2DGaussian

August 7, 2024

[1]:
```
### Plot the bivariate Gaussian from the lecture as a mesh plot
### also as contrours of constant probability density
### You can define a grid of (x,y) values to calculate the joint PDF

# Define the mean vector and covariance matrix for the standard bivariate␣
 ↪normal distribution given
mu <- c(0, 0)  # Mean vector with means of X and Y both equal to 0
C <- matrix(c(1, 0, 0, 1), nrow = 2)  # Covariance matrix with variances 1 and␣
 ↪no covariance (identity matrix)
```

[2]:
```
# Create a grid of (x, y) values
x <- seq(-3, 3, length.out = 40)  # Sequence of x values from -3 to 3
y <- seq(-3, 3, length.out = 40)  # Sequence of y values from -3 to 3
grid <- expand.grid(x = x, y = y)  # Create a data frame containing all␣
 ↪combinations of x and y values
```

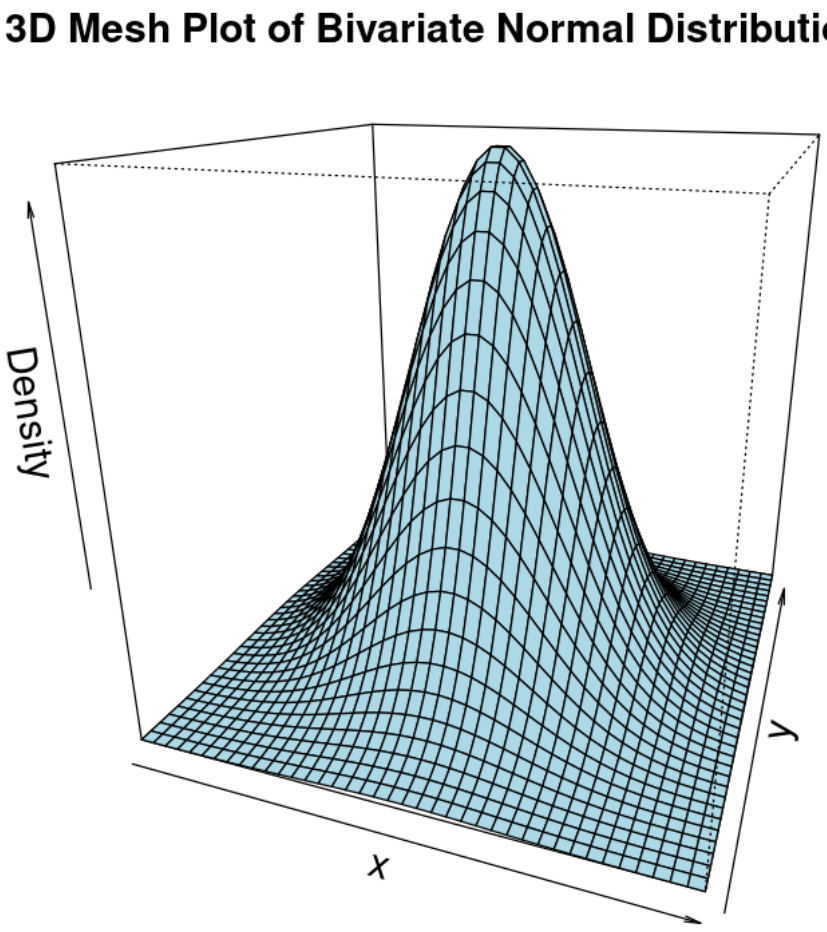
[3]:
```
# Define a function to calculate the density of the bivariate normal␣
 ↪distribution
bivariate_normal_density <- function(x, y, mu, C) {
  # Extract the mean and covariance values
  mu_x <- mu[1]
  mu_y <- mu[2]
  sigma_x <- sqrt(C[1, 1])
  sigma_y <- sqrt(C[2, 2])
  rho <- C[1, 2] / (sigma_x * sigma_y)

  # Calculate the density
  z <- (x - mu_x)^2 / sigma_x^2 + (y - mu_y)^2 / sigma_y^2 - 2 * rho * (x -␣
 ↪mu_x) * (y - mu_y) / (sigma_x * sigma_y)
  density <- exp(-z / (2 * (1 - rho^2))) / (2 * pi * sigma_x * sigma_y * sqrt(1␣
 ↪- rho^2))
  return(density)
}

# Calculate the density values for the grid points we defined above
grid$z <- mapply(bivariate_normal_density, grid$x, grid$y, MoreArgs = list(mu =␣
 ↪mu, C = C))
```

