Tema 1 — Introducción al Tratamiento de Datos (ejemplos con R base)

Tratamiento de Datos · Grado en Ciencia de Datos (UV)

Marcelino Martínez

2026-02-04

1. Introducción al Tratamiento de Datos

Este documento acompaña al **Tema 1** (PDF de teoría) con ejemplos **exclusivamente en R base** (sin tidyverse, sin paquetes externos) para ilustrar:

- Por qué analizamos datos.
- Visión global de un problema de tratamiento de datos.
- Primeros pasos con un dataset.
- Caracterización: estadísticos, valores perdidos/anómalos, transformaciones.
- Idea de **codebook (metadata)**.
- Las **4R** del Análisis Exploratorio (Tukey): *Residuos, Reexpresión, Resistencia, Revelación.*

  Nota: La visualización "avanzada" se verá más adelante; aquí usamos `plot()`, `hist()`, `boxplot()` y `pairs()`.

1.1 ¿Por qué analizar datos?

Un análisis de datos sirve para **comprender**, **predecir** y **tomar decisiones** con evidencia.

```
# Ejemplo mínimo: una serie de medidas (p. ej., tiempos de reacción en ms)
x <- c(210, 205, 198, 220, 215, 400)  # ojo: 400 podría ser un valor atípico
mean(x)     # promedio
```

```
[1] 241.3333
```

```
median(x)   # mediana (más resistente a outliers)
```

```
[1] 212.5
```

Interpretación rápida: - La **media** se "tira" hacia 400. - La **mediana** es más estable si hay valores extremos.

1.2 Visión global de un problema de Tratamiento de Datos

En un problema real, típicamente iteramos por etapas:

1. **Definir el problema** (preguntas, contexto).
2. **Recopilar/obtener datos** (fuentes, formato, volumen).
3. **Limpieza y preprocesamiento** (missing, errores, duplicados).
4. **EDA** (estadística descriptiva + visualización + patrones).
5. **Modelado** (si procede) y **comunicación** (tablas/gráficos/informe).

1.2.1 Simulación de un "raw dataset" y su paso a "technically correct"

```
raw <- data.frame(
  id = c("001", "002", "003", "003"),       # duplicado
  edad = c("19", "20", "-5", "21"),         # edad negativa (inconsistente)
  sexo = c("M", "F", "X", "F"),             # categoría inesperada (X)
  peso = c("72.4", "NA", "68.1", "70.2"),   # "NA" como texto
  stringsAsFactors = FALSE
)
raw
```

```
   id edad sexo peso
1 001   19    M 72.4
2 002   20    F   NA
3 003   -5    X 68.1
4 003   21    F 70.2
```

```
str(raw)
```

```
'data.frame':   4 obs. of  4 variables:
 $ id  : chr  "001" "002" "003" "003"
 $ edad: chr  "19" "20" "-5" "21"
 $ sexo: chr  "M" "F" "X" "F"
 $ peso: chr  "72.4" "NA" "68.1" "70.2"
```

Technically correct: tipos correctos (numéricos donde toca), NA reales, etc.

```
tc <- raw
tc$edad <- as.integer(tc$edad)
tc$peso <- as.numeric(tc$peso)  # convierte "NA" a NA real
tc
```

```
   id edad sexo peso
1 001   19    M 72.4
2 002   20    F   NA
3 003   -5    X 68.1
4 003   21    F 70.2
```

```
str(tc)
```

```
'data.frame':   4 obs. of  4 variables:
 $ id  : chr  "001" "002" "003" "003"
 $ edad: int  19 20 -5 21
 $ sexo: chr  "M" "F" "X" "F"
 $ peso: num  72.4 NA 68.1 70.2
```

1.2.2 "Consistent data": reglas básicas de consistencia

Definimos reglas mínimas (ejemplo didáctico): - `edad` debe ser >= 0 - `sexo` debe ser {M,F} - `id` no debe repetirse

```
# Detección de problemas
which(tc$edad < 0)
```

```
[1] 3
```

```
setdiff(unique(tc$sexo), c("M", "F"))
```

```
[1] "X"
```

```r
duplicated(tc$id)
```

```
[1] FALSE FALSE FALSE  TRUE
```

```r
# Filtrado "rápido" (ejemplo): eliminamos registros inconsistentes o duplicados
consistent <- tc[tc$edad >= 0 & tc$sexo %in% c("M","F") & !duplicated(tc$id), ]
consistent
```

```
  id edad sexo peso
1 001   19    M 72.4
2 002   20    F   NA
```

2. Primeros pasos cuando analizas un conjunto de datos

El Tema 1 propone comprobar rápidamente:

1. Nº de registros (filas), 2. Nº de variables (columnas), 3. Tipos,
2. valores perdidos, 5. variables esperadas, 6. valores consistentes,
3. relaciones razonables.

Usaremos `airquality` (incluido con R) porque tiene **valores perdidos**.

```r
data(airquality)  # dataset base
dim(airquality)   # (filas, columnas)
```

```
[1] 153   6
```

```r
names(airquality)
```

```
[1] "Ozone"   "Solar.R" "Wind"    "Temp"    "Month"   "Day"
```

```r
str(airquality)
```

```
'data.frame':   153 obs. of  6 variables:
 $ Ozone  : int  41 36 12 18 NA 28 23 19 8 NA ...
 $ Solar.R: int  190 118 149 313 NA NA 299 99 19 194 ...
 $ Wind   : num  7.4 8 12.6 11.5 14.3 14.9 8.6 13.8 20.1 8.6 ...
 $ Temp   : int  67 72 74 62 56 66 65 59 61 69 ...
 $ Month  : int  5 5 5 5 5 5 5 5 5 5 ...
 $ Day    : int  1 2 3 4 5 6 7 8 9 10 ...
```

2.1 Valores perdidos (missing values)

```r
colSums(is.na(airquality))   # nº de NA por variable
```

```
  Ozone Solar.R    Wind    Temp   Month     Day
     37       7       0       0       0       0
```

```r
sum(!complete.cases(airquality))  # nº de filas con al menos un NA
```

```
[1] 42
```

```r
# Ejemplo de estrategias "base":
aq_complete <- na.omit(airquality)          # eliminar filas con NA
aq_impute   <- airquality
aq_impute$Ozone[is.na(aq_impute$Ozone)] <- median(aq_impute$Ozone, na.rm = TRUE)  # imputación simple
```

