Copula Estimation - Univariate Case

2025-11-27

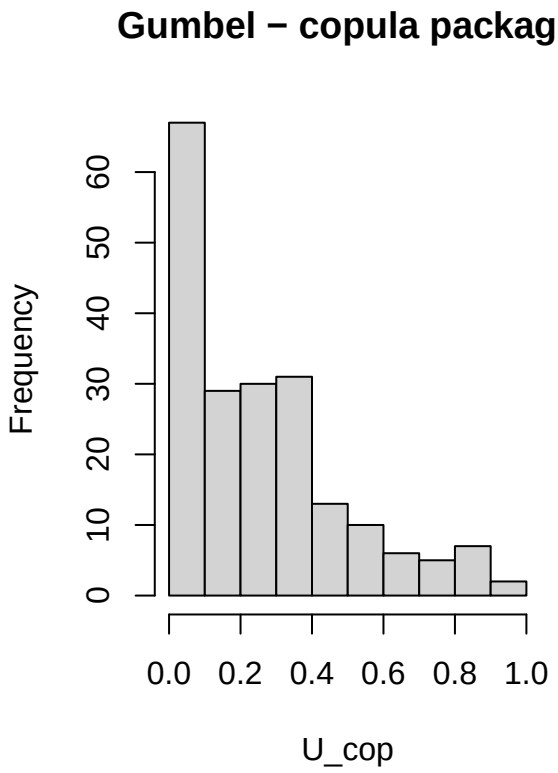
1. Sampling with copula package

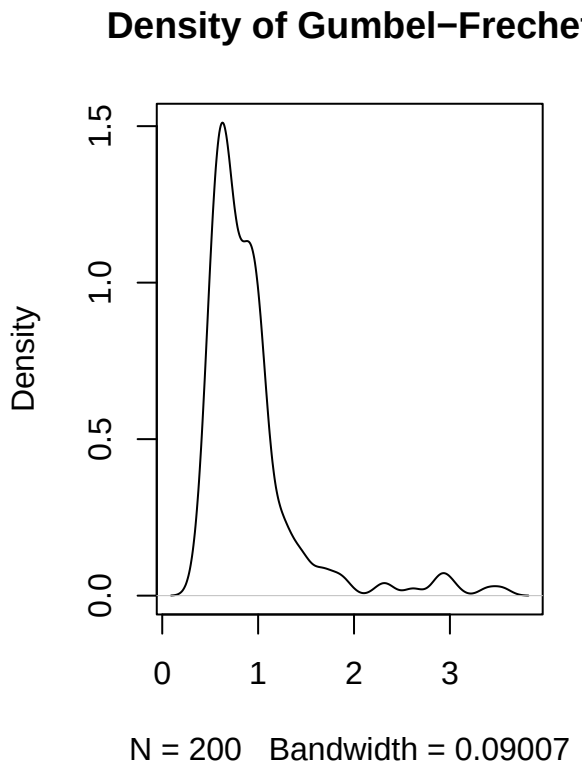
The copula package automatically samples bivariate and multivariate copulas with a convenient function. To use it in our univariate scenario, we conside the number of observations in the dataset as the dimension of the copula. Below the Gumbel case:

```
set.seed(46)
library(copula)
theta <- 1.5
n <- 200

Gcop <- gumbelCopula(param = theta, dim = n)

U_cop <- rCopula(1, Gcop)
```

Choosing the margin allows to construct a sample $X_1, ..., X_n \sim H = C(F(x_1), ..., F(x_n))$ by applying $X_i = F^{-1}(U_i)$. For example, taking $X_i \sim Frechet(2)$:

```
alpha <- 2
X_cop <- qfrechet(U_cop, shape = alpha)
```

