CAPÍTULO 8

ANÁLISIS DE DATOS

1.- LECTURA DE DATOS

Suele ser bastante común que los datos de partida para realizar un análisis estadístico estén recogidos en un archivo de texto. Supongamos para empezar que nuestros datos están en el fichero denominado "Tension de rotura.txt" (para descargar hacer clic aquí; por otra parte, evitaremos en lo posible los acentos en la denominación de archivos). Supongamos así mismo que este archivo se halla en la unidad C. Para leer los datos usaremos la función **`read.table()`** debiendo detallar con precisión la ruta de acceso:

```
> datos<-read.table("C:\\Tensión de rotura.txt")
> datos
     V1
1  4.05
2  4.58
3  4.42
...
49 3.54
50 4.84
> class(datos) # Con esta sentencia confirmamos el tipo de objeto que es datos
[1] "data.frame"
```

Como se ve, el programa ha creado un marco de datos denominado **`datos`** y que consta de 50 valores de una variable a la que automáticamente ha asignado el nombre V1. Ahora crearemos un vector con los valores numéricos correspondientes, lo que se consigue eligiendo (mediante el símbolo **$**) la variable V1 del *data frame* **`datos`**:

```
> tenrot<-datos$V1
> tenrot
[1] 4.05 4.58 4.42 4.20 4.41 4.64 4.76 4.58 3.95
4.17 4.56 3.51 3.27 3.80 3.59
[16] 4.70 3.77 3.80 4.27 3.94 3.96 4.86 4.39 4.04
4.36 3.72 4.00 3.46 4.01 4.08
[31] 3.40 3.89 4.46 4.38 4.41 4.33 4.16 4.58 4.03
3.76 4.05 4.17 4.46 3.60 4.76
[46] 3.99 4.43 4.15 3.54 4.84

> #Podemos ordenar los valores del vector tenrot
> sort(tenrot)
[1] 3.27 3.40 3.46 3.51 3.54 3.59 3.60 3.72 3.76
3.77 3.80 3.80 3.89 3.94 3.95
[16] 3.96 3.99 4.00 4.01 4.03 4.04 4.05 4.05 4.08
4.15 4.16 4.17 4.17 4.20 4.27
[31] 4.33 4.36 4.38 4.39 4.41 4.41 4.42 4.43 4.46
4.46 4.56 4.58 4.58 4.58 4.64
[46] 4.70 4.76 4.76 4.84 4.86
```

2.- GRÁFICOS

En primer lugar vamos a construir una tabla de frecuencias de los datos **tenrot**. Al tratarse de una variable continua se debe hacer una agrupación por intervalos, pues si no se obtendría una tabla tan poco práctica como la siguiente:

```
> table(tenrot)
tenrot
3.27  3.4 3.46 3.51 3.54 3.59  3.6 3.72 3.76 3.77
3.8 3.89 3.94 3.95 3.96 3.99
   1    1    1    1    1    1    1    1    1    1
2    1    1    1    1    1
   4 4.01 4.03 4.04 4.05 4.08 4.15 4.16 4.17  4.2
4.27 4.33 4.36 4.38 4.39 4.41
   1    1    1    1    2    1    1    1    2    1
1    1    1    1    1    2
4.42 4.43 4.46 4.56 4.58 4.64  4.7 4.76 4.84 4.86
   1    1    2    1    3    1    1    2    1    1
```

Como la variable tiene 50 datos un número adecuado de intervalos es 7, ya que no excede a la raíz de 50. Construimos los límites de esos intervalos:

```
> límites<-seq(3,5.1,0.3)
> límites
[1] 3.0 3.3 3.6 3.9 4.2 4.5 4.8 5.1
```

