendance, et en particulier à l'aide de copules, il est important de réaliser un test d'indépendance pour confirmer que l'observation sur les nuages de points et s'assurer qu'il existe une relation entre les deux composantes du vecteur.

On peut commencer avec un test de corrélation de rangs à l'aide du τ de Kendall, avec la fonction `cor.test`

```r
cor.test(x=U_sub[,1], y=U_sub[,4], method="kendall")
```

```
##
##  Kendall's rank correlation tau
##
## data:  U_sub[, 1] and U_sub[, 4]
## z = -34.295, p-value < 2.2e-16
## alternative hypothesis: true tau is not equal to 0
## sample estimates:
##        tau
## -0.4177824
```

La p-valeur permet de rejeter sans scrupule l'hypothèse nulle, soit que les données n'ont pas de relation monotone. Il est possile de réaliser également le test à l'aide de coefficient de Spearman.

```r
cor.test(x=U_sub[,1], y=U_sub[,4], method="spearman") #ici meme chose que le test de pearson puisque le
```

```
## Warning in cor.test.default(x = U_sub[, 1], y = U_sub[, 4], method =
## "spearman"): Cannot compute exact p-value with ties
```

```
##
##  Spearman's rank correlation rho
##
## data:  U_sub[, 1] and U_sub[, 4]
## S = 7.158e+09, p-value < 2.2e-16
## alternative hypothesis: true rho is not equal to 0
## sample estimates:
##       rho
## -0.590658
```

La conclusion est la même.

Nous allons maintenant réaliser un test d'indépendance, à l'aide de la fonction `indepTest`, qui compare la copule empirique à la copule d'indépendance.

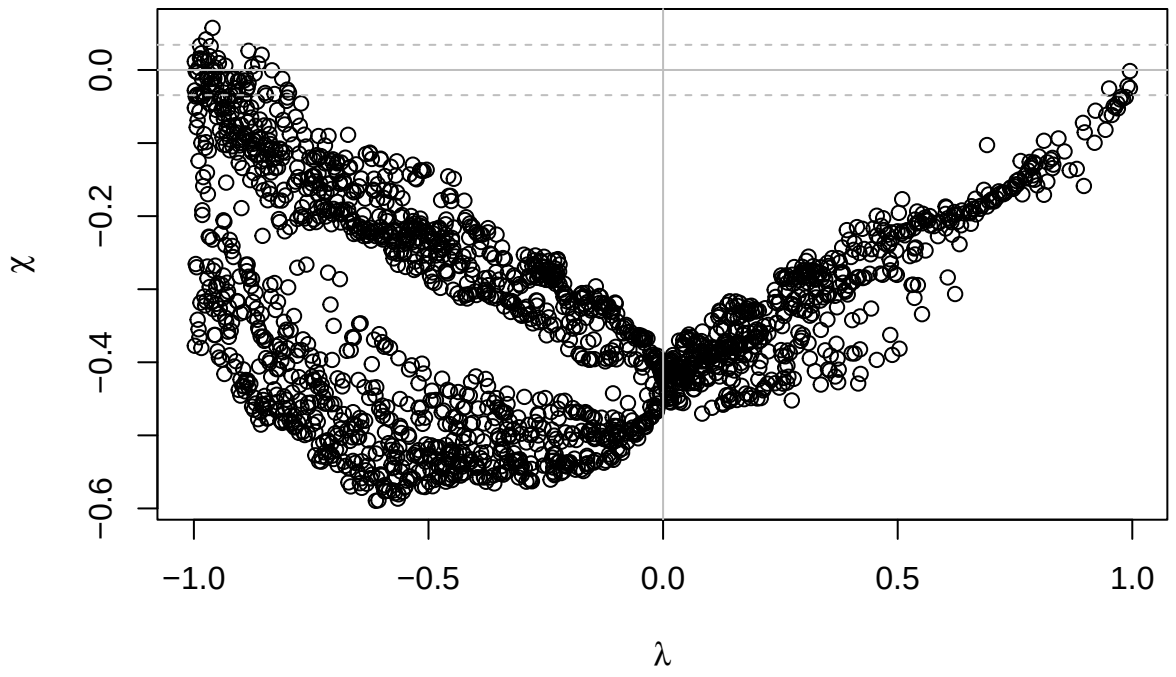
```r
idx <- sample(1:nrow(U), size = 2000)
d <- indepTestSim(n=2000, p = 2, verbose = FALSE)
indepTest(x = U[idx,c(1,4)], d = d, alpha = 0.05)
```

```
##
## Global Cramer-von Mises statistic: 7.531215 with p-value 0.0004995005
## Combined p-values from the Mobius decomposition:
##    0.0004995005  from Fisher's rule,
##    0.0004995005  from Tippett's rule.
```

Nous pouvons ensuite réaliser un χ-plot et un K-plot à partir des pseudo-observations.

