Ejercicios. Programa estadístico R para el Análisis de Datos

*Miguel A. Mayer*

*Mayo 2018*

Ejercicio 1. Explorando el entorno

Abrir los programas R y RStudio y explorar áreas de trabajo de RStudio y menús.

Ejercicio 2. Calculadora y primeros comandos

RStudio como calculadora.

*# En R podemos incluir comentarios mendiante el signo #*

134 + 35 *# suma*

## [1] 169

45 - 34 *# resta*

## [1] 11

200 \* 4.5 *# multiplicación*

## [1] 900

50 / 3.6 *# división*

## [1] 13.88889

3 \*\* 2 *# potencia*

## [1] 9

Primeros comandos en R en el entorno de trabajo de RStudio.

getwd()

## [1] "/Users/mmayer/Documents/Mayer/Curso\_R\_Web"

dir()

## [1] "CursoR\_coib.docx" "CursoR\_coib.html"

## [3] "CursoR\_coib.pdf" "CursoR\_coib.Rmd"

## [5] "CursoR\_files" "CursoR.html"

## [7] "Directorio.png" "EjerciciosR.html"

## [9] "EjerciciosR.pdf" "EjerciciosR.Rmd"

## [11] "Introduccion\_R\_2018.docx" "Introduccion\_R.docx"

## [13] "pacientes.txt" "rsconnect"

## [15] "RStudio.jpg" "RStudio.png"

## [17] "RStudio2.png" "variables.png"

ls()

## character(0)

Ejercicio 3. Creación y manejo de variables.

Creamos dos variables (peso y altura) para el cálculo y creación de otra variable IMC (variable calculada)

peso <- 56 *# peso en kilos*

peso *# vemos su contenido*

## [1] 56

altura <- 1.61 *# altura en metros*

altura

## [1] 1.61

IMC <- peso / (altura\*\*2) *# cálculo IMC*

IMC *# vemos el valor de IMC*

## [1] 21.6041

Creación de las variables “genero” y “edad” y del conjunto de datos “mispacientes”

*# crear la variable genero*

genero <- c("h", "m", "m", "h","m","h","m", "h", "m", "m")

*# creamos la variable edad*

edad <- c(23,45,34,39,60,52,44,59,32,40)

mispacientes <- data.frame(genero, edad)

*# ordenar todos los datos por la variable edad*

mispacientes[order(mispacientes$edad),]

## genero edad

## 1 h 23

## 9 m 32

## 3 m 34

## 4 h 39

## 10 m 40

## 7 m 44

## 2 m 45

## 6 h 52

## 8 h 59

## 5 m 60

Ejercicio 4.

Crear dos nuevas variables e incorporarlas en el data.frame mispacientes. Las nuevas variables deben ser glucemia basal con los siguientes valores: 123,230,180,164,210,220,193,202,152,185 y tratamiento con antidiabéticos orales: no, si, no, no, si, no, si, si, no, no.

Ejercicio 5. Descripción de datos cuantitativos

En este ejercicio aplicamos diversas funciones de estadística descriptiva para el conjunto de datos de la variable “edad”

max(edad) *# nos muestra el valor máximo de edad*

## [1] 60

min(edad) *# nos muestra el valor mínimo de edad*

## [1] 23

mean(edad) *# nos muestra la media*

## [1] 42.8

sd(edad) *# nos indica la desviación estándar*

## [1] 11.82089

round(sd(edad),2) *# redondea la desviación estándar a 2 decimales*

## [1] 11.82

quantile(edad) *# nos muestra los cuantiles*

## 0% 25% 50% 75% 100%

## 23.00 35.25 42.00 50.25 60.00

Ejercicio 5. Representar gráficamente las variables “edad” y “genero”.

Ejercicio 6. Variable edad en intervalos

Transformación de la variable colesterol en determinados intervalos.

mispacientes$colesterol <- c(230,241,190,150,201,278,262,190,220,184 )

mispacientes$colesterolrango[mispacientes$colesterol <= 200] <- '<=200'

mispacientes$colesterolrango[mispacientes$colesterol > 200 & mispacientes$colesterol <= 260] <- '201-260'

mispacientes$colesterolrango[mispacientes$colesterol >260] <- '>260'

Ejercicio 7. Importar el fichero pacientes.txt disponible en:

Dirigirse a la web siguiente, copiando la URL (copiar y pegar) en el navegador:

<https://bit.ly/2FBNeAs>

Descargar el archivo en el ordenador, preferiblemente en el directorio de trabajo y guardarlo.

Posteriormente se importa a RStudio.

Ejercicio 8. Selección de diversos elementos de un conjunto de datos

names(mispacientes) *# mostramos las variables de mispacientes*

## [1] "genero" "edad" "colesterol" "colesterolrango"

names(mispacientes)[3] *# mostramos la tercera variable "colesterol"*

## [1] "colesterol"

head(mispacientes) *# muestra los 6 primeros casos*

## genero edad colesterol colesterolrango

## 1 h 23 230 201-260

## 2 m 45 241 201-260

## 3 m 34 190 <=200

## 4 h 39 150 <=200

## 5 m 60 201 201-260

## 6 h 52 278 >260

tail(mispacientes) *# muestra los últimos 6 casos*

## genero edad colesterol colesterolrango

## 5 m 60 201 201-260

## 6 h 52 278 >260

## 7 m 44 262 >260

## 8 h 59 190 <=200

## 9 m 32 220 201-260

## 10 m 40 184 <=200

mispacientes$genero

## [1] h m m h m h m h m m

## Levels: h m

mispacientes[3,2]

## [1] 34

mispacientes[3,2:3]

## edad colesterol

## 3 34 190

mispacientes[c(3,5), 2:3]

## edad colesterol

## 3 34 190

## 5 60 201

mispacientes[, 2:3]

## edad colesterol

## 1 23 230

## 2 45 241

## 3 34 190

## 4 39 150

## 5 60 201

## 6 52 278

## 7 44 262

## 8 59 190

## 9 32 220

## 10 40 184

mispacientes[2,]

## genero edad colesterol colesterolrango

## 2 m 45 241 201-260

mispacientes[2:4,]

## genero edad colesterol colesterolrango

## 2 m 45 241 201-260

## 3 m 34 190 <=200

## 4 h 39 150 <=200

mispacientes[c(3,5),2]

## [1] 34 60

Ejercicio 9. Comparación de proporciones

Estudio sobre la frecuencia de fumadores en tres grupos de pacientes con pesos diferentes (obesos, sobrepeso, normal)

fumadores <- matrix(c(23, 134, 34, 167, 41, 130), ncol=2)

colnames(fumadores) <- c("Fumadores", "No fumadores")

rownames(fumadores) <- c("Obesos", "sobrepeso", "normal")

fumadores

## Fumadores No fumadores

## Obesos 23 167

## sobrepeso 134 41

## normal 34 130

chisq.test(fumadores)

##

## Pearson's Chi-squared test

##

## data: fumadores

## X-squared = 188.46, df = 2, p-value < 2.2e-16

# Guía de ejercicios básicos de R

#### Marcos Paolinelli

#### 29 de Noviembre de 2018

#### HOLA MUNDO

print("Hola MUNDO")

