Gaussian 1

Install and load required packages

```
R_packages <- c("bookdown", "mvtnorm", "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

```
rho <- 0.5 # correlation parameter
Omega <- matrix(c(1, rho, rho, 1), 2, 2)  # Covariance matrix
mu <- c(0, 0) # Location
gsz <- 1000  # grid size
```

1 Gaussian Density

Plot 3D density

```
# Define grid
x_seq <- seq(-3, 3, length.out = gsz)
y_seq <- seq(-3, 3, length.out = gsz)
grid_xy <- expand.grid(x = x_seq, y = y_seq)

# Compute bivariate Gaussian density
dens <- dmvnorm(as.matrix(grid_xy), mean = mu, sigma = Omega)

# Convert to matrix for plotting
z_mat_3d <- matrix(dens, nrow = length(y_seq), byrow = FALSE)

# Plot 3D density
persp3D(
  x = x_seq,
  y = y_seq,
  z = z_mat_3d,
  theta = 25,
  phi = 50,
  col = viridis::viridis(50),
  xlab = "u1", ylab = "u2", zlab = "density",
  main = "Bivariate Gaussian Density",
  cex.main = 0.6, cex.lab = 0.6,
  colkey = list(length = 0.6, width = 0.6, cex.axis = 0.6)
)
```

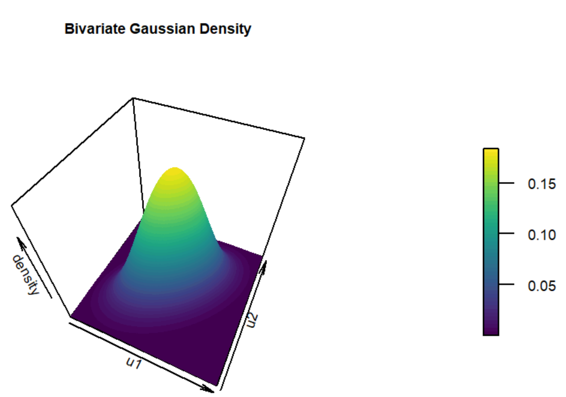

Figure 1.1:
Plot 2D density contour plot

```
z_mat_2d <- matrix(dens, nrow = length(y_seq), byrow = TRUE)

filled.contour(
  x = x_seq, y = y_seq, z = z_mat_2d,
  color.palette = viridis::viridis,
  xlab = "", ylab = "",
  plot.axes = {axis(1, at = seq(-3, 3, by = 1), line = -0.05); axis(2, line = -3.2)},
  plot.title = title(main = "Bivariate Gaussian 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)
)
```

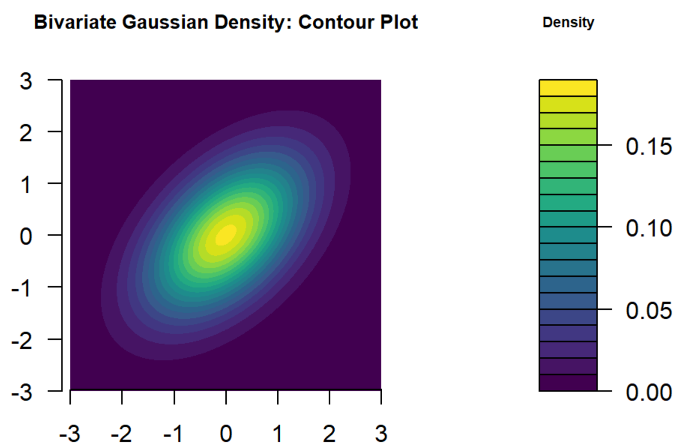

Figure 1.2:

2 Copula Density

2.1 Start with xy-grid and convert to $[0,1]^2$ copula grid

Evaluate copula density

```
# Joint density
f_joint <- dmvnorm(as.matrix(grid_xy), mean = mu, sigma = Omega)

# Marginal densities
f1 <- dnorm(grid_xy$x, mean = mu[1], sd = sqrt(Omega[1,1]))
f2 <- dnorm(grid_xy$y, mean = mu[2], sd = sqrt(Omega[2,2]))

# Copula density
copula_dens <- f_joint/(f1*f2)
```

Plot marginal densities

```
cf1 <- dnorm(x_seq, mean = mu[1], sd = sqrt(Omega[1,1]))
cc1 <- as.data.frame(cbind(x_seq,cf1))

cf2 <- dnorm(y_seq, mean = mu[2], sd = sqrt(Omega[2,2]))
cc2 <- as.data.frame(cbind(y_seq,cf2))

m1 <- ggplot(cc1, aes(x=x_seq, y=cf1)) +
  geom_line(linewidth=0.5) + theme_minimal()  +
  labs(title = "Marginal pdf over X", x = "x", y = "f1(x)")

m2 <- ggplot(cc2, aes(x=y_seq, y=cf2)) +
  geom_line(linewidth=0.5) + theme_minimal()  +
  labs(title = "Marginal pdf over Y", x = "y", y = "f2(y)")

gridExtra::grid.arrange(m1, m2, ncol=2)
```

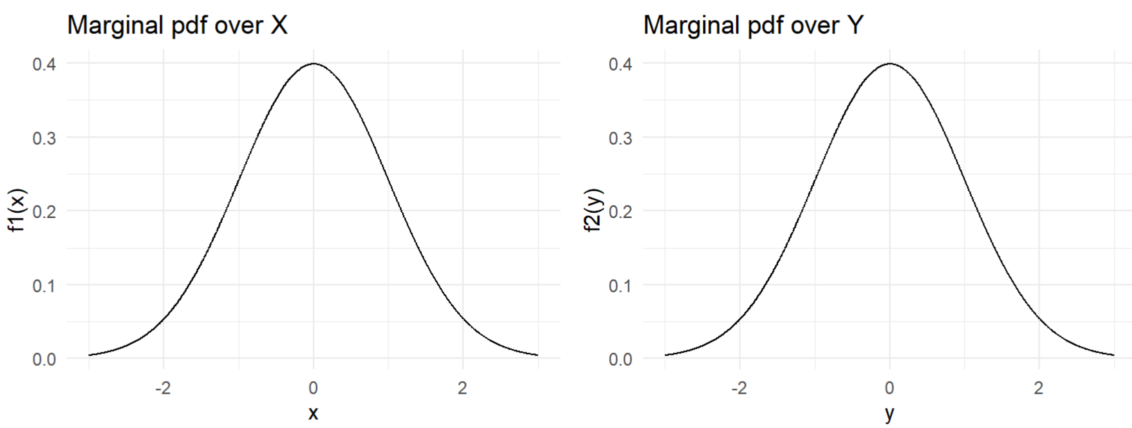

Figure 2.1:

Plot marginal distributions

```
cdf1 <- pnorm(x_seq, mean = mu[1], sd = sqrt(Omega[1,1]))
cdf2 <- pnorm(y_seq, mean = mu[2], sd = sqrt(Omega[2,2]))

cdf_df1 <- data.frame(x = x_seq, cdf = cdf1)
cdf_df2 <- data.frame(y = y_seq, cdf = cdf2)

cdf_plot1 <- ggplot(cdf_df1, aes(x = x, y = cdf)) +
  geom_line(linewidth=0.5, color = "blue") + theme_minimal() +
  labs(title = "Marginal CDF over X", x = "x", y = "F1(x)")

cdf_plot2 <- ggplot(cdf_df2, aes(x = y, y = cdf)) +
  geom_line(linewidth=0.5, color = "blue") +  theme_minimal() +
  labs(title = "Marginal CDF over Y", x = "y", y = "F2(y)")