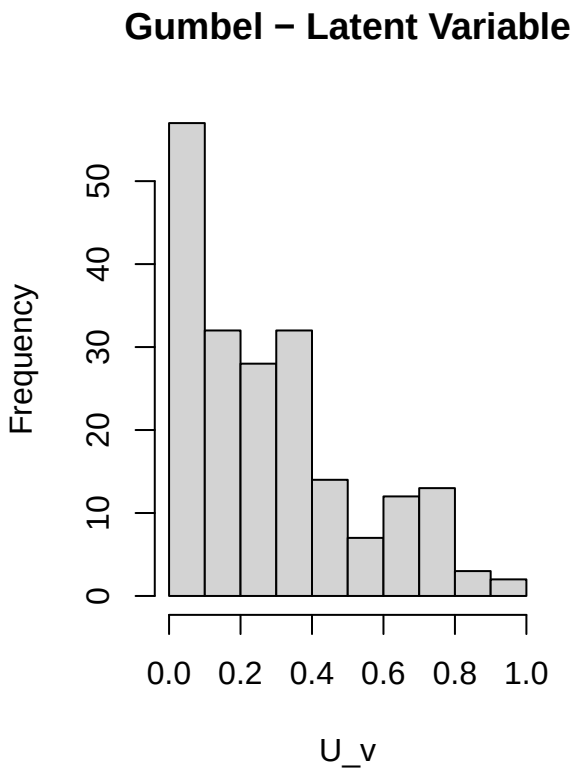


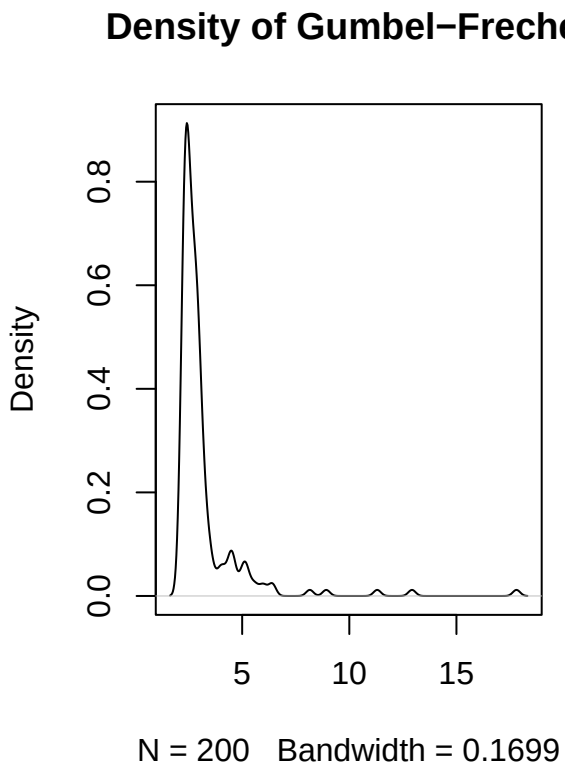


2. Stochastic Representation with latent V

To sample from the same copula, we can use the latent variable representation. The latent variable V whose Laplace transform is the Gumbel generator $\psi(t) = \exp\{-t^{\frac{1}{\theta}}\}$ has distribution $F_V \sim Stable(\alpha = 1/\theta, \beta = 1, \gamma = (cos(\frac{\pi}{2\theta}))^{\theta}, \delta = 0; pm = 1)$.

```
set.seed(46)
library(stabledist)

gum_psi <- function(t, theta){
    exp(- t ^ (1/theta))
}

## Stable parameters for gumbel

V <- rstable(
    n = 1,
    alpha = 1/theta,
    beta = 1,
    gamma = cospi(1/(2 * theta))^theta,
    delta = 0,
    pm = 1
)

E <- rexp(n, V)

U_v <- gum_psi(E, theta)

X_v <- qfrechet(U_v, alpha)
```