[4]:
```
# Convert the grid to a matrix format for 3D plotting
z_matrix <- matrix(grid$z, nrow = length(x), ncol = length(y))

# Plot the 3D mesh plot
persp(x=x, y=y, z=z_matrix, theta = 20, phi = 20, expand = 1.0, col =␣
↪"lightblue",
      xlab = "x", ylab = "y", zlab = "Density", main = "3D Mesh Plot of␣
↪Bivariate Normal Distribution",
      cex.lab = 1.5, cex.axis = 1.2, cex.main = 1.5)  # Increase font sizes
```

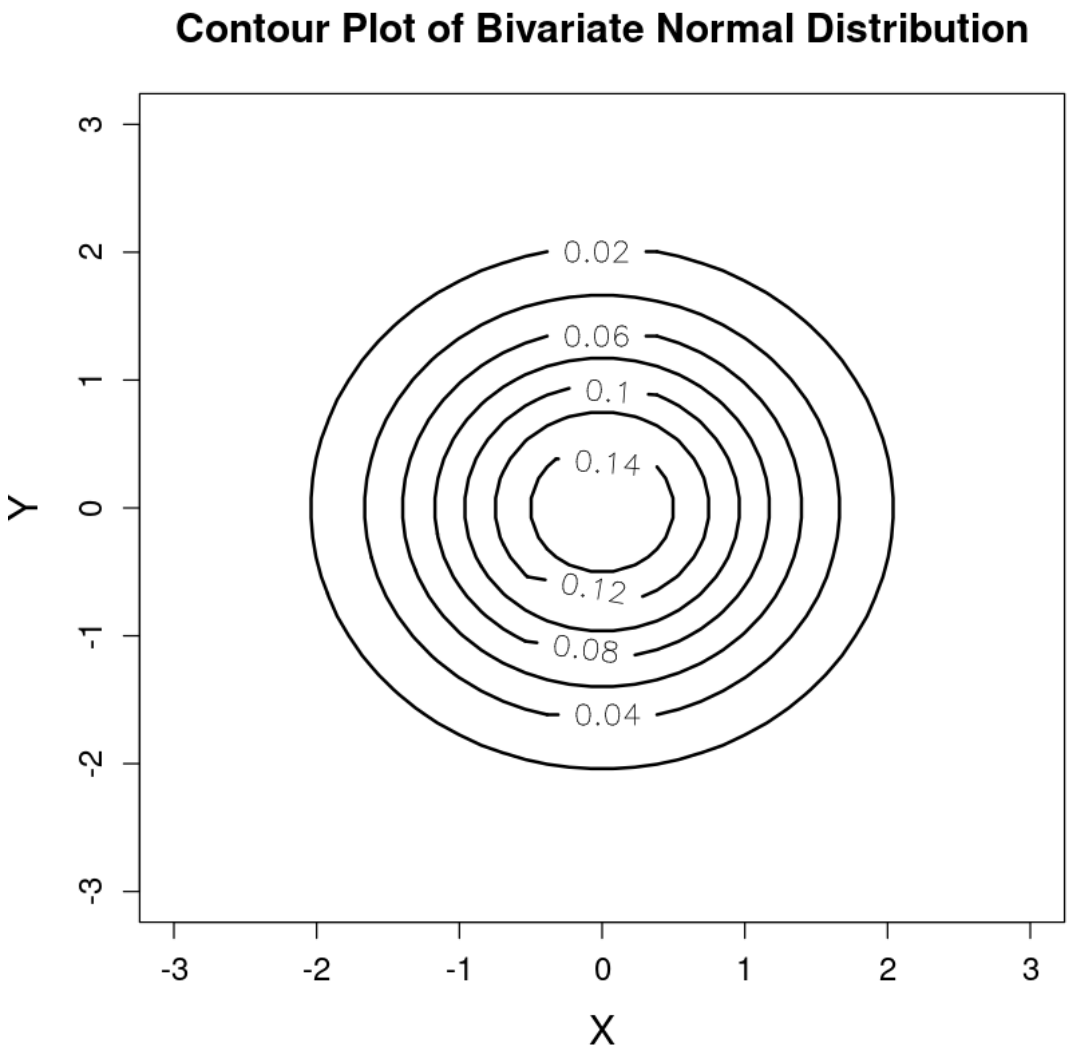


[5]:
```
# Plot the contour plot
# Uncomment the command below to save the contour plot as a PDF file
```

```r
# pdf("contour_bivariate_normal_distribution.pdf", width = 6, height = 6)

contour(x, y, z_matrix, xlab = "X", ylab = "Y", main = "Contour Plot of␣
 ↪Bivariate Normal Distribution",
        lwd = 2,  # Increase line thickness
        cex.lab = 1.5, cex.axis = 1.2, cex.main = 1.5, labcex = 1.2)  #␣
 ↪Increased font sizes

# Close the PDF device
#def.off()
```



[ ]:
```r
# Q. what happens if we try 0.5 instead of 0 for the covariance?
```

[ ]: