Estadística Aplicada 3 - Tarea 2

-Marcelino -David -Daniela

24/11/23

```
#Cargamos paquetes
library(tidymodels)
library(discrim)
library(corrr)
library(paletteer)
library(MASS)
library(dslabs)
library(tidyr)
library(openxlsx)

# Cargamos bases de datos
```

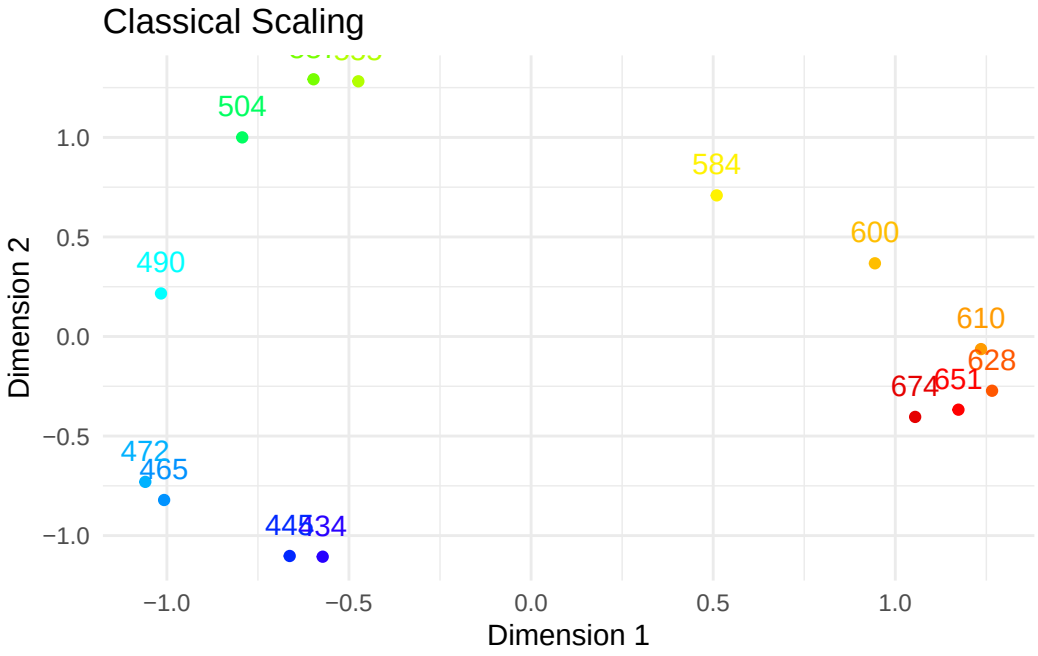
Ejercicio 1

Consider the color-stimuli experiment outlined in Section 13.2.1. The similarity ratings are given in the file color-stimuli on the books's website. Carry out a clasical scaling of the data and show that the solution is a "color circle" ranging from violet (434 $m\mu$) to blue (472 $m\mu$) to green (504 $m\mu$) to yellow (584 $m\mu$) tto red (674 $m\mu$). Compare the solution to the nonmetric scaling solution given in Figure 13.3.

En primer lugar realizamos el `clásico MDS` con la función `cmdscale` del paquete `stats` de R. Y en un principio no obtuvimos los resultados esperados, esto pasó porque la base de datos de `color-stimuli` presenta una matriz de disimilaridad y no de proximidad, la cual por alguna extraña razón es la que utiliza el paquete `cmdscale` para realizar el MDS clásico. Con lo cual realizando los ajustes necesarios obtuvimos lo siguiente mostrados en la figura 1.

```
# Cargamos base de datos
data1 <- read.xlsx("color_stimuli.xlsx")
matrix <- 1- as.matrix(data1)
```

```r
# Clasical scaling
classical <- cmdscale(matrix, k=2, eig=TRUE, add=TRUE)

hexcodes <- c('#2800ff',
              '#0028ff',
              '#0092ff',
              '#00b2ff',
              '#00ffff',
              '#00ff61',
              '#77ff00',
              '#b3ff00',
              '#fff200',
              '#ffbe00',
              '#ff9b00',
              '#ff5700',
              '#ff0000',
              '#e50000'
              )

# Mapear longitudes de onda a colores

wavelengths <- c(434, 445, 465, 472, 490, 504, 537, 555, 584, 600, 610, 628, 651, 674)

# Crear un dataframe para ggplot
df <- data.frame(classical$points, Wavelength = wavelengths, Color = hexcodes)

# Crear la gráfica
ggplot(df, aes(x = X1, y = X2, color = Color, label = Wavelength)) +
  geom_point() +
  geom_text(vjust = -1) +
  scale_color_identity() +
  labs(title = "Classical Scaling", x = "Dimension 1", y = "Dimension 2") +
  theme_minimal()
```



Ahora bien, para poder comparar los resultados obtenidos con el MDS clásico contra los resultados del la `figura 13.3` del libro tenemos la siguiente imagen.

