mixture 1

Install and load required packages

```
R_packages <- c("bookdown", "copula", "sn", "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)
```

1 Mixture Copula

Mixture of Frank, Clayton, and rotated Gumbel (Archimedean copulas)

Define copula parameters and weights

```
theta <- c(-5, 15, 10) # Negative dependence in Frank
flip <- c(TRUE, FALSE) # rotates first variable in Gumbel
w <- c(0.4, 0.4, 0.2)
gsz <- 1000 # grid size
```

Define the mixture

```
cop1 <- frankCopula(param = theta[1])
cop2 <- claytonCopula(param = theta[2])
cop3 <- rotCopula(gumbelCopula(param = theta[3]), flip = flip)
```

Create a grid over [0,1]^2

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

Evaluate densities at each grid point

```
grid_mat <- as.matrix(grid)
dens1 <- dCopula(grid_mat, cop1)
dens2 <- dCopula(grid_mat, cop2)
dens3 <- dCopula(grid_mat, cop3)
```

Mixture density

```
mix_dens <- w[1]*dens1 + w[2]*dens2 + w[3]*dens3
```

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

```
z_mat_3d <- matrix(mix_dens, nrow = gsz, ncol = gsz, byrow = FALSE)
z_mat_3d <- pmin(z_mat_3d, 5) # Truncate for better visualization

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 = "Copula Density",
  cex.main = 0.6, cex.lab = 0.6,
  colkey = list(length = 0.6, width = 0.6, cex.axis = 0.6)
)
```

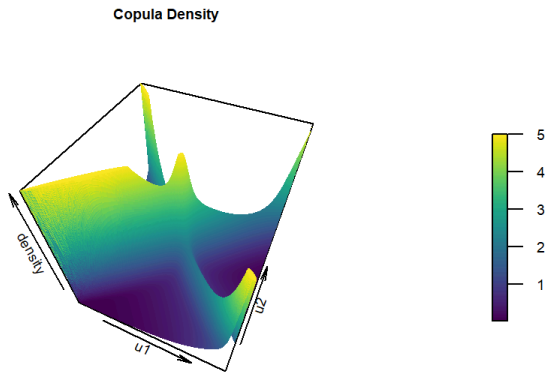

Figure 1.1:
Plot copula density on $[0, 1]^2$ copula grid contour

```
z_mat_2d <- matrix(mix_dens, nrow = gsz, byrow = TRUE)
#z_mat_2d[is.infinite(z_mat_2d)] <- NaN # Replace INF entries with NaN
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)
)
```

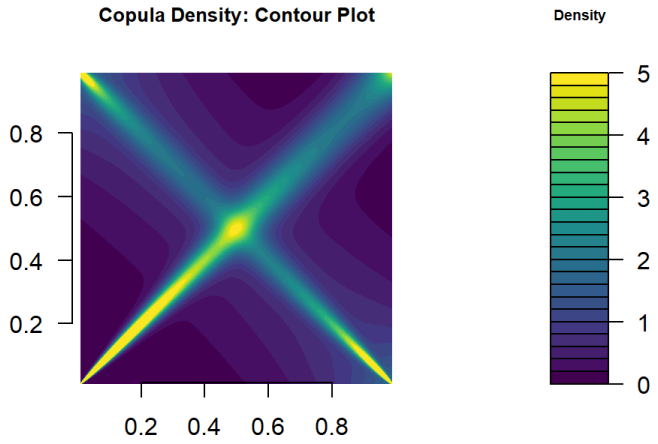

Figure 1.2:

2 Sample Data from Mixture Copula

Sample from the copula mixture

```
n  <- 10000 # Sample size
n1 <- round(n*w[1])
n2 <- round(n*w[2])
n3 <- n - (n1 + n2)

set.seed(123456)
u1 <- rCopula(n1, cop1) # Sample from each copula
u2 <- rCopula(n2, cop2)
u3 <- rCopula(n3, cop3)

u_mix <- rbind(u1, u2, u3) # Combine the samples to form the mixture
```

