Análisis de Datos con el Sistema Estadístico R

Lic. Patricia Vásquez Sotero

Sesión 3

Descripción numérica y gráfica de datos

Contenidos

- Tratamiento y exploración de archivos.
- Estadística descriptiva.
- Distribuciones de Probabilidad. Generación de variables aleatorias.
- Tablas de frecuencias para variables categóricas.
- Medidas de resumen para variables continuas.
- Descripción gráfica de datos en R.
  - Gráficos para datos discretos.
  - Gráficos para datos continuos.
  - Representación de datos multivariantes.

EXLORACIÓN DE ARCHIVOS

Directorio de trabajo

```
getwd()                 # Muestra el directorio de trabajo (wd)
setwd(choose.dir())     # Selecciona el directorio de trabajo de forma interactiva
setwd("C:/myfolder/data")      # Cambia el wd
setwd("H:\\myfolder\\data")    # Cambia el wd
```

Crear directorios / descargar de Internet

```
dir()                   # Enumera los archivos en el directorio de trabajo
dir.create("C:/test")   # Crea la carpeta 'test' en la unidad 'c:'
setwd("C:/test")        # Cambia el directorio de trabajo a "c:/test"

# Descarga el archivo ‘students.xls’ de la internet.
download.file("http://dss.princeton.edu/training/students.xls",
      "C:/test/students.xls",
      method="auto", quiet=FALSE, mode = "wb", cacheOK = TRUE)

# Abril en excel y guardar archivo con extensión csv, luego leer en R
students=read.csv("C:/test/students.csv", header=TRUE, sep=",")
fix(students)
```

Las variables están organizadas por columnas y casos por filas.
Cada variable tiene más de un valor

	A	B	C	D	E	F	G	H	I	J	K	L	M	N
1	ID	Last Name	First Name	City	State	Gender	Student Status	Major	Country	Age	SAT	Average score (grade)	Height (in)	Newspaper readership (times/wk)
2	1	DOE01	JANE01	Los Angeles	California	Female	Graduate	Politics	US	30	2263	67	61	5
3	2	DOE02	JANE02	Sedona	Arizona	Female	Undergraduate	Math	US	19	2006	63	64	7
4	3	DOE16	JOE16	Elmira	New York	Male	Graduate	Math	US	26	2221	78	73	6
5	4	DOE17	JOE17	Lackawana	New York	Male	Graduate	Econ	US	33	1716	78	68	3
6	5	DOE18	JOE18	Defiance	Ohio	Male	Graduate	Econ	US	37	1701	65	71	6
7	6	DOE19	JOE19	Tel Aviv	Israel	Male	Graduate	Econ	Israel	25	1786	69	67	5
8	7	DOE20	JOE20	Cimax	North Carolina	Male	Graduate	Politics	US	39	1577	96	70	5
9	8	DOE03	JANE03	Liberal	Kansas	Female	Undergraduate	Politics	US	21	1842	87	62	5
10	9	DOE04	JANE04	Montreal	Canada	Female	Undergraduate	Math	Canada	18	1813	91	62	6
11	10	DOE05	JANE05	New York	New York	Female	Graduate	Math	US	33	2041	71	66	5
12	11	DOE21	JOE21	Hot Coffe	Mississippi	Male	Undergraduate	Econ	US	18	1787	82	67	3
13	12	DOE06	JANE06	Java	Virginia	Female	Graduate	Math	US	38	1513	79	59	5
14	13	DOE22	JOE22	Varna	Bulgaria	Male	Graduate	Politics	Bulgaria	30	1637	79	63	4
15	14	DOE23	JOE23	Moscow	Russia	Male	Graduate	Politics	Russia	30	1512	70	75	6
16	15	DOE07	JANE07	Drunkard Creek	New York	Female	Undergraduate	Math	US	21	1338	82	64	5
17	16	DOE08	JANE08	Mexican Hat	Utah	Female	Undergraduate	Econ	US	18	1821	80	63	3
18	17	DOE09	JANE09	Amsterdam	Holland	Female	Undergraduate	Math	Holland	19	1494	75	60	3
19	18	DOE10	JANE10	Mexico	Mexico	Female	Graduate	Politics	Mexico	31	2248	95	59	4
20	19	DOE11	JANE11	Caracas	Venezuela	Female	Undergraduate	Math	Venezuela	18	2252	92	68	5
21	20	DOE24	JOE24	San Juan	Puerto Rico	Male	Graduate	Politics	US	33	1923	95	63	7
22	21	DOE12	JANE12	Remote	Oregon	Female	Undergraduate	Econ	US	19	1727	67	62	7
23	22	DOE25	JOE25	New York	New York	Male	Undergraduate	Econ	US	21	1872	82	73	4
24	23	DOE13	JANE13	The X	Massachusetts	Female	Graduate	Politics	US	25	1767	89	68	6
25	24	DOE14	JANE14	Beijing	China	Female	Undergraduate	Math	China	18	1643	79	65	6
26	25	DOE26	JOE26	Stockholm	Sweden	Male	Undergraduate	Politics	Sweden	19	1919	88	64	4
27	26	DOE27	JOE27	Embarrass	Minnesota	Male	Graduate	Econ	US	28	1434	96	71	4
28	27	DOE28	JOE28	Intercourse	Pennsylvania	Male	Undergraduate	Math	US	20	2119	88	71	5
29	28	DOE15	JANE15	Loco	Oklahoma	Female	Undergraduate	Econ	US	20	2309	64	68	6
30	29	DOE29	JOE29	Buenos Aires	Argentina	Male	Graduate	Politics	Argentina	30	2279	85	72	3
31	30	DOE30	JOE30	Acme	Louisiana	Male	Undergraduate	Econ	US	19	1907	79	74	3

Instalación / carga de paquetes / programas escritos por el usuario

```
install.packages("ABC") # Instalará el paquete–-ABC--. Aparecerá una ventana
                          emergente, seleccione un sitio espejo para descargar
                          desde (el más cercano a donde se encuentre) y haga
                          clic en Aceptar.

library(ABC) # Carga el paquete --ABC-- en su espacio de trabajo

# Instalar los siguientes paquetes:

install.packages("foreign")
library(foreign)
install.packages("car")
install.packages("Hmisc")
install.packages("reshape")

http://cran.r-project.org/web/views/  # Lista completa de paquetes por área temática
```

Hacer un seguimiento de su trabajo

```
# Guardar los comandos utilizados durante la sesión
savehistory(file="mylog.Rhistory")

# Cargar los comandos usados en una sesión previa
loadhistory(file="mylog.Rhistory")

history() # Mostrar los últimos 25 comandos

# Puede leer mylog.Rhistory con un procesador de textos. Tener en cuenta que
    el archivo debe tener la extensión *.Rhistory
```

Leer datos de R

```
load("students.RData")

load("students.rda")

/* Agregar la ruta a los datos si es necesario*/

load("D:/quinua.Rdata")
```

Guardar datos en R

```
# Guardando todos los objetos en el archivo *.RData
save.image("mywork.RData")

# Guarda objectos selecionados
save(object1, object2, file="mywork.rda")

# Guardar los datos del archivo quinua y students
save(datos, file="D:/quinua.RData")
save(students,file="C:/test/students.rda")
```

Explorando datos

```
summary(quinua) # Proporciona estadísticas y frecuencias descriptivas básicas.
edit(quinua)    # Abre el editor de datos
str(quinua)     # Proporciona la estructura del conjunto de datos

length(quinua)  # Muestra el número de variables que contienen datos
names(quinua)   # Enumera variables en el conjunto de datos

head(quinua)    # Las primeras 6 filas de conjunto de datos
head(quinua, n=10)      # Primeras 10 filas de conjunto de datos
head(quinua, n= -10)    # Todas las filas, pero las últimas 10

tail(quinua)            # Últimas 6 filas
tail(quinua, n=10)      # Últimas 10 filas
tail(quinua, n= -10)    # Todas las filas, pero las primeras 10

quinua[1:10, ]          # Primeras 10 filas
quinua[1:10,1:3]        # Primeras 10 filas de datos de las 3 primeras variables
```

Explorando datos

R Console

```
> summary(quinua) # Proporciona estadísticas y frecuencias descriptivas básicas.
    IDENTIF          TGRANO         GERMINA         RES_MILD         ALTURA          UNIFORM         HAB_CRECC       LONG_PANJ        P_GRANO
 Min.   : 1.00   Min.   :1.0   Min.   :2.000   Min.   :1.000   Min.   :49.00   Min.   :1.000   Min.   :2.000   Min.   :13.00   Min.   : 33.7
 1st Qu.:10.75   1st Qu.:2.0   1st Qu.:3.750   1st Qu.:1.000   1st Qu.:60.00   1st Qu.:1.000   1st Qu.:3.000   1st Qu.:18.00   1st Qu.:102.2
 Median :20.50   Median :2.0   Median :4.500   Median :2.000   Median :64.00   Median :1.000   Median :3.000   Median :22.00   Median :147.8
 Mean   :20.50   Mean   :1.8   Mean   :4.175   Mean   :2.325   Mean   :65.25   Mean   :1.275   Mean   :3.225   Mean   :22.52   Mean   :146.3
 3rd Qu.:30.25   3rd Qu.:2.0   3rd Qu.:5.000   3rd Qu.:3.000   3rd Qu.:69.25   3rd Qu.:2.000   3rd Qu.:3.000   3rd Qu.:25.25   3rd Qu.:200.0
 Max.   :40.00   Max.   :2.0   Max.   :5.000   Max.   :4.000   Max.   :87.00   Max.   :2.000   Max.   :5.000   Max.   :40.00   Max.   :275.7
>
```

R Console

```
> edit(quinua) # Abre el editor de datos
```

Editor de datos

	IDENTIF	TGRANO	GERMINA	RES_MILD	ALTURA	UNIFORM	HAB_CRECC	LONG_PANJ
1	1	2	2	1	60	1	3	25
2	2	2	5	1	63	1	3	21
3	3	2	5	1	71	1	3	33
4	4	2	2	2	60	1	3	21
5	5	2	5	1	75	1	3	40
6	6	2	3	2	87	1	4	39
7	7	1	5	1	60	1	4	27
8	8	2	4	2	64	1	3	24
9	9	2	5	1	72	1	3	28
10	10	2	2	1	63	2	4	27
11	11	2	5	1	77	1	3	27

Explorando datos

R Console

```
> str(quinua) # Proporciona la estructura del conjunto de datos
'data.frame':   40 obs. of  9 variables:
 $ IDENTIF  : num  1 2 3 4 5 6 7 8 9 10 ...
 $ TGRANO   : num  2 2 2 2 2 2 1 2 2 2 ...
 $ GERMINA  : num  2 5 5 2 5 3 5 4 5 2 ...
 $ RES_MILD : num  1 1 1 2 1 2 1 2 1 1 ...
 $ ALTURA   : num  60 63 71 60 75 87 60 64 72 63 ...
 $ UNIFORM  : num  1 1 1 1 1 1 1 1 1 2 ...
 $ HAB_CRECC: num  3 3 3 3 3 4 4 3 3 4 ...
 $ LONG_PANJ: num  25 21 33 21 40 39 27 24 28 27 ...
 $ P_GRANO  : num  55.1 240 205.8 33.7 275.7 ...
> length(quinua)# Muestra el número de variables que contienen datos
[1] 9
> names(quinua) # Enumera variables en el conjunto de datos
[1] "IDENTIF"   "TGRANO"    "GERMINA"   "RES_MILD"  "ALTURA"    "UNIFORM"   "HAB_CRECC" "LONG_PANJ" "P_GRANO"
> head(quinua) # Las primeras 6 filas de conjunto de datos
  IDENTIF TGRANO GERMINA RES_MILD ALTURA UNIFORM HAB_CRECC LONG_PANJ P_GRANO
1       1      2       2        1     60       1         3        25    55.1
2       2      2       5        1     63       1         3        21   240.0
3       3      2       5        1     71       1         3        33   205.8
4       4      2       2        2     60       1         3        21    33.7
5       5      2       5        1     75       1         3        40   275.7
6       6      2       3        2     87       1         4        39   169.0
> |
```

Explorando el espacio de trabajo

```
objects()                  # Lista los objetos en el espacio de trabajo.
ls()                       # Lo mismo que objects()
remove()                   # Elimina objetos del espacio de trabajo
rm(list=ls())              # Limpia el espacio de la memoria
detach(package:ABC)        # Separa paquetes cuando ya no son necesarios
search()                   # Muestra los paquetes cargados
library()                  # Muestra los paquetes instalados
dir()                      # Muestra archivos en el directorio de trabajo
```

Datos perdidos - Missing

```
rowSums(is.na(students))        # Number of missing per row
colSums(is.na(students))        # Number of missing per column/variable
rowMeans(is.na(students)); length(students)   # No. de missing por fila (otra forma)
                                # length = num. de  variables/elementos en un objeto
# Convertir a datos missing
students[students$age=="& ","age"] <- NA # NOTA: Observar los espacios ocultos
students[students$age==999,"age"] <- NA

# La función complete.cases() devuelve un vector lógico que indica qué casos están
  completos.

# Lista las filas de datos que tienen valores perdidos
students[!complete.cases(students),]

# La función na.omit() devuelve el objeto con borrado de lista de valores perdidos.

# Crear un nuevo conjunto de datos sin datos faltantes
students1 <- na.omit(students)
```

Reemplazar un valor

```
students1 <- na.omit(students)
students1[students1$SAT==1787,"SAT"] <- 1800
students1[students1$Country=="Bulgaria","Country"] <- "US"
```

Renombrar variables

```
# Usando comandos base
fix(students) # Renombrando interactivamente
names(quinua) [c(3)] <- c("GERM") # [c(3)] n° columna donde se hará modif.

# Usando librería --reshape--
library(reshape)
quinua <- rename(quinua, c(TGRANO="Last"))
quinua <- rename(quinua, c(TGRANO="First"))
students <- rename(student, c(Student.Status="Status"))
students <- rename(students, c(Average.score..grade.="Score"))
students <- rename(students, c(Height..in.="Height"))
students <- rename(students, c(Newspaper.readership..times.wk.="Read"))
```

Etiqueta de valores

```
# Usar factor() para datos nominales
students$sex <- factor(students$sex, levels = c(1,2), labels = c("male", "female"))

# Usar ordered() para datos ordinales
students$var2 <- ordered(students$var2, levels = c(1,2,3,4), labels = c("Strongly
        agree", "Somewhat agree", "Somewhat disagree", "Strongly disagree"))

# Como una nueva variable.
students$var8 <- ordered(students$var2, levels = c(1,2,3,4), labels = c("Strongly
        agree", "Somewhat agree", "Somewhat disagree", "Strongly disagree"))
```

Reordenando etiquetas

```
levels(students$Major)

# Sintaxis para reordenar (variable categórica, variable numérica,
    estadística deseada)
students$Major = with(students, reorder(Major,Read,mean)) # El orden va
    de niveles bajos a altos (students$Major)

levels(students$Major)

attr(students$Major, 'scores') # Reordenar crea un atributo llamado ‘scores'
    # (con la estadística) usado para reordenar las etiquetas, en este  caso
    # los valores medios.

# Promedio de tiempo de lectura: 4.4 para Econ, 5.3 para matemáticas,
   4.9 para política (usando students.rda)
```

R Console

```
> attr(students$Major, 'scores')
   Econ    Math Politics
    4.4     5.3      4.9
>
```

Creando identificadores / secuencia de números

```
# Crear una variable con una secuencia de números o indexar

# Creando una variable con una secuencia de números de 1 a n (donde
        'n' es el número total de observaciones)
students$id <- seq(dim(students)[1])

# Creando una variable con el número total de observaciones.
students$total <- dim(students)[1]
```

TRANSFORMACIONES DE DATOS

Transformaciones de datos

- En cualquier momento, podemos modificar o manipular cualquier objeto con operadores (aritméticos, lógicos y comparativos) y funciones. Así, por ejemplo, si una variable no es normal, puedo transformarla para conseguir dicha normalidad.

- **¡Cuidado!** Cualquier modificación que hagamos con las variables de un banco de datos "adjuntado" (attached) no afectarán al propio data.frame.

- Si queremos modificar el contenido interno de los data.frames podemos utilizar la función transform(). En concreto podemos:

  - Transformar variables existentes (para conseguir normalidad, por ejemplo).
  - Crear nuevas variables mediante la transformación de variables existentes.