gridExtra::grid.arrange(cdf_plot1, cdf_plot2, ncol = 2)
```

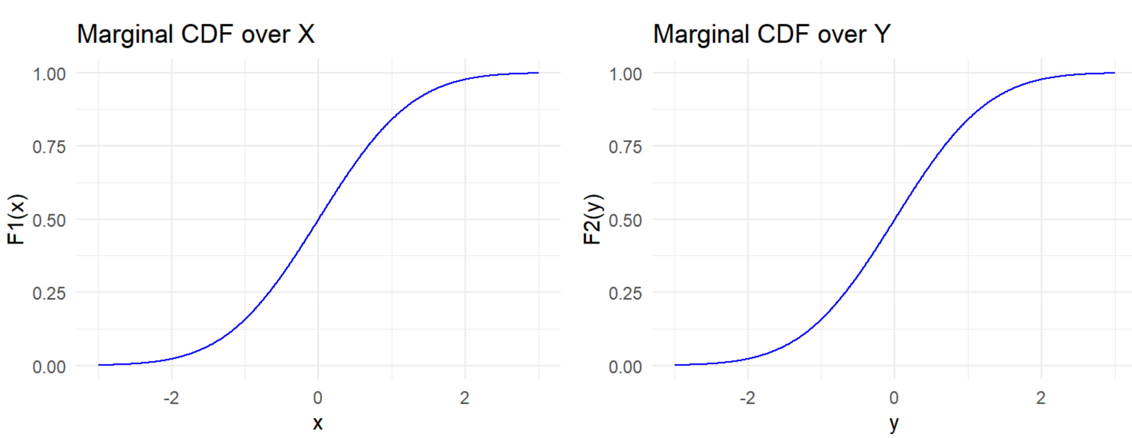

Figure 2.2:

Convert xy-grid to corresponding $[0,1]^2$ copula grid

```
u1 <- pnorm(grid_xy$x, mean = mu[1], sd = sqrt(Omega[1, 1]))
u2 <- pnorm(grid_xy$y, mean = mu[2], sd = sqrt(Omega[2, 2]))
grid_copula <- data.frame(u1 = u1, u2 = u2)

u1_seq <- u1[1:gsz]
u2_seq <- u2[seq(1, gsz^2, by = gsz)]
```

Plot the $[0,1]^2$ copula grid

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

ggplot(gc_df, aes(x=U1, y=U2)) +
  geom_point(alpha=0.1, size=0.001) + coord_fixed(ratio=1) +
  labs(title = "Inverted [0,1]x[0,1] grid",
       x = "U1", y = "U2") +
  theme_minimal() +
  theme(panel.grid = element_blank(),
        panel.border = element_rect(colour = "black", fill = NA, linewidth = 0.5))
```

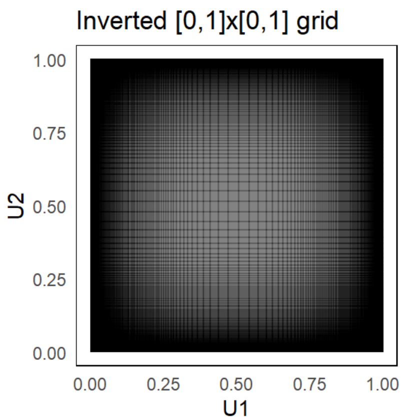

Figure 2.3:

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

```
z_mat_3d <- matrix(copula_dens, nrow = length(x_seq), byrow = FALSE)
z_mat_3d[is.infinite(z_mat_3d)] <- NaN # Replace INF entries with NaN (the plot then ignores them)
z_mat_3d <- pmin(z_mat_3d, 3) # Truncate to see the contours change

persp3D(
  x = u1_seq,
  y = u2_seq,
  z = z_mat_3d,
  theta = 25,
  phi = 50,
  col = viridis::viridis(50),
  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)
)
```