Scatter plot

```
c_df <- as.data.frame(u_mix)
colnames(c_df) <- c("X","Y")

cplot <- ggplot(c_df, aes(x=X, y=Y)) +
  geom_point(alpha=0.7, size=0.5) + coord_fixed(ratio=1) +
  labs(title = "Mixture 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/mix1_true.png")
ggsave(plot_file, plot = cplot, width = 4, height = 4, units = "in", dpi = 300)

cplot
```

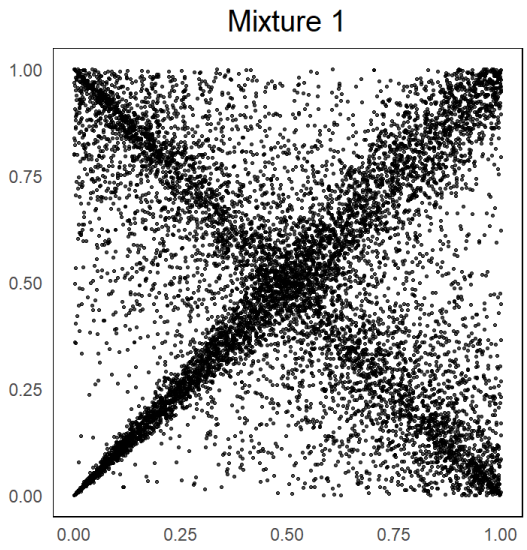

Figure 2.1:

Plot 3D kernel density estimate of simulated data from copula

```
# Kernel density estimation
H <- Hpi(u_mix)  # bandwidth matrix via plug-in method
kde_res <- kde(x = u_mix, 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

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

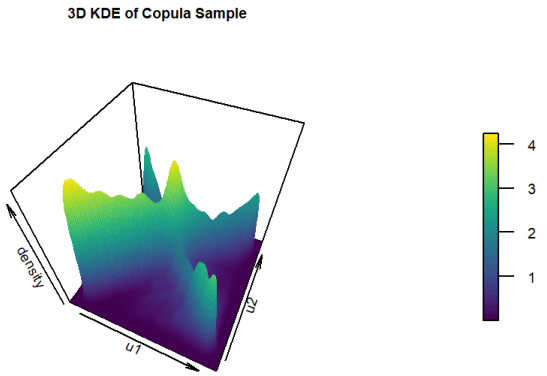

Figure 2.2:

Contour plot

```
filled.contour(
  x = x, y = y, z = t(z),
  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 = "KDE Contour Plot of Copula Sample", 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)
)
```

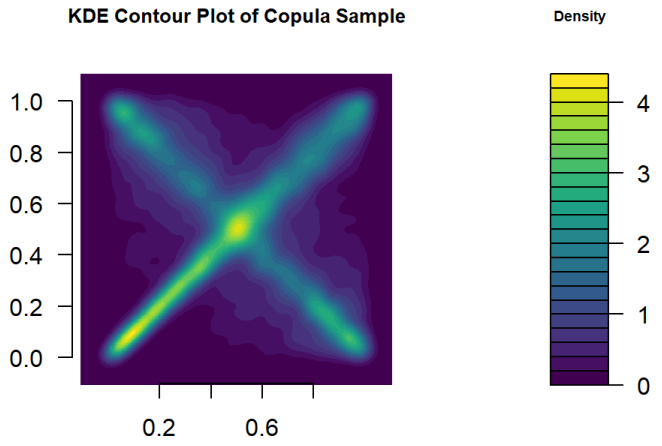

Figure 2.3:

3 Corresponding Bivariate Density

Apply inverse skew-normal marginals