Recodificación de variables

```
# Instalar y cargar librería car
install.packages(car); library(car)

students$Age.rec <- recode(students$Age, "18:19='18to19';
                                20:29='20to29';
                                30:39='30to39'")

students$Age.rec <- as.factor(students$Age.rec)

# Ingreso de datos por teclado
datosimp <- data.frame(anyos=c(1.3,0.4,1.1,2.3,3.1,1.3),
                tipo=c(2,3,3,1,3,1),edad=c(22,21,34,42,17,43),
                sexo=c("H","M","H","H","M","H"))
                attach(datosimp)

ec<-recode(edad, "15:25='joven'; 26:65='adulto' ", as.factor.result=T)
sexo.cod<-recode(sexo, " 'H'=0 ; 'M'=1 ", as.factor.result=TRUE)
```

Recodificación de variables

Resultado

```
> ec<-recode(edad, "15:25='joven'; 26:65='adulto'", as.factor.result=T)
> ec
[1] joven  joven  adulto adulto joven  adulto
Levels: adulto joven
> sexo.cod<-recode(sexo, "'H'=0 ; 'M'=1", as.factor.result=TRUE)
> sexo.cod
[1] 0 1 0 0 1 0
Levels: 0 1
```

Cálculo de variables

Haciendo uso de la función **attach()**

```
datosimp
edad.final <- edad + anyos # no afecta al data.frame
datosimp.1 <- transform(datosimp, edad.final = edad+anyos) # si afecta
datosimp.2 <- transform(datosimp.1, edad = edad+1) # altera la variable
```

```
> edad.final
[1] 23.3 21.4 35.1 44.3 20.1 44.3
> datosimp.1 <- transform(datosimp, edad.final = edad+anyos) #si afecta
> datosimp.1
  anyos tipo edad sexo edad.final
1   1.3    2   22    H       23.3
2   0.4    3   21    M       21.4
3   1.1    3   34    H       35.1
4   2.3    1   42    H       44.3
5   3.1    3   17    M       20.1
6   1.3    1   43    H       44.3
> datosimp.2 <- transform(datosimp.1, edad = edad+1) #altera la variable
> datosimp.2
  anyos tipo edad sexo edad.final
1   1.3    2   23    H       23.3
2   0.4    3   22    M       21.4
3   1.1    3   35    H       35.1
4   2.3    1   43    H       44.3
5   3.1    3   18    M       20.1
6   1.3    1   44    H       44.3
```

Cálculo de variables a partir de otras existentes

```
# Logaritmo natural de una variable P_GRANO
quinua$LGPESO <- with(quinua, log(P_GRANO))
summary(quinua$LGPESO)
```

# LGPESO	es el nombre de la variable a crear en el grupo de datos quinua.
# with()	es la función que permite la creación de la nueva variable luego de realizar la función log()en la base de datos quinua.
# log()	función que permite calcular el logaritmo natural de una variable cuantitativa.

Subconjunto de variables

```
students2 <- students[,1:14] # Selecciona las primeras 14 variables
students2 <- students[c(1:14)]

sat <- students[c("Last", "First", "SAT")]
sat1 <- students[c(2,3,11)]

select <- students[c(1:3, 12:14)] # Escribir names(select) para verificar
select1 <- students[c(-(1:3), -(12:14))] # Excluyendo variables
```

Subconjunto de observaciones

```
students2 <- students2[1:30,] # Selecciona las primeras 30 observaciones

students3a <- students [which(students$Gender=='Female' & students$SAT > 1800), ]
```

Subconjunto de variables/observaciones

```
students2 <- students2[1:30,1:14] # Selecciona las primeras 30 observacions y las
                                  primeras 14
```

Subconjunto utilizando --subset--

```
install.packages("devtools")
library(devtools)

students3 <- subset(students2, Age >= 20 & Age <= 30)
students4 <- subset(students2, Age >= 20 & Age <= 30, select=c(ID,
      Age, SAT, Gender))
students5 <- subset(students2, Gender=="Female" &
      Student.Status=="Graduate" & Age >= 30)
students6 <- subset(students2, Gender=="Female" &
      Student.Status=="Graduate" & Age == 30)
```

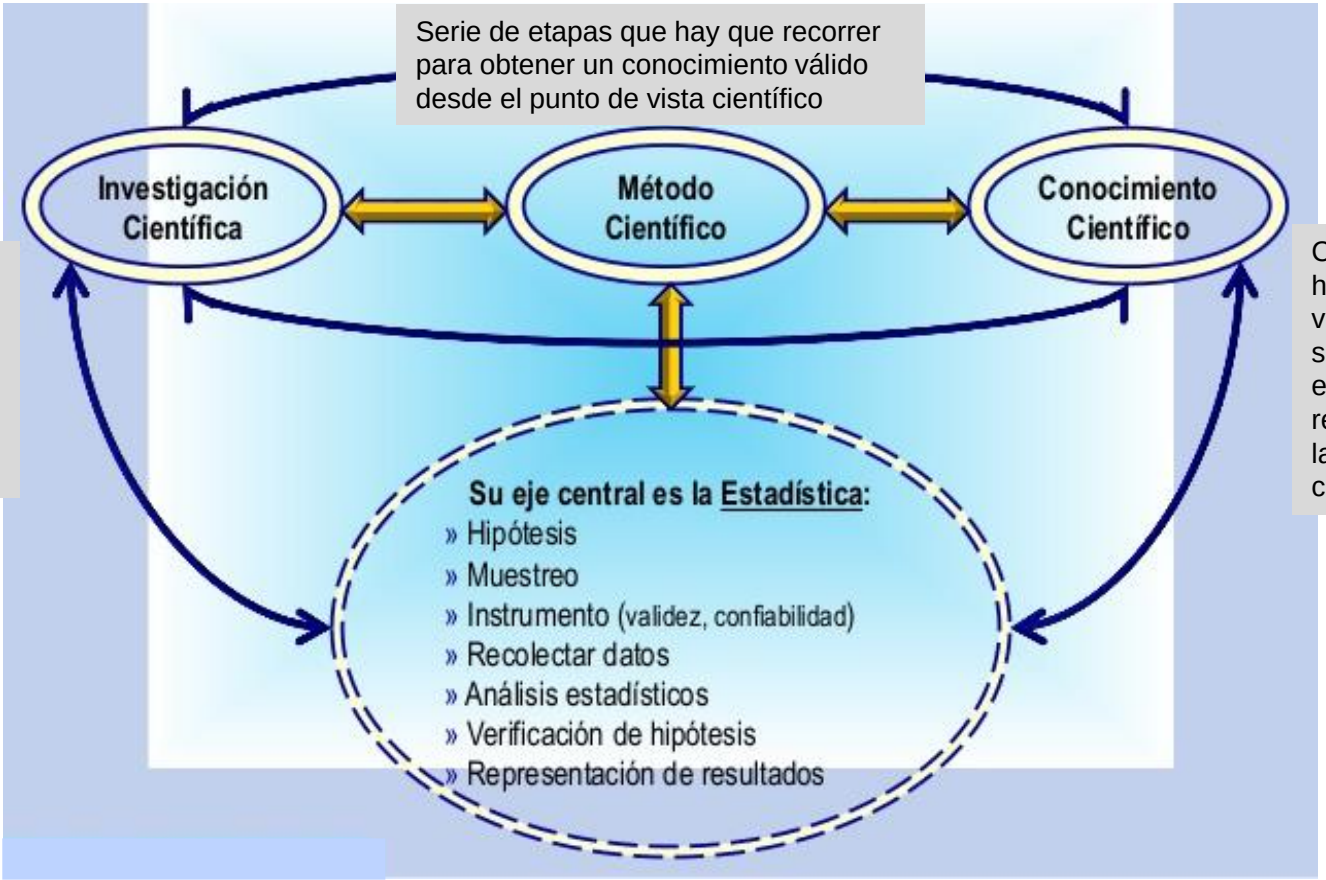
ESTADÍSTICA DESCRIPTIVA

Introducción

Investigación Científica – Método Científico – Estadística

Serie de etapas que hay que recorrer para obtener un conocimiento válido desde el punto de vista científico

Proceso ordenado y sistemático de análisis o indagación en torno a un asunto o tema.



Conjunto de hechos verificables y sustentados en evidencia recogidos por las teorías científicas

¿Qué es la Estadística?

La Estadística es la Ciencia de la:

Descriptiva

- sistematización, recolección, ordenación y presentación de los datos referentes a un fenómeno que presenta variabilidad o incertidumbre para su estudio metódico, con objeto de …

Probabilidad

- deducir las leyes que rigen esos fenómenos

Inferencia

- y poder hacer previsiones sobre los mismos, obtener conclusiones y tomar decisiones.

Etapas del análisis estadístico

Estadística Descriptiva

VARIABLE

- Característica de interés

MUESTRA OBSERVADA

- Conjunto de valores de la variable observados obtenidos de manera homogénea

TAMAÑO MUESTRAL

- Número de datos observados

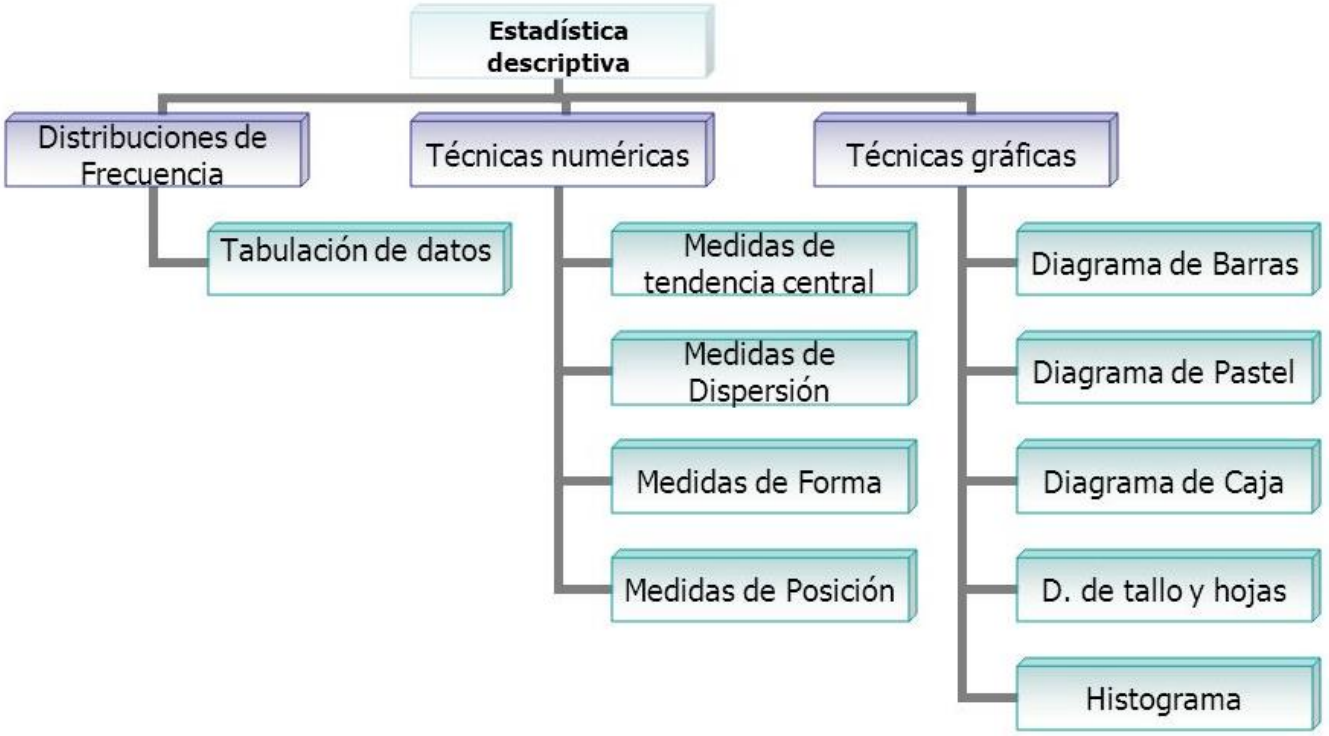
LA DESCRIPCIÓN DE LA MUESTRA DEPENDE DEL TIPO DE ATRIBUTO

- Cualitativo: Intrínsecamente no tiene carácter numérico (categórica)
- Cuantitativo: Intrínsecamente numérico

Análisis Exploratorio de datos

Métodos descriptivos de análisis

Métodos numéricos y métodos gráficos



Nota: Este diagrama es sólo una referencia general y la clasificación puede variar de acuerdo a la fuente consultada.

Tipos de variables en el análisis

1. **Variables cualitativas**: Describen cualidades o atributos (ej. color del pelo; sexo de una persona; etc.).
2. **Variables cuantitativas discretas**: Toman un número pequeño de valores, normalmente enteros (ej. número de hijos).
3. **Variables cuantitativas continuas**: Toman valores en un intervalo (ej. tiempo hasta que llega un autobús).

En los datos sobre contenido de mercurio, ¿de qué tipo es cada una de las variables?

En general, la técnica estadística adecuada para analizar una variable depende de su tipo.

Distribución de una variable

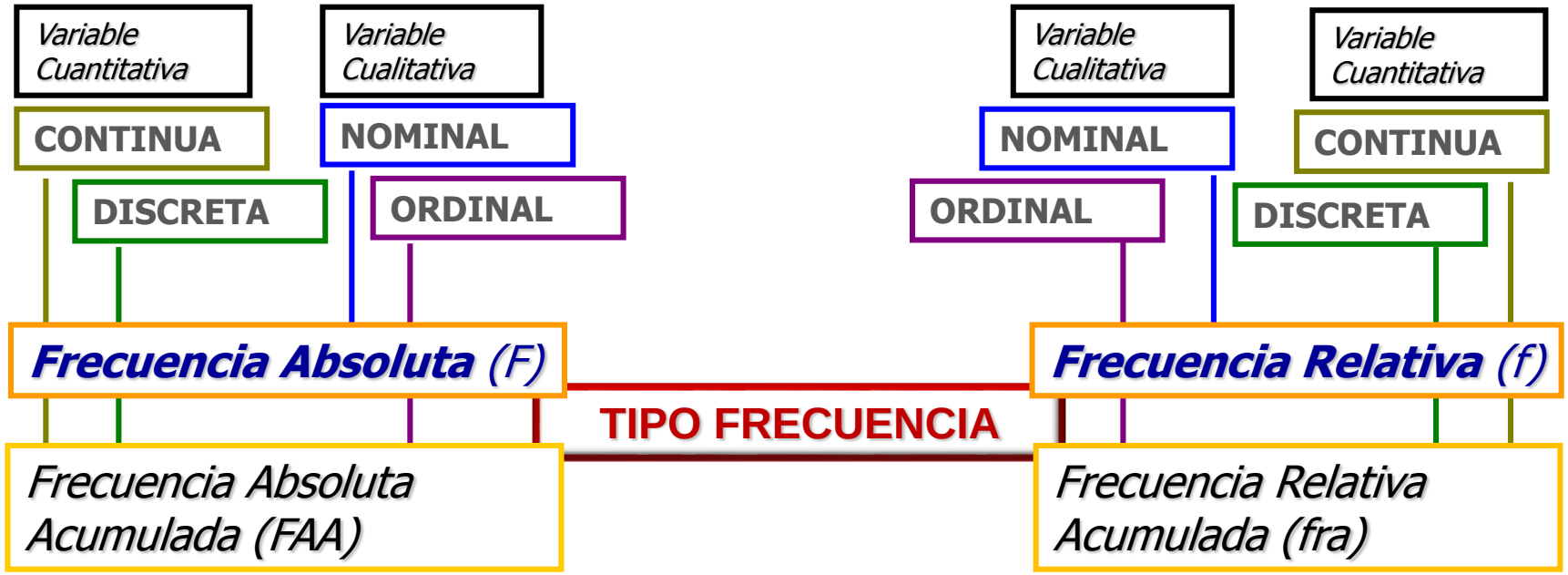
- Normalmente, los valores que toma una variable x en una muestra de tamaño n se suelen representar genéricamente por $x_1, x_2, \ldots, x_n$.
- La **distribución de una variable** viene determinada por los valores que toma esa variable y la frecuencia con la que los toma.
- La **frecuencia absoluta** de un valor (o de un intervalo) es el número de individuos para los que la variable toma ese valor (o pertenece a ese intervalo).
- La **frecuencia relativa** es igual a la frecuencia absoluta dividida por el número total de datos n.
- La frecuencia relativa siempre es un número entre 0 y 1.