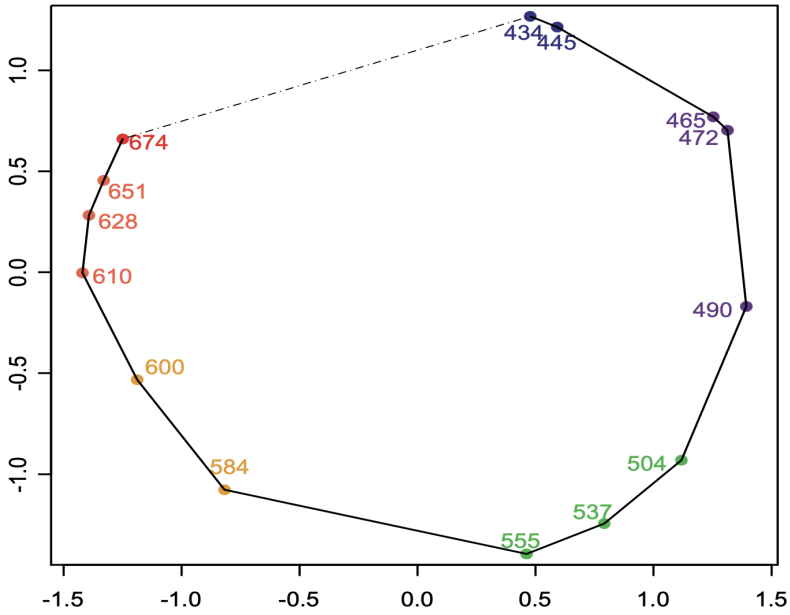


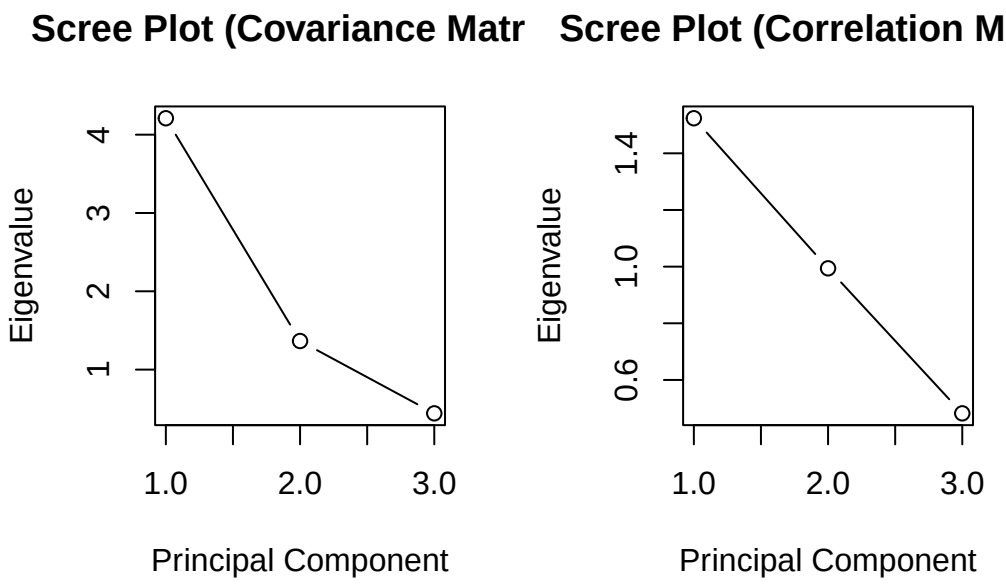
FIGURE 13.3. *Two-dimensional nonmetric MDS representation of color dissimilarities showing the "color circle." The colors correspond to the following wavelengths: 434=indigo, 445=blue, 472=blue-green, 504=green, 555=yellow-green, 600=yellow, 628=orange-yellow, 651=orange, 674=red.*

Y notamos que con el clásico logramos el mismo efecto de círculo de colores esperado y observado con el no métrico. De hecho, realmente los resultados son muy similares obtenidos por los dos métodos si solo nos fijamos en la estructura de los colores sin importar rotaciones o escalas. Claro que hay ligeras torciones observadas en el círculo de colores del clásico contra el no paramétrico, pero no son nada significativas. Como conclusión final, hace mucho sentido lo obtenido por el MDS clásico y no paramétrico, ya que los colores muy similares entre sí están más cerca, mientras que los colores muy diferentes están más lejos.

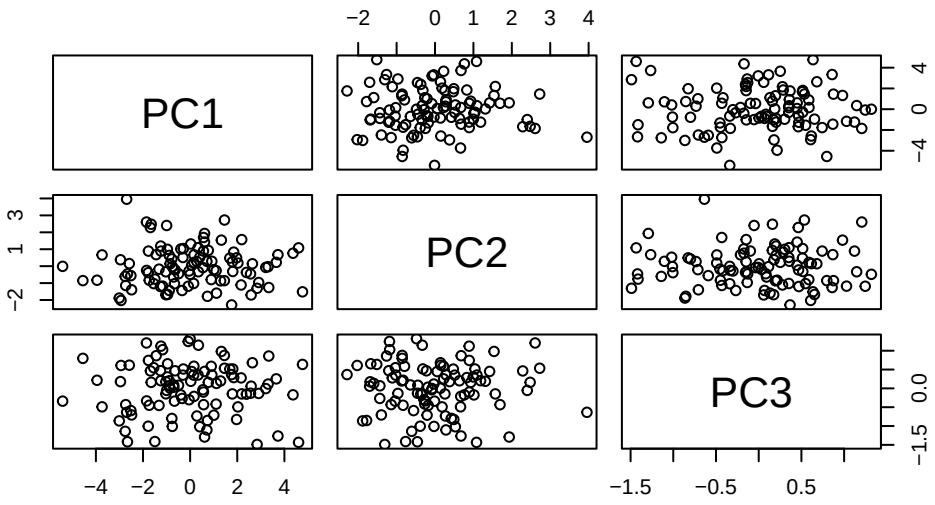
Ejercicio 2

Generate a random sample of size $n = 100$ from a three dimensional Gaussian distribution, where one of the variables has very high variance (relative to the other two). Carry out PCA on these data using the covariance matri and the correlation matrix. In each case, find the eigenvalues and eigenvectors, draw the scree plot, compute the PC scores, and plot all pairwise PC scores in a matrix plot. Compare results.