3. Caracterización de los datos

3.1 Estadísticos descriptivos (media, mediana, varianza, desviación típica)

```
summary(airquality)  # resumen rápido
```

```
     Ozone           Solar.R          Wind             Temp           Month            Day       
 Min.   :  1.00   Min.   :  7.0   Min.   : 1.700   Min.   :56.00   Min.   :5.000   Min.   : 1.0  
 1st Qu.: 18.00   1st Qu.:115.8   1st Qu.: 7.400   1st Qu.:72.00   1st Qu.:6.000   1st Qu.: 8.0  
 Median : 31.50   Median :205.0   Median : 9.700   Median :79.00   Median :7.000   Median :16.0  
 Mean   : 42.13   Mean   :185.9   Mean   : 9.958   Mean   :77.88   Mean   :6.993   Mean   :15.8  
 3rd Qu.: 63.25   3rd Qu.:258.8   3rd Qu.:11.500   3rd Qu.:85.00   3rd Qu.:8.000   3rd Qu.:23.0  
 Max.   :168.00   Max.   :334.0   Max.   :20.700   Max.   :97.00   Max.   :9.000   Max.   :31.0  
 NA's   :37       NA's   :7                                                                      
```

```
mean(airquality$Temp, na.rm = TRUE)
```

```
[1] 77.88235
```

```
var(airquality$Temp, na.rm = TRUE)
```

```
[1] 89.59133
```

```
sd(airquality$Temp, na.rm = TRUE)
```

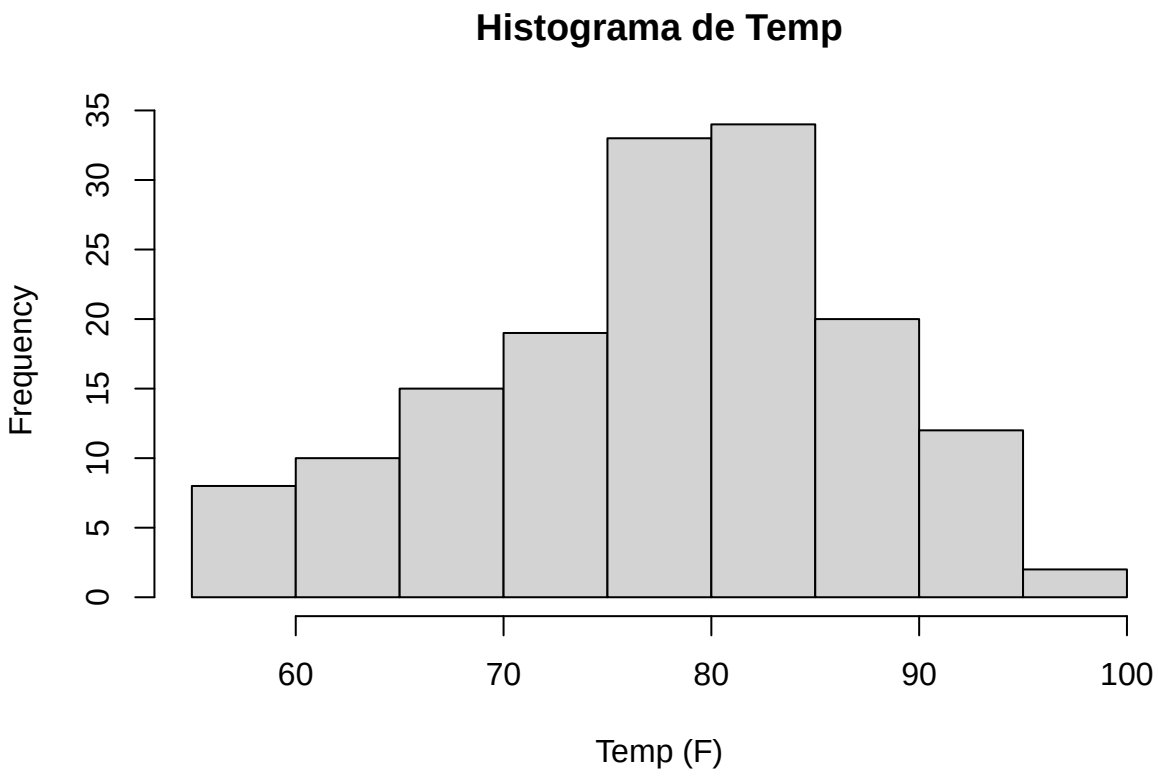
```
[1] 9.46527
```

```
# "5-number summary" (min, Q1, mediana, Q3, max)
fivenum(airquality$Temp)
```