Distribución de una variable

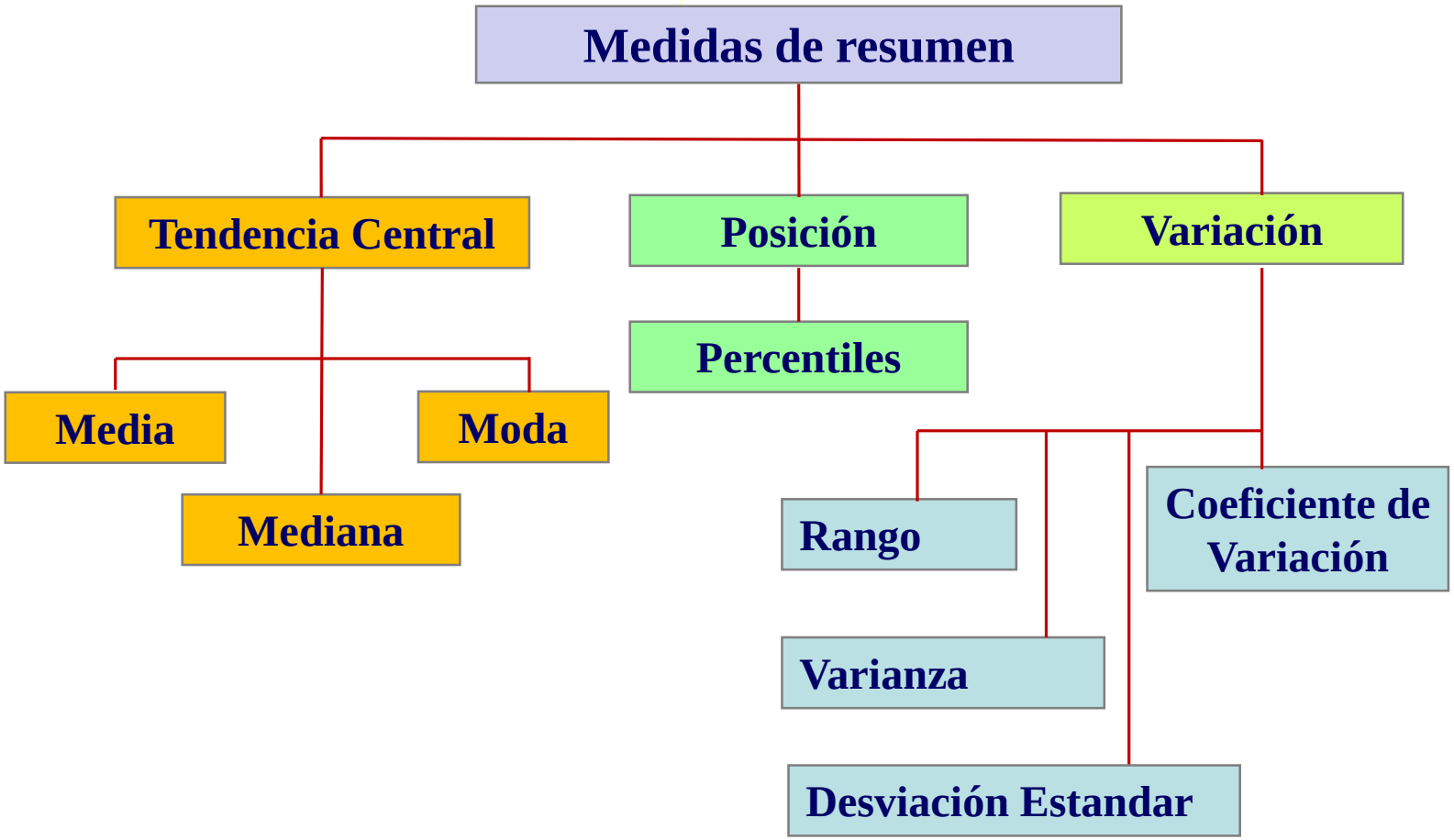
- Su **posición**: entorno a qué valor central toma valores la variable.
- Su **dispersión**: el grado de concentración de los valores que toma la variable alrededor de su posición central.
- Su **forma**: por ejemplo, la simetría, es decir, si los valores se reparten de la misma forma a uno y otro lado del centro.

Distribución de variables categóricas

Frecuencia: desde un conjunto de unidades, corresponde al Número o Porcentaje de veces que se presenta una característica.



Medidas de resumen variables continuas



Métodos numéricos y métodos gráficos

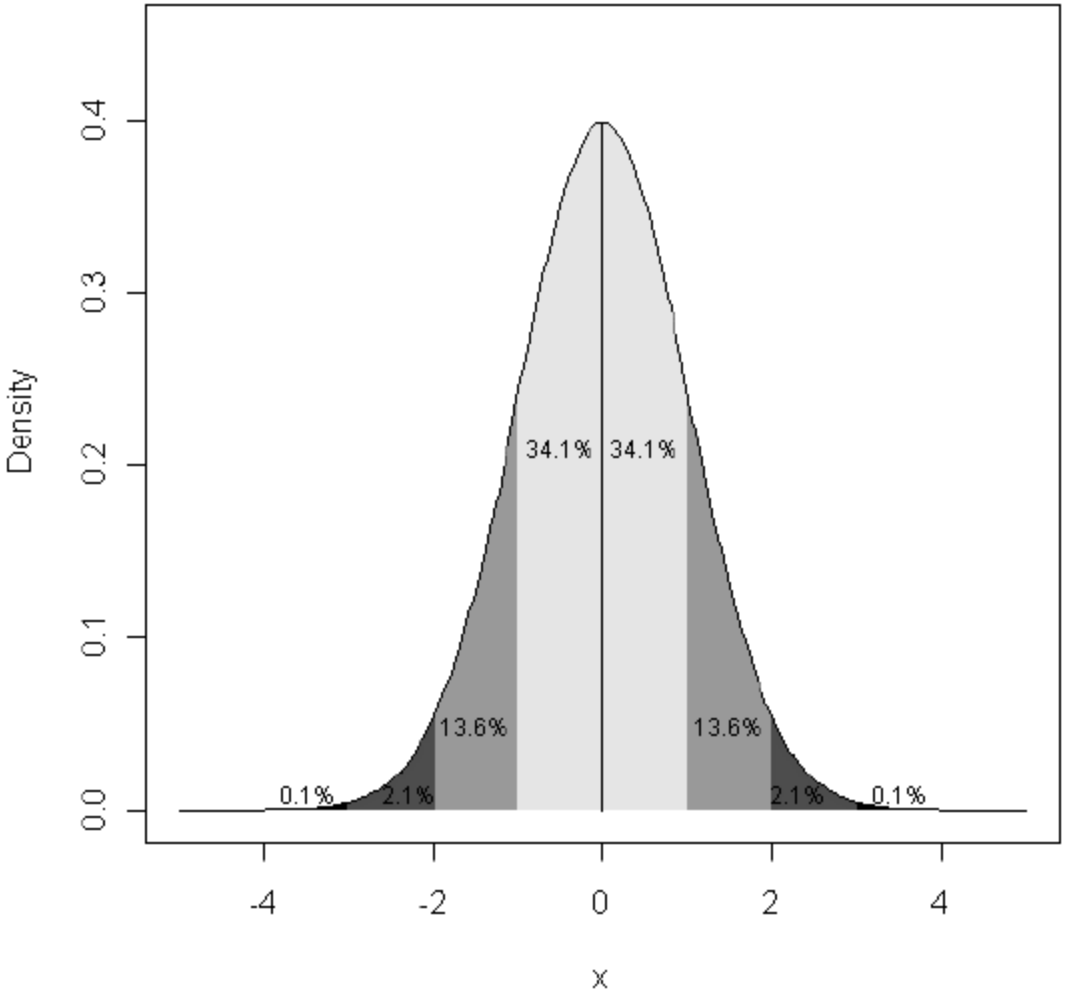
- **Variable categórica simple**
  - Descriptivo: Frecuencia absoluta y relativa, moda
  - Gráficos: Barras, sectores circulares, pictogramas
- **Variable continua simple**
  - Descriptivo: Promedio, mediana, moda, desviación estándar…
  - Gráficos: Diagrama de caja, histograma (diagrama de densidad, tallo y hoja), de probabilidad normal
- **Dos variables categóricas**
  - Descriptivo: Individual, tabla de contingencia
  - Gráficos: Barras agrupadas, mosaicos
- **Dos variables continuas**
  - Descriptivo: Individual
  - Gráficos: Diagrama de dispersión

Métodos numéricos y métodos gráficos

- **Una variable continua, una variable categórica**
  - Descriptivo: promedio, mediana, desviación estándar..., para cada categoría por separado
  - Gráficos: diagrama de caja, histograma, pero para cada categoría por separado

- **Varias variables continuas y/o categóricas**
  - Descriptivo: Como para variables continuas o categóricas
  - Gráficos: Diagrama de dispersión para cada par, diagrama de mosaico

Distribución de probabilidad Normal



- Algunas medidas se distribuyen normalmente en el mundo real
  - Altura
  - Peso
- La media de las observaciones tomados de los datos distribuidos de otra manera también se distribuyen normalmente
- Por lo tanto, muchas pruebas descriptivas y estadísticas han sido basadas en el supuesto de la normalidad

Números aleatorios

- R tiene las distribuciones de probabilidad más comunes implementadas en la librería BASE. En otras librerías disponemos de otras tantas.
- Para cada una de ellas (distrib), disponemos de 4 versiones:

generador de números aleatorios	rdistrib
función densidad/probabilidad	ddistrib
función distribución	pdistrib
función inversa distribución (cuantiles)	qdistrib

Números aleatorios

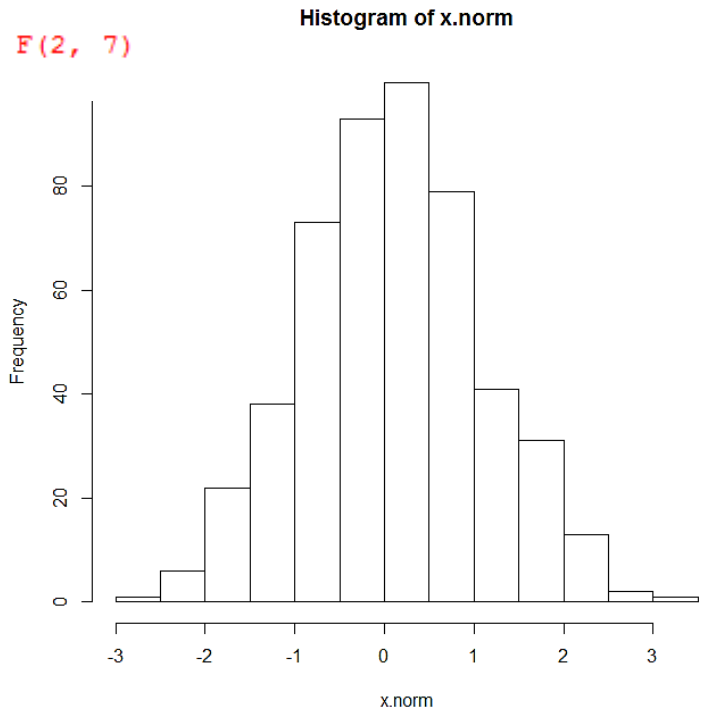
```
x <- rnorm(10, mean=0, sd=1) # Crea 10 números (dist. normal), sintaxis
        general rnorm(n, mean, sd)
x
x <- data.frame(x)
x <- matrix(x)

x.norm <- rnorm(500)      #Simulación de 500 datos normales
2*pt(-2.43, df = 13)      #p-valor de dos colas de una t(13)
qf(0.99, 2,7)             #Percentil 99 de una distribución F(2,7)
```

Números aleatorios

Resultados

R Console

```
> x.norm <- rnorm(500) # Simulación de 500 datos normales
> hist(x.norm)
> 2*pt(-2.43, df = 13) # P-valor de dos colas de una t(13)
[1] 0.0303309
> qf(0.99, 2, 7) # Percentil 99 de una distribución F(2, 7)
[1] 9.546578
```



Distribuciones de Probabilidad en R

Distribuciones de probabilidad en la librería BASE.

Función	Utilidad
Normal	rnorm(n, mean=0, sd=1)
exponencial	rexp(n, rate=1)
gamma	rgamma(n, shape, scale=1)
Poisson	rpois(n, lambda)
Weibull	rweibull(n, shape, scale=1)
Cauchy	rcauchy(n, location=0, scale=1)
beta	rbeta(n, shape1, shape2)
t de Student	rt(n, df)
F (Snedecor)	rf(n, df1, df2)
Pearson χ^2	rchisq(n, df)
binomial	rbinom(n, size, prob)
geométrica	rgeom(n, prob)
hypergeométrica	rhyper(nn, m, n, k)
logística	rlogis(n, location=0, scale=1)
lognormal	rlnorm(n, meanlog=0, sdlog=1)
binomial negativa	rnbinom(n, size, prob)
uniforme	runif(n, min=0, max=1)

Tablas de frecuencia

- Hemos visto que un factor es un vector utilizado para especificar una clasificación discreta de los elementos de otro vector de igual longitud, y que en R existen dos tipos de factores (variables categóricas)
- Del mismo modo, dos factores definen una tabla de doble entrada, y así sucesivamente.
- La función table() calcula tablas de frecuencias a partir de factores de igual longitud.

Tablas de frecuencia

```
table(students$Gender)
x <- as.factor(1:5)  # Convierte vectores en factores
table(x)

summary(quinua[2])
quinua <- transform(quinua, TGRANO=factor(TGRANO,
labels = c("Chico", "Mediano")))
summary(quinua[2])
```

Tablas de frecuencia

```
table(quinua$TGRANO)  # Frecuencias absolutas
cuadro1 <- table(quinua$TGRANO)  # Convierte tabla en objeto
 cuadro1
# En adelante nos podemos referir al objeto cuadro1 cada vez que queramos utilizar la tabla en valores absolutos de la variable TGRANO

cuadro1 / margin.table(cuadro1) # Frecuencias relativas
round(((cuadro1/margin.table(cuadro1))*100),2)  # Tabla de frecuencia
        relativa en porcentajes y redondeada a dos dígitos
cumsum(round(((cuadro1/margin.table(cuadro1))*100),2)) # Frecuencia
        relativa acumulada
```

Tablas cruzadas

```
readgender <- table(students$Read, students$Gender)
readgender
addmargins(readgender) # Agregar márgenes de fila / col

prop.table(readgender,1) # Proporciones fila
round(prop.table(readgender,1), 2) # Redondea prop fila a 2 dígitos
round(100*prop.table(readgender,1), 2) # Redondea prop fila 2 a dígitos (porcent.)
addmargins(round(prop.table(readgender,1), 2),2) # Redondea prop fila a 2 dígitos

prop.table(readgender,2) # Proporciones columna
round(prop.table(readgender,2), 2) # Redondea prop col a 2 dígitos
round(100*prop.table(readgender,2), 2) # Redondea prop col 2 a dígitos (porcent.)
addmargins(round(prop.table(readgender,2), 2),1) # Redondea prop col a 2 dígitos

prop.table(readgender) # Proporciones totales
round(prop.table(readgender),2) # Proporciones totales redondeada
round(100*prop.table(readgender),2) # Proporciones totales redondeada
```

Resumen de variables continuas

Medidas de localización y dispersión más habituales.

Función	Utilidad
sum(..., na.rm=FALSE)	Suma
max(..., na.rm=FALSE)	Máximo
min(..., na.rm=FALSE)	Mínimo
which.min(x)	Posición del máximo
which.max(x)	Posición del mínimo
pmax(...,na.rm=FALSE)	Máximo en paralelo
pmin(...,na.rm=FALSE)	Mínimo en paralelo
cumsum(x), cumprod(x)	Sumas y prods acumulados
cummax(x), cummin(x)	max's y min's acumulados
mean(x, trim=0, na.rm=FALSE)	Media
weighted.mean(x,w,na.rm=FALSE)	Media ponderada
median(x,na.rm=FALSE)	Mediana
quantile(x,prob=(0,0.25,0.5,0.75,1),na.rm=F)	Cuantiles
fivenum(x, na.rm=FALSE)	5-Tukey: min, lower-hinge mediana, upper-hinge, máximo
summary(x, na.rm=FALSE)	min,1c,mediana,media,3c,max
IQR(x, na.rm=FALSE)	Rango inter-cuartílico
range(...,na.rm=FALSE, finite=FALSE)	Rango
var(x, y=x, na.rm=FALSE, use)	Varianza
sd(x, na.rm=FALSE)	Desviación Típica
mad(x,center,constant=1.4426, na.rm=FALSE)	Desviación mediana absoluta

Resumen de variables continuas

```
summary(students) # Mean of all numeric variables
mean(students$SAT)
with(students, mean(SAT))
quantile(students$SAT) # Cuantiles 25%
quantile(students$SAT, c(.3,.6,.9)) # Personalizar cuantiles

x<-rnorm(100)
summary(x)

x <- rgamma(50,1,3)
summary(x); fivenum(x)
mean(x); median(x); quantile(x); quantile(x,c(0.35,0.9))
sd(x); var(x); range(x); IQR(x)
# Resumen con localización indexada
min(x); which.min(x); x[which.min(x)]; pmin(x[1:5],x[6:10])
max(x); which.max(x); x[which.max(x)]; pmax(x[4:8],x[2:6])
```

Resumen de variables continuas

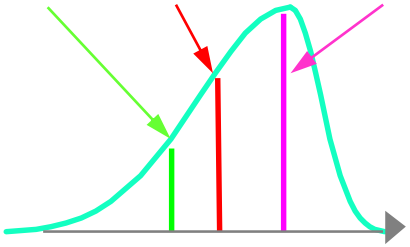
Estadísticas descriptivas por grupos usando --tapply--

```
mean <- tapply(students$SAT, students$Gender, mean) # Adicionar na.rm=TRUE para remover los valores perdidos en la estimación

sd <- tapply(students$SAT, students$Gender, sd)

median <- tapply(students$SAT, students$Gender, median)

max <- tapply(students$SAT, students$Gender, max)

cbind(mean, median, sd, max) # Toma una secuencia de vectores, matriz o argumentos de marcos de datos y combina por columnas o filas, respectivamente
```

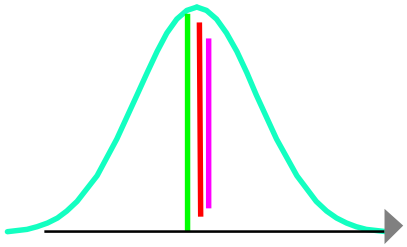
Medidas de forma

Caracterización del perfil de la distribución: **Simétrica o sesgada**

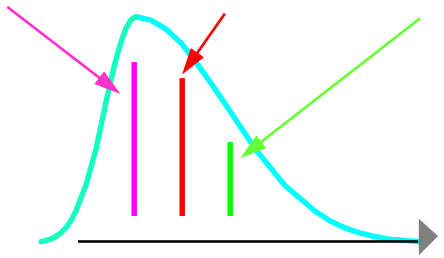
< -1
Sesgada izquierda
Media Mediana Moda



-0.5 < 0 < 0.5
Simétrica
Media = Mediana = Moda

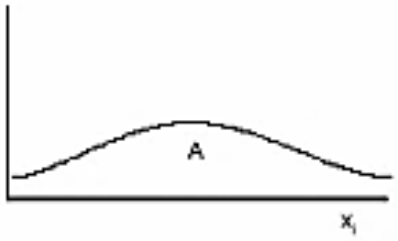


> 1
Sesgada derecha
Mediana Media

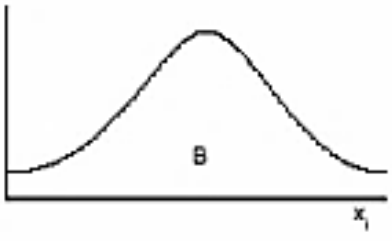


Curtosis (kurtosis), mide la densidad que se concentra en las colas de la distribución. Una distribución normal (del tipo Gauss) tiene una kurtosis igual a tres (Fig. B). Valores menores a 3 indican una forma platocúrtica (Fig. A) y valores mayores a 3 indican una forma leptocúrtica (fig. C) o alta concentración de los datos alrededor de la media de la distribución.