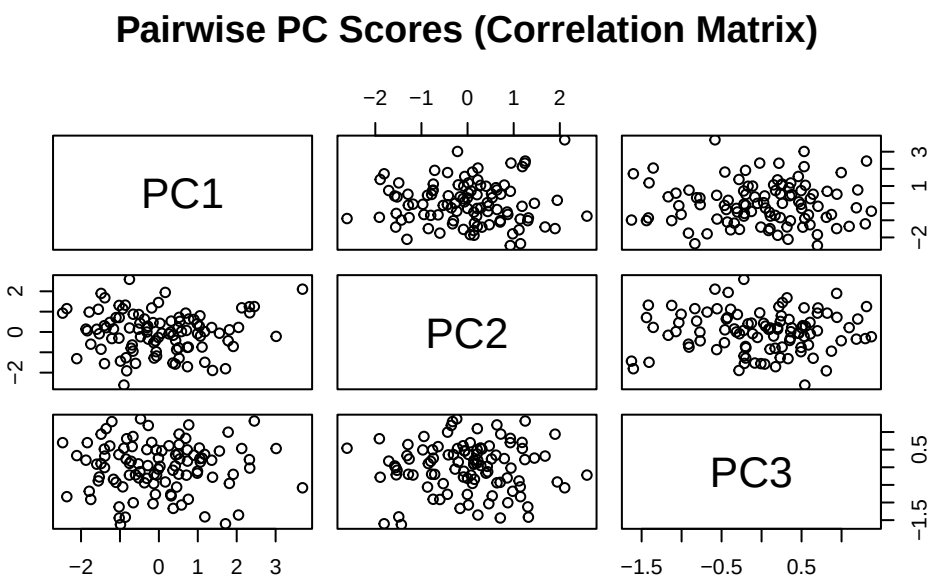
```r
# Load necessary libraries
library(MASS)
library(ggplot2)
#library(GGally)

# Set seed for reproducibility
set.seed(123)

# Generate random sample
n <- 100
mu <- c(0, 0, 0)
sigma <- matrix(c(1, 0.2, 0.5, 0.2, 5, 0.3, 0.5, 0.3, 1), nrow = 3)
data <- mvrnorm(n, mu, sigma)

# Perform PCA using covariance matrix
pca_cov <- prcomp(data, scale. = FALSE)

# Perform PCA using correlation matrix
pca_cor <- prcomp(data, scale. = TRUE)

# Extract eigenvalues and eigenvectors
eigenvalues_cov <- pca_cov$sdev^2
eigenvectors_cov <- pca_cov$rotation

eigenvalues_cor <- pca_cor$sdev^2
eigenvectors_cor <- pca_cor$rotation

# Scree plots
par(mfrow = c(1, 2))
plot(1:3, eigenvalues_cov, type = "b", main = "Scree Plot (Covariance Matrix)", xlab = "Pr
plot(1:3, eigenvalues_cor, type = "b", main = "Scree Plot (Correlation Matrix)", xlab = "P
```




```r
# Compute PC scores
scores_cov <- predict(pca_cov)
scores_cor <- predict(pca_cor)

# Matrix plot of pairwise PC scores
pairs(data.frame(scores_cov), main = "Pairwise PC Scores (Covariance Matrix)")
```



```r
pairs(data.frame(scores_cor), main = "Pairwise PC Scores (Correlation Matrix)")
```



Podemos observar del Scree Plot que la explicación de varianza para el PCA medido con covarianza es mayor para el primer componente, formando allí el codo. Mientras tanto, el PCA de correlación no muestra codo sino una línea recta, con lo cual no es posible medir bien qué componente principal eliminar al ser todos significativos.

Sumado a esto, en las gráficas de Pairwise PCA Score, podemos ver que la distribución de datos para Covarianza son más dispersos y con más caos, lo cual es más informativo de cuáles componentes son informativas para representar los datos. Mientras tanto, los Pairwise PCA Score de correlación presentan menos dispersión debido a su eigenvalor más pequeño, su menor sesgo en representación y por su unicidad (correlación quita la magnitud de varianza de la normal multivariada). El puntaje de covarianza de la Gaussiana multivariada se ve afectada en el PCA Analysis por la mayor magnitud de varianza de la distribución en su segunda dimensión $Var(X_2) = 5$, lo cual genera datos más sesgados y centrados a la segunda variable, lo que a su vez pesa mucho en la varianza total de todos los datos, su dispersión, y el Scree Plot. La desviación del Scree Plot en el codo es principalmente a esto, puesto que en el de correlación no hay codo y la gráfica es una línea recta, lo cual indica que cada componente tiene casi la misma importancia al estandarizarse la varianza a magnitud unitaria.

Ejercicio 3

En esta tabla de datos, debemos dividir los datos de tortugas macho y tortugas hembra por dimensiones de su caparazón (length, width, height). Para esto, estimaremos el vector de

medias, la matriz de covarianzas, los eigenvalores y eigenvectores de la matriz. Después se hará una prueba de PCA para explicar la composición de los datos, y a partir de estos, estimar el volumen de caparazones y comparar volúmenes entre machos y hembras.

