Introducción a los métodos cuantitativos en arqueología con R. Primera parte:métodos descriptivos e inferenciales uni y bivariados

Book · January 2018

CITATIONS
0

READS
9,014

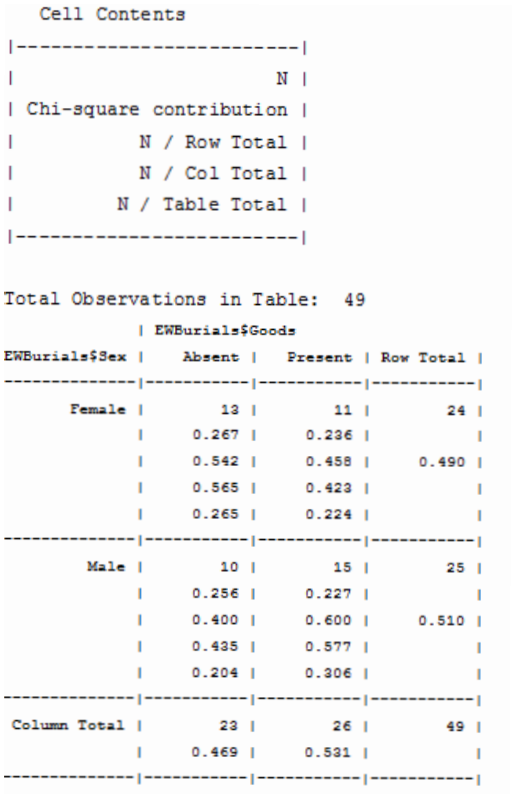
1 author:

Cardillo Marcelo
National Scientific and Technical Research Council
134 PUBLICATIONS **825** CITATIONS

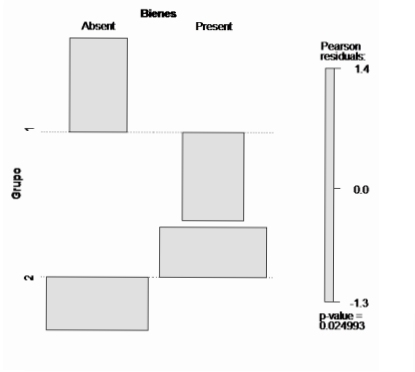
SEE PROFILE

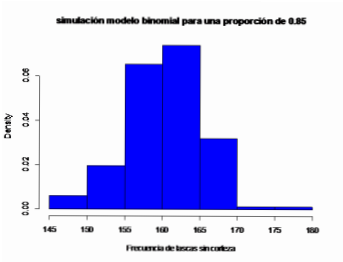
Introducción a los métodos cuantitativos en arqueología con R

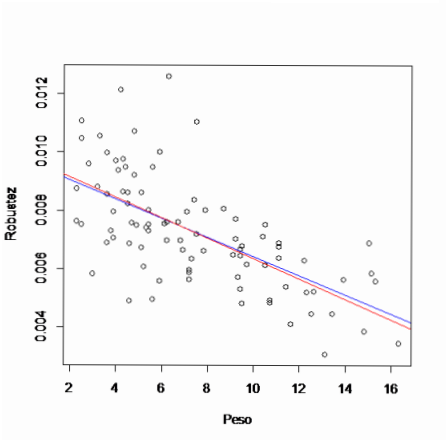
Primera parte: Métodos descriptivos e inferenciales uni y bivariados

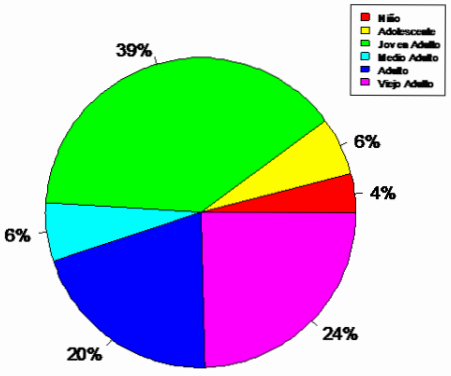


```
   Cell Contents
|-------------------------|
|                       N |
| Chi-square contribution |
|           N / Row Total |
|           N / Col Total |
|         N / Table Total |
|-------------------------|

Total Observations in Table:  49

             | EWBurials$Goods
EWBurials$Sex |    Absent |   Present | Row Total |
--------------|-----------|-----------|-----------|
       Female |        13 |        11 |        24 |
              |     0.267 |     0.236 |           |
              |     0.542 |     0.458 |     0.490 |
              |     0.565 |     0.423 |           |
              |     0.265 |     0.224 |           |
--------------|-----------|-----------|-----------|
         Male |        10 |        15 |        25 |
              |     0.256 |     0.227 |           |
              |     0.400 |     0.600 |     0.510 |
              |     0.435 |     0.577 |           |
              |     0.204 |     0.306 |           |
--------------|-----------|-----------|-----------|
 Column Total |        23 |        26 |        49 |
              |     0.469 |     0.531 |           |
--------------|-----------|-----------|-----------|
```









Marcelo Cardillo

Consejo Nacional de Investigaciones Científicas y Técnicas (CONICET)

Universidad de Buenos Aires

Facultad de Filosofía y Letras.

Introducción a los métodos cuantitativos en arqueología con R : primera parte : métodos descriptivos e inferenciales uni y bivariados / Marcelo Cardillo. - 1a ed ilustrada. - Ciudad Autónoma de Buenos Aires : IMHICIHU - Instituto Multidisciplinario de Historia y Ciencias Humanas, 2018.
Libro digital, PDF

ISBN digital: 978-987-4934-02-4

1. Estadísticas. 2. Arqueología. 3. Software Libre. I. Título.
CDD 310

Versión 1.0- 2018.

Agradecimientos:

Este trabajo fue realizado con el apoyo de CONICET. Al Dr. Ariel Guiance director del IMHICIHU. A D. Hereñú por ayudarme en la edición y J.P Lavagnino por el ISBN.

Prefacio:

El objetivo de este libro es funcionar como una guía introductoria a R orientada a la arqueología, a partir del desarrollo de conceptos y herramientas cuantitativas aplicadas a casos de estudio y problemas arqueológicos. Por consiguiente, no pretende ser un manual de estadística, sino constituir un apoyo para estudiantes de grado y posgrado orientado al manejo R. Asimismo, como es imposible cubrir de forma exhaustiva todas las herramientas que provee este programa, se intentó seleccionar las más ampliamente utilizadas en cursos introductorios o en manuales de estadística básica. Con este fin se busca tomar ventaja de alguna de las ventajas este programa, como su lenguaje comprensivo, abierto, así como de la amplia diversidad de herramientas disponibles de forma libre, incluyendo bases de datos y ayuda on-line (a través de blogs, repositorios, eBooks, foros, sitios, etc.,).

Por último, si bien este libro, a través de sus ejemplos está orientado a problemas arqueológicos se intentó mantener un nivel de generalidad tal que permitiera su uso por estudiantes de otras disciplinas.

Contenidos

Primera parte:

Segunda parte:

Tercera parte

Cuarta parte

Datos

Para los ejemplos se utilizó principalmente el paquete archdata (versión 1.1), (Carlson y Roth, 2016), el cual contiene bases de datos arqueológicos con distintas características, lugares y períodos. De los casos disponibles se seleccionaron tres, que contienen datos en distintos niveles de medición.

Sobre los casos:

Datos categóricos: La base de datos EWBurials(). Tumbas aborígenes del cementerio Ernest Witte, Austin, County, Texas, (U.S.A).

Esta base de datos contiene información sobre cuatro grupos tumbas asignadas a dos períodos temporales diferentes: El grupo 1 que comprende entierros realizados entre el 2000 y el 1200 A.C y el grupo dos con 148 entierros datados entre el 500 y 200 D.C.

Las variables que emplearemos:

Grupo: Cada uno de los períodos temporales, variable categórica ordinal con dos niveles (1,2).

Categoría de edad: Variable categórica ordinal: Infante, Niño, Adolescente, Joven adulto, Adulto, Medio Adulto, Adulto mayor.

Sexo: Variable categórica nominal Femenino y Masculino.

Ajuar: Variable categórica nominal presencia ausencia de bienes.

Datos Cuantitativos: Base de datos DartPoints(): Cinco tipos de puntas de propulsor de Fort Hood, Texas, U.S.A.

Esta base contiene datos métricos y categóricos de 91 puntas de dardo o propulsor, recuperadas en muestreos superficiales en Fort Hood, Texas y clasificados en cinco clases o tipos diferentes.

Las variables que emplearemos:
Clase de punta: Variable categórica nominal, de cinco niveles que representan cada una de las clases: Darl, Ensor, Pedernales, Travis, Wells.
Largo máximo (mm): Cuantitativa continua
Ancho máximo (mm): Cuantitativa continua
Espesor máximo (mm): Cuantitativa continua

Conteos: Frecuencia de doce categorías artefactuales de siete sitios del Achelense en África.
Las variables que emplearemos:
HA Número de hachas de mano
CL Número de hachuelas
KN Número de cuchillos
FS Número de raspadores sobre lasca
D Número de discoides
CS Número de raspadores sobre núcleo
P Número de picos
CH Número de choppers
SP Número de esferoides
OLT Número de grandes instrumentos
SS Número de pequeños raspadores
OST Número de otros pequeños instrumentos

Las bases de datos contienen variables que han sido dejada de lado por el momento con el fin de simplificar su manejo, para más información:
>?1archdata

Primera parte

1. Potenciar el manejo e interpretación de los datos a través de técnicas numéricas.

El empleo de técnicas numéricas para el estudio de los datos arqueológicos es tan antiguo como esta disciplina, si bien sus intereses han cambiado con el tiempo. Sin embargo, el desarrollo de software y de algoritmos que facilitan el análisis de todo tipo de datos ha generalizado el empleo de técnicas de descripción gráfica y numérica así como el contraste de hipótesis. La oferta de estos recursos computacionales (en ocasiones muy específicos) hace imprescindible que el aprendizaje de técnicas cuantitativas deba ir acompañado del manejo de un programa apropiado.

Por otro lado, el empleo generalizado de bases de datos como Excel como herramienta única de almacenamiento y procesamiento de datos cuantitativos presenta importantes limitaciones. Esto puede observarse no tanto en el almacenamiento o creación de bases de datos en sí mismas, siempre crecientes, sino en el manejo y análisis de los mismos. En parte por este motivo, programas de acceso libre específicamente diseñados para el análisis estadístico, como R o Past han comenzado a ser utilizados con fines didácticos en cursos y seminarios de estadística y arqueología.

Algunas de las ventajas de estos programas son el acceso libre, el tipo y/o la diversidad de herramientas analíticas y gráficas que contienen. Con respecto al Past (Hammer y Harper 2001), Barceló (2008) a publicado un libro de introducción a la estadística con este programa. Past posee una interfase amigable con pestañas/ventanas y gran cantidad de herramientas de utilidad para el arqueólogo. En comparación R es en primera

instancia, algo menos accesible ya que como veremos, funciona a partir de comandos y sentencias o scripts. Sin embargo, R posee una mayor flexibilidad, potencia y mayor número de recursos analíticos que la mayoría de los programas, sean o no de acceso libre.

1.2. Acerca de este libro.

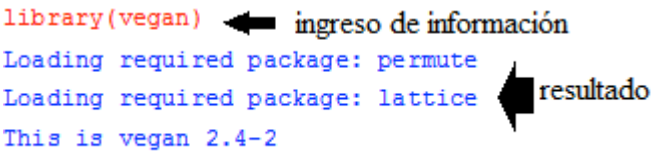
Este libro intenta abarcar los métodos descriptivos e inferenciales univariados y bivariados más comúnmente utilizados y está pensado para complementar el contenido teórico aportado por libros de estadística para arqueólogos como el clásico de Orton (1988), Shennan (1992), Drennan (1996) o el más recientemente publicado, Van pool y Leonard (2010). La breve descripción de los métodos presentes en cada acápite está orientada fundamentalmente, a contextualizar el método en relación a un problema arqueológico determinado, más que a exponer o explicar el procedimiento en sí mismo. Sin embargo, se buscó detallar los aspectos mínimos que hacen al uso e interpretación de cada una de estas aplicaciones, evitando en lo posible recurrir a lenguaje técnico o cuantitativo más elaborado.

En cuanto al formato y la presentación de datos, éste se ajusta en gran medida, al de los tutoriales que presentan los distintos paquetes que se utilizan en R. El tamaño de letra fue modificado en casos que las matrices fuesen muy extensas, para facilitar su visualización.

Siguiendo el formato "por defecto" de la consola de R, los comandos fueron escritos siempre en rojo acompañados de paréntesis. Esto indica que dentro de éste va el nombre de la matriz de datos o del objeto R. El resultado de la aplicación

de las funciones, tal como aparecen en la consola de R, son representadas en azul.

Ejemplo:



El símbolo "#" se utiliza para realizar notas o aclaraciones que acompañan a los comandos. R no registra las líneas que están precedidas por este signo, por consiguiente, aunque los comandos estén acompañados de notas, éstos se pueden copiar, pegar y ejecutar directamente en la consola de R.

El símbolo ">" (mayor que) precede todas las líneas de comando de R y es respetado aquí, aunque no debe copiarse y pegarse directamente en la consola.

Por último, la forma de generar gráficos o de escribir los distintos comandos en este libro no es del todo homogénea, ya que se intentó mostrar alguno de los distintos procedimientos o alternativas que se presentan para la manipulación de los objetos.

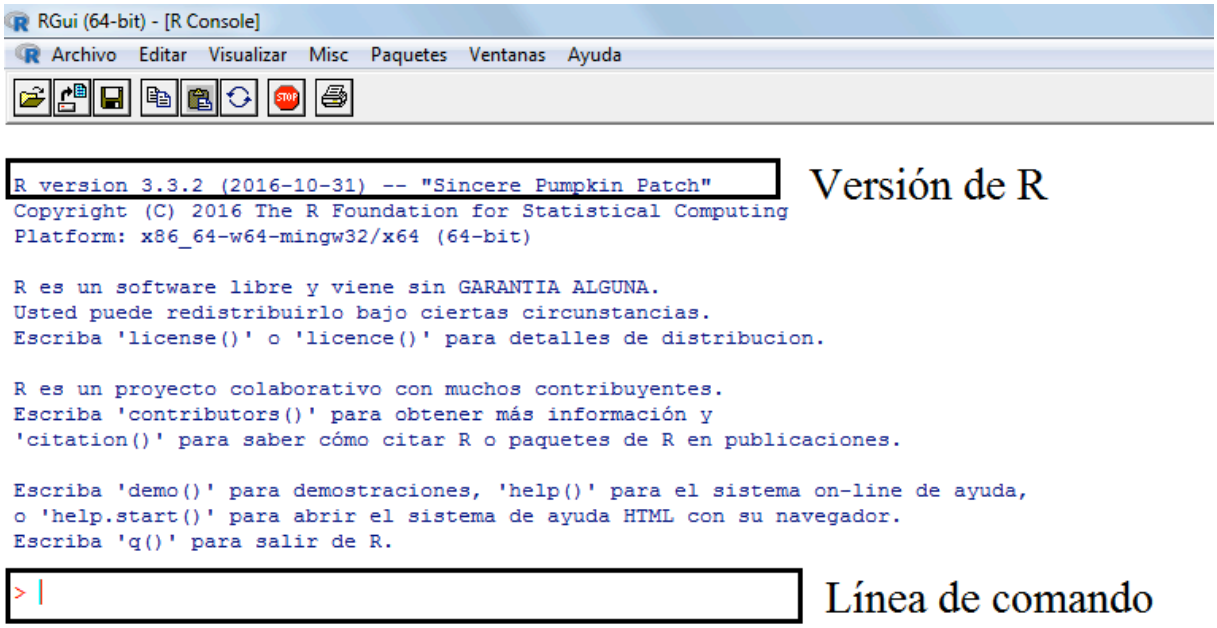
Todos los datos presentados aquí y los ejemplos fueron analizados través del programa R versión 3.3.2. (R core team 2016). Los datos provienen de dos fuentes: el paquete archdata versión 1.1 (Carlson y Roth 2016) y otros generados por el autor.

1.3. Instalando R, paquetes y repositorios.

R consta de más de 9800 paquetes en el repositorio central CRAN, existiendo otros repositorios particulares que en

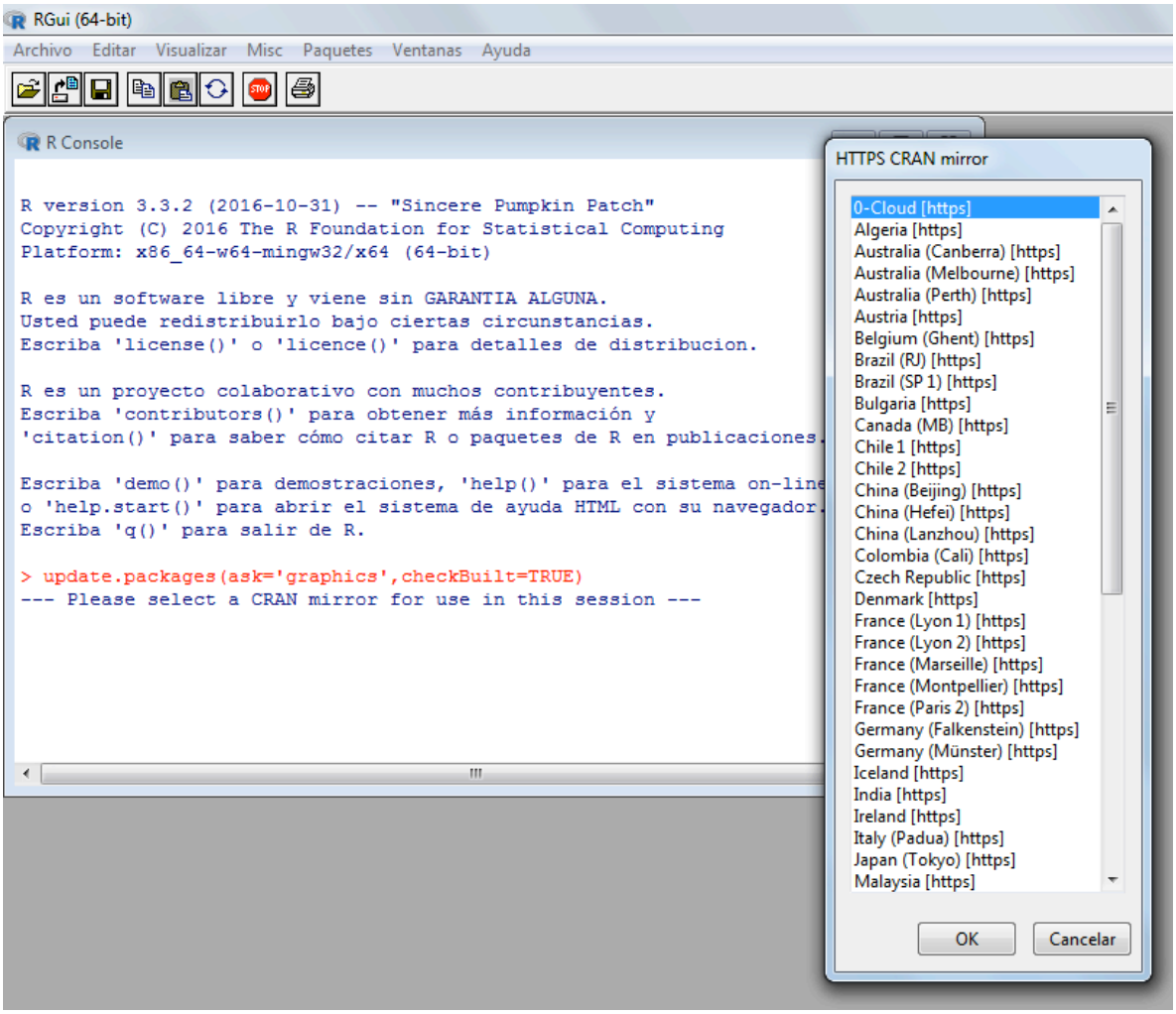
general, contienen conjuntos de paquetes específicos. Se debe tener en cuenta que muchos paquetes son alternativas de los mismos procedimientos estadísticos o permiten potenciar interfaces gráficas, numéricas o están vinculados a la edición y programación dentro del lenguaje R. R puede instalarse bajo distintas plataformas de UNIX, Windows y MacOS. Al instalar R del repositorio (https://cran.r-project.org/), éste lo hace con un conjunto de paquetes base que permiten la mayoría de los procedimientos que vamos a realizar aquí. También emplearemos algunos no presentes en esta base y que permiten el manejo de algunos tipos de datos con mayor eficiencia.

La ventana central de R es muy sencilla y presenta la información referida a la versión del programa sobre la línea de comandos, siempre precedida por el símbolo ">". La barra superior permite el acceso a operaciones básicas como abrir, cerrar, salvar y actualizar R, entre otras.



Ventana principal o consola de R.

Tal como se mencionó, la actualización o instalación de estos paquetes puede hacerse desde los muchos repositorios asociados directamente a CRAN a través de la ventana Paquetes/instalar paquetes/seleccionar repositorios. Una vez seleccionados se depositan automáticamente en la carpeta "library" del programa. La página central donde se puede bajar la base y el sistema R, además de paquetes y manuales es https://CRAN.R-project.org.



Ventana de R con los repositorios disponibles para actualizar o bajar paquetes.

También es posible instalar un paquete desde la misma consola de R, mediante la función install.packages, por ejemplo:

```
>install.packages("vcdExtra")#en este el nombre entre comillas ("")
```

Para utilizarlo, sólo tenemos que llamarlo dentro de la consola con la función library:

```
>library(vcdExtra)# sin ("")
```

De esta manera pueden abrirse todos lo paquetes que sean necesarios, si se quiere cerrar alguno de ellos, puede emplearse la función detach:

```
>detach(package:vcdExtra)
```

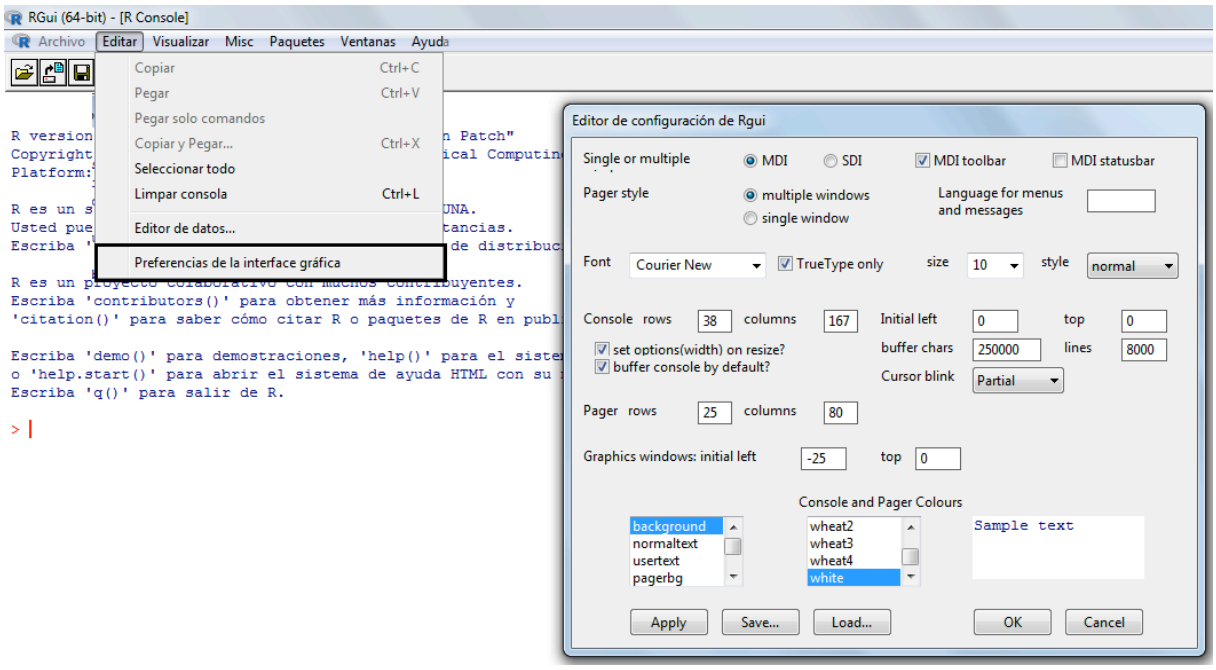
Como R va ampliándose y modificándose de manera continua es necesario actualizarlo regularmente. Esto puede hacerse fácilmente mediante el paquete installr:

```
>library(installr)
>updateR() # actualizar R
```

El proceso puede ser largo ya que incluye la copia de los paquetes que están instalados en la librería de R de la computadora y su actualización. Opcionalmente pueden conservarse los archivos anteriores, así como la versión de R correspondiente.

Los paquetes, pueden a su vez, actualizarse por separado a través de la ventana paquetes/actualizar paquetes, donde el programa, luego de elegir un repositorio, busca automáticamente aquellos paquetes que presentan actualizaciones y las descarga.

Por último, la ventana de R puede modificarse a través de la pestaña de preferencias gráficas, donde puede cambiarse tamaño de letra, tipo y formato, aunque el modelo que aparece por defecto es en general, el más recomendable.



Alternativas de interfaz gráfica en R.

1.4. Lenguaje básico de R.

R maneja muy variado tipo de datos, desde objetos simples que pueden ser factores representados por un código numérico o alfabético ("grande","mediano","pequeño" o 1, 2, 3) a símbolos numéricos, vectores y matrices mixtas. Asimismo, R puede operar con elementos más complejos como fórmulas.

Ejemplo:

```
>F<-function(x) {log(x+1)}# Con la función "function" se genera una función que se denomina F y que puede aplicarse a todos los "x" (x) a los que suma 1 y calcula luego el logaritmo de cada valor.
```

```
>Datos<-c(2,4,5,6,7,5,3,11)#genera un vector con estos valores numéricos
>Datos# al llamarlo automáticamente obtengo el contenido del objeto
[1]  2  4  5  6  7  5  3 11
```

```
>DatosTransformados<-F(Datos)#aplico la función, el símbolo <- indica que esta función se asigna a un nuevo objeto al que llamaré "DatosTransformados". También esta asignación puede hacerse con el símbolo "=".
```

```
>DatosTransformados
[1]  1.098612  1.609438  1.791759  1.945910  2.079442  1.791759 1.386294 2.484907
```

Algunas de las funciones básicas son las siguientes:

-	Menos
+	Más
!	No
~	Depende de, se distribuye como
?	Ayuda
:	Interacción (por ejemplo, en el marco de una regresión)
*	Multiplicación
/	División
^	Exponenciación
<	Menos que
>	Más que
==	Igual a
$	Listar subconjunto de datos u operar sobre ellos: Por ejemplo, la variable "Color", dentro de la matriz "Aspecto", puede llamarse escribiendo Aspecto$Color

1.5. Vectores y Matrices.

Tal como se mencionó R maneja distintos formatos de datos que incluyen tantos valores numéricos como categóricos, de una o más columnas.

La forma numérica más sencilla es el escalar, que está representado por un solo valor, al operar con escalares en R se tiene las mismas o mayores posibilidades que con una calculadora.

Multiplicación de dos escalares:

```
>2*100
[1] 200
```

También podemos generar objetos que contienen la magnitud de estos escalares e ingresar directamente la fórmula:

```
>A<-100
>B<-3.5
>D<-0.5
>(A*B)/D
[1] 700
```

Una forma algo más compleja es un vector, que puede ser tanto una fila como una columna y comprende un conjunto de valores numéricos o niveles de un factor que describen una variable:

```
>Peso<-c(11,12,4,23,12,6)#si los valores numéricos se ingresan manualmente deben ir precedidos del indicador columna "c" y de comas como separador.
>Peso
[1] 11 12  4 23 12  6
```

Un conjunto de vectores constituyen una matriz, donde más de una variable con distintos niveles de medición pueden combinarse.

Ejemplo:

```
>Matriz<-matrix(1:10,  nrow=5,ncol=5)  #construimos una matriz "Matriz", con valores de 1 a 10 (1:10), 5 filas (nrow) y 5 columnas (ncol).

>Matriz
     [,1] [,2] [,3] [,4] [,5]
[1,]    1    6    1    6    1
[2,]    2    7    2    7    2
[3,]    3    8    3    8    3
[4,]    4    9    4    9    4
[5,]    5   10    5   10    5
```

De manera similar que con los escalares o con los vectores, las mismas funciones de cálculo se pueden aplicar a las matrices:

```
>Matriz^2#Elevar al cuadrado
     [,1] [,2] [,3] [,4] [,5]
[1,]    1   36    1   36    1
[2,]    4   49    4   49    4
[3,]    9   64    9   64    9
[4,]   16   81   16   81   16
[5,]   25  100   25  100   25
```

Téngase en cuenta que a menos que yo asigne un objeto con el mismo nombre (por ejemplo: Matriz<- Matriz^2), R no reescribe o modifica los objetos. Asimismo, si a la función no se le asigna un objeto (como en el ejemplo precedente), el resultado no se almacena, sino que simplemente queda en la consola.

Cada elemento dentro de una matriz puede ser identificado por su posición dentro de ésta. Como ocurre en general dentro de los programas estadísticos, la descripción de una matriz detalla primero las filas y luego las columnas: (Misdatos[Filas,Columnas]).

Ejemplo:

```
>Matriz[2,]# Segunda fila de la matriz.
>Matriz [,5]#Quinta columna de la matriz.
>Matriz [1:3,1:2]#Filas 1 a 3 de las columnas 1 a 2.
```

Ejemplo:

```
>Matriz[1:3,1:2]
     [,1] [,2]
[1,]    1    6
[2,]    2    7
[3,]    3    8
```

Podemos extraer todos, parte, o un solo caso, como aquí el tercero de la segunda columna:

```
>Matriz[3,2]
[1] 8
```

La forma más general de un matriz en R es aquella que puede estar compuesta por muy distintos tipos de datos, tanto numéricos como factores. Por ejemplo, podemos construir con un factor denominado "color" con dos niveles, "blanco" y "negro":

```
>Color<-c("blanco", "blanco", "negro", "negro", "blanco")
```

Podemos construir un data.frame() denominado "Matriz":

```
>Misdatos<-data.frame(Matriz,Color)#agrupamos el objeto "Matriz" anterior al factor "Color".
```

Luego, podemos preguntar a R sobre la naturaleza de este nuevo objeto, en algunos casos, R responde como "cierto" True o "falso" F:

```
>is.matrix(Misdatos)
[1] FALSE
>is.data.frame(Misdatos)
[1] TRUE

>Misdatos
  X1 X2 X3 X4 X5  Color
1  1  6  1  6  1 blanco
2  2  7  2  7  2 blanco
3  3  8  3  8  3  negro
4  4  9  4  9  4  negro
5  5 10  5 10  5 blanco
```

Más detallado es el comando "str", que devuelve el contenido y características de la matriz:

```
>str(Misdatos)
'data.frame':   5 obs. of  6 variables:
 $ X1    : int  1 2 3 4 5
 $ X2    : int  6 7 8 9 10
 $ X3    : int  1 2 3 4 5
 $ X4    : int  6 7 8 9 10
 $ X5    : int  1 2 3 4 5
 $ Color: Factor w/ 2 levels "blanco","negro": 1 1 2 2 1.
```

Aquí se observa el tipo de matriz, el número de casos y variables, su contenido y en el caso del factor "color" los niveles de esta variable nominal.

1.6. Comenzando con R, manejo de objetos.

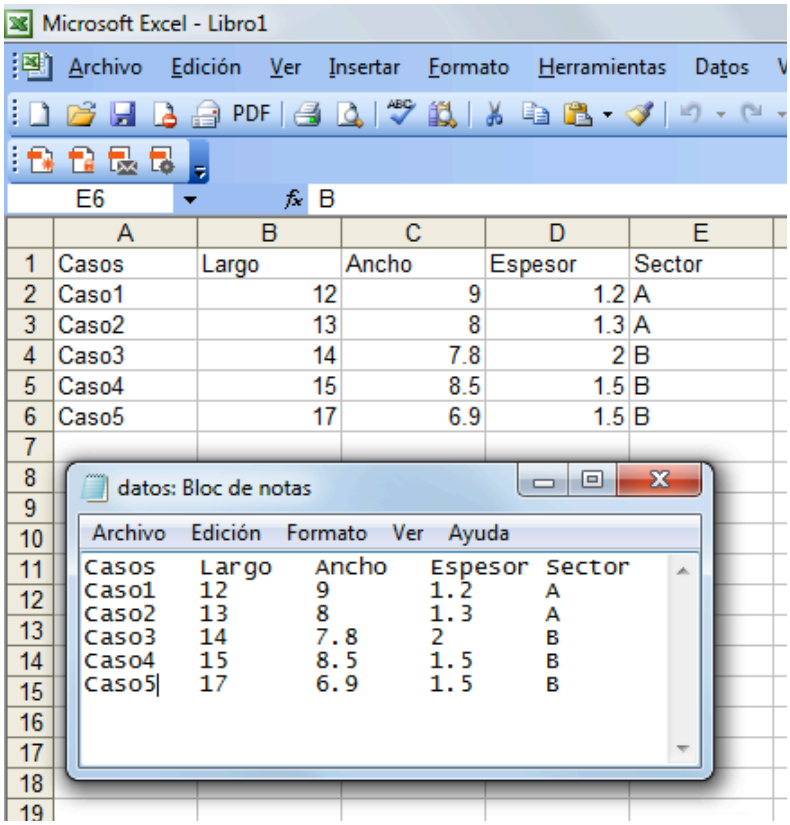
El programa puede manejar una muy amplia variedad de objetos con formatos disímiles, incluyendo datos almacenados en el portapapeles. Es importante conocer previamente la naturaleza de la matriz de datos, así como la extensión (xls, txt, cvs) para poder detallarla en R. De lo contrario, ésta no podrá ser leída o el programa lo hará de forma incompleta.

Veremos los más frecuentemente utilizados:

Misdatos<- read.table("datos".txt) Misdatos<- read.table("datos.txt", header=T, row.names=1)	Leer archivos de texto separados por tabulaciones (txt). Especifica que las columnas tienen nombre y que la primera columna contiene el nombre de los casos (podría ser cualquier columna). De no especificarlos, asume que éstos no se toman en cuenta (F, False)
Misdatos<- read.csv("datos.csv")	Leer archivos separados por comas (csv)

Misdatos<- loadWorkbook("datos.xls")	Leer archivos xls. Necesita instalar previamente el paquete XLConnect. Library(XLConnect)
Misdatos<- read.table("clipboard", header=TRUE, row.names=1, sep="\t")	Leer los archivos directamente de la papelera después de haberlos copiado de otro programa.
setwd("C:/Mis documentos/") getwd()	Especificar un directorio particular desde el cual lee los archivos. Por defecto, los lee de documentos en Word. El comando getwd() indica cual es el directorio actualmente en uso, por ejemplo: getwd() [1] "C:/Users/M/Documents"

Es fácil preparar un archivo txt con el bloc de notas, pegando de otros formatos (por ejemplo de Excel)



Archivo de Excel copiado y pegado en un bloc de notas (.txt), es un formato fácil de leer, aunque no deben dejarse espacios, por ejemplo "Casos 1", tiene que ser "Casos1" o "Casos_1".

```
>Datos<-read.table("datos.txt",header=T,     row.names=1)#row
```

names detalla en que fila se encuentran los nombres de las variables, header la existencia –o no- de encabezado de las variables. Si se omite este comando (o se escribe header=F) R interpreta que las variables no poseen encabezado y automáticamente les asigna un valor, lo mismo ocurre con las filas o casos.

```
>Datos
      Largo Ancho Espesor Sector
Caso1    12   9.0     1.2      A
Caso2    13   8.0     1.3      A
Caso3    14   7.8     2.0      B
```

```
Caso4    15   8.5     1.5      B
Caso5    17   6.9     1.5      B
```

En este caso, el objeto "Datos" posee tres variables cuantitativas (Largo, Ancho, Espesor) y una categórica "Sector". Es importante tener en cuenta que para llamar o nombrar variables o archivos, deben referirse tal cual están en el archivo, sino devuelve error.

Por último una posibilidad más rápida cuando se trabaja desde una base de datos como Excel, es copiar las filas y columnas de interés y desde R, leer directamente desde el portapapeles:

```
>Misdatos<-read.table("clipboard", header=TRUE, row.names=1,
sep="\t")
```

En este caso, se debe especificar la fuente (clipboard) y la tabulación (sep="\t"). También, para muchos comandos, R acepta que se coloque la abreviatura tanto como la palabra completa, como en vez de "FALSE", simplemente "F".

1.7. Reescribir-generar objetos en R.

Es fácil duplicar los archivos sin modificar los anteriores creando un nuevo objeto:

```
>Datos2<-Datos
>Datos2
      Largo Ancho Espesor Sector
Caso1    12   9.0     1.2      A
Caso2    13   8.0     1.3      A
Caso3    14   7.8     2.0      B
Caso4    15   8.5     1.5      B
Caso5    17   6.9     1.5      B
```

Esto es ventajoso ya que permite generar archivos con distintas combinaciones de las variables o generar modificaciones en ellos sin perder los datos originales.

El operador $ indica que Largo está dentro de objeto Datos2, por ejemplo, podemos ver sólo este vector tipeando:

```
>Datos2$Largo
[1] 12 13 14 15 17
```

Por último, se puede ver que la última variable es un factor con dos niveles A y B y no una variable continua, esto puede constatarse haciendo la pregunta:

```
is.factor(Datos2$Sector)
[1] TRUE
```

Se puede modificar un factor y reemplazarlo por números:

```
>Datos2$Sector2<-as.numeric(Datos2$Sector)
>Datos2
      Largo Ancho Espesor Sector Sector2
Caso1    12   9.0     1.2      A       1
Caso2    13   8.0     1.3      A       1
Caso3    14   7.8     2.0      B       2
Caso4    15   8.5     1.5      B       2
Caso5    17   6.9     1.5      B       2
```

En este caso, creamos un nuevo vector que archivamos dentro del mismo objeto y lo llamamos Sector2 (Datos2$Sector2).

Lo contrario puede hacerse con valores numéricos, que pueden transformarse en niveles de un factor:

```
>Datos2$Ancho2<-as.factor(Datos2$Ancho)
```

```
>Datos2
      Largo Ancho Espesor Sector Sector2 Ancho2
Caso1    12   9.0     1.2      A       1      9
Caso2    13   8.0     1.3      A       1      8
Caso3    14   7.8     2.0      B       2    7.8
Caso4    15   8.5     1.5      B       2    8.5
Caso5    17   6.9     1.5      B       2    6.9
```

Es posible también, crear variables bajo distintas condiciones predefinidas como la forma o naturaleza de la distribución. En el caso de variables cuantitativas, el más común es el comando "rnorm" que genera una variable con distribución normal de extensión y parámetros predefinidos (como su media y desvío estándar).

```
>a<-rnorm(20)#crear una variable de distribución normal y 20
casos
>head(a)
[1] -1.3451549 -0.1444586 -0.8906431  0.6468882 -1.2548471 -
1.0334437
```

Definiendo un parámetro:

```
>a<-rnorm(20,mean=5)#20 casos y media de 5
>a
 [1] 6.748111 4.747946 4.853979 5.150027 3.025029 4.432093
4.475280 5.333538
 [9] 4.390996 5.920494 5.495944 4.656308 5.104143 4.658074
4.462880 4.235075
[17] 3.781555 6.061406 5.800750 5.619928
```

```
>summary(a)#summary devuelve el resumen numérico de la
variable o de la matriz.
```

```
     Largo
 Min.   :3.025
 1st Qu.:4.455
 Median :4.801
 Mean   :4.948
 3rd Qu.:5.527
 Max.   :6.748
```

Se comprueba que la media de esta variable aleatoria continua es de 5.

Como veremos más adelante, esta función es muy útil ya que permite generar variables que luego pueden emplearse para explorar distintos test, especialmente los paramétricos (basados en parámetros conocidos como la media).

1.8. Borrar, agrupar o reordenar filas o columnas.

Para extraer una columna o un conjunto de ellas de una matriz y generar un nuevo objeto, se pueden emplear los mismos procedimientos que vimos en el acápite anterior, por ejemplo:

```
>Matriz2<- Misdatos[,5]#del data.frame "Misdatos" de la página 23
>Matriz2
[1] 1 2 3 4 5
```

Una forma sencilla es nombrar directamente la variable que se desea extraer (no su orden numérico como en el caso anterior), utilizando la función que indica su posición subordinada (o anidada) dentro de un conjunto mayor ($).

```
>Matriz2<-data.frame(Misdatos$X3, Misdatos$Color)
>Matriz2
 Misdatos.X3 Misdatos.Color
```

```
1          1       blanco
2          2       blanco
3          3        negro
4          4        negro
5          5       blanco
```

Podemos ahora renombrarlo:

```
>names(Matriz2)<-c("X3","Color")#recordar que al generar un objeto con nombres, éstos siempre van entre comillas.
>Matriz2
  X3  Color
1  1 blanco
2  2 blanco
3  3  negro
4  4  negro
5  5 blanco
```

O extraer filas:

```
>Matriz2<-data.frame(Misdatos[3:4,])#Tercera y cuarta fila
>Matriz2
  X1 X2 X3 X4 X5 Color
3  3  8  3  8  3 negro
4  4  9  4  9  4 negro
```

Existen distintos paquetes y funciones específicas para operar, mover, reordenar y extraer matrices. En el caso de querer borrar una fila o columna la forma más sencilla es la siguiente:

```
>Matriz3<-data.frame(Misdatos[-3])#Creamos un nuevo objeto que excluye la columna 3
>Matriz3
  X1 X2 X4 X5  Color
1  1  6  6  1 blanco
2  2  7  7  2 blanco
3  3  8  8  3  negro
4  4  9  9  4  negro
5  5 10 10  5 blanco

>Matriz4<-data.frame(Misdatos[-2,])])#Creamos un nuevo objeto que excluye la fila 2

>Matriz4
  X1 X2 X3 X4 X5  Color
1  1  6  1  6  1 blanco
3  3  8  3  8  3  negro
4  4  9  4  9  4  negro
5  5 10  5 10  5 blanco
```

Tal como lo mencionamos anteriormente, es importante tener en cuenta que si el objeto a crear posee el mismo nombre que el original, esto reescribirá los datos. Sin embargo, utilizando el operador ($), podemos agregar, a nuestra matriz otra columna anidada con un nombre específico:

```
>Misdatos$X1X2<-Misdatos$X1*Misdatos$X2#Agregamos una nueva columna que el resultado de la operación multiplicativa entre la variable X1 y X2 a la que llamamos "X1X2".

>Misdatos
  X1 X2 X3 X4 X5  Color X1X2
1  1  6  1  6  1 blanco    6
2  2  7  2  7  2 blanco   14
3  3  8  3  8  3  negro   24
```

```
4  4  9  4  9  4  negro    36
5  5 10  5 10  5 blanco    50
```

Esta operación es útil cuando se quiere incluir una nueva variable que resulta de la interacción de otras dos, por ejemplo, en el marco de una regresión. También esta variable puede ser más compleja, como el cálculo del volumen o un índice.

1.9. Reordenamiento de variables.

En ocasiones, hay necesidad de ordenar los datos para poder analizarlos mejor o para extraer un subconjunto de filas o columnas para generar un nuevo objeto. Esto puede realizarse con la función order():

```
>MatrizOrdenada<-Misdatos[order(Color),]#En este caso, la ordenamos por color, sacando provecho de la existencia de un factor "Color".
```

```
>MatrizOrdenada
  X1 X2 X3 X4 X5  Color X1X2
1  1  6  1  6  1 blanco     6
2  2  7  2  7  2 blanco    14
5  5 10  5 10  5 blanco    50
3  3  8  3  8  3  negro    24
4  4  9  4  9  4  negro    36
```

En el caso de querer ordenar los datos a partir de una variable cuantitativa, el programa por defecto lo hace de menor a mayor, de querer el sentido inverso debe agregarse el signo (-).

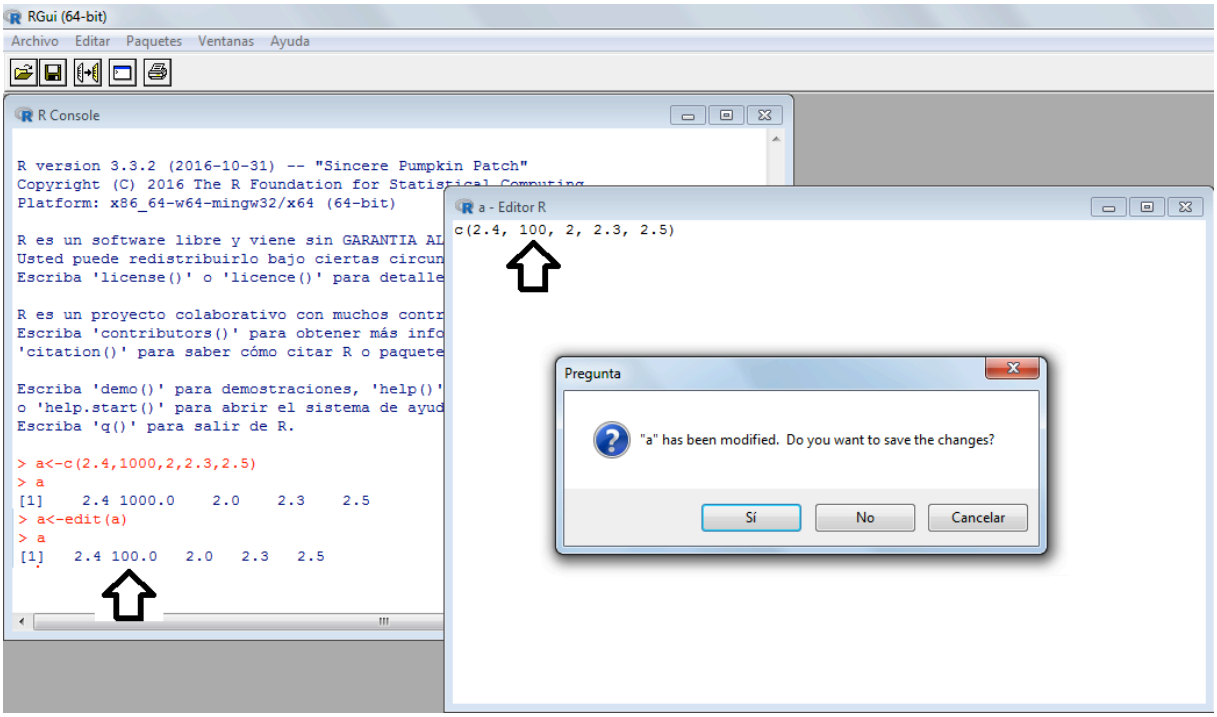
```
>MatrizOrdenada<-    Misdatos[order(Misdatos$X1X2),]#En    este
```
caso, se debe especificar el subconjunto ($).

```
>MatrizOrdenada
  X1 X2 X3 X4 X5  Color X1X2
1  1  6  1  6  1 blanco    6
2  2  7  2  7  2 blanco   14
3  3  8  3  8  3  negro   24
4  4  9  4  9  4  negro   36
5  5 10  5 10  5 blanco   50
```

Es posible también editar la matriz de datos, por ejemplo, suponiendo que se identifica un error de tipeo, a través del comando edit(). Esto abre una planilla con los datos que entonces pueden ser modificados (o pueden agregarse nuevos datos).

Ejemplo:

Se reescribe el objeto "a" y se cambia mediante el comando edit() el valor 1000 por 100.



Esquema del empleo del comando edit().

1.10. Recodificación de las variables.

La recodificación consiste en cambiar el nivel o escala de de una variable, por ejemplo, al expresar una variable cuantitativa (por ejemplo, años) en intervalos de distinta amplitud. Este es el caso de los intervalos de edad (como niño, adulto, anciano) o intervalos de tamaño (pequeño, mediano, grande). La recodificación permite transformar una variable cuantitativa en categórica ordinal al dividirla en niveles, lo que puede tener fines descriptivos (cuantos casos entran en estas nuevas categorías?) o analíticos (test de hipótesis).

Por ejemplo, nos puede interesar analizar que frecuencia de puntas de proyectil de distintas clases que a su vez, están comprendidas dentro de diferentes categorías de tamaño. Para ello creamos una matriz de datos a partir del archivo

DartPoints tomando el largo de las puntas de proyectil. Con este fin utilizamos algunos de los siguientes operadores:

<	Menor que
>	Mayor que
=	Igual que. Al combinarse con > o < se genera los comandos Menor o igual <= o mayor o igual >=
&	Y además. Por ejemplo el intervalo "20 & 30 cm".

Armaremos primeramente, una matriz de datos:

```
>tamaño<-data.frame(DartPoints$Name, DartPoints$Length)
>names(tamaño)<-c("Clase","Largo")#opcional, para renombrar las variables
```

Generemos dentro de la variable largo, una nueva columna con la misma variable codificada (tamaño$Largo2) en tres intervalos de tamaño "Grande", "Mediano" y "Pequeño".

```
>tamaño$Largo2[tamaño$Largo >=80] <-"Grande"
>tamaño$Largo2[tamaño$Largo >=50 & tamaño$Largo <80] <-"Mediano"
>tamaño$Largo2[tamaño$Largo <50] <-"Pequeño"
>head(tamaño,20)#por razones de practicidad veamos los primeros 20. Por defecto, el comando head() muestra los 6 primeros casos.
```

```
  Clase Largo  Largo2
1  Darl  42.8 Pequeño
2  Darl  40.5 Pequeño
3  Darl  37.5 Pequeño
4  Darl  40.3 Pequeño
5  Darl  30.6 Pequeño
6  Darl  41.8 Pequeño
7  Darl  40.3 Pequeño
```

```
8        Darl  48.5 Pequeño
9        Darl  47.7 Pequeño
10       Darl  33.6 Pequeño
11       Darl  32.4 Pequeño
12       Darl  42.2 Pequeño
13       Darl  33.5 Pequeño
14       Darl  41.8 Pequeño
15       Darl  38.0 Pequeño
16       Darl  35.5 Pequeño
17       Darl  31.2 Pequeño
18       Darl  34.5 Pequeño
19       Darl  33.1 Pequeño
20       Darl  32.0 Pequeño
```

Ahora es posible utilizar esta variable codificada como un factor en análisis de distinto tipo, en principio, podemos construir una tabla para ver el comportamiento de la nueva variable. La función xtabs() que veremos en detalle más adelante, permite de forma rápida generar una tabla a partir de datos categóricos:

```
>xtabs(~Largo2+Clase,  data=tamaño)# Largo2 y Clase, se
```

distribuyen (~) en relación a la frecuencia de las distintas categorías.

```
         Clase
Largo2    Darl Ensor Pedernales Travis Wells
  Grande     0     0          2      0     0
  Mediano    2     1         19      5     6
  Pequeño   26     9         11      6     4
```

Se observa que las categorías de tamaño se distribuyen desigualmente en relación a las clases de puntas de proyectil, ya que la clase "Pedernales" posee una mayor frecuencia de tamaño mediano y grande. Como veremos en el capítulo 3 esto puede contrastarse mediante un test de hipótesis.

1.11. Datos faltantes.

El manejo de los datos faltantes es un problema que se presenta de forma habitual durante la obtención y procesamientos de la información. Estos faltantes pueden ser producto de errores de muestreo, tipeo, o de la ausencia del atributo en una variable.

Comúnmente esta falta se codifica como "NA" y el programa automáticamente la reconoce como tal.

```
>X1<-c(3,2.3,3.5,4.6,NA,7.2)
```

Una forma sencilla de identificar estos faltantes, especialmente cuando la muestra es de tamaño reducido es preguntar a R sobre el objeto de interés:

```
>is.na(X1)
[1] FALSE FALSE FALSE FALSE  TRUE FALSE
```

En este caso, la quinta entrada es un dato faltante (TRUE).

O por el contrario, se puede preguntar cuáles están completos:

```
>complete.cases(X1)
[1]  TRUE  TRUE  TRUE  TRUE FALSE  TRUE
```

Uno de los problemas que ocasionan los datos faltantes es en el cómputo o procesamiento de las variables. Por ejemplo, intentemos estimar la media mean().

```
>mean(X1)
[1] NA
```

El resultado señala que la media, no se puede computar directamente por la presencia de un faltante. Lo primero que podemos hacer es utilizar un comando que indique que se deben omitir el o los datos faltantes con el comando "na.rm=TRUE":

```
>mean(X1,na.rm=TRUE)#calcular la media, omitiendo el o los faltantes
[1] 4.12
```

La segunda opción es crear primero un nuevo objeto con el comando "na.omit" que deje de lado este faltante:

```
>X2<- na.omit(X1)
>X2
[1] 3.0 2.3 3.5 4.6 7.2
attr(,"na.action")
[1] 5
attr(,"class")
[1] "omit"
```

Ahora podemos calcular directamente el parámetro:

```
>mean(X2)
[1] 4.12
```

Sin embargo, eliminar los faltantes puede reducir significativamente el tamaño de la muestra, especialmente si ésta es relativamente pequeña. Una estrategia comúnmente empleada en estos casos es reemplazarlos por la media de la variable. Esto puede realizarse con el paquete zoo().

```
>library(zoo)
>X3<- na.aggregate(X1)
>X3
[1] 3.00 2.30 3.50 4.60 4.12 7.20
```

Como vemos, esta función reemplazó automáticamente el valor faltante con la media (4.12). Hay que tener en cuenta, sin embargo, que si hay muchos datos faltantes, el reemplazo por la media tiende a homogeneizar artificialmente la varianza original de la muestra.

1.12. Muestreo y variables aleatorias con R.

Generar números aleatorios o seleccionar escalares de una muestra ya existente, puede ser de gran utilidad en el muestreo, simulación, o en la puesta a prueba de la performance de distintos tipos de análisis.

R puede generar números aleatorios bajo muy distintas distribuciones. La función básica para generar números aleatorios en R es sample():

```
>sample(1:20,10,replace=T)
```

Los números aleatorios son "muestreados" de una distribución cuya extensión es de 1 al 20 (1:20) y en una muestra de tamaño(10).

```
 [1] 14 15 13  4 10 11 16 10 10 14
```

Si repetimos la función se obtienen, cada vez, resultados diferentes. La opción replace=T, indica en este caso que los números pueden repetirse. Si por el contrario replace=F, un mismo número no puede aparecer dos veces:

```
>sample(1:20,10,replace=F)
 [1]  5 15 13  3  9 14 17 16  8 18
```

De esta manera se puede generar una etiqueta numérica o valor para numerar casos o seleccionarlos como parte de un muestreo. También, puede generarse un factor aleatorio con un determinado número de niveles:

```
>sample(c("A","B","D"), 10, replace = TRUE)#Un factor de tres niveles (A,B,C) de tamaño 10.
 [1] "A" "A" "B" "D" "A" "A" "D" "B" "A" "B"
```

La función sample(), también puede tener como objeto muestrear una distribución ya existente y generar una o más muestras que pueden ser menor, mayor o iguales que la original. El generar pseudomuestras o pseudoréplicas con reemplazo a partir de una matriz de datos original, es la base de los procedimientos de remuestreo o bootstrap. En este caso, se deben generar *n* muestras del mismo tamaño sobre las cuales se calculan los parámetros poblacionales.

Por ejemplo, podemos tomar una muestra aleatoria de alguna de las variables métricas de las bifaces Achelenses del paquete archdata().

```
>data(Acheulean)
>Acheulean
                 Lat  Long  HA  CL KN FS  D  CS  P CH SP OLT  SS OST
Olorgesailie   -1.58 36.45 197  96 58 17  5  11  3 32 52   6 213 218
Isimila        -7.90 35.61 246 208 30 28  6  30 16 62 17  15  98  64
Kalambo Falls  -8.60 31.24 337 264 59 96  8 124 18 69  6  17 303  48
Lochard       -19.92 29.02  45  13  3  2 12   1  0 32  3   8  46  22
Kariandusi     -0.45 36.26 132  56 47 23  3   5  7  6  5   8  17  25
Broken Hill   -14.43 28.45   1   8  1  1  0   1  0  4 25   0  35  18
Nsongezi       -1.03 30.78  15  19  2  9  1  28  1 19  0  10  17  70
```

```
>Muestra1<-sample(Acheulean$HA,replace = TRUE)#repetiremos la
operación cuatro veces, remuestreando la variable HA
>Muestra2<-sample(Acheulean$HA,replace = TRUE)
>Muestra3<-sample(Acheulean$HA,replace = TRUE)
>Muestra4<-sample(Acheulean$HA,replace = TRUE)
>Muestras<-data.frame (Muestra1, Muestra2, Muestra3,
>Muestra4)#Ahora pueden ser agrupadas mediante la función
data.frame.

>Muestras
  Muestra1 Muestra2 Muestra3 Muestra4
1      246        1       15       45
2       45      197      246      246
3       15        1      132      132
4      337        1       15      132
5       45      132       45      197
6      337        1      197      246
7      132       15       15        1
```

Se observa que estas pseudorréplicas de la variable original poseen la misma extensión, pero poseen diferencias de la original, ya que cada caso puede ser seleccionado al azar más de una vez en cada réplica.

R puede también, generar distribuciones bajo distintos modelos de probabilidad, como la distribución normal, uniforme, exponencial, etc. La más utilizada es la que genera valores aleatorios esperados bajo una distribución normal (rnorm), la cual mencionamos en acápites anteriores y la distribución uniforme (runif). En estos casos es posible establecer parámetros como la media o la varianza de la distribución.

1.13. Exportar datos y gráficos desde R.

1.13.1. Exportar datos.

Existen distintas opciones para exportar datos generados por R, por ejemplo, los resultados de un análisis. La forma más sencilla de hacerlo es simplemente copiando la salida de la consola de R y pegándolo luego en un procesador de texto. La desventaja de este procedimiento es que se genera una distorsión en el formato y muy probablemente requiera edición posterior, esto es especialmente cierto cuando se trata de matrices multidimensionales que combinan texto, factores y valores numéricos.

En este caso lo más conveniente puede ser salvar esta matriz con la extensión correspondiente, para luego abrirla con un editor apropiado (por ejemplo bloc de notas o Excel) utilizando el comando write():

En este comando pueden precisarse distintos aspectos del formato, como la extensión (table, xls, cvs) así como la localización del archivo.

Ejemplo:

```
>write.table(dat, "c:/Users/M/Desktop/dat.txt", sep="\t")
```

El archivo dat (el nombre del objeto R en este caso) se salvará en el escritorio como "dat.txt", con tabulaciones para separar columnas (sep="\t").

En el caso de no especificar ningún directorio, R lo salva automáticamente en el que está en uso por el programa, y que es aquel desde donde se leen los archivos.

Una forma más directa puede ser salvar automáticamente los resultados en un archivo de texto sin imprimirlos en la pantalla de R, utilizando el comando sink().

```
>sink("resultados.txt")#Nombre que tendrá el archivo que se guardará en el directorio de trabajo de R
>summary(lm(Vol~Caso,data=a))#Resultado del análisis (summary).
>sink()#Guardar
```

Existen diferentes paquetes de R para editar y salvar distintos tipos de datos. En el caso de Excel se puede utilizar el paquete xlxs.

```
>library(xlsx)
>write.xlsx(mydata, "c:/misdatos.xlsx")
```

Es importante tener en cuenta que aunque los archivos se salven como texto con tabulaciones o como delimitado por comas, pueden abrirse igualmente con Excel, especificando previamente el formato.

1.13.2. Exportar gráficos.

R posee paquetes e interfaces gráficos muy sofisticados, como Ploty(), ggplot() o lattice(), pero su uso no es estrictamente necesario y el paquete base de R contiene lo necesario para obtener gráficos de buena calidad. Algunos paquetes incluyen además sus propias opciones gráficas vinculadas a algunos análisis, como veremos más adelante.

Algunos de los formatos más comunes son los siguientes:

TIFF	Puede abrirse con muchos programas, pero no puede modificarse el tamaño del archivo.
JPG	Puede abrirse con muchos programas, pero no puede modificarse el tamaño del archivo.
WMF	Puede abrirse con muchos programas, el tamaño del achivo puede modificarse al pegarlo en Word, Excel o Power Point.
PDF	Puede modificarse con editores de pdf.
EPS (Poscript)	Puede abrirse con muchos programas, el tamaño del achivo puede modificarse en Photoshop, OpenOffice y LaTeX.
PNG	Puede abrirse con muchos programas pero puede modificarse el tamaño del archivo.

Alternativamente, existen dos maneras de exportar gráficos de R:

La primera y más sencilla es copiar o guardar el gráfico a partir de la pestaña Archivo/guardar como/"", o accionando directamente con el botón derecho del mouse sobre el gráfico. Opcionalmente el gráfico puede imprimirse también.

La segunda alternativa que presentamos aquí es salvarlo con un tamaño y resolución predeterminada, lo que se hace con una función de R:

```
>tiff("Migrafico.tiff", width = 4, height = 4, units = 'in', res = 300)
```

La función arma un archivo en blanco con ese nombre y lo salvará en el directorio de trabajo de R. En este caso 4x4 pulgadas (pueden ser cm u otra escala) y 300 dpi.

```
>plot(Migráfico)#Nombre del objeto
>dev.off()#El gráfico no se abre, se salva directamente.
```

1.13.3. Elementos básicos del lenguaje gráfico de R.

El argumento básico para generar un gráfico es plot(), por ejemplo:

```
>x<-rnorm(10,5) #Generemos primero 10 datos continuos, para el eje x de media 5 y distribución normal
>y<-rnorm(10,9) #Generemos primero unos datos continuos, para el eje y, peor con media 9
>Misdatos<-cbind(x,y)
#Usar plot
>Migrafico<-plot(Misdatos)
```

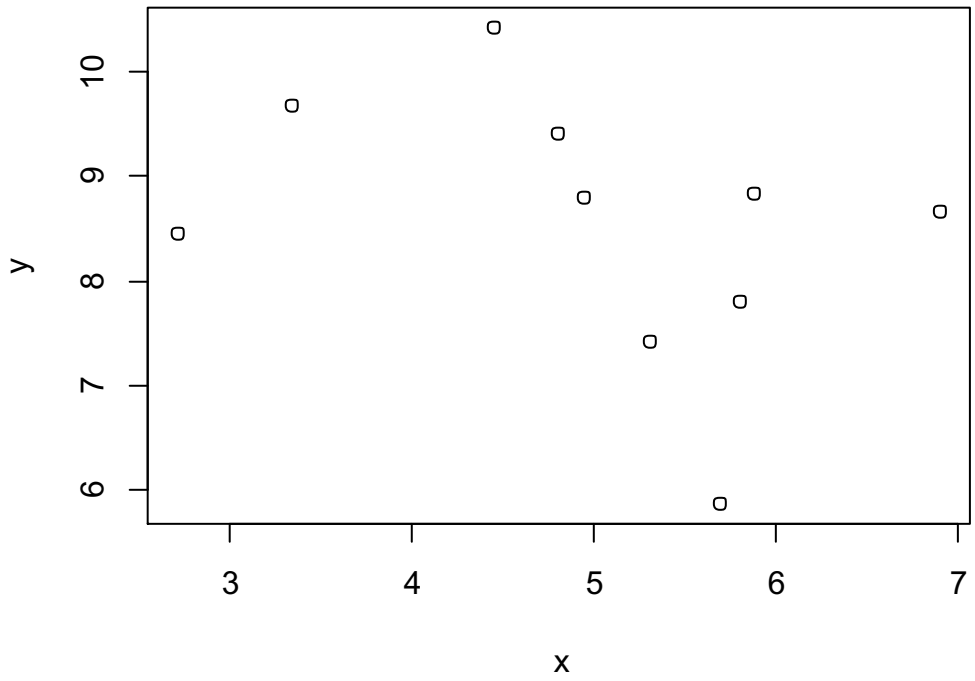
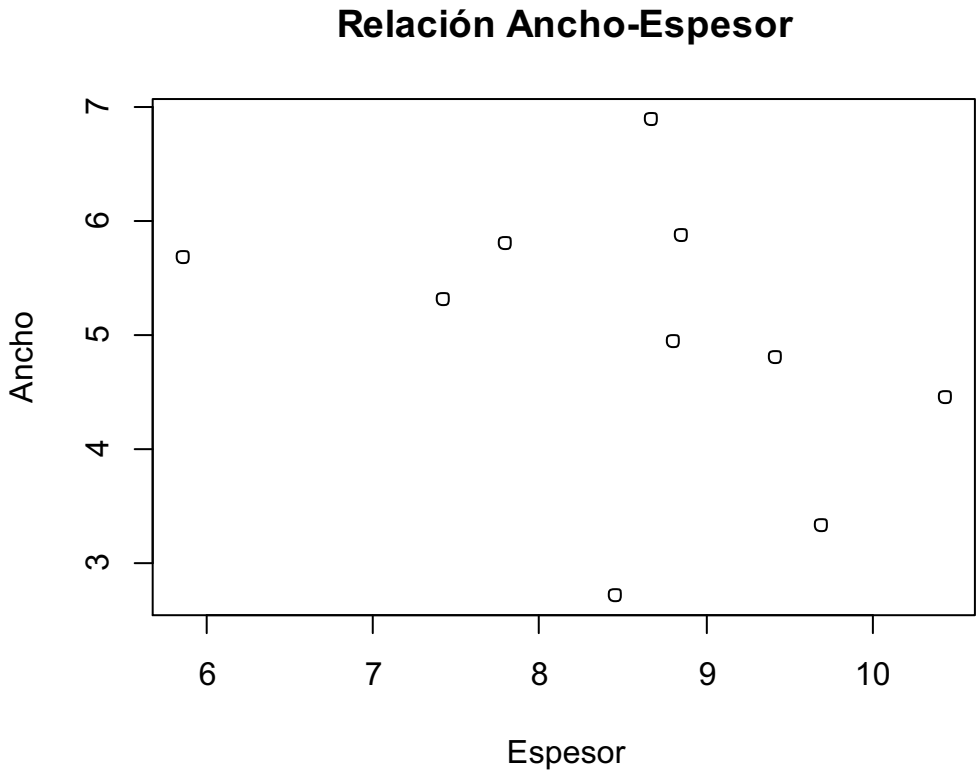


Gráfico de dispersión generado con el comando plot() por defecto.

En este caso al existir solo dos variables cuantitativas continuas, R realiza por defecto un gráfico de dispersión. También puede emplearse el símbolo "~" (se distribuye como), lo que indica que una de las variables es una función de la otra.

Asimismo, podemos incorporar o editar leyenda de los gráficos:

```
>Migrafico<-plot(x~y, data=Misdatos, xlab="Espesor", ylab="Ancho", main="Relación Ancho-Espesor")
```



Como se observa, nombres o leyendas son introducidos entre comillas y las sentencias siempre deben estar entre paréntesis que indican su inicio y finalización.

Muchas otras funciones de formato gráfico son posibles. En principio, letras, líneas, símbolos y aspecto general o

proporción de los distintos elementos puede ser editados. Si bien algunos paquetes tienen comandos especiales como lattice() o ggplot(), muchas funciones son comunes a la mayoría de ellos como:

main=""	Nombre del gráfico, siempre entre paréntesis.
xlab="",ylab=""	Nombre de loe ejes x e y, siempre entre paréntesis. Si no se especifica, R por defecto los nombra como las variables utilizadas.
type=""	El tipo de gráfico a representar: puntos, type="p" líneas, type="l". Ver más opciones tipeando ?plot
col=""	Color del gráfico: rojo, col="red", verde col="green" por defecto el color es negro. Muchos colores son posibles, incluso con distintos niveles de transparencia.
pch=	Tipo y opciones símbolos, por ejemplo: círculo sólido pch= 19, triángulo pch=5. Por defecto R representa círculos. Ver más opciones tipeando ?pch
lty=	Tipo de línea: sólido lty=1 (aparece por defecto), punteado, lty=3. Ver más opciones tipeando ?par
lwd=	Ancho de línea. Por defecto es 1.

1.14. Acerca del empleo de scripts.

R funciona a través de comandos, por ello, cuando se realizan varias operaciones a la vez, es conveniente guardar esos comandos o scripts en archivos aparte. Posteriormente éstos pueden ser pegados en la consola de R para así ejecutar

una serie de operaciones de forma simultánea. Estos comandos aseguran el poder repetir los pasos en que los análisis fueron realizados, lo que asegura el replicarlos de forma exacta las veces que sea necesario. Por otro lado, esto permite compartir información y explicitar los pasos en que se realizaron los análisis.

Es posible acceder a comandos de distinta complejidad publicados en libros o en internet en páginas específicas.

Por ejemplo, el sitio https://www.r-bloggers.com/ publica casos de estudio en R, así como archivos y bases de datos casi a diario.

Estos comandos pueden guardarse como archivos de texto, por ejemplo en un bloc de notas o como scripts de R. El programa permite escribir, salvar y/o abrir scripts a partir de la ventana Archivo. En éstos es conveniente listar previamente los paquetes utilizados y que al pegar el script en la consola de R se ejecutarán secuencialmente.

1.15. Ayuda en R.

?lm help(lm) library(help="archdata")	Brinda información sobre la función o un paquete, en este caso el comando básico de la regresión ordinaria (lm) y el paquete archdata (si es que no fue cargado previamente, de lo contrario, vasta con tipear ?archdata).
??regression help.search("regression")	Busca información sobre el término regresión en todos los paquetes disponibles.

Segunda parte

2. Estadística descriptiva.

2.1. Introducción.

La estadística descriptiva, junto con la inferencial conforman dos de los tres elementos básicos de los análisis cuantitativos. Un tercer elemento es el diseño de investigación, que no trataremos aquí. Tenemos que suponer entonces que los datos son fruto de esa primera etapa en donde se determina el universo de interés y lo que va a constituir el dato arqueológico.

Una vez obtenidos los datos, se deben explorar con el fin de tener un primer pantallazo de su naturaleza, extensión y propiedades comunes. Esta exploración es fundamental porque permite determinar en una primera instancia, si nuestros datos son acordes al problema que está investigando. Por ejemplo, si existen casos anómalos (por errores de registro o tipeo) o cuál es el mejor procedimiento estadístico para obtener la mayor información posible de ellos.

2.2. Exploración inicial de la matriz de datos. Técnicas univariadas.

2.2.1. Medidas de tendencia central y dispersión de variables cuantitativas.

Con el fin de comprender como pueden explorarse los datos a través de R utilizaremos en este acápite el paquete

archdata() y los datos de puntas de proyectil de sitios de Estados Unidos (DartPoints), (ver página 9).

Lo primero que debemos hacer es cargar el paquete y los datos correspondientes.

```
>library(archdata)
>data(DartPoints)
>head(DartPoints) #Podemos dar un primer vistazo a los datos viendo una parte de la matriz con el comando head().
```

```
  Name Catalog     TARL  Quad Length Width Thickness B.Width
1 Darl 41-0322 41CV0536 26/59   42.8  15.8       5.8    11.3
2 Darl 35-2946 41CV0235 21/63   40.5  17.4       5.8      NA
3 Darl 35-2921 41CV0132 20/63   37.5  16.3       6.1    12.1
4 Darl 36-3487 41CV0594 10/54   40.3  16.1       6.3    13.5
5 Darl 36-3321 41CV1023 12/58   30.6  17.1       4.0    12.6
6 Darl 35-2959 41CV0235 21/63   41.8  16.8       4.1    12.7
  J.Width H.Length Weight Blade.Sh Base.Sh Should.Sh
1    10.6     11.6    3.6        S       I         S
2    13.7     12.9    4.5        S       I         S
3    11.3      8.2    3.6        S       I         S
4    11.7      8.3    4.0        S       I         S
5    11.2      8.9    2.3        S       I         S
6    11.5     11.0    3.0        S       E         I
  Should.Or Haft.Sh Haft.Or
1         T       S       E
2         T       S       E
3         T       S       E
4         T       S       E
5         T       S       E
6         T       I       C
```

Tal como se mencionó, el comando head() sin especificar un número determinado de filas, muestra por defecto las primeras 6. Vemos que la matriz tiene tanto el nombre de las clases (Name), como variables métricas y otras categóricas, codificadas como letras. Podemos explorar en mayor detalle estos datos con el comando summary().

```
>summary(DartPoints)
         Name      Catalog             TARL
 Darl      :28   Length:91          Length:91
 Ensor     :10   Class :character   Class :character
 Pedernales:32   Mode  :character   Mode  :character
 Travis    :11
 Wells     :10

     Quad               Length           Width
 Length:91          Min.   : 30.60   Min.   :14.50
 Class :character   1st Qu.: 40.85   1st Qu.:18.55
 Mode  :character   Median : 47.10   Median :21.10
                    Mean   : 49.33   Mean   :22.08
                    3rd Qu.: 55.80   3rd Qu.:25.15
                    Max.   :109.50   Max.   :49.30

   Thickness         B.Width         J.Width
 Min.   : 4.000   Min.   : 7.10   Min.   :10.60
 1st Qu.: 6.250   1st Qu.:11.70   1st Qu.:13.12
 Median : 7.200   Median :13.60   Median :15.55
 Mean   : 7.271   Mean   :13.75   Mean   :15.40
 3rd Qu.: 8.250   3rd Qu.:15.50   3rd Qu.:17.07
 Max.   :10.700   Max.   :21.20   Max.   :21.20
                  NA's   :22      NA's   :1
    H.Length         Weight       Blade.Sh  Base.Sh
 Min.   : 5.80   Min.   : 2.300   E   :42   E   : 6
 1st Qu.:10.50   1st Qu.: 4.550   I   : 4   I   :53
 Median :12.50   Median : 6.800   R   : 3   R   : 4
 Mean   :13.41   Mean   : 7.643   S   :40   S   :26
 3rd Qu.:16.30   3rd Qu.:10.050   NA's: 2   NA's: 2
 Max.   :23.30   Max.   :28.800

 Should.Sh Should.Or Haft.Sh   Haft.Or
 E   : 3   B   : 3   A   : 2   C   : 8
 I   :37   H   :11   E   : 9   E   :32
 S   :46   T   :72   I   :22   P   :27
 X   : 3   X   : 3   R   : 1   T   :17
 NA's: 2   NA's: 2   S   :55   V   : 5
                     NA's: 2   NA's: 2
```

También podemos obtener información sobre la naturaleza de los datos con el comando str() y help().

Al analizar cada variable observamos que debe ser una matriz mixta, o data.frame, podemos chequearlo:

```
>is.data.frame(DartPoints)
[1] TRUE
```

Veamos en detalle la información que se obtiene sobre las variables cuantitativas y categóricas con esta función, por simplicidad, podemos centrarlos en alguna de las variables más comúnmente estudiadas en las puntas de proyectil, como el largo (length):

```
>summary(DartPoints$Length)
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
  30.60   40.85   47.10   49.33   55.80  109.50
```

La salida muestra información muy básica sobre el valor mínimo, el primer cuartil, la mediana, la media, el tercer cuartil y el valor máximo. Los cinco números de resumen aportan la información básica sobre la forma de la distribución, centro y dispersión, más la media.

Veamos ahora una variable categórica, como las que decriben el enmange (haft)

```
>summary(DartPoints$Haft.Sh)
   A    E    I    R    S NA's
   2    9   22    1   55    2
```

Observamos que se trata de un factor con cinco niveles (A,E,I,R,S) e incluye además dos datos faltantes (NA). Vimos

que estos casos pueden eliminarse de nuestra matriz, si es necesario mediante el comando na.omit (), (ver página 37).

Seleccionemos un conjunto más amplio de variables cuantitativas sin los datos faltantes, para obtener información más detallada, podemos elegir Largo, Ancho y Espesor, que podemos separar de la matriz original y renombrarlos, dentro de un nuevo objeto que llamaremos "Tamaño".

```
>Tamaño<-data.frame (DartPoints$Length, DartPoints$Width, DartPoints$Thickness)
>names(Tamaño)<-c("Largo", "Ancho", "Espesor")
>head(Tamaño)
  Largo Ancho Espesor
1  42.8  15.8     5.8
2  40.5  17.4     5.8
3  37.5  16.3     6.1
4  40.3  16.1     6.3
5  30.6  17.1     4.0
6  41.8  16.8     4.1
```

Para mayor comodidad se puede utilizar la función attach() al utilizar sólo este conjunto de datos, lo que nos permite nombrarlos directamente, luego al terminar utilizamos detach().

```
>attach(Tamaño)
```

Ahora obtengamos información más detallada sobre la dispersión de los valores, incluyendo intervalos de confianza para los mismos.

Para ello emplearemos el paquete pastecs()

```
>library(pastecs)#se necesita tener instalado el paquete boot
```

Utilicemos la función stat.desc():

```
>stat.desc(Tamaño)#Por defecto calculará un intervalo del 95%
>stat.desc(Tamaño,norm=F)#Si norm=T automáticamente calcula
```

estadísticos para controlar este supuesto en los datos como la forma y dispersión (skewness y kurtosis) y el test de Shapiro-Wilk y su probabilidad, que discutiremos más adelante.

```
>stat.desc(Tamaño)
```

	Largo	Ancho	Espesor
nbr.val	91.0000000	91.0000000	91.0000000
nbr.null	0.0000000	0.0000000	0.0000000
nbr.na	0.0000000	0.0000000	0.0000000
min	30.6000000	14.5000000	4.0000000
max	109.5000000	49.3000000	10.7000000
range	78.9000000	34.8000000	6.7000000
sum	4489.1000000	2009.0000000	661.7000000
median	47.1000000	21.1000000	7.2000000
mean	49.3307692	22.0769231	7.2714286
SE.mean	1.3351157	0.5405308	0.1605263
CI.mean.0.95	2.6524405	1.0738588	0.3189136
var	162.2105983	26.5877949	2.3449524
std.dev	12.7361925	5.1563354	1.5313237
coef.var	0.2581795	0.2335622	0.2105946

Veamos ahora que información obtuvimos: primero tenemos el número de casos (91) luego cuántos de ellos poseen valores nulos o iguales a 0 (0) y el número de casos faltantes (0). Además de las cinco medidas resumen, la media y el error estándar de la media poblacional (SE.mean), tenemos medidas de dispersión como el rango (range), la varianza (var), el desvío

estándar (std.dev) y el coeficiente de variación (coef.var). El coeficiente de variación resulta de dividir el desvío estándar por la media y constituye una medida de la variación no sujeta a la escala de la variable, como el desvío estándar. Esto puede ser muy útil si se pretende comparar la variación de diferentes variables. Como dijimos, esta función computa también, el intervalo de confianza sobre la media (CI.mean.0.95).

Las diferencias entre la media y la mediana, sugieren, al menos en este caso, que el largo y el ancho poseen distribuciones con mayor variación (ver también el Coeficiente de Variación mayor) y que probablemente, no se distribuyen normalmente, de lo contrario media y mediana coincidirían. Por otro lado, el espesor parece distribuirse de manera más homogénea.

Veamos como podemos complementar la descripción numérica de estas tres variables cuantitativas con el análisis gráfico.

2.2.2. Análisis descriptivo gráfico de variables cuantitativas.

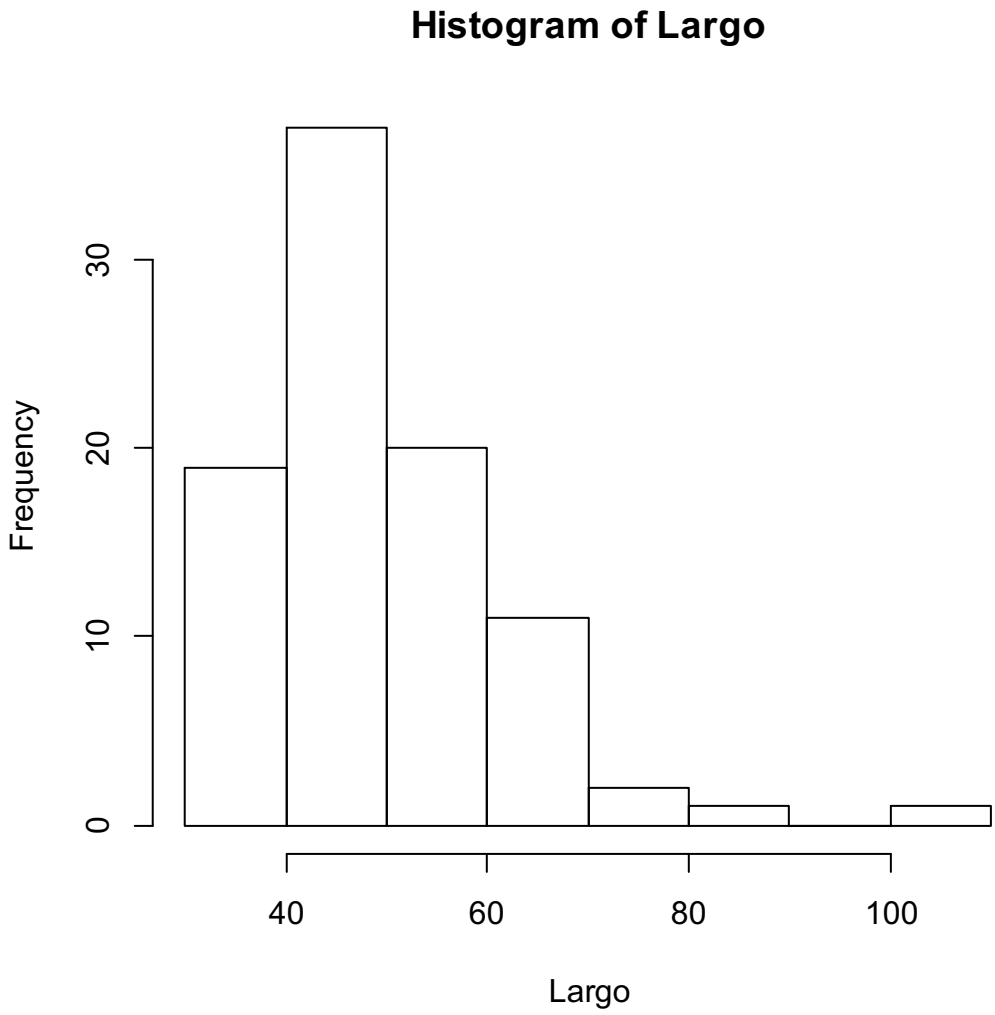
Los métodos de descripción gráfica son el complemento de los numéricos, ya que permiten obtener información sobre la extensión y forma de la distribución, así como de la existencia de datos anómalos o influyentes. Reconocer estos casos es importante, en especial si luego se planea realizar test de hipótesis, en especial los paramétricos (basados en la media).

Histograma:

Uno de los métodos gráficos más frecuentes para datos univariados son los histogramas y los gráficos de caja o box plot. Ambos pueden calcularse en R con los comandos hist() y boxplot().

A partir de nuestras variables cuantitativas de tamaño, realicemos primeramente el histograma para el largo:

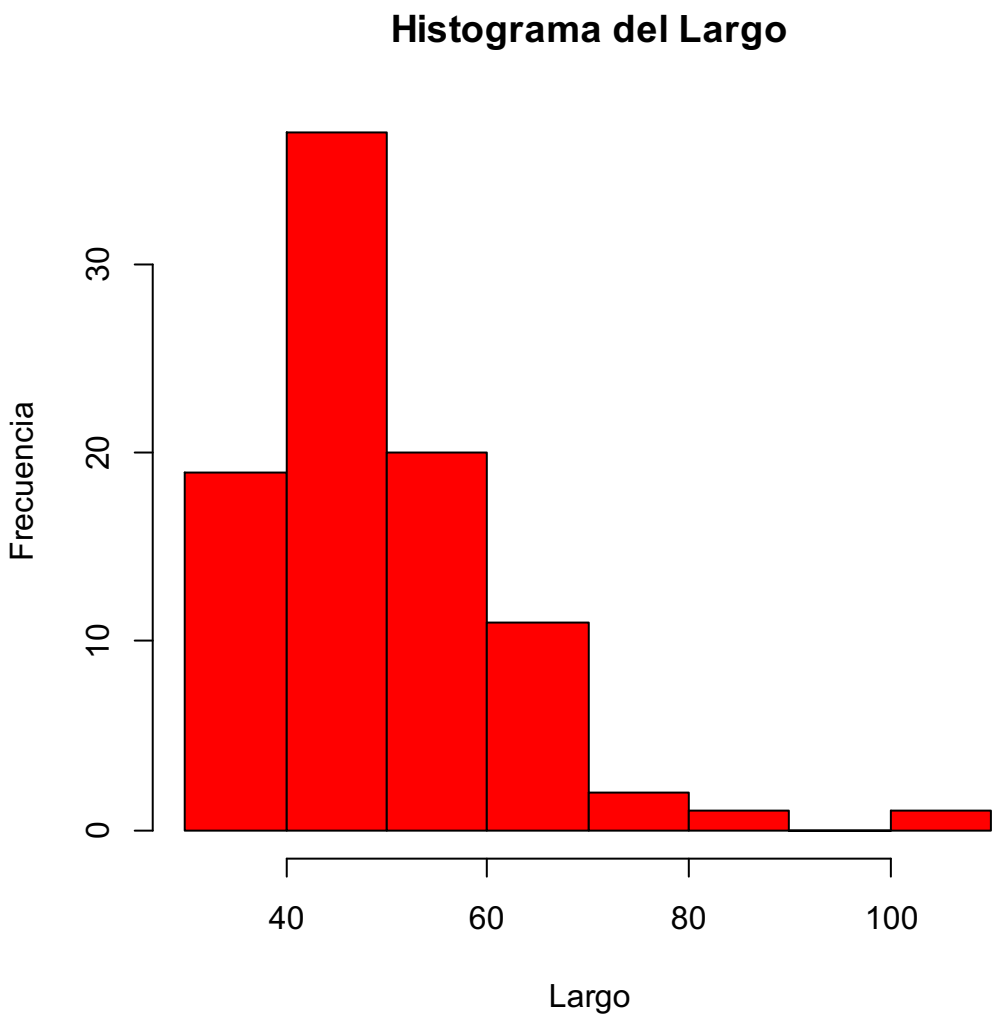
```
>hist(Largo)
```



Histograma del largo tal como R lo genera por defecto.

Aquí se ve claramente que el largo posee una distribución asimétrica positiva, con algunas puntas muy largas. Mejoremos el histograma añadiendo color:

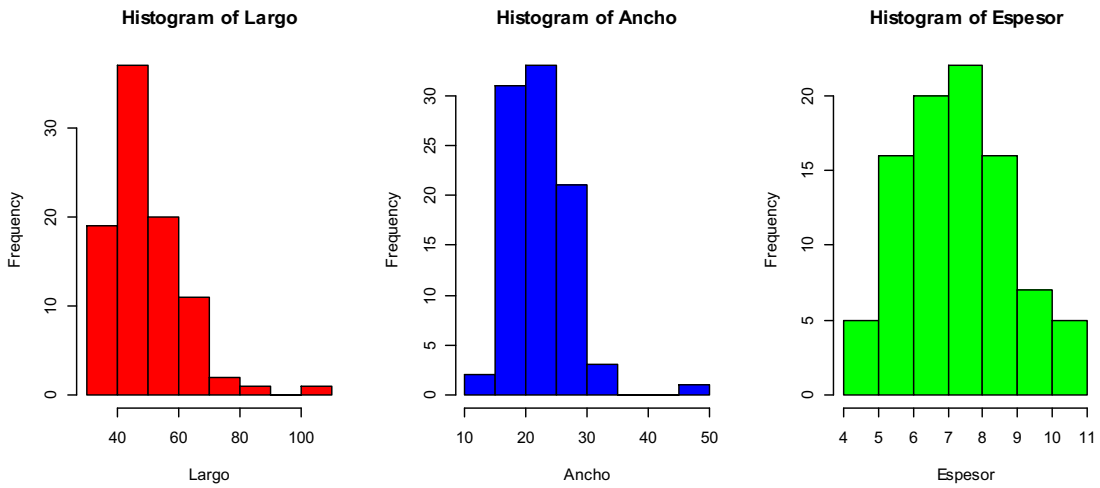
```
>hist(Largo, col="red", ylab="Frecuencia", main="Histograma del Largo")
```



Histograma del largo con nombres y color editados.

Por defecto, R presenta un gráfico a la vez, pero podemos armar un archivo en blanco con el espacio necesario para representar las tres variables juntas (largo, ancho y espesor):

```
>par(mfrow=c(1,3))# Un archivo en blanco de una fila y tres columnas, una para cada figura.
>hist(Largo,col="red")
>hist(Ancho,col="blue")
>hist(Espesor,col="green")
```



Histograma para las tres variables realizadas con el comando par().

Es posible además, editar la cantidad de intervalos con el comando breaks(), por ejemplo, breaks=30 producirá 30 intervalos.

Los histogramas muestran que el espesor es la variable con una distribución más simétrica, (a diferencia de las otras dos), lo que concuerda con la descripción numérica.

Existen muchas posibilidades de histogramas, en principio, podemos acercarnos a la distribución desde otro enfoque, el de la densidad. El cálculo de la densidad en vez de la frecuencia en la distribución, utiliza funciones de estimación más complejas, de las cuales la más común es la de Kernel. La ventaja de esta función es que permite generar un perfil suavizado y continuo de la distribución de las variables.

Si bien hay distintas funciones en R, la más sencilla es density()

Gráfico de densidad:

```
>DL<-density(Largo)  #Calculemos la densidad para las tres variables
>DA<-density(Ancho)
>DE<-density(Espesor)
>par(mfrow=c(1,3))#Grafiquemos sobre el mismo esquema que antes
>plot(DL)
>plot(DA)
>plot(DE)
```

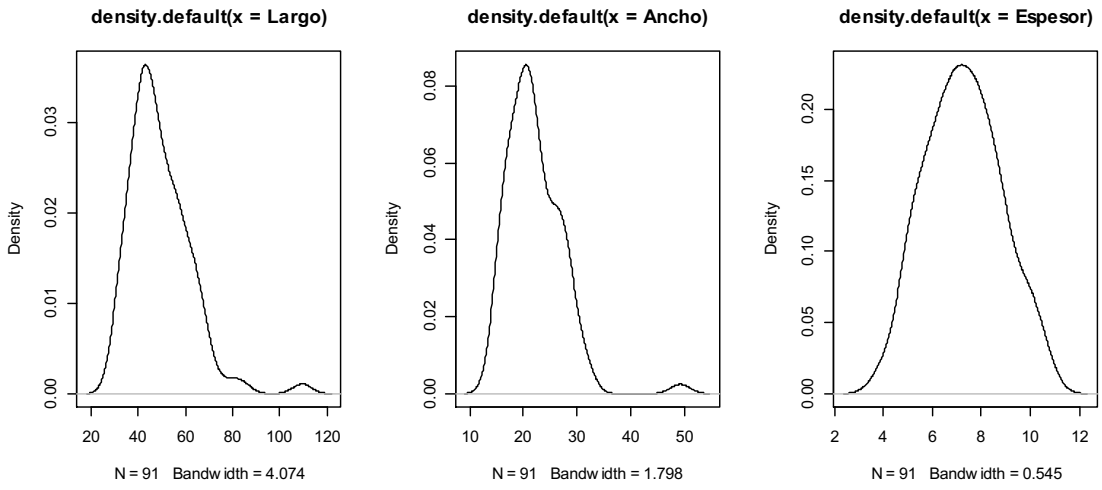


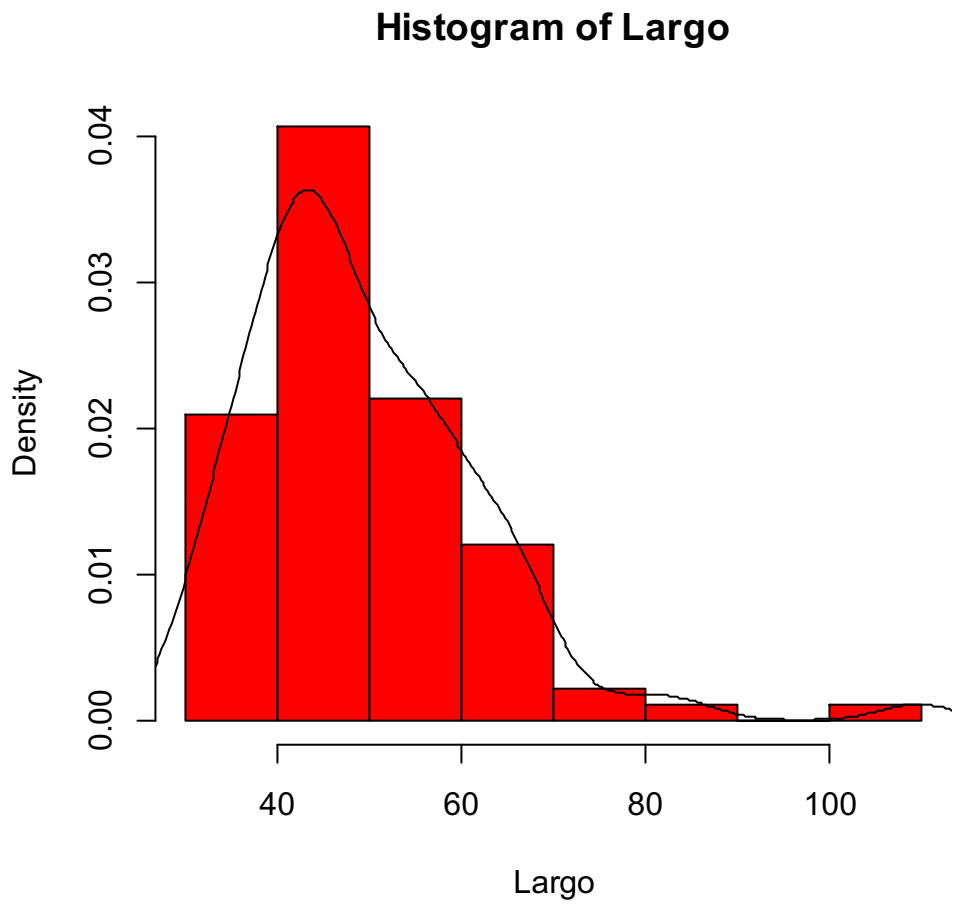
Gráfico de densidad para las tres variables de tamaño.

Vemos más claramente que largo y ancho son asimétricas, mientras que espesor se distribuye de manera más homogénea alrededor de los valores centrales.

Podemos combinar histogramas y densidad para cada variable por separado, pero para ello debemos cambiar el eje de frecuencias del primero por el de densidad del segundo:

En el caso del Largo:

```
>hist(Largo, prob=T)#histograma con probabilidad en vez de frecuencias en el eje de las y (prob=T)
>lines(DL)#La función lines() grafica el contorno estimado por la función de densidad almacenado en el objeto DL.
```



Histograma junto con distribución de densidad de la variable Largo.

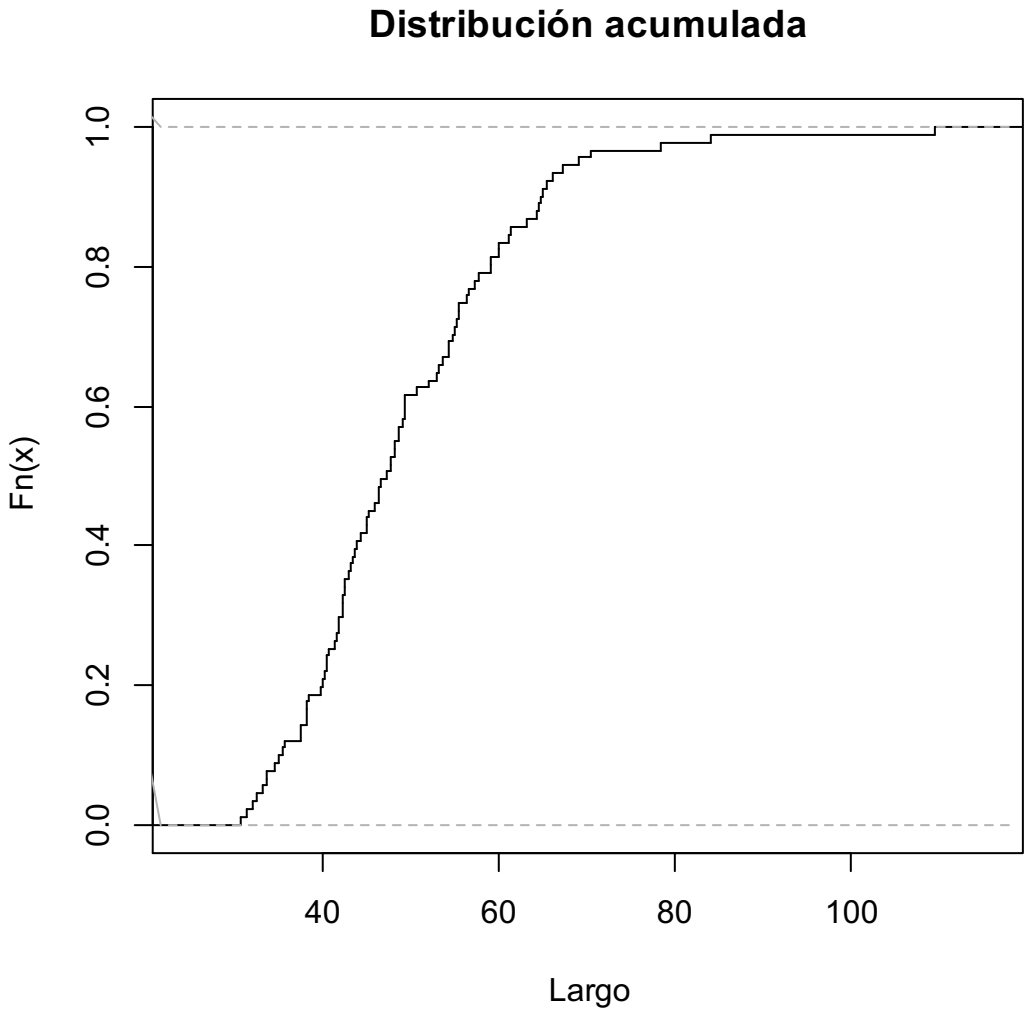
Distribuciones acumuladas empíricas:

Otra forma de representar los datos puede ser a través de distribuciones acumuladas, estas representan la relación entre la proporción creciente de casos (donde el total de los casos

alcanza el máximo de densidad) y la distribución de los mismos a lo largo del recorrido de la variable. Utilicemos la función ecdf().

```
>plot(ecdf(Largo), do.points=FALSE, verticals=TRUE, xlab="Largo", main="Distribución acumulada")#por defecto R
```

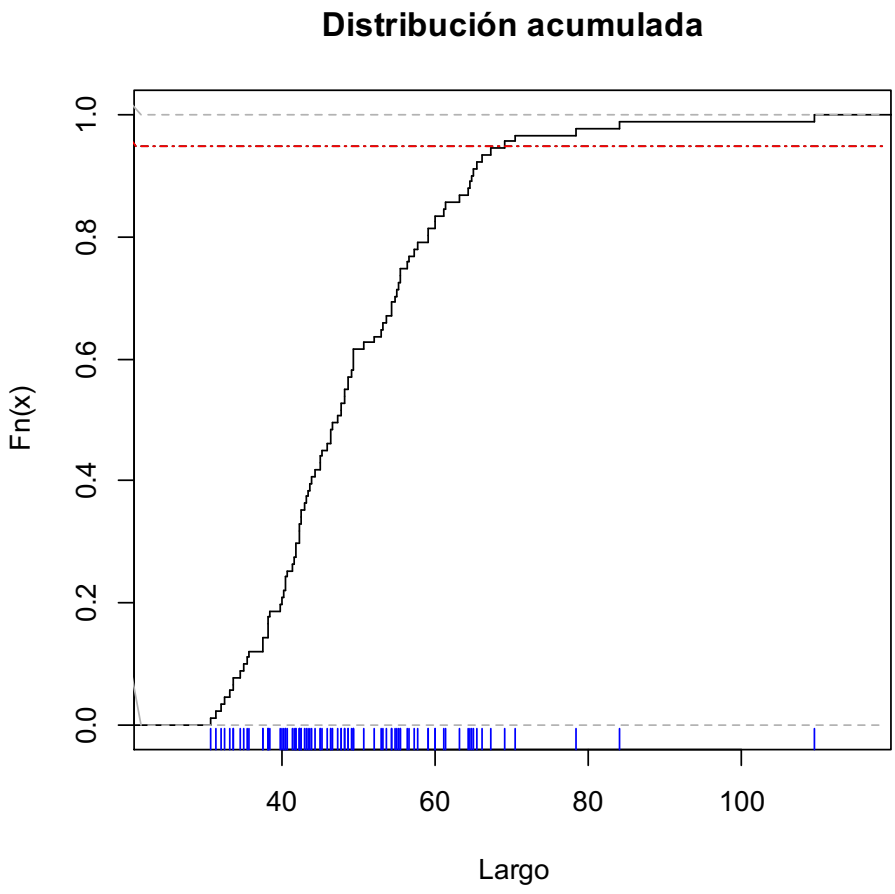
grafica los puntos también, pero en este caso nos interesa sólo la curva acumulada, ya que vamos a agregar los puntos luego. "Verticals", indica cuando o no graficar una línea continua (T) o segmentos verticales (F, por defecto).



Distribución empírica acumulada del Largo.

Ahora supongamos que queremos agregar los puntos, esto puede hacerse aplicando la función rug(), que también se puede aplicar a los gráficos de densidad o histogramas y una línea de corte, que en este caso, indique donde se alcanza el 95% de la distribución (y de esta manera, identificar mejor la frecuencia y distribución de casos con valores altos potencialmente atípicos), utilizando el comando abline().

```
>abline(0.95,0, lty=10, col="red")#queremos que se extienda paralela a 0.95.
>rug(Largo, col="blue")#Los datos de la frecuencia los extraemos directamente del objeto "Tamaño" que tenemos en la memoria, y no del análisis.
```



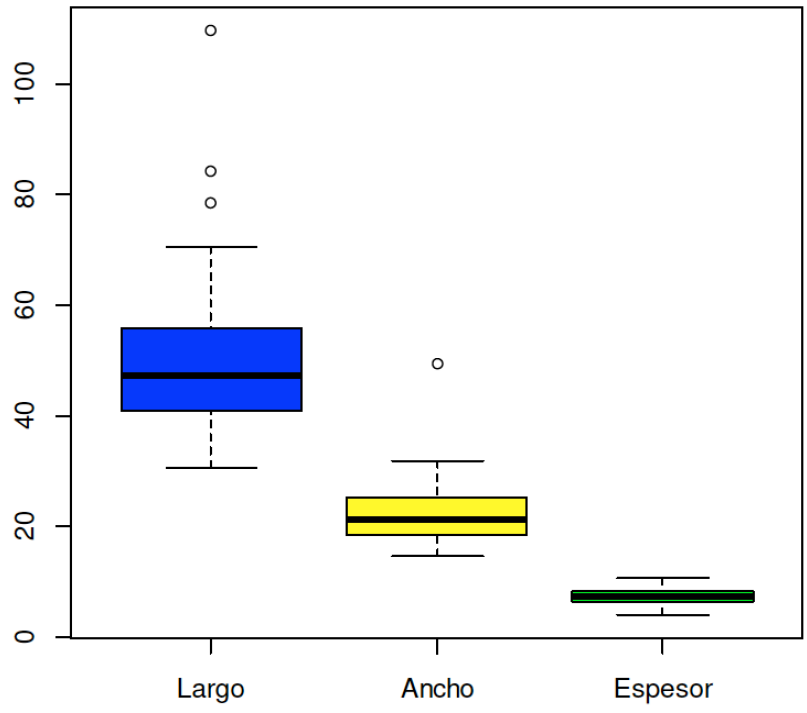
Distribución empírica acumulada del largo, donde agregamos una línea de corte (en rojo) y el número de casos en la base (azul).

El resultado nos señala que la mayor densidad de casos se encuentra entre 40 y 60 mm, mientras que pocos casos están más allá del 95%. También podríamos agregar líneas que indiquen la media, mediana o cuartiles.

Gráfico de caja o boxplot:

Otro procedimiento habitual para representar gráficamente la distribución de una variable cuantitativa es a través de los gráficos de caja o boxplot. La caja contiene el 50% central de los datos, mientras que ambos cuartiles, inferior y superior se extienden a cada lado. Valores más allá del 95% de la distribución -y que es posible considerar anómalos- pueden representarse como puntos aislados. En R éstos aparecen por defecto. La función básica es boxplot().

```
>boxplot(Tamaño, col=c("blue", "yellow", "green"))#Para elegir colores (sino son todos blancos), hay que generar una lista, en este caso azul, amarillo y verde.
```



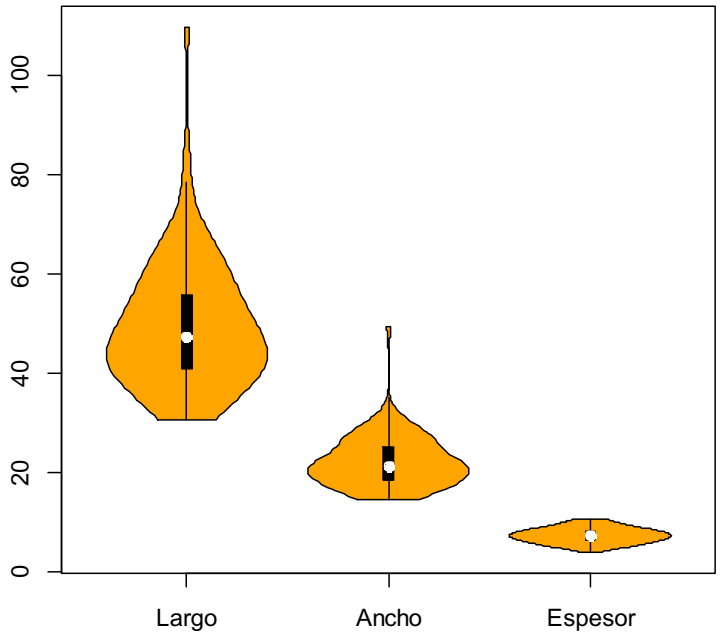
Boxplot para las tres variables de tamaño.

Tal como se observa en el gráfico, el largo es la variable más asimétrica con varios datos extremos, seguido por el ancho, mientras que el espesor no tiene ninguno. La compresión de la caja para el espesor, es en este caso, un efecto de la escala diferente de las tres variables.

Violin Plot:

Es posible también, combinar densidad y box plot en un solo gráfico denominado violin plot. Este gráfico permite analizar la forma de la distribución incorporando además la densidad de casos estimada para cada sector de la misma. Utilicemos para ello, la función vioplot() del paquete vioplot().

```
>library(vioplot)
>vioplot(Largo,Ancho, Espesor, names=c("Largo", "Ancho", "Espesor"), col="orange")#Aquí se deben especificar los nombres de las variables, de lo contrario, por defecto aparece sólo su número de orden.
```



Violin Plot para las tres variables de tamaño.

El gráfico muestra el perfil de densidad de la distribución, el centro de la caja con el 50% de los datos y un punto blanco que indica la mediana. Nuevamente, observamos que el espesor es la variable más simétrica, ya que la densidad se distribuye homogéneamente alrededor del punto central.

```
>detach(Tamaño)#Dejemos esta matriz por el momento para trabajar con otros datos.
```

2.3. Variables de nominales y datos categóricos.

En arqueología, es muy frecuente que los datos estén registrados en forma de frecuencias de determinadas categorías, "número de raspadores", "cantidad de fragmentos cerámicos de cocción oxidante", "abundancia de fragmentos óseos de un taxón determinado", etc. Este nivel de medición, denominado nominal u ordinal (en el caso de existir un determinado orden entre categorías) puede ser tratado al igual que los datos cuantitativos tanto de manera numérica como gráfica.

2.3.1. Tablas de contingencia.

Las tablas de contingencia son la manera más eficiente de presentar y analizar datos categóricos. Una tabla de contingencia es básicamente, un arreglo de filas y columnas que presenta la frecuencia o abundancia de una categoría determinada en relación a otra, donde ambas categorías pueden tener dos o más niveles. R puede construir estas tablas directamente a partir de data frames que incluyan datos categóricos. Las funciones más utilizadas en R son table(), xtabs() y crosstable().

Veamos el archivo sobre entierros del paquete archdata():

```
>data(EWBurials)
>head(EWBurials)
      Group North West   Age       Sex    Direction Looking
011       2  96.96 90.32 Young Adult Male        42     283
014       2 100.20 90.61 Young Adult Male        28     272
015       2 101.74 91.62   Old Adult Male       350     219
016a      2 101.00 90.47 Young Adult Male       335      60
018       2 101.65 90.46   Old Adult Male         3      86
020       1  95.17 90.53 Young Adult Male       142      21
      Goods
011  Present
014  Present
015  Present
016a  Absent
018  Present
020   Absent

>summary(EWBurials)
 Group      North             West
 1:12   Min.   : 83.44   Min.   : 86.35
 2:37   1st Qu.:100.03   1st Qu.: 90.53
        Median :102.83   Median : 93.34
        Mean   :101.42   Mean   : 94.92
        3rd Qu.:104.92   3rd Qu.: 97.37
        Max.   :115.80   Max.   :109.34
           Age          Sex       Direction
 Child       : 2   Female:24   Min.   :  1.0
 Adolescent  : 3   Male  :25   1st Qu.: 28.0
 Young Adult :19               Median : 54.0
 Adult       : 3               Mean   :108.9
 Middle Adult:10               3rd Qu.:144.0
 Old Adult   :12               Max.   :357.0
    Looking          Goods
 Min.   :  8.0   Absent :23
 1st Qu.: 86.0   Present:26
```

```
 Median :180.0
 Mean   :175.4
 3rd Qu.:252.0
 Max.   :356.0
```

Vemos que grupo, sexo y bienes son variables categóricas de nivel nominal que además poseen solo dos niveles, mientras que edad es ordinal (ver página 9).

Una pregunta que nos podemos hacer al ver los datos, es si la frecuencia de observaciones se distribuye homogéneamente en las distintas categorías, por ejemplo, la de bienes en relación al sexo del individuo. Para cruzar estos dos factores, se puede construir una tabla 2x2.