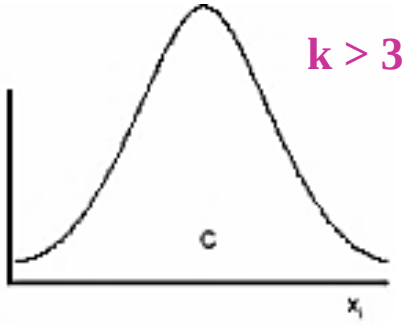
k < 3



k = 3



k > 3



Resumen de variables continuas - Medidas de forma

```
library(e1071)
x <- rgamma(50,1,3)
moment(x,2,center=F) # momento no centrado de orden 2

# Consideramos dos distribuciones asimétricas (Betas) y las vamos a
      comparar con la normal que es simétrica:
nsim <- 5000
s1 <- skewness(rbeta(nsim,2,3))
s2 <- skewness(rbeta(nsim,3,2))
s3 <- skewness(rnorm(nsim,0.5,0.5))
s1;s2;s3

# Consideramos ahora una distribución normal y una Student, más
      achatada, y las comparamos:
k1 <- kurtosis(rnorm(nsim))
k2 <- kurtosis(rt(nsim,3))
k1;k2
```

Representación gráfica de datos

Podemos dividir los comandos para efectuar las gráficas en tres grupos:

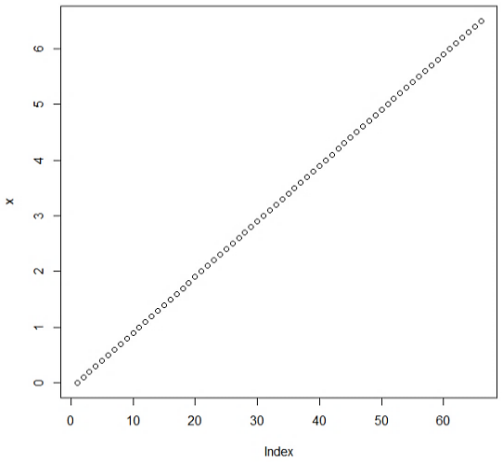
- Funciones para crear gráficas de alto nivel, es decir ya programadas y que admiten diferentes posibilidades
- Funciones de bajo nivel, que permiten un control más fino del dibujo y permiten crear gráficas a medida.
- Funciones para el uso interactivo, para extraer información de una gráfica o una modificación mediante el ratón.

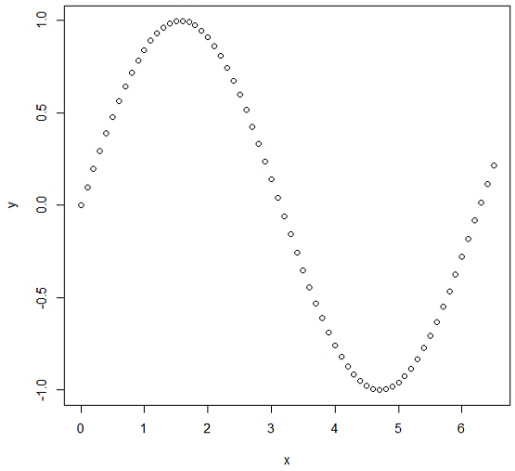
Representación gráfica - Procedimientos de alto nivel

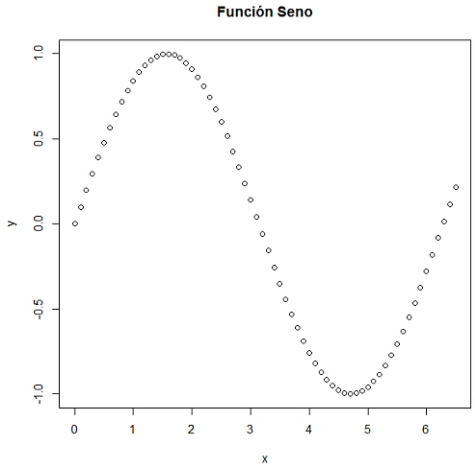
```
x <- (0:65)/10
y <- sin(x)
plot(x)
plot(x, y)
plot(x, y, main="Función Seno")
z <- cos(x)
windows() # Crea una ventana nueva
plot(x, z, main="Función Coseno")
```

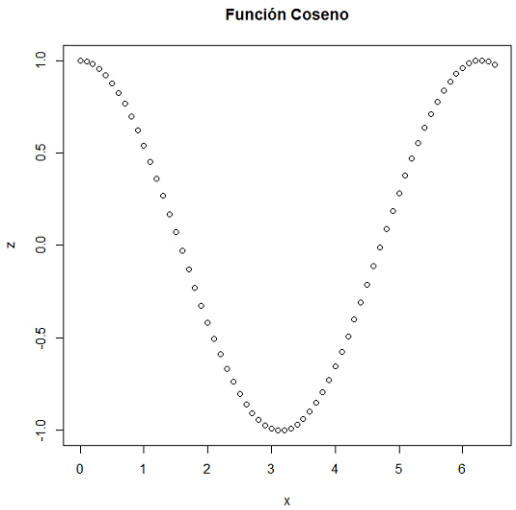
Representación gráfica - Procedimientos de alto nivel

Resultados









Opciones de la función plot()

Algunas de las más útiles

- `main`: Cambia el título del gráfico
- `sub`: Cambia el subtítulo del gráfico
- `type`: Tipo de gráfico (puntos, líneas, etc.)
- `xlab, ylab`: Cambia las etiquetas de los ejes
- `xlim, ylim`: Cambia el rango de valores de los ejes
- `lty`: Cambia el tipo de línea; `lwd`: Cambia el grosor de línea
- `col`: Color con el que dibuja

Función plot()

```
plot(x, y, main="Seno", type="l")
plot(x, z, main="Coseno", lty=2, col="red", type="l")
plot(x, z, main="Coseno", lty=3, col="blue", type="l",
xlim=c(0, 2), ylab="cos(x)")
```

Procedimientos de bajo nivel

Hay una serie de funciones que permiten dibujar sobre una gráfica ya creada.

Los más habituales

- `points(x, y, ...)`: Dibuja una nube de puntos
- `lines(x, y, ...)`: Dibuja una línea que une todos los puntos
- `ablines()`: Dibuja una línea recta dada la interc. y pendiente
- `polygons(x, y, ...)`: Dibuja un polígono cerrado
- `text(x, y, labels, ...)`: Escribe texto en unas coordenadas

Representación gráfica - Procedimientos de bajo nivel

```
plot(x, y, main="Funciones seno y coseno", type="l")
lines(x, z, col="blue", lty=2) # col=4 es equivalente
text(x=c(0.5, 0.5), y=c(0, 1), labels=c("sin(x)", "cos(x)"),
col=c("black", "blue"))
```

Leyendas en gráficos

Descripción

La función `legend(x, y, legend, ...)` permite añadir leyendas a un gráfico:

- `x,y` : Esquina sup. izda. de la leyenda
- `legend`: Texto de la leyenda
- `bty`: Tipo de borde ("n" para omitir)

Representación gráfica - Leyendas

```
plot(x, y, main="Funciones seno y coseno", type="l")
lines(x, z, col="blue", lty=2)
legend(x=3, y=1, legend=c("sin(x)", "cos(x)"), lty=c(1,2),
col=c("black", "blue"))
```

Parámetros gráficos - función par()

- Al igual que en la función plot(), podemos controlar casi todos los aspectos de una gráfica mediante los parámetros gráficos.
- R dispone de una lista (unos 70) de parámetros gráficos que controlan, por ejemplo, el color, tipo de línea, grosor, tipo de punto, justificación del texto, tamaño, etc.
- Cada parámetro tiene un nombre (p.e. "col" para color) y un valor. R dispone de colores de los más variado. Ver el archivo UsingColorInR.
- Cada vez que hacemos una gráfica se abre un dispositivo gráfico con una lista de parámetros que tiene unos valores iniciales por defecto.
- Los parámetros gráficos pueden fijarse:
  - de forma permanente para ese dispositivo mediante la función par()
  - o temporalmente en las llamadas a las funciones gráficas incluyéndolos en la lista de argumentos (si lo permiten).

Función **par()**

```
# Sin argumentos
par()

# Con un argumento, vector de caracteres, con los nombres de algunos
   parámetros, devuelve una lista con los parámetros y sus valores en  activo
par(c("col" , "lty“))

# Con nombres de parámetros = valor, establece los nuevos valores
par(c(col = 4, lty = 2))
```

Parámetros gráficos

Colocar varias gráficas en una ventana

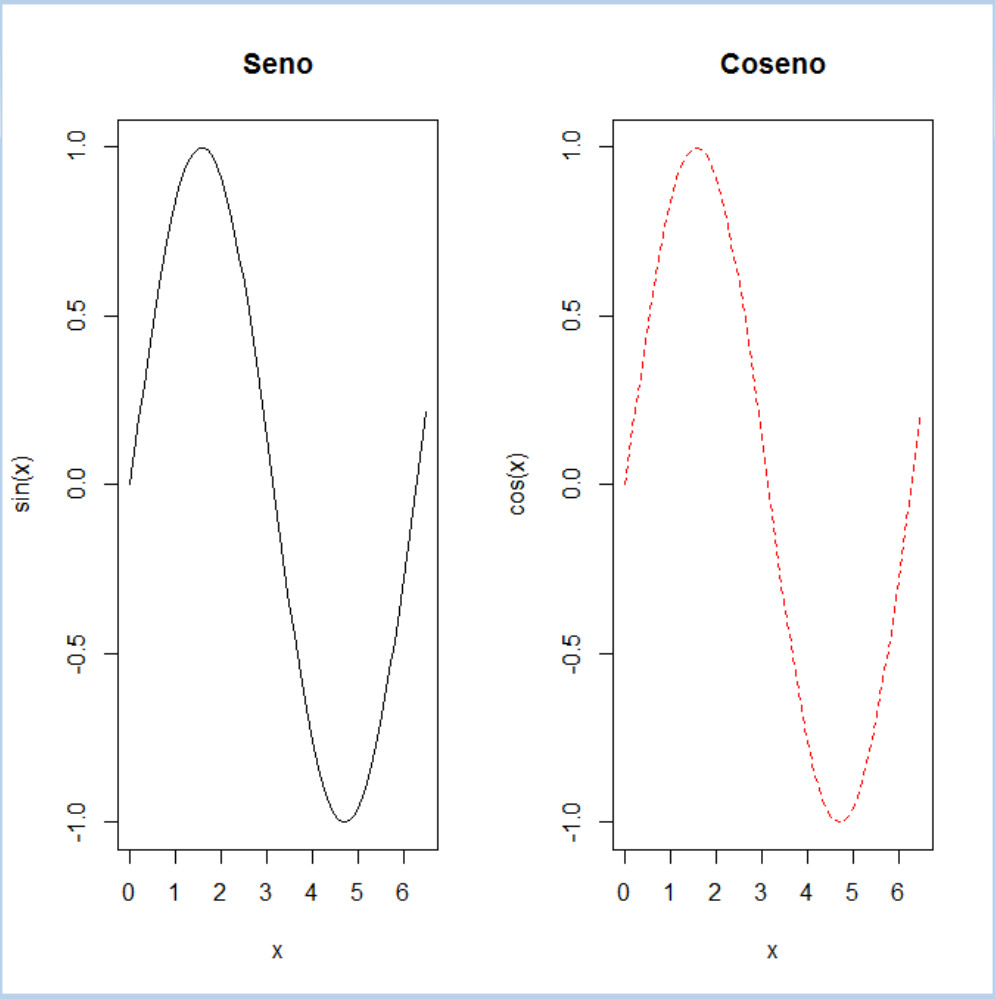
Los siguientes parámetros permiten diseñar el número de gráficas en cada dispositivo gráfico

- `mfrow`: Nº de filas y columnas en la ventana. Los huecos se rellenan por filas.
- `mfcol`: Ídem pero se rellena por columnas.

Parámetros gráficos

```
x<-(0:65)/10
y<-sin(x)
par(mfrow=c(1,2)) # probar también c(2,1)
plot(x, y, main="Seno", type="l", ylab="sin(x)")
plot(x, z, main="Coseno", type="l", lty=2, col="red",
ylab="cos(x)")
```

Resultados



Funciones gráficas interactivas

En R existen una serie de funciones que permiten completar los gráficos de manera interactiva por parte del usuario.

Descripción

- `identify(x, y, etiquetas)` identifica los puntos con el ratón y escribe la correspondiente etiqueta.
- `locator()` devuelve las coordenadas de los puntos.

Funciones gráficas interactivas

```
plot(x, y, main="Funciones seno y coseno", type="l")
lines(x, z, col=2, lty=2)
legend(locator(1),legend=c("sin(x)","cos(x)"),lty=c(1,2),col=c(1,2))

x <- 1:10; y <- sample(1:10)
nombres <- paste("punto", x, ".", y, sep ="")
plot(x, y); identify(x, y, labels = nombres)
```

Representación gráfica de datos discretos

Los más habituales

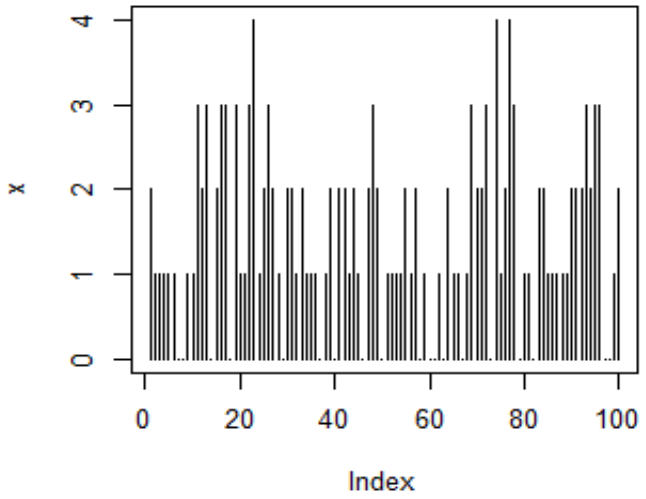
Para representar variables categóricas o cuantitativas discretas (con pocas clases):

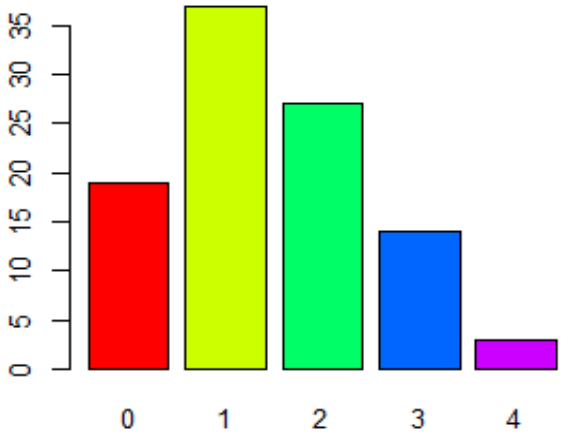
- Diagramas de puntos: `dotplot()`
- Diagramas de barras: `barplot()`
- Diagramas de quesos: `pie()`

Representación gráfica - Procedimientos de alto nivel

```
library(lattice)
x <- rbinom(100,5,0.3)
par(mfrow=c(2,2))
dotplot(x); plot(x,type="h") # Diagrama de puntos
barplot(table(x),col=rainbow(length(table(x)))) # Diagrama barras
pie(table(x)) # Diagrama de sectores circulares
```

Representación gráfica de datos discretos





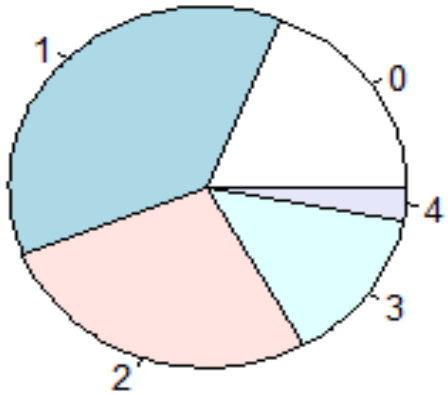
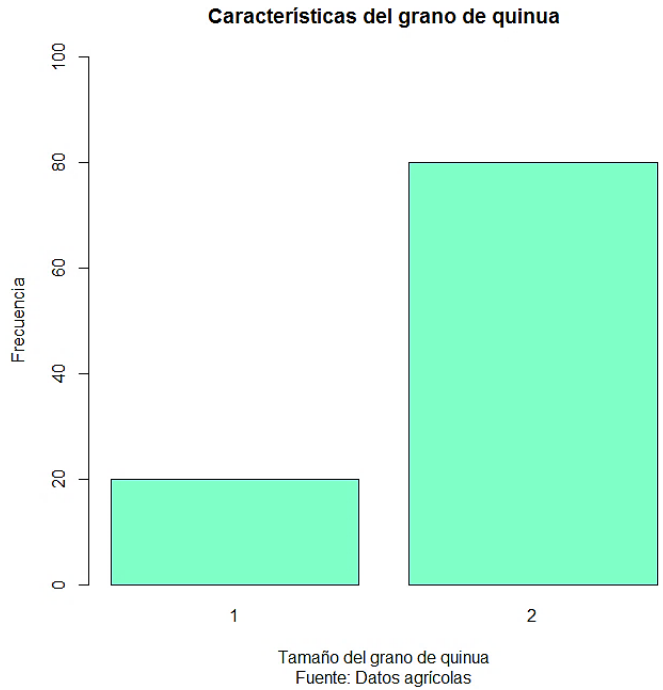


Gráfico de barras - barplot()

El gráfico de barras nos permite observar la forma en que se distribuyen los datos de las variables cualitativas. Esta distribución se puede mostrar en valores absolutos o relativos.

