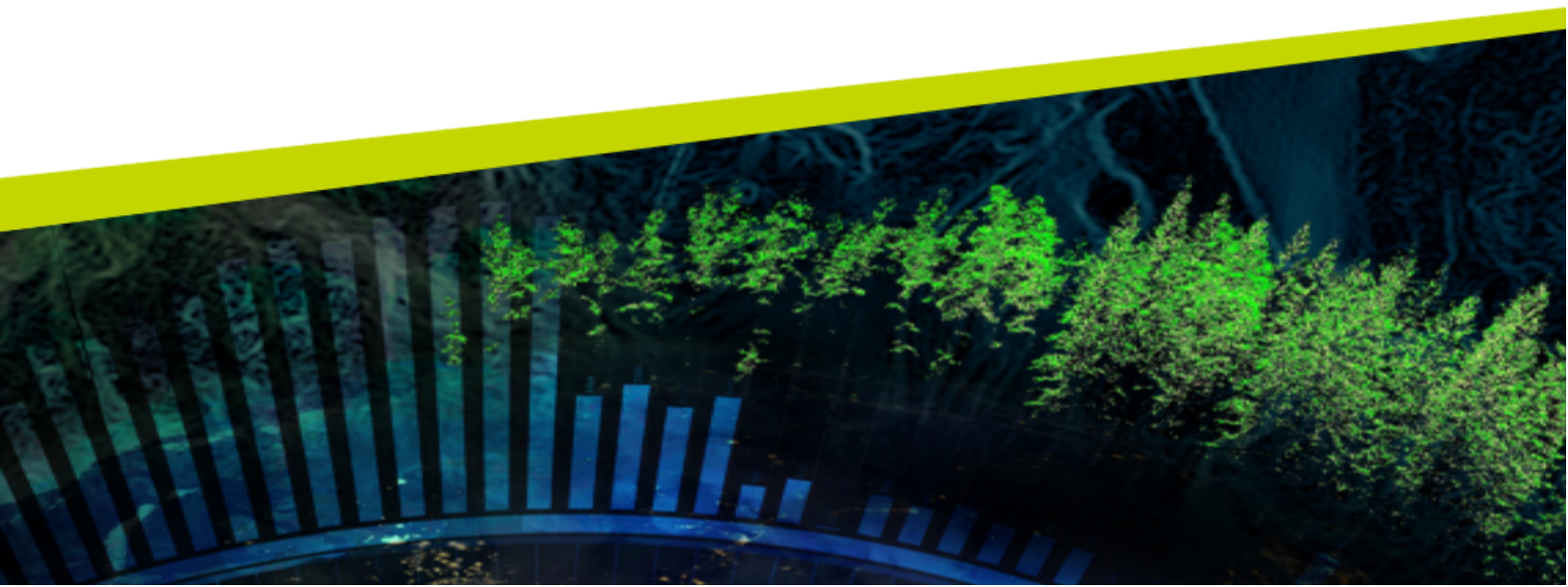
INFERENCIA Y MODELOS ESTADÍSTICOS

Jacqueline Köhler C. y José Luis Jara V.



CAPÍTULO 11. MÉTODOS CLÁSICOS PARA DATOS PROBLEMÁTICOS (VARIABLES NUMÉRICAS)

Como ya sabemos, muchos procedimientos estadísticos requieren que los datos cumplan con ciertas propiedades o condiciones, lo que no siempre ocurre. En este capítulo abordaremos algunos métodos para enfrentar estos problemas cuando se intenta inferir sobre las medias poblacionales de variables aleatorias numéricas.

Para ello nos basaremos principalmente en citet[1,16]lanesfconcepts, Lowry (1999, caps. 11a, 12a, 14a, 15a), Glen (2021b) y Lærd Statistics (2020).

11.1 TRANSFORMACIÓN DE DATOS

Un fenómeno que ocurre a menudo en los estudios, además del incumplimiento de ciertas condiciones, es la necesidad de convertir los datos de una escala a otra diferente. Para hacer tales transformaciones, debemos aplicar una determinada función a una variable aleatoria X, lo que nos entrega como resultado una nueva variable aleatoria Y.

Como explica Lane (s.f., pp. 1, 16), existen diversos métodos que podemos usar para transformar datos, dependiendo de la forma que tengan los datos originales y la que deseemos obtener como resultado. En esta sección conoceremos, entonces, algunas transformaciones de uso frecuente en estadística.

11.1.1 Transformación lineal

Las **transformaciones lineales** son las más sencillas de las transformaciones, y para hacerlas nos basta con aplicar una función lineal, es decir, de la forma presentada en la ecuación 11.1, donde m y n son constantes.

$$y_i = m \cdot x_i + n \tag{11.1}$$

La física nos ofrece muchos escenarios en que es necesario aplicar este tipo de transformaciones, pues es el tipo de operación que realizamos cuando convertimos de una unidad a otra. A modo de ejemplo, consideremos la conversión de grados Celsius a grados Fahrenheit: $F = 1,8 \cdot C + 32$.

En R, podemos hacer este tipo de transformaciones de forma muy sencilla mediante operaciones aritméticas que se aplican vectorialmente, como muestra el script 11.1. En él se transforma un vector con 4 temperaturas en grados Celsius a grados Fahrenheit. El resultado se muestra en la figura 11.1.

Script 11.1: transformación lineal para convertir grados Celsius a grados Fahrenheit.

```
# Crear un vector con cuatro observaciones en grados Celsius.
Celsius <- c(-8, 0, 29.8, 100)

# Aplicar transformación lineal para convertir a grados Fahrenheit.
Fahrenheit <- 1.8 * Celsius + 32

# Mostrar los resultados.
cat("Temperaturas en grados Celsius\n")
print(Celsius)
cat("\nTemperaturas en grados Fahrenheit\n")
print(Fahrenheit)
```

```
Temperaturas en grados Celsius
[1]  -8.0   0.0  29.8 100.0

Temperaturas en grados Fahrenheit
[1]  17.60  32.00  85.64 212.00
```

Figura 11.1: resultado de la transformación lineal del script 11.1.