Probemos primeramente con la función table():

```
>Tabla1<-table(EWBurials$Sex,EWBurials$Goods)
>Tabla1
        Absent Present
  Female     13      11
  Male       10      15
```

Parece que la frecuencia de bienes (su presencia o ausencia) es relativamente similar en cada sexo. Podemos estimar los valores marginales para cada categoría de la siguiente manera:

```
>margin.table(Tabla1, 1)#1 indica que queremos los valores marginales para filas
Female   Male
    24     25

>margin.table(Tabla1, 2)#2 indica que queremos los valores marginales para las columnas
 Absent Present
     23      26
```

Los totales marginales para ambos sexos parecen ser similares, lo mismo puede decirse de la presencia-ausencia de bienes.

Con la función prop.table() podemos extraer las proporciones de cada categoría directamente de la tabla, siguiendo el mismo principio.

```
>prop.table(Tabla1)# En relación al total general
            Absent   Present
  Female 0.2653061 0.2244898
  Male   0.2040816 0.3061224

>prop.table(Tabla1,1)#En relación al total de fila
            Absent   Present
  Female 0.5416667 0.4583333
  Male   0.4000000 0.6000000

>prop.table(Tabla1,2)#En relación al total de columna
            Absent   Present
  Female 0.5652174 0.4230769
  Male   0.4347826 0.5769231
```

Probemos ahora con la función xtabs() la cual utiliza una notación de fórmula (~):

```
>Tabla2<-xtabs(~EWBurials$Sex+EWBurials$Goods)
>Tabla2
             EWBurials$Goods
EWBurials$Sex Absent Present
       Female     13      11
       Male       10      15
```

Este formato, permite además estimar directamente el test de hipótesis de independencia de distinto tipo, como veremos en el capítulo siguiente.

El paquete gmodels() también permite generar tablas y es especialmente útil para tablas de más de dos dimensiones. Emplearemos la función CrossTable() para las mismas variables:

```
>library(gmodels)
>Tabla3<-CrossTable(EWBurials$Sex,EWBurials$Goods)
>Tabla3
```

```
   Cell Contents
|-------------------------|
|                       N |
| Chi-square contribution |
|           N / Row Total |
|           N / Col Total |
|         N / Table Total |
|-------------------------|


Total Observations in Table:  49

              | EWBurials$Goods
EWBurials$Sex |    Absent |   Present | Row Total |
--------------|-----------|-----------|-----------|
       Female |        13 |        11 |        24 |
              |     0.267 |     0.236 |           |
              |     0.542 |     0.458 |     0.490 |
              |     0.565 |     0.423 |           |
              |     0.265 |     0.224 |           |
--------------|-----------|-----------|-----------|
         Male |        10 |        15 |        25 |
              |     0.256 |     0.227 |           |
```

```
             |     0.400 |     0.600 |     0.510 |
             |     0.435 |     0.577 |           |
             |     0.204 |     0.306 |           |
-------------|-----------|-----------|-----------|
 Column Total |        23 |        26 |        49 |
             |     0.469 |     0.531 |           |
-------------|-----------|-----------|-----------|
```

Esta función devuelve la proporción de las distintas categorías y sus totales tal cual se estiman en el cálculo del test de chi2, permite además estimar diversos *test* y presentar sus resultados. Asimismo, vemos que la distribución de bienes es relativamente homogénea en ambas categorías.

Es posible también generar tablas de más de dos dimensiones incluyendo la pertenencia al grupo 1 ó 2, para ello es muy útil la función ftable():

```
>Tabla4<-xtabs(~EWBurials$Sex + EWBurials$Goods + EWBurials$Group)

>ftable(Tabla4)#La función ftable permite representar todos los factores juntos:
                            EWBurials$Group  1  2
EWBurials$Sex EWBurials$Goods
Female        Absent                        5  8
              Present                       2  9
Male          Absent                        4  6
              Present                       1 14
```

2.3.2. Representación gráfica de variables categóricas.

La representación gráfica de variables categóricas, también tiene como función analizar la distribución de frecuencias, proporciones o porcentajes de las variables en cada uno de los niveles en que pueda dividirse.

Con R podemos ingresar esas frecuencias directamente o a través de las funciones de construcción de tabla como margin.table(). Una de las funciones más básicas es barplot() o gráfico de barras.

Gráfico de barras:

Sigamos con el caso anterior:

```
>par(mfrow=c(1,2))# Para graficar sexo y presencia-ausencia bienes juntos, hagamos primero un marco en blanco de una fila y dos columnas

>barplot(margin.table(Tabla1, 1))
>barplot(margin.table(Tabla1, 2))
```

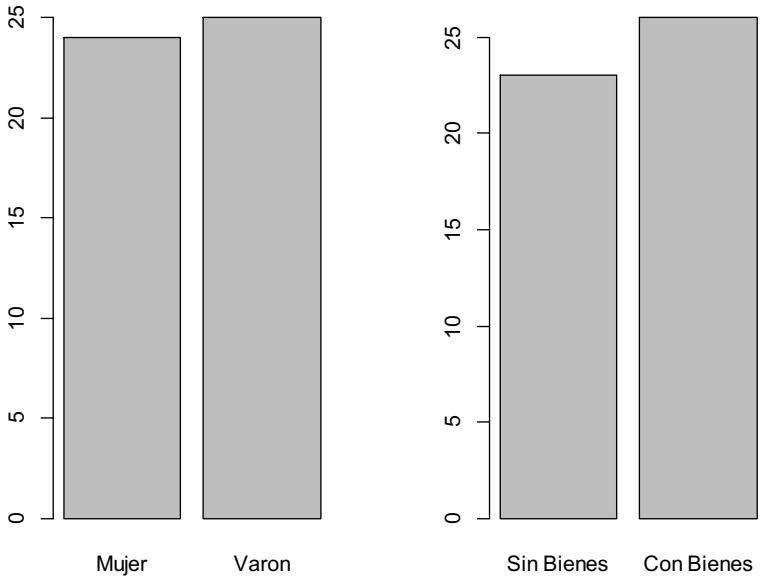


Gráfico de barras para frecuencias marginales de sexo y presencia de bienes tal como genera por defecto.

Podemos representar ambas juntas utilizando directamente el comando barplot() sobre la tabla:

```
>barplot(Tabla2)
```

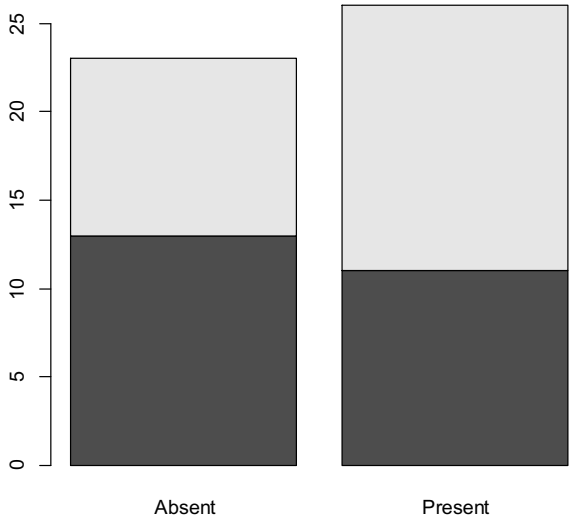


Gráfico de barras apiladas para presencia de bienes por sexo, generado por defecto con la función barplot().

Por defecto, el gráfico muestra las frecuencias apiladas de sexo por bienes, ya que la tabla se construyó tomando como primer factor el sexo (~sex+goods). Igualmente, es claro que la distribución es relativamente proporcional entre las dos categorías.

También es posible graficar automáticamente las proporciones con el mismo criterio, para ello podemos emplear directamente la función spine() del paquete vcdExtra() que está diseñado específicamente para trabajar y graficar datos categóricos.

```
>library(vcdExtra)
>spine(Tabla2)
```

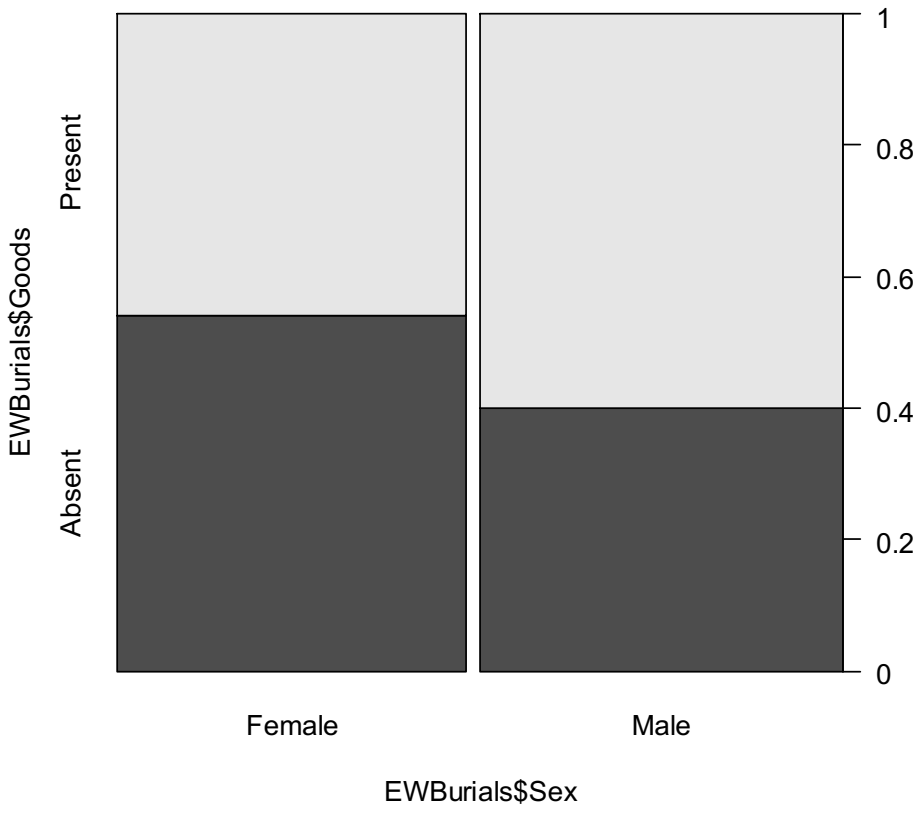


Gráfico de distribución proporcional para ambas categorías, generado por defecto con la función spine().

Este gráfico es más informativo porque nos muestra la frecuencia relativa (con los valores en el borde derecho) de cada nivel del factor, se observa más claramente que existe sólo una pequeña diferencia en la presencia de bienes en tumbas masculinas.

Gráfico de sectores:

Otra forma habitual de representar los datos categóricos es a través de gráficos de sectores donde el porcentaje relativo de cada categoría se representa como una fracción de un círculo de 360°.

Utilicemos la edad, para estudiar como se distribuye la muestra estudiada mediante la función pie().

```
>Edad<-(table(EWBurials$Age))
>Edad
Edad
       Child   Adolescent  Young Adult
           2            3           19
       Adult Middle Adult    Old Adult
           3           10           12

>pie(Edad)
```

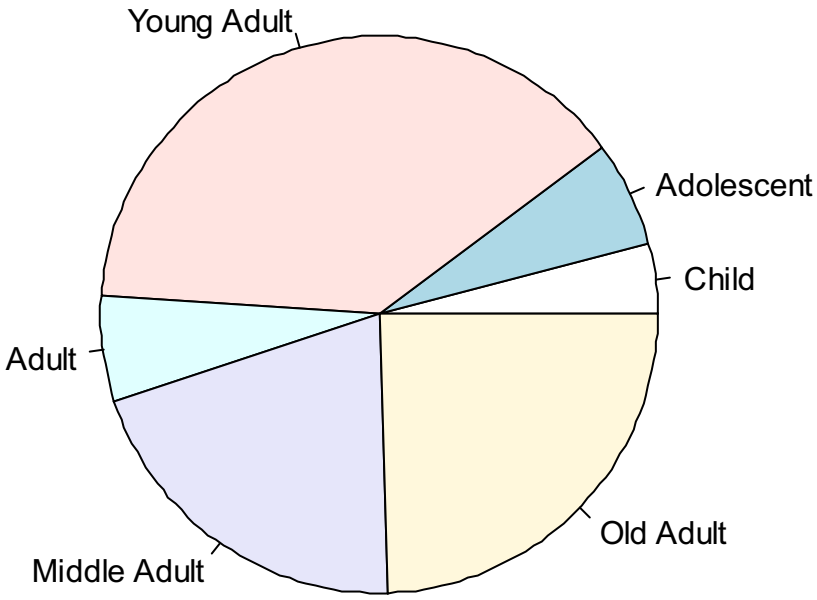


Gráfico de sectores por defecto de la función pie().

Como el sistema gráfico es flexible, es posible programar scripts sencillos para realizar cambios de aspecto o formato. Por ejemplo se pueden calcular y graficar porcentajes y luego pegarlos en el gráfico original, agregando un cuadro con la referencia.

```
>porcentaje          <-round(Edad/sum(Edad)*100)          #round,
```

automáticamente devuelve valores redondeados sin decimales y puede utilizarse en otras funciones.

```
>porcentaje# Devuelve el porcentaje para cada categoría y que
```

utilizaremos como referencia

```
       Child    Adolescent  Young Adult         Adult
           4             6           39             6
Middle Adult     Old Adult
          20            24
```

```
>porcentaje<-paste(porcentaje,"%",sep="")#La función paste,
```
permite agregar un símbolo de % al objeto porcentaje

```
>porcentaje
[1] "4%"  "6%"  "39%" "6%"  "20%" "24%"
```

```
>pie(Edad,labels = porcentaje, col=rainbow(length(Edad)))#la
```
función de color es rainbow (arcoiris) y asigna un color por cada categoría de edad.

Ahora armemos una leyenda que va a ir ubicada arriba a la derecha, con los nombres de las categorías de edad

```
>legend("topright", c("Niño","Adolescente","Joven Adulto",
"Medio Adulto", "Adulto", "Adulto Mayor"),cex=0.6,
fill=rainbow(length(Edad)))
```

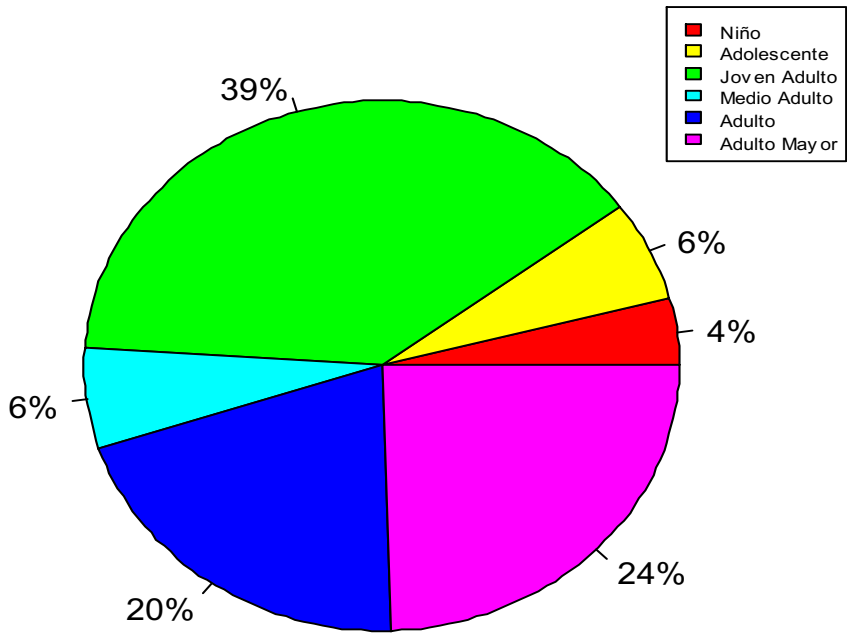


Gráfico de sectores realizado al combinar distintas funciones.

Se observa que joven adulto posee el mayor porcentaje, seguido por adulto mayor. Sin embargo, no podemos, empleando estos métodos establecer fehacientemente de que manera las diferentes variables se relacionan entre sí, o si existe influencia determinante de una variable por sobre otra. Para analizar numérica y gráficamente relaciones entre dos variables, emplearemos métodos bivariados.

2.4. Análisis descriptivo exploratorio de dos variables: técnicas bivariadas.

2.4.1.Estadística bivariada para datos cuantitativos.

En muchas ocasiones, es necesario explorar dos variables cuantitativas al mismo tiempo, para entender su distribución y la existencia de posibles asociaciones entre ellas. Asimismo, una vez establecida esta asociación, puede interesarnos evaluar la dirección, forma e intensidad de la misma.

En arqueología, es muy común observar correlaciones entre variables cuantitativas vinculadas al tamaño, peso o volumen de los artefactos líticos, cerámicos etc. Esto puede deberse a diversas causas, como el proceso de confección, requerimientos funcionales de los artefactos, o por cambios producidos a lo largo de la historia de vida de los mismos, por ejemplo, por el mantenimiento en el caso de algunos instrumentos.

Veremos primeramente como explorar cuantitativamente y visualmente estas asociaciones.

2.4.2. Análisis numérico: la correlación.

Para explorar la relación entre variables cuantitativas, uno de los procedimientos más comunes es el análisis de

correlación. Este sirve para establecer la intensidad y sentido en la asociación de dos variables entre sí. Una relación puede medirse a partir del promedio de las distancias que existen entre las distintas observaciones y su media. Con este fin, puede emplearse el coeficiente de correlación, que expresa la relación entre dos variables cuantitativas a partir de la multiplicación de la distancia de cada par estandarizado, con respecto al total de la muestra.

El más conocido y utilizado de estos índices es el coeficiente de correlación r de Pearson, del cual mencionaremos sólo alguna de sus características principales:

Los valores posibles de r van desde -1 hasta +1 indicando ambos valores, una asociación perfecta. El signo de r indica la dirección de la asociación, esto es; r<0 indica asociación negativa, r=0 indica asociación nula, r>0 indica asociación positiva. Por otro lado, el valor de r indica la intensidad de la asociación lineal.

Comencemos por volver a trabajar con los datos del archivo "Tamaño" de puntas de proyectil armando primeramente, un objeto con las variables que vamos a utilizar:

```
>Tamaño<-data.frame (DartPoints$Length, DartPoints$Width, DartPoints$Thickness)
>names(Tamaño)<-c("Largo", "Ancho", "Espesor")
```

En R hay varios paquetes que estiman distintos tipos de correlación, algunos están disponibles en la base de R, como cor() y cor.test().

La primera función devuelve, por defecto, una matriz de correlación con los valores de este índice a ambos lados de la diagonal:

```
>cor(Tamaño)
            Largo     Ancho   Espesor
Largo   1.0000000 0.7689932 0.5890989
Ancho   0.7689932 1.0000000 0.5459291
Espesor 0.5890989 0.5459291 1.0000000
```

Vemos que el largo y el ancho (independientemente aquí de las clases de puntas que componen la muestra) son las dos variables más correlacionadas entre sí (0.77), mientras que el espesor es la que presenta menor correlación lineal con las demás. En todos los casos las asociaciones son positivas y media a media-alta. Podemos comparar pares mediante la función cor.test() para obtener el nivel de significación y los intervalos de confianza de estas correlaciones.

```
>cor.test(Largo, Ancho)

        Pearson's product-moment correlation
data:  Largo and Ancho
t = 11.349, df = 89, p-value < 2.2e-16
alternative hypothesis: true correlation is not equal to 0
95 percent confidence interval:
 0.6689977 0.8416464
sample estimates:
      cor
0.7689932

>cor.test(Largo, Espesor)
        Pearson's product-moment correlation
data:  Largo and Espesor
t = 6.8776, df = 89, p-value = 8.125e-10
alternative hypothesis: true correlation is not equal to 0
95 percent confidence interval:
 0.4360573 0.7090237
```

```
sample estimates:
      cor
0.5890989

>cor.test(Ancho, Espesor)
        Pearson's product-moment correlation
data:  Ancho and Espesor
t = 6.1472, df = 89, p-value = 2.184e-08
alternative hypothesis: true correlation is not equal to 0
95 percent confidence interval:
 0.3830512 0.6758835
sample estimates:
      cor
0.5459291
```

El reporte del análisis, nos muestra el test de la *t* para determinar la significación, los grados de libertad (df) y el valor de p en notación exponencial (e). En los tres casos el resultado es significativo. Como era de esperarse, el mayor nivel de significación se observa entre el largo y el ancho. Los intervalos de confianza junto con la correlación son dados al final del reporte. Como se observa, éstos coinciden en ambas funciones.

Otras correlaciones no paramétricas pueden realizase con la misma función, como la de Spearman o de Kendall, que son adecuadas para distribuciones de este tipo, como en el caso de los datos cuantitativos discretos, los valores de abundancia como MNE o MNI en análisis faunísticos, conteos de artefactos (por ejemplo la riqueza), etc. En estos casos se debe especificar el método (method), para ambas funciones.

Recordemos que este método solo capta la porción lineal en la correlación entre dos variables por lo que es posible que exista otra forma de asociación no lineal entre ellas y que

ésta no sea detectada a menos que se represente gráficamente esta correlación. Veremos a continuación cómo realizar estos gráficos.

2.4.3. Gráficos bivariados y trivariados para variables cuantitativas.

Gráfico de dispersión:

El comando más básico es plot() que ya mencionamos en el capítulo anterior, éste realiza automáticamente distintos tipos de gráfico dependiendo de la naturaleza de las variables. Gráficos que se basan en píxeles o grillas (no veremos ninguno aquí) utilizan en general el comando image(). Existen además gráficos específicos con su notación propia dependiendo del análisis y del tipo de paquete, tal como vimos en el acápite de estadística univariada.

Sigamos con nuestro ejemplo de las puntas de proyectil. En el acápite anterior, vimos que largo y ancho poseen una correlación alta y significativa, veamos como puede representarse en un diagrama bivariado de dispersión.

```
>plot(Largo,Ancho)
```

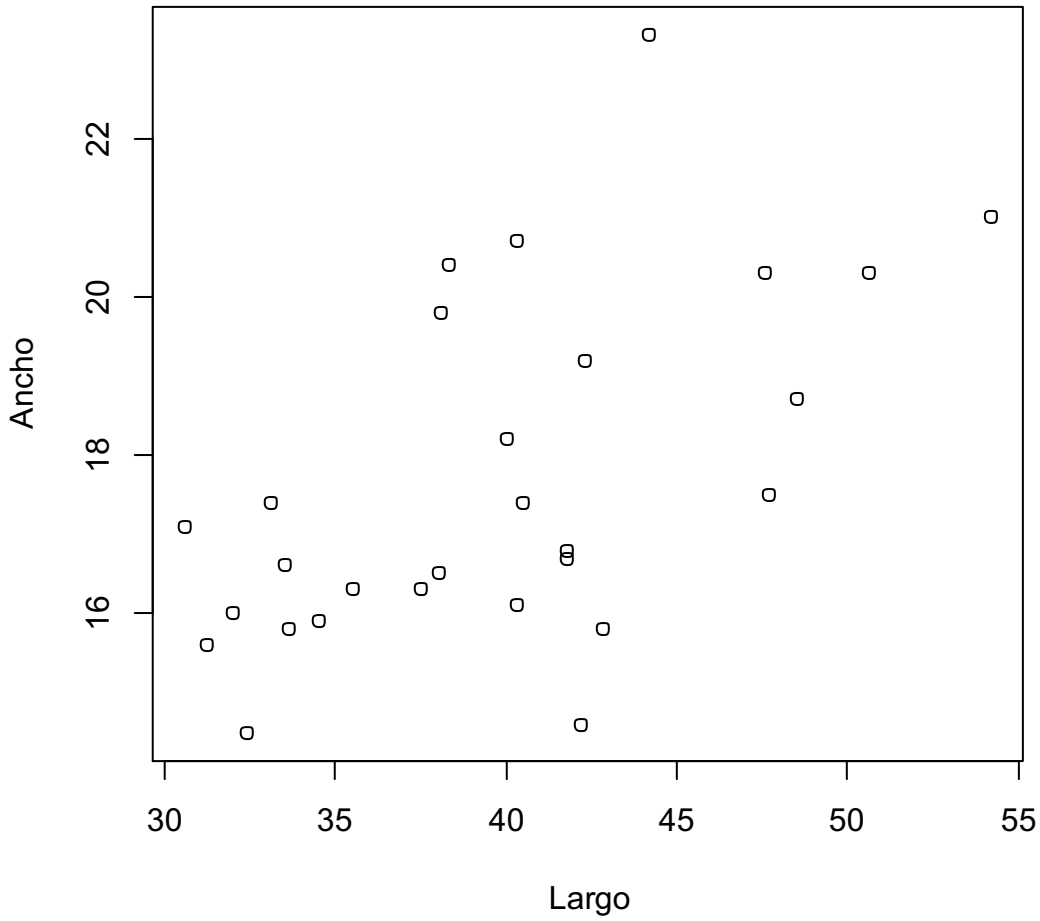


Gráfico de dispersión con el comando plot().

Por defecto, el gráfico no identifica los puntos. Pero podemos incluir más información si lo precisamos, por ejemplo, de las clases. Para ello, debemos primeramente dejar de lado aquellos casos en los que hay datos faltantes:

```
>library(car)
>PP<-na.omit(DartPoints)
>Tamaño2<-data.frame(PP$Name,PP$Length,PP$Width, PP$Thickness)
>names(Tamaño2)<-c("Clases","Largo","Ancho","Espesor")
>head(Tamaño2)
  Clases Largo Ancho Espesor
1   Darl  42.8  15.8     5.8
2   Darl  37.5  16.3     6.1
```

```
3    Darl  40.3  16.1     6.3
4    Darl  30.6  17.1     4.0
5    Darl  41.8  16.8     4.1
6    Darl  40.3  20.7     5.9
```

Ya tenemos un nuevo objeto con las clases como factor, podemos emplearlo en los gráficos:

```
>attach(Tamaño2)
>plot(Largo,Ancho,pch=unclass(Clases))# con el comando pch especificamos que queremos un símbolo por clase.
>legend("bottomright", legend=levels(Clases), pch=c(1:3))
#posteriormente determinamos posición de la leyenda en función al factor (legend=levels)
```

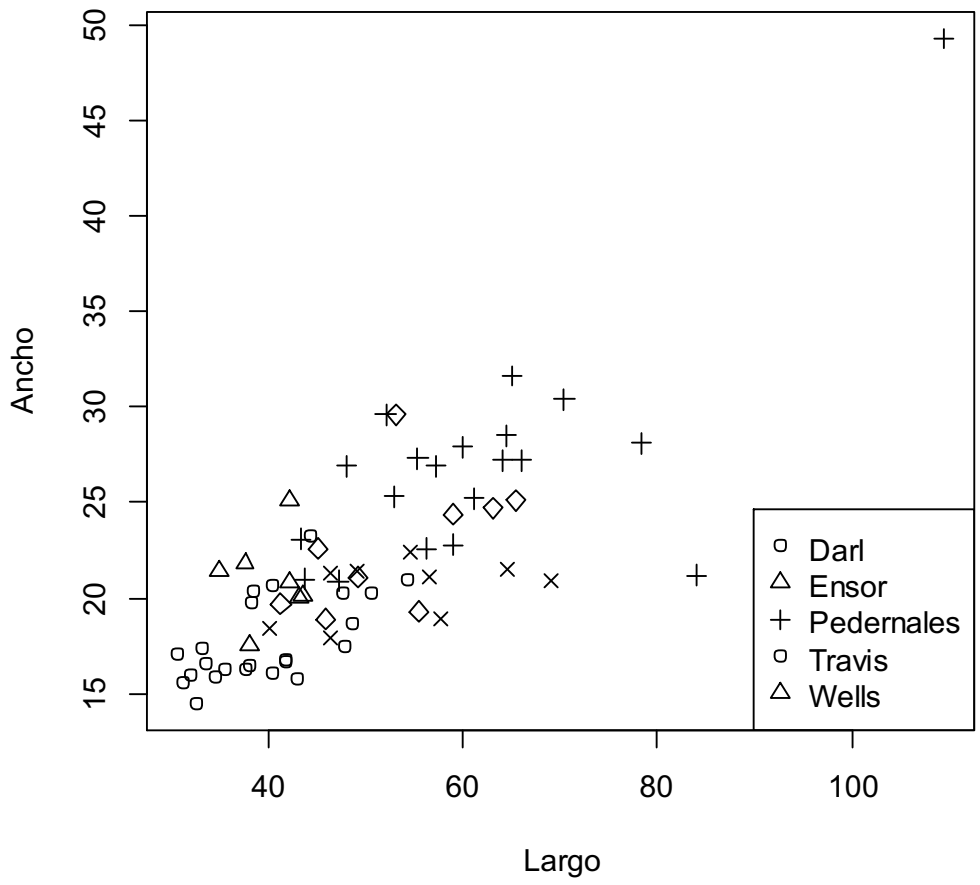


Gráfico bivariado representando los distintos conjuntos a partir del factor "Clases".

Podemos utilizar el mismo factor para agregar color:

```
>plot(Largo,Ancho,pch=unclass(Clases), col=Clases)
>legend("bottomright", legend=levels(Clases), pch=c(1:3))
```

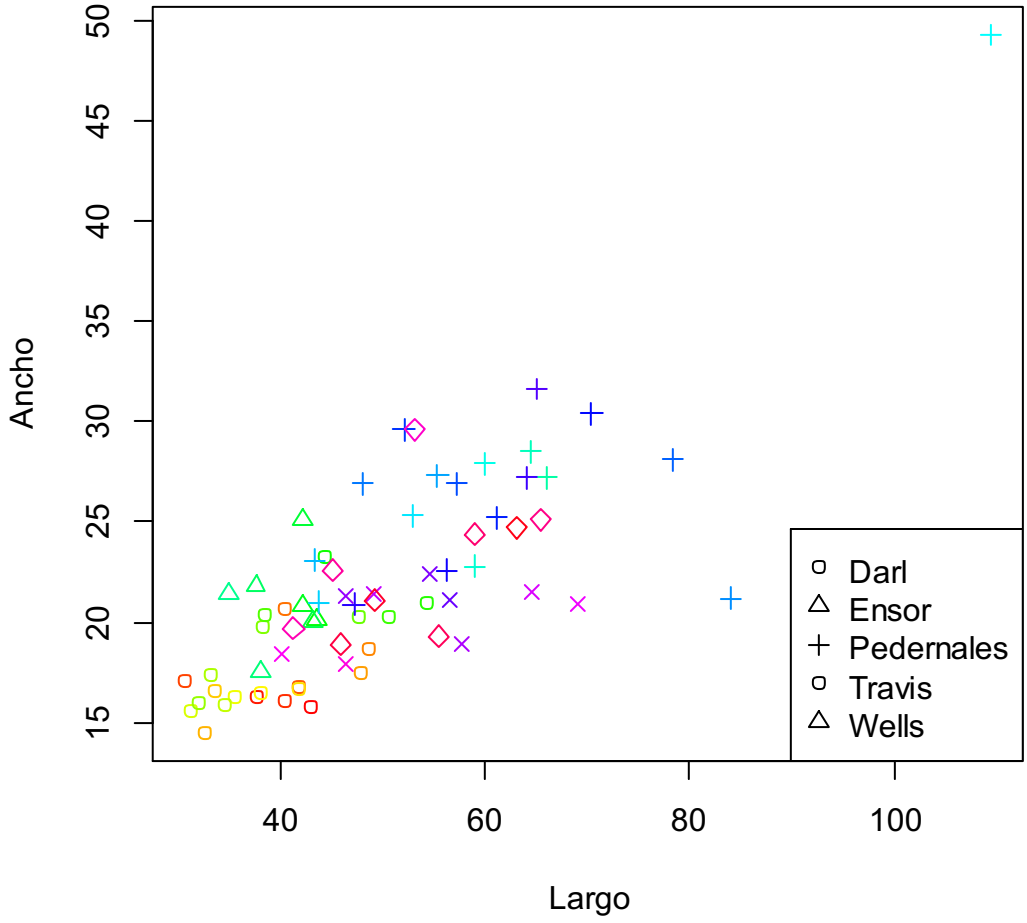


Gráfico bivariado representando los distintos conjuntos a partir del factor "Clases" e incorporando color por factor.

Otra posibilidad es especificar la inicial de las clases como símbolo:

```
>plot(Largo,Ancho,pch=as.character(Clases))# el símbolo es aquí un carácter (as.character)
```

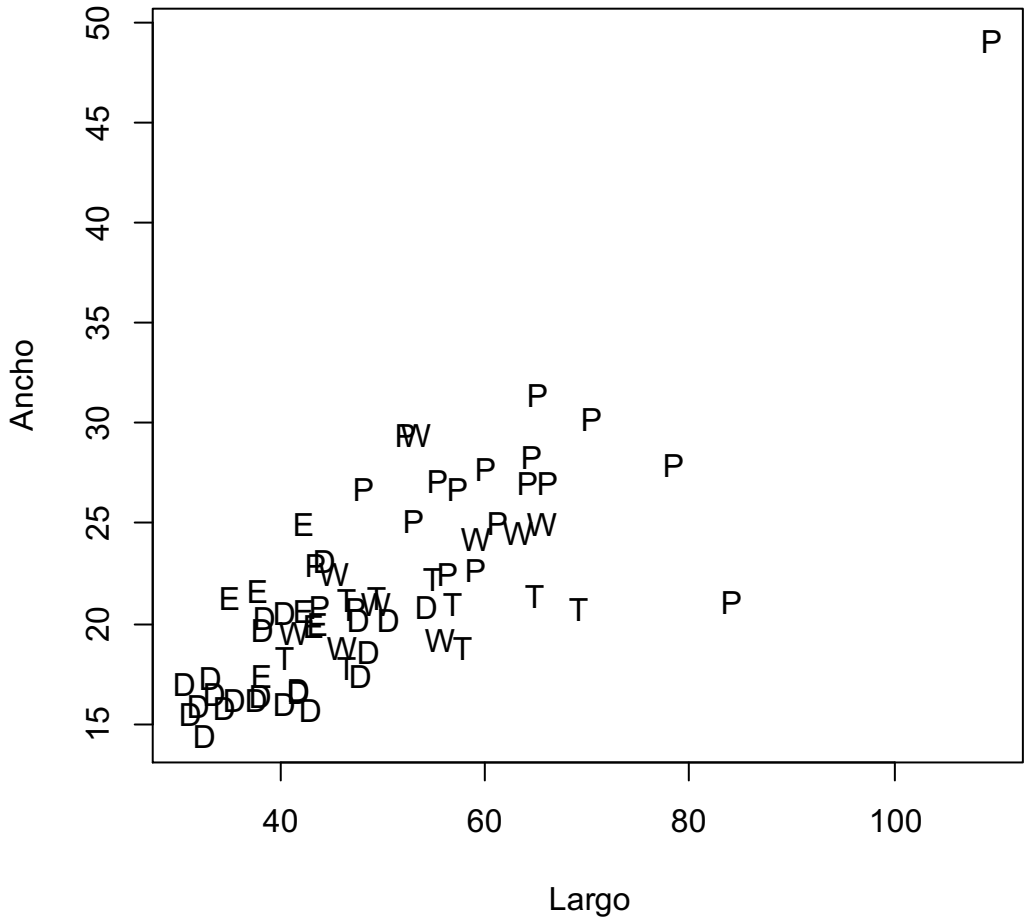


Gráfico bivariado representando los distintos conjuntos a partir del factor "Clases" utilizando la inicial de cada clase.

Por último combinemos el nombre de la clase y el color mediante el comando rainbow().

```
>plot(Largo,Ancho)
>text(Largo ,Ancho, labels=Clases, cex= 0.7, pos=1,
col=rainbow(length(Clases)))
```

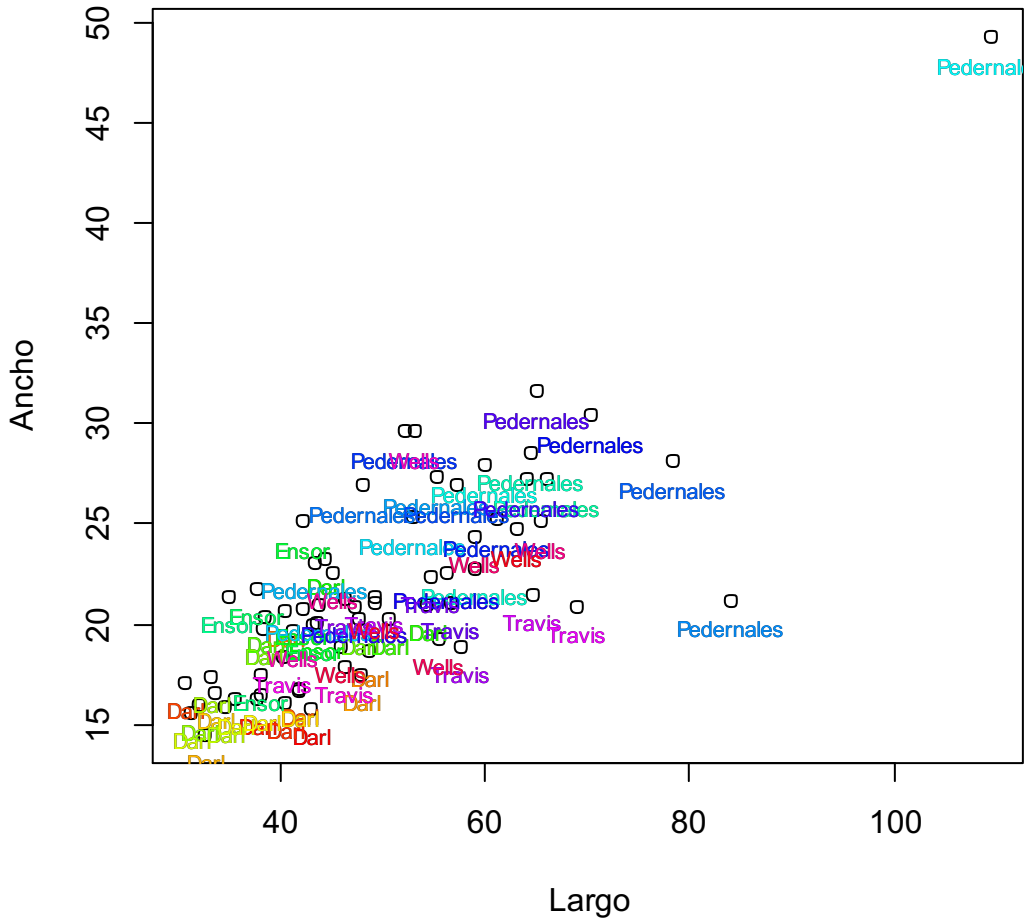


Gráfico bivariado representando los distintos conjuntos a partir del factor "Clases" utilizando nombre y color.