Gráfico de barras **barplot()**

```
cuadro2 <- table(quinua$TGRANO)
cuadro2<-round(((cuadro1/
margin.table(cuadro1))*100),2) barplot(cuadro2,
main="Características del grano de quinua",
sub="Fuente: Datos agrícolas",
xlab="Tamaño del grano de quinua",
ylab="Cantidad", ylim=c(0, 100),
col="aquamarine")
```



R maneja 657decolores, podemos tener acceso a través de la función colors().

Gráfico de sectores en porcentajes - pie()

Al igual que los gráficos de barras, los gráficos de sectores nos permiten observar la forma en que se distribuyen los datos de las variables cualitativas y también se puede mostrar en valores absolutos o relativos.

Gráfico de sectores **pie()**

```
load("D:/quinua.Rdata")
cuadro3 <- table(quinua$TGRANO)
c3por<-round(((cuadro3/margin.table(cuadro3))*100),1)
etiquetas <- c("Chico", "Mediano")
etiquetas <- paste(etiquetas, c3por, "%", sep="")
pie(c3por,labels=etiquetas,
clockwise=TRUE,
main="Características del grano de quinua",
sub="Fuente: Datos agrícolas")
```

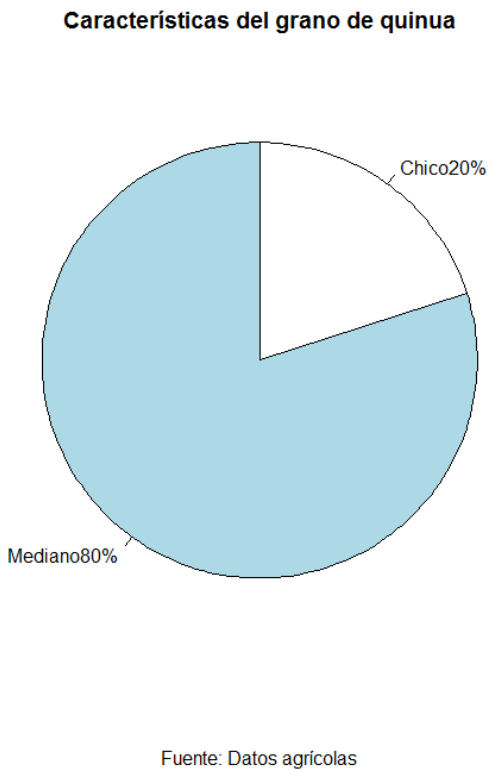
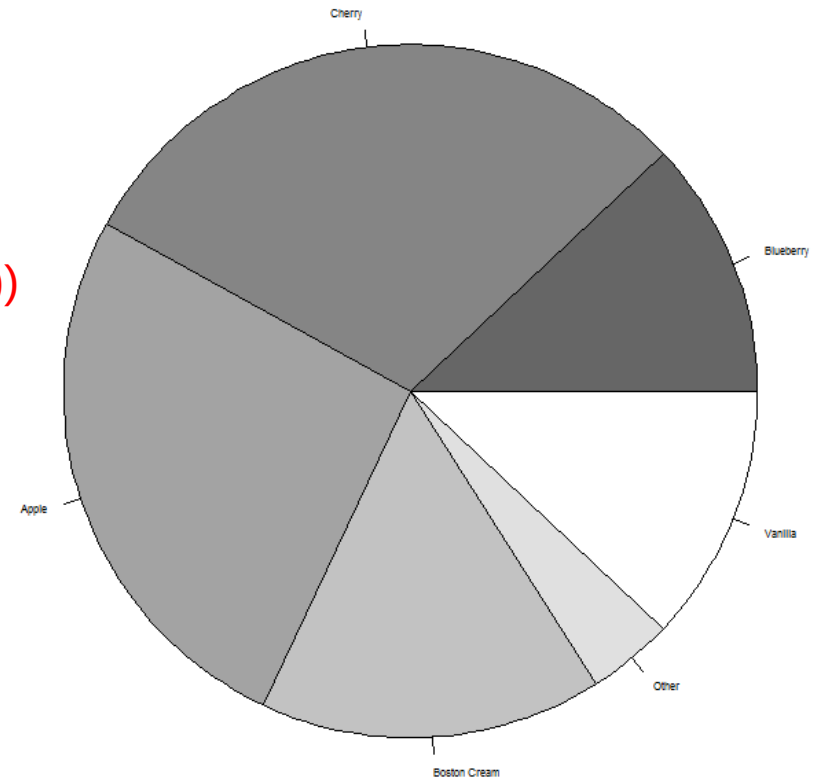


Gráfico de sectores **pie()**

```
par(mar=c(0, 2, 1, 2), xpd=FALSE, cex=0.5)
pie.sales <- c(0.12, 0.3, 0.26, 0.16, 0.04, 0.12)
names(pie.sales) <- c("Blueberry", "Cherry",
"Apple", "Boston Cream", "Other", "Vanilla")
pie(pie.sales, col = gray(seq(0.4,1.0,length=6)))
```

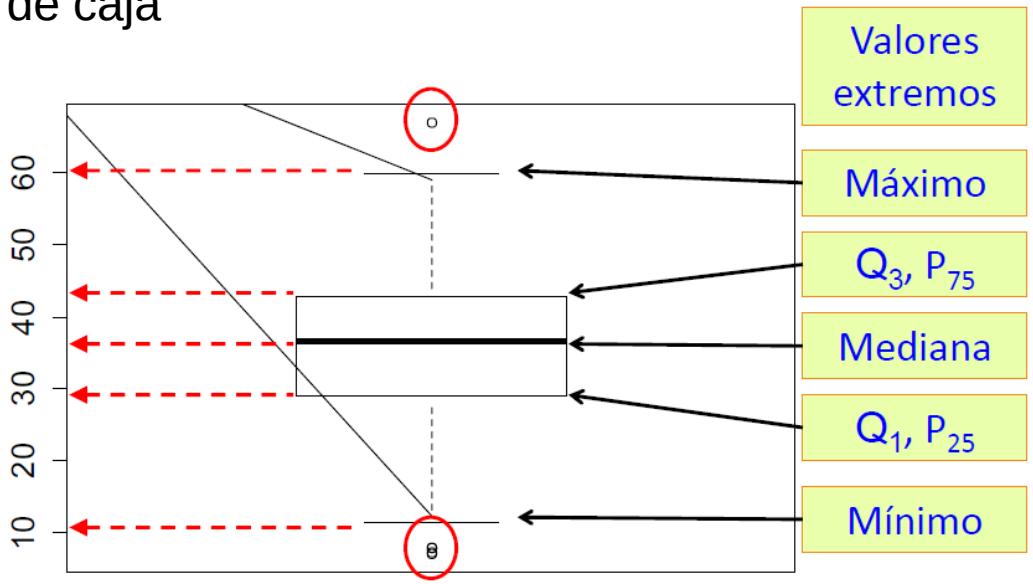


Representación gráfica para datos continuos

Los más habituales

- Diagramas de cajas: `boxplot()`
- Diagramas de tallo y hojas: `stem()`
- Diagramas de puntos: `stripchart()`

Diagrama de caja



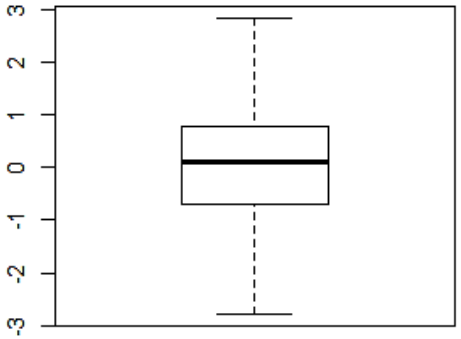
Gráficos para datos continuos

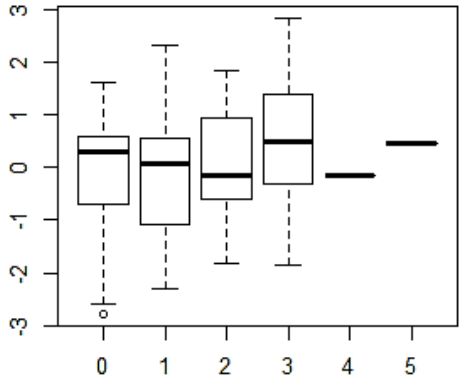
```
y <- rnorm(100); y.f <- rbinom(100,5,0.3)

# Gráfico de tallos y hojas
stem(y)
par(mfrow=c(2,2)); m<-mean(par("usr")[1:2]) # medidas ventana usuario

# Diag. de cajas
boxplot(y); boxplot(y~y.f); boxplot(split(y,y.f),col="cyan")

# diagrama de puntos, tres métodos
stripchart(y); text(m, 1.04, "stripchart método overplot")
stripchart(y,method="jitter",add=T,at=1.2); text(m,1.35,"método jitter")
stripchart(round(y,1),method="stack",add=T,at=0.7); text(m,0.85,"método
stack")
```

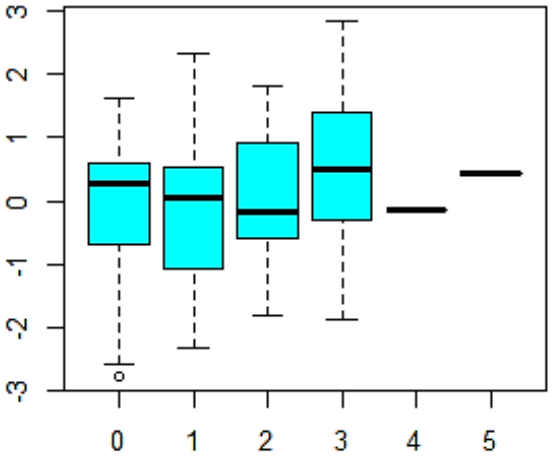
Resultados

The decimal point is at the |

-2 | 863
-1 | 9988876443332211110
-0 | 98777777666555444332221
 0 | 011112223333333344444555557788999 9
 1 | 001112223356677888
 2 | 38







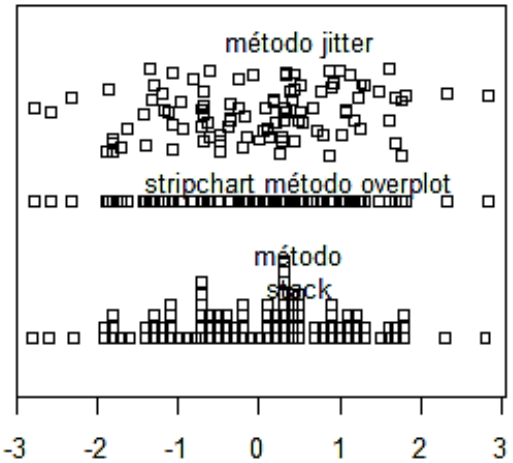
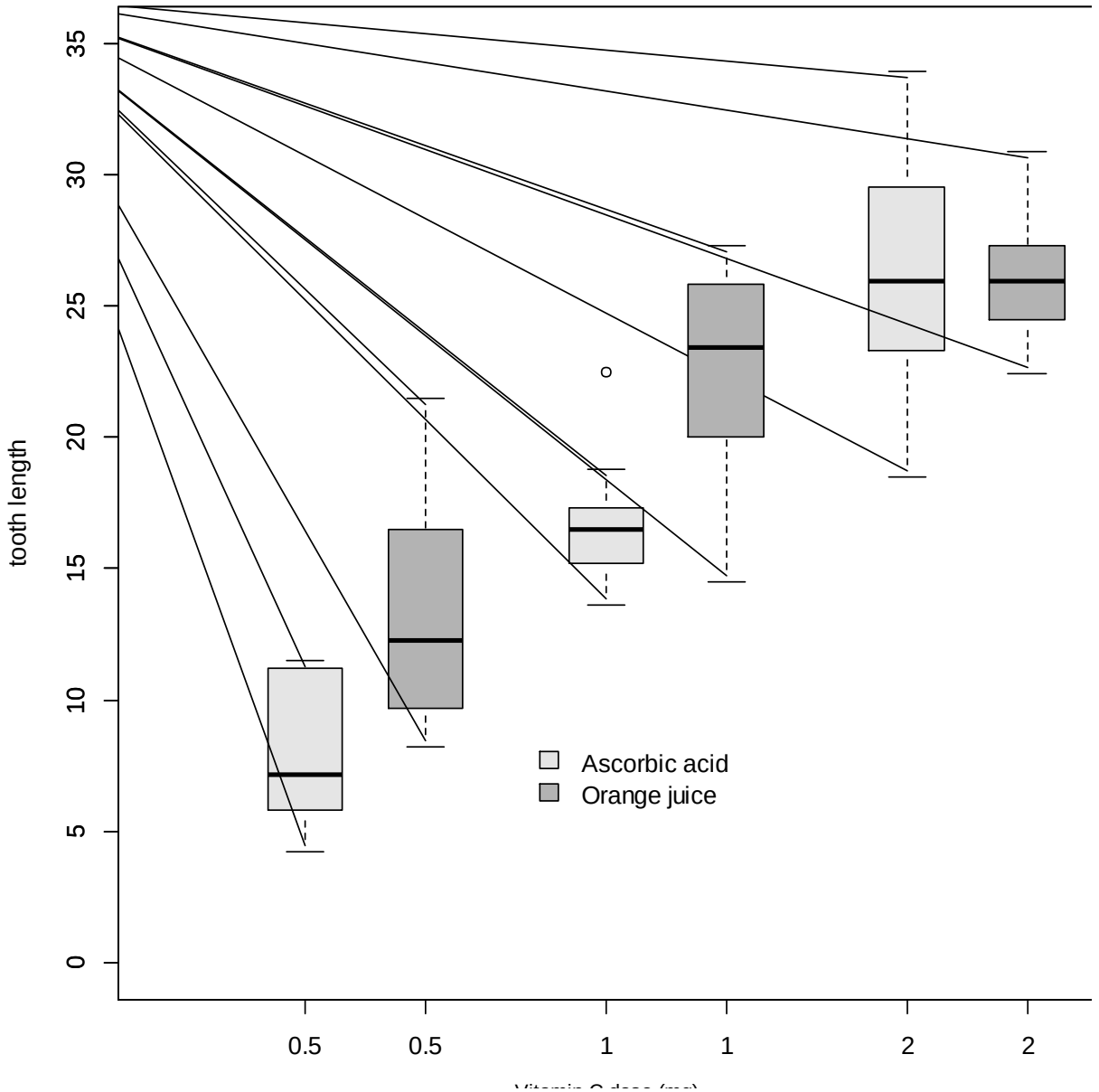


Diagrama de caja

```
par(mar=c(3, 4.1, 2, 0))
    boxplot(len ~ dose, data = ToothGrowth,
            boxwex = 0.25, at = 1:3 - 0.2,
            subset= supp == "VC", col="grey90",
            xlab="",
            ylab="tooth length", ylim=c(0,35))
    mtext("Vitamin C dose (mg)", side=1, line=2.5, cex=0.8)
    boxplot(len ~ dose, data = ToothGrowth, add = TRUE,
            boxwex = 0.25, at = 1:3 + 0.2,
            subset= supp == "OJ", col="grey70")
    legend(1.5, 9, c("Ascorbic acid", "Orange juice"), bty="n",
           fill = c("grey90", "grey70"))
par(mar=c(5.1, 4.1, 4.1, 2.1))
```

Resultados


tooth length
0
5
10
15
20
25
30
35
0.5
0.5
1
1
2
2
Ascorbic acid
Orange juice

Representación de datos multivariantes

Cuando queremos representar varias variables conjuntamente para detectar relaciones entre ellas, disponemos de diversos tipos de gráficos:

