# Transformaciones

Catherine Rojas

2024-08-14

## 1. Selecciona una variable, que no sea Calorías, y encuentra la mejor transformación de datos posible para que la variable seleccionada se comporte como una distribución Normal.

Protein

## Carga de Datos

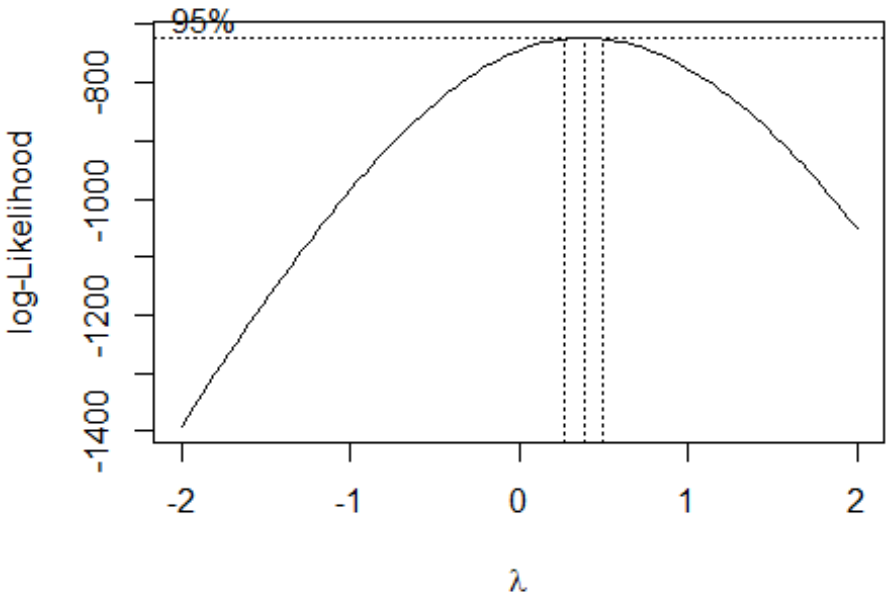
```
# Cargar el archivo de datos
data <- read.csv("mc-donalds-menu.csv")

# Seleccionar la variable Carbohydrates
x <- data$Protein
```

## 2. Utiliza la transformación Box-Cox. Utiliza el modelo exacto y el aproximado de acuerdo con las sugerencias de Box y Cox para la transformación

```
# Cargar la librería 'MASS' para 'boxcox'
library(MASS)

# Box Cox
bc <- boxcox((x+1) ~ 1)
```




**Intepretación:** La gráfica muestra la relación entre los valores de $\lambda$ y la log-verosimilitud. - El punto más alto de la curva indica el valor de $\lambda$ que maximiza la log-verosimilitud, es decir, el mejor $\lambda$ para transformar los datos hacia una distribución normal.

- Las líneas punteadas delimitan un intervalo de confianza al 95% para el rango de valores de $\lambda$ que son estadísticamente aceptables para la transformación.

```
# Encontrar el mejor lambda
lambda <- bc$x[which.max(bc$y)]
lambda

## [1] 0.3838384

# Transformación Box-Cox exacta
trans_exact <- (x^lambda - 1) / lambda

# Aplicar la transformación Box-Cox aproximada
trans_aprox <- sqrt(x)
```

**a)Transformación Exacta de Box-Cox:**

$$x_{\text{transformado}} = \frac{x^{\lambda} - 1}{\lambda} \quad \text{si } \lambda \neq 0$$

$$x_{\text{transformado}} = \frac{x^{0.3838384} - 1}{0.3838384} \quad \text{si } \lambda \neq 0$$

Donde:

- $x_{\text{transformado}}$ es el valor transformado de la variable protein
- $x$ es el valor original de la variable protein
- $\lambda$ es el valor óptimo encontrado que maximiza la log-verosimilitud en el análisis de Box-Cox.

En este caso, $\lambda$ = {0.3838384}

**b)Transformación Aproximada de Box-Cox:**

$$x_{\text{transformado}} = \sqrt{x}$$

Donde:

- $x_{\text{transformado}}$ es el valor transformado de la variable protein
- $x$ es el valor original de la variable

La elección entre la transformación exacta y la aproximada depende del valor óptimo de ($\lambda$ ) encontrado en el análisis de Box-Cox.

- Si ($\lambda$ ) es cercano a 0, la transformación aproximada se puede utilizar como una simplificación razonable. Sin embargo, si ($\lambda$ ) es significativamente diferente de 0, es preferible utilizar la transformación exacta para obtener una corrección más precisa.

## 3. Analiza la normalidad de las transformaciones obtenidas con los datos originales. Utiliza como argumento de normalidad:

**a) Compara las medidas: Mínimo, máximo, media, mediana, cuartil 1 y cuartil 3, sesgo y curtosis.**

```
# Resumen de los datos originales
cat("\nMedidas de los datos originales:\n")

##
## Medidas de los datos originales:

resumen_x <- summary(x)
resumen_x

##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##    0.00    4.00   12.00   13.34   19.00   87.00

library(e1071)

## Warning: package 'e1071' was built under R version 4.3.3

# Sesgo y curtosis de los datos originales
sesgo_x <- skewness(x)
```

```r
curtosis_x <- kurtosis(x)

cat("\nSesgo con los datos originales:\n", sesgo_x, "\n")
```

```
##
## Sesgo con los datos originales:
##  1.561741
```

```r
cat("\nCurtosis con los datos originales:\n", curtosis_x, "\n")
```

```
##
## Curtosis con los datos originales:
##  5.7955
```

```r
# Resumen de los datos transformados con Box-Cox exacto
cat("\nMedidas con Box-Cox exacto:\n")
```

```
##
## Medidas con Box-Cox exacto:
```

```r
resumen_trans_exact <- summary(trans_exact)
resumen_trans_exact
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##  -2.605   1.830   4.157   3.509   5.461  11.860
```

```r
# Sesgo y curtosis de los datos transformados con Box-Cox exacto
sesgo_trans_exact <- skewness(trans_exact)
curtosis_trans_exact <- kurtosis(trans_exact)

cat("\nSesgo con Box-Cox exacto:\n", sesgo_trans_exact, "\n")
```

```
##
## Sesgo con Box-Cox exacto:
##  -0.6179156
```

```r
cat("\nCurtosis con Box-Cox exacto:\n", curtosis_trans_exact, "\n")
```

```
##
## Curtosis con Box-Cox exacto:
##  -0.1314858
```

```r
# Resumen de los datos transformados con Box-Cox aproximado
cat("\nMedidas con Box-Cox aproximado:\n")
```

```
##
## Medidas con Box-Cox aproximado:
```

```r
resumen_trans_aprox <- summary(trans_aprox)
resumen_trans_aprox
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   0.000   2.000   3.464   3.207   4.359   9.327
```

```r
# Sesgo y curtosis de los datos transformados con Box-Cox aproximado
sesgo_trans_aprox <- skewness(trans_aprox)
curtosis_trans_aprox <- kurtosis(trans_aprox)

cat("\nSesgo con Box-Cox aproximado:\n", sesgo_trans_aprox, "\n")
```

```
##
## Sesgo con Box-Cox aproximado:
##  -0.1736453
```

```r
cat("\nCurtosis con Box-Cox aproximado:\n", curtosis_trans_aprox, "\n")
```

```
##
## Curtosis con Box-Cox aproximado:
##  -0.2544237
```

```r
# Tabla con resultados
tabla_resultados <- data.frame(
  "Medida" = c("Min", "1st Qu.", "Median", "Mean", "3rd Qu.", "Max",
"Sesgo", "Curtosis"),
  "Datos_Originales" = c(resumen_x["Min."], resumen_x["1st Qu."],
resumen_x["Median"], resumen_x["Mean"], resumen_x["3rd Qu."],
resumen_x["Max."], sesgo_x, curtosis_x),
  "Box-Cox_Exacto" = c(resumen_trans_exact["Min."],
resumen_trans_exact["1st Qu."], resumen_trans_exact["Median"],
resumen_trans_exact["Mean"], resumen_trans_exact["3rd Qu."],
resumen_trans_exact["Max."], sesgo_trans_exact, curtosis_trans_exact),
  "Box-Cox_Aproximado" = c(resumen_trans_aprox["Min."],
resumen_trans_aprox["1st Qu."], resumen_trans_aprox["Median"],
resumen_trans_aprox["Mean"], resumen_trans_aprox["3rd Qu."],
resumen_trans_aprox["Max."], sesgo_trans_aprox, curtosis_trans_aprox)
)

tabla_resultados
```

```
##     Medida Datos_Originales Box.Cox_Exacto Box.Cox_Aproximado
## 1      Min         0.000000     -2.6052632          0.0000000
## 2  1st Qu.         4.000000      1.8302649          2.0000000
## 3   Median        12.000000      4.1568554          3.4641016
## 4     Mean        13.338462      3.5091457          3.2074020
## 5  3rd Qu.        19.000000      5.4612587          4.3588989
## 6      Max        87.000000     11.8595735          9.3273791
## 7    Sesgo         1.561741     -0.6179156         -0.1736453
## 8 Curtosis         5.795500     -0.1314858         -0.2544237
```