11.1.2 Transformación logarítmica

La transformación logarítmica nos puede servir cuando tenemos distribuciones muy asimétricas, pues ayuda a reducir la desviación y así facilita el cumplimiento de la condición de normalidad requerida por muchas de las pruebas estadísticas que ya conocemos. Para ver este efecto de manera más clara, usaremos un conjunto de datos que registra el peso corporal (en kilogramos) y el peso del cerebro (en gramos) de diversos animales, algunos de ellos extintos (Rousseeuw & Leroy, 1987, p. 57). En R, esta transformación puede hacerse gracias a la función `log(x, base)`, aunque debemos tener cuidado con posibles valores iguales a 0. El script 11.2 aplica esta transformación al peso corporal y al peso cerebral de los animales (líneas 22–23). La figura 11.2 (script 11.2, líneas 28–43) muestra gráficamente el resultado de esta transformación para el peso cerebral de los animales.

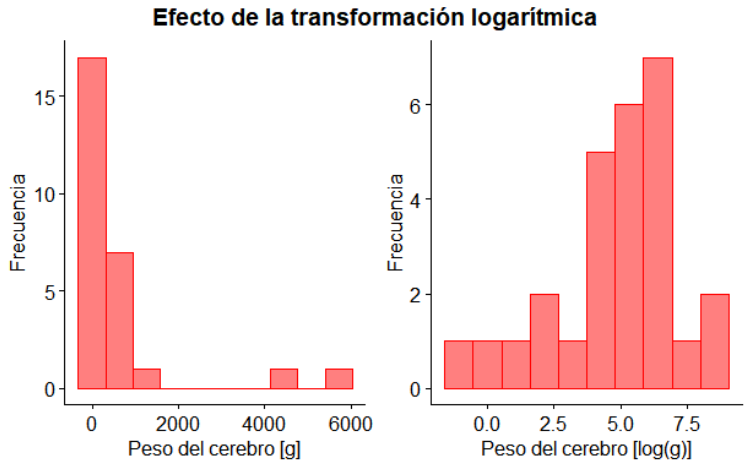


Figura 11.2: histogramas del peso cerebral antes y después de la transformación logarítmica.

Muchas veces la transformación logarítmica hace que nos sea más fácil interpretar los datos, evidenciando patrones más claros para la relación entre variables. En la figura 11.3 (script 11.2, líneas 47–60), a la derecha, se evidencia una fuerte relación entre el peso corporal y el peso del cerebro tras transformar ambas variables, relación que no podemos percibir con los datos originales (izquierda).

Script 11.2: transformación logarítmica.

```
1  library(ggpubr)
2
3  # Cargar datos
4  animal <- c("Mountain beaver", "Cow", "Grey wolf", "Goat", "Guinea pig",
5              "Dipliodocus", "Asian elephant", "Donkey", "Horse",
6              "Potar monkey", "Cat", "Giraffe", "Gorilla", "Human",
7              "African elephant", "Triceratops", "Rhesus monkey", "Kangaroo",
8              "Golden hamster", "Mouse", "Rabbit", "Sheep", "Jaguar",
9              "Chimpanzee", "Brachiosaurus", "Mole", "Pig")
10
11 body_weight <- c(1.35, 465, 36.33, 27.66, 1.04, 11700, 2547, 187.1, 521, 10,
12                  3.3, 529, 207, 62, 6654, 9400, 6.8, 35, 0.12, 0.023, 2.5,
```