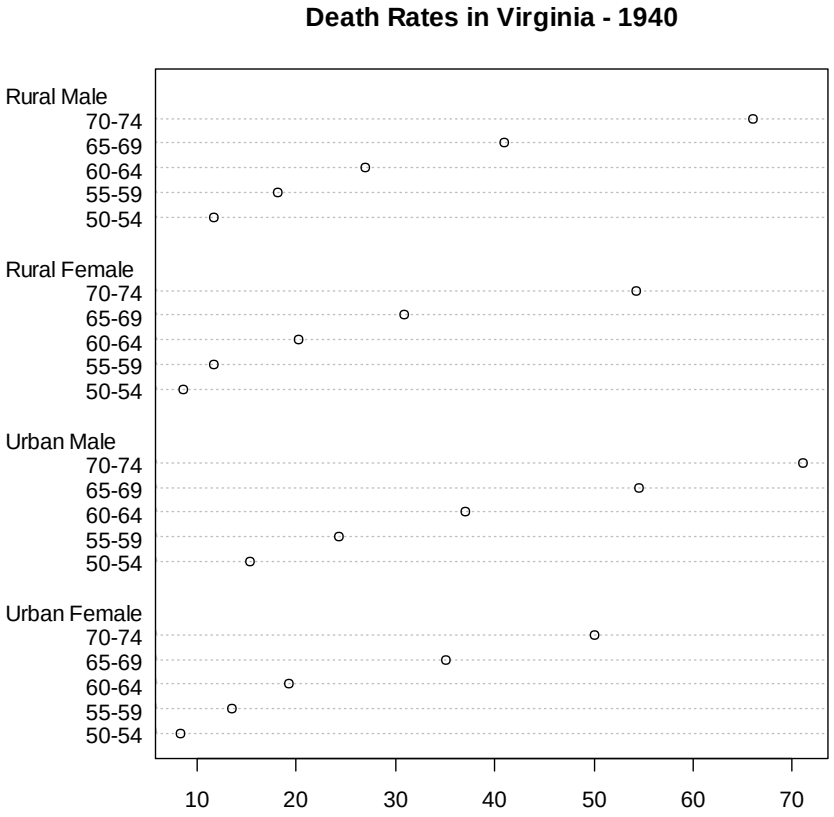
Los más habituales

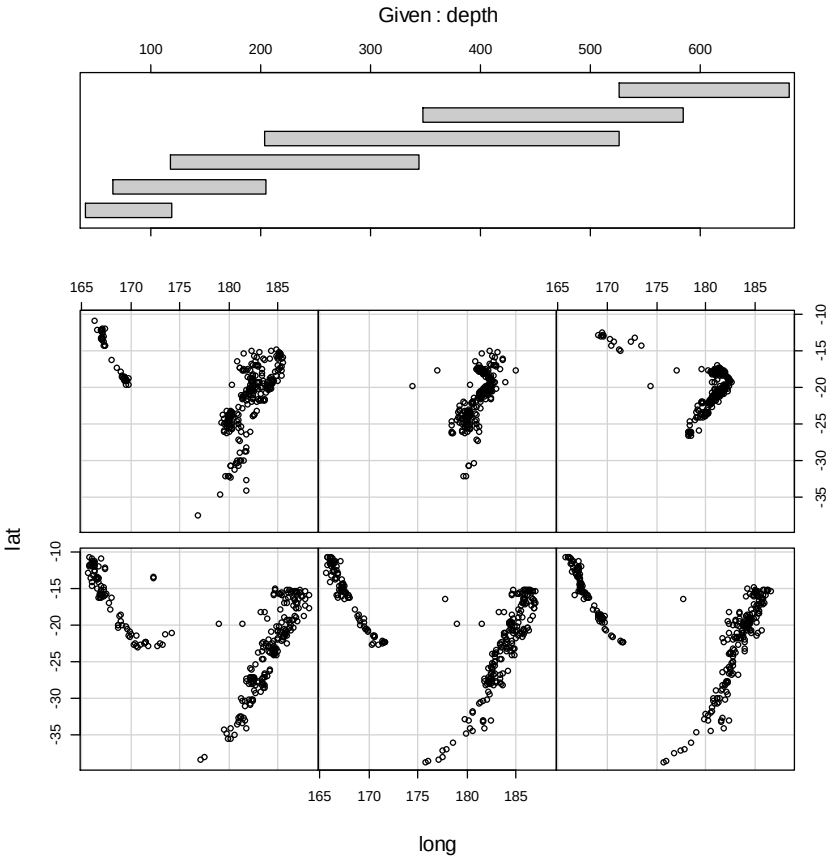
- Gráficos de tendencias para tablas de contingencia: `dotchart()`
- Gráficos de dispersión: `plot()` y `pairs()`
- Gráficos condicionados: `coplot()`.

Gráficos de tendencias para tablas de contingencia

```
data(VADeaths)
dotchart(VADeaths, main = "Death Rates in Virginia - 1940")

# Gráficos condicionados
data(quakes)
coplot(lat~long | depth, data = quakes)
```

Resultados


Death Rates in Virginia - 1940
Rural Male
70-74
65-69
60-64
55-59
50-54
Rural Female
70-74
65-69
60-64
55-59
50-54
Urban Male
70-74
65-69
60-64
55-59
50-54
Urban Female
70-74
65-69
60-64
55-59
50-54
10
20
30
40
50
60
70

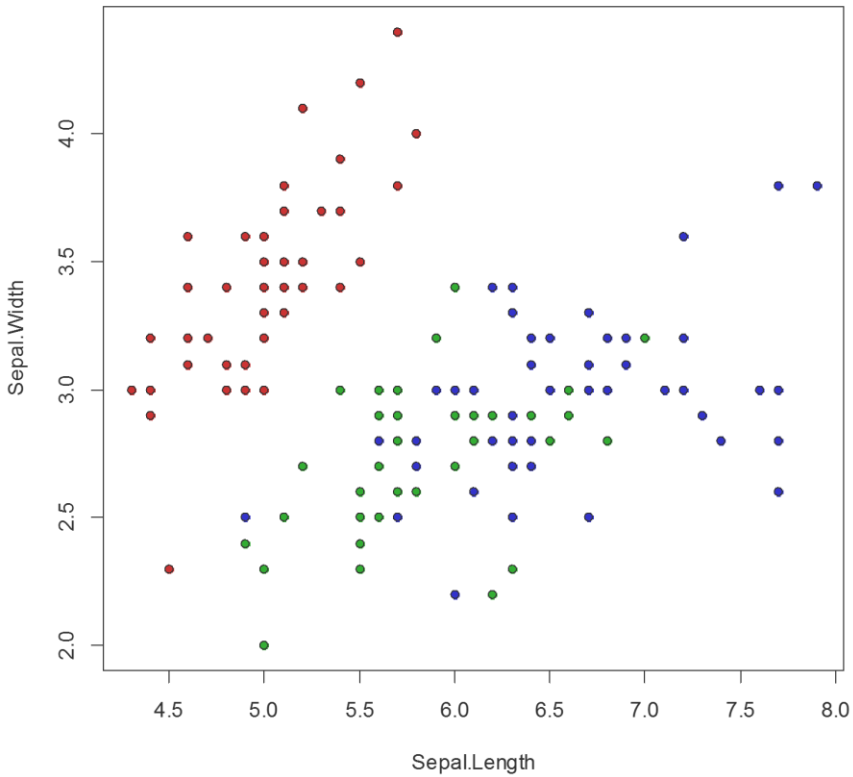

Given : depth
100
200
300
400
500
600
165
170
175
180
185
-10
-15
-20
-25
-30
-35
lat
long

Gráficos de dispersión para revisar relaciones entre variables

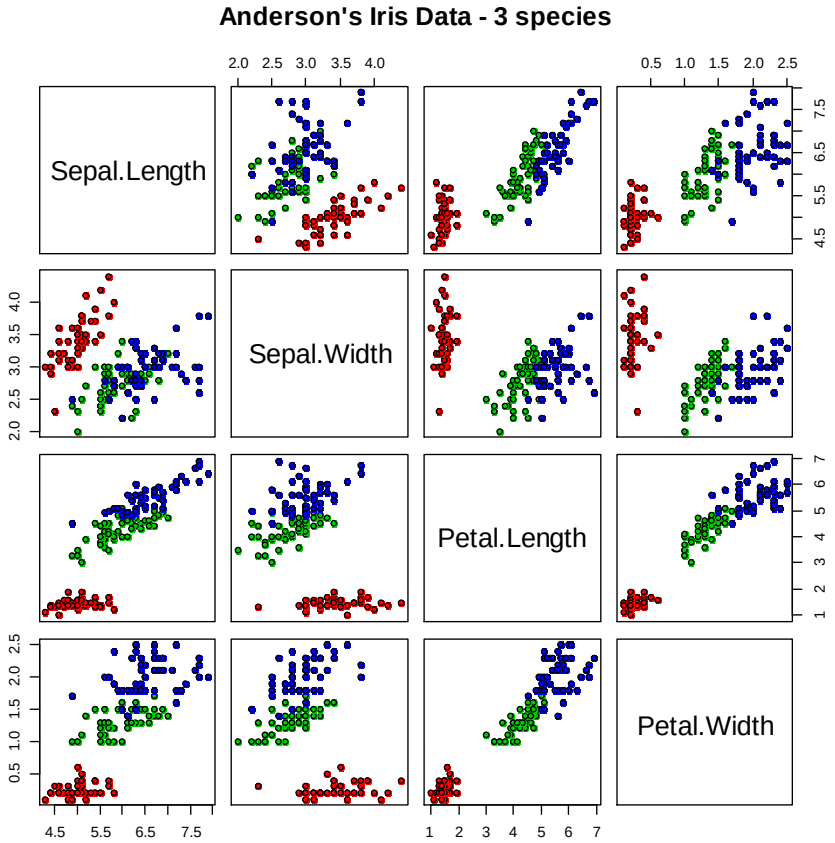
```
X <- matrix(rnorm(1000), ncol = 2); colnames(X) <- c("a", "b")
plot(X)

X <- matrix(rnorm(1000), ncol = 5)
colnames(X) <- c("a", “id", "edad", "loc", "peso")
pairs(X)

data(iris)
razas<-unclass(iris$Species)
plot(iris[1:2],pch=21,bg = c("red", "green3", "blue")[razas])

pairs(iris[1:4], main = "Anderson's Iris Data - 3 species", pch = 21, bg
= c("red", "green3", "blue")[razas])

data(swiss)
pairs(swiss, panel = panel.smooth, lwd = 2, cex= 1.5, col="blue")
```

Resultados 1/2

Diagrama de Dispersión

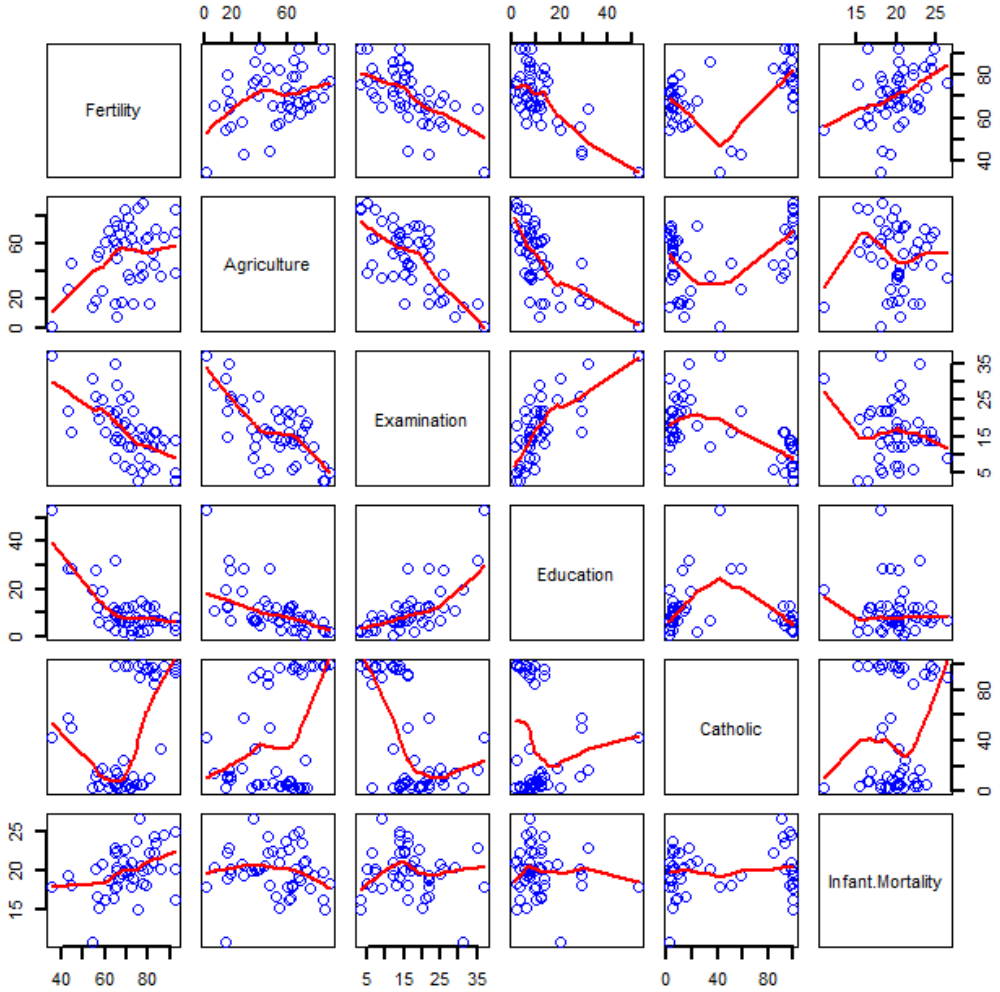


Matriz de Dispersión



Resultados 2/2

Matriz de Dispersión



Gráficos para estudiar la distribución de unos datos

Cuando queremos estudiar cual es la posible distribución de unos datos disponemos de diferentes funciones:

Los más habituales

- Histogramas: `hist()`
- Gráficos *qq*: `qqplot()`, `qqnorm()` y `qqline()`. Dos posibles usos:
  - Comparación de cuantiles empíricos versus cuantiles teóricos: para comprobar si los datos se parecen a una determinada distribución
  - Comparación de dos distribuciones empíricas entre sí
- Estimación de la función de distribución empírica: `ecdf()`
- Estimación kernel de la función de densidad: `density()`

Histograma

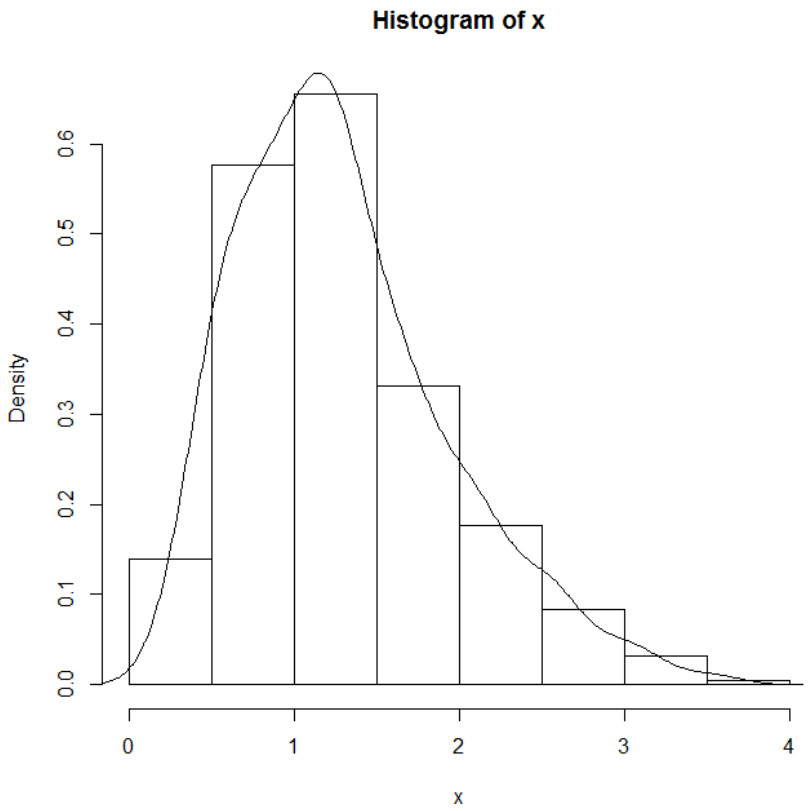
```
y<-rnorm(500); hist(y); hist(y,5)
```