#### para obtener ayuda de cualquier función usamos

?print

#### lo podemos usar como calculadora

1+1+2+3+5+8

8-13

13\*21

34\*34

55/34 \* sqrt(89)

#### Para guardar alguno de estos resultados necesitamos asignarlos a un “objeto”. Para esto podemos usar el símbolo “=” o “<-”

x = 2+1 #x <- 2+1

x

#### Se pueden hacer operaciones directamente con los objetos

otro\_numero = 55/34 \* sqrt(89)

otro\_numero

otro\_numero\*x

#### para eliminar un objeto usamos rm

rm(otro\_numero)

otro\_numero

#### Para crear un objeto con más de un elemento usamos c()

Ventas = c(1,1,2,3,5,8,13)

Ventas

#### R posee un método recursivo que nos permite realizar una operación sobre el objeto múltiples veces

Ventas + 5

#### También podemos usar R con funciones. Una función es una palabra seguida por paréntesis word()

sum(Ventas)

mean(Ventas)

max(Ventas)

min(Ventas)

range(Ventas)

unique(Ventas)

#### Para ordenar usamos sort()

sort\_increase=sort(Ventas)

## **OBJETOS Y VECTORES**

frutas = c("manzana" , "banana" , "sandía" , "ananá" , "limón" , "frutilla" , "naranja")

#### para asignar nombres a elementos en un objeto existe la función names()

names(Ventas) = frutas

Ventas

## Quick Examples of names() Function

#### # Create a vector

#### vec <- c(10,30,50,70,90)

#### # Set name to Vector

#### names(vec) <- c('a','b','c','d','e')

#### # Get name of vector

#### names(vec)

#### # Remove vector name

#### names(vec) <- NULL

#### # Create list object

#### li <- list('Spark','R','Java','Python')

#### # Assign name to the list

#### names(li) <- c('AA','AB','AC','AD')

#### # Get name of the list object

#### names(li)

#### #Change names of the DataFrame

#### names(df) <- c('A','B','C','D','E')

#### ¿Qué pasó?

#### El objeto Ventas es un vector. El vector posee múltiples elementos en una determinada posición que se indica mediante el empleo de los corchetes: Ventas[posición]

Ventas[3]

#### Múltiples series de elementos pueden ser obtenidos al mismo tiempo. En este caso necesitamos usar c()

Ventas[c(1,3:7)]

#### Y si por el contrario, queremos eliminar esos elementos:

Ventas[-c(1,5)]

#### EJERCICIO: ¿CUál es la media de frutas vendidas sin contar “frutilla”? Utilizar lo aprendido. El resultado debe ser 4.166667

## **OPERADORES RELACIONALES**

#### A veces queremos saber características de los números. Por ejemplo: El número de tiburones con longitud mayor a 2 metros. Esto se puede responder con operadores relacionales (más grande que, más largo que, igual a, etc.)

#### Con el objeto ventas. ¿Qué frutas se vendieron más de 5 veces?

Ventas>5

Ventas == 5

#### En el caso anterior usamos “==” para evitar asignar 5 al objeto Ventas. Como estas comparaciones nos dan vectores lógicos TRUE/FALSE o 1/0. Esto puede resultar muy útil para hacer algunos cálculos:

sum(Ventas > 5)

Ventas[Ventas > 5]

#### EJERCICIO: ¿Cómo sabemos que fruta se vendió más caro que la media?

## **AHORA VAMOS A HACER GRÁFICOS!!!**

#### R es muy popular por la calidad y variedad de gráficas que puede generar. Una vez que los datos están en un objeto, graficarlo es muy simple. Por ejemplo: grafiquemos las frutas vendidas

plot(Ventas)

#### Si queremos otro tipo de gráficas

barplot(Ventas)

pie(Ventas)

hist(Ventas)

#### Podemos ver todas las opciones para cada tipo de gráfico usando “?”, pero sigamos con ejemplos

hist(Ventas, col="skyblue", main="Histograma en azúl", ylab="Frutas vendidas")

#### Existen varias maneras de guardar gráficas. Por ejemplo: como pdf, pero primero definimos la carpeta de trabajo

getwd()

setwd("./curso/")

pdf("histogram.pdf")

hist(Ventas, col="skyblue", main="Histograma en azúl", ylab="Frutas vendidas")

dev.off()

## **LECTURA DE ARCHIVOS Y MANIPULACIÓN DE TABLAS**

#### Cuando ya tenemos valores en un archivo de texto, podemos importarlo a R, para editarlo. Buscar archivo “file1.txt”" en el directorio de trabajo. Deben primero descargar desde el [drive](https://drive.google.com/drive/folders/1NcxBVwjK3KZLsw4svn_cOa4dN_-0KiOw?usp=sharing):

setwd("./<Especificar\_directorio/")

getwd()

list.files()

#### sin preocuparnos por el formato del archivo podemos revisarlo con scan. PUEDEN USAR TAB para AUTOCOMPLETAR

num = scan("../file1.txt")

num

#### usemos un ejemplo más realista con el archivo “tab.txt”. Descargar del [drive](https://drive.google.com/drive/folders/1NcxBVwjK3KZLsw4svn_cOa4dN_-0KiOw?usp=sharing)

tab = read.table("tab.txt")

head(tab)

#### existe una forma rápida de visualizar este tipo de datos

?boxplot

boxplot(x = tab, col =c("red","green","blue"))

#### hemos visto previamente que usando “[ ]” podemos restringir los elementos del vector. Podemos hacer lo mismo con tablas pero en este caso necesitamos agregar una coma para referirnos a [lineas,columnas]

tab[c(1,6,10), c(1,3)]

## **EJEMPLOS DE TRABAJO EN DATOS DE EXPRESIÓN**

#### La tabla del archivo “expr.txt” del [drive](https://drive.google.com/drive/folders/1NcxBVwjK3KZLsw4svn_cOa4dN_-0KiOw?usp=sharing) contiene el valor de expresión génica de 10000 genes entre dos condiciones, con tres réplicas cada uno

expr = read.table("expr.txt")

head(expr)

dim(expr)

boxplot(expr, col=rainbow(6), main="Boxplot, datos crudos")

#### ¿Cómo se ve esa gráfica? ¿Qué pasa si graficamos la distribución mediante un histograma?

hist(expr[,1], breaks=50, main=1, col="skyblue", xlab="gene expression")

#### Pero acá sólo obtenemos el histograma para los datos de la primera columna, ¿Cómo hacemos para graficar todas las columnas?. Necesitamos usar un control de estructura, al emplear “n1:n2”, crea una secuencia de n1 a n5 aumentando de a 1

for (i in 1:5) {

print(i)

}

#### Volvamos a los datos, ¿cuantos histogramas necesitamos? Si no nos acordamos las dimensiones podemos usar dim(), nrow() o ncol()

ncol(expr)

#### Ahora si, para producir 6 histogramas usamos:

for (i in 1:ncol(expr)) {

hist(expr[,i], breaks=50, main=i, col="skyblue", xlab="gene expression")