Primero, volvemos logartimo natural los datos y separamos en dos grupos

```
library(ggplot2)
library(dplyr)
library(logisticPCA)

turtles <- read.xlsx("turtles.xlsx")
turtles <- turtles[1:4]
turtles$length <- log(turtles$length)
turtles$width <- log(turtles$width)
turtles$height <- log(turtles$height)


turtlesF <- turtles[turtles$sex=="f ",]
turtlesM <- turtles[turtles$sex=="m ",]
```

Luego, estimamos las medias y covarianzas de ambos grupos

```
#Mean Vector
meanF <- colMeans(turtlesF[2:4])
meanM <- colMeans(turtlesM[2:4])
#Covariate Matrix
covF <- cov(turtlesF[2:4])
covM <- cov(turtlesM[2:4])

meanF
```

```
  length    width   height
4.900356 4.622909 3.938253
```

```
covF
```

```
           length      width     height
length 0.02639101 0.02012395 0.02544294
width  0.02012395 0.01619045 0.01978187
height 0.02544294 0.01978187 0.02589861
```

```
meanM
```

```
  length    width   height
4.725444 4.477574 3.703186
```

```
covM
```

```
            length       width      height
length 0.011072004 0.008019142 0.008159648
width  0.008019142 0.006416726 0.006005271
height 0.008159648 0.006005271 0.006772758
```

Esta información sirve para distinguir media logarítmica y varianzas en los datos de las medidas para hembras (F) y machos (M).

Luego, calculamos los eigenvalores y eigenvectores. Esto nos da:

```
# Calculate Eigenvalues and Eigenvectors
eigen_resF <- eigen(covF)
eigen_resM <- eigen(covM)

# Extract Eigenvalues and Eigenvectors
eigenvaluesF <- eigen_resF$values
eigenvectorsF <- eigen_resF$vectors
eigenvaluesM <- eigen_resM$values
eigenvectorsM <- eigen_resM$vectors

eigenvaluesF
```

```
[1] 0.0671996134 0.0007504524 0.0005300013
```

```
eigenvectorsF
```

```
           [,1]       [,2]        [,3]
[1,] -0.6222644  0.4551794  0.63686634
[2,] -0.4840714  0.4156193 -0.77002303
[3,] -0.6151926 -0.7874467 -0.03828576
```

```
eigenvaluesM
```

```
[1] 0.0233033471 0.0005983049 0.0003598360
```

```
eigenvectorsM
```

```
           [,1]       [,2]       [,3]
[1,] -0.6831023 -0.1594791  0.7126974
[2,] -0.5102195 -0.5940118 -0.6219534
[3,] -0.5225392  0.7884900 -0.3244015
```

Obtenido esto, ahora podemos hacer una prueba de PCA para determinar:

1. Independencia y maximizar varianza explicada
2. Determinar la varianza explicada
3. Describir el volúmen de los caparazones

Primero haremos las pruebas:

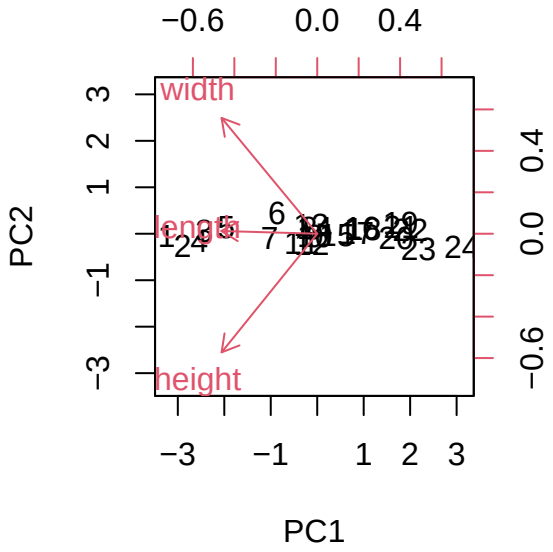
```
#PCA
#calculate principal components
resultsF <- prcomp(turtlesF[2:4], scale = TRUE)
resultsM <- prcomp(turtlesM[2:4], scale = TRUE)


#reverse the signs
resultsF$rotation <- -1*resultsF$rotation
resultsM$rotation <- -1*resultsM$rotation

#display principal components
resultsF$rotation
```

```
              PC1         PC2        PC3
length -0.5783156  0.01667633  0.8156427
width  -0.5769015  0.69855716 -0.4233233
height -0.5768325 -0.71535990 -0.3943659
```

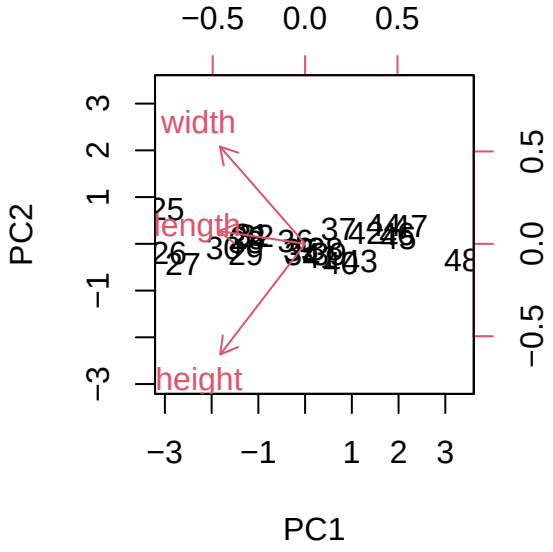
```
resultsM$rotation
```

```
              PC1         PC2        PC3
length -0.5822240  0.08657634 -0.8084057
width  -0.5758517  0.65800535  0.4852049
height -0.5739425 -0.74801972  0.3332514
```

```r
#Plot the graph of PCA
biplot(resultsF, scale = 0)
```




```r
biplot(resultsM, scale = 0)
```



Notemos que PC1 es casi una expresión lineal del vector $(1, 1, 1)$, con lo cual, PC1 se puede considerar un proxy para estimar el volúmen de caparazones.