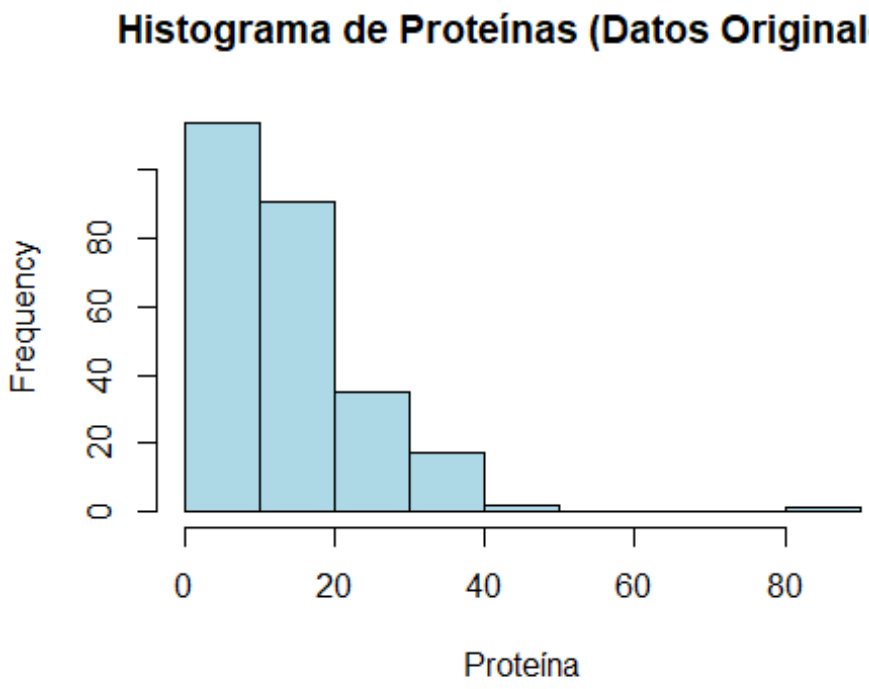
**Observaciónes de los resultados** - La mediana y la media de los datos transformados es menor que la de los datos originales, lo que indica una compresión de los valores altos hacia el centro de la distribución.

- Los valores máximos se reducen considerablemente tras las transformaciones, lo que indica una reducción de los valores extremos.

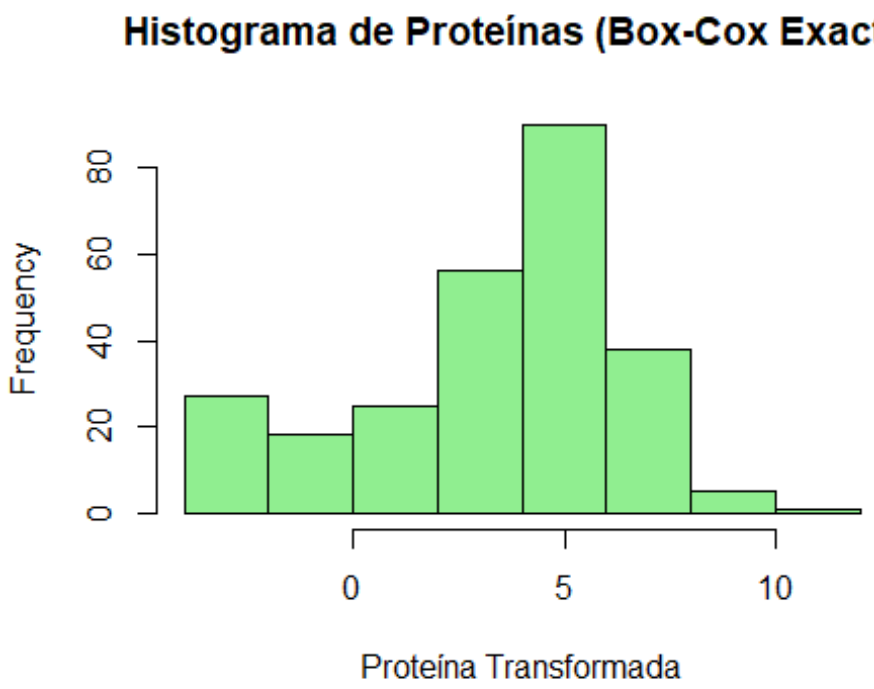
- Las transformaciones reducen la curtosis y el sesgo en los datos originales es positivo, indicando una distribución sesgada a la derecha. Después de las transformaciones, el sesgo se reduce e incluso se vuelve negativo, sugiriendo una distribución más simétrica.