χ-plot sur U

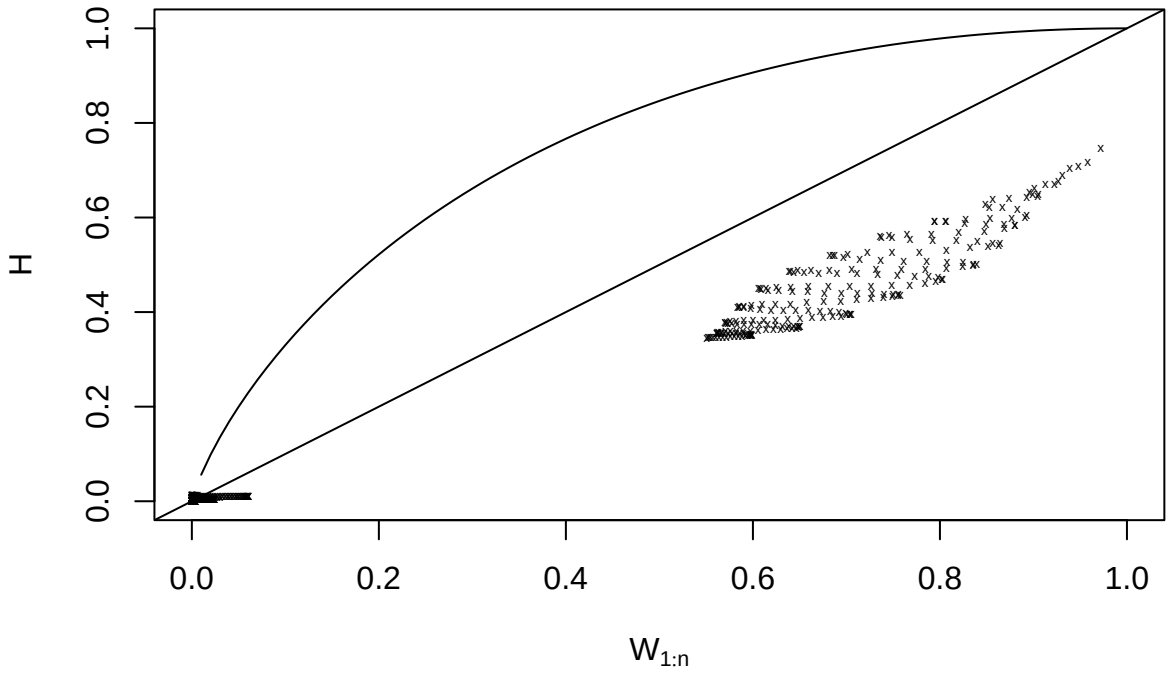
```
BiCopChiPlot(u1 = U[idx,1], u2 = U[idx,4], PLOT = T)
```



dépendance réelle. Beaucoup de points négatifs -> dépendance négative. la structure n'est pas symétrique -> Asymétrie de dépendance de queue

K-plot

```
BiCopKPlot(u1 = U[idx,1], u2 = U[idx,4])
```



Coefficients de dépendance de queue

Pour estimer les coefficients de dépendance de queue, nous créons des fonctions qui permettent de calculer, pour q petit,

$$\lambda_U(q) = \frac{2q - 1 + C(1-q, 1-q)}{q}$$

et

$$\lambda_L(q) = \frac{C(q,q)}{q}$$

On utilise donc les versions empiriques

$$\hat{\lambda}_U(q) = \frac{\sum_{i=1}^n \mathbf{1}\{U_i > 1-q,\, V_i > 1-q\}}{\sum_{i=1}^n \mathbf{1}\{U_i > 1-q\}}.$$

et

$$\hat{\lambda}_L(q) = \frac{\sum_{i=1}^n \mathbf{1}\{U_i \leq q,\, V_i \leq q\}}{\sum_{i=1}^n \mathbf{1}\{U_i \leq q\}}.$$

Puis on les calculs pour plusieurs petites valeurs de $q = 0.01, 0.05, 0.1$

```r
lambda_U_hat <- function(U, q ) {
  u <- 1- U[,1] #on tourne le nuage de points pour avoir la dépendance forte de queue en bas à gauche
  v <- U[,4]
  num <- sum(u > 1-q & v > 1-q)
  den <- sum(u > 1-q)
  if (den == 0) return(NA_real_)
  num / den
}

# lower tail
lambda_L_hat <- function(U, q ) {
  u <- 1-U[,1]
  v <- U[,4]
  num <- sum(u <= q & v <= q)
  den <- sum(u <= q)
  if (den == 0) return(NA_real_)
  num / den
}
```

```r
## lambda_L
lambda_L_hat(U=U, q = 0.1)
```

```
## [1] 0.5588421
```

```r
lambda_L_hat(U=U, q = 0.05)
```

```
## [1] 0.4432968
```

```r
lambda_L_hat(U=U, q = 0.01)
```

```
## [1] 0.3385254
```

```r
## lambda_U
lambda_U_hat(U=U, q = 0.1)
```

```
## [1] 0.2230842
```

```r
lambda_U_hat(U=U, q = 0.05)
```

```
## [1] 0.08950813
```

```r
lambda_U_hat(U=U, q = 0.01)
```

```
## [1] 0.005257624
```

On observe que λ_U est proche de 0 pour des valeurs de q petites mais déja pour q =0.1 plus si proche de 0. λ_L est croissant en q.