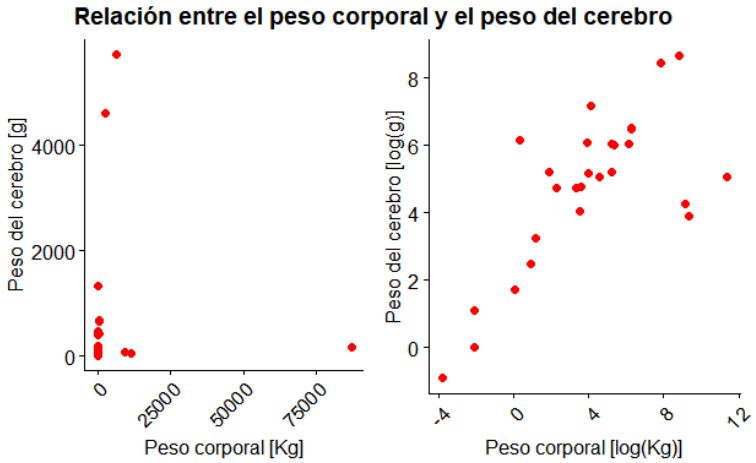


Figura 11.3: gráficos de dispersión para el peso corporal y el peso del cerebro antes y después de las transformaciones logarítmicas.

```
                  55.5, 100, 52.16, 87000, 0.122, 192)

brain_weight <- c(465, 423, 119.5, 115, 5.5, 50, 4603, 419, 655, 115, 25.6,
                  680, 406, 1320, 5712, 70, 179, 56, 1, 0.4, 12.1, 175, 157,
                  440, 154.5, 3, 180)

datos <- data.frame(animal, body_weight, brain_weight)

# Aplicar transformación logarítmica
log_cuerpo <- log(body_weight)
log_cerebro <- log(brain_weight)
datos <- data.frame(datos, log_cuerpo, log_cerebro)

# Histogramas para el peso cerebral antes y después de la transformación
# logarítmica.
g3 <- gghistogram(datos, x = "brain_weight", bins = 10,
                  xlab = "Peso del cerebro [g]", ylab = "Frecuencia",
                  color = "red", fill = "red")

g4 <- gghistogram(datos, x = "log_cerebro", bins = 10,
                  xlab = "Peso del cerebro [log(g)]", ylab = "Frecuencia",
                  color = "red", fill = "red")

# Crear una única figura con ambos histogramas.
histograma  <- ggarrange(g3, g4, ncol = 2, nrow = 1)
titulo  <- text_grob("Efecto de la transformación logarítmica",
                     face = "bold", size = 14)

histograma <- annotate_figure(histograma, top = titulo)
print(histograma)

# Gráficos de dispersión para la relación entre peso corporal y peso del
# cerebro, antes y después de aplicar la transformación logarítmica.
g1 <- ggscatter(datos, x = "body_weight", y = "brain_weight",
                color = "red", xlab = "Peso corporal [Kg]",
                ylab = "Peso del cerebro [g]") + rotate_x_text(45)

g2 <- ggscatter(datos, x = "log_cuerpo", y = "log_cerebro",
                color = "red", xlab = "Peso corporal [log(Kg)]",
```

```
                ylab = "Peso del cerebro [log(g)]") + rotate_x_text(45)

# Crear una única figura con los gráficos de dispersión.
dispersion  <- ggarrange(g1, g2, ncol = 2, nrow = 1)
texto <- "Relación entre el peso corporal y el peso del cerebro"
titulo  <- text_grob(texto, face = "bold", size = 14)
dispersion <- annotate_figure(dispersion, top = titulo)

print(dispersion)
```

Eso sí, tenemos que ser muy cuidadosos al usar esta transformación, porque cuando comparamos medias de datos tras una transformación logarítmica, ¡en realidad estamos comparando ¡**medias geométricas**! Recordemos que la media geométrica se calcula de acuerdo a la ecuación 11.2 y suele ocuparse para representar tasas de crecimiento o de interés (Glen, 2021a).

$$\left(\prod_{i=1}^{n} x_i\right)^{\frac{1}{n}} = \sqrt[n]{x_1 \cdot x_2 \cdot \ldots \cdot x_n} \tag{11.2}$$

Para ilustrar esta idea, supongamos que aplicamos una transformación logarítmica con base 10 al vector $[1, 10, 100]$, obteniendo como resultado $[0, 1, 2]$.

La media aritmética del vector transformado es:

$$\frac{0+1+2}{3} = 1$$

Al revertir la transformación para la media, tenemos que:

$$10^1 = 10$$

Lo que es distinto a la media aritmética de los datos originales:

$$\frac{1+10+100}{3} = 37$$

A su vez, la media geométrica del vector original es:

$$\sqrt[3]{1 \cdot 10 \cdot 100} = 10$$

Así, si dos variables a las que se ha aplicado la transformación logarítmica tienen igual media, entonces **las medias geométricas de las variables originales son iguales**.

11.1.3 Escalera de potencias de Tukey

Más general que la transformación logarítmica, la **escalera de potencias de Tukey** nos ayuda a cambiar la forma de una distribución asimétrica para que se asemeje a la normal. Este método consiste en explorar relaciones de la forma que muestra la ecuación 11.3, donde λ puede tomar cualquier valor real y se escoge de modo que la distribución de los datos transformados sea lo más cercana a la normal posible. También es útil al explorar la relación entre dos variables, en cuyo caso se busca obtener un gráfico de dispersión en que los puntos se asemejen a una recta.

$$y = x^{\lambda} \tag{11.3}$$

Formalmente, la transformación de Tukey se define según la ecuación 11.4, aunque (por la falta de computadores) suelen usarse únicamente aquellas que se muestran en la tabla 11.1. Fijémonos en que si $\lambda = 1$, no

se realiza transformación alguna, y que para el caso de $\lambda = 0$, se tiene que $x^0 = 1$, por lo que se reemplaza en este caso por la transformación logarítmica.

$$\tilde{x}_\lambda = \begin{cases} x^\lambda & \lambda > 0 \\ \log(x) & \lambda = 0 \\ -(x^\lambda) & \lambda < 0 \end{cases} \tag{11.4}$$

λ	-2	-1	$-\frac{1}{2}$	0	$\frac{1}{2}$	1	2
$\tilde{x}$	$-\frac{1}{x^2}$	$-\frac{1}{x}$	$-\frac{1}{\sqrt{x}}$	$log(x)$	$\sqrt{x}$	x	x^2

Tabla 11.1: escalera de transformaciones de Tukey.

Usemos ahora la población total de Estados Unidos entre los años 1610 y 1850 (United States Census Bureau, 2004, 2021) como ejemplo para entender mejor esta transformación. La figura 11.4 (líneas 18–30 del script 11.3) muestra, a la izquierda, el histograma para la población (en millones de habitantes) y, a la derecha, un gráfico de dispersión para la población por año. Podemos ver claramente que la distribución de la población presenta una fuerte asimetría hacia la izquierda y que la población parece aumentar de manera exponencial durante ese periodo.