Función de densidad

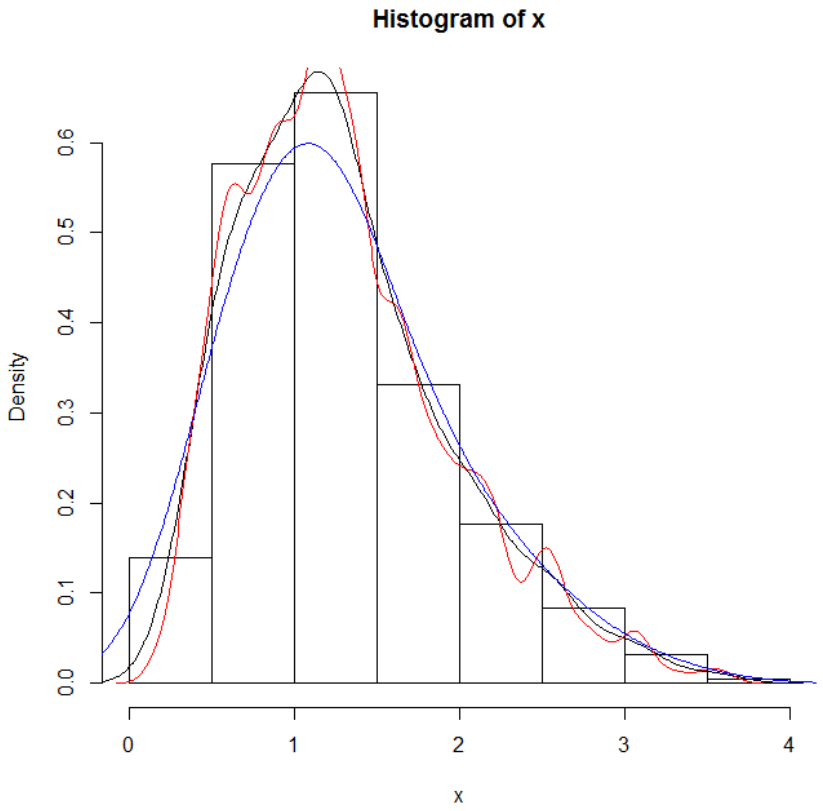
```
x<-rgamma(500,4,3)
hist(x,prob=T) # prob=T equivale a freq=F

# Estimador kernel de la densidad
lines(density(x))
bw.x<-density(x)$bw
bw.x # amplitud de la banda

# Podemos modificar la banda para la estimación
lines(density(x,bw=bw.x/2),col=2)
lines(density(x,bw=bw.x*2),col=4)
```

Resultados

Densidad de x con función de de



Densidad de x modificada

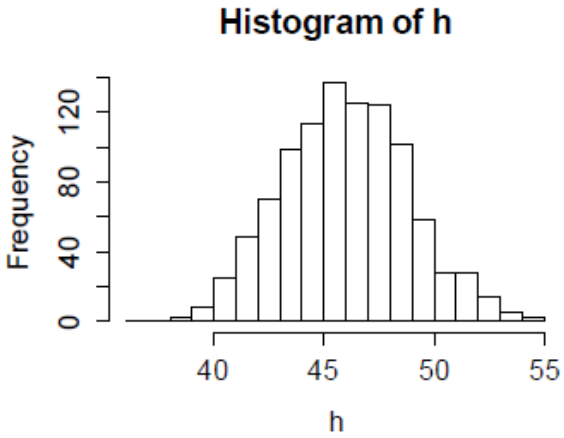


Estimación de funciones de densidad

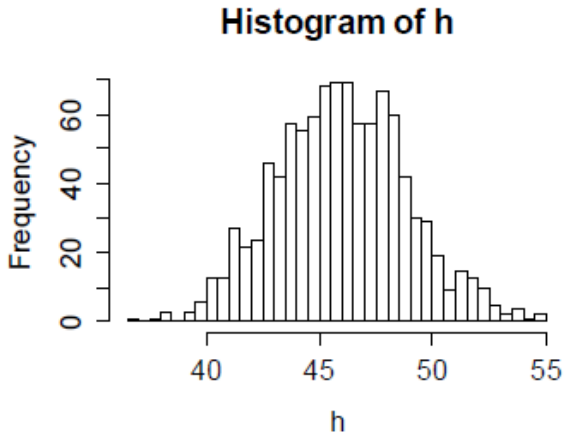
Opciones específicas de los HISTOGRAMAS

Utilizando la opción breaks

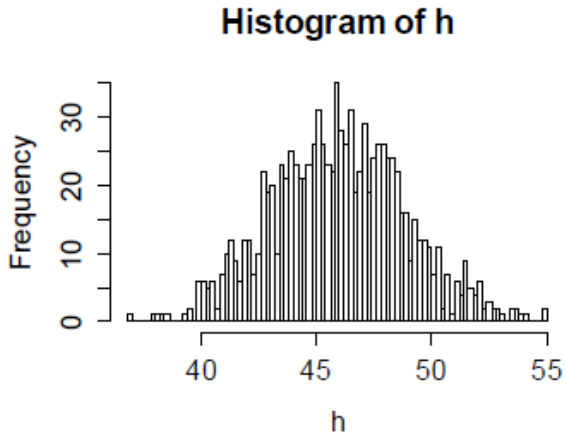
```
hist(h, breaks=15)
```




```
hist(h, breaks=30)
```




```
hist(h, breaks=70)
```

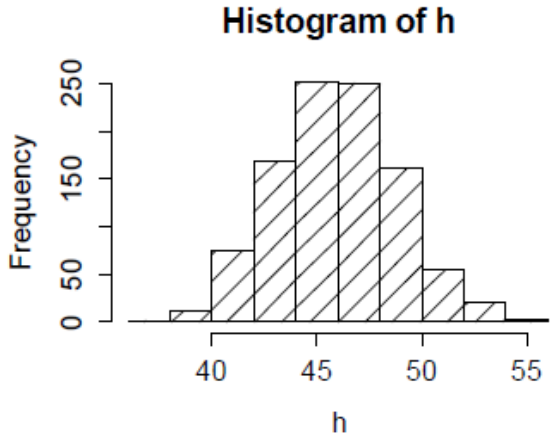


Estimación de funciones de densidad

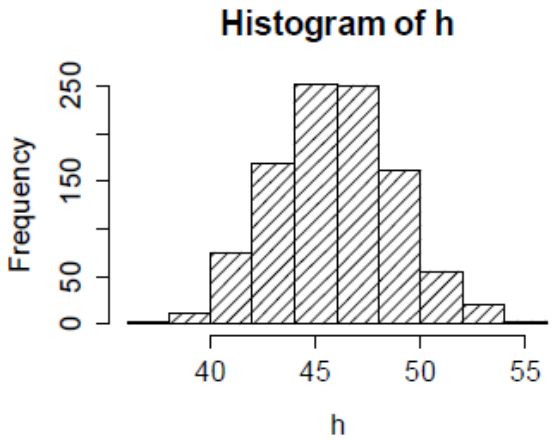
Opciones específicas de los HISTOGRAMAS

Utilizando la opción density

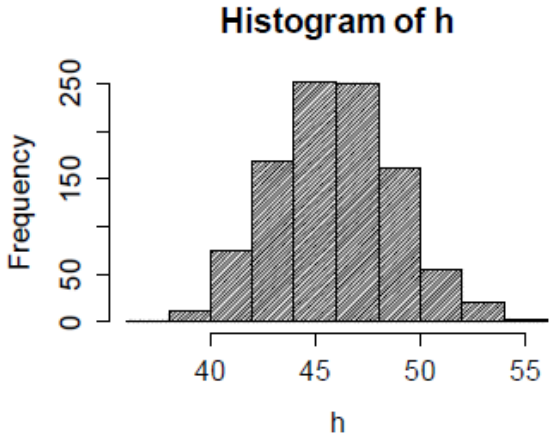
```
hist(h, density=5)
```




```
hist(h, density=10)
```




```
hist(h, density=30)
```

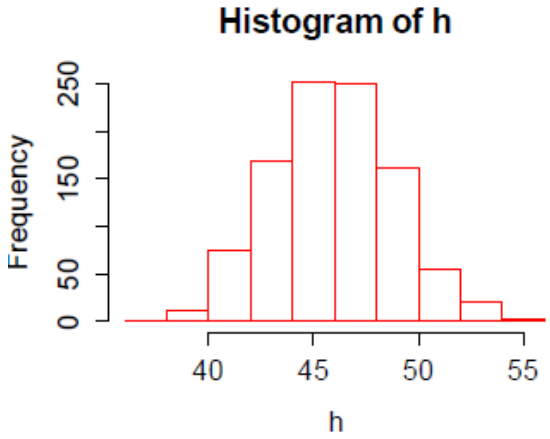


Estimación de funciones de densidad

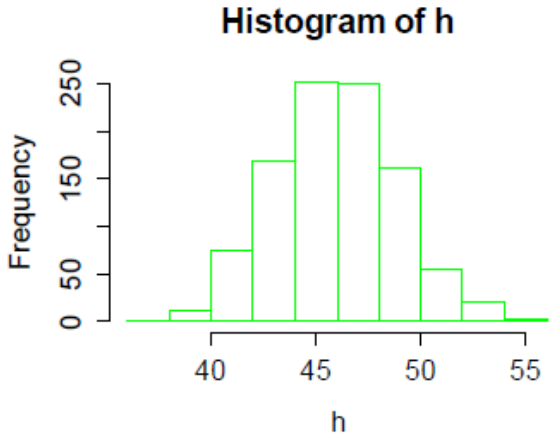
Opciones específicas de los HISTOGRAMAS

Utilizando la opción border

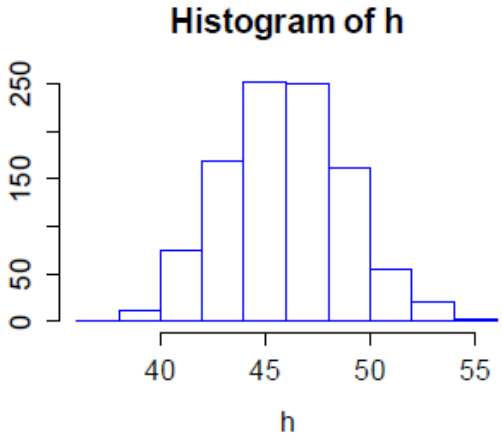
```
hist(h, border="red")
```




```
hist(h, border="green")
```




```
hist(h, border="blue")
```

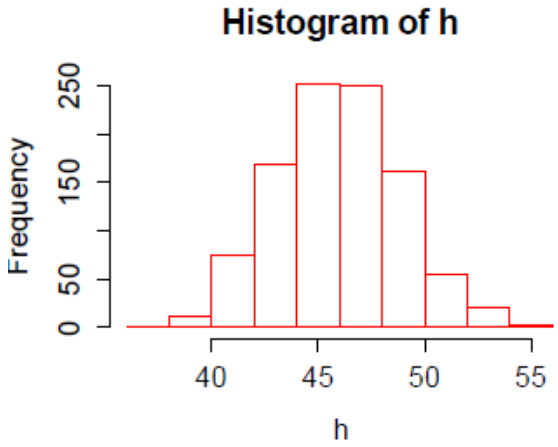


Estimación de funciones de densidad

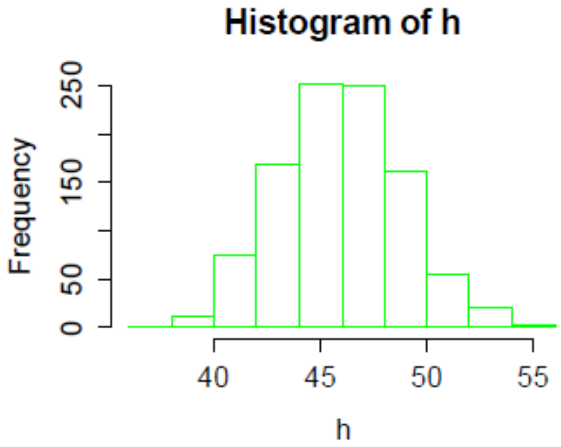
Opciones específicas de los HISTOGRAMAS

Utilizando la opción border

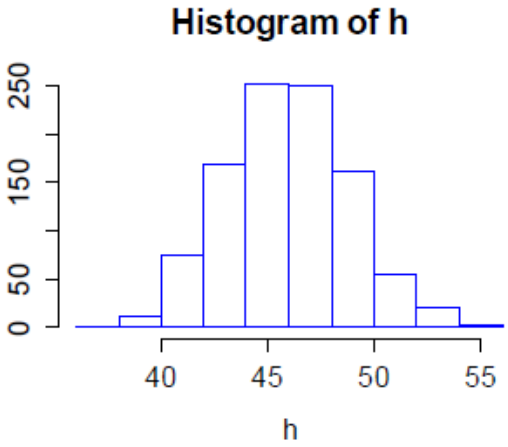
```
hist(h, border="red")
```




```
hist(h, border="green")
```




```
hist(h, border="blue")
```

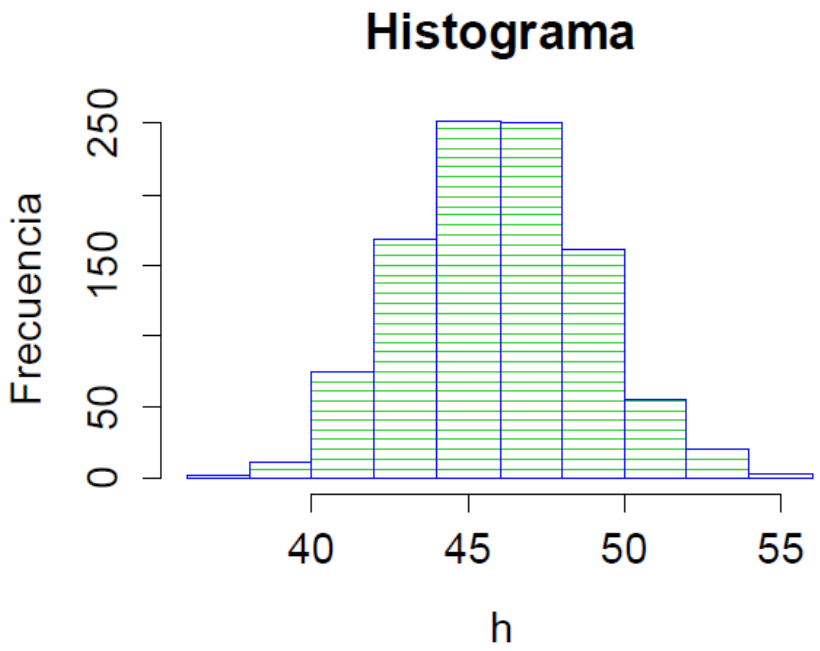


Estimación de funciones de densidad

Opciones específicas de los HISTOGRAMAS

Las opciones que hemos visto (y más) se pueden combinar

```
hist(h,
     col = "limegreen",
     border = "blue",
     density = 15,
     angle = 180,
     main = "Histograma",
     ylab = "Frecuencia"
     )
```

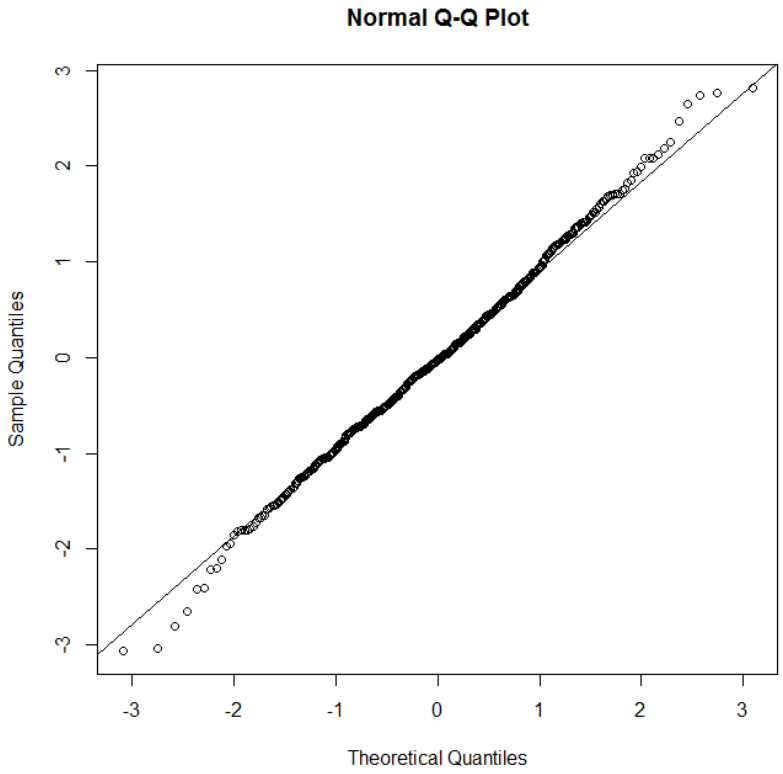


Gráficos qq

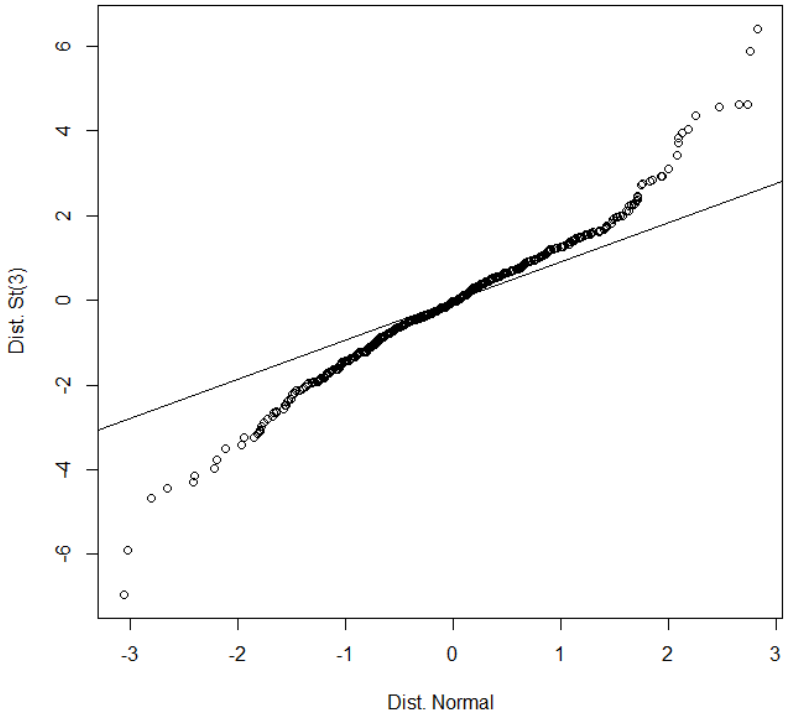
```
y<-rnorm(500)
# Comparación de los cuantiles muestrales con los de una Normal
qqnorm(y); qqline(y)

# Comparación de los cuantiles muestrales de dos muestras
y.t<-rt(500,3)
qqplot(y,y.t,xlab="Dist. Normal", ylab="Dist. St(3)"); qqline(y)

# Comparar cuantiles muestrales con los de una distribución dada
library(lattice)
qqmath(y,distribution=function(p){qt(p,df=5)})
qqmath(y,distribution=function(p){qgamma(p,shape=3,rate=5)})
```