Estimation par la méthode de moments

```r
tau_emp <- cor(U_sub[,1], U_sub[,4], method = "kendall")
theta_clay270_mm <- iTau(claytonCopula(), -tau_emp) #on prends -tau car on les tourne -> correlation né
theta_gumb90_mm <- iTau(gumbelCopula(), -tau_emp)

theta_gumb90_mm; theta_clay270_mm
```

```
## [1] 1.717571
```

```
## [1] 1.435142
```

Estimation par maximum de vraisemblance

Pour l'estimation par maximum de vraisemblance, il suffit d'utiliser la fonction `fitCopula` avec l'option `method='ml'`.

```r
copulas <- list(
  clay270  = rotCopula(claytonCopula(dim=2), flip=c(TRUE, FALSE)),
  #gumb270  = rotCopula(gumbelCopula(dim=2),  flip=c(TRUE, FALSE)),
  gumb90 = rotCopula(gumbelCopula(dim = 2), flip = c(FALSE, TRUE)),
  frank270 = rotCopula(frankCopula(dim=2),   flip=c(TRUE, FALSE))
)

# Estimation par pseudo-vraisemblance
fits <- lapply(copulas, fitCopula, data=U_sub[,c(1,4)], method="mpl")


#fits <- list(
  #gauss = fitCopula(normalCopula(dim=2), U_sub[,c(1,4)]),
  #t     = fitCopula(tCopula(dim=2),      U_sub[,c(1,4)]),
  #clayR = fitCopula(rotCopula(claytonCopula(dim=2)), U_light[,c(1,4)])

 # clayR = fitCopula(rotCopula(claytonCopula(dim=2), flip=c(TRUE, FALSE)), U_sub[,c(1,4)]))
```

```r
theta_gumb_mle <- coef(fits$gumb90)
theta_clay270_mle <- coef(fits$clay270)
theta_gumb_mle; theta_clay270_mle #maximum de vraisemblance
```

```
##    alpha
## 1.699651
```

```
##    alpha
## 1.435142
```

```r
theta_gumb90_mm; theta_clay270_mm #méthode des moments
```

```
## [1] 1.717571
```

```
## [1] 1.435142
```

les resultats entre mle et méthode des moments sont semblables.

Comparaison des deux copules

```
sapply(fits, AIC)
```

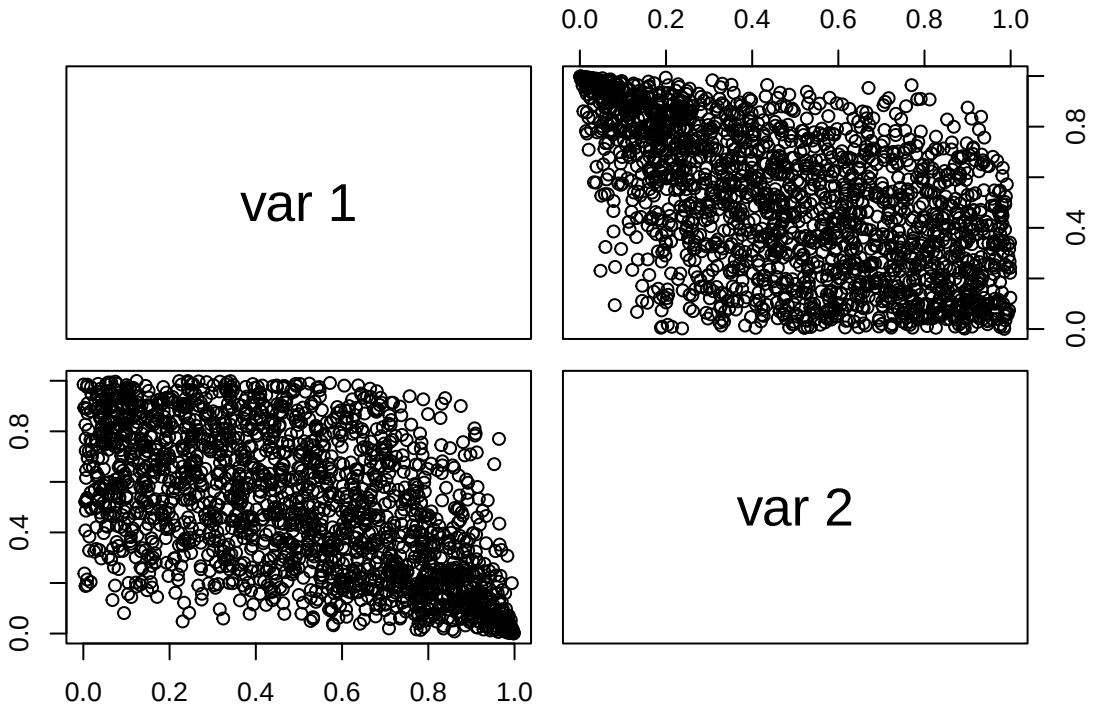
```
##   clay270    gumb90  frank270
## -1509.991 -1492.691 -1260.740
```

```
sapply(fits, BIC)
```