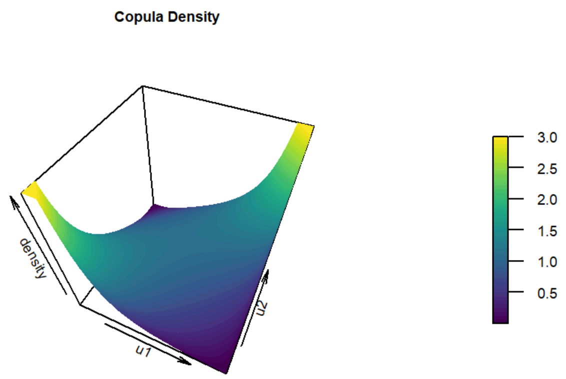

Figure 2.4:
Plot copula density on $[0,1]^2$ copula grid contour

```
z_mat_2d <- matrix(copula_dens, nrow = length(y_seq), byrow = TRUE)
z_mat_2d[is.infinite(z_mat_2d)] <- NaN # Replace INF entries with NaN
z_mat_2d <- pmin(z_mat_2d, 3) # Truncate to see the contours change

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)
)
```

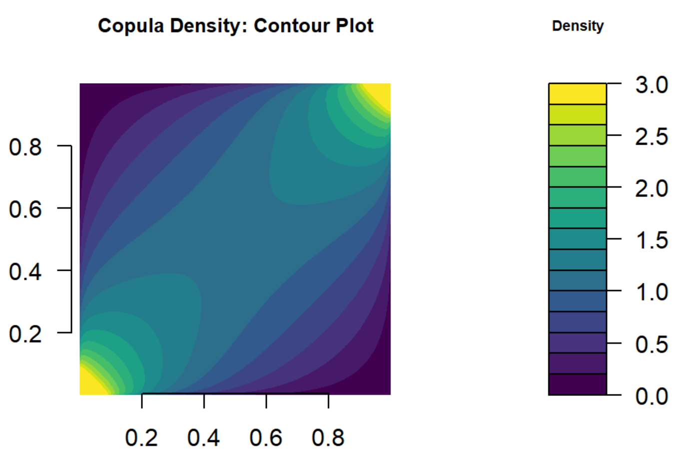

Figure 2.5:

2.2 Start with $[0,1]^2$ copula grid

Create $[0,1]^2$ copula grid

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

# Inverse CDF with solver = "RFB" to xy-grid
xq <- qnorm(grid$u1, mean = 0, sd = 1)
yq <- qnorm(grid$u2, mean = 0, sd = 1)
```

Plot inverted xy-grid

```
gr_qsn <- cbind(xq, yq)
colnames(gr_qsn) <- c("x", "y")

ggplot(gr_qsn, aes(x=x, y=y)) +
  geom_point(alpha=0.1, size=0.001) + coord_fixed(ratio=1) +
  labs(title = "Inverted xy-grid",
       x = "x", y = "y") +
  theme_minimal() +
  theme(panel.grid = element_blank(),
        panel.border = element_rect(colour = "black", fill = NA, linewidth = 0.5))
```

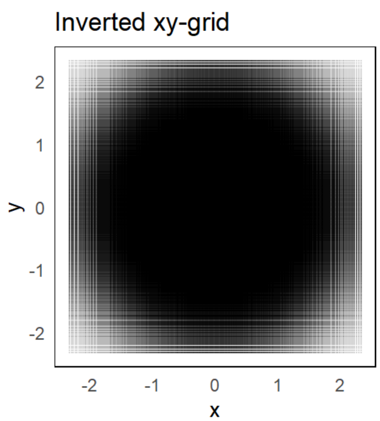

Figure 2.6:

Evaluate copula density

```
# Joint density and marginals
fq_joint <- dmvnorm(cbind(xq, yq), mean = mu, sigma = Omega)
fq1 <- dnorm(xq, mean = 0, sd = 1)
fq2 <- dnorm(yq, mean = 0, sd = 1)

# Copula density
cq_dens <- fq_joint/(fq1*fq2)
```

Plot copula density on copula grid 3D

```
z_mat_3d <- matrix(cq_dens, nrow = length(x_seq), byrow = TRUE)
z_mat_3d[is.infinite(z_mat_3d)] <- NaN # Replace INF entries with NaN
z_mat_3d <- pmin(z_mat_3d, 3) # Truncate for better visualization

