Gumbel 1

Note: Unlike the Gaussian, Student-t, or the Skew-Normal copula, the Gumbel copula is evaluated directly on the $[0, 1]^2$ grid without the need for an inverse CDF transformation.

Install and load required packages

```
R_packages <- c("bookdown", "copula", "ggplot2", "gridExtra", "ks", "plot3D", "viridis")
options(repos = c(CRAN="http://cran.rstudio.com"))
if (!requireNamespace("librarian", quietly = TRUE)) install.packages("librarian")
librarian::shelf(R_packages)
```

Set parameters

```
a <- 2 # copula parameter
gsz <- 1000 # grid size
```

1 Gumbel Copula

Define the Gumbel copula

```
gumbel_cop <- gumbelCopula(param = a, dim = 2)
```

Create grid for u1, u2 in [0, 1]

```
u1_seq <- seq(0.01, 0.99, length.out = gsz)
u2_seq <- seq(0.01, 0.99, length.out = gsz)
grid <- expand.grid(u1 = u1_seq, u2 = u2_seq)
```

Compute copula density on $[0, 1]^2$ copula grid

```
copula_dens <- dCopula(as.matrix(grid), gumbel_cop)
z_mat_3d <- matrix(copula_dens, nrow = gsz, byrow = FALSE)
z_mat_3d <- pmin(z_mat_3d, 5) # Truncate for better visualization
```

Plot copula density on $[0, 1]^2$ copula grid 3D

```
persp3D(
  x = u1_seq,
  y = u2_seq,
  z = z_mat_3d,
  theta = 25,
  phi = 50,
  col = viridis::viridis(100),
  xlab = "u1", ylab = "u2", zlab = "density",
  main = "Bivariate Gumbel Copula Density (a = 2)",
  cex.main = 0.6, cex.lab = 0.6,
  colkey = list(length = 0.6, width = 0.6, cex.axis = 0.6)
)
```

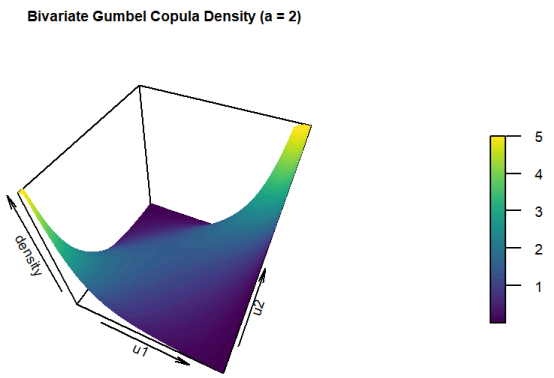

Figure 1.1:
Plot copula density on $[0, 1]^2$ copula grid contour

```
z_mat_2d <- matrix(copula_dens, nrow = gsz, byrow = TRUE)
z_mat_2d <- pmin(z_mat_2d, 5) # Truncate for better visualization

filled.contour(
  x = u1_seq, y = u2_seq, z = z_mat_2d,
  color.palette = viridis::viridis,
  xlab = "", ylab = "",
  plot.axes = {axis(1, at = seq(0.2, 0.8, by = 0.2), line = -0.05); axis(2, line = -3.2)},
  plot.title = title(main = "Copula Density: Contour Plot", cex.main = 0.8),
  key.title = title(main = "Density", cex.main = 0.6), asp = 1, frame.plot = FALSE,
  colkey = list(addlines = FALSE, length = 0.6, width = 0.5, cex.clab = 0.8)
)
```

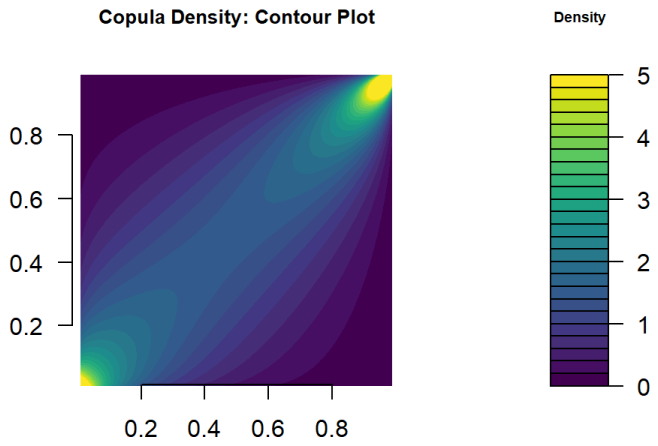

Figure 1.2:

2 Simulated Data

Simulate data

```
n <- 10000 # simulated sample size
set.seed(123456)
cs <- rCopula(n, gumbel_cop)
```

Scatter plot of simulated draws from copula

```
cs <- as.data.frame(cs)
colnames(cs) <- c("U1", "U2")

cplot <- ggplot(cs, aes(x=U1, y=U2)) +
  geom_point(alpha=0.7, size=0.5) + coord_fixed(ratio=1) +
  labs(title = "Gumbel 1", x = "", y = "") +
  theme_minimal() + theme(plot.margin = unit(c(0.1, 0.01, 0, 0), "in")) +
  theme(panel.grid = element_blank(), plot.title = element_text(hjust=0.5, size = 15),
        panel.border = element_rect(colour = "black", fill = NA, linewidth = 0.5))

plot_file = paste0("../graphs_sim/Gumbel1_true.png")
ggsave(plot_file, plot = cplot, width = 4, height = 4, units = "in", dpi = 300)

cplot
```