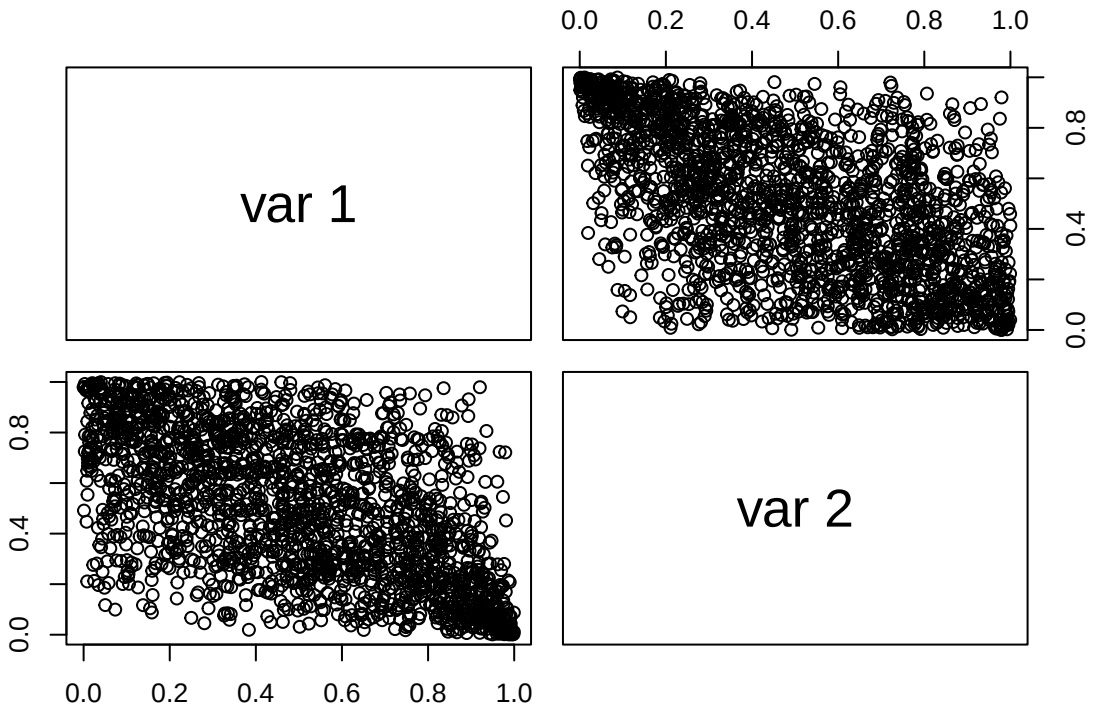
```
##   clay270    gumb90  frank270
## -1503.985 -1486.684 -1254.734
```

la copule de clayton semble etre la meilleur

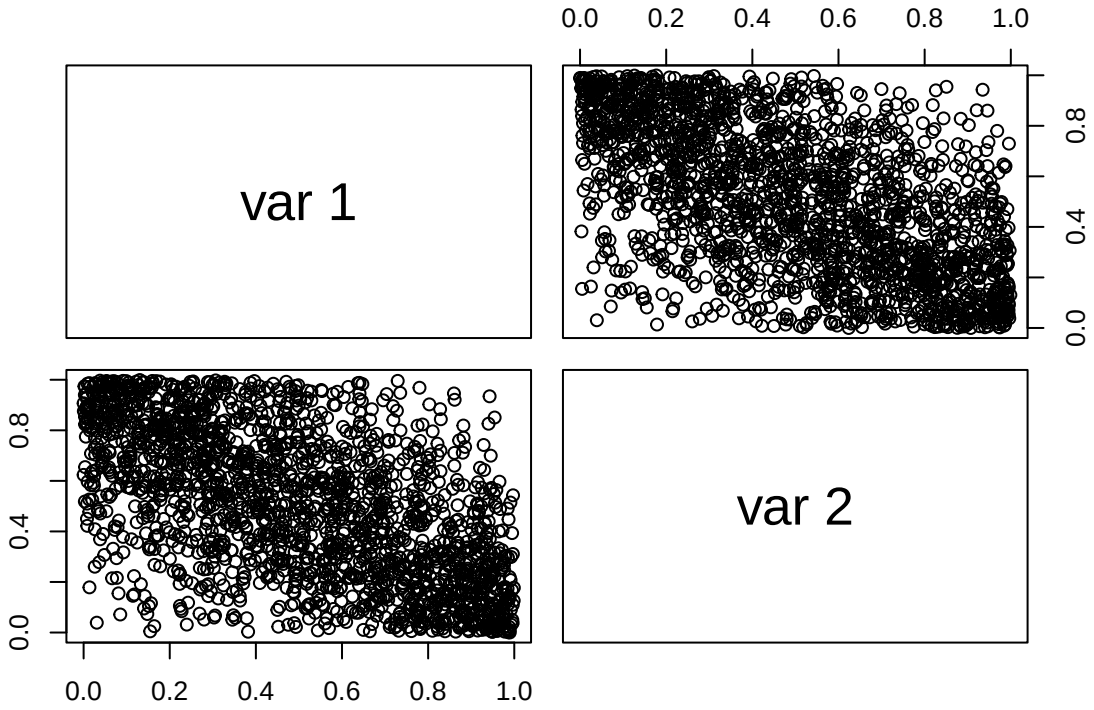
```
pairs(rCopula(2000, fits$clay270@copula))
```