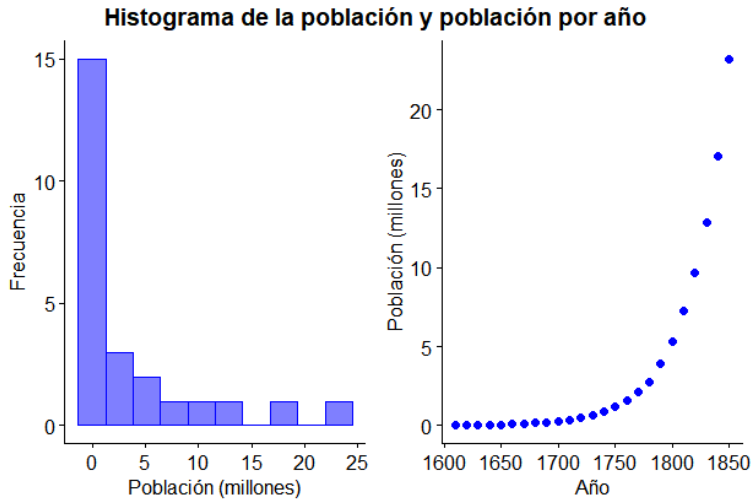


Figura 11.4: histograma de la población histórica de Estados Unidos y gráfico de dispersión de la población por año.

La figura 11.5 (líneas 46–75 del script 11.3) muestra los gráficos de dispersión para la población por año tras aplicar la transformación de Tukey con diferentes valores de λ a la población. En ella podemos observar que la curva cambia gradualmente de convexa a cóncava a medida que aumenta el valor de λ, lo que deja entrever que, para obtener un resultado que sea lo más cercano posible a una línea recta, tenemos que resolver un problema de optimización que minimice los valores residuales tras ajustar una recta a los puntos transformados. De las transformaciones presentadas en la figura 11.5, la más cercana a una recta se obtiene para $\lambda = 0$.

En párrafos anteriores mencionamos que la transformación de Tukey también permite reducir la asimetría en la distribución de los datos, como muestra la figura 11.6 (líneas 79–108 del script 11.3). En ella podemos notar que, a medida que λ aumenta, se reduce la asimetría negativa.

Como podemos suponer, reducir la asimetría nos ayuda a cumplir el requisito de normalidad que imponen muchas pruebas estadísticas, permitiéndonos así lograr resultados teóricamente más precisos. Sin embargo, una vez más tenemos que ser cuidadosos y tener en cuenta la transformación realizada al momento de interpretar los resultados. Si bien tenemos certeza que si se encuentran diferencias significativas en la variable

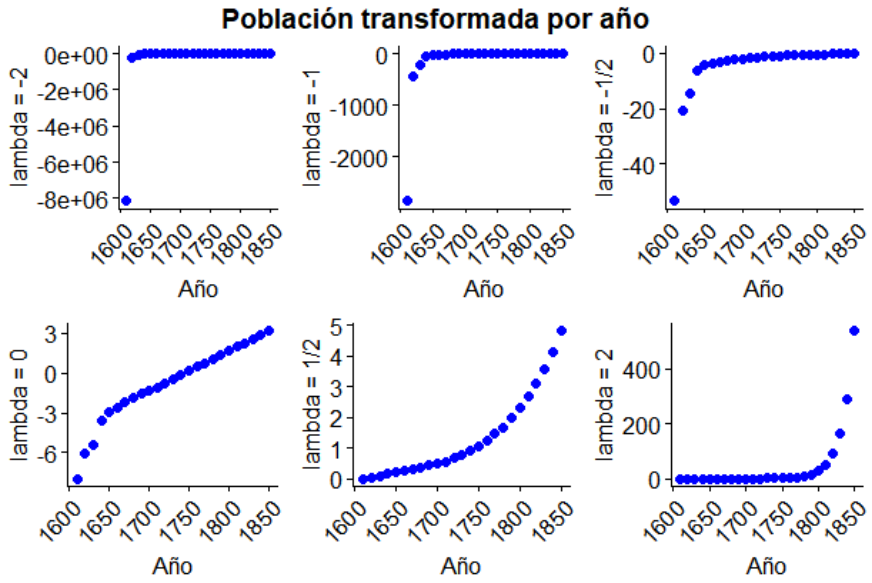


Figura 11.5: población de Estados Unidos por año tras aplicar la transformación de Tukey con distintos valores de λ.

transformada, estas diferencias **también existen en la variable original**, los estadísticos y los intervalos de confianza ¡**no son los mismos** que arrojarían las pruebas con los datos originales!

En R, la función `transformTukey(x, start, end, int, plotit, verbose, quiet, statistic, returnLambda)` incluida en el paquete `rcompanion` permite aplicar la escalera fácilmente, donde:

- `x`: vector de valores a transformar.
- `start`: valor inicial de λ a evaluar.
- `end`: valor final de λ a evaluar.
- `int`: intervalo entre los valores de λ a evaluar.
- `plotit`: si toma valor `TRUE`, entrega los siguientes gráficos:
  - Estadístico de la prueba de normalidad versus λ.
  - Histograma de los valores transformados.
  - Gráfico Q-Q de los valores transformados.
- `verbose`: si toma valor `TRUE`, muestra información adicionar sobre la prueba de normalidad con respecto a λ.
- `quiet`: si toma valor `TRUE`, no muestra información alguna por pantalla.
- `statistic`: si toma valor 1, usa la prueba de normalidad de Shapiro-Wilk. Con valor 2, usa la prueba de Anderson-Darling.
- `returnLambda`: si toma valor `TRUE`, devuelve el valor de λ. Si toma valor `FALSE`, devuelve los datos transformados.

Tras llamar a la función `transformTukey()` con los datos de la población de Estados Unidos (script 11.3, líneas 111–112), obtenemos que el valor óptimo de λ es 0,12 y los gráficos de la figura 11.7. El gráfico 11.7a nos muestra que el valor óptimo de λ es aquel que maximiza el estadístico entregado por la prueba de normalidad. A su vez, en la figura 11.7b vemos que la distribución obtenida con $\lambda = 0,12$ se asemeja mucho más a la normal que la de los datos originales o que cualquiera de las presentadas en la figura 11.6, lo que se ve confirmado por el gráfico Q-Q de la figura 11.7c.