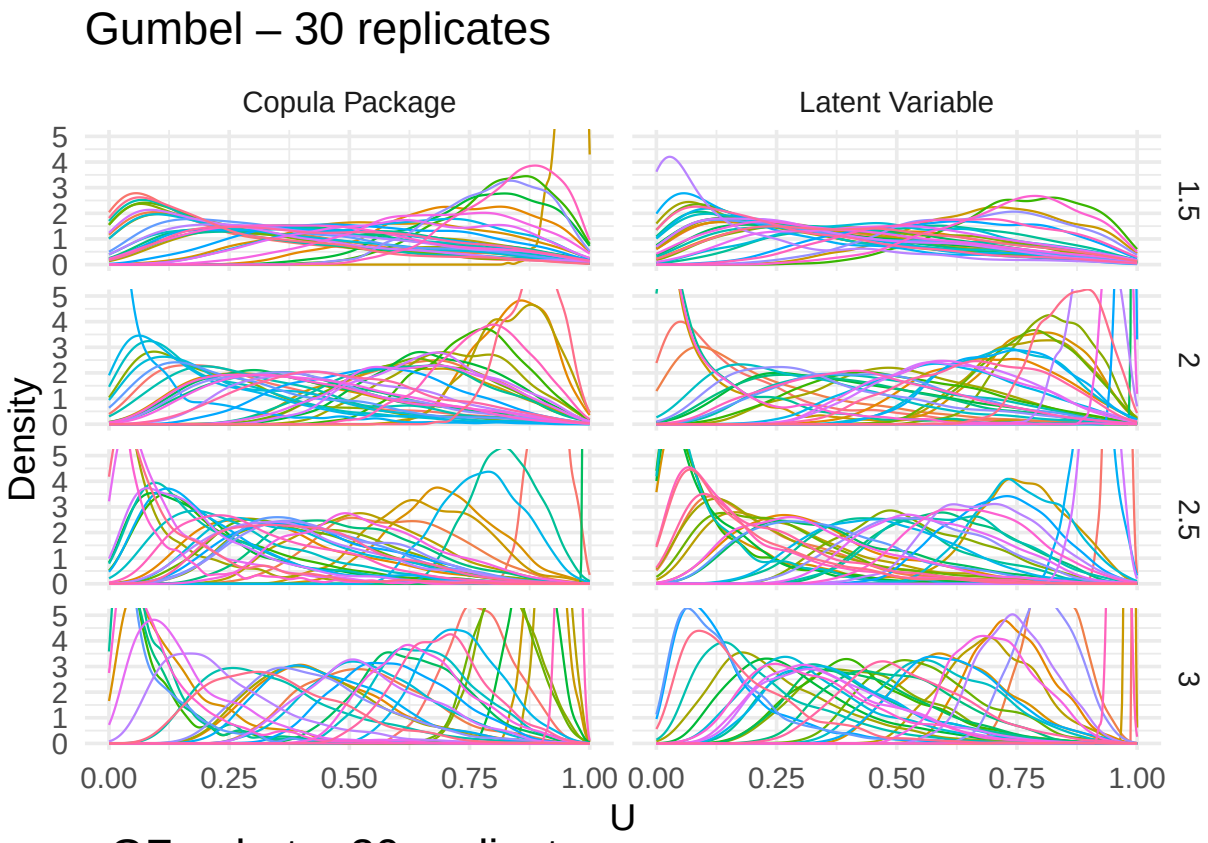
For convenience, we write the sampler function for the U values (the quantile function is left unspecified to allow the freedom to choose the margin F):