Gráficos de dispersión de 3 variables cuantitativas:

Comandos similares a los de la representación en dos dimensiones pueden ser empleados para representar tres variables cuantitativas al mismo tiempo. Utilicemos el paquete scatterplot3d() para representar de forma conjunta las tres variables que hemos estado analizando:

```
>library(scatterplot3d)
>scatterplot3d(Largo,Ancho,Espesor, pch=unclass(Clases))
#utilizamos la misma serie de especificaciones, aunque agregando tres variables
```

```
>legend("topleft", legend=levels(Clases),cex=0.6,pch=c(1:3))
```

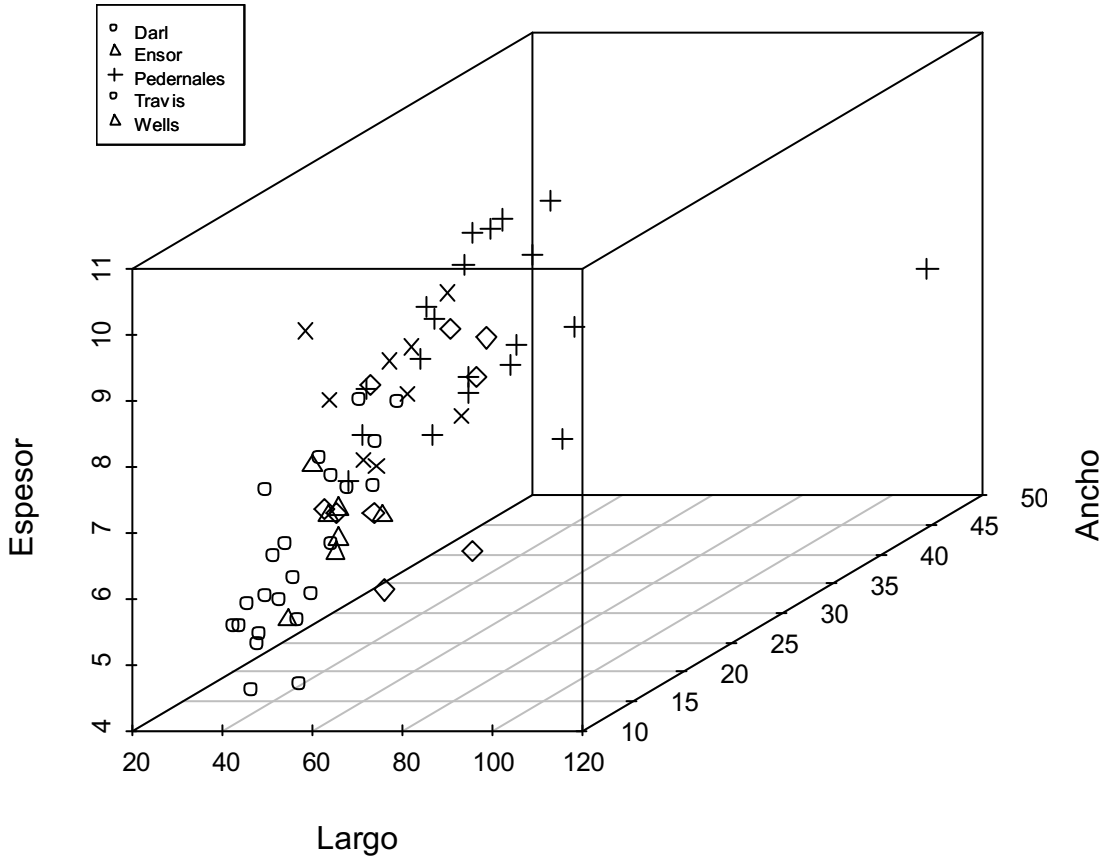


Gráfico de dispersión en tres dimensiones, diferenciando por clase.

2.5. Combinando resúmenes numéricos y gráficos.

En muchos casos en que se está trabajando con múltiples variables cuantitativas, es de gran utilidad representar conjuntamente gráficos bivariados y los valores de correlación conjuntamente, utilizando el formato de una matriz de asociación, esto nos brindará mucha información sobre el comportamiento y las posibilidades de asociación entre las distintas variables entre sí. Con este fin, emplearemos el paquete psych(), y la función pair.panels() aunque muchas otras funciones en R permiten este tipo de análisis.

```
>library(psych)
>detach(Tamaño2)
>pairs.panels(PP[,5:7],    pch=19)#Tomemos    directamente    la
```

información que necesitamos del objeto "PP" que no posee datos faltantes, especificando que queremos todas las filas de la variables 5 a la 7 (Largo, Ancho y Espesor).

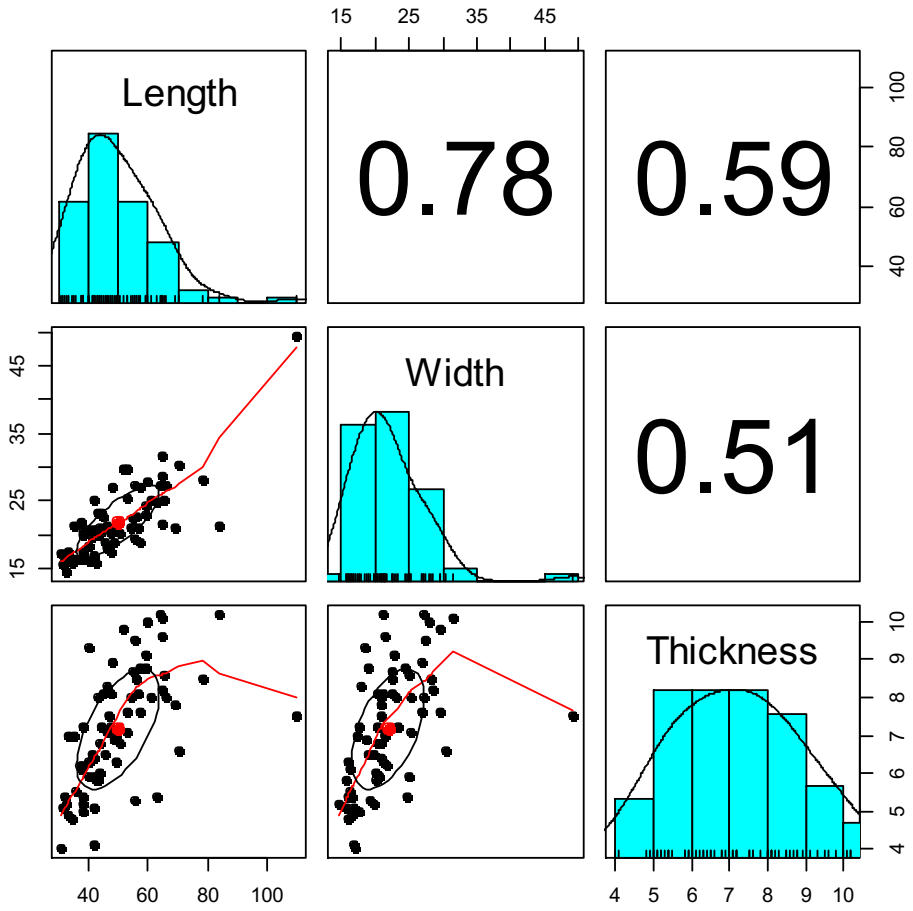


Gráfico combinando la descripción gráfica y numérica bivariada en forma de una matriz de correlación.

Este gráfico nos muestra el valor de correlación de los distintos pares de variables en el triángulo superior, un histograma de cada variable, incluyendo una curva de densidad superpuesta y la distribución de los casos (como en el comando

rug()) sobre el eje de las abscisas. En el triángulo inferior muestra la dispersión de cada par, junto con una línea suavizada de la tendencia en la relación (que puede no ser lineal), así como una elipse en torno al centroide o media de la distribución bivariada.

Muchas modelos de tablas son posibles, combinando distintos gráficos y resúmenes numéricos dentro de esta misma función.

2.6. Regresión.

En algunas oportunidades, una recta expresa adecuadamente a una variable de respuesta como función lineal de una variable explicativa. Esa recta es denominada recta de regresión. En este caso, el interés en establecer una recta de regresión es predictivo: cómo podemos explicar de la forma más sencilla posible el comportamiento de una variable desconocida a partir de otra que conocemos sin error?.

Continuando con el caso del tamaño de las puntas de proyectil que abordamos en el acápite anterior, podríamos preguntarnos si existe vinculación entre alguna variable morfológica y el tamaño general de la punta, representado por el peso. Primeramente, podemos construir una variable que resulte de la combinación del largo, ancho y espesor y que nos informe de la morfología general de la punta de proyectil. Una variable útil es la robustez, resulta de la multiplicación de largo por el ancho y su división posterior por el espesor.

Esta variable es un índice que nos informa, al relacionarla con el peso, si el tamaño general puede explicar la morfología (por ejemplo, por cambios en la historia de vida del artefacto, como la reactivación o por elecciones de diseño

vinculadas a la performance de los instrumentos o sistemas técnicos de los que forman parte).

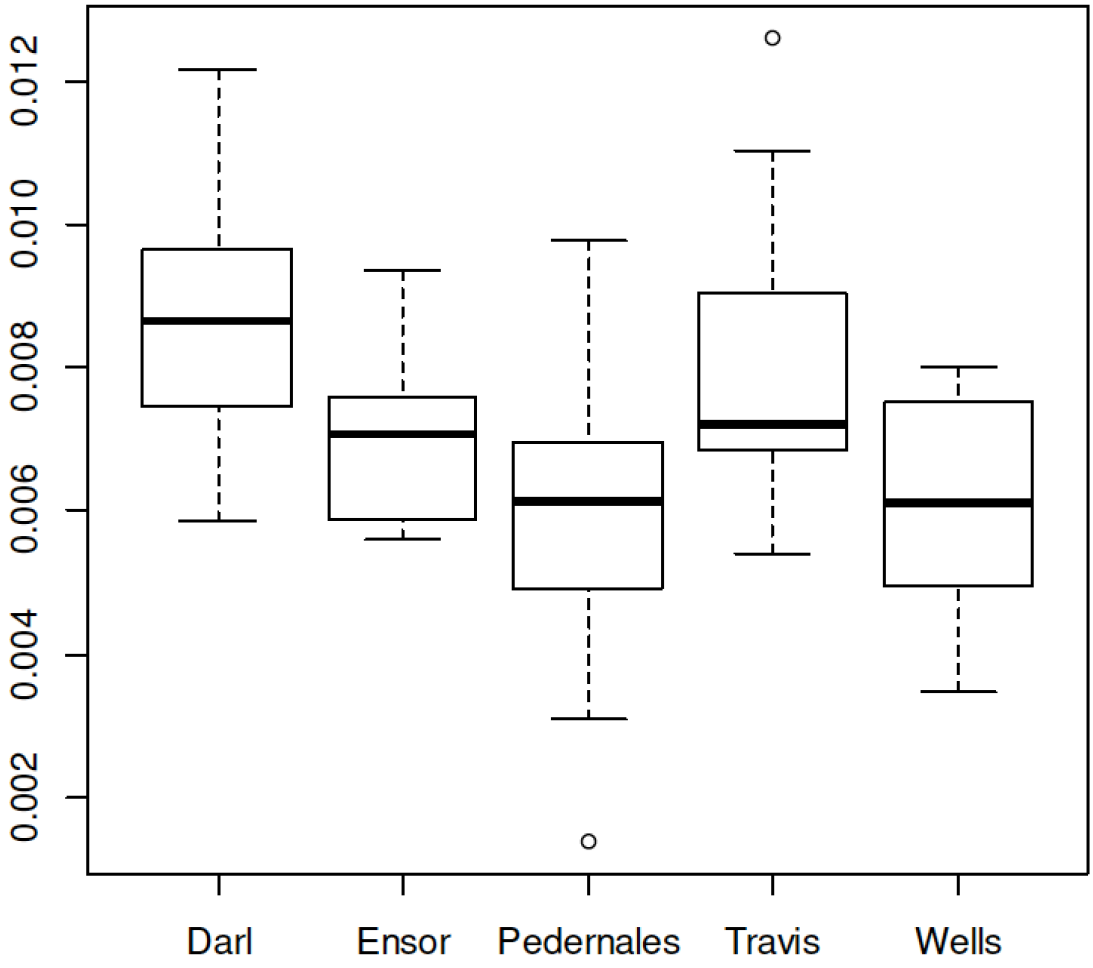
Armaremos primeramente, un nuevo objeto con el peso y con el índice de robustez, lo haremos paso por paso.

```
>b<-data.frame(DartPoints$Name,DartPoints[,5:7],
DartPoints[,11])
>head(b)
  DartPoints.Name Length Width Thickness DartPoints...11.
1            Darl   42.8  15.8       5.8              3.6
2            Darl   40.5  17.4       5.8              4.5
3            Darl   37.5  16.3       6.1              3.6
4            Darl   40.3  16.1       6.3              4.0
5            Darl   30.6  17.1       4.0              2.3
6            Darl   41.8  16.8       4.1              3.0

>names(b)<-c("Casos", "Largo","Ancho","Espesor", "Peso")
>head(b)
  Casos Largo Ancho Espesor Peso
1  Darl  42.8  15.8     5.8  3.6
2  Darl  40.5  17.4     5.8  4.5
3  Darl  37.5  16.3     6.1  3.6
4  Darl  40.3  16.1     6.3  4.0
5  Darl  30.6  17.1     4.0  2.3
6  Darl  41.8  16.8     4.1  3.0

>b$Robustez<-b$Espesor/(b$Largo*b$Ancho)#armemos el índice
>head(b)
  Casos Largo Ancho Espesor Peso    Robustez
1  Darl  42.8  15.8     5.8  3.6 0.008576837
2  Darl  40.5  17.4     5.8  4.5 0.008230453
3  Darl  37.5  16.3     6.1  3.6 0.009979550
4  Darl  40.3  16.1     6.3  4.0 0.009709785
5  Darl  30.6  17.1     4.0  2.3 0.007644383
```

```
6   Darl   41.8   16.8      4.1   3.0 0.005838460
```

Podemos darle ahora un primer vistazo a este índice para cada una de las clases:

```
>boxplot(b$Robustez~b$Casos)
```



Box plot del índice de robustez para cada clase.

Vemos que si bien hay algunos casos extremos en Pedernales y Travis, a los que podemos identificar como *outliers*, la distribución es en general, bastante homogénea en todos los casos. Utilizaremos la función lineal básica de R lm():

```
>lm1<-lm(b$Robustez~b$Peso)
>plot(b$Peso,b$Robustez,    pch=unclass(b$Casos))#Hagamos    un
```
gráfico bivariado y utilicemos la función lineal obtenida para observar la estimación

```
>legend("right", legend=levels(b$Casos), pch=c(1:3))
>abline(lm1)#el comando abline()toma la pendiente de regresión
```
estimada, (almacenada en el objeto lm1) y la utiliza en el gráfico

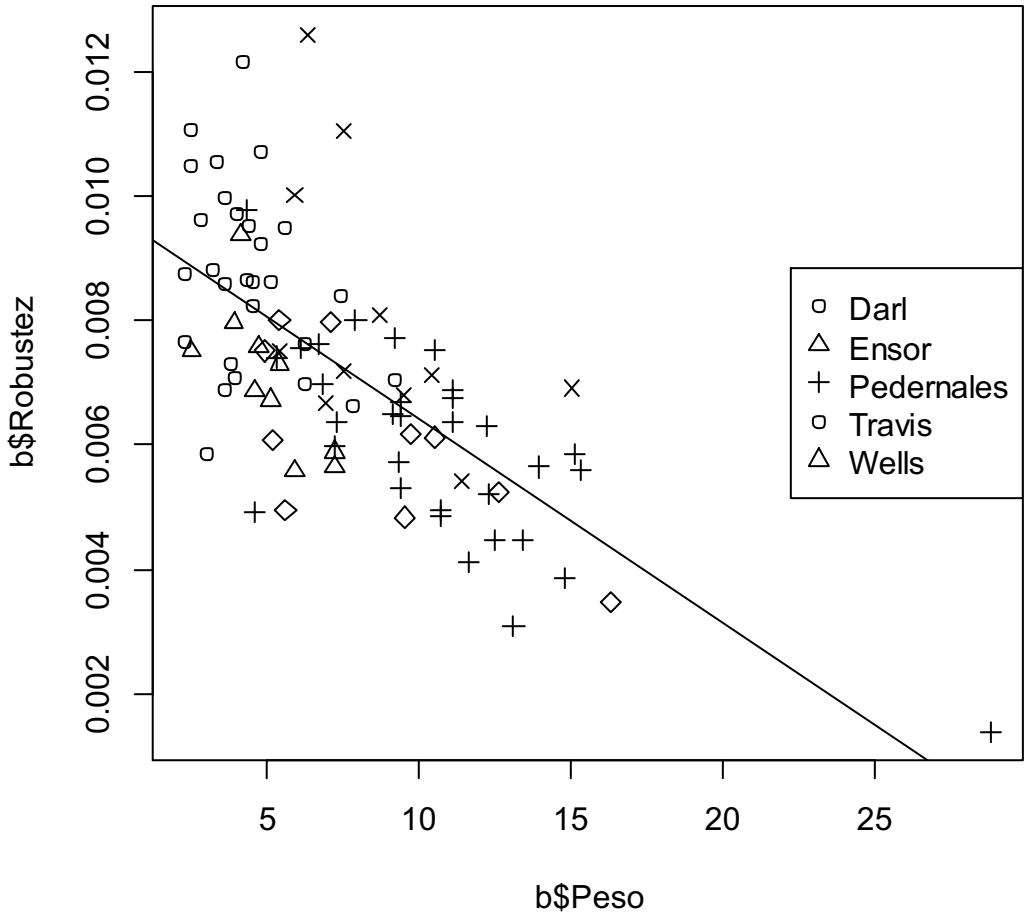


Gráfico de regresión entre el peso y el índice de robustez, identificando además, las distintas clases.

El gráfico nos muestra una tendencia relativamente lineal entre la variable explicativa y la variable respuesta, con un caso que parece ser muy influyente perteneciente a la muestra

de Pedernales, ya que posee un gran peso y muy baja robustez. La dirección de la recta nos señala además que la relación es negativa, a medida que se incrementa el peso, las piezas son menos robustas, es decir más alargadas.

Pero para comprender mejor el ajuste de nuestro modelo lineal, debemos ver los resultados con mayor detalle:

```
>summary(lm1)
Call:
lm(formula = b$Robustez ~ b$Peso)

Residuals:
       Min         1Q     Median         3Q        Max
-0.0032739 -0.0009659 -0.0001330  0.0009133  0.0049707

Coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept)  9.700e-03  3.187e-04  30.437  < 2e-16 ***
b$Peso      -3.279e-04  3.658e-05  -8.967 4.41e-14 ***

Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 0.00146 on 89 degrees of freedom
Multiple R-squared:  0.4746,    Adjusted R-squared:  0.4687
F-statistic:  80.4 on 1 and 89 DF,  p-value: 4.406e-14
```

El comando summary() nos devuelve los parámetros estimados en la regresión y el contraste de la H0 que sostiene que la pendiente no es diferente de forma significativa de 0. También obtenemos el valor de probabilidad para el coeficiente de determinación R^2, que en este caso es del 0.47, el estadístico F se utiliza para contrastar si el coeficiente de determinación explica una porción significativa de la varianza

total. En este caso, tanto la pendiente, como la varianza explicada, se apartan de manera significativa a lo esperado si la H0 fuese cierta.

Como todo modelo, la regresión por mínimos cuadrados posee supuestos, siendo el más importante la distribución normal de los residuos, que señala que utilizamos una aproximación adecuada para el estudio de nuestros datos.

Podemos ver primeramente estos residuos y realizar una correlación con los valores ajustados por el modelo lineal, en este caso, no esperamos que pueda detectarse ningún patrón:

```
>plot(resid(lm1) ~ fitted(lm1),xlab = "Valores ajustados", ylab = "Residuales")#resid y fitted son los nombres de los residuales del modelo y los valores ajustados respectivamente, que están almacenados en lm1.
>abline(h=0, col="red")# trazamos una recta en y=0, ya que esperamos que los valores se distribuyan aleatoriamente en torno a una pendiente nula.
```

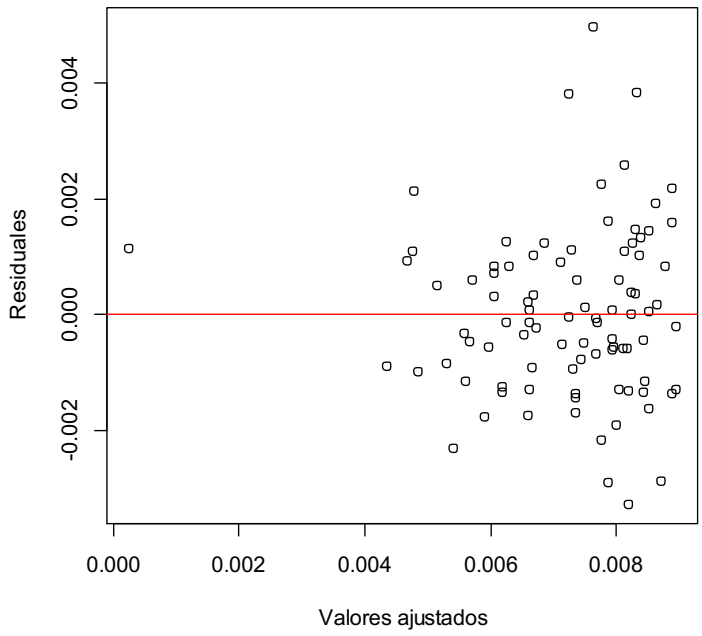


Gráfico de residuales vs valores ajustados.

No es difícil apreciar en el gráfico que los residuos no se distribuyen homogéneamente a lo largo de la dispersión, sino que existe una mayor varianza en relación a los valores más altos, mientras que uno de los residuos se aleja de los demás sobre el eje de las "x". Este puede ser un valor influyente dentro de nuestro modelo. Pero cómo podemos estar seguros de qué valores son los potencialmente influyentes en nuestro modelo?. R posee muchas herramientas diagnósticas para modelos de regresión, una de las más prácticas es el gráfico de valores influyentes influencePlot() del paquete car().

La función crea un gráfico de "burbuja" que compara los residuales estandarizados en relación a los valores predichos y donde el tamaño de cada una es proporcional a la distancia de Cook. Esta medida de distancia es comúnmente empleada en regresión de mínimos cuadrados para evaluar casos influyentes.

```
>library(car)
>influencePlot(lm1)
     StudRes         Hat       CookD
44 0.9229558 0.29199035 0.17594815
78 3.6562256 0.01212103 0.07200472
```

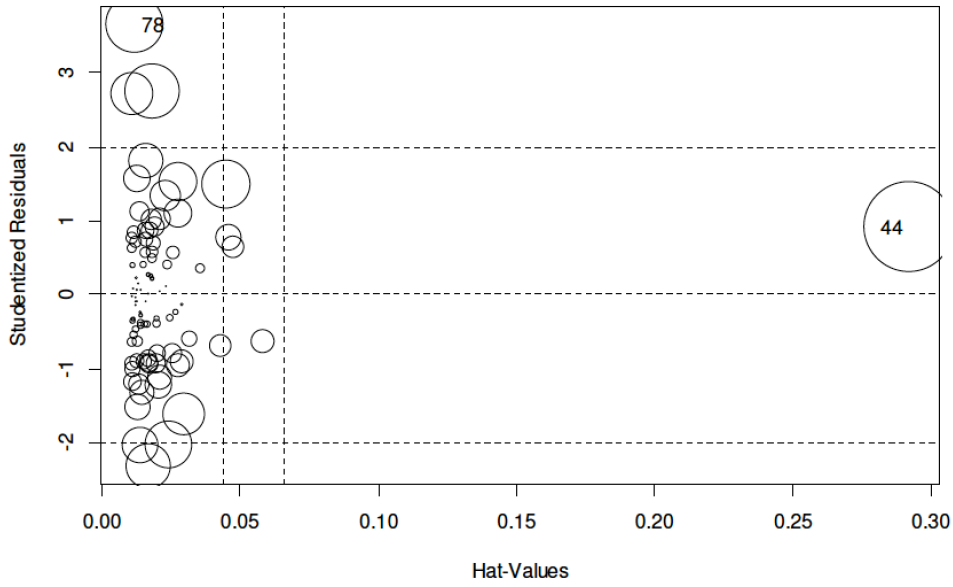


Gráfico de valores influyentes para la regresión.

El resultado señala dos casos como atípicos e influyentes, el 44 y el 78. De ambos, el 44 es particularmente influyente porque posee un valor más alto en el eje de los valores predichos (Hat) y un bajo residuo (por lo que se encuentra casi sobre la recta de regresión). El 78 en cambio, parece ser más bien un caso atípico, con un alto residuo, más alejado de la recta predicha por lm1. Podemos remover entonces este caso para explorar su importancia y/o reevaluar los resultados.

```
>b2<-b[-c(44), ]
>lm2<-lm(b2$Robustez~b2$Peso)
>influencePlot(lm2)
      StudRes        Hat      CookD
75  1.6449023 0.06165796 0.08720489
77  3.6464714 0.01218723 0.07196841
90 -0.4886035 0.08045000 0.01053434
```

Al volver a hacer la estimación sin el caso 44 vemos que otros casos nuevos poseen influencia en el análisis, aunque no es tan grande la distancia de Cook. El caso 78 (ahora 77) sigue siendo atípico y pobremente predicho, con un alto residuo. Veamos el nuevo gráfico y el resultado del ajuste del lm2:

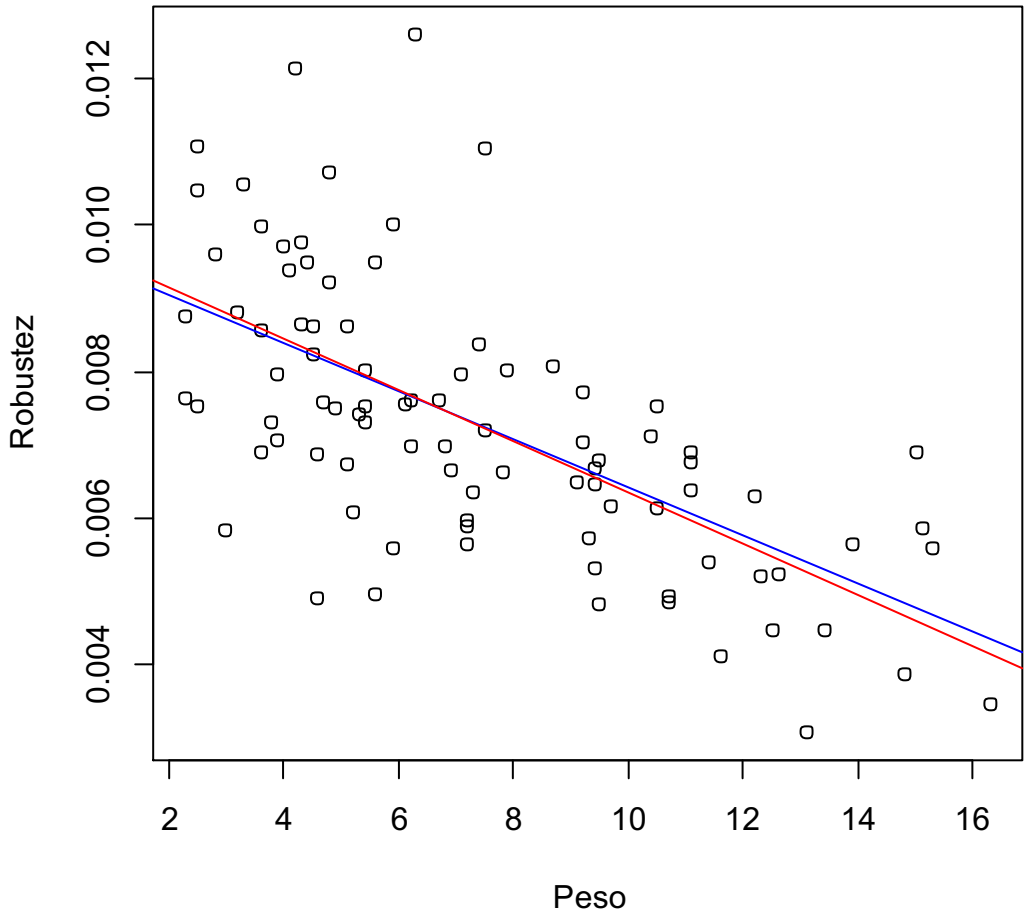
```
>plot(Robustez ~ Peso, data=b2)
>summary(lm2)

Call:
lm(formula = b2$Robustez ~ b2$Peso)

Residuals:
       Min         1Q     Median         3Q        Max
-0.0033211 -0.0009213 -0.0000802  0.0009293  0.0049598
```

```
Coefficients:
              Estimate Std. Error t value Pr(>|t|)
(Intercept)  9.845e-03  3.556e-04  27.687  < 2e-16 ***
b2$Peso     -3.492e-04  4.326e-05  -8.072  3.3e-12 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 0.001461 on 88 degrees of freedom
Multiple R-squared:  0.4254,    Adjusted R-squared:  0.4189
F-statistic: 65.16 on 1 and 88 DF,  p-value: 3.3e-12
```

Al quitar el valor influyente, tal como suele ocurrir, nuestro porcentaje de la varianza explicada (si bien los resultados son significativos) cae de 47% a 41%.

Podemos agregar a nuestro nuevo gráfico las dos rectas de regresión para ver el cambio en la pendiente entre lm1 y lm2:

```
>abline(lm1, col="blue")
>abline(lm2, col="red")
```

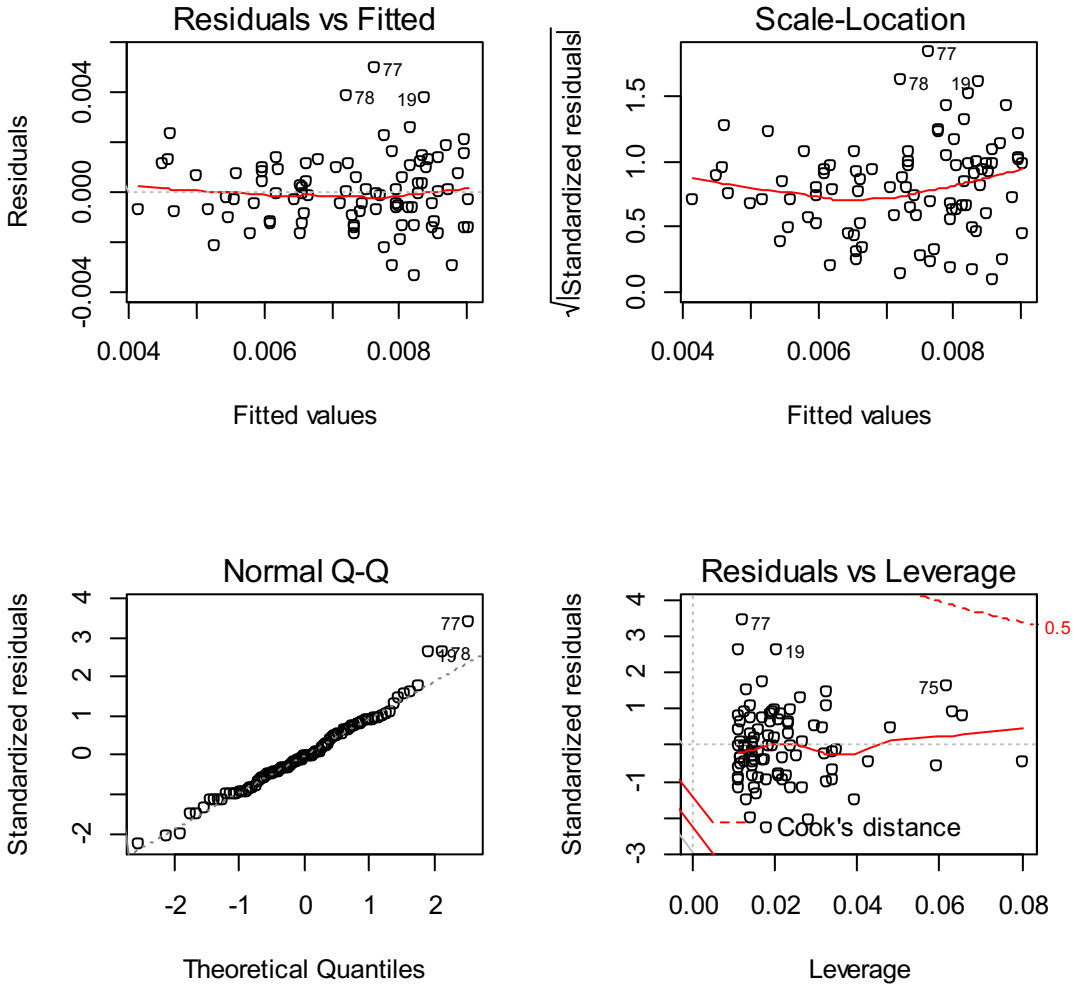


Gráficos de ajuste para el modelo lineal con dos pendientes.

Como se observa al remover este caso, no tiene un gran impacto predictivo, ya que las pendientes son similares. No es conveniente comparar mediante un test ambas regresiones porque son muestras con diferente número de casos. Sin embargo podemos apreciar una mejora en el ajuste en el segundo modelo.

Por defecto, la función lm() posee un conjunto de herramientas para la evaluación del ajuste de los modelos lineales, basta con tipear plot() para que se generen cuatro gráficos comparando los valores ajustados, la varianza del modelo y el QQ plot sobre la distribución normal de los residuos.

```
>par(mfrow=c(2,2))#    Grafiquemos    los    cuatro    gráficos
diagnósticos juntos
>plot(lm2)
```



Gráficos de ajuste para el modelo lineal.

Nuevamente, podemos ver que algunos casos, como el 77 poseen un bajo ajuste en relación al modelo, aunque la importancia de los distintos casos varía en relación a los distintos parámetros que se evalúan.