**b) Obten el histograma de los 2 modelos obtenidos (exacto y aproximado) y los datos originales.**

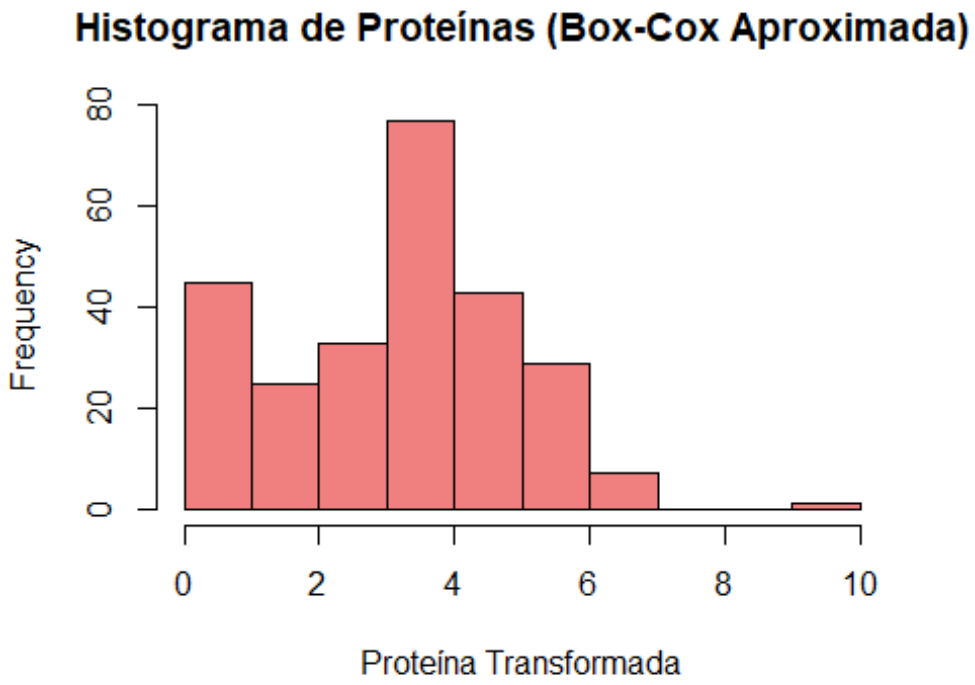
```
# Histograma de los datos originales
hist(x, main = "Histograma de Proteínas (Datos Originales)", xlab =
"Proteína", col = "lightblue", border = "black")
```




```
# Histograma de la transformación Box-Cox exacta
hist(trans_exact, main = "Histograma de Proteínas (Box-Cox Exacta)", xlab
= "Proteína Transformada", col = "lightgreen", border = "black")
```




```r
# Histograma de la transformación Box-Cox aproximada
hist(trans_aprox, main = "Histograma de Proteínas (Box-Cox Aproximada)",
xlab = "Proteína Transformada", col = "lightcoral", border = "black")
```




**Interpretación de los gráficos** - El histrograma de los datos originales muestra una distribución segada hacia la derecha con un alto número de observaciones en los valores cercanos a cero, y pocos valores altos.

- El histogrma con Box-Cox exacto muestra una distribución más simética en comparación con la de los datos originales y la mayoria de los valores se concentran alrededor del centro de distribución. El sesgo se ha reducido considerablemente y los datos parecen aproximarse a una distribución normal.
- En el histrograma de Box-Cox aproximado, la asimetría y el sesgo a reducido, pero sigue siendo más visible el caso de la transformación exacta.

**c) Realiza la prueba de normalidad de Anderson-Darling o de Jarque Bera para los datos transformados y los originales**

Para todas las pruebas de normalidad, la hipótesis nula $[H_0]$ es que los datos siguen una distribución normal.

```
library(nortest)

# Prueba de normalidad de Anderson-Darling para los datos originales
ad_test_x <- ad.test(x)
ad_test_x

##
##  Anderson-Darling normality test
##
```

```
## data:  x
## A = 4.7515, p-value = 8.515e-12

# Prueba de normalidad de Anderson-Darling para los datos con Box-Cox exacto
ad_test_exact <- ad.test(trans_exact)
ad_test_exact

##
##  Anderson-Darling normality test
##
## data:  trans_exact
## A = 6.1008, p-value = 4.942e-15

# Prueba de normalidad de Anderson-Darling para los datos con Box-Cox aproximado
ad_test_aprox <- ad.test(trans_aprox)
ad_test_aprox

##
##  Anderson-Darling normality test
##
## data:  trans_aprox
## A = 3.0949, p-value = 8.785e-08

library(tseries)

## Warning: package 'tseries' was built under R version 4.3.3

## Registered S3 method overwritten by 'quantmod':
##   method            from
##   as.zoo.data.frame zoo

# Prueba de normalidad de Jarque-Bera para los datos originales
jb_test_x <- jarque.bera.test(x)
jb_test_x

##
##  Jarque Bera Test
##
## data:  x
## X-squared = 479.38, df = 2, p-value < 2.2e-16

# Prueba de normalidad de Jarque-Bera para Box-Cox exacto
jb_test_exact <- jarque.bera.test(trans_exact)
jb_test_exact

##
##  Jarque Bera Test
##
## data:  trans_exact
## X-squared = 16.867, df = 2, p-value = 0.0002174
```

```r
# Prueba de normalidad de Jarque-Bera para Box-Cox aproximado
jb_test_aprox <- jarque.bera.test(trans_aprox)
jb_test_aprox
```

```
##
##  Jarque Bera Test
##
## data:  trans_aprox
## X-squared = 1.9109, df = 2, p-value = 0.3846
```

```r
# Tabla con resultados
resultados_normalidad <- data.frame(
  Test = c("Anderson-Darling", "Anderson-Darling", "Anderson-Darling",
"Jarque-Bera", "Jarque-Bera", "Jarque-Bera"),
  Transformation = c("Original", "Box-Cox Exacto", "Box-Cox Aproximado",
"Original", "Box-Cox Exacto", "Box-Cox Aproximado"),
  Statistic = c(ad_test_x$statistic, ad_test_exact$statistic,
ad_test_aprox$statistic,
                jb_test_x$statistic, jb_test_exact$statistic,
jb_test_aprox$statistic),
  P_Value = c(ad_test_x$p.value, ad_test_exact$p.value,
ad_test_aprox$p.value,
              jb_test_x$p.value, jb_test_exact$p.value,
jb_test_aprox$p.value)
)

resultados_normalidad
```

```
##               Test     Transformation  Statistic      P_Value
## 1 Anderson-Darling           Original   4.751457 8.515383e-12
## 2 Anderson-Darling     Box-Cox Exacto   6.100798 4.941787e-15
## 3 Anderson-Darling Box-Cox Aproximado   3.094944 8.784536e-08
## 4      Jarque-Bera           Original 479.383612 0.000000e+00
## 5      Jarque-Bera     Box-Cox Exacto  16.867311 2.174253e-04
## 6      Jarque-Bera Box-Cox Aproximado   1.910856 3.846474e-01
```

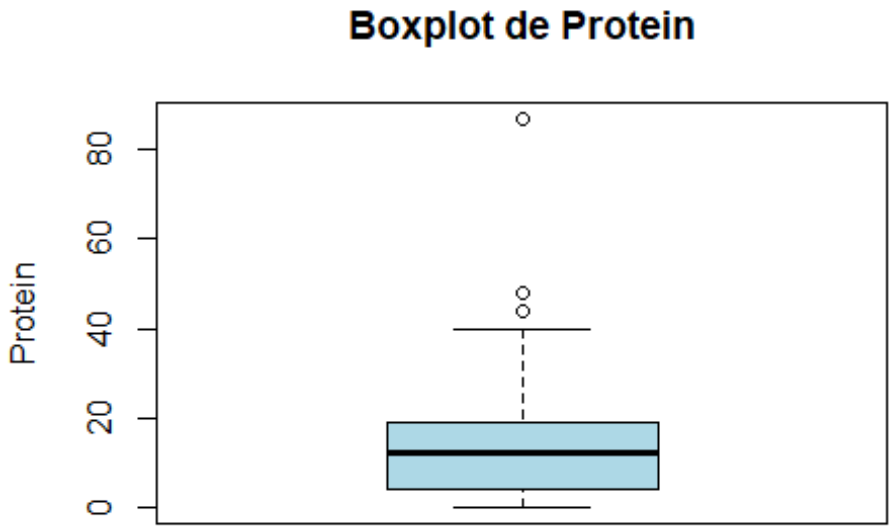
**Interpretacoión de resultados** - Un valor p bajo (generalmente < 0.05) indica que podemos rechazar la hipótesis nula, lo que sugiere que los datos no siguen una distribución normal.