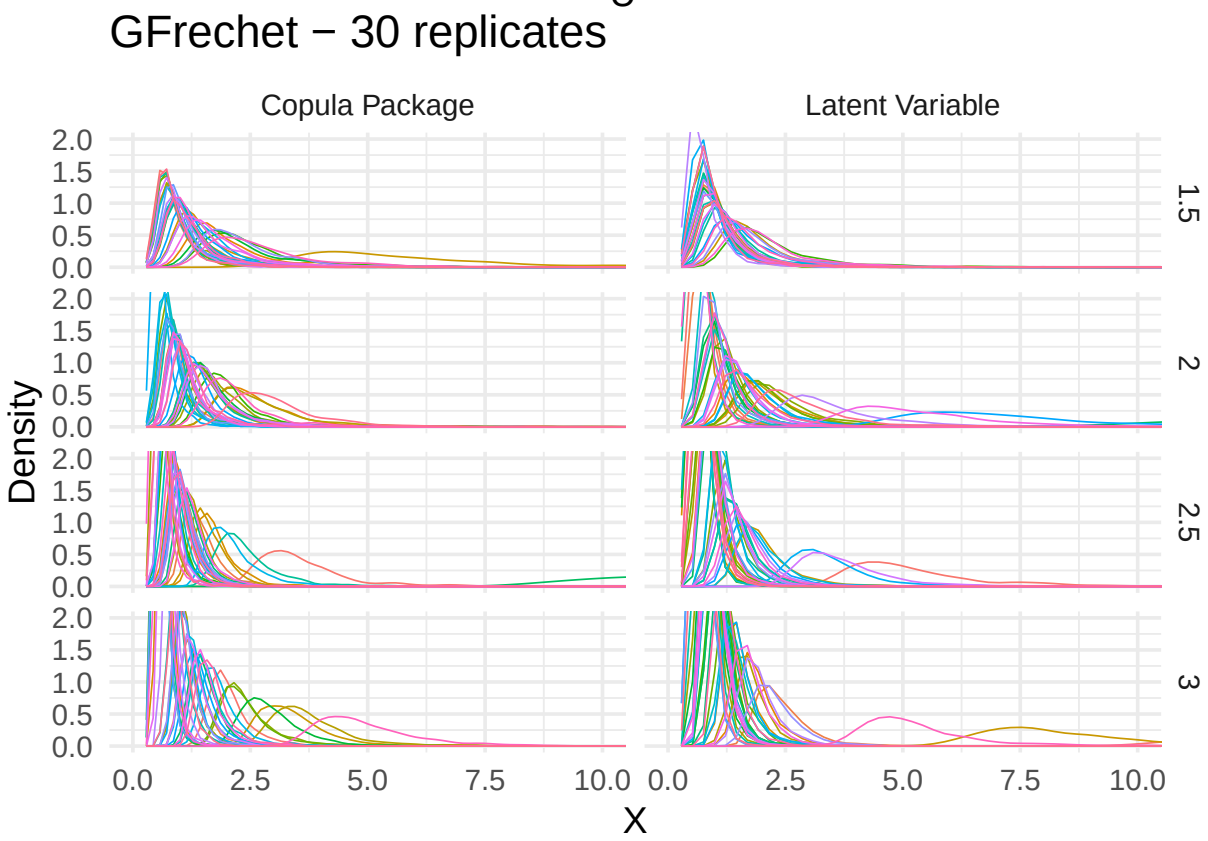
```
# Latent variable
rGumbV <- function(n, theta) {
    require(stabledist)

    # Gumbel V r.v.
    V <- rstable(
        n = 1,
        alpha = 1/theta,
        beta = 1,
        gamma = cospi(1/(2 * theta))^theta,
        delta = 0,
        pm = 1
    )

    E <- rexp(n, V)

    # Gumbel generator
    gum_psi <- function(t, theta){
        exp(- t ^ (1/theta))
    }

    #(U_1, ..., U_n) ~ C_psi
    U <- gum_psi(E, theta)

    return(U)
}
```