```
qF1 <- function(u) qsn(u, xi = -2, omega = 1, alpha = 8)   # Left-shifted, right-skewed
qF2 <- function(u) qsn(u, xi = 2, omega = 1, alpha = -8)   # Right-shifted, left-skewed

x <- qF1(u_mix[,1])
y <- qF2(u_mix[,2])

dens_draws <- as.data.frame(cbind(x,y))
```

Scatter plot

```
data_df <- as.data.frame(dens_draws)
colnames(data_df) <- c("X","Y")

ggplot(data_df, aes(x=X, y=Y)) +
  geom_point(alpha=0.5, size=0.5) +
  labs(title = "Mixture Density Sample", x = "x", y = "y") +
  theme_minimal() + xlim(-2.5,1) +
  theme(panel.grid = element_blank(),
        panel.border = element_rect(colour = "black", fill = NA, linewidth = 0.5))
```

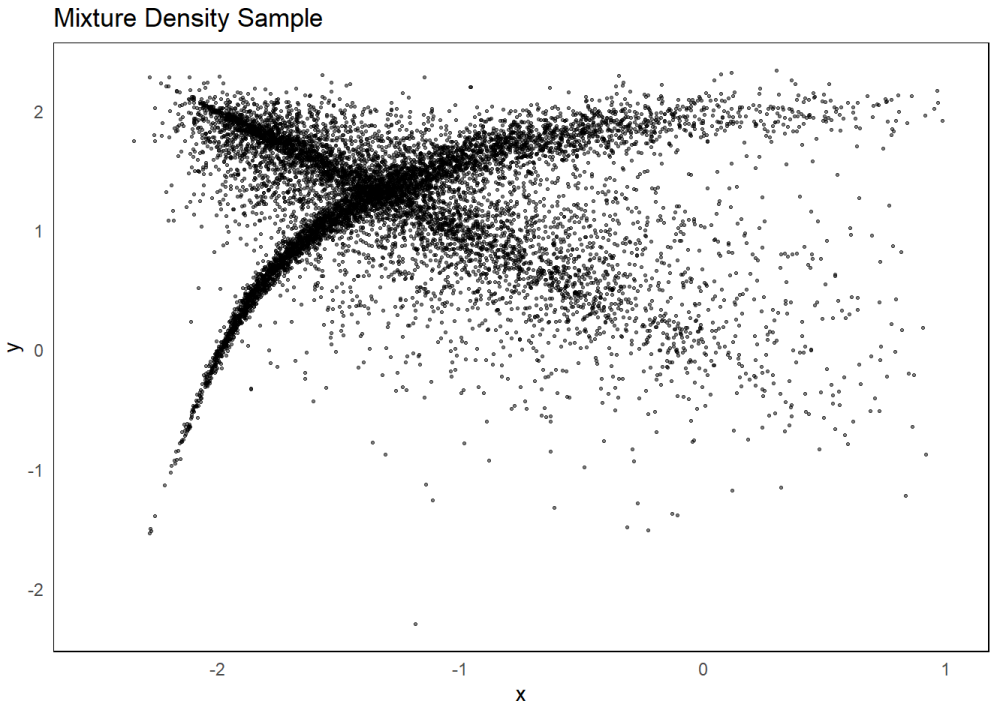

Figure 3.1:
Plot 3D kernel density estimate of simulated data

```
# Kernel density estimation
subset_draws <- dens_draws[dens_draws$x < 0 & dens_draws$y > -1, ] # focus on modes
H <- Hpi(subset_draws)  # bandwidth matrix via plug-in method
kde_res <- kde(x = subset_draws, 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