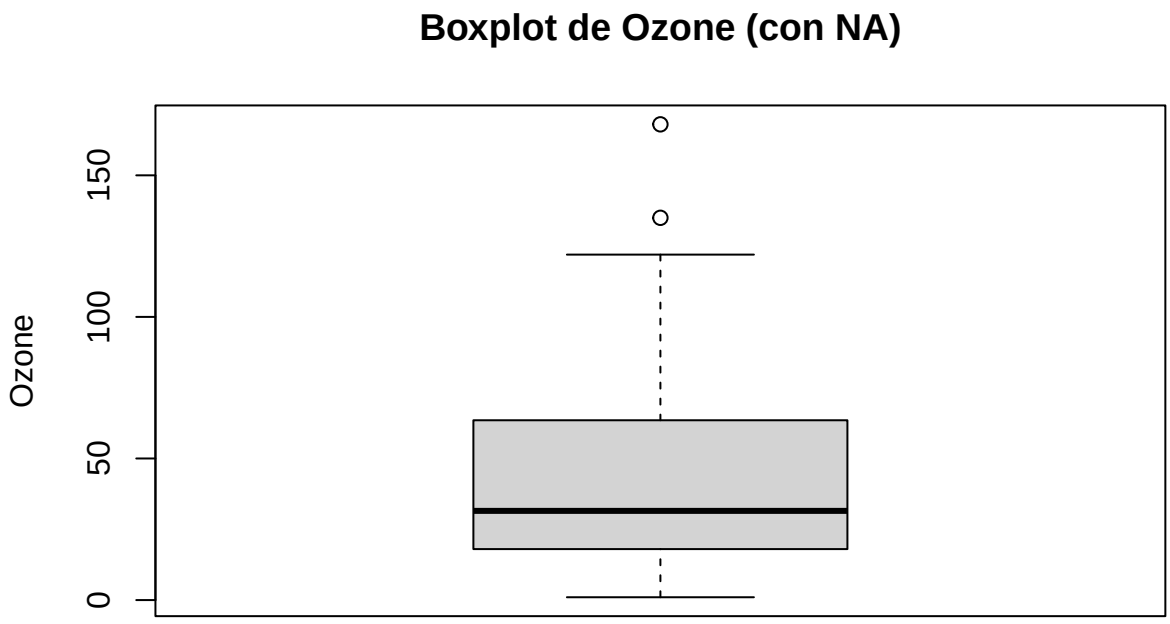
```
[1] 56 72 79 85 97
```

3.2 Análisis visual (R base)

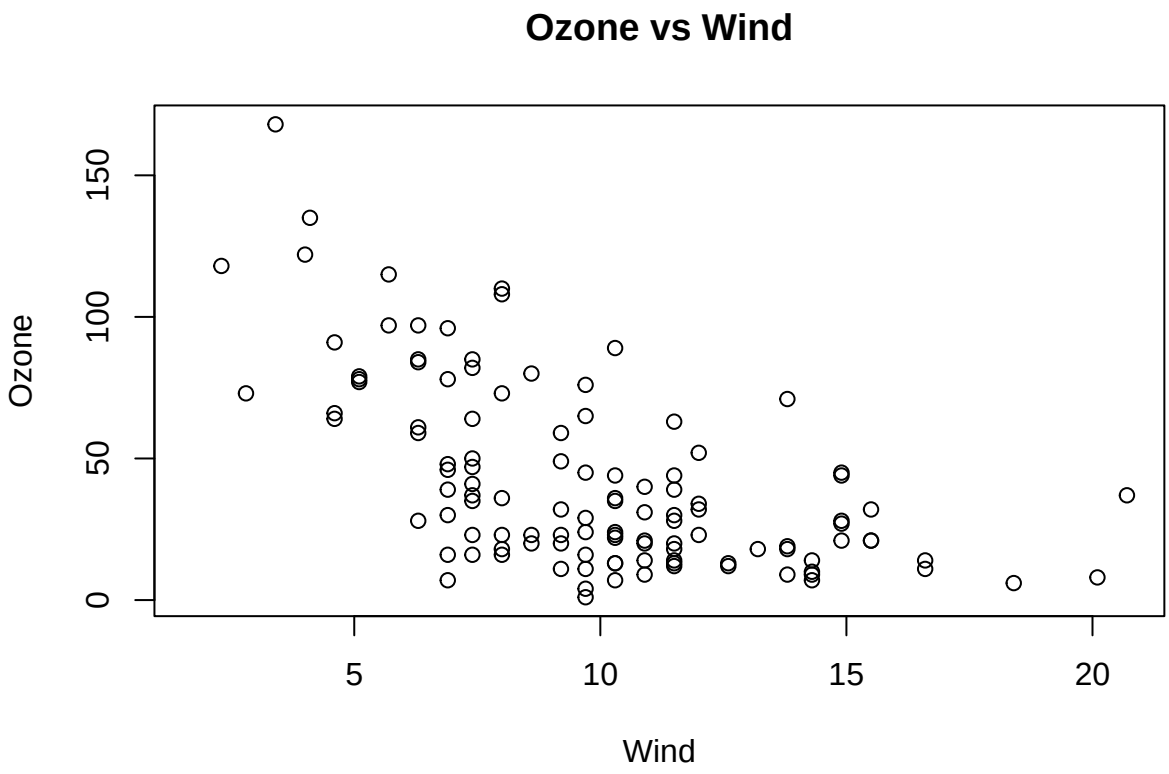
```
hist(airquality$Temp, main="Histograma de Temp", xlab="Temp (F)")
```




```r
boxplot(airquality$Ozone, main="Boxplot de Ozone (con NA)", ylab="Ozone")
```




```r
# Relaciones bivariantes
plot(airquality$Wind, airquality$Ozone,
     xlab="Wind", ylab="Ozone", main="Ozone vs Wind")
```



3.3 Datos inusuales / anómalos / perdidos

Una detección simple de outliers usa el IQR (rango intercuartílico).

```
oz <- airquality$Ozone
q <- quantile(oz, probs=c(0.25, 0.75), na.rm=TRUE)
iqr <- q[2] - q[1]
lim_inf <- q[1] - 1.5*iqr
lim_sup <- q[2] + 1.5*iqr

which(oz < lim_inf | oz > lim_sup)  # índices candidatos a outlier (con NA no pasa nada)
```

```
[1]  62 117
```

```
c(lim_inf=lim_inf, lim_sup=lim_sup)
```

```
lim_inf.25% lim_sup.75%
    -49.875     131.125
```

4. Codebook (metadata): idea y ejemplo mínimo

Un **codebook** describe el dataset y cada variable (definición, unidades, rango, códigos de missing...).