Sumado a esto, también tenemos la "Gráfica de Codo" y la explicación de varianza:

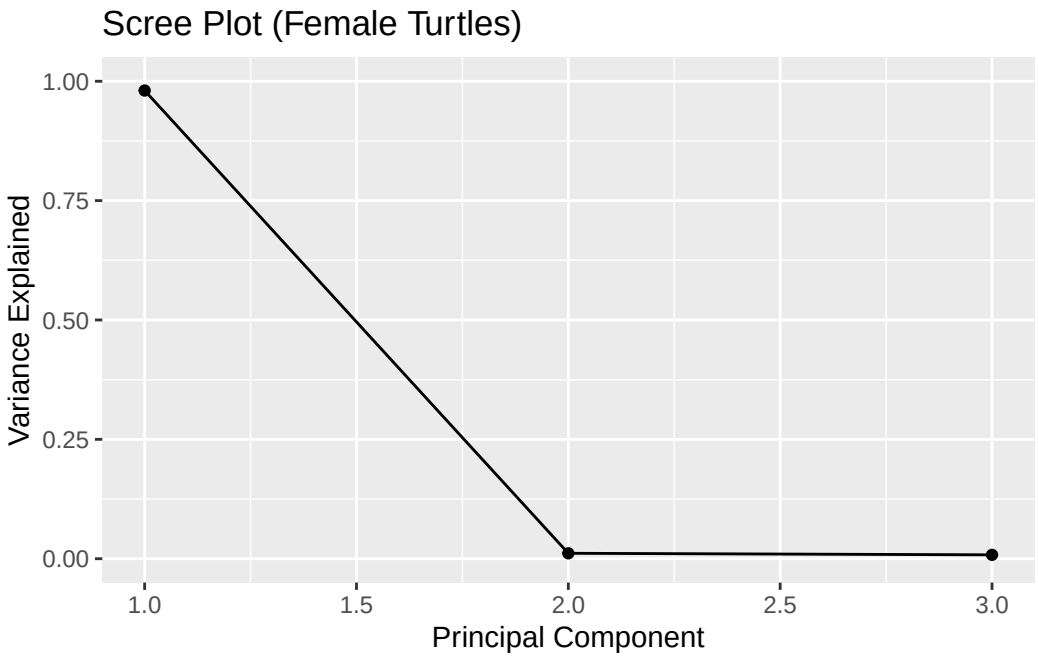
```
#calculate total variance explained by each principal component
var_explainedF = resultsF$sdev^2 / sum(resultsF$sdev^2)
var_explainedM = resultsM$sdev^2 / sum(resultsM$sdev^2)

var_explainedF
```

```
[1] 0.980621921 0.011317924 0.008060156
```

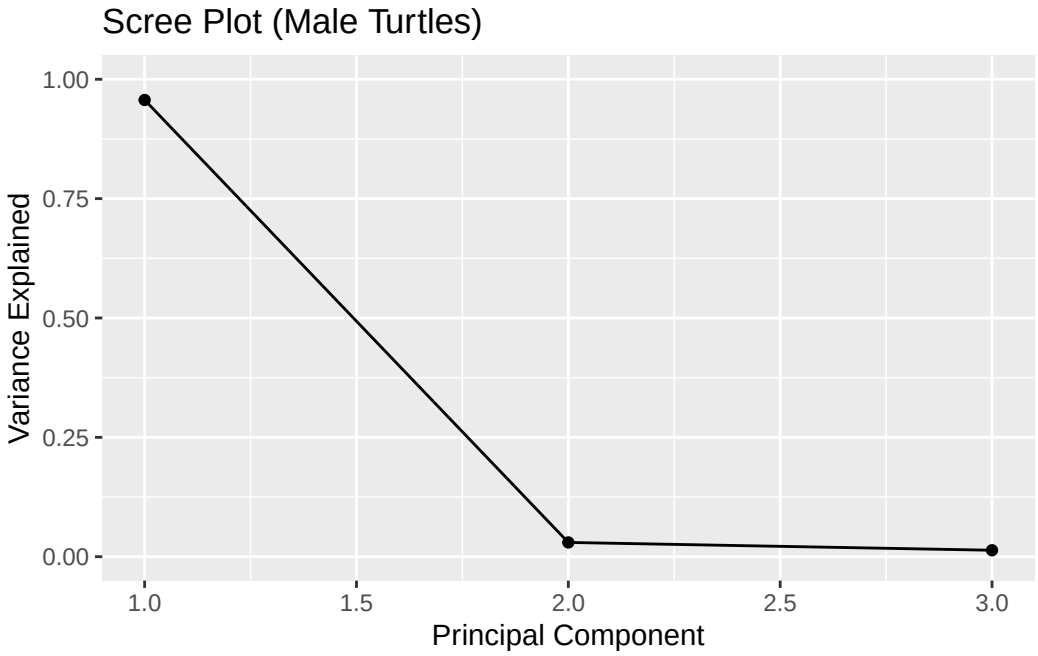
```
var_explainedM
```

```
[1] 0.95661453 0.02987152 0.01351394
```

```
#create scree plot
qplot(c(1:3), var_explainedF) +
  geom_line() +
  xlab("Principal Component") +
  ylab("Variance Explained") +
  ggtitle("Scree Plot (Female Turtles)") +
  ylim(0, 1)
```