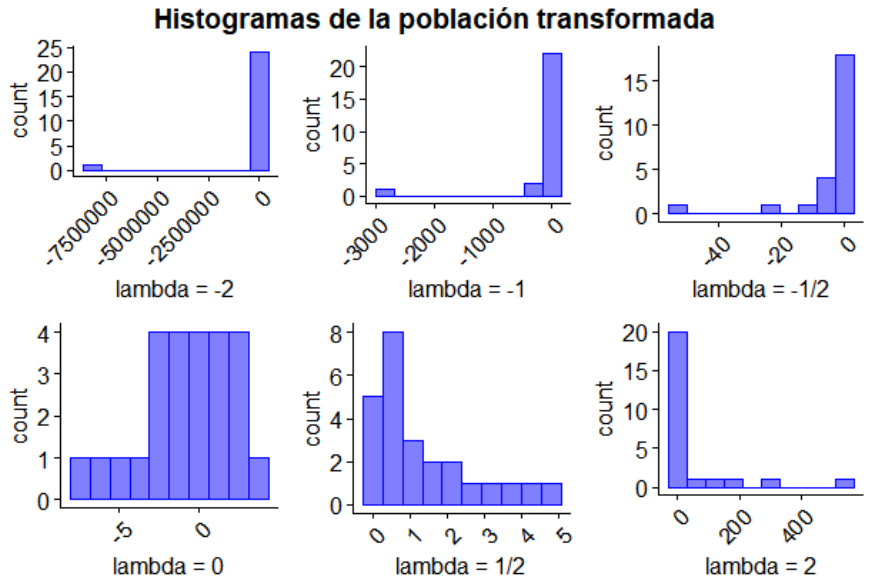
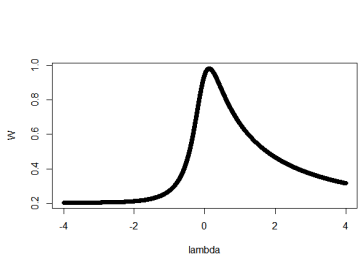
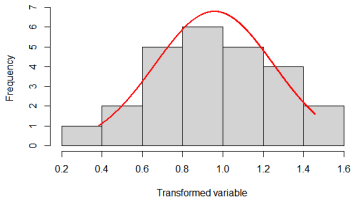


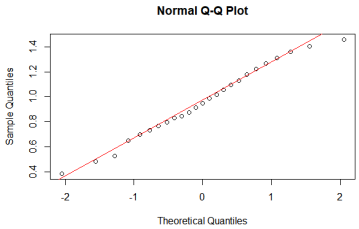
Figura 11.6: histograma de la población de Estados Unidos tras aplicar la transformación de Tukey con distintos valores de λ.



(a) estadístico W de la prueba de Shapiro-Wilk por cada valor de λ.



(b) histograma de la población transformada con el valor óptimo de λ.



(c) gráfico Q-Q de la población transformada con el valor óptimo de λ.

Figura 11.7: gráficos entregados por `transformTukey()`.