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

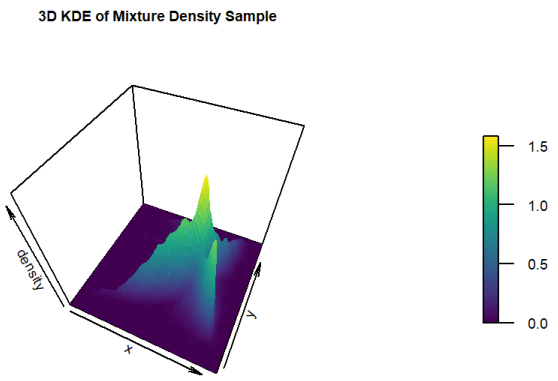

Figure 3.2:
Plot contours

```
x <- kde_res$eval.points[[1]]
y <- kde_res$eval.points[[2]]
z <- kde_res$estimate  # z is a matrix of density values, dim = length(x) x length(y)

# Plotly contour plot (2D projection of density)
filled.contour(
  x = x, y = y, z = t(z),
  color.palette = viridis::viridis,
  xlab = "x", ylab = "y",
  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 = "Contour Plot of KDE of Mixture Density Sample", 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)
)
```

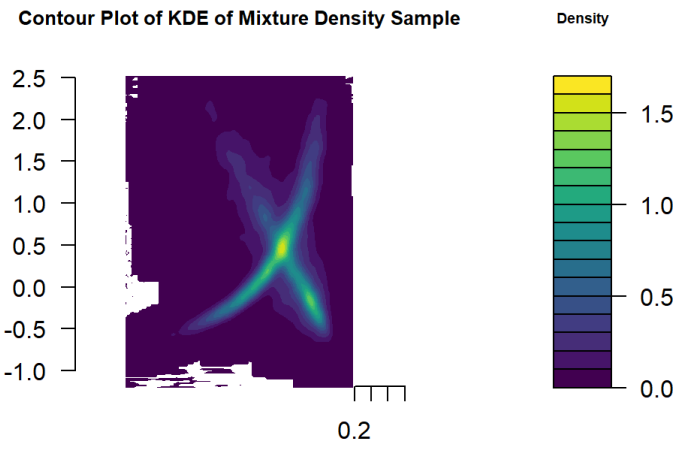

Figure 3.3:

4 Sparse Bernstein Copula

4.1 Set parameters

```
rname <- "mix1"
true_dens <- mix_dens
sample_sizes <- c(100, 200, 300, 400, 500, 1000, 10000)
kmax <- 10 # number of batches for each sample size
```

4.2 Generate Data of Varying Sample Sizes for Fortran Estimation

```
set.seed(123456)

for (n in sample_sizes) {

  n1 <- round(n*w[1])
  n2 <- round(n*w[2])
  n3 <- n - (n1 + n2)

  for (k in 1:kmax) {

    u1 <- rCopula(n1, cop1) # Sample from each copula
    u2 <- rCopula(n2, cop2)
    u3 <- rCopula(n3, cop3)

    u_mix <- rbind(u1, u2, u3) # Combine the samples to form the mixture

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

Run Fortran estimation of SBP copula

4.3 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
```

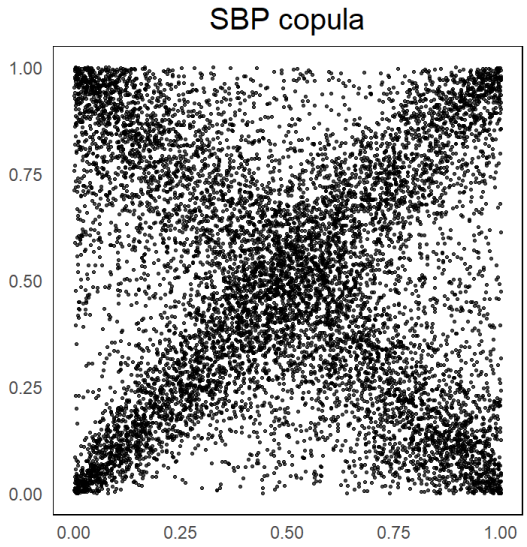

Figure 4.1:

4.4 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] 0.35 0.29 0.28 0.25 0.23 0.21 0.17
```

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

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