# Plot
persp3D(
  x = u_seq,
  y = u_seq,
  z = z_mat_3d,
  theta = 25,
  phi = 50,
  col = viridis::viridis(50),
  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)
)
```

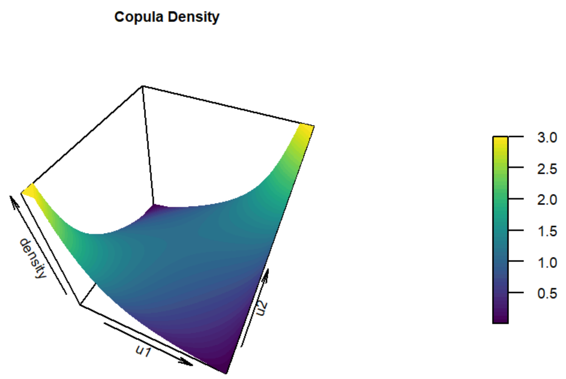

Figure 2.7:
Contour plot

```
z_mat_2d <- matrix(cq_dens, nrow = length(y_seq), byrow = TRUE)
z_mat_2d[is.infinite(z_mat_2d)] <- NaN # Replace INF entries with NaN
z_mat_2d <- pmin(z_mat_2d, 3) # Truncate for better visualization

filled.contour(
  x = u_seq, y = u_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)
)
```

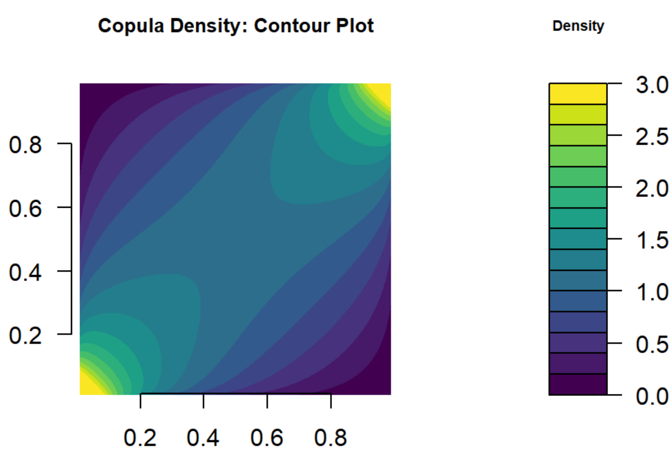

Figure 2.8:

3 Simulated Data

Simulate data

```
n <- 10000 # simulated sample size
set.seed(123456)
data <- rmvnorm(n, mean=mu, sigma=Omega)
```

Scatter plot

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

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

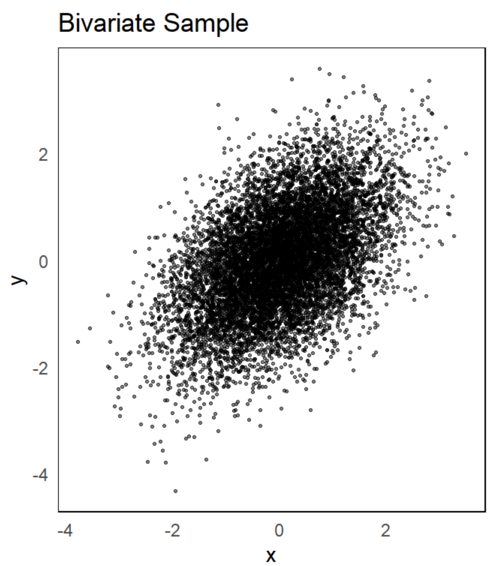

Figure 3.1:

3.1 Use theoretical CDF transformation to [0,1]

```
u1_th <- pnorm(data[,1], mean = mu[1], sd = sqrt(Omega[1,1]))
u2_th <- pnorm(data[,2], mean = mu[2], sd = sqrt(Omega[2,2]))

cs_th <- as.data.frame(cbind(u1_th, u2_th))
colnames(cs_th) <- c("U1", "U2")
```

Plot resulting marginal cdfs