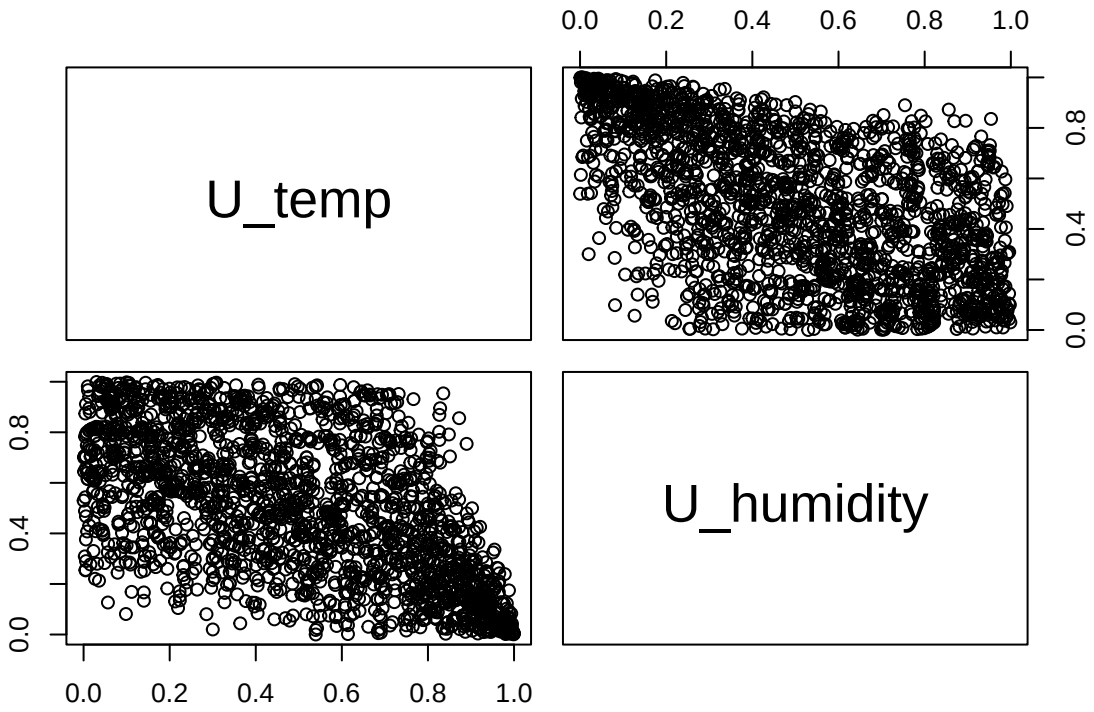

```
pairs(rCopula(2000, fits$gumb90@copula))
```




```
pairs(rCopula(2000, fits$frank270@copula))
```