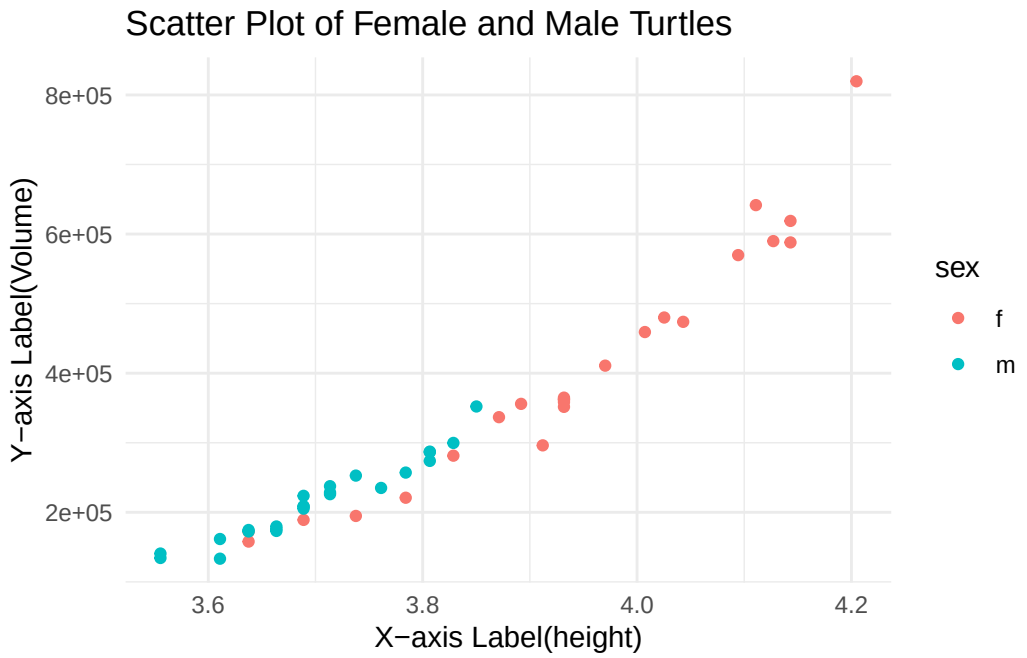

```
qplot(c(1:3), var_explainedM) +
  geom_line() +
  xlab("Principal Component") +
  ylab("Variance Explained") +
  ggtitle("Scree Plot (Male Turtles)") +
  ylim(0, 1)
```



Más del 95% de la varianza lo explica el PCA1. Por tanto, para describir el volumen podemos usar casi a la perfección el PCA1 y hacer una simple regresión para diferenciar caparazones de machos y hembras en sus volúmenes.

Por último, creamos un Dataframe con el PCA1 y los volúmenes calculados, los cuales se usará con la fórmula $Vol = (\pi/6) * \exp(\text{loglength} + \text{logwidth} + \text{logheight})$

```
#Volumen y regresión

turtles$Volume <- pi/6*exp(turtles$length+turtles$width+turtles$height)
ggplot(turtles, aes(x = height, y = Volume, color = sex)) +
  geom_point() +
  labs(title = "Scatter Plot of Female and Male Turtles",
       x = "X-axis Label(height)", y = "Y-axis Label(Volume)") +
  theme_minimal()
```




```r
#PCA Data
pca_scores <- as.matrix(turtlesF[2:4]) %*% resultsF$rotation

# Extract the first principal component (PCA1)
pca1 <- pca_scores[, 1]

# Create a new data frame with only PCA1
df_pca1 <- data.frame(PCA1 = pca1)
df_pca1
```

```
       PCA1
1  -7.285000
2  -7.334759
3  -7.406065
4  -7.389043
5  -7.478906
6  -7.648170
7  -7.618584
8  -7.747102
9  -7.764324
10 -7.764324
11 -7.722261
```