Basado en estos resultados, la transformación Box-Cox aproximado es la más efectiva para normalizar la distribución de los datos de proteínas, ya que tiene el valor de P más alto en la prueba de Jarque-Bera, indicando que los datos transformados se asemejan más a una distribución normal.

## 4. Detecta anomalías y corrige tu base de datos (datos atípicos, ceros anámalos, etc).

### a) Graficar el diagrama de caja y bigote

```r
# Boxplot para Protein
boxplot(x, main="Boxplot de Protein", ylab="Protein", col="lightblue")
```




**Interpretación de la gráfica** - La mediana pareciera estar cercana al valor de 15g, esto significa que la mitad de los elementos del menú tienen un contenido de proteínas menor o igual a 15g y la otra mitad tiene un contenido mayor.

Aproximadamente el 50% de los datos se encuentran entre el primer (Q1) y tercer cuartil (Q3) que es aproximadamente entre el valor de 10 y 25 gramos respectivamente.

Se puede apreciar que existe una baja variabilidad en los datos debido al tamaño de la caja, sin embargo existen valores atípicos por encima del bigote superior, lo que indica que hay alimentos en el menú con un contenido de proteínas significativamente mayor que la mayoría. Estos valores se encuentran cerca del valor de 45g y 50g y otro más en el de 90g.

**b) Calcula el rango intercuartílico y los cuartíles**

```r
# Calcular los cuartiles y el rango intercuartílico (IQR) para Protein
quartiles_x <- quantile(x, probs=c(0.25, 0.5, 0.75))

IQR_x<- IQR(x)


# Tabla con resultados
resultados_quartil <- data.frame(
```

```
  Variable = c("Protein"),
  Q1 = c(quartiles_x[1]),
  Q2 = c(quartiles_x[2]),
  Q3 = c(quartiles_x[3]),
  IQR = c(IQR_x)
)

resultados_quartil
```

```
##     Variable Q1 Q2 Q3 IQR
## 25%  Protein  4 12 19  15
```

**Interpretación de resultados**

- 1er cuartil (Q1): El 25% de los elementos del menú tienen un contenido de proteínas menor o igual a 4 gramos.
- Mediana (Q2): La mediana del contenido de proteínas es de 12 gramos, lo que significa que el 50% de los elementos tienen un contenido de proteínas menor o igual a este valor.
- 3er Cuartil (Q3): El 75% de los elementos del menú tienen un contenido de proteínas menor o igual a 19 gramos.
- Rango Intercuartílico (IQR): La diferencia entre Q3 y Q1 (19 - 4), indica la dispersión del contenido de proteínas en el 50% central de los datos.

**c) Identifica la cota de 1.5 rangos intercuartílicos para datos atípicos, ¿hay datos atípicos de acuerdo con este criterio?**

```
# Calcular los límites inferior y superior para detectar datos atípicos
en Protein
limite_inferior_x <- quartiles_x[1] - 1.5 * IQR_x
limite_superior_x <- quartiles_x[3] + 1.5 * IQR_x

# Tabla con resultados
resultados_limites <- data.frame(
  Variable = c("Protein"),
  Limite_Inferior = c(limite_inferior_x),
  Limite_Superior = c(limite_superior_x)
)

resultados_limites
```

```
##     Variable Limite_Inferior Limite_Superior
## 25%  Protein           -18.5            41.5
```

**Interpretación de resultados** De acuerdo con el criterio de 1.5 veces el rango intercuartílico, cualquier valor de proteínas por debajo de -18.5 gramos o por encima de 41.5 gramos se consideraría un dato atípico. Debido a que el contenido de

proteínas no puede ser negativo, solo los valores por encima de 41.5 gramos se considerarían atípicos.

**d) Identifica la cota de 3 desviaciones estándar alrededor de la media, ¿hay datos atípicos de acuerdo con este criterio?**

```
# Calcular la media y la desviación estándar para Protein
media_x <- mean(x, na.rm = TRUE)
sd_x <- sd(x, na.rm = TRUE)

# Calcular los límites inferior y superior para detectar datos atípicos
en Protein
limite_inferior_x <- media_x - 3 * sd_x
limite_superior_x <- media_x + 3 * sd_x


# Tabla con resultados
resultados_sd <- data.frame(
  Variable = c("Protein"),
  Limite_Inferior = c(limite_inferior_x),
  Limite_Superior = c(limite_superior_x)
)

resultados_sd
```

```
##   Variable Limite_Inferior Limite_Superior
## 1  Protein       -20.93998         47.6169
```

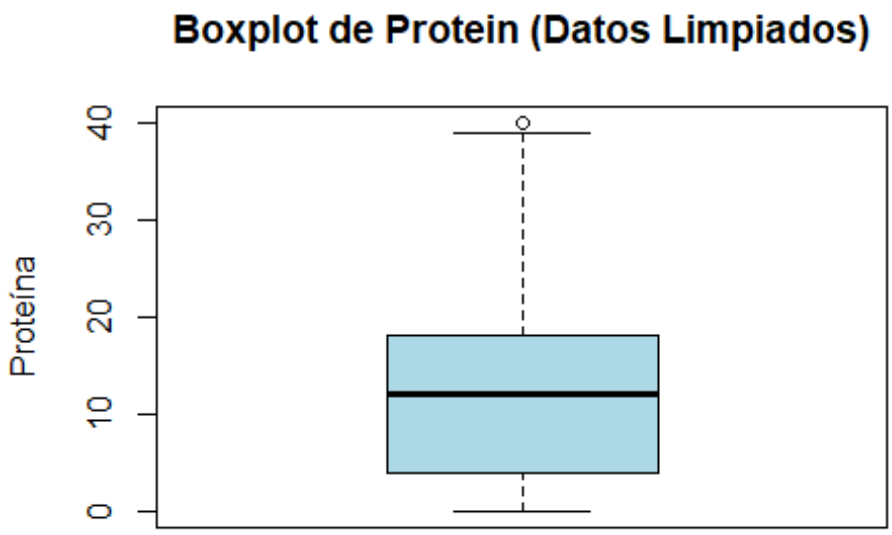
**Interpretación de resultados** Cualquier valor de proteínas por debajo de -20.93998 gramos o por encima de 47.6169 gramos se consideraría un dato atípico. Al igual que en el criterio anterior, solo se consideran los valores positivos, es decir, los valores por encima de 47.6169 gramos se considerarían atípicos.