Veamos ahora en qué intervalo queda cada uno de los 50 valores leídos. Por defecto *R* crea intervalos abiertos por la izquierda y cerrados por la derecha. Si queremos hacer intervalos cerrados por la izquierda y abiertos por la derecha indicamos la opción **right=F**:

```
> tenrot.intervalos<-cut(tenrot,límites,right=F)
> tenrot.intervalos
[1] [3.9,4.2) [4.5,4.8) [4.2,4.5) [4.2,4.5)
[4.2,4.5)    [4.5,4.8) [4.5,4.8)
[8] [4.5,4.8) [3.9,4.2) [3.9,4.2) [4.5,4.8)
[3.3,3.6) [3,3.3)   [3.6,3.9)
[15] [3.3,3.6) [4.5,4.8) [3.6,3.9) [3.6,3.9)
[4.2,4.5) [3.9,4.2) [3.9,4.2)
[22] [4.8,5.1) [4.2,4.5) [3.9,4.2) [4.2,4.5)
[3.6,3.9) [3.9,4.2) [3.3,3.6)
[29] [3.9,4.2) [3.9,4.2) [3.3,3.6) [3.6,3.9)
[4.2,4.5) [4.2,4.5) [4.2,4.5)
[36] [4.2,4.5) [3.9,4.2) [4.5,4.8) [3.9,4.2)
[3.6,3.9) [3.9,4.2) [3.9,4.2)
[43] [4.2,4.5) [3.6,3.9) [4.5,4.8) [3.9,4.2)
[4.2,4.5) [3.9,4.2) [3.3,3.6)
[50] [4.8,5.1)
7 Levels: [3,3.3) [3.3,3.6) [3.6,3.9) [3.9,4.2)
[4.2,4.5) ... [4.8,5.1)
```

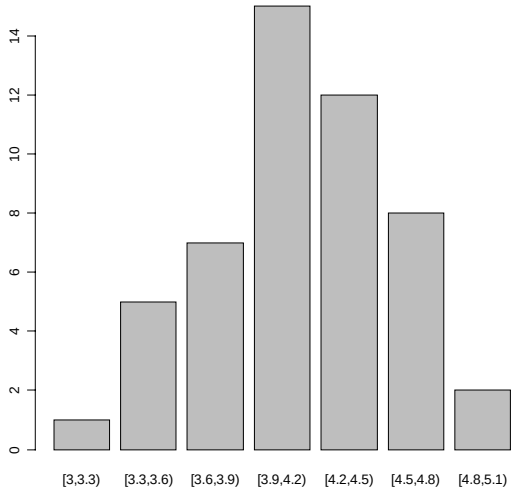
Utilizamos ahora la función **`table`** para contar el número de veces que aparece cada intervalo, lo que es propiamente una tabla de frecuencias:

```
> table(tenrot.intervalos)
tenrotporintervalos
[3,3.3) [3.3,3.6) [3.6,3.9) [3.9,4.2) [4.2,4.5)
[4.5,4.8) [4.8,5.1)
1         5         7        15        12
8   2
```

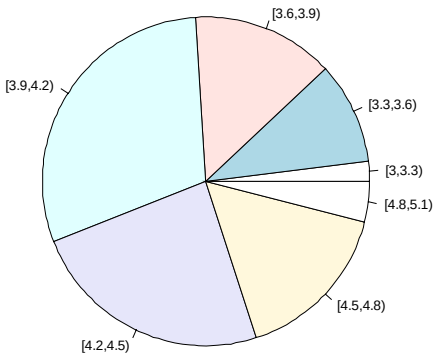
Es decir, en el primer intervalo [3,3.3) hay un valor, en el segundo intervalo [3.3,3.6) hay 5 valores, etc.

Una vez que tenemos agrupada por intervalos la variable podemos hacer un diagrama de barras (aunque este diagrama es más adecuado para variables discretas) y otro de sectores:

```
> barplot(table(tenrot.intervalos))
```




```
> pie(table(tenrot.intervalos))
```

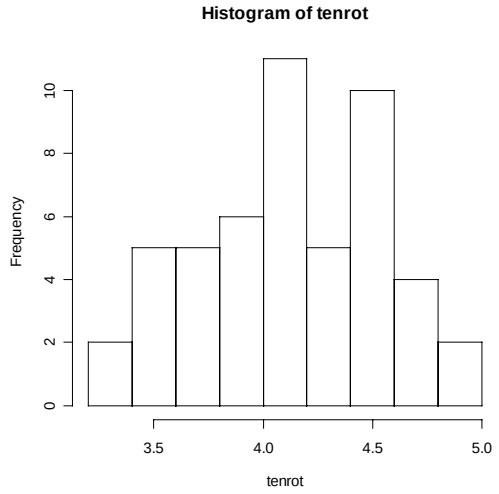


Un diagrama de tallos y hojas (*stem and leaf)* es una representación sencilla de los datos similar a un histograma, pero con la ventaja de que se conserva la información numérica de todos y cada uno de los datos.

```
> stem(tenrot)
The decimal point is 1 digit(s) to the left of the
|
          32 | 7
          34 | 06149
          36 | 0267
          38 | 0094569
          40 | 01345585677
          42 | 073689
          44 | 1123666888
          46 | 4066
          48 | 46
```