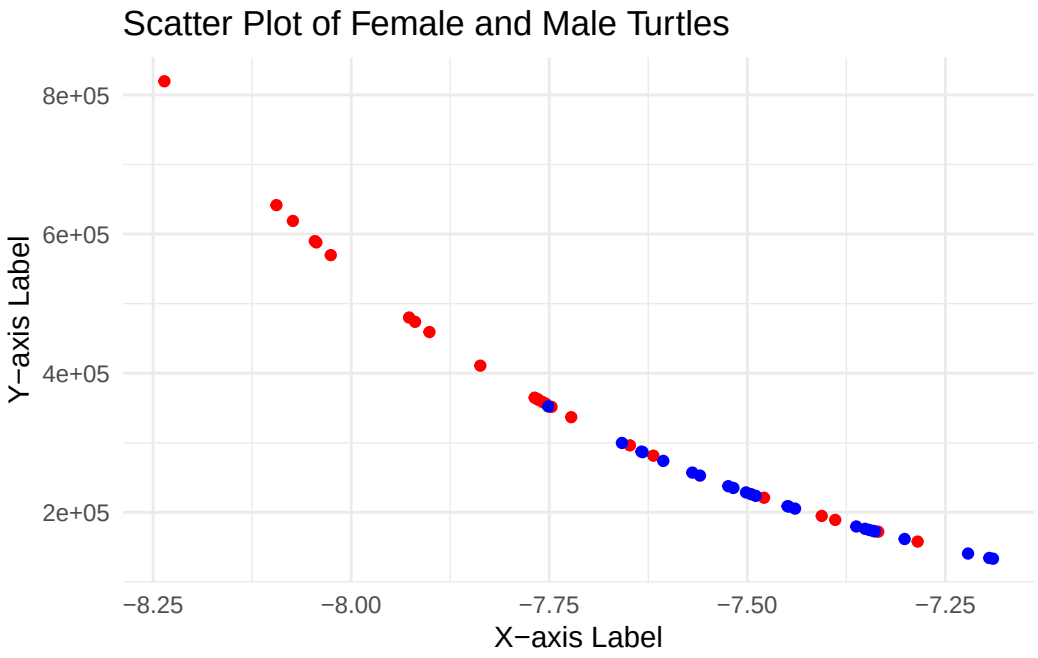
```
12 -7.754147
13 -7.758381
14 -7.768444
15 -7.837015
16 -7.919337
17 -7.901182
18 -7.926896
19 -8.043945
20 -8.025748
21 -8.045801
22 -8.073536
23 -8.094350
24 -8.235747
```

```
df_pca1$Volume <-pi/6*exp(turtlesF$length+turtlesF$width+turtlesF$height)



#PCA Data
pca_scores2 <- as.matrix(turtlesM[2:4]) %*% resultsM$rotation

# Extract the first principal component (PCA1)
pca2 <- pca_scores2[, 1]

# Create a new data frame with only PCA1
df_pca2 <- data.frame(PCA1 = pca2)
df_pca2
```

```
       PCA1
25 -7.189951
26 -7.194599
27 -7.221436
28 -7.341201
29 -7.338844
30 -7.301461
31 -7.351347
32 -7.362437
33 -7.346015
34 -7.449217
35 -7.447886
```

```
36 -7.439777
37 -7.517590
38 -7.495252
39 -7.501615
40 -7.524003
41 -7.489386
42 -7.569405
43 -7.559790
44 -7.606071
45 -7.633595
46 -7.632132
47 -7.658235
48 -7.751182
```

```
df_pca2$Volume <-pi/6*exp(turtlesM$length+turtlesM$width+turtlesM$height)



plot <- ggplot()+
  geom_point(data=df_pca1, aes(x = PCA1, y = Volume), color="red")+
  geom_point(data=df_pca2, aes(x = PCA1, y = Volume),color="blue") +
  labs(title = "Scatter Plot of Female and Male Turtles",
       x = "X-axis Label", y = "Y-axis Label") +
  theme_minimal()
print(plot)
```



Con esto podemos ver una fuerte correlación exponencial entre el PCA1 de cada sexo y su volúmen de caparazón. Con esto queda descrito el problema de la simplificación de los datos a una sola variable explicatoria por medio de PCA Analysis.

Ejercicio 4

Let X and Y be random variables with a joint distribution function given by

$$H(x, y) = (1 + e^{-x} + e^{-y})^{-1}$$

for all x, y in $\overline{R}$.

a. Show that X and Y have standard (univariate) logistic distributions, i. e.,

$$F(x) = (1 + e^{-x})^{-1} G(y) = (1 + e^{-y})^{-1}$$

b. Show that the copula of X and Y is the copula given by (2.3.4) in Example 2.8.

Solution:

a) Buscaremos las distribuciones marginales:

$F(x) = \lim_{y\to\infty} (1 + e^{-x} + e^{-y})^{-1} = (1 + e^{-x})^{-1}$

$G(y) = \lim_{x\to\infty} (1 + e^{-x} + e^{-y})^{-1} = (1 + e^{-y})^{-1}$

Q.E.D.

b) La cópula de X e Y está dada por:

$C(u, v) = \frac{H(F^{-1}(u), G^{-1}(v))}{uv}$

Para encontrar la copula, necesitamos las funciones inversas de las distribuciones marginales $F^{-1}(u)$ y $G^{-1}(v)$.

Para la distribución logística estándar, las funciones inversas son:

$F^{-1}(u) = -\ln\left(\frac{1}{u} - 1\right)$ $G^{-1}(v) = -\ln\left(\frac{1}{v} - 1\right)$

Sustituyendo estas funciones inversas en la fórmula de la copula:

$C(u, v) = \frac{H(-\ln(\frac{1}{u}-1), -\ln(\frac{1}{v}-1))}{uv}$

$= \left(1 + e^{\ln(\frac{1}{u}-1)} + e^{\ln(\frac{1}{v}-1)}\right)^{-1} / uv$

$= \left(1 + \frac{1}{u} - 1 + \frac{1}{v} - 1\right)^{-1} / uv$

$= \left(\frac{1}{u} + \frac{1}{v} - 1\right)^{-1} / uv$

$= \frac{1}{(\frac{uv}{u} + \frac{uv}{v} - uv)}$

$= \frac{1}{(u+v-uv)}$

Que era lo que queríamos demostrar.

Ejercicio 5

(A) Show that the following algorithm (Devroye 1987) generates random variables (x,y) from the Marshall-Olkin bivariate exponential distribution with parameters λ_1, λ_2 and $\lambda_{1,2}$.

1.-Generate three independent uniform (0,1) variates r, s, t

2.-Set $x = min\left(\frac{-lnr}{\lambda_1}, \frac{-lnt}{\lambda_{1,2}}\right)$, $y = min\left(\frac{-lns}{\lambda_2}, \frac{-lnt}{\lambda_{1,2}}\right)$

3.- The desired pair is (x, y).