}

#### Esto nos grafica los 6 pero cada uno por separado, para poder graficarlos todos juntos, necesitamos establecer cómo queremos la estructura de los gráficos. La estructura por defecto es mfrow=c(1,1), pero nosotros queremos 2 líneas y 3 columnas. [ mfrow: number of Multiple Figures (use ROW-wise)]

pdf("multiplot.pdf")

par(mfrow = c(3,2))

for (i in 1:ncol(expr)) {

hist(expr[,i], breaks=50, main=i, col="skyblue", xlab="gene expression")

}

dev.off()

#### ¿Cómo se ven esos gráficos? ¿Es similar la distribución de los datos? Si asumimos una distribución idéntica, con algunos valores que se desvían, podemos normalizar usando el valor de la mediana

exprNorm = expr

for (i in 1:ncol(exprNorm)) {

exprNorm[,i] = exprNorm[,i] - median(exprNorm[,i])

}

#### ¿Funcionó? Hagamos una imagen para responder

par(mfrow=c(1,2))

boxplot(expr, col=rainbow(6), main="Raw data")

boxplot(exprNorm, col=rainbow(6), main="Normalized data")

## **COMPARACIÓN DE VALORES ENTRE CONDICIONES**

#### Este experimento es muy simple porque sólo tenemos dos condiciones, por lo que queremos determinar cuales son los genes expresados diferencialente al comparar la condicion “ko”y “wt”.¿Cómo lo podemos hacer? Primero, observamos los datos

head(exprNorm)

#### 1)Necesitamos determinar el valor promedio en ko y wt

#### 2)Comparar el valor promedio de cada gen en ko y wt

#### Para lo primero hacemos una evaluación con sólo algunos datos usando head() y cambiamos de la estructura de data.frame a matrix

mini=head(exprNorm,3)

mini = as.matrix(head(exprNorm, 3))

mini

rownames(mini)

#### rownames nos regresa los nombres de los renglones

for (i in rownames(mini)) {

koMean = mean(mini[i,1:3])

wtMean = mean(mini[i,4:6])

print("#############")

print(i)

print(koMean)

print(wtMean)

}

#### for() es fácil de usar pero, el resultado puede ser difícil de asignar a valores,por lo que no es muy eficiente. Una alternativa eficiente es apply(). Esto nos permite aplicar una función sobre muchos elementos en un sólo paso

apply(mini[,1:3], 1, mean)

apply(mini[,4:6], 1, mean)

#### apply() necesita de 3 elementos

#### 1. Una tabla (como matriz)

#### 2. Especificar las dimensiones (1=rows, 2=columns)

#### 3. La función a aplicar sobre los datos

#### EJERCICIO: Usar apply() para obtener la expresión promedio por condición (6 condiciones), no por gen con todos los datos en exprNorm para todos los genes, obtener un vector que contenga la diferencia entre el valor promedio de ko y wt.

#### Crear el histograma correspondiente

#### ¿Es lo que esperábamos?

exprNorm=as.matrix(exprNorm)

meanko=apply(exprNorm[,1:3],1,mean)

meanwt=apply(exprNorm[,4:6],1,mean)

diff=meanwt-meanko

head(diff)

hist(diff)

#### ahora obtuvimos las diferencias, pero ¿son significativas?

#### El objetivo principal cuando analizamos expresión es detectar los cambios asociados a la expresión génica. Sin embargo grandes cambios no es indicativo de “más confiable”. Necesitamos considerar la variación de los valores dentro de las muestras de la misma condición. Para esto se han desarrollado distintos tests y métodos estadísticos. Lo que queremos verificar mediante el test estadístico es si los cambios obtenidos ocurrieron al azar (hipótesis nula). Si encontramos que la probabilidad de que el cambio sea al azar es muy bajo, rechazaremos la hipótesis nula y diremos que el cambio es estadísticamente significativo

#### Usaremos como ejemplo T-test o T-Student. En este caso asumimos una distribución normal, T-test se ha usado para el análisis de microarreglos. Trabajaremos primero con la tabla mini, sobre el primer gen

mini[1,1:3]

mini[1,4:6]

t.test(mini[1,1:3], mini[1,4:6])

#### podemos ver que nos da un p-value=0.06. ¿Esto es realmente significativo? Analicemos más genes con la herramienta for() y sólo imprimimos pvalue

for (i in rownames(mini)) {

t = t.test(mini[i,1:3], mini[i,4:6])

print(i)

print(t$p.value) #Imprimir sólo el valor de 't'

}

#### ahora analicemos todos los genes, filtrando aquellos que no superen el umbral de p-value de 1

exprNorm$p.value = 1

#### para crear una nueva columna en exprNorm (se llenará de 1)

for (i in rownames(exprNorm)) {

exprNorm[i,"p.value"] = t.test(exprNorm[i,1:3], exprNorm[i,4:6])$p.value

}

head(exprNorm)

pvals = exprNorm$p.value

names(pvals) = rownames(exprNorm)

head(pvals)

#### EJERCICIO:¿Cómo obtienen los pvals más chicos? ¿y los más grandes?¿Cuántos pvals son menores a un umbral, por ejemplo 0.05 o 0.001?

pvals\_0.05=pvals[pvals<0.05]

length(pvals)

length(pvals\_0.05)

pvals\_0.001=pvals[pvals<0.001]

length(pvals\_0.001)

head(pvals\_0.001)

#### ¿Cómo interpretan este resultado?

#### ¿Creen que hay algún problema con el análisis hasta ahora?

# INSTALAR PAQUETES REQUERIDOS para la práctica de mañana

# Guía de ejercicios básicos de R

#### Marcos Paolinelli

#### 29 de Noviembre de 2018

#### HOLA MUNDO

print("Hola MUNDO")

#### para obtener ayuda de cualquier función usamos

?print

#### lo podemos usar como calculadora

1+1+2+3+5+8

8-13

13\*21

34\*34

55/34 \* sqrt(89)

#### Para guardar alguno de estos resultados necesitamos asignarlos a un “objeto”. Para esto podemos usar el símbolo “=” o “<-”

x = 2+1 #x <- 2+1

x

#### Se pueden hacer operaciones directamente con los objetos

otro\_numero = 55/34 \* sqrt(89)

otro\_numero

otro\_numero\*x

#### para eliminar un objeto usamos rm

rm(otro\_numero)

otro\_numero

#### Para crear un objeto con más de un elemento usamos c()

Ventas = c(1,1,2,3,5,8,13)

Ventas

#### R posee un método recursivo que nos permite realizar una operación sobre el objeto múltiples veces

Ventas + 5

#### También podemos usar R con funciones. Una función es una palabra seguida por paréntesis word()

sum(Ventas)

mean(Ventas)

max(Ventas)

min(Ventas)

range(Ventas)

unique(Ventas)

#### Para ordenar usamos sort()

sort\_increase=sort(Ventas)

## **OBJETOS Y VECTORES**

frutas = c("manzana" , "banana" , "sandía" , "ananá" , "limón" , "frutilla" , "naranja")

#### para asignar nombres a elementos en un objeto existe la función names()

names(Ventas) = frutas

Ventas

#### ¿Qué pasó?

#### El objeto Ventas es un vector. El vector posee múltiples elementos en una determinada posición que se indica mediante el empleo de los corchetes: Ventas[posición]

Ventas[3]

#### Múltiples series de elementos pueden ser obtenidos al mismo tiempo. En este caso necesitamos usar c()

Ventas[c(1,3:7)]

#### Y si por el contrario, queremos eliminar esos elementos:

Ventas[-c(1,5)]

#### EJERCICIO: ¿CUál es la media de frutas vendidas sin contar “frutilla”? Utilizar lo aprendido. El resultado debe ser 4.166667

## **OPERADORES RELACIONALES**

#### A veces queremos saber características de los números. Por ejemplo: El número de tiburones con longitud mayor a 2 metros. Esto se puede responder con operadores relacionales (más grande que, más largo que, igual a, etc.)