Ejemplo didáctico (mini-codebook en R):

```
codebook <- data.frame(
  variable = c("Ozone", "Solar.R", "Wind", "Temp", "Month", "Day"),
  tipo     = c("num", "num", "num", "num", "int", "int"),
```

```
  unidad   = c("ppb", "Ly", "mph", "F", "mes", "día"),
  rango    = c("0..~200", "0..~350", "0..~25", "50..100", "5..9", "1..31"),
  missing  = c("NA", "NA", "NA", "NA", "NA", "NA"),
  stringsAsFactors = FALSE
)
kable(codebook)
```

variable	tipo	unidad	rango	missing
Ozone	num	ppb	0..~200	NA
Solar.R	num	Ly	0..~350	NA
Wind	num	mph	0..~25	NA
Temp	num	F	50..100	NA
Month	int	mes	5..9	NA
Day	int	día	1..31	NA

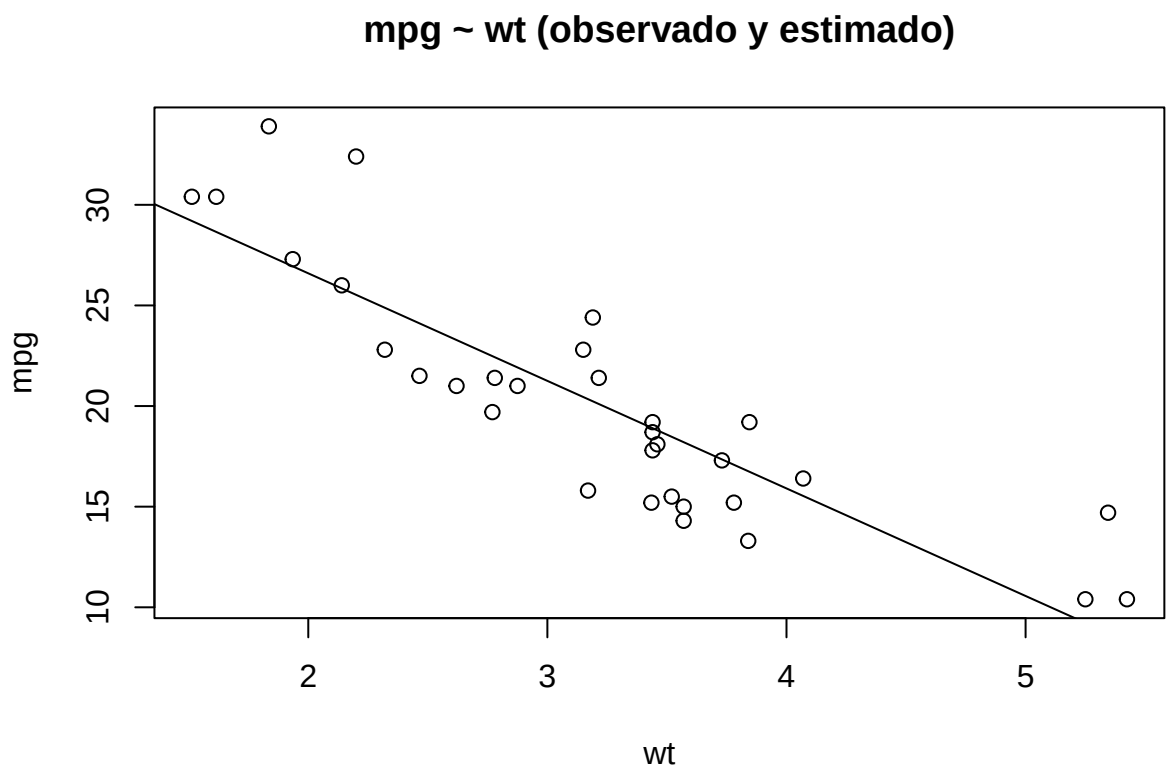
5. Las 4R del Análisis Exploratorio (Tukey) con ejemplos en R base

5.1 Residuos: lo que el modelo no explica

Ajustamos un modelo lineal simple y analizamos residuos.

```
data(mtcars)  # dataset base
fit <- lm(mpg ~ wt, data = mtcars)

# Señal observada vs estimada
plot(mtcars$wt, mtcars$mpg, xlab="wt", ylab="mpg", main="mpg ~ wt (observado y estimado)")
abline(fit)
```

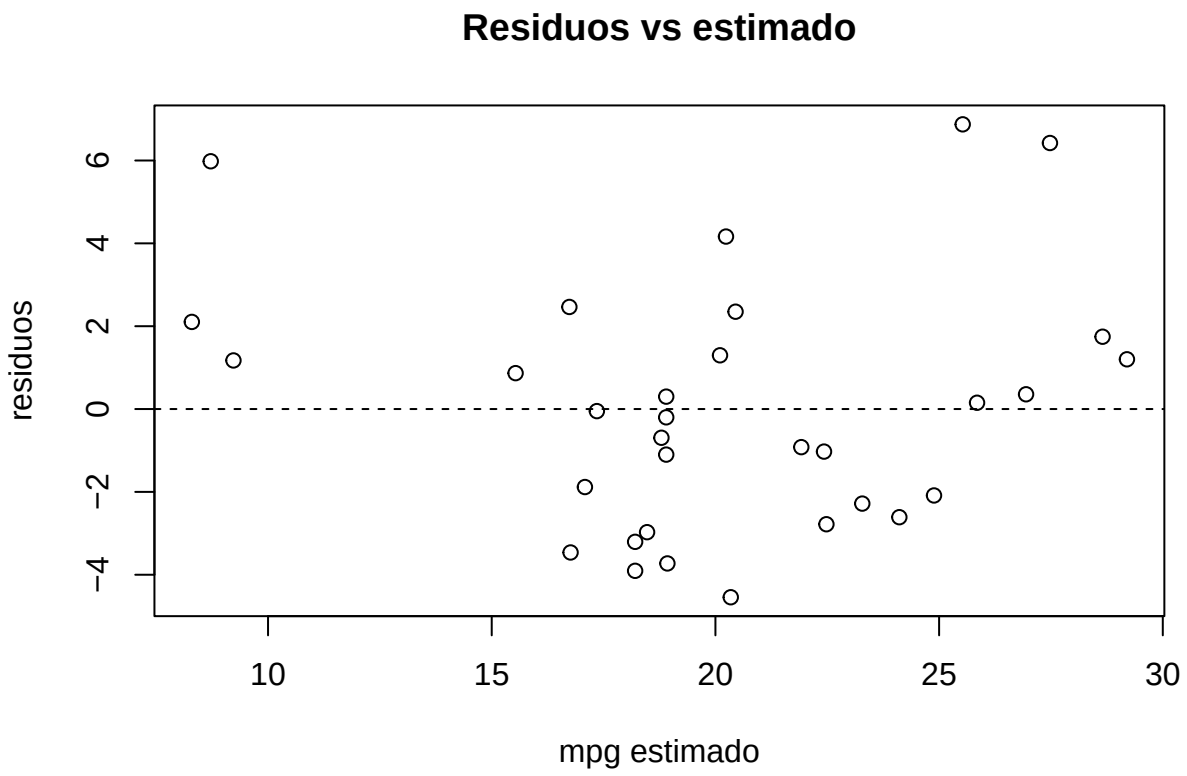



```
# Residuos
res <- residuals(fit)
summary(res)
```