Primero notemos que si Z es una variable aleatoria con distribución $Exp(\lambda)$, entonces su distribución acumulada es $F_Z(z) = 1 - e^{-\lambda z}$ y por lo tanto $F_Z^{-1}(u) = -\frac{ln(1-u)}{\lambda}$.

Con lo cual por el teorema de la transformada inversa, si U es una variable aleatoria con distribución uniforme en $(0, 1)$, entonces $F_Z^{-1}(U) = -\frac{ln(1-U)}{\lambda}$ tiene distribución $Exp(\lambda)$.

Por lo tanto con el algoritmo, tenemos que $Z_1 = -\frac{ln(1-r)}{\lambda_1}$, $Z_2 = -\frac{ln(1-s)}{\lambda_2}$ y $Z_3 = -\frac{ln(1-t)}{\lambda_{1,2}}$ tienen distribución $Exp(\lambda_1)$, $Exp(\lambda_2)$ y $Exp(\lambda_{1,2})$ respectivamente. Las cuales son exponenciales independientes porque son transformaciones individuales de variables que igualmente son independientes. Y de acuerdo con la definición del libro, $X = min(Z_1, Z_3)$ y $Y = min(Z_2, Z_3)$ tienen distribución Marshall-Olkin bivariada con parámetros λ_1, λ_2 y $\lambda_{1,2}$.

(B) Show that $u = e^{-(\lambda_1+\lambda_{1,2})x}$ and $v = e^{-(\lambda_2+\lambda_{1,2})y}$ are uniform (0,1) variates whose joint distribution is the Marshall-Olkin copula given by 3.1.3.

En este caso necesitamos demostrar el siguiente lema. Sean X y Y variables aleatorias independientes con distribución $Exp(\lambda)$ y $Exp(\mu)$ respectivamente. Entonces $Z = \min(X, Y) \sim Exp(\lambda + \mu)$.

Para demostrarlo notemos que $P(Z \geq z) = P(X \geq x, Y \geq y)$ y como son independientes obtenemos el siguiente desarrollo:

$$\begin{aligned} P(Z \geq z) = P(X \geq z, Y \geq z) &= P(X \geq z)P(Y \geq z) \\ &= e^{-\lambda z}e^{-\mu z} \\ &= e^{-(\lambda+\mu)z} \end{aligned}$$

Es decir, $Z \sim Exp(\lambda + \mu)$.

Por lo tanto, sean Z_1, Z_2 y Z_3 las variables aleatorias independientes con distribución $Exp(\lambda_1)$, $Exp(\lambda_2)$ y $Exp(\lambda_{1,2})$ respectivamente definidas como en el anterior inciso. Entonces $X = min(Z_1, Z_3)$ y $Y = min(Z_2, Z_3)$ tienen distribución $Exp(\lambda_1 + \lambda_{1,2})$ y $Exp(\lambda_2 + \lambda_{1,2})$ respectivamente.

Con lo cual, $u = e^{-(\lambda_1+\lambda_{1,2})x}$ y $v = e^{-(\lambda_2+\lambda_{1,2})y}$ son uniformes en $(0, 1)$ utilizando el mismo argumento que en el inciso anterior del teorema de la transformada inversa, aplicada a las funciones de distribución de X y Y acumuladas. Ahora solo notemos que la distribución conjunta de u y v es la siguiente:

$$
\begin{aligned}
S_{U,V}(u,v) &= P(e^{-(\lambda_1+\lambda_{1,2})X} \geq u, e^{-(\lambda_2+\lambda_{1,2})Y} \geq v) \\
&= P(X \geq S_X^{-1}(u), Y \geq S_Y^{-1}(v)) \\
&= P(X \geq -\frac{ln(u)}{\lambda_1+\lambda_{1,2}}, Y \geq -\frac{ln(v)}{\lambda_2+\lambda_{1,2}}) \\
&= \mathbf{exp}\{-\lambda_1 S_X^{-1}(u) - \lambda_2 S_Y^{-1}(v) - \lambda_{1,2}\max(S_X^{-1}(u), S_Y^{-1}(v))\} \\
&= \mathbf{exp}\{-(\lambda_1+\lambda_{1,2})S_X^{-1}(u) - (\lambda_2+\lambda_{1,2})S_Y^{-1}(v) + \\
&\lambda_{1,2}\min(S_X^{-1}(u), S_Y^{-1}(v))\} \\
&= S_X(S_X^{-1}(u))S_Y(S_Y^{-1}(v))\min(\mathbf{exp}(\lambda_{1,2}(S_X^{-1}(u))), \mathbf{exp}(S_Y^{-1}(v))) \\
&= uv\min(u^{-\alpha}, v^{-\beta})
\end{aligned}
$$

donde $\alpha = \frac{\lambda_{1,2}}{\lambda_1+\lambda_{1,2}}$ y $\beta = \frac{\lambda_{1,2}}{\lambda_2+\lambda_{1,2}}$.

Y como pudimos observar, la distribución conjunta de u y v es la misma que la cópula de supervivencia dada en el libro. Además, notemos que $u = e^{-(\lambda_1+\lambda_{1,2})x}$ y $v = e^{-(\lambda_2+\lambda_{1,2})y}$ son uniformes en $(0,1)$ utilizando el mismo argumento que en el inciso anterior del teorema de la transformada inversa, aplicada a las funciones de distribución de X y Y acumuladas