Al igual que el diagrama de tallos y hojas, un histograma (similar al diagrama de barras obtenido anteriormente pero con las barras pegadas, lo que da idea de variable continua) muestra la forma en que se distribuyen los datos. La instrucción correspondiente para que *R* genere un histograma como “considere oportuno” es

```
> hist(tenrot)
```



3.- MEDIDAS DE CENTRALIZACIÓN Y DE DISPERSIÓN

Obtenemos a continuación la media y la mediana de los valores que forman el vector **tenrot**:

```
> mean(tenrot);median(tenrot)
[1] 4.1448
[1] 4.155
```

Para obtener la media recortada al 10 %, o sea la media aritmética de todos los valores exceptuando el 10% de los que están por arriba y el 10% de los que están por abajo, hacemos:

```
> mean(tenrot,trim=0.1)
[1] 4.1535
```

La instrucción **var** nos devuelve un estimador insesgado de la varianza muestral, que se denomina *cuasivarianza*. Para obtener el valor de la varianza de los datos debemos multiplicar el valor anterior por **(length(tenrot)-1)/length(tenrot))** donde **length(tenrot)** es la longitud del vector, o sea el número de datos (en nuestro ejemplo 50):

```
> varianza.muestra<-((50-1)/50)*var(tenrot)
> varianza.muestra
[1] 0.1583210
> sqrt(varianza.muestra)
[1] 0.3978957
```

El valor anterior es la desviación típica de los datos. Este valor también puede obtenerse a través de la función **sd** que da la *cuasidesviación típica.*

```
> desvtípica.muestra<-sqrt(varianza.muestra)
> desvtípica.muestra
[1] 0.3978957
> sqrt((50-1)/50)*sd(tenrot)
[1] 0.3978957
```

Los percentiles son los valores que dividen el rango de los datos en cien unidades, de modo que, por ejemplo, el percentil 20 es el valor que deja por debajo de sí el 20% de las observaciones; el percentil 50 deja por debajo la mitad de las observaciones, o sea es la mediana; etc. La función **quantile** sirve para calcular percentiles.

```
>quantile(tenrot,0.3)#Con esta sentencia calculamos el percentil 30
    30%
3.957
>quantile(tenrot,0.5)#Volvemos a calcular la mediana de otro modo
    50%
4.155
```

Mediante la función anterior se puede calcular también un resumen de cinco números: mínimo, primer cuartil (Q1), mediana o segundo cuartil (Q2), tercer cuartil (Q3), y máximo.

```
> quantile(tenrot)
  0%    25%    50%    75%   100%
3.2700 3.9025 4.1550 4.4275 4.8600
```

Si además de los cinco valores anteriores queremos conocer la media podemos hacer:

```
> summary(tenrot)
          Min. 1st Qu.   Median      Mean 3rd Qu.
Max.
  3.270   3.903   4.155   4.145   4.428   4.860
```

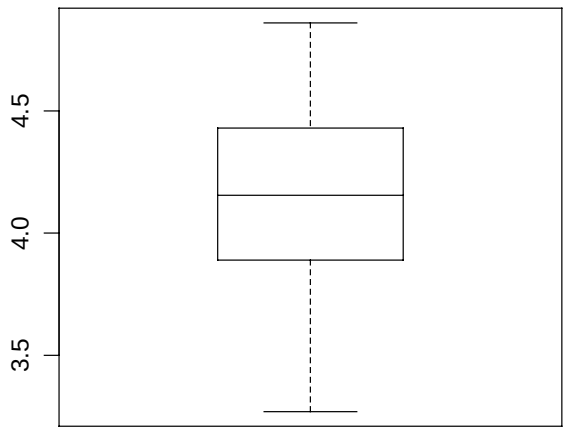
Conocidos los valores anteriores podemos calcular el recorrido intercuartílico RIQ = Q3-Q1.

```
> RIQ<-quantile(tenrot,0.75)-quantile(tenrot,0.25)
> RIQ
  75%
0.525
```