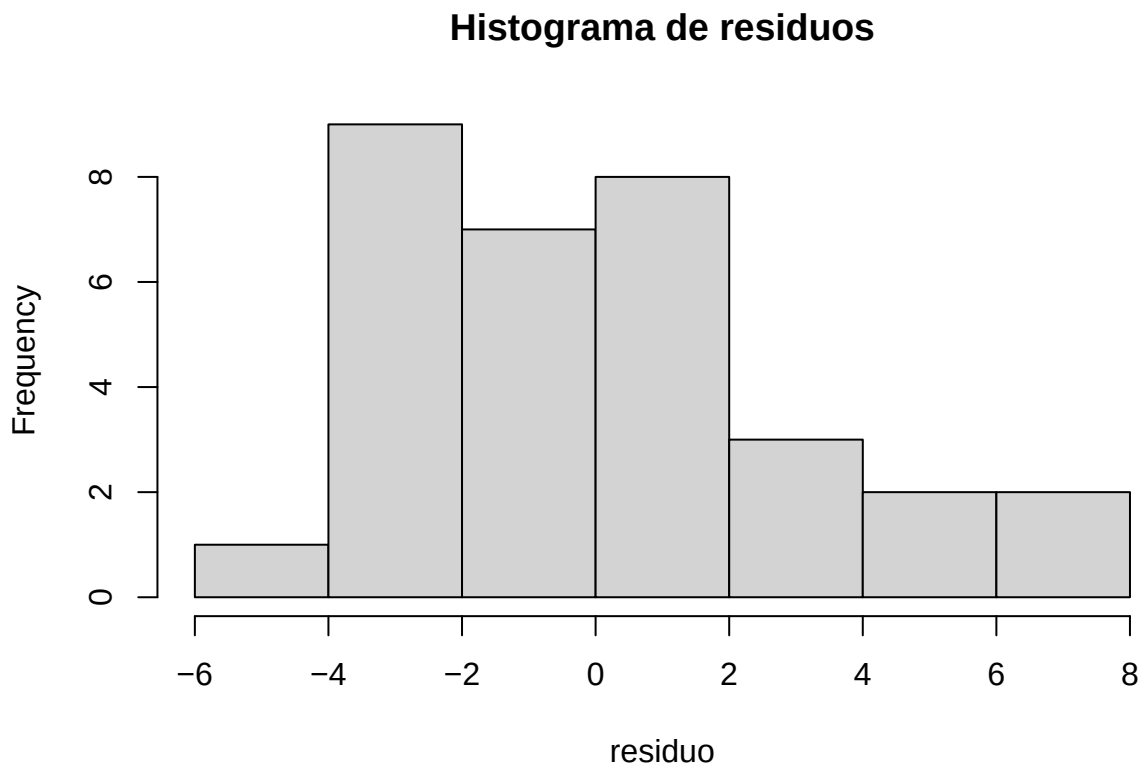
```
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
-4.5432 -2.3647 -0.1252  0.0000  1.4096  6.8727
```

Visualizamos residuos (deberían oscilar alrededor de 0 sin patrón claro si el modelo es razonable).

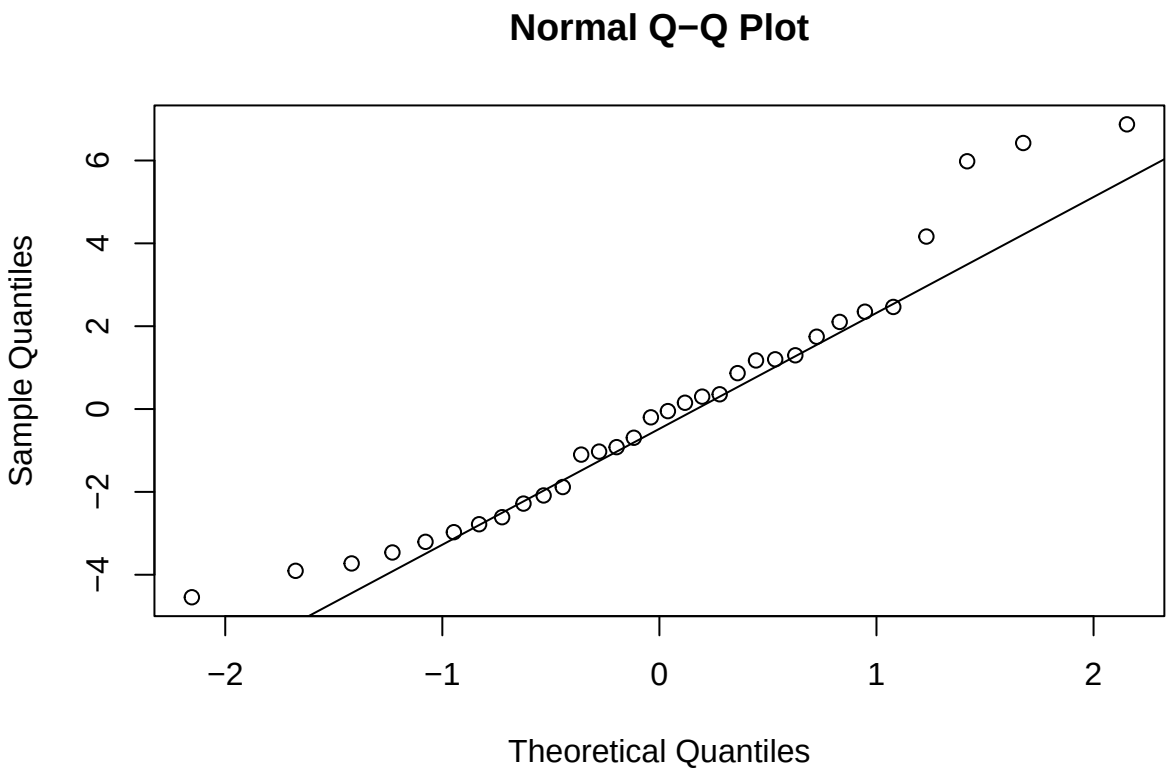
```
plot(fitted(fit), res, xlab="mpg estimado", ylab="residuos", main="Residuos vs estimado")
abline(h=0, lty=2)
```




```
hist(res, main="Histograma de residuos", xlab="residuo")
```




```
qqnorm(res); qqline(res)
```