#### Con el objeto ventas. ¿Qué frutas se vendieron más de 5 veces?

Ventas>5

Ventas == 5

#### En el caso anterior usamos “==” para evitar asignar 5 al objeto Ventas. Como estas comparaciones nos dan vectores lógicos TRUE/FALSE o 1/0. Esto puede resultar muy útil para hacer algunos cálculos:

sum(Ventas > 5)

Ventas[Ventas > 5]

#### EJERCICIO: ¿Cómo sabemos que fruta se vendió más caro que la media?

## **AHORA VAMOS A HACER GRÁFICOS!!!**

#### R es muy popular por la calidad y variedad de gráficas que puede generar. Una vez que los datos están en un objeto, graficarlo es muy simple. Por ejemplo: grafiquemos las frutas vendidas

plot(Ventas)

#### Si queremos otro tipo de gráficas

barplot(Ventas)

pie(Ventas)

hist(Ventas)

#### Podemos ver todas las opciones para cada tipo de gráfico usando “?”, pero sigamos con ejemplos

hist(Ventas, col="skyblue", main="Histograma en azúl", ylab="Frutas vendidas")

#### Existen varias maneras de guardar gráficas. Por ejemplo: como pdf, pero primero definimos la carpeta de trabajo

getwd()

setwd("./curso/")

pdf("histogram.pdf")

hist(Ventas, col="skyblue", main="Histograma en azúl", ylab="Frutas vendidas")

dev.off()

## **LECTURA DE ARCHIVOS Y MANIPULACIÓN DE TABLAS**

#### Cuando ya tenemos valores en un archivo de texto, podemos importarlo a R, para editarlo. Buscar archivo “file1.txt”" en el directorio de trabajo. Deben primero descargar desde el [drive](https://drive.google.com/drive/folders/1NcxBVwjK3KZLsw4svn_cOa4dN_-0KiOw?usp=sharing):

setwd("./<Especificar\_directorio/")

getwd()

list.files()

#### sin preocuparnos por el formato del archivo podemos revisarlo con scan. PUEDEN USAR TAB para AUTOCOMPLETAR

num = scan("../file1.txt")

num

#### usemos un ejemplo más realista con el archivo “tab.txt”. Descargar del [drive](https://drive.google.com/drive/folders/1NcxBVwjK3KZLsw4svn_cOa4dN_-0KiOw?usp=sharing)

tab = read.table("tab.txt")

head(tab)

#### existe una forma rápida de visualizar este tipo de datos

?boxplot

boxplot(x = tab, col =c("red","green","blue"))

#### hemos visto previamente que usando “[ ]” podemos restringir los elementos del vector. Podemos hacer lo mismo con tablas pero en este caso necesitamos agregar una coma para referirnos a [lineas,columnas]

tab[c(1,6,10), c(1,3)]

## **EJEMPLOS DE TRABAJO EN DATOS DE EXPRESIÓN**

#### La tabla del archivo “expr.txt” del [drive](https://drive.google.com/drive/folders/1NcxBVwjK3KZLsw4svn_cOa4dN_-0KiOw?usp=sharing) contiene el valor de expresión génica de 10000 genes entre dos condiciones, con tres réplicas cada uno

expr = read.table("expr.txt")

head(expr)

dim(expr)

boxplot(expr, col=rainbow(6), main="Boxplot, datos crudos")

#### ¿Cómo se ve esa gráfica? ¿Qué pasa si graficamos la distribución mediante un histograma?

hist(expr[,1], breaks=50, main=1, col="skyblue", xlab="gene expression")

#### Pero acá sólo obtenemos el histograma para los datos de la primera columna, ¿Cómo hacemos para graficar todas las columnas?. Necesitamos usar un control de estructura, al emplear “n1:n2”, crea una secuencia de n1 a n5 aumentando de a 1

for (i in 1:5) {

print(i)

}

#### Volvamos a los datos, ¿cuantos histogramas necesitamos? Si no nos acordamos las dimensiones podemos usar dim(), nrow() o ncol()

ncol(expr)

#### Ahora si, para producir 6 histogramas usamos:

for (i in 1:ncol(expr)) {

hist(expr[,i], breaks=50, main=i, col="skyblue", xlab="gene expression")

}

#### Esto nos grafica los 6 pero cada uno por separado, para poder graficarlos todos juntos, necesitamos establecer cómo queremos la estructura de los gráficos. La estructura por defecto es mfrow=c(1,1), pero nosotros queremos 2 líneas y 3 columnas. [ mfrow: number of Multiple Figures (use ROW-wise)]

pdf("multiplot.pdf")

par(mfrow = c(3,2))

for (i in 1:ncol(expr)) {

hist(expr[,i], breaks=50, main=i, col="skyblue", xlab="gene expression")

}

dev.off()

#### ¿Cómo se ven esos gráficos? ¿Es similar la distribución de los datos? Si asumimos una distribución idéntica, con algunos valores que se desvían, podemos normalizar usando el valor de la mediana

exprNorm = expr

for (i in 1:ncol(exprNorm)) {

exprNorm[,i] = exprNorm[,i] - median(exprNorm[,i])

}

#### ¿Funcionó? Hagamos una imagen para responder

par(mfrow=c(1,2))

boxplot(expr, col=rainbow(6), main="Raw data")

boxplot(exprNorm, col=rainbow(6), main="Normalized data")

## **COMPARACIÓN DE VALORES ENTRE CONDICIONES**

#### Este experimento es muy simple porque sólo tenemos dos condiciones, por lo que queremos determinar cuales son los genes expresados diferencialente al comparar la condicion “ko”y “wt”.¿Cómo lo podemos hacer? Primero, observamos los datos

head(exprNorm)

#### 1)Necesitamos determinar el valor promedio en ko y wt

#### 2)Comparar el valor promedio de cada gen en ko y wt

#### Para lo primero hacemos una evaluación con sólo algunos datos usando head() y cambiamos de la estructura de data.frame a matrix

mini=head(exprNorm,3)

mini = as.matrix(head(exprNorm, 3))

mini

rownames(mini)

#### rownames nos regresa los nombres de los renglones

for (i in rownames(mini)) {

koMean = mean(mini[i,1:3])

wtMean = mean(mini[i,4:6])

print("#############")

print(i)

print(koMean)

print(wtMean)

}

#### for() es fácil de usar pero, el resultado puede ser difícil de asignar a valores,por lo que no es muy eficiente. Una alternativa eficiente es apply(). Esto nos permite aplicar una función sobre muchos elementos en un sólo paso

apply(mini[,1:3], 1, mean)

apply(mini[,4:6], 1, mean)

#### apply() necesita de 3 elementos

#### 1. Una tabla (como matriz)

#### 2. Especificar las dimensiones (1=rows, 2=columns)

#### 3. La función a aplicar sobre los datos

#### EJERCICIO: Usar apply() para obtener la expresión promedio por condición (6 condiciones), no por gen con todos los datos en exprNorm para todos los genes, obtener un vector que contenga la diferencia entre el valor promedio de ko y wt.