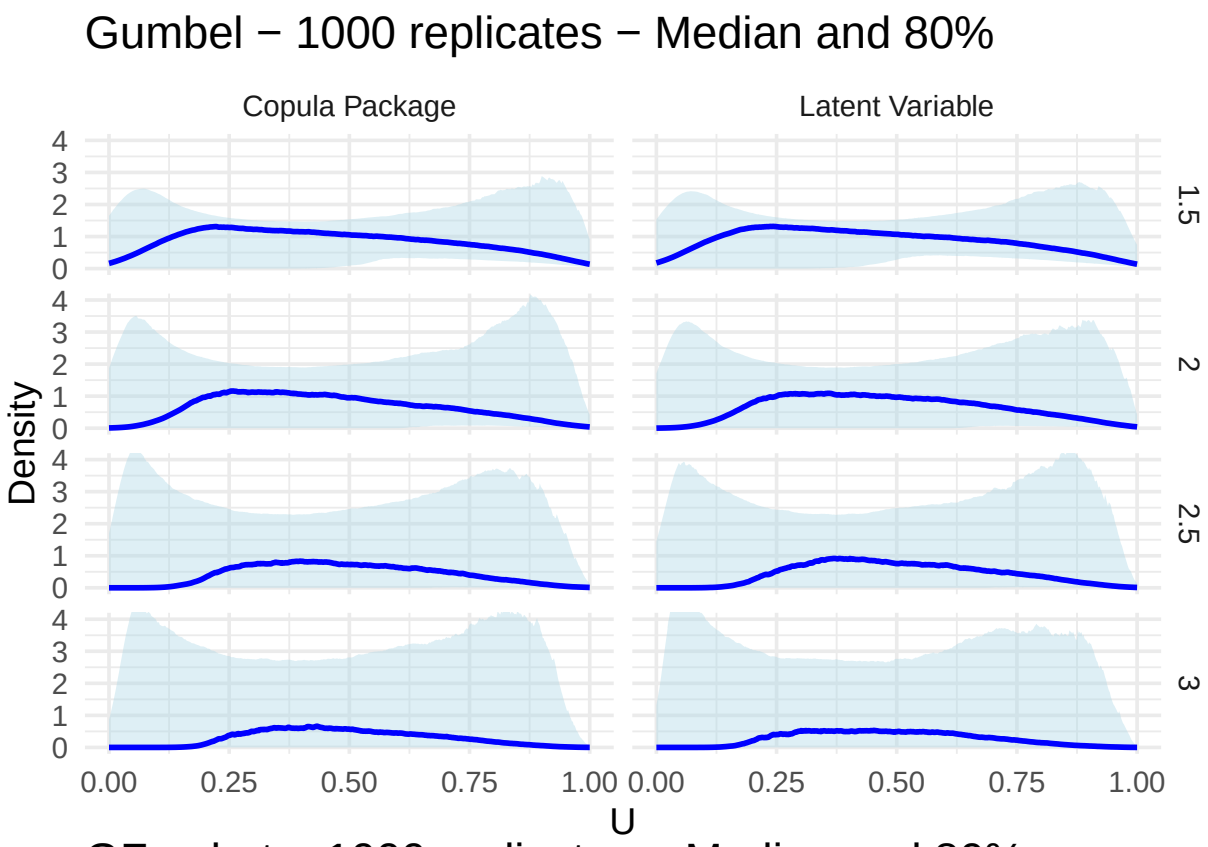
3. Model likelihood and identifiability

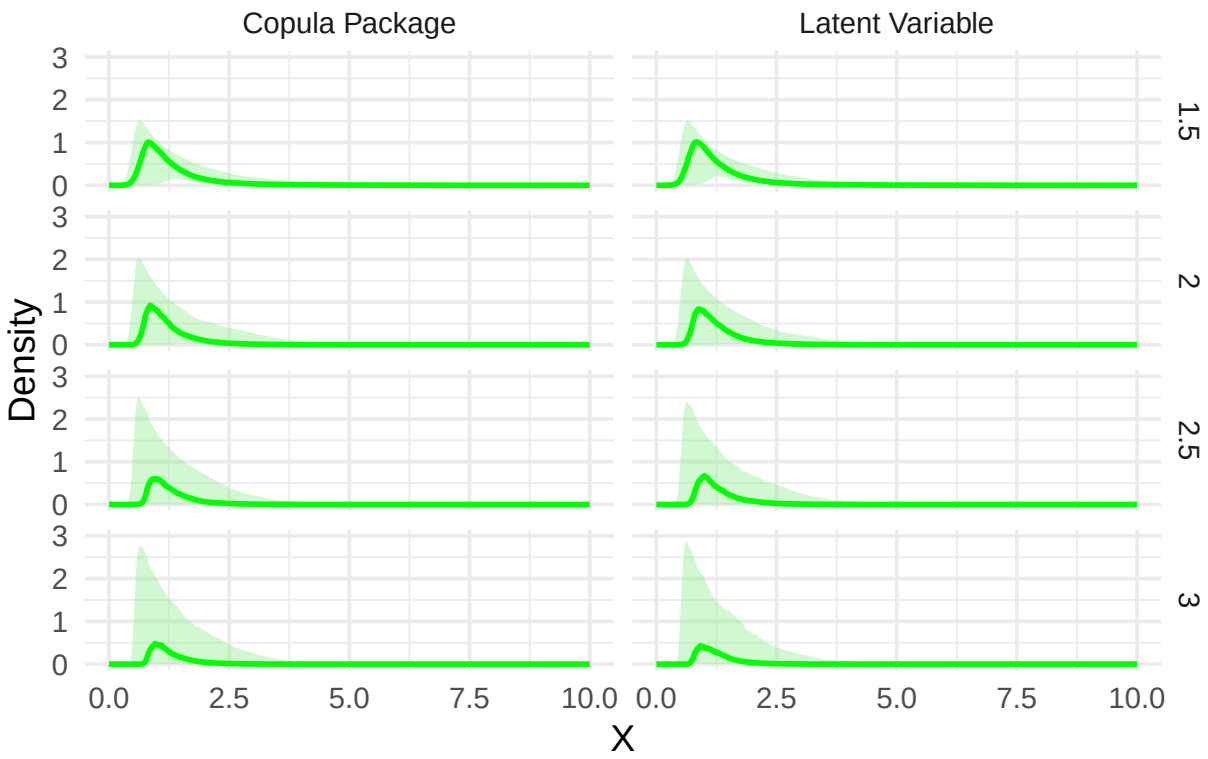
3.1 Simulation

Below we can see different realisations from a Gumbel-Frechet model and different theta parameters. It seems that different theta parameters lead to very similar realisations. When we perform estimation of the parameter, could this create non identifiability?


Gumbel – 30 replicates
Copula Package
Latent Variable
Density
1.5
2
2.5
3
0.00
0.25
0.50
0.75
1.00
U


GFrechet − 30 replicates
Copula Package
Latent Variable
Density
1.5
2
2.5
3
0.0
2.5
5.0
7.5
10.0
X


Gumbel − 1000 replicates − Median and 80%
Copula Package
Latent Variable
Density
1.5
2
2.5
3
0.00
0.25
0.50
0.75
1.00
U


GFrechet − 1000 replicates − Median and 80%
Copula Package
Latent Variable
Density
1.5
2
2.5
3
0.0
2.5
5.0
7.5
10.0
X

3.2 Likelihood

The aim is to show what is the shape of the likelihood of the model.

$$\text{Model:} \qquad (X_1, \dots, X_n) \sim H_n(\cdot) \qquad \text{with} \qquad H_n(x_1, \dots, x_n) = C_\theta\big(F(x_1), \dots, F(x_n)\big),$$

where F is the marginal CDF, f its density, and C_θ a copula (with copula density c_θ).