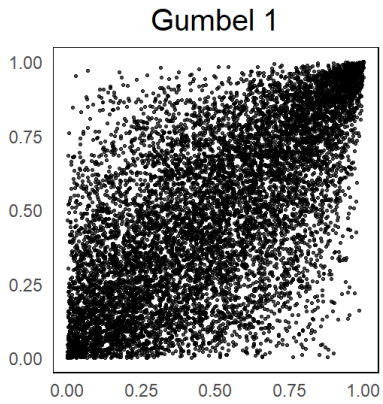

Figure 2.1:

Plot 3D kernel density estimate of the sample

```
# Kernel density estimation
H <- Hpi(cs)  # bandwidth matrix via plug-in method
kde_res <- kde(x = cs, H = H, compute.cont = FALSE)

# Extract grid and density matrix
x <- kde_res$eval.points[[1]]
y <- kde_res$eval.points[[2]]
z <- kde_res$estimate  # matrix of density values

persp3D(
  x = x,
  y = y,
  z = z,
  theta = 25,
  phi = 50,
  col = viridis::viridis(100),
  xlab = "u1", ylab = "u2", zlab = "density",
  main = "3D KDE of Gumbel Copula Sample",
  cex.main = 0.6, cex.lab = 0.6,
  colkey = list(length = 0.6, width = 0.6, cex.axis = 0.6)
)
```

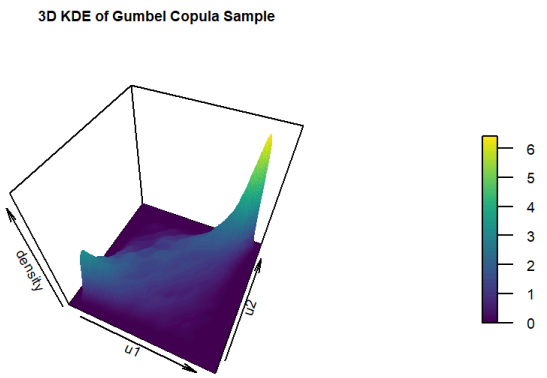

Figure 2.2:

3 Sparse Bernstein Copula

3.1 Set parameters

```
rname <- "Gumbel1"
true_dens <- copula_dens
sample_sizes <- c(100, 200, 300, 400, 500, 1000, 10000)
kmax <- 10 # number of batches for each sample size
```

3.2 Generate Data of Varying Sample Sizes for Fortran Estimation

```
set.seed(123456)

for (n in sample_sizes) {

  for (k in 1:kmax) {

    cs <- rCopula(n, gumbel_cop)

    # Write out for Fortran input
    write.table(cs, file = paste0("../data_sim/",rname,"_dat_",n,"_",k,".csv"),
                sep = ",", row.names = FALSE, col.names = FALSE)
  }
}
```

Run Fortran estimation of SBP copula

3.3 Plot Simulated Draws from SBP copula

```
n  <- 10000 # for sample size
k <- 1 # for data from batch k
csim_SBP <- read.table(file = paste0("../output_sim/",rname,"_csim_",n,"_",k,".out"),
                       header = FALSE)
```

Plot

```
csim_SBP_df <- as.data.frame(csim_SBP)
colnames(csim_SBP_df) <- c("U1", "U2")

csplot <- ggplot(csim_SBP_df, aes(x=U1, y=U2)) +
  geom_point(alpha=0.7, size=0.5) + coord_fixed(ratio=1) +
  labs(title = "SBP copula", x = "", y = "") +
  theme_minimal() + theme(plot.margin = unit(c(0.1, 0.01, 0, 0), "in")) +
  theme(panel.grid = element_blank(), plot.title = element_text(hjust=0.5, size = 15),
        panel.border = element_rect(colour = "black", fill = NA, linewidth = 0.5))

plot_file = paste0("../graphs_sim/",rname,"_SBP.png")
ggsave(plot_file, plot = csplot, width = 4, height = 4, units = "in", dpi = 300)

csplot
```

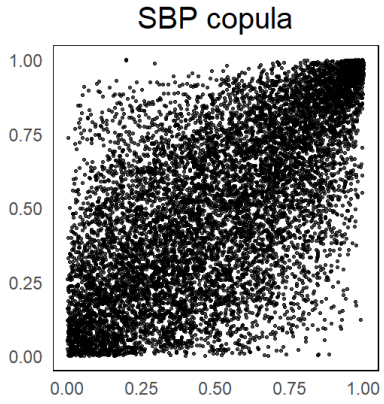

Figure 3.1:

3.4 Evaluate Kullback-Leibler Distance from True Density

```
KL <- numeric()
KLse <- numeric()

for (n in sample_sizes) {

  KL_k <- numeric()
  epsilon <- 1e-10 # Add small epsilon to avoid division by 0

  for (k in 1:kmax) {

    cdens_SBP <- read.table(file = paste0("../output_sim/",rname,"_cdens_",n,"_",k,".out"),
                            header = FALSE)
    cdens_SBP <- as.numeric(unlist(cdens_SBP))
    ratio <- (true_dens + epsilon)/(cdens_SBP + epsilon)
    KL_k_val <- log(ratio)*true_dens # Compute KL integrand
    KL_km <- mean(KL_k_val) # approximate integral over the grid
    KL_k <- c(KL_k, KL_km) # add new row

  }

  KL <- c(KL, mean(KL_k)) # add new row
  KLse <- c(KLse, sd(KL_k)/sqrt(length(KL_k))) # add new row

}

print(round(KL, 2))
```

```
## [1] 0.13 0.10 0.07 0.07 0.05 0.04 0.01
```

```
print(round(KLse, 2))
```

```
## [1] 0.01 0.01 0.00 0.00 0.00 0.00 0.00
```