**e) Toma una decisión de si conviene o no quitar los datos atípicos (para ello interpreta la variable en el contexto del problema y determina si es necesario quitarlos o no quitarlos)**

Los criterios anteriores identifican valores superiores a aproximadamente 41.5 y 47.6169 como atípicos. Ya que el objetivo es realizar un análisis que asuma normalidad, puede ser conveniente eliminaf los datos atípicos para mejorar la normalidad de los datos. Si se buscara aplicar algún modelo estadístico o machine learning, los valores atípicos pueden influir significativamente en los resultados, por lo que la eliminación de estos podría estabilizar las estimaciones.

```
# Eliminar valores atípicos según el criterio del IQR
x_cleaned <- x[x <= 41.5]

# Verificación del boxplot después de la eliminación
boxplot(x_cleaned, main = "Boxplot de Protein (Datos Limpiados)", ylab =
"Proteína", col = "lightblue")
```




**Interpretación de la gráfica** El boxplot de los datos de proteínas limpiados muestra la distribución de los datos después de eliminar los valores atípicos superiores a 41.5, según el criterio la cota de 1.5 rangos intercuartílicos.

El boxplot original mostraba varios valores atípicos significativos por encima de 41.5. Después de la limpieza, la mayoría de estos valores atípicos se han eliminado. La presencia de solo uno o pocos valores atípicos restantes sugiere que la limpieza fue efectiva para eliminar la mayoría de las anomalías extremas.

La dispersión de los datos (representada por la longitud de la caja y los bigotes) es menor, indicando que los datos están más concentrados y menos influenciados por valores extremos.

### f) Revisión de datos cero

```r
# Calcular la frecuencia de ceros
num_zeros <- sum(x == 0)
total_values <- length(x)
percent_zeros <- (num_zeros / total_values) * 100

# Mostrar la frecuencia de ceros
cat("Número de ceros:", num_zeros, "\n")

## Número de ceros: 27

cat("Porcentaje de ceros:", percent_zeros, "%\n")

## Porcentaje de ceros: 10.38462 %
```

```r
# Ver los valores de otras variables donde Protein es cero
zeros_protein <- data[data$Protein == 0, ]
summary(zeros_protein)
```

```
##    Category             Item           Serving.Size          Calories
##  Length:27          Length:27          Length:27          Min.   :
0.00
##  Class :character   Class :character   Class :character   1st Qu.:
0.00
##  Mode  :character   Mode  :character   Mode  :character   Median :
80.00
##                                                           Mean   :
86.85
##                                                           3rd
Qu.:140.00
##                                                           Max.
:280.00
##  Calories.from.Fat   Total.Fat Total.Fat....Daily.Value. Saturated.Fat
##  Min.   :0         Min.   :0   Min.   :0                 Min.   :0
##  1st Qu.:0         1st Qu.:0   1st Qu.:0                 1st Qu.:0
##  Median :0         Median :0   Median :0                 Median :0
##  Mean   :0         Mean   :0   Mean   :0                 Mean   :0
##  3rd Qu.:0         3rd Qu.:0   3rd Qu.:0                 3rd Qu.:0
##  Max.   :0         Max.   :0   Max.   :0                 Max.   :0
##  Saturated.Fat....Daily.Value.   Trans.Fat  Cholesterol
##  Min.   :0                     Min.   :0   Min.   :0
##  1st Qu.:0                     1st Qu.:0   1st Qu.:0
##  Median :0                     Median :0   Median :0
##  Mean   :0                     Mean   :0   Mean   :0
##  3rd Qu.:0                     3rd Qu.:0   3rd Qu.:0
##  Max.   :0                     Max.   :0   Max.   :0
##  Cholesterol....Daily.Value.     Sodium     Sodium....Daily.Value.
##  Min.   :0                   Min.   : 0.0   Min.   :0.0000
##  1st Qu.:0                   1st Qu.: 2.5   1st Qu.:0.0000
##  Median :0                   Median :10.0   Median :0.0000
##  Mean   :0                   Mean   :20.0   Mean   :0.8148
##  3rd Qu.:0                   3rd Qu.:30.0   3rd Qu.:1.0000
##  Max.   :0                   Max.   :90.0   Max.   :4.0000
##  Carbohydrates   Carbohydrates....Daily.Value. Dietary.Fiber
##  Min.   : 0.00   Min.   : 0.000                Min.   :0
##  1st Qu.: 0.00   1st Qu.: 0.000                1st Qu.:0
##  Median :21.00   Median : 7.000                Median :0
##  Mean   :23.37   Mean   : 7.741                Mean   :0
##  3rd Qu.:38.00   3rd Qu.:12.500                3rd Qu.:0
##  Max.   :76.00   Max.   :25.000                Max.   :0
##  Dietary.Fiber....Daily.Value.     Sugars      Protein
##  Min.   :0                     Min.   : 0   Min.   :0
##  1st Qu.:0                     1st Qu.: 0   1st Qu.:0
##  Median :0                     Median :19   Median :0
##  Mean   :0                     Mean   :23   Mean   :0
```

```
##  3rd Qu.:0                    3rd Qu.:38   3rd Qu.:0
##  Max.   :0                    Max.   :76   Max.   :0
##  Vitamin.A....Daily.Value. Vitamin.C....Daily.Value.
Calcium....Daily.Value.
##  Min.   :0                 Min.   :  0.00           Min.   : 0.0000
##  1st Qu.:0                 1st Qu.:  0.00           1st Qu.: 0.0000
##  Median :0                 Median :  0.00           Median : 0.0000
##  Mean   :0                 Mean   :  9.63           Mean   : 0.4444
##  3rd Qu.:0                 3rd Qu.:  0.00           3rd Qu.: 0.0000
##  Max.   :0                 Max.   :160.00           Max.   :10.0000
##  Iron....Daily.Value.
##  Min.   :0
##  1st Qu.:0
##  Median :0
##  Mean   :0
##  3rd Qu.:0
##  Max.   :0
```

**Observaciones** - Los productos con 0 gramos de proteína tienen un amplio rango de calorías, desde 0 hasta 280. - El sodio también varía, con algunos productos sin sodio y otros con hasta 90 mg. - Hay productos con 0 carbohidratos y otros con hasta 76 g, lo que indica una amplia variación. - Otros Nutrientes (Grasas, Fibra Dietética, Azúcares, Vitaminas, Minerales) también son 0 en los productos sin proteína.

Para identificar ceros anómalos en la variable Protein, se decidio aplicar una regla de negocio basada en la relación entre Protein y las calorías.

**Regla de Negocio** Si Protein es cero y Calories es mayor a un umbral razonable, entonces el valor cero de Protein se considera anómalo.

Esta regla se basa en la suposición de que es poco probable que un producto tenga una cantidad significativa de calorías sin contener proteínas.