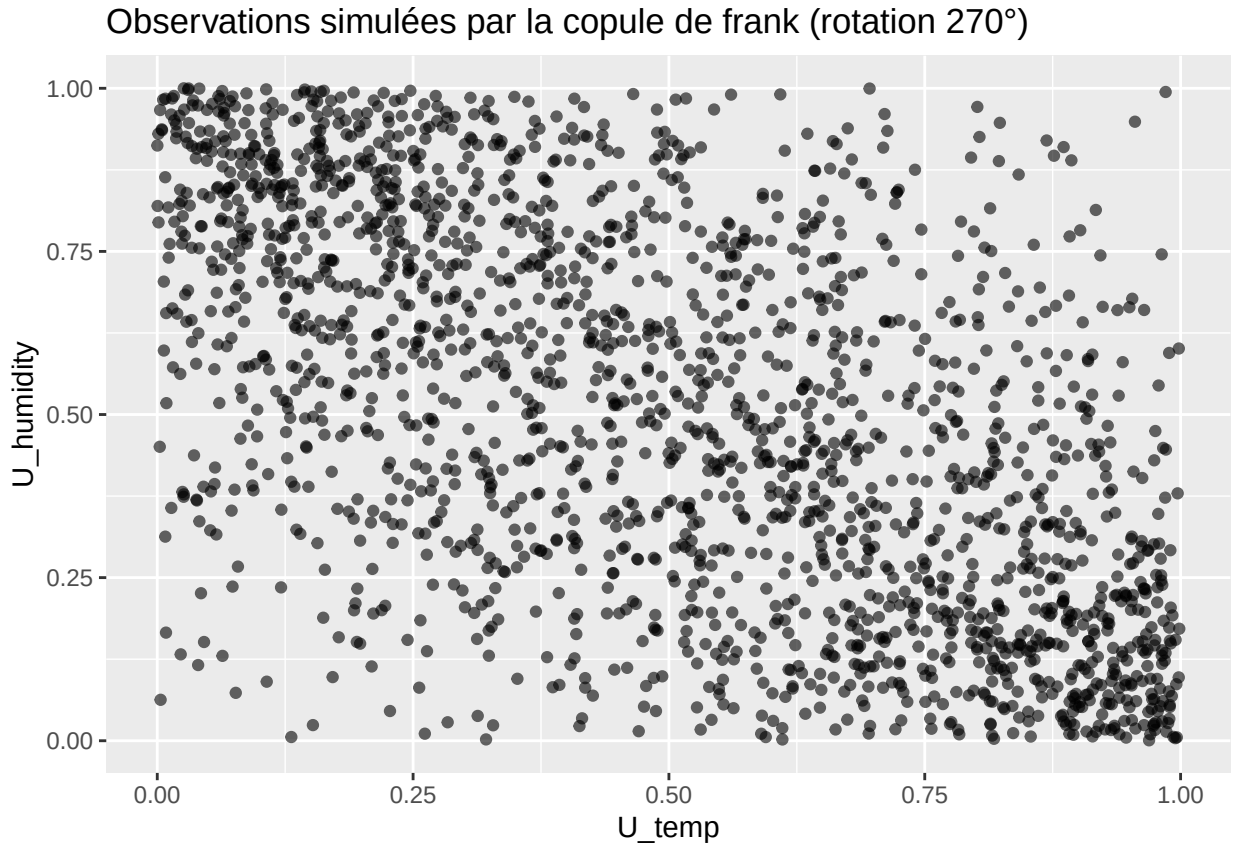



```
idx <- sample(1:nrow(U), size = 2000)
pairs(U[idx, c(1,4) ])
```




```r
#idx <- sample(1:nrow(U), size = 2000)
sim <- as.data.frame(rCopula(2000, fits$frank270@copula))
colnames(sim) <- c("U_temp", "U_humidity")

ggplot(sim, aes(x = U_temp, y = U_humidity)) +
  geom_point(alpha = 0.6) +
  labs(
    title = "Observations simulées par la copule de frank (rotation 270°)",
    x = "U_temp",
    y = "U_humidity"
  )
```



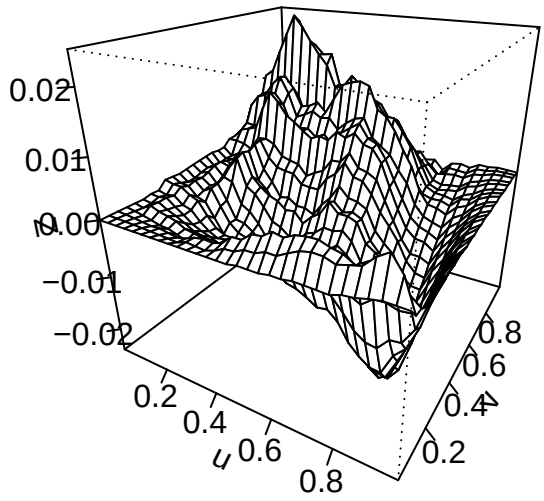

```r
C_emp <- empCopula(U_sub[,c(1,4)])
C_theo <- fits$gumb90@copula

u <- seq(0.01, 0.99, length = 30)
grid <- expand.grid(u, u)
C_emp_val  <- pCopula(as.matrix(grid), C_emp)
C_theo_val <- pCopula(as.matrix(grid), C_theo)

Z <- matrix(C_emp_val - C_theo_val, nrow = length(u))
zlim <- range(Z)

persp(
  u, u, Z,
  theta = 30, phi = 25,
  xlab = "u",
  ylab = "v",
  #zlab = "C_emp - C_theo",
  main = "Différence entre copule empirique et théorique gumbel",
  zlim = zlim,
  ticktype = "detailed"
)
```

Différence entre copule empirique et théorique gumbel

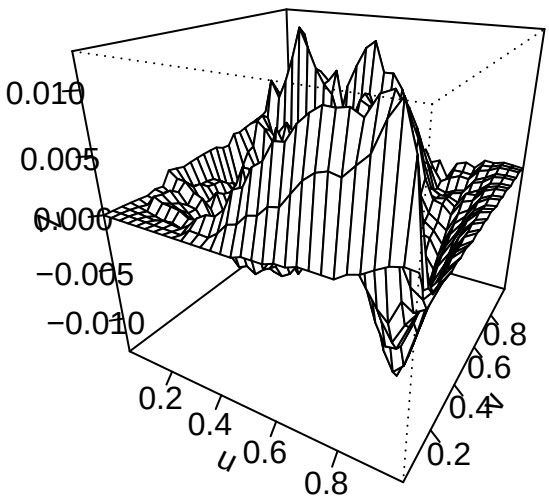



```r
C_emp <- empCopula(U_sub[,c(1,4)])
C_theo <- fits$clay270@copula

u <- seq(0.01, 0.99, length = 30)
grid <- expand.grid(u, u)
C_emp_val  <- pCopula(as.matrix(grid), C_emp)
C_theo_val <- pCopula(as.matrix(grid), C_theo)

Z <- matrix(C_emp_val - C_theo_val, nrow = length(u))
zlim <- range(Z)

persp(
  u, u, Z,
  theta = 30, phi = 25,
  xlab = "u",
  ylab = "v",
  #zlab = "C_emp - C_theo",
  main = "Différence entre copule empirique et théorique clayton",
  zlim = zlim,
  ticktype = "detailed"
)
```



Tests de qualité d'ajustement

Test de Cramér-von Mises

Nous commençons par effectuer un test de Cramér-von Mises à l'aide de la fonction `gofCopula`.

```
gof_clay270 <- gofCopula(
  copula = fits$clay270@copula,
  x = U_sub[,c(1,4)],
  method = "Sn",
  N = 30,
  optim.method = "BFGS"
)
```

```
## Warning in .gofPB(copula, x, N = N, method = method, estim.method =
## estim.method, : argument 'ties' set to TRUE
```

```
gof_gumb90 <- gofCopula(
  copula = fits$gumb90@copula,
  x = U_sub[,c(1,4)],
  method = "Sn",
  N = 30,
  optim.method = "BFGS"
)
```

```
## Warning in .gofPB(copula, x, N = N, method = method, estim.method =
## estim.method, : argument 'ties' set to TRUE
```

```
gof_frank270 <- gofCopula(
  copula = fits$frank270@copula,
  x = U_sub[,c(1,4)],
  method = "Sn",
  N = 30,
  optim.method = "BFGS"
)
```

```
## Warning in .gofPB(copula, x, N = N, method = method, estim.method =
## estim.method, : argument 'ties' set to TRUE
```

```
c(
  clay270  = gof_clay270$p.value,
  gumb90   = gof_gumb90$p.value,
  frank270 = gof_frank270$p.value
)
```

```
##    clay270     gumb90   frank270
## 0.01612903 0.01612903 0.01612903
```

Comparaison des lignes de niveau

On cherche à superposer les lignes de niveau de la copule empirique avec lignes de niveau des copules de Gumbel et de Clayton. On commmence par calculer la colupe empirique

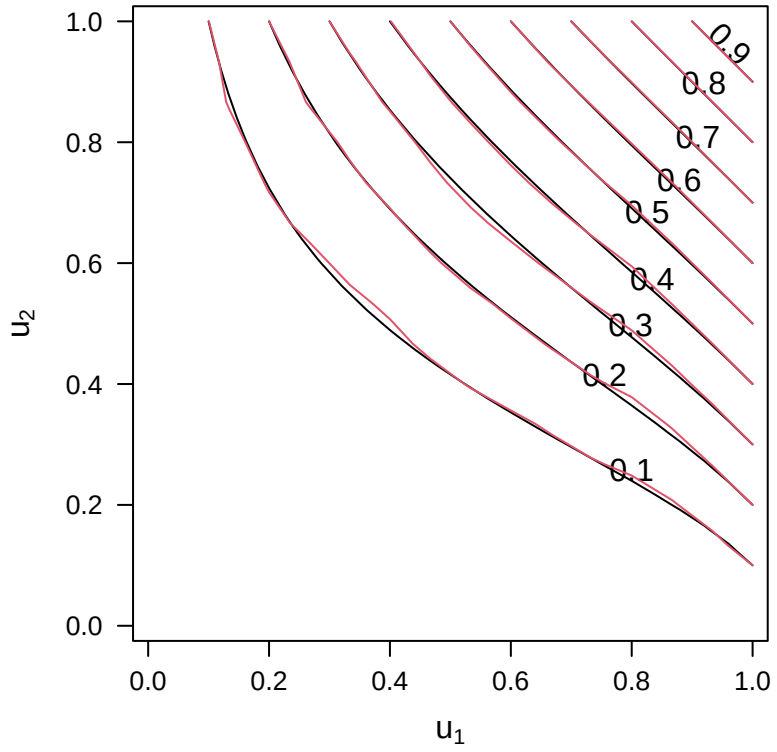
```
emp_cop <- empCopula(U_sub[,c(1,4)])
```

Copule de clayton

```
cp_clay <- contourplot2(fits$clay270@copula, FUN = pCopula, region = FALSE)

u <- seq(0, 1, length.out = 16)
grid <- as.matrix(expand.grid(u1 = u, u2 = u))
val <- cbind(grid, z=C.n(grid, X = U_sub[,c(1,4)]))
cp_emp <- contourplot2(val, region = F, labels = F, col = 2)

cp_clay + cp_emp
```



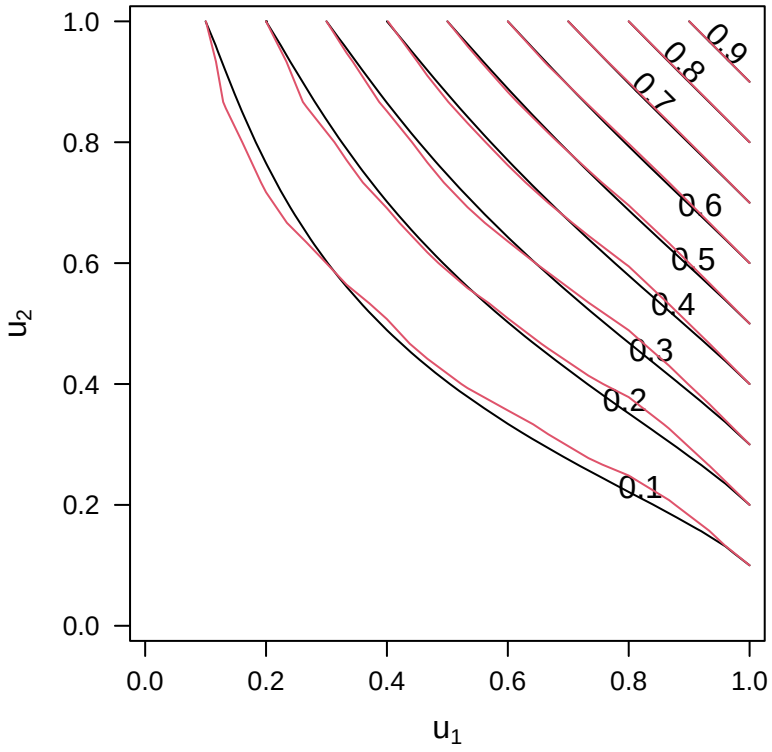
les lignes de niveau sont assez proches

Copule de gumbel

```
cp_gumb <- contourplot2(fits$gumb90@copula, FUN = pCopula, region = FALSE)

u <- seq(0, 1, length.out = 16)
grid <- as.matrix(expand.grid(u1 = u, u2 = u))
val <- cbind(grid, z=C.n(grid, X = U_sub[,c(1,4)]))
cp_emp <- contourplot2(val, region = F, labels = F, col = 2)

cp_gumb + cp_emp
```

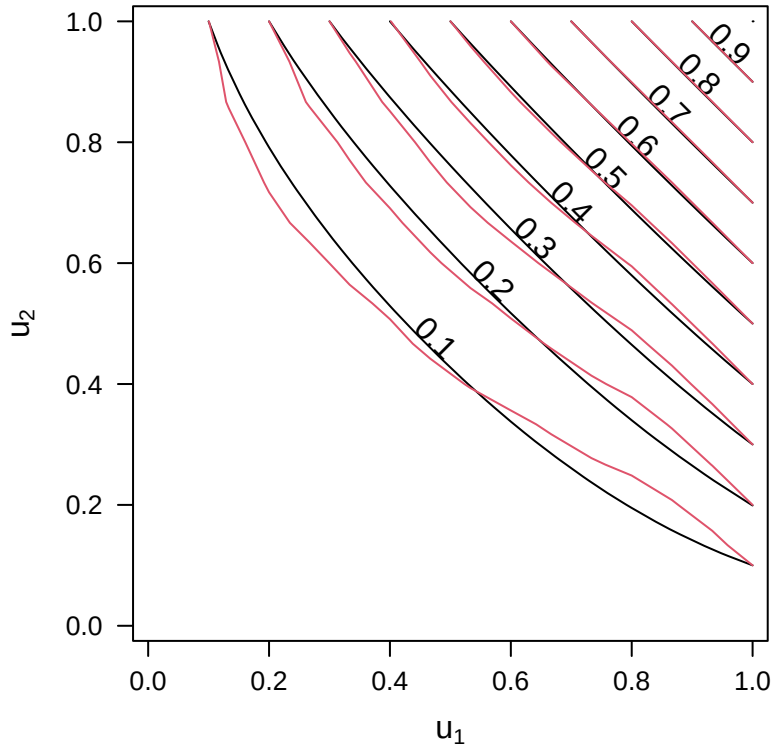


Copule de frank

```
cp_frank <- contourplot2(fits$frank270@copula, FUN = pCopula, region = FALSE)

u <- seq(0, 1, length.out = 16)
grid <- as.matrix(expand.grid(u1 = u, u2 = u))
val <- cbind(grid, z=C.n(grid, X = U_sub[,c(1,4)]))
cp_emp <- contourplot2(val, region = F, labels = F, col = 2)

cp_frank + cp_emp
```



On observe que les lignes de niveau de la copule de Gumbel sont pas vraiment proches de celles de la copule empirique.