#### Crear el histograma correspondiente

#### ¿Es lo que esperábamos?

exprNorm=as.matrix(exprNorm)

meanko=apply(exprNorm[,1:3],1,mean)

meanwt=apply(exprNorm[,4:6],1,mean)

diff=meanwt-meanko

head(diff)

hist(diff)

#### ahora obtuvimos las diferencias, pero ¿son significativas?

#### El objetivo principal cuando analizamos expresión es detectar los cambios asociados a la expresión génica. Sin embargo grandes cambios no es indicativo de “más confiable”. Necesitamos considerar la variación de los valores dentro de las muestras de la misma condición. Para esto se han desarrollado distintos tests y métodos estadísticos. Lo que queremos verificar mediante el test estadístico es si los cambios obtenidos ocurrieron al azar (hipótesis nula). Si encontramos que la probabilidad de que el cambio sea al azar es muy bajo, rechazaremos la hipótesis nula y diremos que el cambio es estadísticamente significativo

#### Usaremos como ejemplo T-test o T-Student. En este caso asumimos una distribución normal, T-test se ha usado para el análisis de microarreglos. Trabajaremos primero con la tabla mini, sobre el primer gen

mini[1,1:3]

mini[1,4:6]

t.test(mini[1,1:3], mini[1,4:6])

#### podemos ver que nos da un p-value=0.06. ¿Esto es realmente significativo? Analicemos más genes con la herramienta for() y sólo imprimimos pvalue

for (i in rownames(mini)) {

t = t.test(mini[i,1:3], mini[i,4:6])

print(i)

print(t$p.value) #Imprimir sólo el valor de 't'

}

#### ahora analicemos todos los genes, filtrando aquellos que no superen el umbral de p-value de 1

exprNorm$p.value = 1

#### para crear una nueva columna en exprNorm (se llenará de 1)

for (i in rownames(exprNorm)) {

exprNorm[i,"p.value"] = t.test(exprNorm[i,1:3], exprNorm[i,4:6])$p.value

}

head(exprNorm)

pvals = exprNorm$p.value

names(pvals) = rownames(exprNorm)

head(pvals)

#### EJERCICIO:¿Cómo obtienen los pvals más chicos? ¿y los más grandes?¿Cuántos pvals son menores a un umbral, por ejemplo 0.05 o 0.001?

pvals\_0.05=pvals[pvals<0.05]

length(pvals)

length(pvals\_0.05)

pvals\_0.001=pvals[pvals<0.001]

length(pvals\_0.001)

head(pvals\_0.001)

#### ¿Cómo interpretan este resultado?

#### ¿Creen que hay algún problema con el análisis hasta ahora?

# INSTALAR PAQUETES REQUERIDOS para la práctica de mañana

# Práctica 1: Introducción a R, vectores y estadística descriptiva de 1 variable

#### EI1012-MT1012 Estadística y Optimización (2015/2016)

#### Pablo Gregori, Universitat Jaume I de Castellón

# 1. Introducción a R

## **1.1. El proyecto R y el programa**

* En el [proyecto R](http://www.r-project.org/) se puede conseguir el programa, y conocer todo lo que le rodea.
* Instala y ejecuta “R” en el equipo.
* R Console: terminal donde programar y obtener resultados.
* Recomendaciones:
  + Comparte el escritorio entre la consola de R y el navegador de internet con este documento.
  + La interfaz [RStudio](http://www.rstudio.com/) es muy conveniente, aunque aquí no la vamos a explicar.
* Menú Archivo de la consola de R:
  + Cambiar dir...: establece la ruta para interactuar con ficheros.
  + Guardar área de trabajo: crea fichero con sesión de R (para continuación).
  + Cargar área de trabajo: carga fichero con sesión de R guardada.
* En RStudio está dentro del menú Session.

## **1.2. El lenguaje de programación R**

* Como lenguaje, tiene constantes, variables, funciones, etc. y es de tipo “orientado a objetos”.
* Una variable se define con un nombre, y se le asigna un valor igualándola a una constante (o a otra variable ya definida).
* Las funciones que devuelven texto, lo hacen por la ventana “R Console”.
* Las funciones que devuelven figuras, lo hacen por la ventana “R Graphics”.
* Las funciones que piden ayuda abren el navegador de internet con la ayuda solicitada.
* Los comentarios a partir de la almohadilla (#).
* La función q() cierra la consola con la opción de guardar la sesión de R en un archivo, que se puede abrir posteriormente para continuar.
* Ejemplo: copia y pega el siguente bloque en la consola de R:

*# línea de comentario empieza por almohadilla*

5 *# la constante numérica 5*

demo(graphics) *# función 'demo()' ejecuta demos de librerías*

help(demo) *# función 'help()' devuelve ayuda*

load(url('https://goo.gl/uDzU8v')) *# función 'load()' carga sesión de R*

ls() *# función 'ls()' lista las variables definidas en la sesión*

q() *# cierra R, pero te permite guardar sesión para reanudar. ¡Cancela!*

## **1.3. R como calculadora de constantes numéricas**

* +, -, \*, /, ^, … operaciones matemáticas
* sqrt(), log(), exp(), sin(), abs(), … otras funciones matemáticas
* Ejemplo: copia y pega el siguente bloque en la consola de R

5+3 *# suma*

5-3 *# resta*

5\*3 *# producto*

5/3 *# cociente*

5/0 *# sorpresa: R maneja el infinito*

0/0 *# sorpresa: R maneja las indeterminaciones (NaN, Not a Number)*

5^3 *# potencia*

sqrt(5) *# raíz cuadrada*

exp(5) *# el número "e" elevado a (función exponencial)*

log(5) *# logaritmo neperiano*

log10(5) *# logartimo en base 10*

pi *# el número "pi" está definido*

sin(pi/2) *# la función seno (ángulo en radianes)*

factorial(5) *# el factorial, 5!*

choose(5,3) *# número combinatorio (5 sobre 3)*

**EJERCICIO 1:** Calcula los valores numéricos aproximados de 0.3⋅0.150.3⋅0.15+0.2⋅0.8+0.5⋅0.120.3⋅0.150.3⋅0.15+0.2⋅0.8+0.5⋅0.12, 566!e−5566!e−5 y (207)0.470.613(207)0.470.613.

* Sol.: 0.16981130.1698113, 0.14622280.1462228 y 0.16588230.1658823.

**FIN EJERCICIO 1**

# 2. Variable tipo vector

* Es el tipo de variable más básico del lenguaje R.
* Es una lista ordenada de constantes (números, cadenas de texto entrecomilladas, o lógicos TRUE/FALSE).
* Ideal para albergar muestras univariantes (cuantitativas o cualitativas).
* Se pueden leer desde ficheros con la función scan() (no la vamos a trabajar, pide ayuda help(scan) si necesitas saber más).
* Si se invoca en la línea de comandos, se imprime en pantalla, por filas, con un indicador de la posición de la componente más a la izquierda de cada fila.