Esto se considera razonable, ya que productos con muy pocas calorías (por ejemplo, agua, bebidas dietéticas) tienen cero proteínas y productos con muchas calorías que no contienen proteínas podrían ser errores de entrada o datos anómalos, ya que es raro que productos calóricos no tengan proteínas (a menos que estén compuestos exclusivamente de azúcares o grasas, lo cual también es raro).

```
# Aplicar una regla de negocio para identificar ceros anómalos
anomalous_zeros <- data[x == 0 & data$Calories > 100, ]
anomalous_zeros
```

```
##         Category                      Item Serving.Size Calories
## 111    Beverages  Coca-Cola Classic (Small) 16 fl oz cup      140
## 112    Beverages Coca-Cola Classic (Medium) 21 fl oz cup      200
## 113    Beverages  Coca-Cola Classic (Large) 30 fl oz cup      280
## 119    Beverages          Dr Pepper (Small) 16 fl oz cup      140
## 120    Beverages         Dr Pepper (Medium) 21 fl oz cup      190
## 121    Beverages          Dr Pepper (Large) 30 fl oz cup      270
```

```
## 127    Beverages             Sprite (Small) 16 fl oz cup      140
## 128    Beverages            Sprite (Medium) 21 fl oz cup      200
## 129    Beverages             Sprite (Large) 30 fl oz cup      280
## 145 Coffee & Tea          Sweet Tea (Child) 12 fl oz cup      110
##     Calories.from.Fat Total.Fat Total.Fat....Daily.Value.
Saturated.Fat
## 111                 0         0                        0
0
## 112                 0         0                        0
0
## 113                 0         0                        0
0
## 119                 0         0                        0
0
## 120                 0         0                        0
0
## 121                 0         0                        0
0
## 127                 0         0                        0
0
## 128                 0         0                        0
0
## 129                 0         0                        0
0
## 145                 0         0                        0
0
##     Saturated.Fat....Daily.Value. Trans.Fat Cholesterol
## 111                             0         0           0
## 112                             0         0           0
## 113                             0         0           0
## 119                             0         0           0
## 120                             0         0           0
## 121                             0         0           0
## 127                             0         0           0
## 128                             0         0           0
## 129                             0         0           0
## 145                             0         0           0
##     Cholesterol....Daily.Value. Sodium Sodium....Daily.Value.
Carbohydrates
## 111                           0      0                      0
39
## 112                           0      5                      0
55
## 113                           0      5                      0
76
## 119                           0     45                      2
37
## 120                           0     65                      3
53
## 121                           0     90                      4
```

```
72
## 127                            0     30                        1
37
## 128                            0     45                        2
54
## 129                            0     60                        3
74
## 145                            0      5                        0
27
##     Carbohydrates....Daily.Value. Dietary.Fiber
Dietary.Fiber....Daily.Value.
## 111                            13             0
0
## 112                            18             0
0
## 113                            25             0
0
## 119                            12             0
0
## 120                            18             0
0
## 121                            24             0
0
## 127                            12             0
0
## 128                            18             0
0
## 129                            25             0
0
## 145                             9             0
0
##     Sugars Protein Vitamin.A....Daily.Value. Vitamin.C....Daily.Value.
## 111     39       0                         0                         0
## 112     55       0                         0                         0
## 113     76       0                         0                         0
## 119     35       0                         0                         0
## 120     51       0                         0                         0
## 121     70       0                         0                         0
## 127     37       0                         0                         0
## 128     54       0                         0                         0
## 129     74       0                         0                         0
## 145     27       0                         0                         0
##     Calcium....Daily.Value. Iron....Daily.Value.
## 111                       0                    0
## 112                       0                    0
## 113                       0                    0
## 119                       0                    0
## 120                       0                    0
## 121                       0                    0
## 127                       0                    0
```

```
## 128                        0                      0
## 129                        0                      0
## 145                        0                      0
```

Se decide eliminar los valores cero de protein de acuerdo a la regla de negocio descrita.

```
# Eliminar registros donde Protein == 0 y Calories > 100
x_clean <- data[!(x == 0 & data$Calories > 100), ]

# Verificar los cambios
summary(x_clean$x)
```

```
## Length  Class   Mode
##      0   NULL   NULL
```

## 5. Utiliza la transformación de Yeo Johnson y encuentra el valor de lambda que maximiza el valor p de la prueba de normalidad que hayas utilizado (Anderson-Darling o Jarque Bera).

Utilizaremos Jaque Bera, ya que en esta se obtuvo el valor p más alto.

```
library(MASS)
library(tseries)
library(VGAM)
```

```
## Warning: package 'VGAM' was built under R version 4.3.3
```

```
## Loading required package: stats4
```

```
## Loading required package: splines
```

```
# Función para aplicar la transformación de Yeo-Johnson y realizar la prueba de Jarque-Bera
yeo_johnson_jb <- function(lambda, data) {
  transformed_data <- yeo.johnson(data, lambda)
  jb_test <- jarque.bera.test(transformed_data)
  return(jb_test$p.value)
}

# Definir el rango de lambda a probar
lambda_seq <- seq(-2, 2, by = 0.01)

# Aplicar la búsqueda del mejor lambda
p_values <- sapply(lambda_seq, yeo_johnson_jb, data = x)

# Encontrar el lambda que maximiza el valor p
lambda_yj <- lambda_seq[which.max(p_values)]
best_p_value <- max(p_values)
```

```r
#
cat("Mejor lambda:", lambda_yj, "\n")

## Mejor lambda: 0.48

cat("Valor p correspondiente:", best_p_value, "\n")

## Valor p correspondiente: 0.6057529

# Aplicar la transformación de Yeo-Johnson con el mejor lambda
x_transformed <- yeo.johnson(x, lambda_yj)
```

**Interpretación del resultado** El valor de $\lambda$ que maximiza el valor p de la prueba de normalidad de Jarque-Bera es 0.48, este valor es la mejor opción para aproximar los datos a una distribución normal según esta prueba.

El valor p de 0.6057529 es relativamente alto y esto sugiere que no hay evidencia suficiente para rechazar la hipótesis nula de que los datos siguen una distribución normal. Esto significa que, después de la transformación de Yeo-Johnson con $\lambda = 0.48$ los datos transformados parecen seguir una distribución normal.

## 6. Escribe la ecuación del modelo encontrado.

$$f(x,\lambda) = \begin{cases} \dfrac{(x+1)^{\lambda} - 1}{\lambda}, & \text{si } x \geq 0 \text{ y } \lambda \neq 0 \end{cases}$$

$$f(x,\lambda) = \begin{cases} \dfrac{(x+1)^{0.48} - 1}{0.48}, & \text{si } x \geq 0 \text{ y } \lambda \neq 0 \end{cases}$$

$x(\lambda)$ es el valor transformado con el parámetro $\lambda$.

$x$ son los datos originales que deseas transformar.

## 7. Analiza la normalidad de las transformaciones obtenidas con los datos originales. Utiliza como argumento de normalidad:

**a) Compara las medidas: Mínimo, máximo, media, mediana, cuartil 1 y cuartil 3, sesgo y curtosis.**

```r
# Resumen de los datos originales
cat("\nMedidas de los datos originales:\n")

##
## Medidas de los datos originales:
```

```r
resumen_x_transformed <- summary(x_transformed)
resumen_x_transformed
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   0.000   2.428   5.053   4.750   6.692  15.786
```

```r
# Sesgo y curtosis de los datos originales
sesgo_x_transformed <- skewness(x_transformed)
curtosis_x_transformed <- kurtosis(x_transformed)

cat("\nSesgo con los datos originales:\n", sesgo_x_transformed, "\n")
```

```
##
## Sesgo con los datos originales:
##  0.09005902
```

```r
cat("\nCurtosis con los datos originales:\n", curtosis_x_transformed,
"\n")
```

```
##
## Curtosis con los datos originales:
##  -0.2655429
```

```r
# Tabla con resultados
resultados_todos <- data.frame(
  "Medida" = c("Min", "1st Qu.", "Median", "Mean", "3rd Qu.", "Max",
"Sesgo", "Curtosis"),
  "Datos_Originales" = c(resumen_x["Min."], resumen_x["1st Qu."],
resumen_x["Median"], resumen_x["Mean"], resumen_x["3rd Qu."],
resumen_x["Max."], sesgo_x, curtosis_x),
  "Box-Cox_Exacto" = c(resumen_trans_exact["Min."],
resumen_trans_exact["1st Qu."], resumen_trans_exact["Median"],
resumen_trans_exact["Mean"], resumen_trans_exact["3rd Qu."],
resumen_trans_exact["Max."], sesgo_trans_exact, curtosis_trans_exact),
  "Box-Cox_Aproximado" = c(resumen_trans_aprox["Min."],
resumen_trans_aprox["1st Qu."], resumen_trans_aprox["Median"],
resumen_trans_aprox["Mean"], resumen_trans_aprox["3rd Qu."],
resumen_trans_aprox["Max."], sesgo_trans_aprox, curtosis_trans_aprox),
  "Yeo Johnson" =  c(resumen_x_transformed["Min."],
resumen_x_transformed["1st Qu."], resumen_x_transformed["Median"],
resumen_x_transformed["Mean"], resumen_x_transformed["3rd Qu."],
resumen_x_transformed["Max."], sesgo_x_transformed,
curtosis_x_transformed)
)

resultados_todos
```

```
##     Medida Datos_Originales Box.Cox_Exacto Box.Cox_Aproximado
Yeo.Johnson
## 1      Min         0.000000     -2.6052632          0.0000000
0.00000000
```