Podemos ver los predichos utilizando la función predict(), veremos los primeros 6:

```
>head(predict(lm2))
```

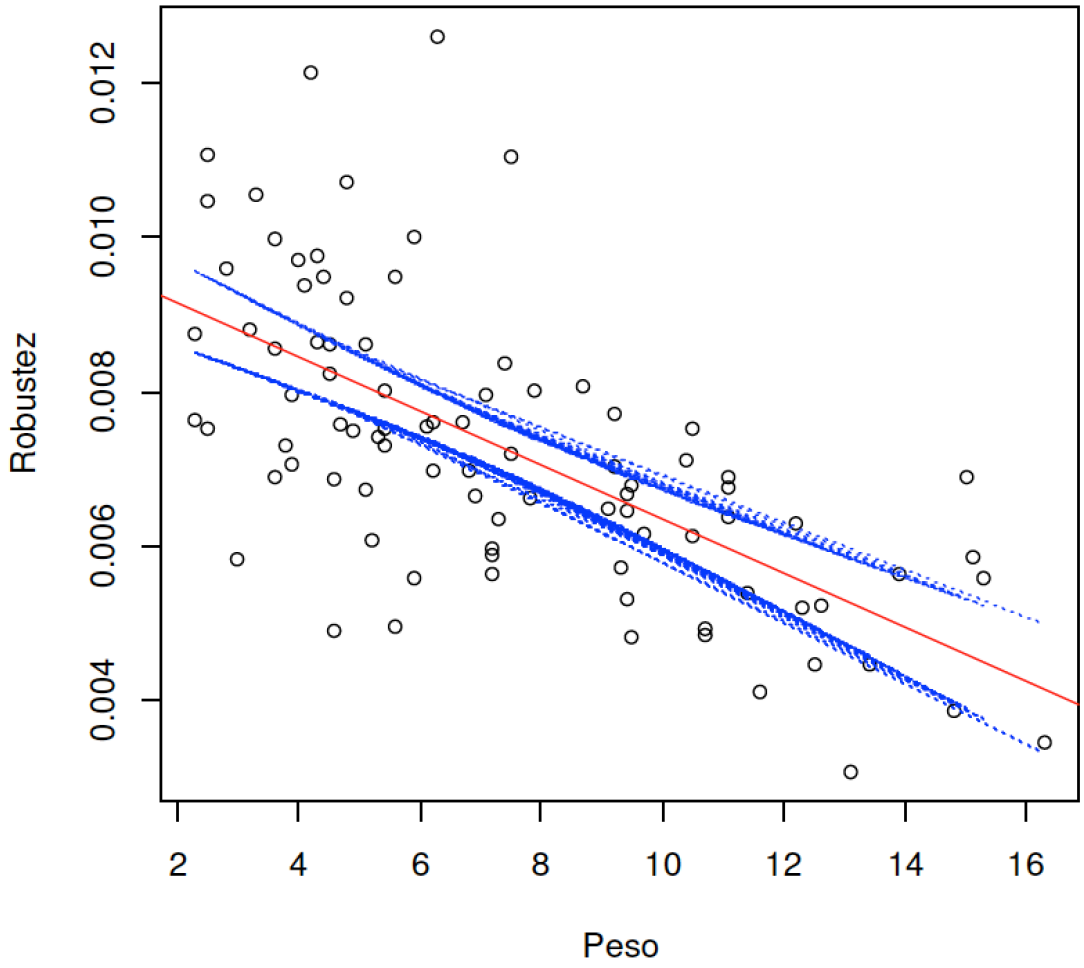
```
          1           2           3           4           5           6
0.008587550 0.008273236 0.008587550 0.008447855 0.009041559 0.008797093
```

Asimismo, en algunos casos, nos puede interesar graficar el error o intervalo para cada predicción en torno a la recta. Podemos estimar el intervalo para cada predicho de la siguiente manera:

```
>head(predict(lm2, interval="confidence"))#por defecto, este intervalo es del 95%
          fit         lwr         upr
1 0.008587550 0.008139388 0.009035712
2 0.008273236 0.007878053 0.008668420
3 0.008587550 0.008139388 0.009035712
4 0.008447855 0.008024165 0.008871545
5 0.009041559 0.008506276 0.009576843
6 0.008797093 0.008309970 0.009284216
```

Mejor aún, podemos agregar este error de predicción a nuestra línea de regresión:

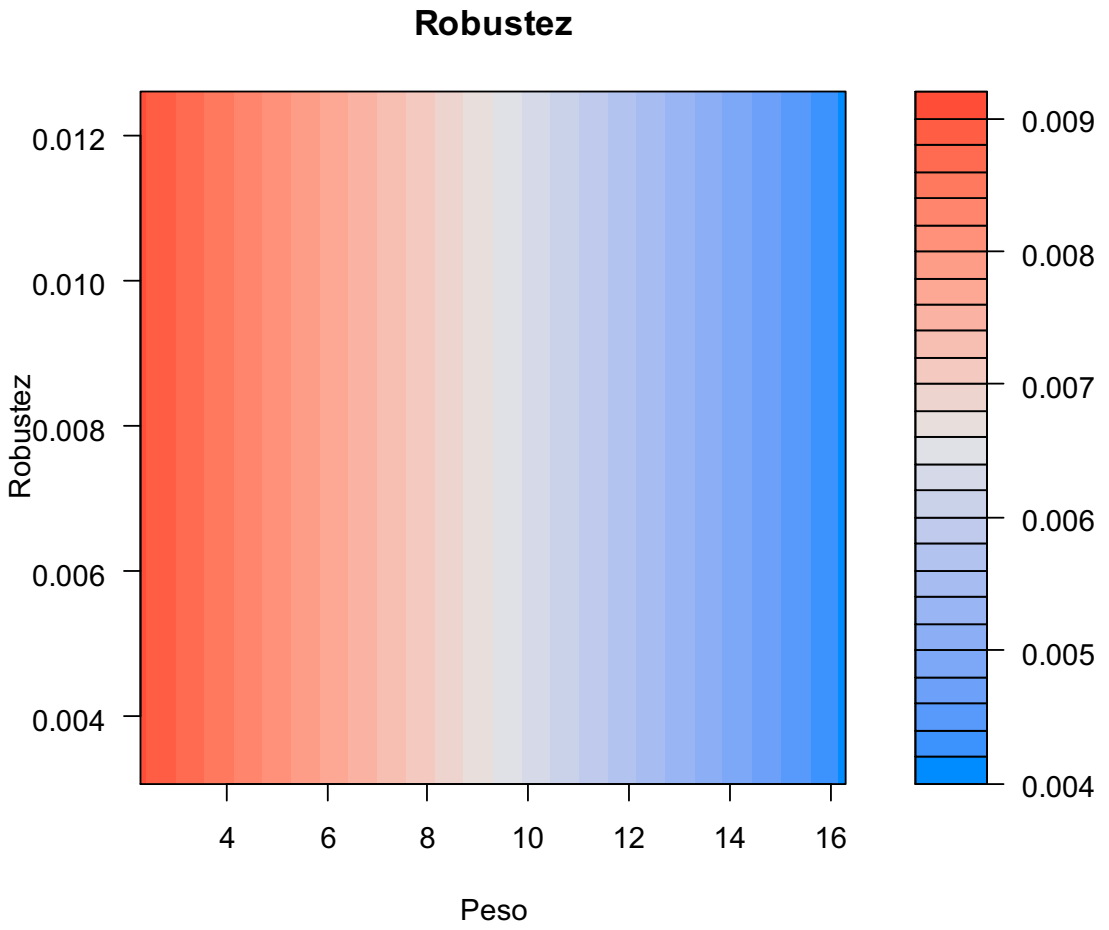
```
>pred <- predict(lm2, interval="confidence")
>plot(b2$Peso,b2$Robustez, ylab="Robustez", xlab="Peso")
>abline(lm2, col="red")
>lines(b2$Peso,pred[,2], lty=3, col="blue"intervalo inferior
>lines(b2$Peso,pred[,3], lty=3, col="blue")#intervalo superior
```



Regresión robustez~peso con intervalos de confianza en torno a la recta.

En este acápite de regresión, hemos realizado muchos cálculos y gráficos "a mano", pero existen numerosos paquetes que poseen resultados gráficos más sofisticados de forma prácticamente automática, como el paquete visreg(). Podemos graficar, por ejemplo los predichos como un mosaico que muestra los cambios de y (robustez) para cada cambio de x (peso).

```
>library(visreg)
>visreg2d(lm1,   x="Peso",   y="Robustez",   plot.type="image")
#otras alternativas son posibles ver ?visreg.
```



Valores de robustez predichos por la recta de regresión en forma de mosaico, la barra a la derecha, indica los valores correspondientes a cada cambio de color y tono.

Si la distribución no se ajusta a la regresión, es probable que sea conveniente emplear otros procedimientos de ajuste o transformar los datos, y que en algunos casos eso los "normaliza". Otros procedimientos como los modelos lineales generalizados, o el empleo de polinomios, (en donde se agregan nuevos términos a la recta para mejorar su ajuste), pueden ser más convenientes, aunque implican más supuestos.

Tercera Parte

3. Test de hipótesis. Métodos paramétricos y no-paramétricos.

3.1. Distribuciones de probabilidad y test de hipótesis.

3.1.1. Ley de los grandes números.

La ley de los grandes números sostiene, que al repetir un experimento aleatorio un número grande de veces, la frecuencia relativa de cada suceso (por ejemplo su media, o la proporción que aparece cara o seca en una moneda) tiende a aproximarse a un número fijo, denominado como la probabilidad del suceso.

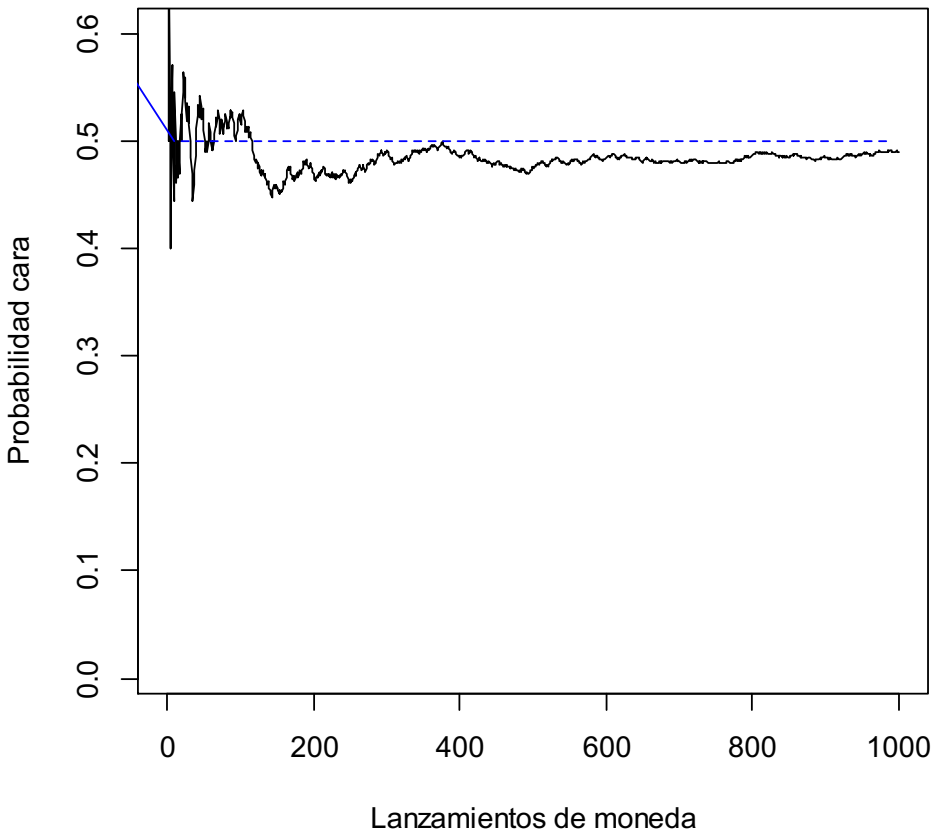
Por ejemplo si realizamos la prueba del lanzamiento de la moneda cada lanzamiento es un experimento de Bernoulli, en el que existen sólo dos posibilidades con una la probabilidad de ½ cada una.

Sin embargo al lanzar un número repetido de veces y calcular uno u otro resultado, la proporción estimada de éxitos tenderá a alcanzar la proporción poblacional. Esto demuestra que, al incrementarse el tamaño de la muestra los valores muestrales tienden a confluir con los poblacionales.

Podemos modelar la ley de los grandes números con diferentes parámetros para entender mejor sus implicancias en

relación al tamaño muestra y la estimación de los valores poblacionales con un script:

```
>Muestra <- sample(0:1, 1000, repl=T)#números aleatorios 0 ó 1, (ver página 39)
>Acum <- cumsum(Muestra)#el comando cumsum() indica sumas acumulativas
>Distribucion <- Acum/(1:1000)#La frecuencia acumulada se divide por la misma extensión de la muestra (números enteros de 1 a 1000)
>plot(Distribucion, ylim=c(.01, .60), type="l", xlab="Lanzamientos de moneda", ylab="Probabilidad cara")
lines(c(0,1000), c(.50,.50),col="blue", lty=2)
```



Simulación de la ley de los grandes números para estimar una proporción poblacional de 0.5 en el lanzamiento de una moneda.

La simulación muestra también que en 1000 repeticiones, el valor estimado aún posee un grado de error, ya que no alcanza el verdadero valor del parámetro aunque se tiende a aproximar cada vez más.

3.2. Variables cuantitativas continuas: La distribución normal.

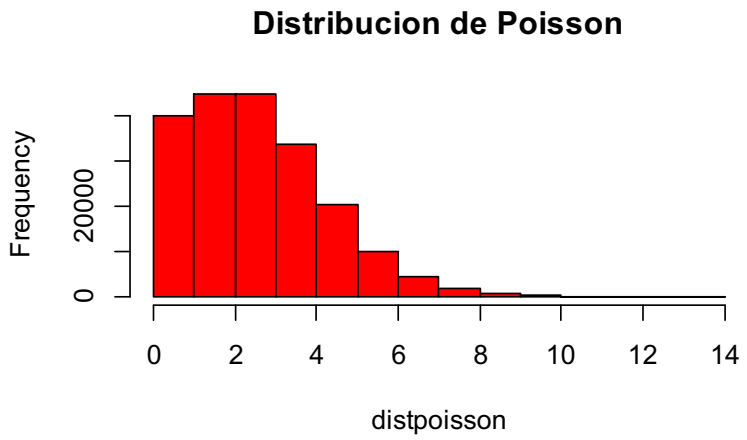
Las distribuciones continuas son aquellas que describen el comportamiento de variables aleatorias continuas. A diferencia de las variables aleatorias discretas, las variables continuas pueden tomar infinitos valores, por lo que es imposible asignar una probabilidad a cada valor individual ya que ésta es prácticamente 0.

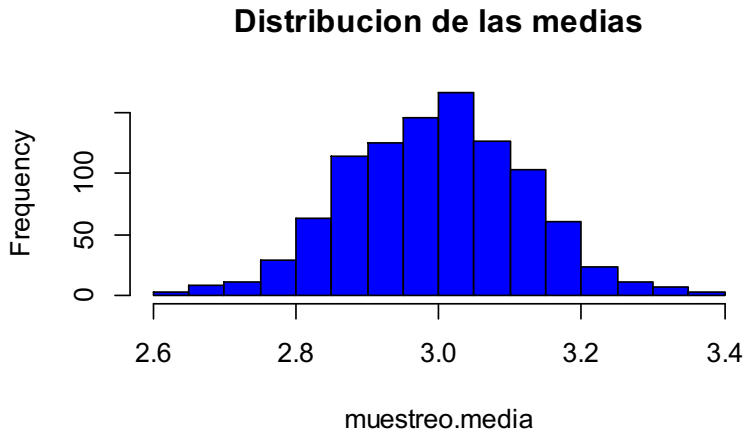
Por este motivo, la probabilidad para una variable aleatoria continua es descripta a partir de áreas que abarcan un rango de valores determinados dentro de las distribuciones continuas de probabilidad. De éstas, la distribución normal es la más conocida y la más importante. La relevancia de la distribución normal está explicada por el teorema central del límite que prueba que aunque una muestra no se distribuya normalmente, los estadísticos (media, varianza) estimados de muestreos repetidos se distribuirán normalmente.

Veamos este teorema en R, generando primero números aleatorios para simular una distribución no-normal, en este caso, de conteos. La más conocida de las distribuciones para este tipo de datos, es la de Poisson, que depende solamente de una tasa o frecuencia de sucesos en relación a tiempo o espacio:

```
>a<-1000
>n <- 200
```

```
>distpoisson <- rpois(a*n,lambda=3)#queremos construir una matriz de a*n=20000 casos que serán numeros aleatorios enteros.
>sample.means <- function(distpoisson, a, n) {
  rowMeans(matrix(distpoisson,nrow=a,ncol=n))
}#Luego se realizarán a muestreos de los cuales estimaremos la media sobre la distribución de Poisson.
>muestreo.media <- sample.means(distpoisson, a, n)
>par(mfrow=c(2,1))
>hist(distpoisson, col="red",main="Distribucion de Poisson")
>hist(muestreo.media, col="blue",main="Distribucion de las medias")
```



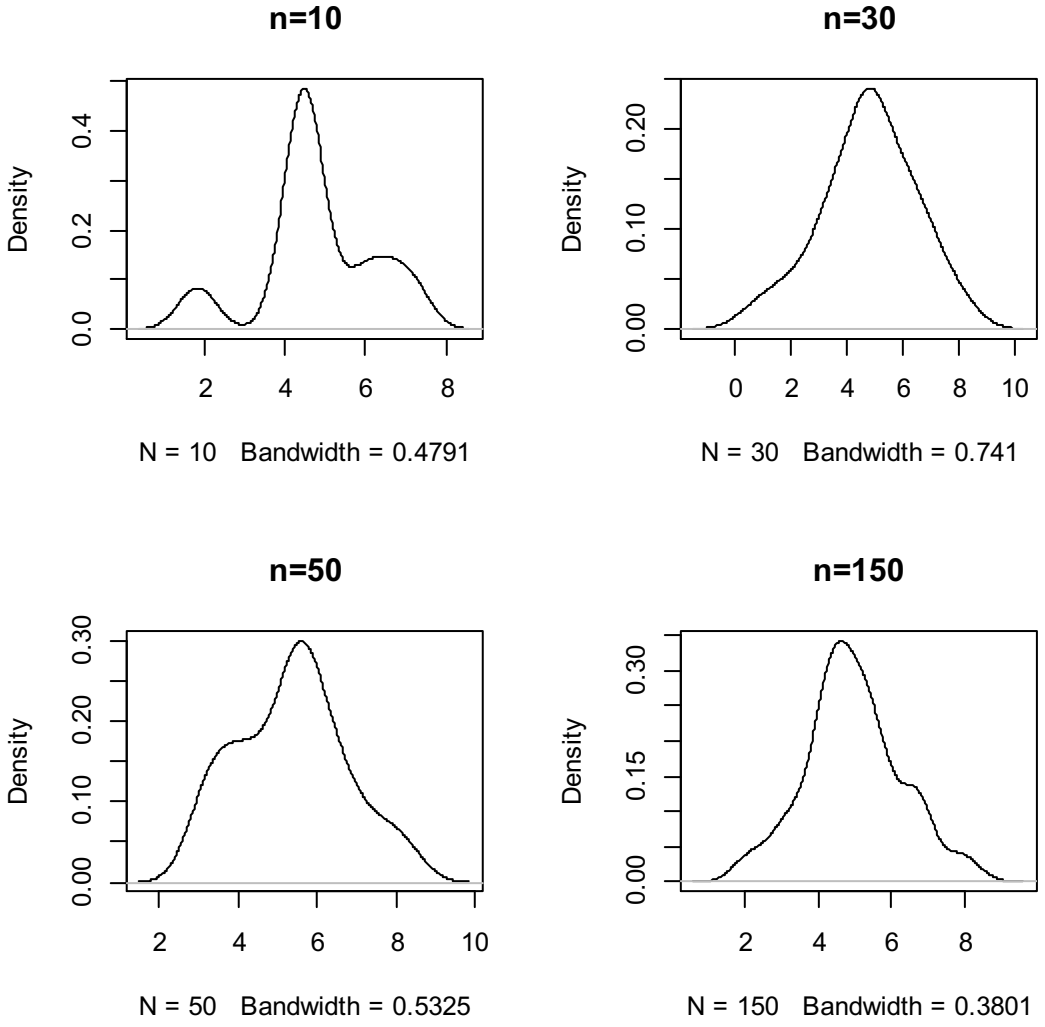


Histograma de una distribución de Poisson de 20000 casos y histograma de las 1000 medias estimadas mediante muestreo aleatorio sobre ésta.

Este ejemplo sostiene la propiedad fundamental de la distribución de los estadísticos estimados en repetidos muestreos aleatorios del mismo tamaño, aunque dichas muestras provengan de una distribución no-normal. Si bien hay distintas distribuciones normales, cada una puede describirse por su media y desvío estándar. El desvío estándar determina el ancho de la curva, mientras que la media representa el punto de mayor densidad (altura) de la distribución y puede ser cualquier valor numérico. En distribución normal (o gaussiana) ideal la distribución es simétrica y sus colas se prolongan hasta el infinito sin tocar el eje de las x.

En la realidad todos estos rasgos raramente se cumplen, sino que se darán distintos grados de aproximación a esta forma teórica, veamos algunas distribuciones generadas aleatoriamente, con diferente tamaño de muestra (10, 30,50,150):

```
>a<-rnorm(10, mean = 5, sd = 1.5)#rnom es la función para generar números aleatorios con una distribución normal
>ab<-density(a)
>b<-rnorm(30, mean = 5, sd = 1.5)
>bb<-density(b)
>c<-rnorm(50, mean = 5, sd = 1.5)
>cb<-density(c)
>d<-rnorm(150, mean = 5, sd = 1.5)
>db<-density(d)
>par(mfrow=c(2,2))
>plot(ab, main="n=10")
>plot(bb, main="n=30")
>plot(cb, main="n=50")
>plot(db, main="n=150")
```



Distintas distribuciones de densidad para números aleatorios normalmente distribuídos. La simetría tiende a incrementarse con el tamaño de la muestra.

Nos interesa la distribución normal ya que ésta es la base del contraste de hipótesis paramétrico. Conocer la forma en que se distribuyen los datos es fundamental al momento de realizar estos tipos de *test*. Para ello puede ser muy útil comparar nuestros datos contra una o más distribuciones teóricas.

Por ejemplo, en el caso de las puntas de proyectil, ya vimos que el largo, no se distribuía del todo homogéneamente

(la más simétricamente distribuída era el espesor). A qué distribución se acerca esta variable aleatoria?. El paquete fitdistrplus() y la función descdist(), ajusta distintas distribuciones a nuestros datos simultáneamente y grafica su cercanía a una u otra.

```
>library(fitdistrplus)
>descdist(puntas$Largo, discrete = FALSE)
summary statistics
------
min:  30.6   max:  109.5
median:  47.1
mean:  49.33077
estimated sd:  12.73619
estimated skewness:  1.524863
estimated kurtosis:  7.637621
```

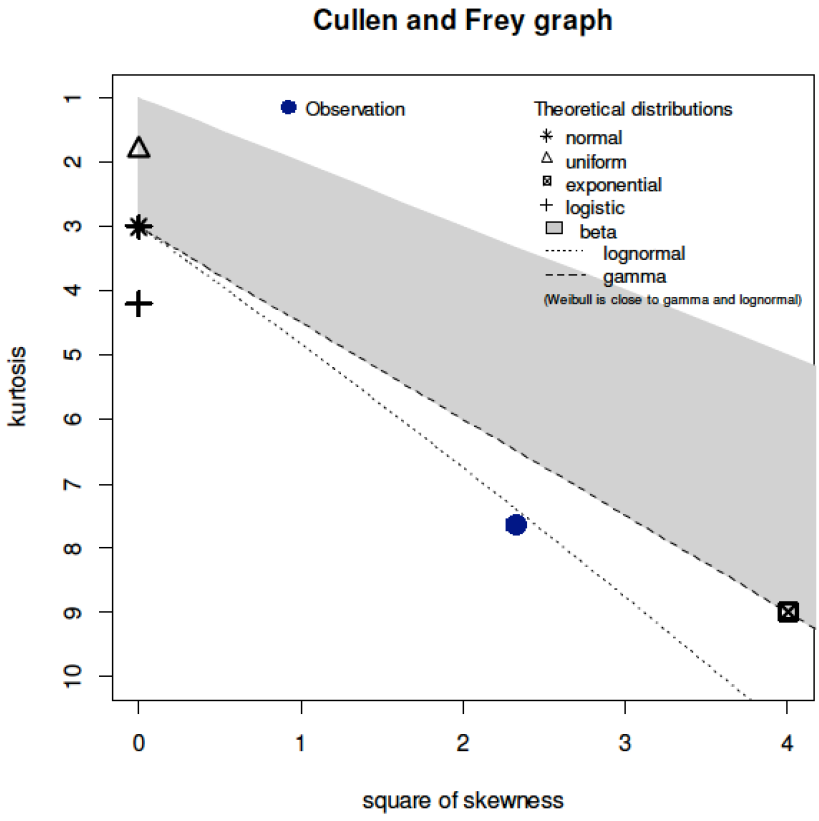
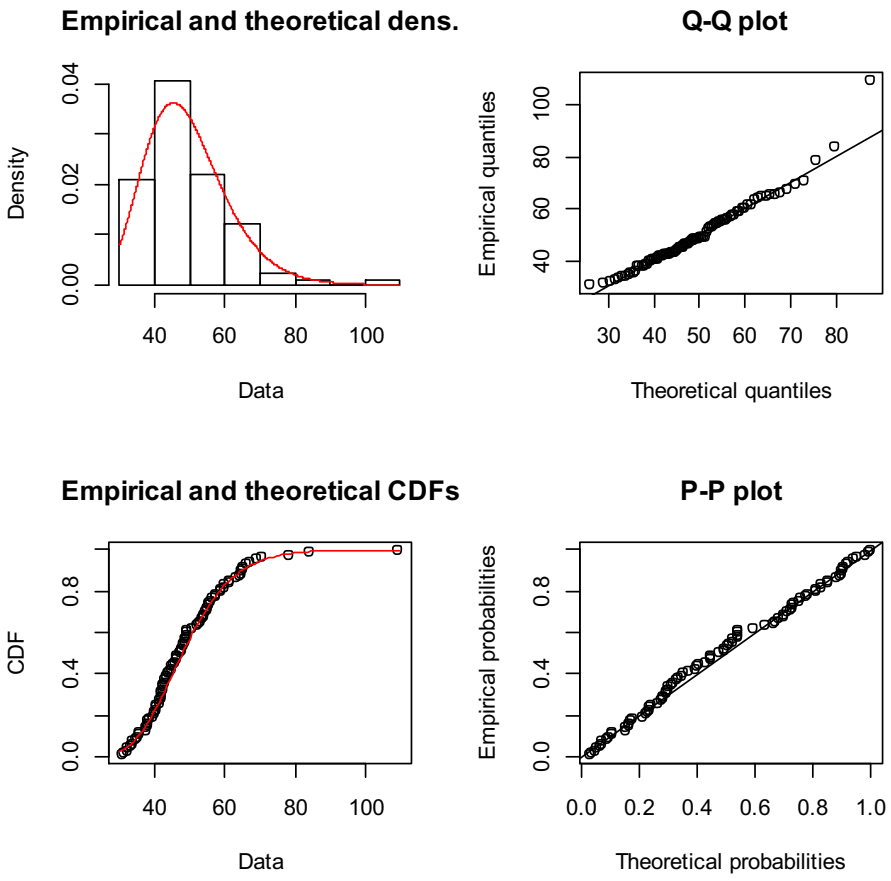


Gráfico de distancia entre nuestra observación (círculo azul) y distintas distribuciones.

Parece ser que la variable Largo, se distribuye más bien de forma logarítmico normal o lognormal. Es posible emplear la función fitdist() y comparar ahora, nuestros datos directamente con la distribución logarítmico normal, que parece ser la que mejor los describe.

```
>fit<-fitdist(puntas$Largo, "lnorm")
>fit
Fitting of the distribution ' lnorm ' by maximum likelihood
Parameters:
         estimate Std. Error
meanlog 3.8693439 0.02481457
sdlog   0.2367159 0.01754514

>plot(fit)
```



Gráficos de ajuste de la función logarítmico normal a la variable "Largo".

Si se tiene que decidir un *test* de hipótesis o interpretar los datos a partir de más de una posible distribución, se pueden comparar mediante el índice de Akaike (AIC), donde el valor menor indica un mejor ajuste de cada modelo y por lo tanto es el preferible. Si la diferencia es pequeña puede ser conveniente utilizar la distribución más sencilla de aplicar.

Podemos ajustar ambas distribuciones y luego ver cuál de ellas posee menor AIC

```
>fit<-fitdist(puntas$Largo, "lnorm")#logarítmico normal
>fit2<-fitdist(puntas$Largo, "norm")#normal
>fit$aic #El valor de AIC o Akaike ya se encuentra dentro de los resultados
[1] 704.2246
>fit2$aic
[1] 724.3308
```

Como se observa en los resultados obtenidos, el más parsimonioso de los ajustes es efectivamente, el logarítmico normal. En casos como éste, es sencillo transformar los datos originales y ajustarlos a la distribución gaussiana aplicando el logaritmo natural mediante la función log(x)).

3.3. Estimación de probabilidades mediante simulación y remuestreo.

Con el avance de las técnicas computacionales se han desarrollado ampliamente el empleo de simulaciones en procedimientos de contraste de hipótesis. Estos métodos permiten, entre otras cosas, estimar intervalos de confianza en fenómenos que no cumplen los supuestos clásicos de normalidad (por ejemplo, en muestras pequeñas) ni se ajustan a las distribuciones matemáticas más conocidas. Por

consiguiente, estos métodos son especialmente útiles en el contraste de hipótesis basadas en datos cuyos parámetros o su distribución no pueden establecerse con certeza.

En general se los denomina métodos de remuestreo o simulación, de manera más específica existen distintos procedimientos como bootstrap, jackknife, Montecarlo, permutaciones. etc. En R son múltiples los paquetes que permiten el contraste de hipótesis basados en estos métodos como alternativas a los métodos paramétricos o no paramétricos más tradicionales. El paquete coin() o lperm() que utilizaremos más adelante ofrece opciones vinculadas a los test clásicos.

3.4. Test paramétricos.

Volvamos a armar un archivo de la variable Largo pero incorporando las clases, ya que nos interesa comparar entre sí alguna de ellas:

```
>puntas<-data.frame(DartPoints$Name,DartPoints$Length)
>head(puntas)
  DartPoints.Name DartPoints.Length
1            Darl              42.8
2            Darl              40.5
3            Darl              37.5
4            Darl              40.3
5            Darl              30.6
6            Darl              41.8

>names(puntas) <- c("Casos", "Largo")
>head(puntas)
  Casos Largo
1  Darl  42.8
```

```
2   Darl   40.5
3   Darl   37.5
4   Darl   40.3
5   Darl   30.6
6   Darl   41.8
```

Sabemos que la variable Largo no se distribuye normalmente, pero en este caso nos interesa comparar algunas clases entre sí, por ese motivo, grafiquemos el largo en función de las clases:

```
>boxplot(Largo~Casos, data=puntas)
```

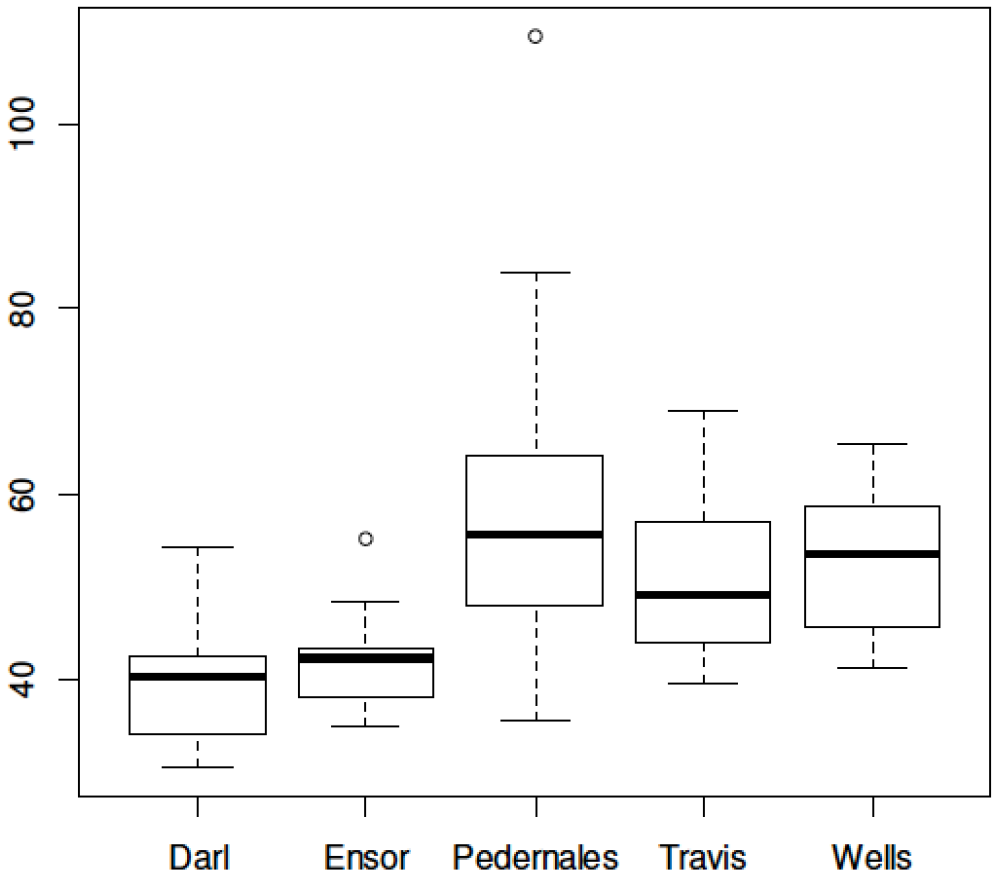


Gráfico de caja del Largo en función de las Clases.

Tal como se observa, parecen existir diferencias en las distribuciones del largo de las distintas clases de puntas de proyectil (Darl y Ensor parecen diferentes a Pedernales, Travis y Wells). También se observan posibles *outliers*, lo que era esperable dado que ya sabemos que la variable no se distribuye normalmente.

Identifiquemos los datos extremos:

```
>box<-boxplot(Largo~Casos, data=puntas)
>box$out# el objeto box ya contiene información sobre los outliers
[1]  55.2  109.5
```

Ahora sabemos que dos casos son outliers en esta distribución (de largo 55.2 y 109.5) y claramente se corresponden a Ensor y Pedernales respectivamente. No es claro, en principio, que peso pueden tener estos datos extremos en el ajuste de los distintos test de hipótesis, pero es factible que esto afecte su performance. Tomaremos entonces, dos casos de distribución más o menos homogénea para compararlos entre sí.

3.4.1. Test paramétricos bivariados.

Que ocurre cuando tenemos dos muestras de las cuales pensamos que pueden existir diferencias significativas, en al menos una de las variables que empleamos para describirlas: ¿pertenecen estas dos muestras a la misma población?. Si la distribución estudiada corresponde a una variable cuantitativa, una posibilidad es realizar un *test* de hipótesis sobre las dos medias. El más usual es el test de la *t*.

3.4.2. El test de la *t*.

El test de la *t* es el más conocido de los *test* estadísticos para comparar dos variables cuantitativas a partir de sus medias. Se basa en la distribución de *t* que es similar a la distribución normal, especialmente al incrementarse el tamaño de la muestra.

Mediante el test de la *t* se contrasta la H0 de que no existen diferencias entre las medias. Al igual que la mayoría de los test estadísticos, asume que las muestras pertenecen a poblaciones independientes entre sí y al ser basado en la media, requiere que se cumplan los supuestos de normalidad y el de la igualdad de las varianzas.

En este caso, utilizando el archivo de puntas, podemos estar interesados en establecer si existen diferencias significativas en el largo de dos grupos de puntas de nuestra muestra. En el box plot que hicimos anteriormente, se observa que la muestra de Darl y Wells, parecen tener distribuciones con valores centrales diferentes. ¿Cómo podemos compararlas?

Un primer paso, puede ser armar un nuevo objeto que agrupe sólo a estos dos casos, mediante el comando rbind.

```
>puntas3<-rbind(puntas[1:28,1:2], puntas[82:91,1:2])
>puntas3
   Casos Largo
1   Darl  42.8
2   Darl  40.5
3   Darl  37.5
4   Darl  40.3
5   Darl  30.6
6   Darl  41.8
7   Darl  40.3
8   Darl  48.5
```

```
9   Darl  47.7
10  Darl  33.6
11  Darl  32.4
12  Darl  42.2
13  Darl  33.5
14  Darl  41.8
15  Darl  38.0
16  Darl  35.5
17  Darl  31.2
18  Darl  34.5
19  Darl  33.1
20  Darl  32.0
21  Darl  38.1
22  Darl  47.6
23  Darl  42.3
24  Darl  38.3
25  Darl  50.6
26  Darl  54.2
27  Darl  44.2
28  Darl  40.0
82 Wells  53.1
83 Wells  41.2
84 Wells  45.0
85 Wells  54.2
86 Wells  65.4
87 Wells  58.9
88 Wells  55.4
89 Wells  45.8
90 Wells  49.1
91 Wells  63.1
```

Podemos ver que adecuadamente agrupó los datos en una sola columna. Luego emplearemos la función t.test() para comparar las medias de estos dos grupos.