## **2.1 Creación de vectores**

* La función c() concatena constantes (y vectores) en un vector.
* La función : crea un vector de números consecutivos.
* Ejemplo: copia y pega el siguente bloque en la consola de R:

*# Vectores de constantes numéricas*

v1 = 5

v2 = 1:10

v3 = c(v1, v2, pi, -5, 0, 1/3)

v1

v2

v3

*# Vectores de constantes tipo texto*

*# Ojo: siempre entrecomilladas, o parecerán nombres de variables*

v4 = 'a'

v5 = c('b','c','d')

v6 = c(v4, v5, v5, v4, 'hola')

v4

v5

v6

*# Vectores de constantes lógicas*

*# Ojo: sólo TRUE y FALSE*

v7 = TRUE

v8 = c(TRUE,FALSE,FALSE)

v9 = c(v7, v8)

v7

v8

v9

## **2.2 Llamadas a funciones en R**

* Las funciones constan de un nombre seguido de paréntesis que encierran a ninguno, o uno o más argumentos separados por comas. A veces aparece el argumento ... (puntos suspensivos), que indica una cantidad indeterminada de argumentos.
* Ejemplo:

*# Función 'sample()' simula lanzamientos aleatorios de un dado abstracto*

*# Argumentos:*

*# x: número de caras del dado*

*# size: número de lanzamientos*

*# replace: ¿se puede repetir resultado?*

*# prob: probabilidad de cada cara (todas iguales si vale NULL)*

sample(x=6, size=100, replace=TRUE, prob=NULL) *# llamada completa*

sample(6, 100, TRUE, NULL) *# llamada sin nombres de argumento*

sample(TRUE, 6, NULL, 100) *# llamada errando el orden de los argumentos*

sample(100, 6, TRUE, NULL) *# otra llamada errando el orden de los argumentos*

sample(replace=TRUE, x=6, prob=NULL, size=100) *# llamada ok*

## **2.3 Algunas funciones y operaciones importantes con vectores**

Primeras funciones importantes de R:

* ls() sin argumentos, da el vector de nombres de variables definidas.
* a:b crea vector de enteros consecutivos, de a a b.
* c(...) da un vector con los argumentos ... concatenados.
* length(x) da la longitud del vector argumento x.
* sum(...) SUMa las componentes de los argumentos.
* prod(...) PRODucto de las componentes de los argumentos.
* sort(x) da el vector argumento x reordenado de manera creciente.
* sort(x,decreasing=TRUE) lo mismo (pero decreciente).
* min(...) da el valor MÍNimo de los argumentos.
* max(...) da el valor MÁXimo de los argumentos.
* which(x) da el vector de “posiciones” donde x es TRUE. El argumento suele ser una comparación.

Las siguientes funciones/operadores actúan **componente por componente**, devolviendo otro vector de igual tamaño:

* +, -, \*, /, ^, … operaciones matemáticas
* sqrt(), log(), exp(), sin(), abs(), … otras funciones matemáticas
* <, ==, >, <=, >=, != operadores de comparación
* &, |, xor(), ! operadores lógicos (and, or, or exclusivo, not)
* Ejemplo: copia y pega el siguente bloque en la consola de R

ls()

1:10

10:1

v3

length(v3)

sum(v3)

prod(v3)

sort(v3)

sort(v3, decreasing=TRUE)

sqrt(v3)

v3 < 3

which(v3<3)

v3 == 5

which(v3==5)

**EJERCICIO 2:** Realiza las siguientes sumas:

1. 1+2+3+⋯+10001+2+3+⋯+1000
   * Sol.: 500500500500
2. 1+2+4+8+16+⋯+10241+2+4+8+16+⋯+1024
   * Sol.: 20472047

**FIN EJERCICIO 2**

**EJERCICIO 3:** El vector grupo representa el grupo al que pertenece una serie de alumnos.

1. Visualízalo en pantalla
   * Sol.: -
2. ¿Cuántas componentes tiene?
   * Sol.: 192192
3. ¿En qué posiciones del vector está la letra ‘A’?
   * Sol.: 2,8,17,21,28,84,101,108,111,115,123,136,190,1922,8,17,21,28,84,101,108,111,115,123,136,190,192

**FIN EJERCICIO 3**

**EJERCICIO 4:** El vector nota representa la nota de un examen, de los mismos alumnos cuyo grupo se ha guardado en el vector grupo, y en el mismo orden.

1. Visualízalo en pantalla
   * Sol.: -
2. ¿Cuántas componentes tiene?
   * Sol.: 192192
3. ¿Cuánto suman todas las notas?
   * Sol.: 962962
4. ¿Cuál es la media aritmética de todas las notas?
   * Sol.: 5.01041675.0104167
5. ¿En qué posiciones están las notas mayores de 7.07.0?
   * Sol.: 81,103,120,15181,103,120,151
6. Visualiza las notas ordenadas de menor a mayor
   * Sol.: -
7. Visualiza las notas ordenadas de mayor a menor
   * Sol.: -
8. ¿Cuál ha sido la nota máxima?
   * Sol.: 7.77.7
9. ¿En qué posición del vector está esa nota máxima?
   * Sol.: 120120

**FIN EJERCICIO 4**

# 3. Subvectores

* A veces interesa una parte de la muestra, por lo que es muy habitual usar este operador.
* El operador corchete [] selecciona componentes de un vector.
* Ejemplo: copia y pega el siguente bloque en la consola de R

v3

v3[2] *# la segunda componente*

v3[-2] *# todo menos la segunda componente*

v3[1:3] *# la 1a, 2a y 3a comp.*

v3[ c(2,5) ] *# la 2a y la 5a componente ( ojo a la c() necesaria )*

v3[ c(TRUE,TRUE,FALSE)] *# 1a y 2a sí, 3a no, etc.*

v3[ v3<4 ] *# las comp. que cumplen la condición v3<4 (los TRUEs)*

v3[ v3>1 & v3<4 ] *# las comp. que cumplen v3>1 y v3<4*

*# Utilidad: elegir parte de una muestra*

*# Suma los 3 primeros datos de v3*

v3[1:3] *# da los 3 primeros datos de v3*

sum(v3[1:3]) *# ahora los suma*

*# ¿Cuántos datos de v3 son menores que 4?*

v3[ v3<4 ] *# da los datos de v3 que cumplen v3<4*

length(v3[ v3<4 ]) *# ahora los cuenta*

*# ¿Cuántos alumnos son del grupo A?*

grupo[ grupo=='A' ] *# da los datos del grupo A*

length(grupo[ grupo=='A' ]) *# ahora los cuenta*

*# ¿Cuál ha sido la máxima nota del grupo A?*

nota[ grupo=='A' ] *# da las notas del grupo A*

max(nota[ grupo=='A' ]) *# ahora da la máxima*

*# ¿Cuántos alumnos del A tienen menos de 3 en la nota?*

nota[ grupo=='A' & nota<3 ] *# da las notas de los del grupo A y con menos de un 3*

length(nota[ grupo=='A' & nota<3 ]) *# ahora los cuenta*

**EJERCICIO 5:** A partir de los vectores grupo y nota definidos:

1. Visualiza las notas de los 10 primeros alumnos
   * Sol.: -
2. Suma las notas de los 10 primeros alumnos del vector
   * Sol.: 51.851.8
3. ¿Cuántos alumnos hay del grupo ‘C’?
   * Sol.: 3939.
4. Suma las notas de los alumnos del grupo ‘A’
   * Sol.: 63.663.6.
5. ¿Cuántos alumnos han aprobado?
   * Sol.: 102102.
6. ¿Cuántos alumnos del grupo ‘B’ han aprobado?
   * Sol.: 1212
7. ¿Qué porcentaje de alumnos del grupo ‘C’ han aprobado?
   * Sol.: 58.97435958.974359
8. ¿Cuáles han sido las notas máxima y mínima del grupo ‘D’?
   * Sol.: 6.8 y 2.5. respectivamente.
9. ¿De qué grupos son la máxima y mínima notas de toda la muestra?
   * Sol.: E� y B� respectivamente.
10. Nota media de alumnos de los grupos ‘A’ y ‘B’, reunidos, teniendo en cuenta sólo a los que han aprobado.
    * Sol.: 5.8255.825.