4.- DETECCIÓN DE DATOS ATÍPICOS

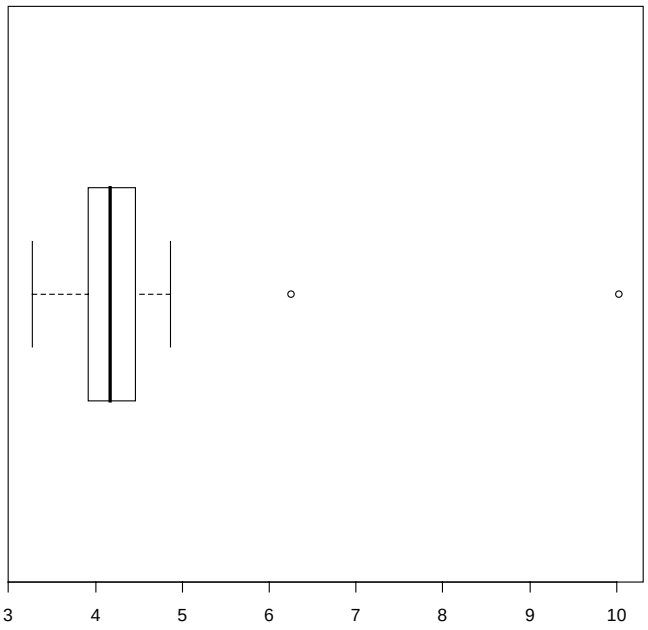
Una forma adecuada para detectar datos atípicos cuando se tiene una variable unidimensional es utilizar un diagrama de cajas o *boxplot*. Para ello utilizamos la función **`boxplot`**, que en su forma más simple (tiene varias versiones) es:

```
> boxplot(tenrot)
```



Los extremos del diagrama son el valor máximo y el mínimo. Los valores atípicos o *outliers* (valores mayores que 3/2*RIQ por encima de Q3 o menores que 3/2*RIQ por debajo de Q1), si los hay, vienen señalados por un pequeño círculo y, entonces, los extremos que aparecen son el máximo y el mínimo de los valores que quedan al eliminar esos valores. Como se ve en la figura, en este caso no hay datos atípicos. Sin embargo, si al conjunto de datos **`tenrot`** le añadimos los valores 6.25 y 10.03 el gráfico boxplot cambiaría:

```
> tenrot.nuevo<-c(tenrot,6.25,10.03)
> boxplot(tenrot.nuevo,horizontal=T)
> #Con el argumento añadido se obtiene un diagrama
de cajas en horizontal
```



Los datos atípicos pueden ser detectados así mismo mediante la función **boxplot.stats()**

```
> boxplot.stats(tenrot.nuevo)
$stats
[1] 3.270 3.915 4.165 4.460 4.860
$n
[1] 52
$conf
[1] 4.045587 4.284413
$out
[1]  6.25 10.03
```

La salida anterior se interpreta del siguiente modo: `$stats` da las 5 medidas que definen el diagrama de cajas; `$n` es el número de valores que contiene el vector de datos; `$conf` son los valores inferior y superior de las "muescas" o *notch* (en caso de hacer un boxplot con muescas éstas indican los extremos del intervalo de confianza para la mediana). `$out` expresa los dos valores atípicos.

5.- DATOS BIDIMENSIONALES

Hasta ahora hemos estudiado cómo describir con *R* el valor que toma una variable en los individuos de una muestra o población. Sin embargo, en ocasiones interesa estudiar qué valores toman dos variables cuantitativas de forma conjunta, con objeto de estudiar si existe alguna relación entre

ellas. Por ejemplo, podemos querer analizar a la vez el peso y la talla de los individuos de una población, para tratar de averiguar una posible relación entre ambas variables.