Antes de aplicar el *test* y como comenzamos en este acápite a trabajar variables cuantitativas y con factores, podemos realizar una estadística descriptiva del largo a partir de los niveles del factor "Casos" con el paquete psych() y la función describeBy().

```
>describeBy(puntas3$Largo,puntas3$Casos)
$Darl
   vars  n  mean   sd median trimmed  mad  min  max range skew
kurtosis
X1    1 28 39.75 6.18  40.15   39.44 6.45 30.6 54.2  23.6 0.43
-0.63
     se
X1 1.17
$Ensor
NULL
$Pedernales
NULL
$Travis
NULL
$Wells
   vars  n  mean   sd median trimmed  mad  min  max range skew
kurtosis
X1    1 10 53.12 7.94  53.65   53.08 9.71 41.2 65.4  24.2 0.07
-1.46
     se
X1 2.51
attr(,"call")
by.default(data = x, INDICES = group, FUN = describe, type =
type)
```

Vemos que Wells tiene una media bastante mayor que Darl, incluyendo en la estimación que no toma en cuenta posibles valores extremos que potencialmente puedan afectarla (trimmed

mean). Esto es acorde a la información gráfica que obtuvimos previamente. También se observan las otras clases, que como las recortamos de nuestra matriz, aparecen con valores nulos (Null). Veamos ahora si los resultados del test sostienen esta apreciación:

```
>t.test(Largo~Casos, data=puntas3)
        Welch Two Sample t-test
data:  Largo by Casos
t = -4.8256, df = 13.101, p-value = 0.0003242
alternative hypothesis: true difference in means is not equal
to 0
95 percent confidence interval:
 -19.345801  -7.387056
sample estimates:
 mean in group Darl mean in group Wells
           39.75357            53.12000
```

El resultado indica diferencias significativas entre las medias del largo de Darl y Wells. El test de Welch es una variante del test de la *t* original que es robusto a las diferencias en la varianza. Sin embargo, la igualdad de las varianzas es un supuesto básico de este test, así que podemos compararlas con el test de la varianza (var.test()) o F y determinar si se cumple dicho supuesto.

```
>var.test(Largo~Casos, data=puntas3)
        F test to compare two variances
data:  Largo by Casos
F = 0.60438, num df = 27, denom df = 9, p-value = 0.2995
alternative hypothesis: true ratio of variances is not equal
to 1
95 percent confidence interval:
 0.1686119 1.5900398
```

```
sample estimates:
ratio of variances
         0.6043813
```

Como vemos, el test no rechaza la H0 de igualdad (p=0.29), así que podemos considerar que nuestro resultado es robusto y el test se empleó apropiadamente. Si esto no ocurriera, podemos aún realizar el test de la *t* empleando un test de permutaciones. El paquete coin(), el paquete Deducer() o perm(), poseen funciones adecuadas para ello:

```
>library(coin)
>oneway_test(Largo~Casos)
        Asymptotic Two-Sample Fisher-Pitman Permutation Test
data:  Largo by Casos (Darl, Wells)
Z = -4.0884, p-value = 4.344e-05
alternative hypothesis: true mu is not equal to 0

>library(perm)
>permTS(Largo~Casos)
        Permutation Test using Asymptotic Approximation
data:  Largo by Casos
Z = -4.0884, p-value = 4.344e-05
alternative hypothesis: true mean Casos=Darl - mean
Casos=Wells is not equal to 0
sample estimates:
mean Casos=Darl - mean Casos=Wells
                      -13.36643
```

Vemos que ambos procedimientos dan resultados parecidos al test de la *t*, lo que es esperable ya que se cumplen los supuestos y la probabilidad de que la H0 sea cierta es pequeña. Si esto no fuese así, sería esperable encontrar

mayores diferencias entre el *test* de la *t* y el de permutaciones. Distintas opciones como el control de los intervalos de confianza, dirección del *test* (dos colas, una cola) son posibles en ambas funciones.

3.4.3. Más de dos niveles: Análisis de la varianza.

El análisis de la varianza se plantea en un contexto donde una variable explicativa es un factor que posee más de dos niveles. En el caso que vimos anteriormente comparábamos el promedio de dos clases de puntas entre sí. Sin embargo, tal como se observa en la matriz original, existen más de dos niveles, representados por las distintas clases de la muestra (n=5).

En estos casos, nuestro interés puede ser comparar como se distribuye una misma variable cuantitativa entre estas clases y establecer con un determinado nivel de probabilidad, si pertenecen o no a una misma población. Por consiguiente, el objetivo del análisis de la varianza es contrastar en que medida, tres o más medias pertenecen a la misma población (H0). La hipótesis alternativa del test (Ha) sostiene que al menos uno de los casos (o niveles del factor), es distinto.

El *test* más ampliamente utilizado es el ANOVA o análisis de la varianza, de ellos el más conocido es el denominado de un factor o one-way ANOVA. Existen distintas formas de ajustar en R un ANOVA, ya que al tratarse de modelos lineales, puede abordarse como una regresión -o más comúnmente-, como una comparación entre medias. La función para esto último es aov().

Uno de los supuestos básicos del *test* es la homogeneidad de las varianzas, lo que aparentemente no se cumple para el largo, ya que en el acápite anterior, vimos que esta variable

tenía una distribución logarítmico normal. El espesor en cambio, es la variable más homogéneamente distribuida. Veremos que ocurre si queremos comparar esta variable entre todas las clases.

Primero armemos un archivo con el espesor y todas las clases o niveles del factor "Casos".

```
>puntas<-data.frame(DartPoints$Name,DartPoints$Thickness)
>names(puntas)<-c("Casos","Espesor")
```

Antes de conducir el *test*, podemos utilizar un método descriptivo gráfico que nos ayude a tener una primera idea de la distribución de los valores medios y la variación, con un intervalo de confianza del 95%. Para ello, utilizaremos la función plotmeans() del paquete gplot().

```
>library(gplot)
>plotmeans(Espesor ~ Casos, connect=T, p=0.95,  data=puntas,
col="red")
```

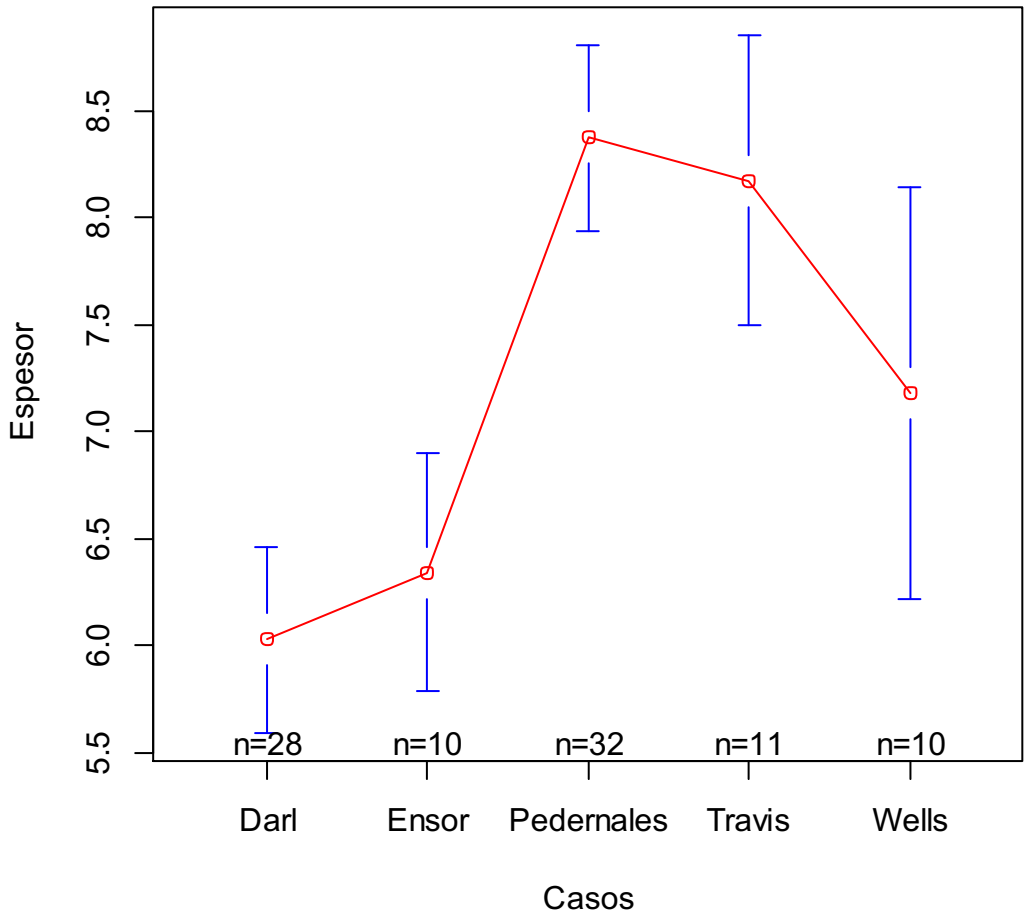


Gráfico de medias (círculos rojos) e intervalos de confianza del 95% (azul) para cada nivel del factor Casos.

Este gráfico es muy útil, ya que nos ayuda a entender mejor el comportamiento de los datos y está vinculado al procedimiento de *test* de hipótesis sobre las medias. Aquí vemos el tamaño de cada muestra y el intervalo de confianza del 95% en torno a ella para cada caso. Todo parece indicar que Darl y Ensor son semejantes (ya que solapan ampliamente su IC) y que ambos son diferentes a Pedernales, Travis y Wells (como ocurría con el Largo). Por otro lado Perdernales y Travis parecen ser bastante semejantes, aunque diferentes de Wells, que muestra también una amplia varianza en torno a la

media. Esto sin embargo, no es un test de hipótesis, ajustemos mejor el test de ANOVA:

```
>anova1<-aov(Espesor~Casos, data=puntas)
>anova1
Call:
   aov(formula = Espesor ~ Casos, data = puntas)
Terms:
                    Casos Residuals
Sum of Squares   99.94671 111.09901
Deg. of Freedom         4        86

Residual standard error: 1.136595
Estimated effects may be unbalanced
```

Al tipear el nombre del objeto nos devuelve información básica del modelo, como sus grados de libertad (uno por cada clase y 86 de los casos) y la suma de cuadrados de los casos y de los residuales.

Para obtener más información utilizamos el comando summary(), como en casos anteriores.

```
>summary(anova1)
            Df Sum Sq Mean Sq F value   Pr(>F)
Casos        4  99.95  24.987   19.34 2.22e-11 ***
Residuals   86 111.10   1.292
---
Signif. codes:
0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

El test nos devuelve el nivel de significación sobre el test F Fisher, éste es equivalente a la razón entre las medias de cuadrados entre grupos y dentro de los grupos. Cuanto mayor

es la diferencia entre grupos en relación a la variación dentro de los grupos, mayor será el valor de F. El resultado indica que al menos uno de las clases, es diferente a las demás.

Otra posibilidad, como se mencionó es ajustar primero un modelo lineal y luego extraer la tabla de ANOVA.

Análisis de la varianza a partir de un modelo de regresión lineal:

```
>anova2<-lm(Espesor~Casos,data=puntas)#ajustaremos primero un modelo lineal
>anova2
Call:
lm(formula = Espesor ~ Casos, data = puntas)
Coefficients:
    (Intercept)       CasosEnsor  CasosPedernales
          6.025            0.315            2.347
    CasosTravis       CasosWells
          2.148            1.155
```

Nos devuelve los coeficientes como en la regresión por mínimos cuadrados, veamos summary().

```
>summary(anova2)
Call:
lm(formula = Espesor ~ Casos, data = puntas)
Residuals:
    Min      1Q  Median      3Q     Max
-2.0250 -0.8484 -0.1400  0.8250  2.3281

Coefficients:
                Estimate Std. Error t value Pr(>|t|)
```

```
(Intercept)       6.0250     0.2148  28.050  < 2e-16 ***
CasosEnsor        0.3150     0.4187   0.752  0.45392
CasosPedernales   2.3469     0.2941   7.979 5.83e-12 ***
CasosTravis       2.1477     0.4044   5.310 8.46e-07 ***
CasosWells        1.1550     0.4187   2.758  0.00709 **
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 1.137 on 86 degrees of freedom
Multiple R-squared:  0.4736,    Adjusted R-squared:  0.4491
F-statistic: 19.34 on 4 and 86 DF,  p-value: 2.223e-11.
```

El resultado nos muestra que Pedernales, Travis y Wells poseen diferencias significativas. Como realizamos una regresión, obtenemos el valor de R^2 ajustado (para la cantidad de niveles) y sin ajustar. El ajuste se realiza porque R^2 tiende a aumentar cuanto más niveles o variables se utilicen, lo que incrementa sesgadamente la varianza explicada. Por otro lado, nótese que falta una de las categorías (Darl) esto es debido a que por defecto, el programa toma el primer factor como medida de comparación, asumiendo que su parámetro es 0.

Obtengamos ahora la tabla de anova, usando el comando anova():

```
>anova(anova2)
Analysis of Variance Table
Response: Espesor
          Df  Sum Sq Mean Sq F value    Pr(>F)
Casos      4  99.947 24.9867  19.342 2.223e-11 ***
Residuals 86 111.099  1.2918
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Como vemos, el resultado general es idéntico al anterior lo que demuestra además, que el principio subyacente en ambos procedimientos es semejante.

A esta altura del análisis, sabemos que los datos poseen valores medios significativamente diferentes y por el modelo lineal, cuáles son los casos significativos al compararlos con el primero de ellos como referencia. Pero cuáles casos son diferentes entre sí?. Para determinarlo debemos utilizar un test post-hoc. El más conocido es el test de Tukey de pares. Emplearemos la función TukeyHSD().

Comparación de a pares: 1er caso

```
>TukeyHSD(anova1)
  Tukey multiple comparisons of means
    95% family-wise confidence level

Fit: aov(formula = Espesor ~ Casos, data = puntas)

$Casos
                       diff         lwr         upr     p adj
Ensor-Darl          0.3150000 -0.85174905  1.48174905 0.9432848
Pedernales-Darl     2.3468750  1.52730455  3.16644545 0.0000000
Travis-Darl         2.1477273  1.02073324  3.27472131 0.0000083
Wells-Darl          1.1550000 -0.01174905  2.32174905 0.0537043
Pedernales-Ensor    2.0318750  0.88447637  3.17927363 0.0000377
Travis-Ensor        1.8327273  0.44891245  3.21654209 0.0035137
Wells-Ensor         0.8400000 -0.57637962  2.25637962 0.4685628
Travis-Pedernales -0.1991477 -1.30609662  0.90780116 0.9870315
Wells-Pedernales  -1.1918750 -2.33927363 -0.04447637 0.0377204
Wells-Travis      -0.9927273 -2.37654209  0.39108755 0.2752964
```

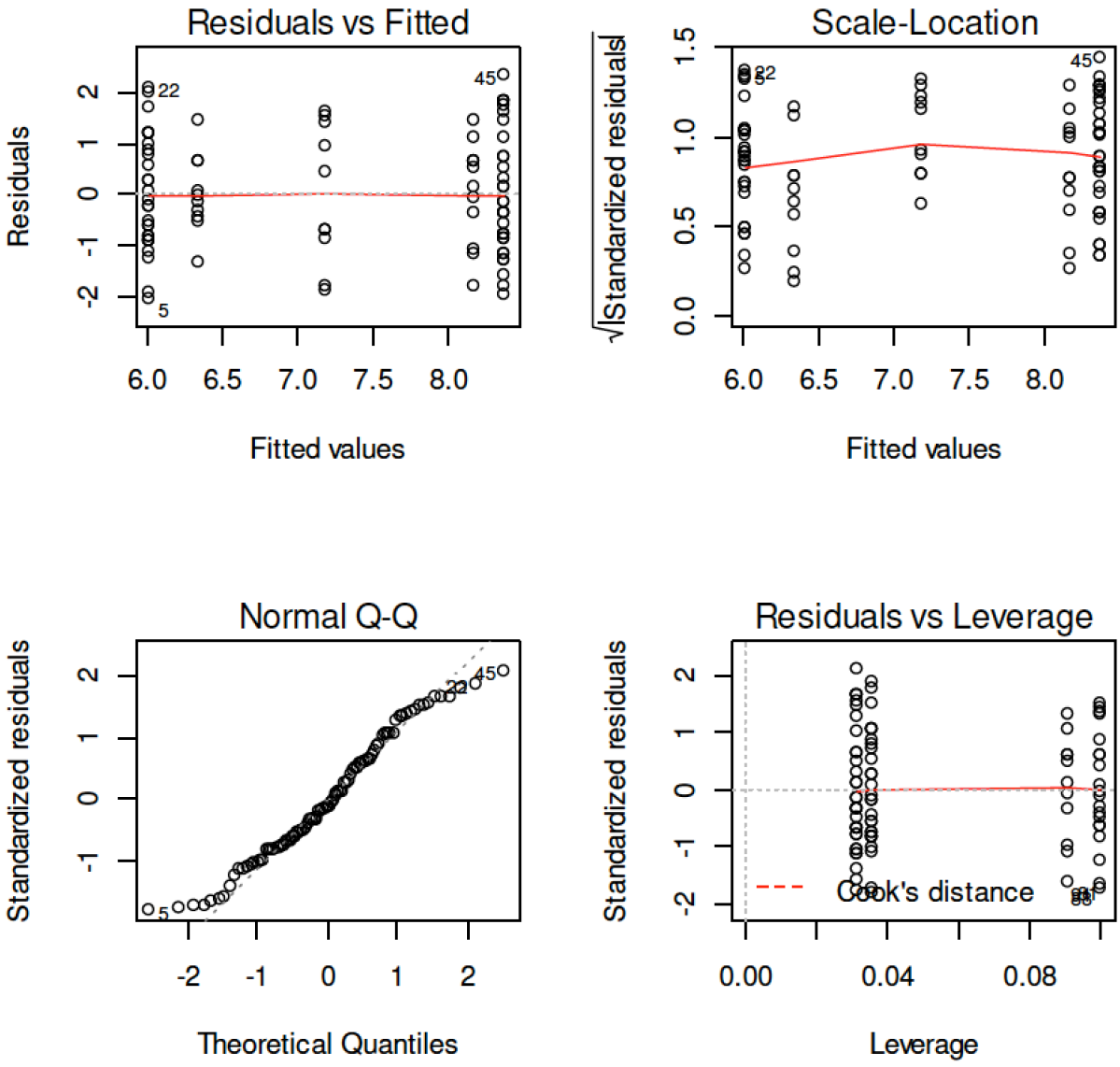
El resultado nos muestra la magnitud de la diferencia entre pares de casos, los intervalos de confianza y la significación de cada una de las comparaciones. Como vemos, no hay diferencias significativas entre Darl y Ensor, mientras que este par es diferente a los demás, con excepción de Wells. Si volvemos al gráfico de la página 122, Wells posee un límite inferior de su IC que se solapa con las medias de ambos, especialmente de Ensor. Pedernales y Travis, tal como se observó en el mismo gráfico, son semejantes entre sí y diferentes a las antes mencionadas. Wells posee además, una amplia varianza y en su IC superior, se solapa con la media de Travis. Entonces, es posible decir que ambos procedimientos, descriptivo e inferencial, se complementan perfectamente en este caso.

Control del modelo ajustado:

Además de los procedimientos gráficos, una práctica común es controlar los supuestos del modelo luego de su ajuste mediante *test* de hipótesis sobre los residuos, ya que se espera que éstos se distribuyan normalmente. El procedimiento más sencillo es realizar un test de Shapiro-Wilk contra la H0 que sostiene que éstos se distribuyen normalmente.

Una primera aproximación constituye, graficar el ajuste del modelo mediante el comando plot() y el nombre del objeto.

```
>par(mfrow=c(2,2))#armamos un marco para los gráficos de ajuste, ya que son 4.
>plot(anova1)
```



Gráficos de ajuste del modelo ANOVA.

Los resultados sugieren una distribución relativamente homogénea de los residuos, aunque como se ve en el gráfico de cuantiles, valores bajos y altos de residuos estandarizados tienden a apartarse de lo esperado bajo la distribución normal. La distancia de Cook y la comparación explicados *vs* ajustados también detectan algunos pocos casos con residuos muy altos, positivos o negativos (22, 45, 5).

Análisis numérico de los residuos:

Veamos primero, mediante el commando names() que elementos se encuentran dentro del resultado de anova1

```
>names(anova1)
 [1] "coefficients"  "residuals"     "effects"
 [4] "rank"          "fitted.values" "assign"
 [7] "qr"            "df.residual"   "contrasts"
[10] "xlevels"       "call"          "terms"
[13] "model"
```

Podemos extraer solo uno de ellos para analizarlo, en este caso el elemento anova1$residuals

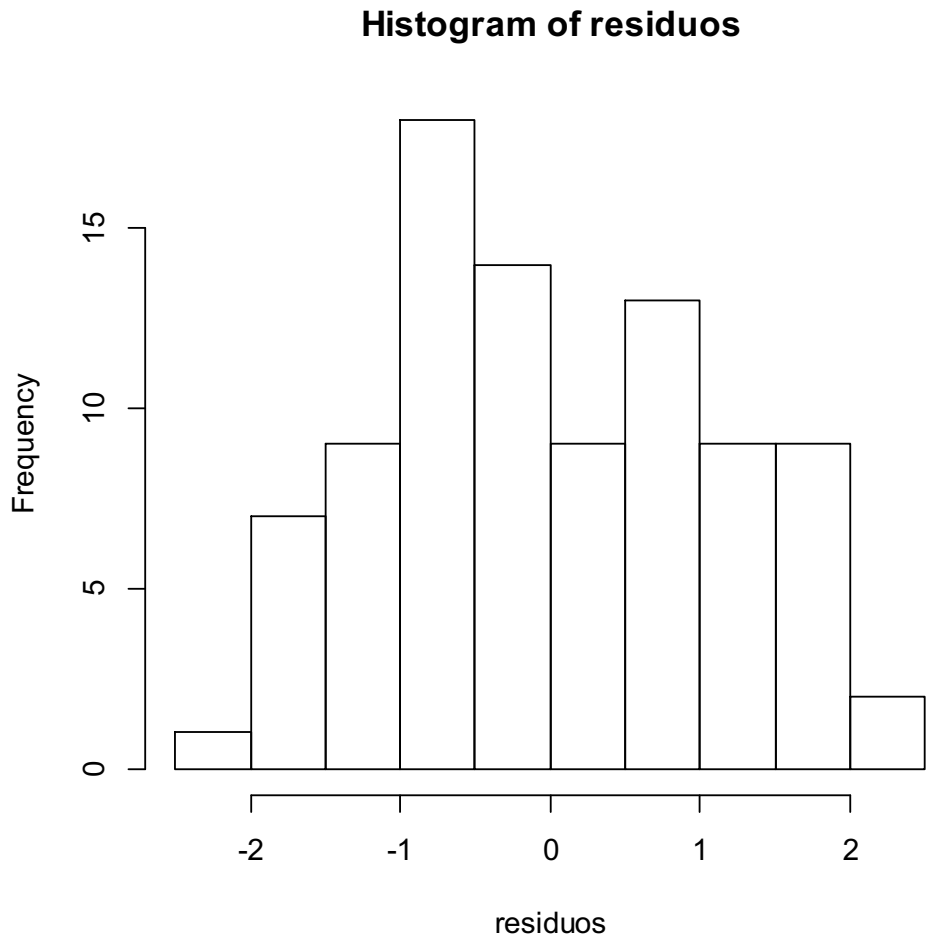
```
>residuos<-anova1$res#con tipear solo la primera parte del nombre, R reconoce el objeto.
```

```
>head(residuos)
     1      2      3      4      5      6
-0.225 -0.225  0.075  0.275 -2.025 -1.925
```

Los casos positivos mayores a 0 son sobreestimaciones del parámetro, mientras que los menores, lo contrario. Los errores deberían distribuirse aleatoriamente en torno a la media 0.

Podemos en principio, ver su forma:

```
>hist(anova1$res, xlab=residuos)
```



Histograma de los residuos.

Parece que la distribución es relativamente simétrica, pero para estar seguros, conduzcamos el test de Shapiro-Wilk.

```
>shapiro.test(residuos)
        Shapiro-Wilk normality test

data:  residuos
W = 0.97196, p-value = 0.04656
```

El valor del *test* señala un resultado marginalmente significativo, por lo que podemos pensar que si bien no es

perfecto, fue adecuado para nuestros datos. Como mencionamos anteriormente, es más importante que se cumpla el supuesto de homogeneidad de las varianzas en los casos estudiados, esto podemos controlarlo mediante el test de Barlett, (bartlett.test()).

```
>bartlett.test(Espesor~Casos, data=puntas)
        Bartlett test of homogeneity of variances

data:  Espesor by Casos
Bartlett's K-squared = 3.0928, df = 4, p-value
= 0.5424
```

Como vemos, no puede rechazarse la hipótesis de que las varianzas son semejantes, por lo que podemos considerar a nuestros datos y a nuestros resultados, si bien no perfectos, adecuados. Una posibilidad para mejorar el ajuste podría ser remover los datos extremos detectados en el test de Cook o en el gráfico de residuos *vs* predichos o transformar los datos (por ejemplo, aplicando el logaritmo), para homogeneizar la varianza de la variable, antes del análisis.

Por último, podemos optar por un método de permutaciones en casos de que los supuestos no se cumplan o que queramos tener una segunda validación de nuestro análisis. El paquete lmPerm() y aovp() realiza *test* basados en modelos lineales con distintos grados de complejidad. En nuestro caso, veamos los datos de Espesor para las diferentes clases.

3.4.4. Análisis de la varianza mediante permutaciones.

```
>library(lmPerm)
```

```
>permANOVA<-aovp(Espesor~Casos,perm= "Exact")#Exact produce la probabilidad de permutaciones del método exacto
[1] "Settings:  unique SS "
>permANOVA
Call:
   aovp(formula = Espesor ~ Casos, perm = "Exact")
Terms:
                    Casos Residuals
Sum of Squares   99.94671 111.09901
Deg. of Freedom         4        86

Residual standard error: 1.136595
Estimated effects may be unbalanced

>summary(permANOVA)
Component 1 :
            Df R Sum Sq R Mean Sq Iter  Pr(Prob)
Casos        4   99.947   24.9867 5000 < 2.2e-16 ***
Residuals   86  111.099    1.2918
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Vemos que el resultado es significativo como en ocasiones anteriores. Si queremos el valor de F, podemos obtenerlo realizando las permutaciones sin especificar el método:

```
>permANOVA<-aovp(Espesor~Casos,perm="")
>summary(permANOVA)
            Df Sum Sq Mean Sq F value   Pr(>F)
Casos        4  99.95  24.987   19.34 2.22e-11 ***
Residuals   86 111.10   1.292
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Vemos que las probabilidades de permutaciones y del *test* de Fisher son diferentes, lo que es esperable, aunque en los dos el nivel de probabilidad es muy pequeño, indicando el rechazo de la H0. Sin embargo, una opción más clásica a los *test* de permutaciones, siguen siendo los no paramétricos, que veremos a continuación.

3.5. Test no-paramétricos.

Tal como su nombre lo indica, en estos casos no existen parámetros que definan la distribución cuantitativa, como la media. En cambio, la forma de las distribuciones de las variables aleatorias sobre las que se emplean, son en principio, de forma desconocida. Estos procedimientos son apropiados tanto cuando no se cumplen los supuestos básicos de los *test* paramétricos, o por la naturaleza de la distribución estudiada (por ejemplo, de nivel nominal). Veremos alguno de los más empleados:

wilcox.test()	Wilcox o Mann-Withney *test*. Análogo al test de la *t*. Para distribuciones no normales de dos variables independientes donde se estima la mediana de los datos transformados a rangos.
kruskal.test()	Análogo a ANOVA pero para más de dos variables independientes de distribución no-normal.
chisq.test()	Dos o más variables categóricas expresadas en forma de frecuencias en una tabla de contingencia.
CMHtest()	Cochran-Mantel-Haenszel para datos ordinales.
binom.test()	Dos proporciones, donde una de ellas es un valor esperado que representa el parámetro poblacional.
fisher.test()	Dos variables categóricas expresadas en forma frecuencias en una tabla 2x2.

3.5.1. Dos variables: Mann-Whitney.

El test de Mann-Whitney U es un procedimiento no paramétrico que emplea los *rankings* o rango de valores en lugar de los valores originales. Para contrastar la H0 primeramente se ordenan juntas las muestras en orden decreciente y se le asigna un rango, si un valor se repite, se estima el promedio de los rangos y se le asigna a cada valor el mismo. Si no existen diferencias en los valores centrales de las muestras, la diferencia entre estos rangos es similar a la esperada por azar, por lo que los rangos de una variable, tenderán a ser similares a los rangos en la otra.

Primeramente armemos una matriz o data frame con alguna de las variables del archivo DartPoints(). Utilizaremos nuevamente espesor, cuya distribución y propiedades conocemos, pero como es un análisis bivariado sólo tomaremos dos clases:

```
>Esp<-data.frame(DartPoints$Name,DartPoints$Thickness)
>Esp2<-rbind(Esp[29:38,1:2], Esp[39:70,1:2])#Tomaremos Darl y
Endsor
>names(Esp2)<-c("Casos ", "Espesor ")
>str(Esp2)#controlamos que esté todo correcto
'data.frame':   42 obs. of  2 variables:
 $ Casos   : Factor w/ 5 levels "Darl","Ensor",..: 2 2 2 2 2 2
2 2 2 2 ...
 $ Espesor : num  6 6.4 5.9 5.8 6.2 7 5 6.3 7 7.8 ...
```

Podemos describir la variable numéricamente a partir de los factores como hicimos en acápites anteriores mediante el paquete psych()y la función describeBy().

```
>describeBy(Esp2$Espesor,Esp2$Casos)
$Darl
NULL
```

```
$Ensor
vars   n  mean    sd median trimmed   mad min max range skew
kurtosis
X1     1 10 6.34 0.78   6.25    6.33 0.59   5 7.8   2.8 0.21
-0.72
    se
X1 0.25
$Pedernales
vars   n  mean    sd median trimmed   mad min   max range skew
kurtosis
X1     1 32 8.37 1.21    8.1    8.35 1.33 6.4 10.7   4.3 0.32
-1.14
     se
X1 0.21
$Travis
NULL
$Wells
NULL

attr(,"call")
by.default(data = x, INDICES = group, FUN = describe, type =
type)
```

Vemos que tanto la media como la mediana de Pedernales son mayores que las de Darl, aún tomando en cuenta la media robusta a los datos extremos (trimmed mean). Llevemos adelante el *test*:

```
>attach(Esp2)
>wilcox.test(Espesor~Casos)
        Wilcoxon rank sum test with continuity correction
data:  Espesor by Casos
W = 19, p-value = 3.286e-05
alternative hypothesis: true location shift is not equal to 0
```

```
Warning message:
In wilcox.test.default(x = c(6, 6.4, 5.9, 5.8, 6.2, 7, 5, 6.3,
7,  :
  cannot compute exact p-value with ties
```

El resultado indica diferencias significativas entre la mediana de los rangos de ambos grupos. Podemos explorar también la alternativa a este *test* mediante permutaciones.

3.5.2. Wilcox_test mediante permutaciones.

```
>library(coin)
>wilcox_test(Espesor~Casos, distribution="exact")

        Exact Wilcoxon-Mann-Whitney Test
data:  Espesor by Casos (Ensor, Pedernales)
Z = -4.1674, p-value = 2.389e-06
alternative hypothesis: true mu is not equal to 0.
```

Se obtienen también resultados significativos, en este caso, el valor crítico del *test* proviene del estadístico Z, obtenido a partir de la distribución generada por los estadísticos obtenidos mediante permutación.

3.5.3. Más de dos niveles: Kruskal-Wallis.

En caso en que la muestra se componga por una variable cuantitativa con más de dos niveles, pero en la cual no se cumplen los supuestos de normalidad, o en casos en que las muestras sean de nivel menor al de razón; la prueba de Kruskal-Wallis es una de las más adecuadas. Es una extensión del *test* de Mann-Whitney a más de dos variables y es similar al test de ANOVA, pero emplea rangos y las medianas en vez de las varianzas y los promedios. Contrasta la H0 de medianas

semejantes entre grupos, aunque tiene menos potencia que ANOVA (ver capítulo siguiente).

Al igual que en el caso de ANOVA, volvamos a construir un archivo que contenga todos los niveles de la variable Espesor de las puntas de proyectil:

```
>detach(Esp2)
>puntas<-data.frame(DartPoints$Name,DartPoints$Thickness)
>names(puntas)<-c("Casos","Espesor")
>head(puntas)
  Casos Espesor
1  Darl     5.8
2  Darl     5.8
3  Darl     6.1
4  Darl     6.3
5  Darl     4.0
6  Darl     4.1
>kruskal.test(puntas$Espesor~puntas$Casos)
        Kruskal-Wallis rank sum test
data:  puntas$Espesor by puntas$Casos
Kruskal-Wallis chi-squared = 44.215, df = 4, p-value = 5.789e-
09
```

El test detecta diferencias significativas entre las distintas clases de puntas. Al igual que en ANOVA, podemos realizar un test post-hoc entre pares para establecer diferencias entre cada uno. Para esta comparación utilizaremos el paquete PMCMR()y la función posthoc.kruskal.nemenyi.test().

```
>require(PMCMR)
>posthoc.kruskal.nemenyi.test(puntas$Espesor,     puntas$Casos,
dist="Tukey")
        Pairwise comparisons using Tukey and Kramer (Nemenyi)
test with Tukey-Dist approximation for independent samples
data:  puntas$Espesor and puntas$Casos
```

```
           Darl    Ensor   Pedernales Travis
Ensor      0.99753 -       -          -
Pedernales 4.1e-08 0.00104 -          -
Travis     0.00034 0.01689 0.99991    -
Wells      0.20067 0.56348 0.25680    0.52295

P value adjustment method: none
Warning message:
In         posthoc.kruskal.nemenyi.test.default(puntas$Espesor,
puntas$Casos,  :
  Ties are present, p-values are not corrected.
```

La comparación de a pares muestra una matriz con los valores de significación del *test*, allí obtuvimos evidencia que soporta la hipótesis alternativa de que existen diferencias en los espesores de las puntas de proyectil entre la mayoría de las clases, excepto Wells que parece ocupar una posición intermedia entre ellas.

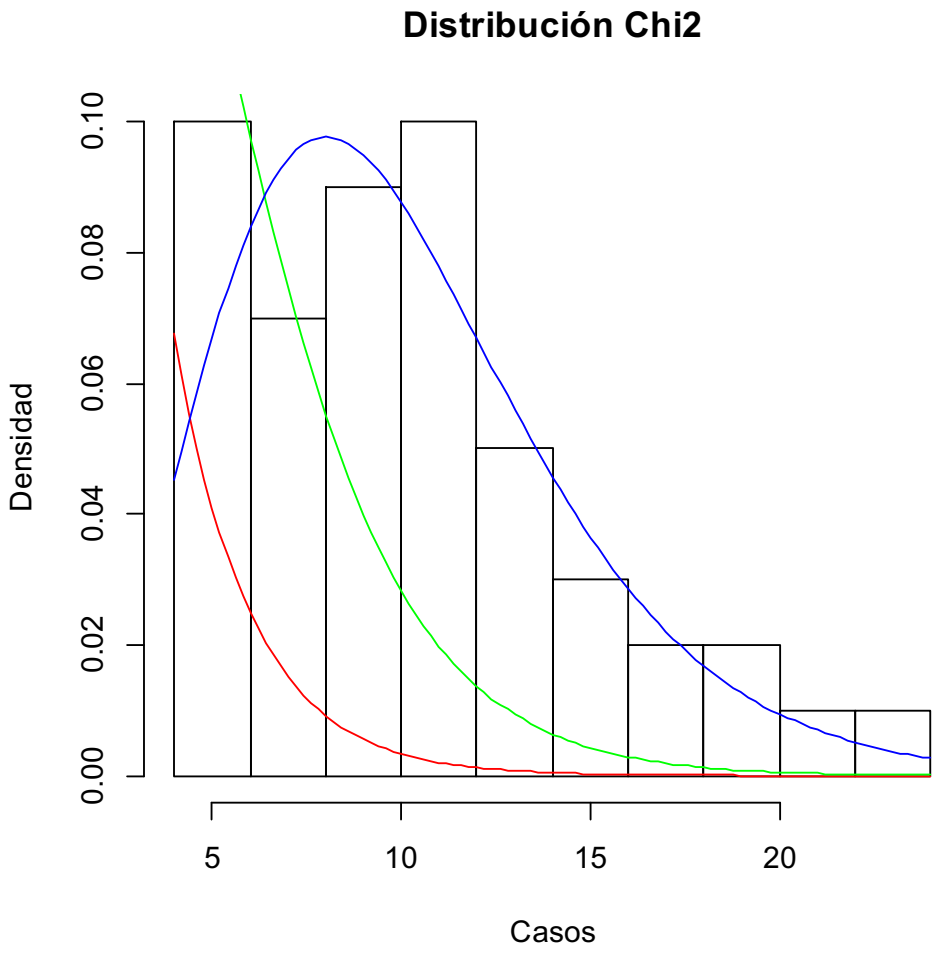
3.5.4. Test no-paramétricos: la distribución de chi^2.

Tal como lo mencionamos, este método permite la comparación de variables cualitativas nominales y ordinales. Se basa en la construcción de un modelo "nulo" de distribución aleatoria de los casos a través de las distintas categorías o niveles que los componen. Esta distribución nula (esperada) se compara luego con la distribución empírica (observada). La diferencia entre ambas distribuciones constituye el estadístico Chi^2 (también X^2 o ji^2).