$$\begin{aligned}
& H_n(x_1, \dots, x_n) = C_\theta\big(u_1, \dots, u_n\big), \qquad u_i := F(x_i) \\
\implies \quad & h_n(x_1, \dots, x_n) \;=\; \frac{\partial^n}{\partial x_1 \cdots \partial x_n} H_n(x_1, \dots, x_n) \\
\overset{\text{(chain rule)}}{=} \quad & \frac{\partial^n}{\partial u_1 \cdots \partial u_n} C_\theta(u_1, \dots, u_n) \cdot \prod_{i=1}^{n} \frac{\partial u_i}{\partial x_i} \\
\Rightarrow \quad & \boxed{h_n(x_1, \dots, x_n) = c_\theta\big(F(x_1), \dots, F(x_n)\big) \prod_{i=1}^{n} f(x_i)}
\end{aligned} \tag{1}$$

Now the likelihood (viewed as a function of the copula parameter θ) for the observed vector $(x_1, \dots, x_n)$ is

$$L(\theta) \;=\; h_n(x_1, \dots, x_n; \theta) \;=\; c_\theta\big(F(x_1), \dots, F(x_n)\big) \prod_{i=1}^{n} f(x_i). \tag{2}$$

Taking logarithms yields the log-likelihood

$$\ell(\theta) := \log L(\theta) \;=\; \log c_\theta\big(F(x_1), \dots, F(x_n)\big) \;+\; \sum_{i=1}^{n} \log f(x_i). \tag{3}$$

Note that in the case of estimating only θ, the log-likelihood is constant in the second term w.r.t. the parameter. Note as well that for a too large sample size, the evaluation of the copula density is not feasible.

```
cop <- gumbelCopula(param = theta_true, dim = 1000)
U <- as.numeric(rCopula(1, cop))
dCopula(U, copula = cop)
```

```
## Error in polyG(lx[i.m[[meth]]], alpha = alpha, d = d, method = meth, log = log): d > 220 not yet sup
```

To illustrate what happens to the MLE, we can compute the profile log-likelihood for a given sample with the following implementation. A crucial factor seems to be the estimation of the margin to obtain the U-values used to estimate the paremeter from the log-likelihood.