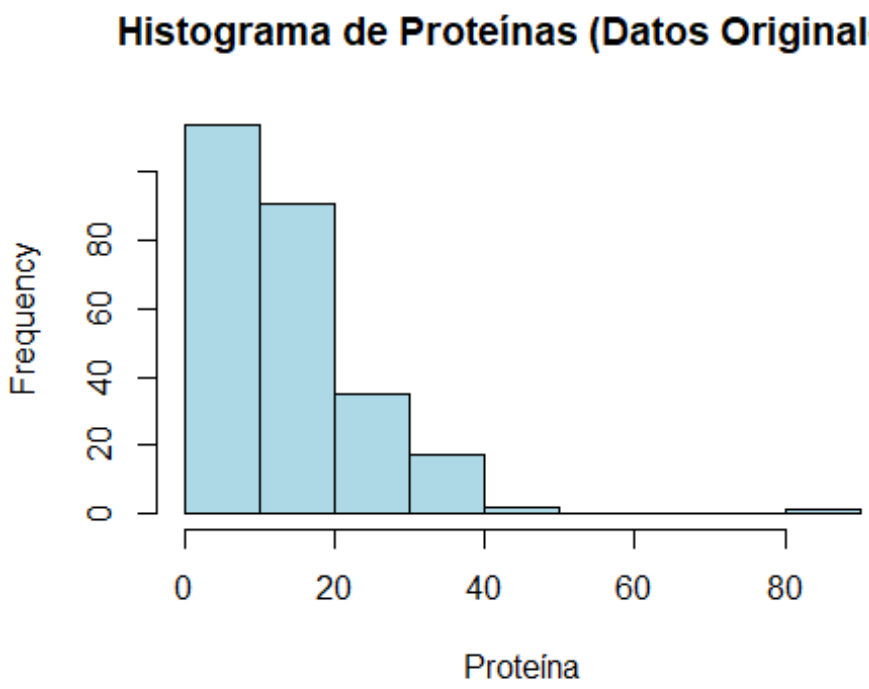
```
## 2  1st Qu.         4.000000      1.8302649          2.0000000
2.42757877
## 3   Median        12.000000      4.1568554          3.4641016
5.05261294
## 4     Mean        13.338462      3.5091457          3.2074020
4.74967685
## 5  3rd Qu.        19.000000      5.4612587          4.3588989
6.69178861
## 6      Max        87.000000     11.8595735          9.3273791
15.78608620
## 7    Sesgo         1.561741     -0.6179156         -0.1736453
0.09005902
## 8 Curtosis         5.795500     -0.1314858         -0.2544237 -
0.26554293
```

**Interpretación de los resultados y comparación con otras transformaciones** La transformación de Yeo-Johnson parece haber sido efectiva para mejorar la normalidad de la distribución de los datos de proteínas.

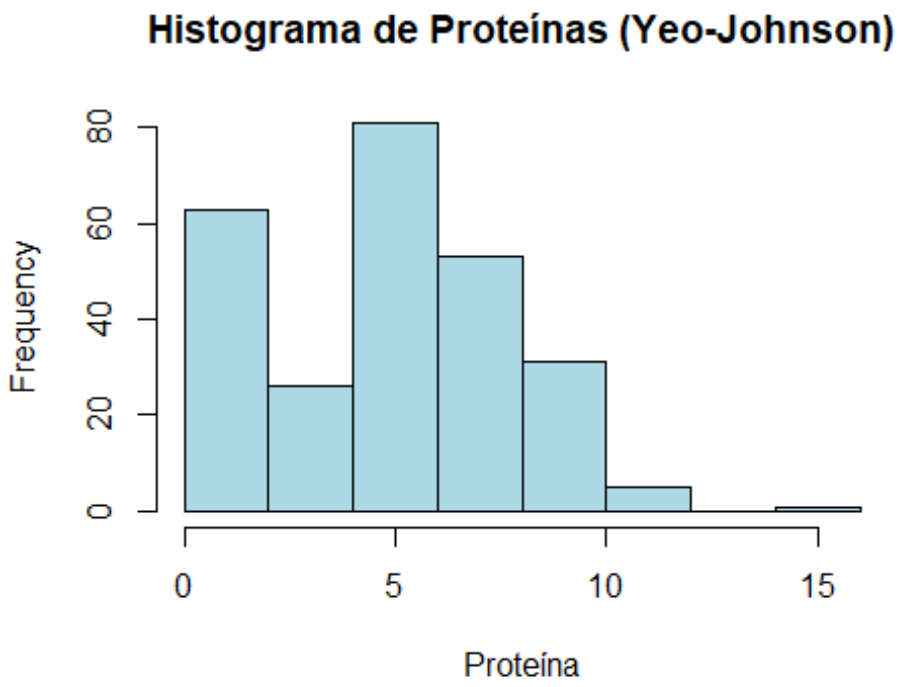
- El sesgo y curtosis en Yeo-Johnson se reduce a casi 0, lo que indica una distribución más normal. Aunque la dispersión de los datos (rango, media, mediana) es mayor para Yeo-Johnson en comparación con las transformaciones Box-Cox, esto sugiere que Yeo-Johnson ha permitido una mayor variabilidad dentro de los datos transformados.

**b) Obten el histograma de los 2 modelos obtenidos (exacto y aproximado) y los datos originales.**

```
# Histograma de los datos originales
hist(x, main = "Histograma de Proteínas (Datos Originales)", xlab =
"Proteína", col = "lightblue", border = "black")
```




```r
# Histograma de los datos con Yeo-Johnson
hist(x_transformed, main = "Histograma de Proteínas (Yeo-Johnson)", xlab
= "Proteína", col = "lightblue", border = "black")
```




**Interpretación de las gráficas** - Los datos originales tienen una distribución altamente sesgada a la derecha, con una gran concentración de valores bajos y una cola larga hacia valores altos.

- La transformación Yeo-Johnson tiene una distribución más simétrica y normal, con valores de proteínas distribuidos de manera más uniforme, aunque todavía hay cierta asimetría, la cola a la derecha es mucho menos pronunciada.