```
cdf_plot1 <- ggplot(data.frame(x = data[,1], u = u1_th), aes(x = x, y = u)) +
  geom_line(linewidth=0.5, color = "blue") + theme_minimal() +
  labs(title = "Marginal CDF over X", x = "x", y = "F1(x)")

cdf_plot2 <- ggplot(data.frame(x = data[,2], u = u2_th), aes(x = x, y = u)) +
  geom_line(linewidth=0.5, color = "blue") +  theme_minimal() +
  labs(title = "Marginal CDF over Y", x = "y", y = "F2(y)")

gridExtra::grid.arrange(cdf_plot1, cdf_plot2, ncol = 2)
```

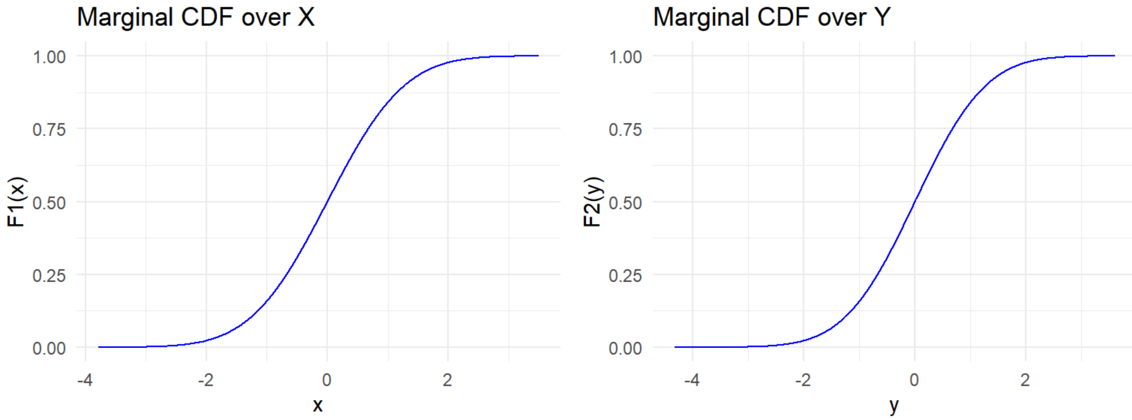

Figure 3.2:

Scatter plot of simulated draws from copula

```
cplot <- ggplot(cs_th, aes(x=U1, y=U2)) +
  geom_point(alpha=0.7, size=0.5) + coord_fixed(ratio=1) +
  labs(title = "Gaussian 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/Gaussian1_true.png")
ggsave(plot_file, plot = cplot, width = 4, height = 4, units = "in", dpi = 300)

cplot
```

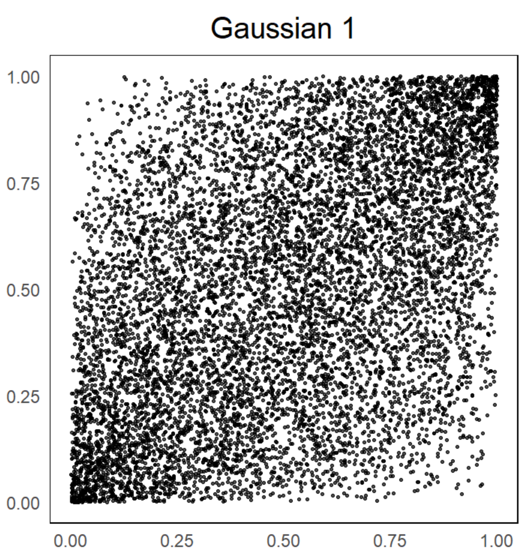

Figure 3.3:

Plot 3D kernel density estimate of simulated data