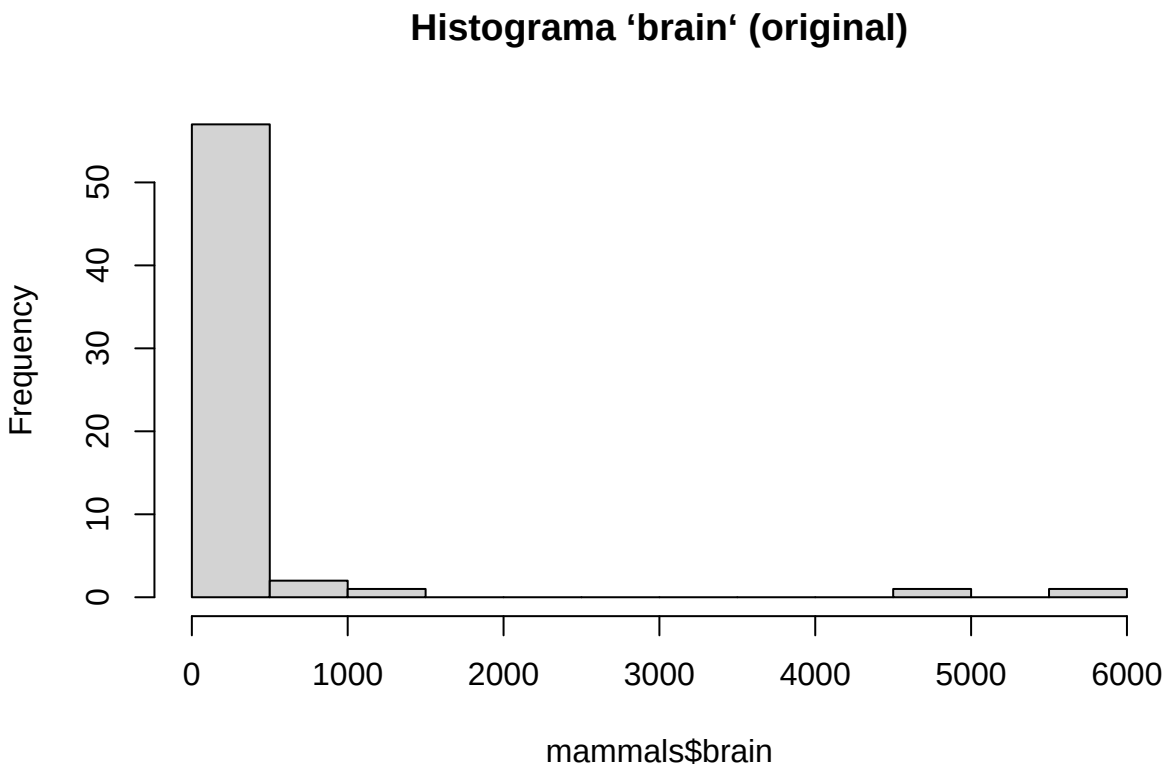


5.2 Reexpresión: transformaciones para revelar estructura

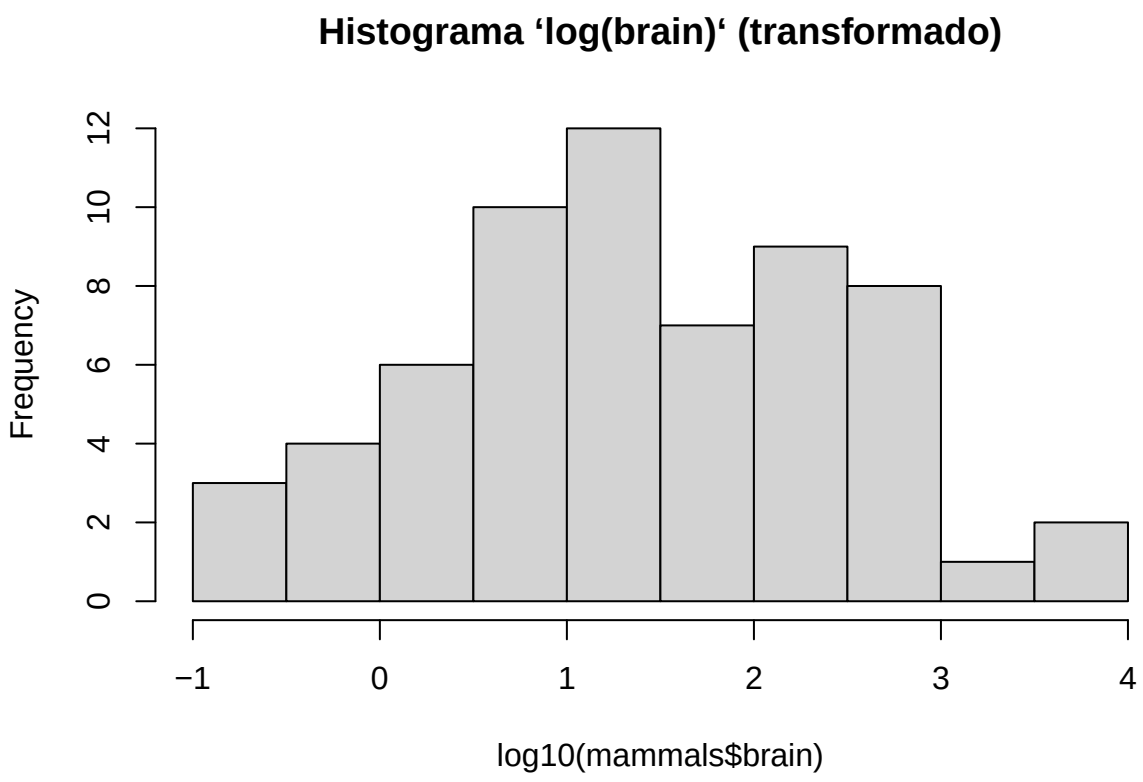
Cuando una variable es muy sesgada, una transformación (p. ej. `log`) puede ayudar.

```
# Ejemplo: relación no lineal simple
y <- mtcars$mpg
x <- mtcars$wt
library(MASS)

# Transformación log (solo como ejemplo: no siempre procede)
hist(mammals$brain,breaks = 10,main="Histograma `brain` (original)")
```