En las próxima líneas vamos a utilizar el conjunto de datos **rmr,** que está incluido en el paquete **ISwR** y que lo podemos cargar así:

Packages → Load package... → ISwR

o bien con la instrucción

```
> library(ISwR)
```

Si queremos obtener información sobre el paquete hacemos

```
> library(help=ISwR)
```

Una vez cargado el paquete, leemos el *data frame* **rmr**. Este marco de datos contiene 44 pares de datos referidos a las variables peso (body.weight) y tasa de metabolismo basal (metabolic.rate) de 44 mujeres elegidas al azar.

```
> data(rmr)
> rmr
body.weight metabolic.rate
1          49.9           1079
2          50.8           1146
3          51.8           1115
...
43        125.2           1630
44        143.3           1708
```

Con objeto de que a partir de ahora las dos variables sean accesibles debemos hacer

```
> attach(rmr)
```

Ahora sí estamos en condiciones de poder utilizar las variables:

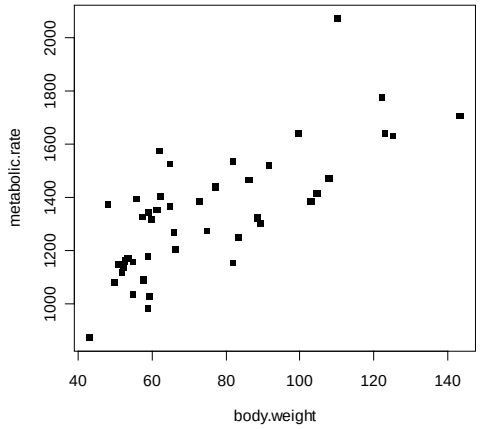
```
> body.weight
 [1]   49.9   50.8   51.8   52.6   57.6   61.4   62.3
64.9  43.1  48.1  52.2  53.5
[13]   55.0   55.0   56.0   57.8   59.0   59.0   59.2
59.5  60.0  62.1  64.9  66.0
[25]   66.4   72.8   74.8   77.1   82.0   82.0   83.4
86.2  88.6  89.3  91.6  99.8
[37]  103.0  104.5  107.7  110.2  122.0  123.1  125.2
143.3


> metabolic.rate
```

```
 [1] 1079 1146 1115 1161 1325 1351 1402 1365  870
1372 1132 1172 1034 1155 1392
[16] 1090  982 1178 1342 1027 1316 1574 1526 1268
1205 1382 1273 1439 1536 1151
[31] 1248 1466 1323 1300 1519 1639 1382 1414 1473
2074 1777 1640 1630 1708
```

Lo primero que se debe hacer al tratar de estudiar la posible relación entre dos variables estadísticas es visualizar los datos mediante un *diagrama de dispersión* o *scatterplot,* en el que cada par de valores de las variables serán las coordenadas de una serie de puntos del plano (nube de puntos):

```
> plot(body.weight,metabolic.rate,pch=15)
> #La opción pch=15 es una de las posibles
alternativas existentes para marcar un punto
```



En el gráfico anterior se observa un posible valor atípico en la parte superior. Teniendo en cuenta que se puede interactuar con el gráfico que está activo podemos detectar ese punto, para lo cual escribimos la instrucción **identify**, pulsamos Enter y después accionamos el botón izquierdo del ratón al colocar éste sobre el punto en el gráfico. Así se obtiene el índice (número de orden) que corresponde al punto buscado.

```
> identify(body.weight,metabolic.rate,n=1)
[1] 40
> rmr[40,]
   body.weight metabolic.rate
40       110.2           2074
```

Una primera visión del gráfico sugiere una cierta relación entre el peso X y la tasa metabólica Y en el sentido de que para valores grandes de X se encuentran más bien valores grandes de Y, y para valores pequeños de X valores también pequeños de Y.

La figura anterior nos da idea de una cierta “asociación lineal” entre las dos variables. Para cuantificar tal asociación se utiliza el *coeficiente de correlación lineal* (valor comprendido entre –1 y 1):

$$r(x,y) = \frac{s_{xy}}{s_x s_y} = \frac{\dfrac{\sum_{i=1}^{n}(x_i-\overline{x})(y_i-\overline{y})}{n}}{\sqrt{\dfrac{\sum_{i=1}^{n}(x_i-\overline{x})^2}{n}}\sqrt{\dfrac{\sum_{i=1}^{n}(y_i-\overline{y})^2}{n}}}$$

siendo s_{xy} la *covarianza* entre x e y, y s_x y s_y las desviaciones típicas respectivas.

```
> cor(body.weight,metabolic.rate)
[1] 0.7442379
> cor(metabolic.rate,body.weight)
[1] 0.7442379
```

Vemos que el coeficiente de correlación entre X e Y es el mismo que entre Y y X.