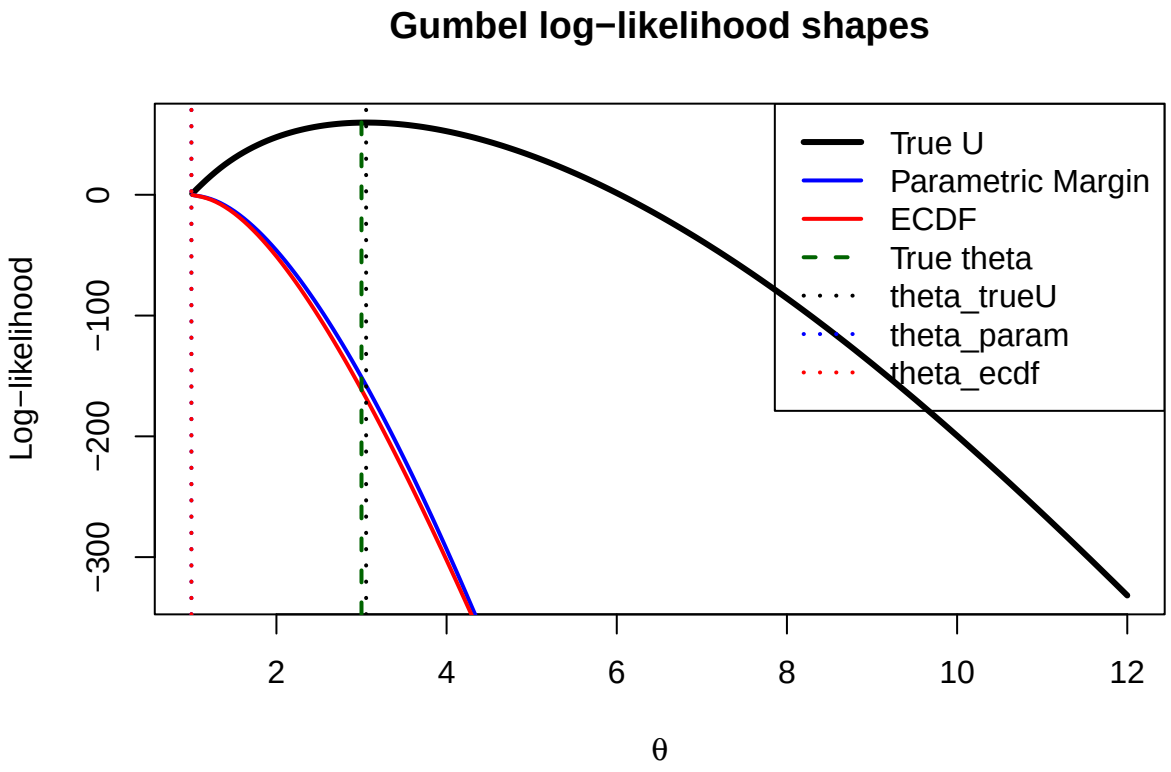
```
set.seed(123)

# ------------------------------------------------------------
# 1. TRUE parameter and dimension
# ------------------------------------------------------------
n <- 100
theta_true <- 3

# ------------------------------------------------------------
# 2. Simulate ONE 100-dimensional vector from Gumbel copula
# ------------------------------------------------------------
cop <- gumbelCopula(param = theta_true, dim = n)
```

```r
U <- as.numeric(rCopula(1, cop))   # true uniforms

# ------------------------------------------------------------
# 3. Generate X via lognormal margins
# ------------------------------------------------------------
mu <- 0
sigma <- 1
X <- qlnorm(U, meanlog = mu, sdlog = sigma)

# ------------------------------------------------------------
# 4A. Parametric margins
# ------------------------------------------------------------
mu_hat <- mean(log(X))
sigma_hat <- sd(log(X))
u_hat_param <- plnorm(X, meanlog = mu_hat, sdlog = sigma_hat)

# ------------------------------------------------------------
# 4B. ECDF pseudo-observations
# ------------------------------------------------------------
u_hat_ecdf <- rank(X) / (n + 1)

# ------------------------------------------------------------
# 5. Gumbel log-likelihood
# ------------------------------------------------------------
loglik_gumbel <- function(theta, u) {
  if (theta <= 1) return(-1e10)   # keep optimizer stable
  cop <- gumbelCopula(param = theta, dim = length(u))
  ll <- log(dCopula(u, copula = cop))
  return(ll)
}

# Negative log-likelihood for optimization
negLL <- function(theta, u) -loglik_gumbel(theta, u)

# ------------------------------------------------------------
# 6. Compute estimators via optim()
# ------------------------------------------------------------

# TRUE UNIFORMS
est_trueU <- optim(par = 5, fn = negLL, u = U, method="L-BFGS-B",
                   lower = 1.001, upper = 30)$par

# PARAMETRIC MARGINS
est_param <- optim(par = 5, fn = negLL, u = u_hat_param, method="L-BFGS-B",
                   lower = 1.001, upper = 30)$par

# ECDF PSEUDO-OBSERVATIONS
est_ecdf <- optim(par = 5, fn = negLL, u = u_hat_ecdf, method="L-BFGS-B",
                   lower = 1.001, upper = 30)$par

# ------------------------------------------------------------
# 7. Likelihood curves for comparison
# ------------------------------------------------------------
```

```r
theta_grid <- seq(1.01, 12, length.out = 200)

ll_trueU  <- sapply(theta_grid, loglik_gumbel, u = U)
```

```
## Loading required namespace: Rmpfr
```

```r
ll_param  <- sapply(theta_grid, loglik_gumbel, u = u_hat_param)
ll_ecdf   <- sapply(theta_grid, loglik_gumbel, u = u_hat_ecdf)
```




```
##
## ESTIMATED THETAS:
## theta_true = 3
## theta_trueU   = 3.054531
## theta_param   = 1.001
## theta_ecdf    = 1.001
```