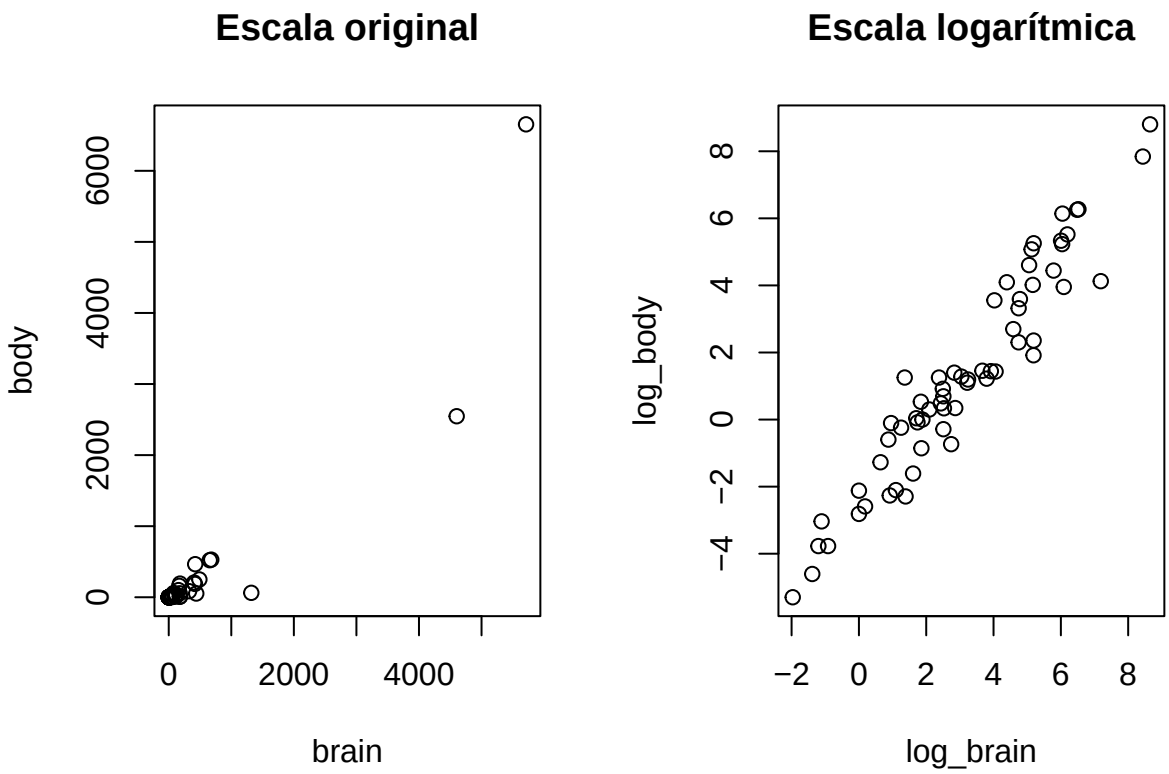

```r
hist(log10(mammals$brain),breaks = 10,main="Histograma `log(brain)` (transformado)")
```




```r
library(MASS)
data(mammals)

# Transformación logarítmica
mammals$log_body <- log(mammals$body)
mammals$log_brain <- log(mammals$brain)

# Visualización
par(mfrow=c(1,2))
plot(body ~ brain, data=mammals, main="Escala original")
plot(log_body ~ log_brain, data=mammals, main="Escala logarítmica")
```



5.3 Resistencia: medidas poco sensibles a outliers

Comparación de media vs mediana; ejemplo con un outlier artificial.

```
z <- c(10, 11, 9, 10, 10, 200)   # 200 es outlier
mean(z)
```

```
[1] 41.66667
```

```
median(z)
```

```
[1] 10
```

```
# Media recortada (trim) en R base
mean(z, trim = 0.2)
```