Un coeficiente de correlación relativamente alto entre dos variables (próximo a 1 o a –1), como en este caso, indica una cierta asociación lineal entre las variables, aunque no necesariamente una relación real entre ellas.

Una vez comprobado que existe una asociación lineal entre dos variables trataremos de calcular la recta que “mejor se ajusta” a la nube de puntos mediante el criterio de ajuste denominado de *mínimos cuadrados*. Obtenemos así la *recta de regresión (RR)*.

Cuando se tienen datos de dos variables X e Y es posible calcular dos rectas de regresión: la de Y sobre X, que sirve para predecir Y conociendo X, y la de X sobre Y que nos da el mejor pronóstico de X conociendo la Y.

Calculemos en primer lugar la RR de la tasa de metabolismo basal Y respecto del peso X:

```
> lm(metabolic.rate~body.weight)
Call:
lm(formula = metabolic.rate ~ body.weight)
Coefficients:
(Intercept)  body.weight
     811.23         7.06
```

Este resultado significa que la RR de Y sobre X es

$$Y = 811.23 + 7.06X$$

Así, por ejemplo, para un peso de 70 kg la tasa metabólica estimada es

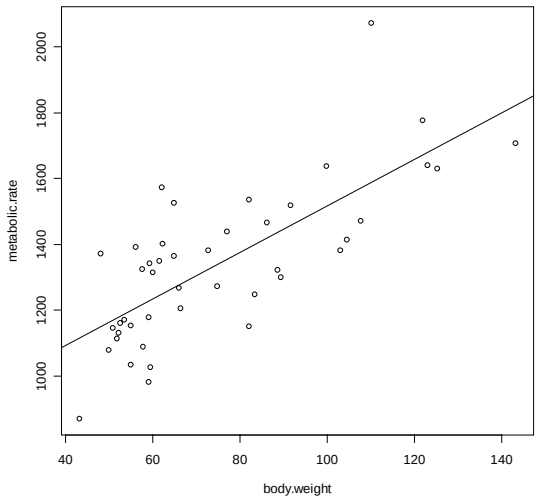
```
>pron.de.la.tasa.metabólica.para.70kg<-
811.23+7.06*70
> pron.de.la.tasa.metabólica.para.70kg
[1] 1305.43
```

La RR de X sobre Y se obtiene así:

```
> lm(body.weight~metabolic.rate)
Call:
lm(formula = body.weight ~ metabolic.rate)
Coefficients:
   (Intercept)  metabolic.rate
     -30.24427         0.07846
```

Dibujemos la RR de Y sobre X superpuesta a la nube de puntos:

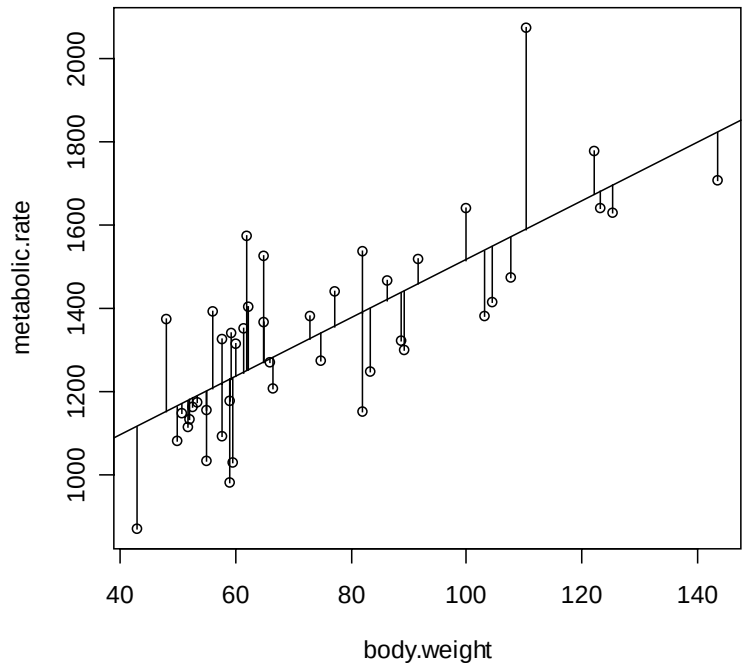
```
> plot(metabolic.rate~body.weight)
> abline(lm(metabolic.rate~body.weight))
```



Una manera efectiva de detectar las posibles deficiencias de un modelo de regresión lineal consiste en llevar a cabo un análisis de los *residuos*, o sea de las diferencias entre las ordenadas de los puntos de la muestra y las correspondientes a la recta de regresión.