**FIN EJERCICIO 5**

* También se pueden modificar ciertas componentes de un vector o definir vectores a partir de otros por condiciones
* Ejemplo: copia y pega el siguente bloque en la consola de R

nota2 = nota

nota

nota2

nota2[ nota < 5 ] = 'Suspenso'

nota2

nota2[ nota >= 5 ] = 'Aprobado'

nota2

# 4. Estadística descriptiva de una variable con R

## **4.1 La muestra**

* Los datos a analizar deben estar almacenados en una variable de tipo vector.
* Llamaremos x al vector con los datos de la muestra.
* **Pero puede tener cualquier nombre, o estar indexado con el operador corchete para usar parte de un vector**.

## **4.2 Argumentos especiales de las funciones gráficas**

* Las funciones gráficas de R tienen argumentos comunes para especificar:
  + Título superior: main. Igualar a una cadena de texto.
  + Título inferior: sub. Igualar a una cadena de texto.
  + Etiqueta eje X: xlab. Igualar a una cadena de texto.
  + Etiqueta eje Y: ylab. Igualar a una cadena de texto.
  + Color o colores: col. Igualar a un número o nombre de colores (o un vector de números o nombres de colores). Para conocer los nombres de colores escribe colors() en la consola.
  + Forma del punto: pch. Igualar a un número entre 1 y 25. Sólo cuando la figura dibuje puntos.
* Ejemplo: copia y pega el siguente bloque en la consola de R

plot(x=1:25, y=1:25, main='Puntos: formas y colores',

sub='Cada punto una forma y un color',

xlab='Número de punto y color',

ylab='Etiqueta eje Y', col=1:25, pch=1:25)

## **4.3 Estadísticas de datos cualitativos**

Suponemos que x es la variable que contiene los datos.

### 4.3.1 Tabla de frecuencias:

* **Absolutas:** table(x)
* **Relativas:** table(x)/length(x)
* **Porcentajes:** table(x)/length(x) \* 100

### 4.3.2 Gráficas:

* **Diagrama de barras:** barplot(height=table(x))
* **Diagrama de sectores (tarta):** pie(x=table(x))

**EJERCICIO 6:** Escribe la tabla de frecuencias absolutas y relativas de la variable grupo (para saber cuántos datos hay de cada grupo), y haz un diagrama de barras y otro de sectores, cambiando los colores y añadiendo títulos para practicar.

**FIN EJERCICIO 6**

## **4.4 Estadísticas de datos cuantitativos**

Suponemos que x es la variable que contiene los datos:

### 4.4.1 Tabla de frecuencias:

* Primero hay que “cortar” la recta en intervalos, y luego se pide la tabla de frecuencias de los intervalos.
* table(cut(x, breaks))
  + x los datos
  + breaks el número de intervalos, o el vector con los extremos sucesivos
* Ejemplo: copia y pega el siguente bloque en la consola de R

nota *# los datos*

table(nota) *# ¡no lo hace bien!*

table( cut(nota, breaks=0:10) ) *# la tabla de frecuencias con intervalos*

### 4.4.2 Estadísticos:

* **Media:** mean(x)
* **Mediana:** median(x)
* **Mínimo:** min(x)
* **Máximo:** max(x)
* **Cuantil de orden**p�**:** quantile(x, prob=p�)
* **Recorrido:** hay que calcularlo con su definición (max(x)-min(x)).
* **Recorrido intercuartílico:** IQR(x)
* **Cuasivarianza:** var(x)
* **Cuasidesviación típica:** sd(x)
* **Varianza:** ¡No está programada! Se puede conseguir con la operación var(x)\*(length(x)-1)/length(x)
* **Desviación típica:** ¡No está programada! Se puede conseguir con la operación sd(x)\*sqrt((length(x)-1)/length(x))
* **Coeficiente de variación**: ¡No está programado! Se puede conseguir una versión “cuasi” con la operación sd(x)/abs(mean(x)), y la versión “sin cuasi” con la operación (sd(x)\*sqrt((length(x)-1)/length(x)))/abs(mean(x)).

### 4.4.3 Gráficos

* **Diagrama de puntos:** stripchart(x) (para pocos datos)
* **Histograma:** hist(x, breaks, ...)
  + x los datos
  + breaks el número de intervalos, o el vector con los extremos sucesivos
* **Diagrama de caja:** boxplot(...) (se pueden poner varias muestras para comparar)

**EJERCICIO 7:** Escribe la tabla de frecuencias absolutas y relativas de la variable nota donde los intervalos correspondan a números enteros. Dibuja el histograma con esa misma partición en intervalos. Calcula la nota media y la cuasidesviación típica, así como la mínima y máxima.

* Sol.: Media = 5.01041675.0104167, cuasidesviación típica = 1.07020371.0702037, mínima = 1.71.7 y máxima = 7.77.7.

**FIN EJERCICIO 7**

**EJERCICIO 8:** Calcula la nota media y la cuasidesviación típica de cada grupo.

* Sol.: A = 4.54285714.5428571 y 0.73139940.7313994; B = 5.12173915.1217391 y 1.1855021.185502; C = 5.22820515.2282051 y 1.01746561.0174656; D = 4.77962964.7796296 y 1.03475081.0347508; E = 5.13870975.1387097 y 1.11404671.1140467.

**FIN EJERCICIO 8**

**EJERCICIO 9:** Calcula el percentil 66 de las notas de todos los alumnos, y también de los alumnos del grupo C.

* Sol.: 5.55.5 y 5.8085.808.

**FIN EJERCICIO 9**

**EJERCICIO 10:** Un alumno tiene una nota de 4.9. ¿Qué porcentaje, del total de alumnos, tiene una nota menor o igual que la suya? ¿Y qué porcentaje tiene una nota mayor o igual que la suya?