```
# Kernel density estimation
H <- Hpi(cs_th)  # bandwidth matrix via plug-in method
kde_res <- kde(x = cs_th, 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 = z,
  theta = 25,
  phi = 50,
  col = viridis::viridis(50),
  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)
)
```

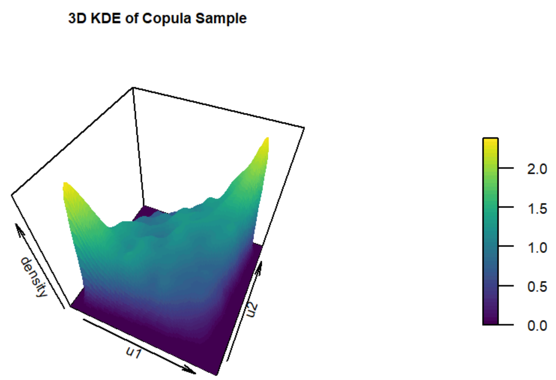

Figure 3.4:

3.2 Use RANK estimate of empirical CDF for transformation to [0,1]

```
u1_EDF <- rank(data[,1])/(n+1)
u2_EDF <- rank(data[,2])/(n+1)

cs_EDF <- cbind(u1_EDF, u2_EDF)
colnames(cs_EDF) <- c("U1", "U2")
```

Plot the resulting cdfs

```
cdf_plot1 <- ggplot(data.frame(x = data[,1], u = u1_EDF), aes(x = x, y = u)) +
  geom_line(linewidth=0.5, color = "red") + theme_minimal() +
  labs(title = "Marginal CDF over X", x = "x", y = "F1(x)")

cdf_plot2 <- ggplot(data.frame(x = data[,2], u = u2_EDF), aes(x = x, y = u)) +
  geom_line(linewidth=0.5, color = "red") +  theme_minimal() +
  labs(title = "Marginal CDF over Y", x = "y", y = "F2(y)")

gridExtra::grid.arrange(cdf_plot1, cdf_plot2, ncol = 2)
```

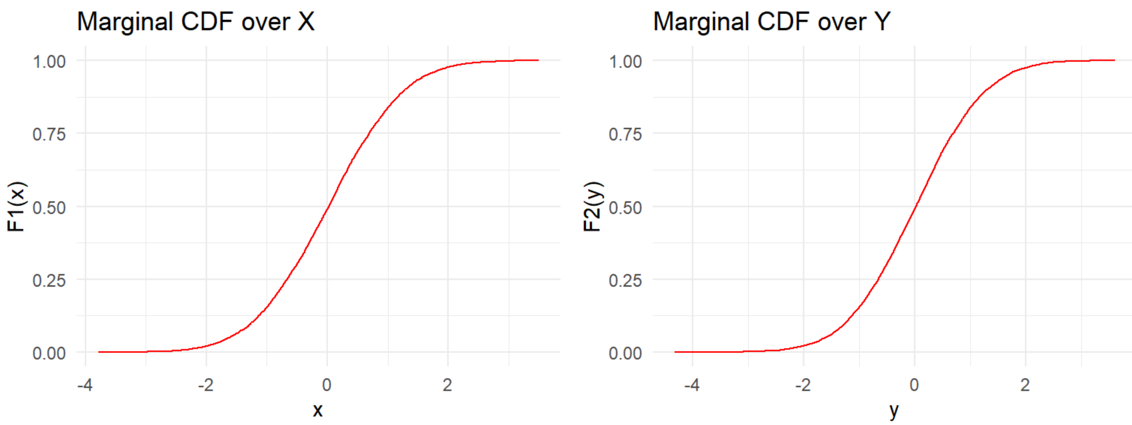

Figure 3.5:
Scatter plot