Script 11.3: transformación de Tukey para la población total de Estados Unidos.

```
library(ggpubr)
library(rcompanion)

# Cargar datos
Year <- c(1610, 1620, 1630, 1640, 1650, 1660, 1670, 1680, 1690, 1700, 1710,
          1720, 1730, 1740, 1750, 1760, 1770, 1780, 1790, 1800, 1810, 1820,
          1830, 1840, 1850)

Population <- c(0.00035, 0.002302, 0.004646, 0.026634, 0.050368, 0.075058,
                0.111935, 0.151507, 0.210372, 0.250888, 0.331711, 0.466185,
                0.629445, 0.905563, 1.17076, 1.593625, 2.148076, 2.780369,
                3.929214, 5.308483, 7.239881, 9.638453, 12.86602, 17.069453,
                23.191876)

datos <- data.frame(Year, Population)

# Gráfico de dispersión e histograma.
g1 <- gghistogram(datos, x = "Population", bins = 10,
```

```
                  xlab = "Población (millones)", ylab = "Frecuencia",
                  color = "blue", fill = "blue")

g2 <- ggscatter(datos, x = "Year", y = "Population", color = "blue",
                xlab = "Año", ylab = "Población (millones)")

# Histograma de la población y población por año
original  <- ggarrange(g1, g2, ncol = 2, nrow = 1)
texto <- "Histograma de la población y población por año"
titulo  <- text_grob(texto, face = "bold", size = 14)
original <- annotate_figure(original, top = titulo)
print(original)

# Transformaciones de la población
lambda_menos_dos <- -1 / (datos$Population ** 2)
lambda_menos_uno <- -1 / datos$Population
lambda_menos_un_medio <- -1 / sqrt(datos$Population)
lambda_cero <- log(datos$Population)
lambda_un_medio <- sqrt(datos$Population)
lambda_dos <- datos$Population ** 2

transformaciones <- data.frame(datos, lambda_menos_dos, lambda_menos_uno,
                               lambda_menos_un_medio, lambda_cero,
                               lambda_un_medio, lambda_dos)

# Gráficos de dispersión para la transformación de Tukey de la población y el
# año, usando distintos valores de lambda.
gt1 <- ggscatter(transformaciones, x = "Year", y = "lambda_menos_dos",
                 color = "blue", xlab = "Año",
                 ylab = "lambda = -2") + rotate_x_text(45)

gt2 <- ggscatter(transformaciones, x = "Year", y = "lambda_menos_uno",
                 color = "blue", xlab = "Año",
                 ylab = "lambda = -1") + rotate_x_text(45)

gt3 <- ggscatter(transformaciones, x = "Year", y = "lambda_menos_un_medio",
                 color = "blue", xlab = "Año",
                 ylab = "lambda = -1/2") + rotate_x_text(45)

gt4 <- ggscatter(transformaciones, x = "Year", y = "lambda_cero",
                 color = "blue", xlab = "Año",
                 ylab = "lambda = 0") + rotate_x_text(45)

gt5 <- ggscatter(transformaciones, x = "Year", y = "lambda_un_medio",
                 color = "blue", xlab = "Año",
                 ylab = "lambda = 1/2") + rotate_x_text(45)

gt6 <- ggscatter(transformaciones, x = "Year", y = "lambda_dos",
                 color = "blue", xlab = "Año",
                 ylab = "lambda = 2") + rotate_x_text(45)

# Crear una única figura con todos los gráficos de dispersión.
dispersion  <- ggarrange(gt1, gt2, gt3, gt4, gt5, gt6, ncol = 3, nrow = 2)
texto <- "Población transformada por año"
titulo  <- text_grob(texto, face = "bold", size = 14)
dispersion <- annotate_figure(dispersion, top = titulo)
print(dispersion)

# Histogramas para la transformación de Tukey de la población y el año,
# usando distintos valores de lambda.
```

```
h1 <- gghistogram(transformaciones, bins = 10, x = "lambda_menos_dos",
                  color = "blue", fill = "blue",
                  xlab = "lambda = -2") + rotate_x_text(45)

h2 <- gghistogram(transformaciones, bins = 10, x = "lambda_menos_uno",
                  color = "blue", fill = "blue",
                  xlab = "lambda = -1") + rotate_x_text(45)

h3 <- gghistogram(transformaciones, bins = 10, x = "lambda_menos_un_medio",
                  color = "blue", fill = "blue",
                  xlab = "lambda = -1/2") + rotate_x_text(45)

h4 <- gghistogram(transformaciones, bins = 10, x = "lambda_cero",
                  color = "blue", fill = "blue",
                  xlab = "lambda = 0") + rotate_x_text(45)

h5 <- gghistogram(transformaciones, bins = 10, x = "lambda_un_medio",
                  color = "blue", fill = "blue",
                  xlab = "lambda = 1/2") + rotate_x_text(45)

h6 <- gghistogram(transformaciones, bins = 10, x = "lambda_dos",
                  color = "blue", fill = "blue",
                  xlab = "lambda = 2") + rotate_x_text(45)

# Crear una única figura con todos los gráficos de dispersión.
histograma  <- ggarrange(h1, h2, h3, h4, h5, h6, ncol = 3, nrow = 2)
texto <- "Histogramas de la población transformada"
titulo  <- text_grob(texto, face = "bold", size = 14)
histograma <- annotate_figure(histograma, top = titulo)
print(histograma)

# Buscar la mejor transformación de Tukey usando una función de R.
transformacion <- transformTukey(datos$Population, start = -4, end = 4,
                                 int = 0.001, returnLambda = TRUE)
```

11.1.4 Transformaciones Box-Cox

La **transformación Box-Cox** es una versión escalada de la transformación de Tukey, dada por la ecuación 11.5:

$$x'_\lambda = \frac{x^\lambda - 1}{\lambda} \tag{11.5}$$

Si bien a primera vista parece muy diferente a la ecuación 11.4, podemos reescribirla como:

$$x'_\lambda = \frac{x^\lambda - 1}{\lambda} \approx \frac{\left(1 + \lambda \log(x) + \frac{1}{2}\lambda^2 \log(x)^2 + \ldots\right) - 1}{\lambda}$$

De donde:

$$\lim_{\lambda \to 0} \frac{\left(1 + \lambda \log(x) + \frac{1}{2}\lambda^2 \log(x)^2 + \ldots\right) - 1}{\lambda} = \frac{0}{0}$$

Tras aplicar la regla de l'Hôpital, obtenemos finalmente que:

$$\lim_{\lambda \to 0} x'_\lambda = \log(x)$$

Por lo que, al igual que en la escalera de potencias de Tukey, pero de forma más natural, empleamos la transformación logarítmica para $\lambda = 0$.

La figura 11.8 (creada con el script 11.4, líneas 39–64) muestra los gráficos de dispersión para la población total de Estados Unidos por año tras aplicar la transformación de Box-Cox con diferentes valores de λ. Podemos ver que el resultado se parece al que obtuvimos con la transformación de Tukey (figura 11.5).