Como esta es una de las distribuciones de probabilidad más importantes, veremos brevemente algunas de sus propiedades,

como su forma, que a diferencia de la distribución normal, es asimétrica hacia la derecha y posee siempre valores positivos, al igual que la distribución F. Mediante la función rchisq() podemos generar variables aleatorias que siguen esta distribución. La función dchisq() en cambio, genera distribuciones de densidad, y en este caso con valores predeterminados de grados de libertad (df), ya que la distribución de Chi^2 depende de las dimensiones de la tabla de contingencia.

```
>chi<- rchisq(50, 10)#50 casos con 10 grados de libertad equivalente a una tabla de 6 (-1gl)*3 (-1gl)
>hist(chi, prob=TRUE, main="Distribución Chi2", xlab="Casos", ylab="Densidad")
>curve(dchisq(x, df=2), col="red", add=TRUE)#no nos interesa la frecuencia, sólo simular la curva con gl predeterminados para superponerla al histograma.
>curve(dchisq(x, df=5), col="green", add=TRUE)
>curve(dchisq(x, df=10), col="blue", add=TRUE)
```



Histograma de la distribución de Chi^2 para 50 casos con 10gl y curvas de densidad para dos (rojo), cinco (verde) y diez (azul) gl respectivamente.

Como era de esperar, la curva de densidad para 10 gl es la que mejor se adapta al histograma que corresponde a números aleatorios distribuidos en una tabla de 6x3. A medida que el tamaño de la muestra se aproxima el estadístico de Chi^2 se aproxima a la distribución normal.

3.5.5. Test de chi^2.

Supongamos que tenemos un conjunto de datos categóricos, como por ejemplo, distintas variables codificadas en forma de

frecuencia de bienes, edades o grupos cronológicos tal como el caso de la base de datos EWBurials(). Nuestro interés puede ser determinar si estas frecuencias se distribuyen de forma desigual entre las distintas categorías, tal que sugiera la existencia de una correlación entre ellas.

Compararemos primero los grupos cronológicos, la edad y bienes presentes en las distintas tumbas, para ello podemos construir dos tablas de contingencia con estos datos:

```
>library(archdata)
>data(EWBurials)
>head(EWBurials)
>head(EWBurials)
```

	Group	North	West	Age	Sex	Direction	Looking	Goods
011	2	96.96	90.32	Young Adult	Male	42	283	Present
014	2	100.20	90.61	Young Adult	Male	28	272	Present
015	2	101.74	91.62	Old Adult	Male	350	219	Present
016a	2	101.00	90.47	Young Adult	Male	335	60	Absent
018	2	101.65	90.46	Old Adult	Male	3	86	Present
020	1	95.17	90.53	Young Adult	Male	142	21	Absent

```
>Tabla<-data.frame(EWBurials$Group,EWBurials$Age,
EWBurials$Goods)
>names(Tabla)<-c("Grupo","Edad","Bienes")
>Tabla1<-xtabs(~Edad+Bienes,data=Tabla)
>Tabla1#veamos la tabla 1
```

Edad	Bienes: Absent	Present
Child	1	1
Adolescent	2	1
Young Adult	8	11
Adult	1	2
Middle Adult	6	4
Old Adult	5	7

```
>Tabla2<-xtabs(~Grupo+Bienes,data=Tabla)
>Tabla2#veamos la tabla 2
     Bienes
Grupo Absent Present
    1      9       3
    2     14      23
```

Trabajaremos primero con la tabla uno y luego con la dos:

```
>summary(Tabla1)#la función summary() sobre una tabla de contingencia hecha con el comando xtabs() nos devuelve automáticamente el test de Chi².
Call: xtabs(formula = ~Edad + Bienes, data = Tabla)
Number of cases in table: 49
Number of factors: 2
Test for independence of all factors:
        Chisq = 1.6964, df = 5, p-value = 0.8894
        Chi-squared approximation may be incorrect
```

No hay diferencias significativas entre las frecuencias esperadas y las observadas. Hay que tener en cuenta que en nuestra tabla hay frecuencias muy bajas que probablemente afecten la performance del test de Chi^2, que es inexacto con frecuencias esperadas menores a 5 (por este motivo R nos devuelve este mensaje: "Chi-squared approximation may be incorrect").

Veamos ahora la tabla 2:

```
>summary(Tabla2)
Call: xtabs(formula = ~Grupo + Bienes, data = Tabla)
Number of cases in table: 49
Number of factors: 2
Test for independence of all factors:
        Chisq = 5.024, df = 1, p-value = 0.02499
```

En este caso obtenemos un resultado significativo, por lo que podemos inferir que la presencia-ausencia de bienes se distribuye desigualmente entre los dos grupos cronológicos.

Una forma de analizar en detalle el ajuste del *test* entre los distintos niveles en que está dividida la variable categórica es a través de los residuos, que son la diferencia entre las frecuencias observadas y las esperadas. Cuanto más grande el residuo, mayor es la diferencia en una categoría particular. Asimismo los residuos pueden representarse de distinta manera, si se transforman en relación a las frecuencias marginales de filas y columnas, se pueden interpretar como a los desvíos estándar de una distribución normal estandarizada.

Veamos el primer caso:

```
>library(vcdExtra)
>t1<-assoc(Tabla2,  shade=T)#La opción shade, colorea de
```

diferente manera los residuos mayores (negativos o positivos) a 1.96 desvíos estándar y que indican el rechazo de H0 en una celda en particular.

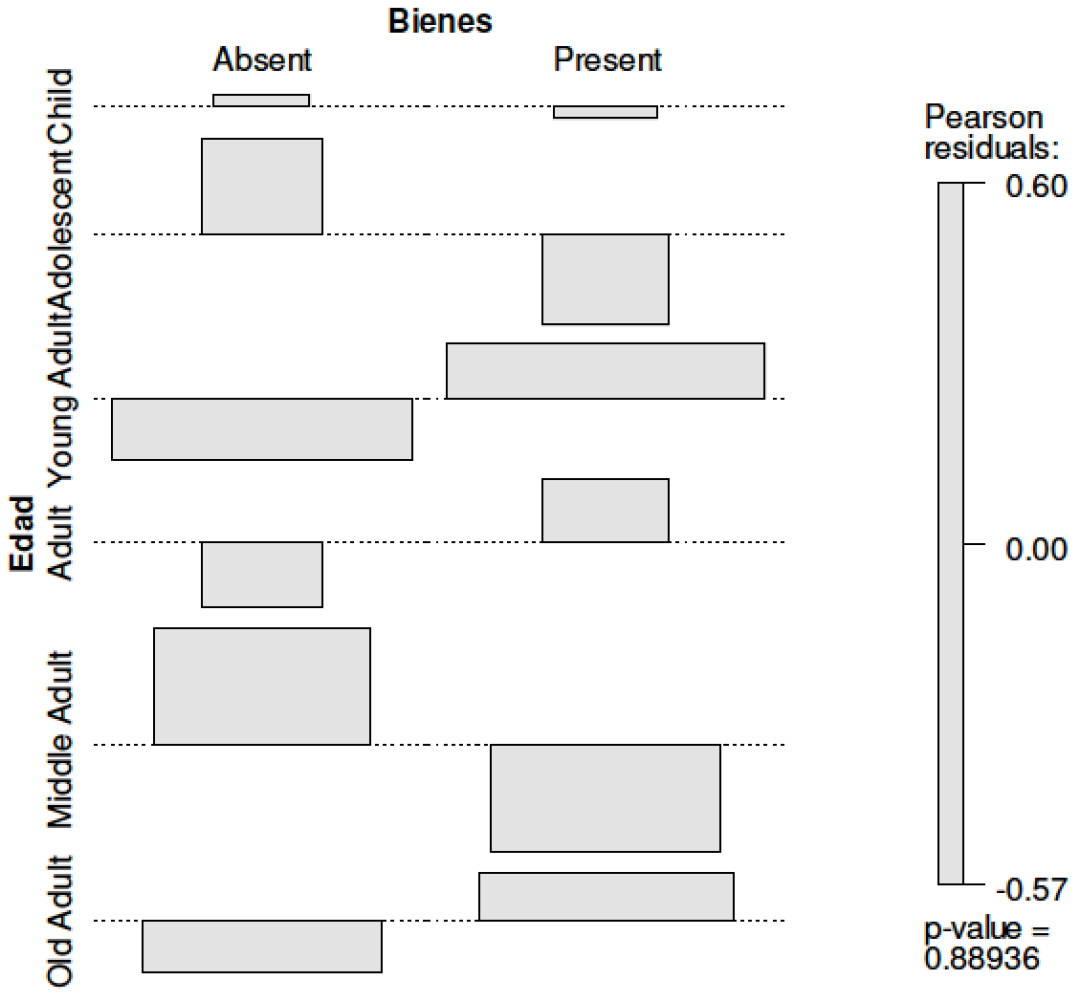


Gráfico de asociación entre edad y presencia-ausencia de bienes. La posición de la columna (inferior o superior respecto a la línea punteada), indica si el residuo es positivo o negativo y su tamaño (medido como unidades de desviación estándar). El ancho de cada una indica la frecuencia observada.

El segundo caso:

```
>t2<-assoc(Tabla2, shade=T)
```

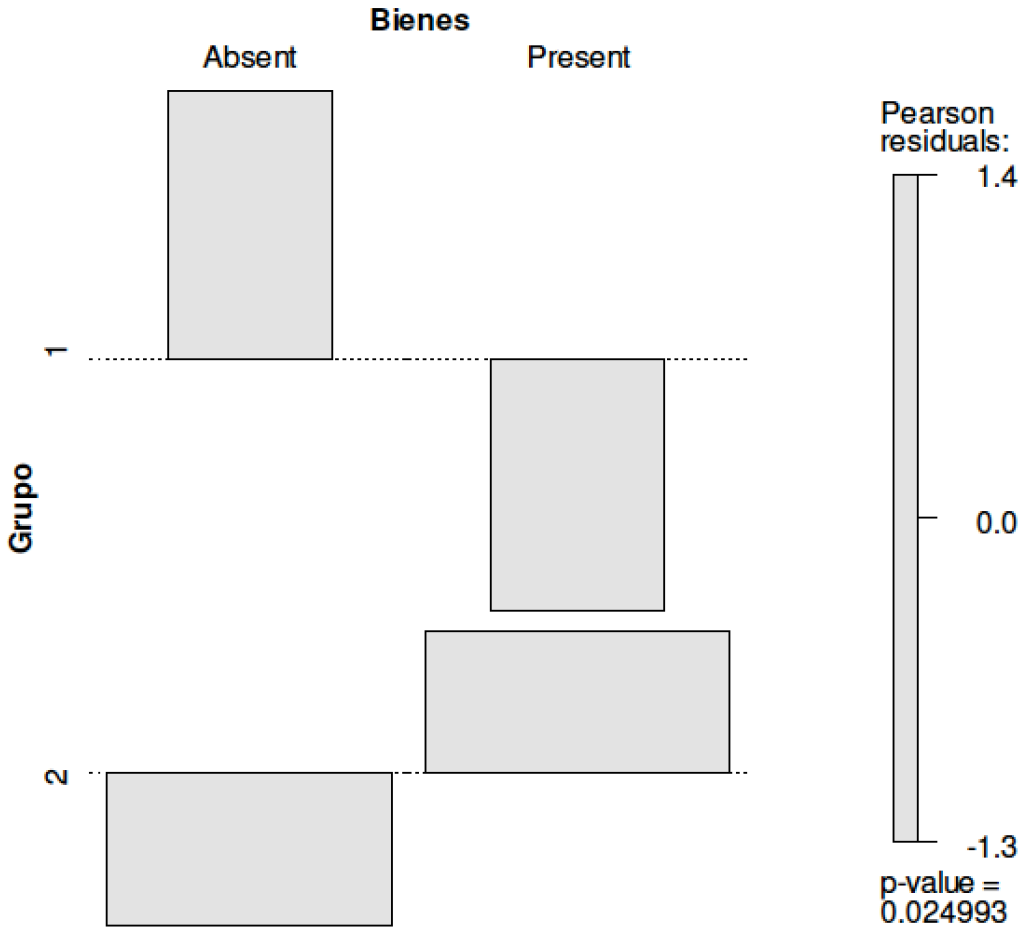


Gráfico de asociación entre grupo cronológico y presencia-ausencia de bienes.

Otra alternativa es realizar un gráfico de mosaico con la función mosaic() del mismo paquete (vcdExtra()).

```
>mosaic(Tabla2, shade=T)
```

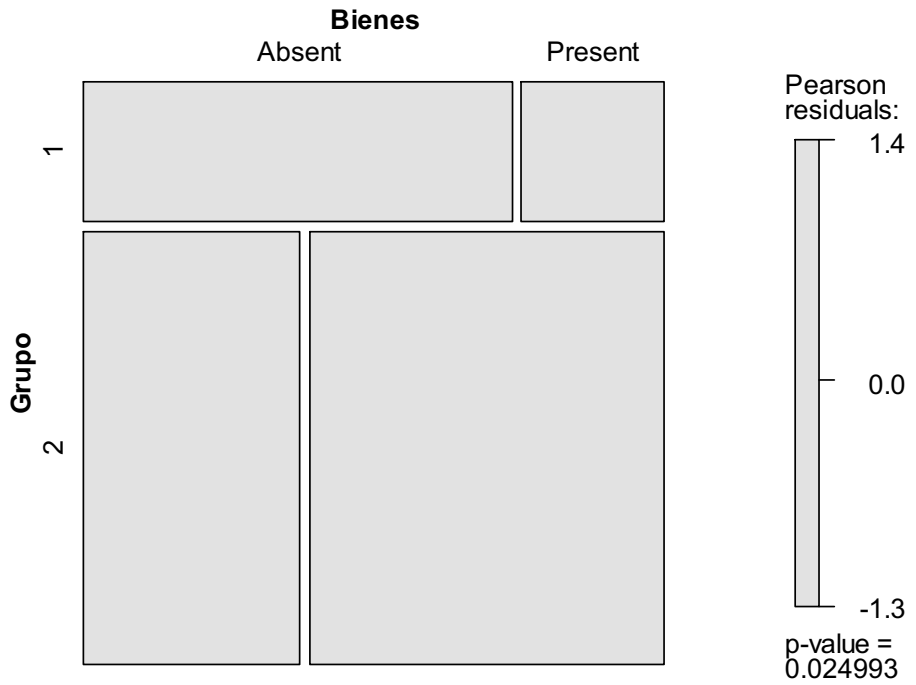


Gráfico de mosaico entre grupo cronológico y presencia-ausencia de bienes. El tamaño de cada mosaico depende del residuo y la frecuencia de cada celda.

En la segunda tabla, si bien se obtuvieron resultados significativos, no es posible determinar si existe una celda en particular (con un residuo igual/mayor o menor a 1.96) que posea un gran residuo; indicando su rol específico en la determinación del valor crítico de Chi^2 para nuestro nivel de significación. Sin embargo es posible observar que en el primer grupo temporal parece existir una muy baja proporción de bienes, (independientemente del tamaño de la muestra).

Tal como lo mencionamos, Chi^2 posee una aproximación a la distribución de probabilidad que es inexacta cuando los valores esperados son menores a 5, lo que ocurre en muestras pequeñas. Una alternativa conveniente en estos casos es la aleatorización, o en el caso de tablas 2x2, del test exacto de Fisher que es siempre más preciso, y puede estimarse con frecuencias bajas o hasta nulas.

Veamos la aleatorización para el primer caso:

```
>chisq.test(Tabla1, simulate.p.value =T, B = 10000)#Utiliza
```

Monte Carlo para generar una distribución asintótica la distribución de Chi^2, en este caso a partir de 10000 simulaciones (B=10000). Si simulate.p.value =F realiza el test habitual.

```
        Pearson's Chi-squared test with simulated p-value
(based on 10000
        replicates)

data:  Tabla1
X-squared = 1.6964, df = NA, p-value = 0.9042
```

Para el segundo caso:

```
>chisq.test(Tabla2, simulate.p.value =T,B = 10000)
        Pearson's Chi-squared test with simulated p-value
(based on 10000
        replicates)

data:  Tabla2
X-squared = 5.0243, df = NA, p-value = 0.0455.
```

En ambos casos el resultado es muy similar al que se obtiene sin permutaciones. Tal como se observa para el segundo *test*, el resultado es significativo aunque cercano al umbral de significación. Esto se debe al hecho de que los *test* basados en este procedimiento son más conservadores (es más difícil rechazar H0).

3.5.6. Test de Cochran-Mantel-Haenszel para datos ordinales.

Si bien chi^2 es de gran utilidad para datos categóricos, existen alternativas más apropiadas cuando el nivel de medición es ordinal, como el test Cochran-Mantel-Haenszel. El test toma en cuenta el orden en que se presentan las distintas categorías realizando un contraste al componente lineal entre las diferencias ordenadas entre filas y columnas. El *test* puede emplearse también combinando variables cuantitativas con ordinales, en donde funciona de manera similar al ANOVA de un factor que vimos en la página 124. El paquete vcdExtra(), realiza este test a través de la función CMHtest().

Utilicemos nuevamente los datos de presencia-ausencia de bienes y las categorías de edad como variable ordinal. La pregunta aquí es si es posible que la presencia-ausencia de bienes se distribuye diferencialmente de acuerdo a los distintos niveles de la variable edad:

```
>library(vcdExtra)
>data(EWBurials)
>Tabla<-data.frame(EWBurials$Age, EWBurials$Goods)
>names(Tabla)<-c("Edad","Bienes")
>Tabla2<-xtabs(~Edad+Bienes,data=Tabla)
>Tabla2
              Bienes
Edad           Absent Present
  Child             1       1
  Adolescent        2       1
  Young Adult       8      11
  Adult             1       2
  Middle Adult      6       4
  Old Adult         5       7

>CMHtest(Tabla2)
Cochran-Mantel-Haenszel Statistics for Edad by Bienes
```

```
                  AltHypothesis     Chisq Df    Prob
cor         Nonzero correlation 0.0061343  1 0.93757#1
rmeans   Row mean scores differ 1.6617497  5 0.89368#2
cmeans   Col mean scores differ 0.0061343  1 0.93757#3
general     General association 1.6617497  5 0.89368#4
```

El resultado de los análisis no es significativo para estas dos variables. El resumen del *test* devuelve valores de probabilidad para la H0 de ausencia correlación entre celdas y columnas (1), para la H0 de la falta de asociación entre el puntaje promedio de filas (2) y columnas (3) y el valor general del test (4).

Es importante tener en cuenta que también podríamos aplicar este u otros *test* sobre datos categóricos sin realizar previamente la tabla de contingencia, utilizando el operador (~). Por ejemplo, contrastemos la H0 sobre la distribución diferencial de edades entre grupos cronológicos, directamente de la base EWBurials():

```
>CMHtest(Group~Age, data=EWBurials)
Cochran-Mantel-Haenszel Statistics for Age by Group

                  AltHypothesis   Chisq Df    Prob
cor         Nonzero correlation 0.52877  1 0.46713
rmeans   Row mean scores differ 3.10773  5 0.68338
cmeans   Col mean scores differ 0.52877  1 0.46713
general     General association 3.10773  5 0.68338
```

Nuevamente obtenemos resultados nulos para los distintos tipos de asociación posibles entre filas y columnas.

3.5.7. Test exacto de Fisher.

La probabilidad exacta de observar un conjunto concreto de frecuencias a, b, c y d en una tabla 2 x 2 cuando se asume independencia y los totales de filas y columnas se consideran fijas, viene dada por la distribución hipergeométrica. Recordemos que test "exacto" se denomina así porque se calculan todas las posibles combinaciones para los valores de una tabla de 2x2, dejando los totales de filas y columnas fijos (a+b), (c+d), (a+c) y (b+d).

Se calculan de esta manera todas las probabilidades posibles para la tabla de contingencia, dados los totales observados. Se estima así la diferencia observada entre categorías es mayor o igual a lo que se sostiene bajo la H0 (hipótesis de independencia). Este *test* es más apropiado que el de Chi^2 cuando las frecuencias esperadas son menores a 5 y la tabla de contingencia tiene 2x2 dimensiones.

Volvamos sobre los datos de la Tabla 1 y 2:

```
>fisher.test(Tabla1)
        Fisher's Exact Test for Count Data

data:  Tabla1
p-value = 0.9068
alternative hypothesis: two.sided

>fisher.test(Tabla2)
        Fisher's Exact Test for Count Data

data:  Tabla2
p-value = 0.04405
alternative hypothesis: true odds ratio is not equal to 1
95 percent confidence interval:
  0.9741362 32.0348896
```

```
sample estimates:
odds ratio
  4.764757.
```

Vemos que el *test* exacto de Fisher, da valores similares en el primer caso al test de Chi^2 y un resultado similar a la simulación en el segundo. Se puede tomar entonces este segundo resultado como más sólido que en el caso del de Chi^2 donde los supuestos básicos no se cumplían.

3.5.8. Test binomial para proporciones.

El *test* binomial es de una gran utilidad en casos en donde el problema se plantea en función de sólo dos posibles resultados: presencia o ausencia, éxito o fracaso. Este procedimiento compara la proporción de ambos posibles resultados en función a una hipótesis nula que plantea que, en general, ambos son igualmente probables. El ejemplo clásico es el del lanzamiento de una moneda, donde cara o seca poseen una probabilidad semejante (0.5). Es decir, la mitad de las veces la moneda caerá de uno u otro lado. Sin embargo, esta hipótesis nula puede ser más "informada", por ejemplo, si se conoce una proporción determinada en que aparece un ítem arqueológico a nivel regional o en determinado período de tiempo, podemos comparar esta proporción con la que se observa en nuestra área de trabajo y así determinar la existencia de un patrón diferente o no.

En R, la función básica es binom.test(). Donde se debe consignar primero la frecuencia observada de un ítem, el total de observaciones y la proporción poblacional. Por defecto, el test se realiza con un nivel de significación del 0.05, pero puede modificarse.

Por ejemplo, en el área costera del norte de Patagonia, Argentina, un sitio arqueológico denominado "Punta Odriozola" se caracteriza por una alta diversidad y redundancia ocupacional. En éste, la proporción entre lascas sin corteza/con corteza es de 0.86/0.14, es decir un 86% más de lascas sin reserva corteza que con corteza (Cardillo et al MS). Esta proporción es empleada, junto con otros indicadores, como una medida de la intensidad de reducción lítica en un sitio o un área determinada. Sabemos por investigaciones previas que el promedio de la proporción regional es de 0.78. ¿Es la proporción de lascas sin corteza en este sitio arqueológico significativamente mayor a la esperada para toda el área?

La frecuencia de lascas sin reserva de corteza en el sitio es de 162, con corteza 27, y la proporción regional es p=0.78, con estos valores podemos llevar adelante el *test*.

```
>binom.test(162,189,p=0.78)#lascas sin corteza, total de
lascas, y proporción esperada bajo la H0
        Exact binomial test
Exact binomial test
data:  162 and 189
number of successes = 162, number of trials = 189, p-value =
6.485e-07
alternative hypothesis: true probability of success is not
equal to 0.7
95 percent confidence interval:
 0.7989965 0.9036981
sample estimates:
probability of success
             0.8571429
```

El resultado indica que la proporción de 0.86 está más alejada de 0.78 de lo esperado por azar con un intervalo del

95%, que va de 0.79 a 0.90 (que por lo tanto no incluye al parámetro poblacional 0.78), (ver también VanPool y Leonard. 2010).

Cuarta parte

4. Estimación tamaño de la muestra y test de potencia.

Está demostrado que el incremento del tamaño de la muestra mejora las propiedades de ésta, tanto al disminuir la heterogeneidad, como al incrementar su representatividad con respecto a la población de interés a la vez que aumenta la performance (potencia) de los *test* de hipótesis.

4.1. Estimación del tamaño de la muestra para determinar una proporción poblacional.

El tamaño mínimo que debe tener una muestra para cumplir con suficiencia estos requisitos puede estimarse de distintas maneras. El primer problema que nos podemos plantear es que tamaño de muestra necesitamos para analizar con una confiabilidad predeterminada (digamos del 95%) una proporción poblacional. Por ejemplo, la frecuencia que necesitamos en un muestreo, de una clase determinada de artefacto para su estudio. Esta estimación puede hacerse mediante la distribución estandarizada de Z, que tiende a la normalidad (al estudiar una proporción poblacional) al incrementarse el tamaño de la muestra. En este caso las opciones son sólo dos: presencia-ausencia de dicho artefacto.

Por ejemplo, en sitios arqueológicos la proporción de instrumentos en relación a lascas y desechos de talla es bastante baja, en muchos casos menor al 10%. Esto es equivalente a una proporción poblacional de 0.01 (10/100). Este dato hay que poseerlo a priori y puede provenir de un muestreo piloto, de información procedente de la bibliografía o de colecciones previas.

Entonces, ¿qué tamaño de muestra debería yo tener para generar una estimación de esta proporción poblacional con una confianza estadística del 95%?. Se puede aplicar la siguiente fórmula (ver también Shennan 1992):

z^2*p*(1–p) / E^2

Donde Z es valor de Z para una confianza determinada (en este caso 1.96 para el 95%, E el error tolerado (0.05) y p la proporción poblacional que en este caso estimamos en 0.1).

Creamos 3 objetos:

```
>Z=1.96
>p=0.1
>E=0.05
>Z^2 * p * (1-p) / E^2
[1] 138.2976
```

Necesitamos al menos 138 artefactos tomados al azar para establecer con un 95% de confianza que la proporción poblacional de instrumentos es del 10%.

Nótese que cuando la proporción que queremos establecer se acerca a un 50%, se incrementa el tamaño de la muestra para determinar con una confiabilidad predeterminada su proporción, lo mismo ocurre si disminuimos el nivel de error tolerado.

```
>Z=1.96
>E=0.05
>p=0.5
>Z^2 * p * (1-p) / E^2
[1] 384.16
```

Por ejemplo, al reducir el error a la mitad, el tamaño de muestra necesario se incrementa.

```
>Z=1.96
>p=0.1
>E=0.025
>Z^2 * p * (1-p) / E^2
[1] 553.1904
```

Como se basa en parámetros que no conocemos con certeza, esta función sirve más bien como una guía que es útil en la medida que nos permite establecer un tamaño mínimo para un muestreo.

4.1.2. Establecer una media poblacional con una confianza determinada.

Algo similar podemos hacer para establecer una media con una confianza y nivel de error predeterminado. Aquí también dependemos de información previa, por ejemplo al querer establecer el tamaño promedio de huesos de guanaco en un sitio, necesitaríamos información sobre el desvío poblacional (sigma), esto podemos hacerlo también con el mismo procedimiento que antes.

Z^2*sigma^2/ E^2

Donde Z, al igual que en el acápite anterior, es un valor predeterminado por el nivel de confiabilidad seleccionado (en general 95% que corresponde a Z=1.96). El sigma o desvío estándar poblacional, en este caso, supongamos que es de 2.5 mm. El error en este caso, es la medición en milímetros, podemos considerar que un error de cómo mucho 1 mm es altamente deseable. Entonces tenemos:

```
>Z=1.96
>E=1#mm
>sigma=2.5#mm
>Z^2*sigma^2/ E^2
[1] 24.01
```

Necesitamos medir 24 huesos de ese taxón para establecer con un nivel de confianza del 95% la media poblacional de medición que realicemos. De manera similar, el tamaño de la muestra aumentará si queremos disminuir el error o si el sigma se incrementa. Supongamos que éste es el doble:

```
>Z=1.96
>E=1#mm
>sigma=5#mm
>Z^2*sigma^2/ E^2
[1] 96.04
```

Al igual que en el caso anterior, este procedimiento puede ser una buena guía para determinar un tamaño muestral antes de realizar el análisis de los materiales. Podemos tomar una medida conservadora y tomar un tamaño de muestra que cubra

ampliamente los requisitos establecidos por este cálculo (ver también Shennan 1992).

4.2. Test de potencia.

Como último punto, veremos los *test* de potencia, que como la palabra lo indica, refieren a la fuerza o potencia que un *test* posee para detectar, con un nivel de significación y un tamaño de muestra determinado, diferencias entre dos muestras y rechazar la H0 (Cohen 1988).

La potencia (P) depende, entre otras cosas, del tamaño de la muestra y del efecto tamaño. Este último puede entenderse como el grado de diferencia que existe entre dos muestras, el cual, en principio, desconocemos. Por otro lado cuanto mayor es el tamaño de la muestra más potencia posee el *test* para detectar diferencias (efectos) cada vez más pequeños. Una muestra muy grande puede encontrar diferencias "significativas" que en realidad sean muy pequeñas y no tengan importancia en términos del problema que planteamos.

Por el contrario *test* basados en muestras pequeñas tienden a detectar sólo efectos o diferencias grandes (un efecto de 0.5 o más según Cohen 1988). Se considera que una potencia igual o mayor a 0.8 es adecuada para un *test* y una de 0.9 es óptima. Asimismo, Cohen (1988) propone, como guía tres efectos: pequeño (0.1), mediano (0.30) y tal como mencionamos; el grande (0.5). Evaluar que potencia tienen nuestros análisis para detectar patrones y rechazar la H0 es una forma de validar o simplemente reflexionar sobre nuestros resultados, especialmente cuando no estamos muy seguros de ellos.

Veremos las posibilidades que brinda, en este caso el paquete pwr() para dos de los métodos que hemos visto.

4.2.1. Test de potencia para el test de Chi^2.

Realicemos *test* de potencia para el análisis de chi^2 donde observamos diferencias estadísticamente significativas entre grupo y bienes para las tres magnitudes de efecto tamaño:

```
>pwr.chisq.test(w =0.1, N = 49, df = 2, sig.level =0.05, power
= )#efecto tamaño=w, tamaño de la muestra=N, grados de
libertad=df, nivel de significación=sig.level, la potencia la
queremos establecer, entonces la dejamos en blanco.
     Chi squared power calculation

              w = 0.1
              N = 49
             df = 2
      sig.level = 0.05
          power = 0.08875532

NOTE: N is the number of observations

>pwr.chisq.test(w =0.3, N = 49, df = 2, sig.level =0.05, power
= )
     Chi squared power calculation

              w = 0.3
              N = 49
             df = 2
      sig.level = 0.05
          power = 0.4524434

NOTE: N is the number of observations
>pwr.chisq.test(w =0.5, N = 49, df = 2, sig.level =0.05, power
= )
```

```
     Chi squared power calculation

              w = 0.5
              N = 49
             df = 2
      sig.level = 0.05
          power = 0.8898742

NOTE: N is the number of observations
```

Nuestro *test*, tiene entonces potencia correcta o relativamente alta (0.89) para detectar efectos grandes (0.5), con este tamaño de muestra. Con esta frecuencia de casos y el nivel de probabilidad elegido, es difícil que detectemos patrones que posean un efecto más reducido, por ejemplo que estén presentes en una parte más pequeña de la muestra.

También nos podemos plantear, por ejemplo, qué tamaño de muestra necesitaríamos con este nivel de probabilidad y una potencia predeterminada, digamos de 0.8, para detectar pequeñas diferencias (w=0.1) en la distribución de los bienes entre períodos, la fórmula entonces sería:

```
>pwr.chisq.test(w =0.1, N = , df = 2, sig.level =0.05, power
=0.80)
     Chi squared power calculation

              w = 0.1
              N = 963.4689
             df = 2
      sig.level = 0.05
          power = 0.8

NOTE: N is the number of observations
```

Necesitaríamos al menos 963 casos. Podemos tener cierta confianza ahora que de existir un número reducido de casos que presenten un patrón particular (por ejemplo un subconjunto de la población), no podríamos detectarlo hasta poseer una frecuencia mayor de ellos.

Ahora que sabemos la potencia de nuestro *test*, podemos preguntarnos también que efecto podemos detectar con una potencia de 0.8 y un tamaño de muestra de 49.

```
>pwr.chisq.test(w =, N = 49, df = 2, sig.level =0.05, power
=0.80)

     Chi squared power calculation

              w = 0.4434221
              N = 49
             df = 2
      sig.level = 0.05
          power = 0.8

NOTE: N is the number of observations
```

Podemos detectar un efecto más bien grande (de 0.44), con la potencia de 0.8 y el tamaño de la muestra existentes. Según nuestros intereses de investigación estos resultados pueden ser positivos o por el contrario, llevarnos a ampliar el tamaño de la muestra para alcanzar la máxima sensibilidad posible.

4.2.2. Test potencia para una correlación.

Para establecer el nivel de potencia de una correlación sólo necesitamos el tamaño de la muestra (n), el valor de correlación r de pearson (r), el nivel de significación del test (sig.level). En este caso, no poseemos un valor para el efecto tamaño deseado, sino un valor de correlación que viene ya dado por r.

Volviendo a la correlación entre variables métricas de puntas de proyectil (largo, ancho y espesor) de la página 78, podemos preguntarnos que potencia tenían estos *test* para la correlación largo-ancho (r=0.77), largo-espesor (r=0.59) y ancho-espesor (r=0.54):

```
>pwr.r.test(n =91 , r =0.77 , sig.level = 0.05, power = )
     approximate correlation power calculation (arctangh
transformation)
              n = 91
              r = 0.77
      sig.level = 0.05
          power = 1
    alternative = two.sided

>pwr.r.test(n =91 , r =0.59 , sig.level = 0.05, power = )
     approximate correlation power calculation (arctangh
transformation)
              n = 91
              r = 0.59
      sig.level = 0.05
          power = 0.9999952
    alternative = two.sided

>pwr.r.test(n =91 , r =0.54 , sig.level = .05, power = )
```

```
     approximate correlation power calculation (arctangh
transformation)
              n = 91
              r = 0.54
      sig.level = 0.05
          power = 0.9999058
    alternative = two.sided
```

Vemos que los niveles de correlación entre las variables métricas se detectaron casi con la máxima potencia posible en los tres casos, lo que concuerda con los resultados que obtuvimos previamente. Pero que hubiese pasado si la correlación que queremos detectar es mucho más baja, por ejemplo r=0.20:

```
>pwr.r.test(n =91 , r =0.20 , sig.level = .05, power = )

 approximate correlation power calculation (arctangh
transformation)

                  n = 91
                  r = 0.2
          sig.level = 0.05
              power = 0.4804186
        alternative = two.sided
```

Es fácil entender las implicancias de este resultado, detectar un patrón "significativo" en correlaciones tan bajas, con una potencia alta o al menos aceptable, requiere de muestras más grandes, pero ¿cuánto para este caso hipotético?. Al igual que en los ejemplos del acápite anterior, dejemos en

blanco, en este caso, el n y determinemos una potencia estándar de 0.8:

```
>pwr.r.test(n = , r =0.20 , sig.level = .05, power = 0.8)

approximate correlation power calculation (arctangh transformation)

              n = 193.0867

              r = 0.2

      sig.level = 0.05

          power = 0.8

    alternative = two.sided
```

Necesitamos al menos 193 casos. Esto también señala un problema potencial en los *test* de hipótesis, con muestras suficientemente grandes, podemos detectar diferencias "significativas" muy pequeñas, aunque éstas puedan ser espurias o no poseer valor real en términos de nuestro problema de investigación. Por este motivo, siempre debe haber una reflexión de la importancia en términos del problema planteado, de los *test* realizados; e interpretar éstos como un sostén provisorio de la Ha.

Se puede estimar la potencia para otros tipos de análisis estadísticos con el mismo criterio (ver Kabacoff 2011 o http://www.statmethods.net/stats/power.html del mismo autor), aunque variando algunos parámetros así como los valores de magnitud del efecto determinados por Cohen (1988).

Referencias:

Barceló, Juan A. 2008. Arqueología y Estadística. Introducción al estudio de la variabilidad de las evidencias arqueológicas. Bellaterra: Servicio de Publicaciones. Universitat Autónoma de Barcelona.

Carlson, D. L.; y G. Roth. 2016. archdata: Example Datasets from Archaeological Research. https://cran.r-project.org/web/packages/archdata/archdata.pdf.

Drennan, R. 1996. Statistics for Archaeologist. A Commonsense Approach. Plenum Press. New York-London.

Cohen, J. 1988. Statistical power analysis for the behavioral sciences. Segunda Edición. Elsevier, Amsterdam.

Hammer, Ø.; D.A.T. Harper y P. Ryan. 2001. PAST. Palaeontological Statistics software package for education and data analysis. Palaeontologia Electronica, 4(1), 1-9.

Kabacoff, R. 2011. R in Action: Data Analysis and Graphics with R. Manning.

Orton, C. 1988. Matemáticas para arqueólogos. Alianza, Madrid.

R Development Core Team. 2016. R: A language and environment for statistical computing. R Foundation for Statistical Computing, Vienna, Austria. ISBN 3-900051-07-0, URL https://www.r-project.org/.

Shennan, S. 1992. Arqueología cuantitativa. Crítica, Barcelona.

VanPool, T. L; y R.D. Leonard. 2010. Quantitative Analysis in Archaeology. Wiley.

Indice

V

ISBN 978-987-4934-02-4

Marcelo Cardillo 2017.
marcelo.cardillo@gmail.com