```
ggplot(cs_EDF, aes(x=U1, y=U2)) +
  geom_point(alpha=0.5, size=0.5) + coord_fixed(ratio=1) +
  labs(title = "Copula Sample with RANK estimate\nof empirical CDFs",
       x = expression(U[1]), y = expression(U[2])) +
  theme_minimal() +
  theme(panel.grid = element_blank(),
        panel.border = element_rect(colour = "black", fill = NA, linewidth = 0.5),
        plot.title = element_text(size = 10, hjust = 0.5))
```

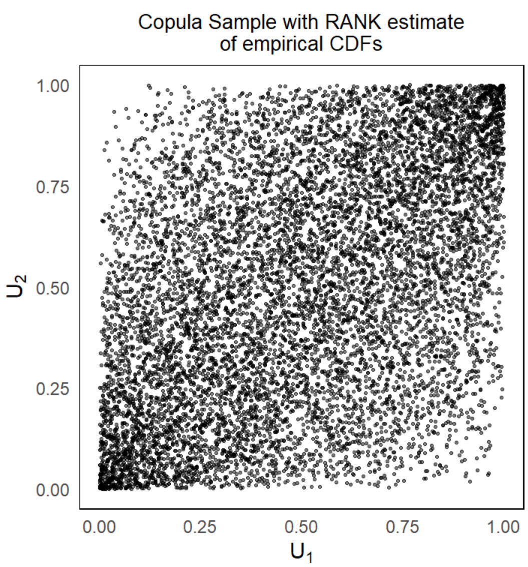

Figure 3.6:

Plot 3D kernel density estimate of simulated data

```
# Kernel density estimation
H <- Hpi(cs_EDF)  # bandwidth matrix via plug-in method
kde_res <- kde(x = cs_EDF, 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 = z,
  theta = 25,
  phi = 50,
  col = viridis::viridis(50),
  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)
)
```

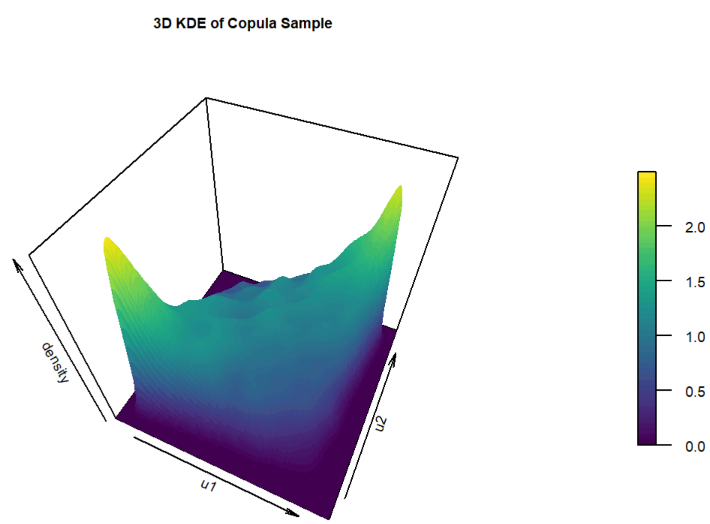

Figure 3.7:

4 Sparse Bernstein Copula

4.1 Set parameters

```
rname <- "Gaussian1"
true_dens <- cq_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) {

  for (k in 1:kmax) {

    data <- rmvnorm(n, mean=mu, sigma=Omega)
    u1_th <- pnorm(data[,1], mean = mu[1], sd = sqrt(Omega[1,1]))
    u2_th <- pnorm(data[,2], mean = mu[2], sd = sqrt(Omega[2,2]))
    cs_th <- as.data.frame(cbind(u1_th, u2_th))

    # Write out for Fortran input
    write.table(cs_th, 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
```

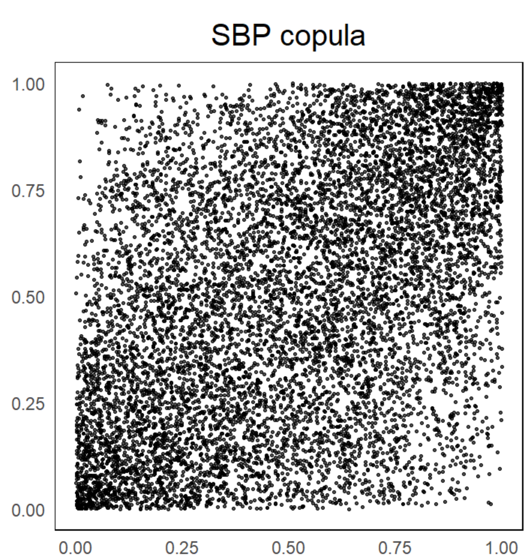

Figure 4.1:

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] 0.24 0.08 0.06 0.05 0.04 0.03 0.01
```

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

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