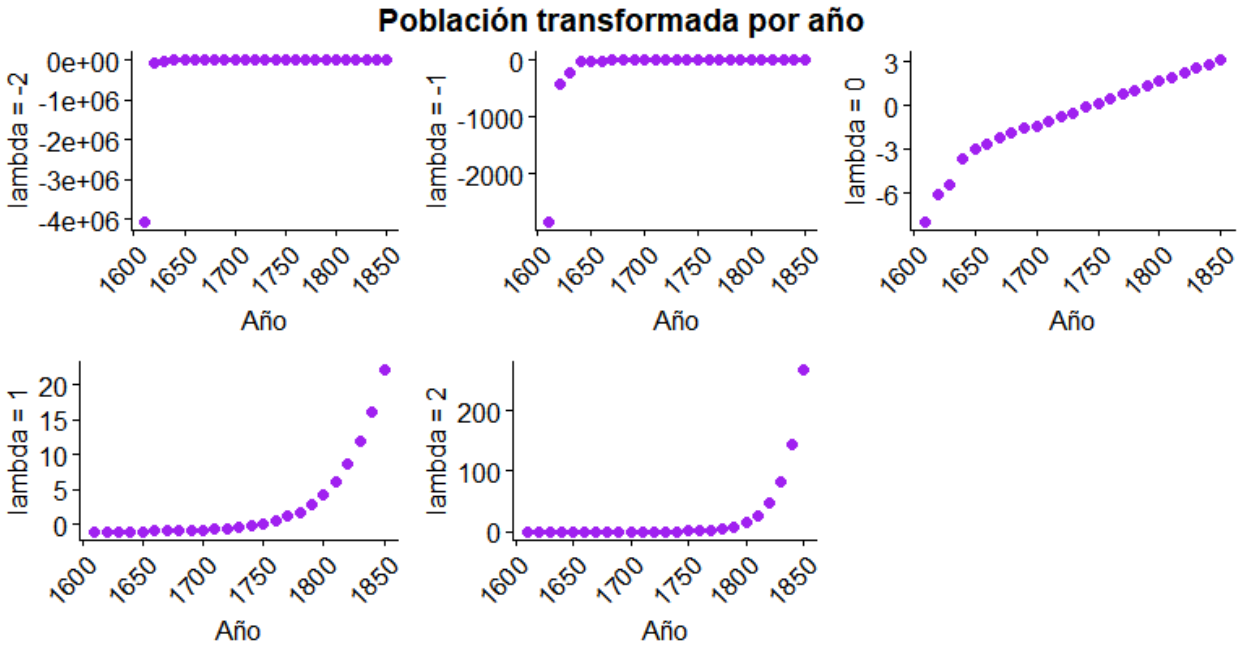


Figura 11.8: población de Estados Unidos por año tras aplicar la transformación de Box-Cox con distintos valores de λ.

Una característica interesante de esta transformación es que, para cualquier valor de λ, $x'_\lambda = 0$ cuando $x = 1$. Podemos observar esto claramente en la figura 11.9, que compara transformaciones Box-Cox con distintos valores de λ con $\log(x)$.

El paquete `DescTools` de R incluye varias funciones que permiten efectuar la transformación Box-Cox (Carchedi [...], s.f.). Destacan entre ellas:

- `BoxCoxLambda(x, lower, upper)`: devuelve el valor óptimo de λ para la transformación Box-Cox del vector `x`.
- `BoxCox(x, lambda)`: devuelve un vector correspondiente a la transformación Box-Cox de `x` con parámetro `lambda`.
- `BoxCoxInv(x, lambda)`: revierte la transformación Box-Cox del vector `x` con parámetro `lambda`.

Donde:

- `x`: vector numérico.
- `lower`: limite inferior para los posibles valores de λ.
- `upper`: limite superior para los posibles valores de λ.
- `lambda`: parámetro de la transformación.

En las líneas 67–70 del script 11.4 se determina el valor óptimo del parámetro λ, obteniéndose como resultado $\lambda = 0,09942656$, para luego efectuar la transformación Box-Cox correspondiente de la población de Estados Unidos. La figura 11.10 (script 11.4, líneas 84–87) muestra gráficamente el resultado de la transformación. En el gráfico 11.10a podemos ver que la relación entre la población transformada y el año se asemeja a una recta, mientras que el histograma de la figura 11.10b se parece bastante a una distribución normal, hecho que vemos confirmado por el gráfico Q-Q de la figura 11.10c.

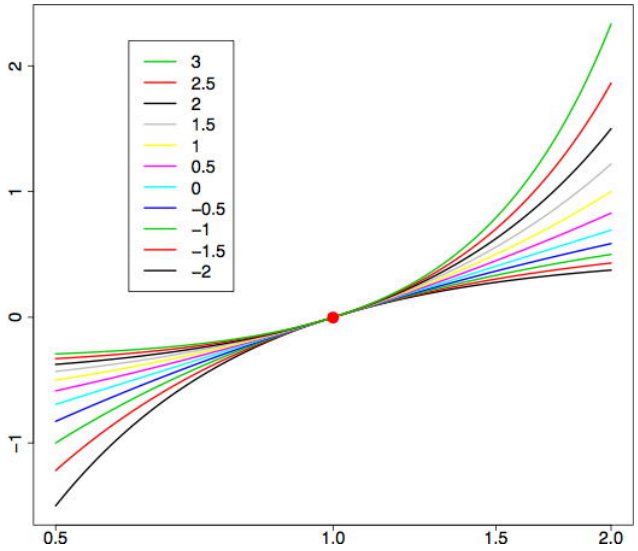
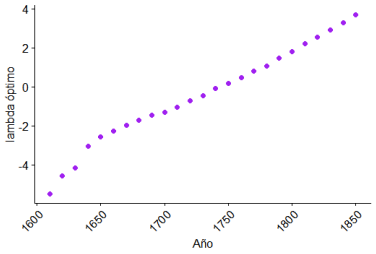
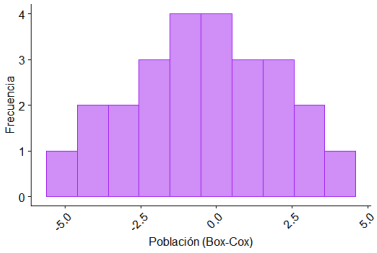


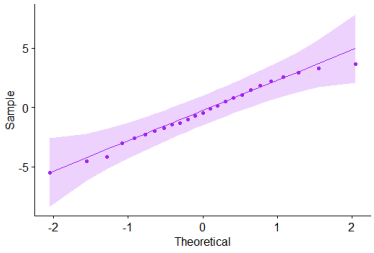
Figura 11.9: ejemplos de la transformación Box-Cox versus $\log(x)$. Fuente: (Lane, s.f., p. 16).



(a) población de Estados Unidos por año tras aplicar la transformación de Box-Cox con λ óptimo.



(b) histograma de la población transformada con el valor óptimo de λ.



(c) gráfico Q-Q de la población transformada con el valor óptimo de λ.