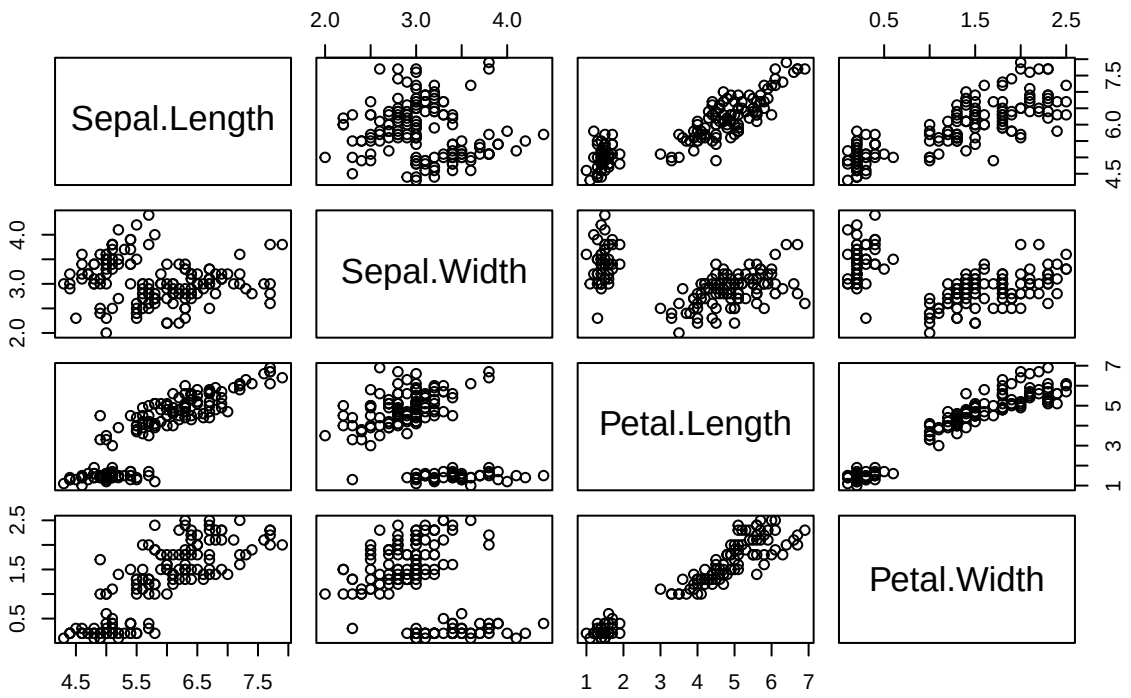
```
[1] 10.25
```

5.4 Revelación: visualizar para "ver" patrones

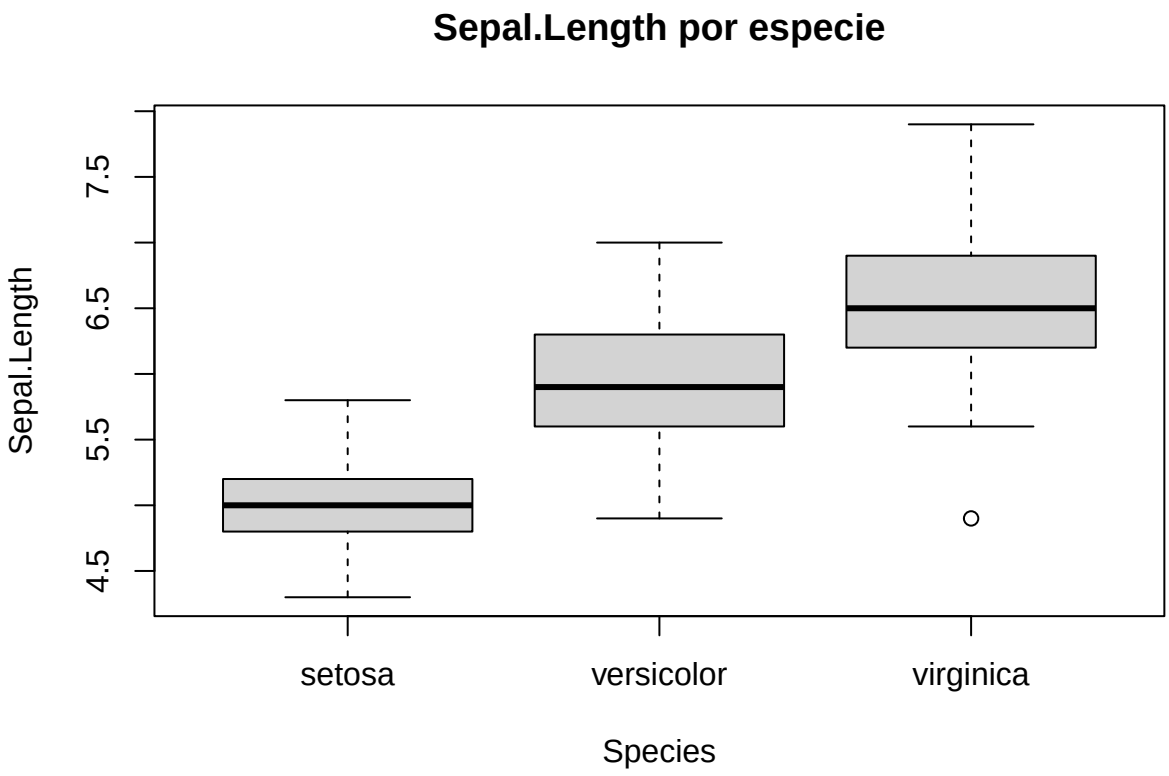
Ejemplo: matriz de dispersión `pairs()` y boxplots por grupos (iris está en R base).

```
data(iris)

pairs(iris[1:4], main="Iris: matriz de dispersión (R base)")
```




```r
boxplot(Sepal.Length ~ Species, data=iris, main="Sepal.Length por especie", ylab="Sepal.Length")
```



6. Fuentes de datos y cómo importarlas (sin paquetes)

Ejemplo de importación "clásica" con `read.csv()`: - Desde fichero local: `read.csv("mis_datos.csv")` - Desde URL (si la red lo permite): `read.csv("https://.../archivo.csv")`

Aquí creamos un CSV temporal para simular el flujo completo:

```
tmp <- tempfile(fileext = ".csv")
write.csv(head(airquality, 10), tmp, row.names = FALSE)

d <- read.csv(tmp)
head(d)
```

```
  Ozone Solar.R Wind Temp Month Day
1    41     190  7.4   67     5   1
2    36     118  8.0   72     5   2
3    12     149 12.6   74     5   3
4    18     313 11.5   62     5   4
5    NA      NA 14.3   56     5   5
6    28      NA 14.9   66     5   6
```

```
str(d)
```

```
'data.frame':   10 obs. of  6 variables:
 $ Ozone  : int  41 36 12 18 NA 28 23 19 8 NA
 $ Solar.R: int  190 118 149 313 NA NA 299 99 19 194
 $ Wind   : num  7.4 8 12.6 11.5 14.3 14.9 8.6 13.8 20.1 8.6
```

```
$ Temp   : int  67 72 74 62 56 66 65 59 61 69
$ Month  : int  5 5 5 5 5 5 5 5 5 5
$ Day    : int  1 2 3 4 5 6 7 8 9 10
```

Bibliografía (del Tema 1)

- Pearson, R. K. *Exploratory Data Analysis Using R.* Chapman & Hall/CRC (2018).