**c) Realiza la prueba de normalidad de Anderson-Darling para los datos transformados y los originales.**

```
# Prueba de normalidad de Anderson-Darling para los datos con Box-Cox
aproximado
ad_test_transformed <- ad.test(x_transformed)
ad_test_transformed

##
##  Anderson-Darling normality test
##
## data:  x_transformed
## A = 2.5718, p-value = 1.661e-06

# Tabla con resultados
resultados_normalidad <- data.frame(
  Test = c("Anderson-Darling", "Anderson-Darling", "Anderson-Darling",
"Jarque-Bera", "Jarque-Bera", "Jarque-Bera", "Anderson-Darling" ),
  Transformation = c("Original", "Box-Cox Exacto", "Box-Cox Aproximado",
"Original", "Box-Cox Exacto", "Box-Cox Aproximado", "Yeo-Johnson"),
```

```
  Statistic = c(ad_test_x$statistic, ad_test_exact$statistic,
ad_test_aprox$statistic,
                jb_test_x$statistic, jb_test_exact$statistic,
jb_test_aprox$statistic, ad_test_transformed$statistic),
  P_Value = c(ad_test_x$p.value, ad_test_exact$p.value,
ad_test_aprox$p.value,
              jb_test_x$p.value, jb_test_exact$p.value,
jb_test_aprox$p.value, ad_test_transformed$p.value)
)

resultados_normalidad
```

```
##               Test     Transformation  Statistic      P_Value
## 1 Anderson-Darling           Original   4.751457 8.515383e-12
## 2 Anderson-Darling     Box-Cox Exacto   6.100798 4.941787e-15
## 3 Anderson-Darling Box-Cox Aproximado   3.094944 8.784536e-08
## 4      Jarque-Bera           Original 479.383612 0.000000e+00
## 5      Jarque-Bera     Box-Cox Exacto  16.867311 2.174253e-04
## 6      Jarque-Bera Box-Cox Aproximado   1.910856 3.846474e-01
## 7 Anderson-Darling        Yeo-Johnson   2.571845 1.661049e-06
```

**Observaciones de los resultados**

- Datos Originales: Ambos tests indican una fuerte no normalidad.
- Box-Cox Exacto: No mejora significativamente la normalidad.
- Box-Cox Aproximado: Mejora significativamente la normalidad, especialmente según la prueba de Jarque-Bera.
- Yeo-Johnson: Mejora la normalidad, pero no tanto como el Box-Cox Aproximado según la prueba de Jarque-Bera.

## 8. Define la mejor transformación de los datos de acuerdo a las características de los modelos que encontraste. Toma en cuenta los criterios del inciso anterior para analizar normalidad y la economía del modelo.

Para determinar la mejor transformación de los datos en términos de normalidad y economía del modelo, se debe considerar las medidas estadísticas, así como los resultados de las pruebas de normalidad.

Basado en la prueba de Jarque-Bera, la transformación de Box-Cox Aproximado es la más efectiva para normalizar los datos de proteínas, ya que tiene el valor p más alto (0.3846474), indicando que los datos transformados no se desvían significativamente de una distribución normal.

La transformación de Yeo-Johnson también mejora la normalidad en comparación con los datos originales y Box-Cox Exacto, pero no es tan efectiva como Box-Cox Aproximado según la prueba de Jarque-Bera.

En cuanto a la económia del modelo, Box-Cox aproximado es simple de aplicar y proporciona una mejora significativa en la normalidad sin introducir complejidad adicional. Por lo tanto, Box-Cox Aproximado es la mejor transformación para los datos de proteínas basándose en las características del modelo y la economía del mismo.

## 9. Concluye sobre las ventajas y desventajas de los modelos de Box Cox y de Yeo Johnson.

### Box-Cox

#### Ventajas

**Normalización de Datos Positivos**: Es muy efectiva para normalizar datos positivos, reduciendo el sesgo y aproximando la distribución a una normal.

**Simplicidad y Comprensión**: Es relativamente fácil de aplicar y entender, con una fórmula simple que se adapta bien a muchas situaciones.

**Parámetro Lambda**: Ofrece flexibilidad en la transformación mediante el ajuste del parámetro lambda.

#### Desventajas

**Limitación a Datos Positivos**: No puede manejar datos que incluyen valores cero o negativos sin modificaciones adicionales.

**Complejidad en Interpretación**: La interpretación de los datos transformados puede ser menos intuitiva en algunos contextos, especialmente cuando lambda es diferente de 1 o 0.

### Yeo-Johnson Transformation

#### Ventajas

**Manejo de Datos Positivos y Negativos**: Puede transformar datos que incluyen valores negativos, positivos y cero, ofreciendo mayor flexibilidad en comparación con Box-Cox.

**Reducción del Sesgo**: Es efectiva en reducir el sesgo y aproximar los datos a una distribución normal, similar a Box-Cox.

**Parámetro Lambda**: Similar a Box-Cox, permite el ajuste del parámetro lambda para optimizar la transformación.

#### *Desventajas*

**Complejidad en Implementación**: Puede ser más compleja de implementar y entender en comparación con Box-Cox, especialmente en términos de la fórmula de transformación.

**Menor Efectividad en Algunos Contextos**: Aunque es versátil, en algunos casos puede no ser tan efectiva como Box-Cox para datos exclusivamente positivos.

**Economía del Modelo**: La interpretación de los datos transformados puede ser aún más compleja que en Box-Cox.

### Conclusión

- Box-Cox es preferible cuando se trabaja con datos exclusivamente positivos y se requiere una transformación simple y eficaz para normalizar la distribución y Yeo-Johnson es una excelente opción cuando los datos contienen una mezcla de valores positivos, negativos y ceros. Por otro lado, Box-Cox ofrece una solución más sencilla, lo que puede resultar en una implementación más económica y rápida a diferencia de Yeo-Johnson, sin embargo esta última ofrece mayor flexibilidad y robustez para datos más diversos

## 10. Analiza las diferencias entre la transformación y el escalamiento de los datos:

**a) Escribe al menos 3 diferencias entre lo que es la transformación y el escalamiento de los datos**

**Objetivo** - En la transformación see modifica la forma de la distribución de los datos para ajustarla a una distribución específica (normal, uniforme, etc.), mientras que en el escalamiento de datos se busca ajustar la escala de los datos para que todas las características contribuyan equitativamente a los análisis posteriores.

**Métodos y Técnicas** - Las transformaciones incluye técnicas como logaritmo, raíz cuadrada, Box-Cox, Yeo-Johnson, que alteran la distribución de los datos. Por el contrario, el escalamiento de datos aplica técnicas como normalización y estandarización, que ajustan la escala de los valores sin cambiar la forma de la distribución.

**Impacto en la Distribución de los Datos** En la transformación cambia la forma de la distribución de los datos, ayudando a cumplir con supuestos estadísticos como la normalidad. El escalamiento de datos no cambia la forma de la distribución, solo ajusta la escala de los valores para asegurar una contribución equitativa de todas las características al modelo.

**b) Indica cuándo es necesario utilizar cada uno** En la transformaciónse utiliza para normalizar la distribución de los datos, reducir el sesgo, estabilizar la varianza y linearizar relaciones.

En el escalamiento de datos se utiliza para igualar la importancia de las características, mejorar la convergencia de algoritmos de optimización, mantener la interpretabilidad de las medidas de distancia y preparar los datos para modelos predictivos.