Figura 11.10: gráficos de población de Estados Unidos por año usando la transformación de Box-Cox.

Script 11.4: transformación de Box-Cox para la población total de Estados Unidos.

```
library(ggpubr)
library(DescTools)

# Cargar datos
Year <- c(1610, 1620, 1630, 1640, 1650, 1660, 1670, 1680, 1690, 1700, 1710,
          1720, 1730, 1740, 1750, 1760, 1770, 1780, 1790, 1800, 1810, 1820,
          1830, 1840, 1850)

Population <- c(0.00035, 0.002302, 0.004646, 0.026634, 0.050368, 0.075058,
                0.111935, 0.151507, 0.210372, 0.250888, 0.331711, 0.466185,
                0.629445, 0.905563, 1.17076, 1.593625, 2.148076, 2.780369,
                3.929214, 5.308483, 7.239881, 9.638453, 12.86602, 17.069453,
                23.191876)

datos <- data.frame(Year, Population)

# Transformación de Box-cox
box_cox <- function(x, lambda) {
  if(lambda == 0) {
    return(log(x))
```

```
  }

  resultado <- (x ** lambda -1) / lambda
  return(resultado)
}

# Transformaciones de la población
lambda_menos_dos <- box_cox(datos$Population, -2)
lambda_menos_uno <- box_cox(datos$Population, -1)
lambda_cero <- box_cox(datos$Population, 0)
lambda_uno <- box_cox(datos$Population, 1)
lambda_dos <- box_cox(datos$Population, 2)

transformaciones <- data.frame(datos, lambda_menos_dos, lambda_menos_uno,
                               lambda_cero, lambda_uno, lambda_dos)

# Gráficos de dispersión para la transformación de Box-Cox de la población y
# el año, usando distintos valores de lambda.
gt1 <- ggscatter(transformaciones, x = "Year", y = "lambda_menos_dos",
                 color = "purple", xlab = "Año",
                 ylab = "lambda = -2") + rotate_x_text(45)

gt2 <- ggscatter(transformaciones, x = "Year", y = "lambda_menos_uno",
                 color = "purple", xlab = "Año",
                 ylab = "lambda = -1") + rotate_x_text(45)

gt3 <- ggscatter(transformaciones, x = "Year", y = "lambda_cero",
                 color = "purple", xlab = "Año",
                 ylab = "lambda = 0") + rotate_x_text(45)

gt4 <- ggscatter(transformaciones, x = "Year", y = "lambda_uno",
                 color = "purple", xlab = "Año",
                 ylab = "lambda = 1") + rotate_x_text(45)

gt5 <- ggscatter(transformaciones, x = "Year", y = "lambda_dos",
                 color = "purple", xlab = "Año",
                 ylab = "lambda = 2") + rotate_x_text(45)

# Crear una única figura con todos los gráficos de dispersión.
dispersion  <- ggarrange(gt1, gt2, gt3, gt4, gt5, ncol = 3, nrow = 2)
texto <- "Población transformada por año"
titulo  <- text_grob(texto, face = "bold", size = 14)
dispersion <- annotate_figure(dispersion, top = titulo)
print(dispersion)

# Buscar la mejor transformación Box-Cox usando funciones de R.
lambda <- BoxCoxLambda(datos$Population, lower = -4, upper = 4)
cat("Lambda óptimo:", lambda)
transformacion <- BoxCox(datos$Population, lambda)
datos <- data.frame(datos, transformacion)

# Graficar los datos transformados.
g1 <- ggqqplot(transformacion, color = "purple")
print(g1)

g2 <- gghistogram(datos, bins = 10, x = "transformacion", color = "purple",
                  fill = "purple", xlab = "Población (Box-Cox)",
                  ylab = "Frecuencia") + rotate_x_text(45)

print(g2)
```

```r

# Gráfico de dispersión para la transformación de Box-Cox de la población y
# el año, usando lambda óptimo.
g3 <- ggscatter(datos, x = "Year", y = "transformacion", color = "purple",
                xlab = "Año", ylab = "lambda óptimo") + rotate_x_text(45)

print(g3)
```

11.1.5 Ejercicios propuestos

1. En la década del 1920 se hicieron los primeros estudios sobre la relación entre la velocidad de un automóvil con la distancia que necesita para detenerse. Los datos de estas pruebas se pueden encontrar el conjunto `cars` del paquete `datasets`. Con ellos, responde la siguiente pregunta: en promedio, ¿se requieren más de 40 pies para detener un vehículo que viaja a más de 10 millas por hora?
2. El conjunto `airquality` del paquete `datasets` contiene mediciones diarias de la calidad del aire en la ciudad de New York, EE.UU., registradas de mayo a septiembre de 1973.
   a) Verifica si las mediciones de ozono [partes por billón] son, en promedio, iguales en mayo y junio. Usa la escalera de potencias de Tukey si los datos presentan problemas para su análisis.
   b) Verifica si las mediciones de ozono [partes por billón] son, en promedio, iguales en agosto y septiembre. Usa la transformación Box-Cox si los datos presentan problemas para su análisis.
3. El conjunto `WorldPhones` del paquete `datasets` contiene la cantidad de teléfonos (en miles) instalados para diferentes regiones del planeta en algunos años entre 1951 y 1961.
   a) Verifica si, en promedio, la cantidad de teléfonos instalados en aquella época eran iguales en Centroamérica, Sudamérica y África. Usa la escalera de potencias de Tukey si los datos presentan problemas para su análisis.
   b) Verifica si, en promedio, los datos sugieren que la cantidad de teléfonos instalados en Asia, Oceanía y África era similares por aquellos años. Usa la transformación Box-Cox si los datos presentan problemas para su análisis.