Para generar un gráfico que muestre los residuos mediante segmentos que unen las observaciones con los correspondientes puntos de la recta ajustada hacemos lo que sigue a continuación. Previamente es conveniente almacenar el modelo lineal utilizando, por ejemplo, el nombre **`lm.tasa`**:

```
> lm.tasa<-lm(metabolic.rate~body.weight)
> segments(body.weight,fitted(lm.tasa),
+ body.weight,metabolic.rate)
```



Mediante la función **segments** se consigue dibujar segmentos que unen los puntos de coordenadas (x_1,y_1,x_2,y_2).

Si deseamos calcular los valores de esos 44 residuos hacemos

```
> resid(lm.tasa)
         1          2          3          4
5          6          7
 -84.49711    -23.85069    -61.91022    -21.55784
107.14452  106.31832  150.96474
         8          9         10         11
12         13         14
  95.60997  -245.49232   221.21004   -47.73403   -
16.91141 -165.50070  -44.50070
        15         16         17         18
19         20         21
 185.43977   -129.26738   -245.73882    -49.73882
112.84928 -204.26858   81.20166
        22         23         24         25
26         27         28
 324.37665    256.60997     -9.15551    -74.97932
56.83970  -66.27936   83.48373
        29         30         31         32
33         34         35
 145.89204  -239.10796  -151.99130    46.24203  -
113.70084 -141.64251   61.12058
        36         37         38         39
40         41         42
 123.23245   -156.35804   -134.94733    -98.53782
484.81336  104.51093  -40.25455
```

```
        43          44
 -65.07956 -114.85701
> residuos<-resid(lm.tasa)
```

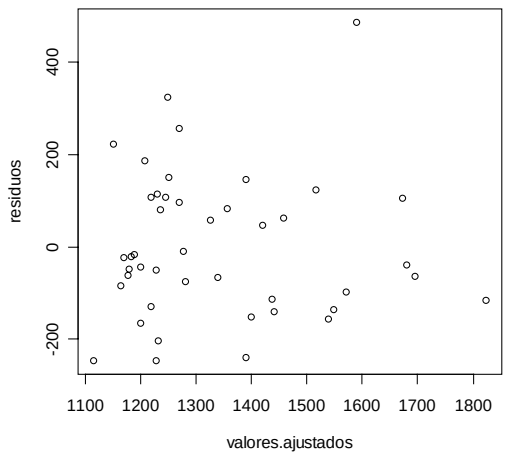
Los valores ajustados, es decir, los que toma la recta de regresión para cada valor de la variable observada **body.weight**, son

```
> fitted(lm.tasa)
       1          2          3          4          5
6        7        8
1163.497  1169.851  1176.910  1182.558  1217.855
1244.682 1251.035 1269.390
       9         10         11         12         13
14       15       16
1115.492  1150.790  1179.734  1188.911  1199.501
1199.501 1206.560 1219.267
      17         18         19         20         21
22       23       24
1227.739  1227.739  1229.151  1231.269  1234.798
1249.623 1269.390 1277.156
      25         26         27         28         29
30       31       32
1279.979  1325.160  1339.279  1355.516  1390.108
1390.108 1399.991 1419.758
      33         34         35         36         37
38       39       40
1436.701  1441.643  1457.879  1515.768  1538.358
1548.947 1571.538 1589.187
      41       42       43       44
1672.489 1680.255 1695.080 1822.857

> valores.ajustados<-fitted(lm.tasa)
```

Graficando los residuos frente a los valores ajustados podemos detectar deficiencias en el modelo:

```
> plot(valores.ajustados,residuos)
```



Sin entrar en detalles, el gráfico anterior sugiere en principio una no adecuación del modelo, pues parece que según aumentan los valores ajustados los residuos tienden a disminuir. Da la impresión en definitiva de que se viola la hipótesis de homocedasticidad o varianza constante del error (residuo). De todos modos, existen técnicas estadísticas para adecuar los datos muestrales a un modelo lineal en una situación como la presente.