Resultados 1/2

Comparación de los cuantiles muestrales con la normal

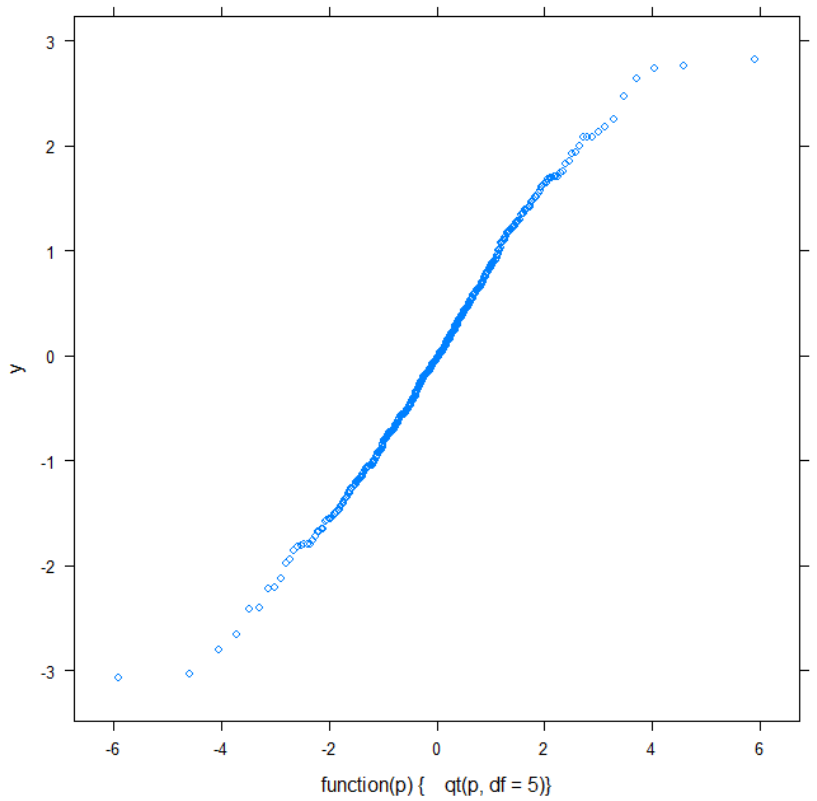


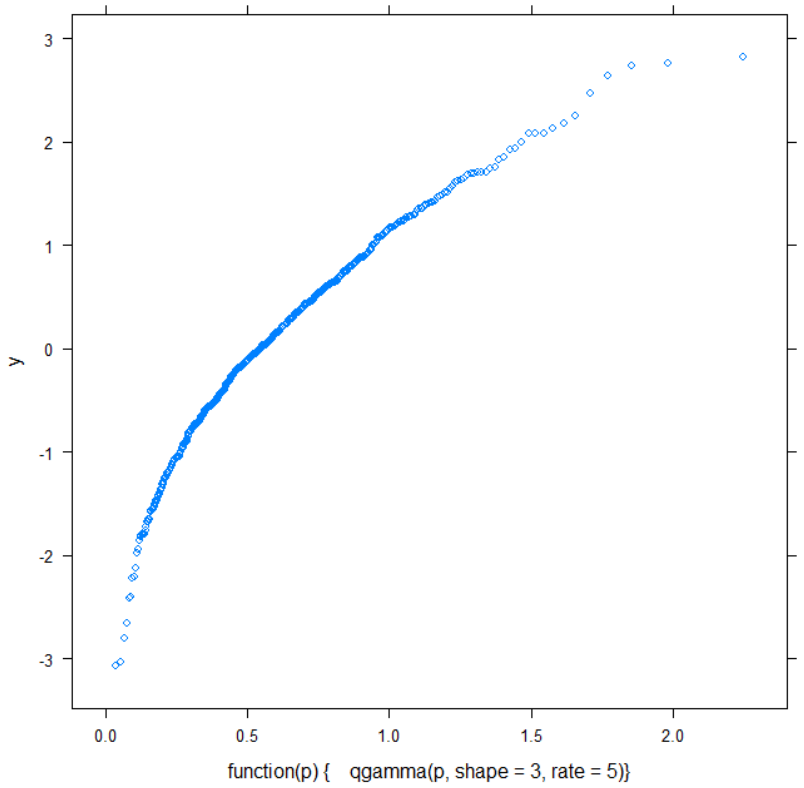
Comparación de los cuantiles muestrales de dos muestras



Resultados 2/2

Comparación de los cuantiles muestrales con los de una distribución dada

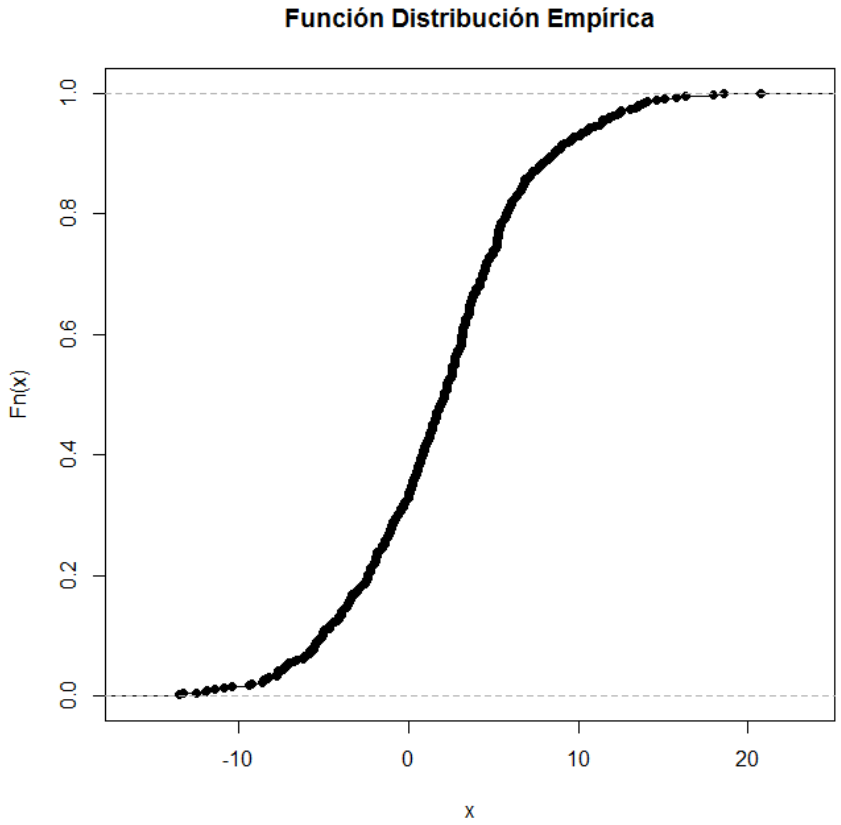


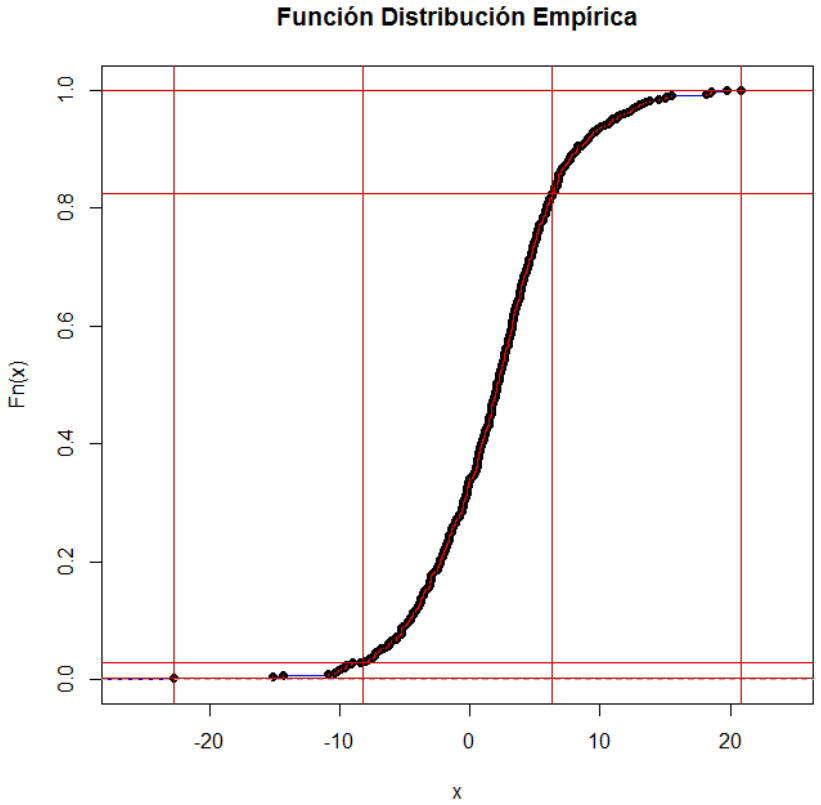


Función de distribución empírica

```
x<-rlogis(500,2,3)
Fn.x<-ecdf(x); summary(Fn.x)
plot(Fn.x,main="Función Distribución Empírica")
plot(Fn.x,add=T,verticals=T,col.v=2,col.h=4)
x.seq<-seq(min(x),max(x),length=4)

# Para evaluar Fn en cualquier punto
points(x.seq,Fn.x(x.seq),pch=21,bg="red3")
abline(v=x.seq,col="red3")
abline(h=Fn.x(x.seq),col="red3")
```

Resultados


Función Distribución Empírica
Fn(x)
0.0
0.2
0.4
0.6
0.8
1.0
-10
0
10
20
x


Función Distribución Empírica
Fn(x)
0.0
0.2
0.4
0.6
0.8
1.0
-20
-10
0
10
20
x

Representación gráfica en 3D

Cuando queremos representar una función bivariante disponemos de diversos tipos de gráficos:

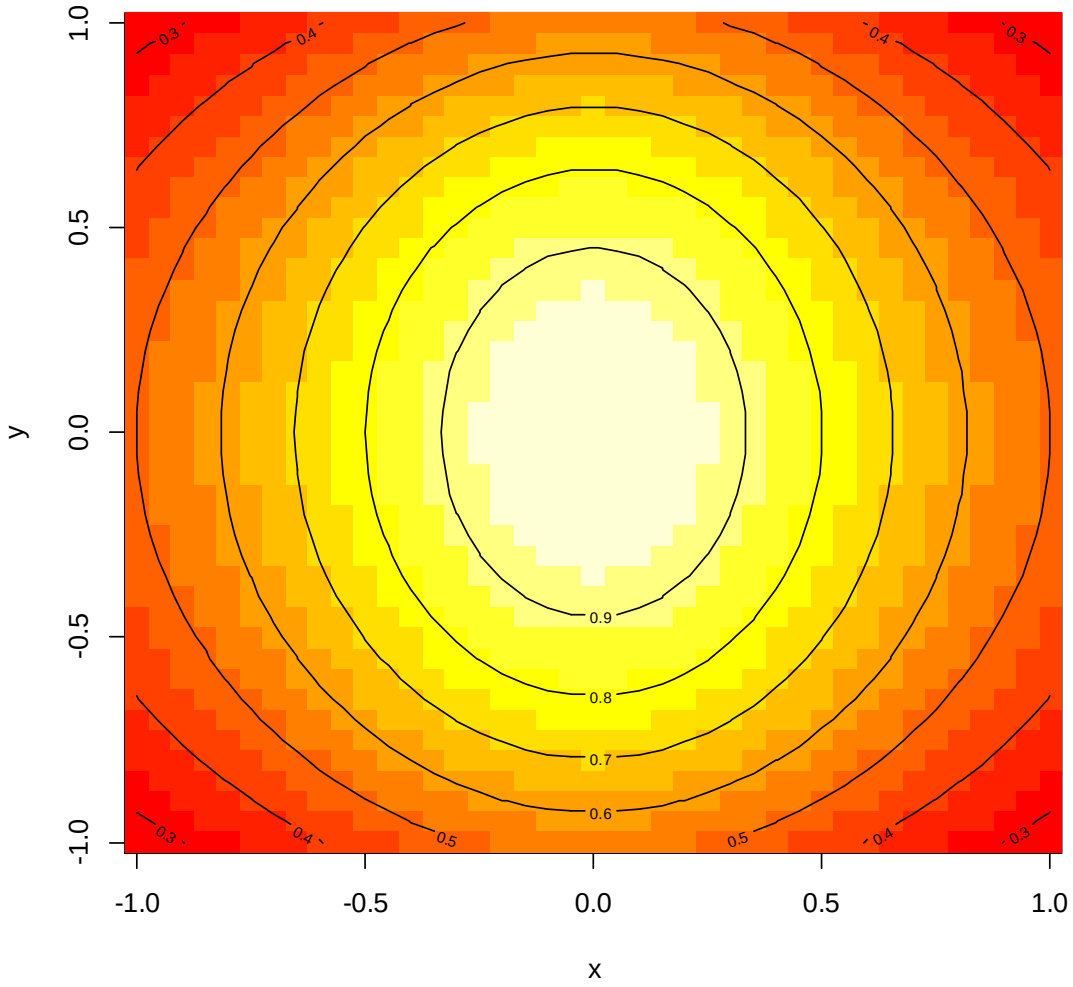
Los más habituales

- Gráficos en tres dimensiones: `image()`
- Gráficos de contorno: `contour()`. Permite añadir líneas de nivel.
- Las librerías `MASS` y `ks` tienen funciones para estimar kernels bivariantes.

Funciones gráficas 3D

```
x<-seq(-1,1,0.05); y<-seq(-1,1,0.05)
f <- function(x, y) cos(y)/(1 + x^2)
z <- outer(x, y, f)
image(x,y,z); contour(x,y,z,add=T)
```

Gráfico en 3D



Dispositivos gráficos (device drivers)

- Por defecto, cuando realizamos la primera gráfica, R abre un dispositivo gráfico.
- Ponemos abrir nuevas ventanas gráficas llamando a la función windows(). Con ello tendremos varios dispositivos donde dibujar.
- Para cerrar un dispositivo abierto utilizamos dev.off(). Si no tenemos claro cual cerrar, la función dev.list() nos puede ayudar a saber qué dispositivos hay abiertos y que numeración tienen.
- Siempre hay uno activo, podemos saber cuál es con dev.cur(). Si queremos activar otro podemos utilizar dev.set().
- Con la opción histórico grabando activa R nos permite disponer de todos los gráficos e ir accediendo al resto con Av.Pág. y Re.Pág.

Exportando gráficos

- Para guardar una gráfica, podemos copiar y pegar desde la ventana gráfica a un tratamiento de textos que lo permita.
- Desde el menú Archivo -> Guardar como podemos guardar la gráfica como un archivo metafile, pdf, png, bmp, postscript, tif o jpg.
- Sin embargo esta opción no es la mejor ya que no tenemos control sobre la propia gráfica y como queda guardada. Sobre todo a nivel de escala.
- Lo mejor es enviar directamente la gráfica a un dispositivo (pdf, postscript, etc.) utilizando funciones como pdf() o postscript().

Exportando gráficos

```
pdf("prueba.pdf", paper="special", width=13, height=7)
hist(x<-rnorm(100),prob=T)
dev.off()
```

Links útiles para gráficos

http://www.stat.auckland.ac.nz/~paul/RGraphics/rgraphics.html

http://addictedtor.free.fr/graphiques/

http://addictedtor.free.fr/graphiques/thumbs.php?sort=votes

http://www.statmethods.net/advgraphs/layout.html

Comunicación constante con la Escuela del INEI

Correo de la Dirección Técnica de la ENEI
Sr. Eduardo Villa Morocho (Eduardo.villa@inei.gob.pe)

Coordinación Académica
Sra. María Elena Quirós Cubillas (Maria.Quiros@inei.gob.pe)

Correo de la Escuela del INEI
enei@inei.gob.pe

Área de Educación Virtual
Sr. Gonzalo Anchante (gonzalo.anchante@inei.gob.pe)