* Sol.: 46.87546.875% y 56.2556.25%

**FIN EJERCICIO 10**

**EJERCICIO 11:** Dibuja los diagramas de caja de las notas de cada grupo, para comparar el nivel de los grupos.

**FIN EJERCICIO 11**

# 5. Ejercicios de entrenamiento

**EJERCICIO 12:** Los siguientes vectores contienen datos sobre una muestra de coches americanos. Cada vector representa:

* coche: modelo de coche
* efi: eficiencia, en millas recorridas por galón de gasolina.
* ncil: número de cilindros del motor.
* cil: cilindrada total, en pulgadas cúbicas.
* pot: potencia, en CV (caballos de vapor).
* peso: peso, en miles de libras.
* tiempo: tiempo, en segundos, en recorrer un cuarto de milla, desde parado.
* tipo: tipo, automático o manual.
* marchas: número de marchas (hacia adelante).
* carbur: número de carburadores.

de modo que el primer datos de cada vector es del mismo coche, y así sucesivamente. Realiza las siguientes tareas de análisis de datos:

1. ¿Cuántos coches hay en la muestra?
   * Sol.: 3232.
2. ¿Cuántos de los coches son automáticos?
   * Sol.: 1313.
3. Escribe la tabla de frecuencias del tipo de coche. ¿Qué tipo es más abundante en la muestra?
   * Sol.: Autom=13;Manual=19�����=13;������=19. Es más abundante el Manual.
4. Calcula la media aritmética y la cuasidesviación típica de la eficiencia de los coches automáticos, y de los manuales, por separado. ¿Qué tipo es más eficiente?
   * Sol.: 24.392307724.3923077 y 6.16650386.1665038, para los automáticos, y 17.147368417.1473684 y 3.83396643.8339664, para los manuales. El más eficiente es el Automático, por tener la media más alta.
5. Dibuja un diagrama de sectores que informa sobre el número de cilindros de los coches de la muestra.
   * Sol.: -
6. Calcula la media de la potencia de los coches automáticos de 8 cilindros.
   * Sol.: 299.5299.5 CV.
7. Calcula la media de tiempos, en recorrer el cuarto de milla, para los coches manuales de 44 marchas delanteras.
   * Sol.: 20.02520.025 seg.
8. ¿Qué modelo de coche tiene la mayor cilindrada de la muestra? Escribe los cinco modelos de mayor cilindrada.
   * Sol.: Cadillac Fleetwood es el de mayor cilindrada, y los otros son, Hornet Sportabout, Duster 360, Lincoln Continental, Chrysler Imperial, Pontiac Firebird.
9. Calcula la media de cilindrada **por cilindro (no por motor)** de toda la muestra de cilindros de todos los coches.
   * Sol.: 35.028645835.0286458 pulgadas cúbicas.

**FIN EJERCICIO 12**

**EJERCICIO 13:** Si la variable conc recoge la concentración de plomo (en ppm) en el aire de cierta zona durante un día completo (1 muestreo cada 5 minutos):

1. ¿Cuántas observaciones se han recogido?
   * Sol.: 288288.
2. ¿Cuál ha sido la concentración máxima?
   * Sol.: 47.3447.34 ppm.
3. ¿En cuántos de los muestreos se ha superado la concentración de 40.040.0 ppm?
   * Sol.: 6161.
4. ¿Cuál ha sido la concentración media del día?
   * Sol.: 24.072291724.0722917 ppm.
5. ¿Cuáles fueron las 10 mediciones más bajas del día?
   * Sol.: 0.93,1.07,1.77,2.03,2.58,2.73,2.75,2.88,2.88,2.910.93,1.07,1.77,2.03,2.58,2.73,2.75,2.88,2.88,2.91.
6. Si la primera medida fue a las 00:00, ¿a qué hora del día se alcanzó la concentración máxima?
   * Sol.: A las 11:45.

**FIN EJERCICIO 13**

# Práctica 1: Introducción a R, vectores y estadística descriptiva de 1 variable

#### EI1012-MT1012 Estadística y Optimización (2015/2016)

#### Pablo Gregori, Universitat Jaume I de Castellón

# 1. Introducción a R

## **1.1. El proyecto R y el programa**

* En el [proyecto R](http://www.r-project.org/) se puede conseguir el programa, y conocer todo lo que le rodea.
* Instala y ejecuta “R” en el equipo.
* R Console: terminal donde programar y obtener resultados.
* Recomendaciones:
  + Comparte el escritorio entre la consola de R y el navegador de internet con este documento.
  + La interfaz [RStudio](http://www.rstudio.com/) es muy conveniente, aunque aquí no la vamos a explicar.
* Menú Archivo de la consola de R:
  + Cambiar dir...: establece la ruta para interactuar con ficheros.
  + Guardar área de trabajo: crea fichero con sesión de R (para continuación).
  + Cargar área de trabajo: carga fichero con sesión de R guardada.
* En RStudio está dentro del menú Session.

## **1.2. El lenguaje de programación R**

* Como lenguaje, tiene constantes, variables, funciones, etc. y es de tipo “orientado a objetos”.
* Una variable se define con un nombre, y se le asigna un valor igualándola a una constante (o a otra variable ya definida).
* Las funciones que devuelven texto, lo hacen por la ventana “R Console”.
* Las funciones que devuelven figuras, lo hacen por la ventana “R Graphics”.
* Las funciones que piden ayuda abren el navegador de internet con la ayuda solicitada.
* Los comentarios a partir de la almohadilla (#).
* La función q() cierra la consola con la opción de guardar la sesión de R en un archivo, que se puede abrir posteriormente para continuar.
* Ejemplo: copia y pega el siguente bloque en la consola de R:

*# línea de comentario empieza por almohadilla*

5 *# la constante numérica 5*

demo(graphics) *# función 'demo()' ejecuta demos de librerías*

help(demo) *# función 'help()' devuelve ayuda*

load(url('https://goo.gl/uDzU8v')) *# función 'load()' carga sesión de R*

ls() *# función 'ls()' lista las variables definidas en la sesión*

q() *# cierra R, pero te permite guardar sesión para reanudar. ¡Cancela!*

## **1.3. R como calculadora de constantes numéricas**

* +, -, \*, /, ^, … operaciones matemáticas
* sqrt(), log(), exp(), sin(), abs(), … otras funciones matemáticas
* Ejemplo: copia y pega el siguente bloque en la consola de R

5+3 *# suma*

5-3 *# resta*

5\*3 *# producto*

5/3 *# cociente*

5/0 *# sorpresa: R maneja el infinito*

0/0 *# sorpresa: R maneja las indeterminaciones (NaN, Not a Number)*

5^3 *# potencia*

sqrt(5) *# raíz cuadrada*

exp(5) *# el número "e" elevado a (función exponencial)*

log(5) *# logaritmo neperiano*

log10(5) *# logartimo en base 10*

pi *# el número "pi" está definido*

sin(pi/2) *# la función seno (ángulo en radianes)*

factorial(5) *# el factorial, 5!*

choose(5,3) *# número combinatorio (5 sobre 3)*

**EJERCICIO 1:** Calcula los valores numéricos aproximados de 0.3⋅0.150.3⋅0.15+0.2⋅0.8+0.5⋅0.120.3⋅0.150.3⋅0.15+0.2⋅0.8+0.5⋅0.12, 566!e−5566!e−5 y (207)0.470.613(207)0.470.613.

* Sol.: 0.16981130.1698113, 0.14622280.1462228 y 0.16588230.1658823.

**FIN EJERCICIO 1**

# 2. Variable tipo vector

* Es el tipo de variable más básico del lenguaje R.
* Es una lista ordenada de constantes (números, cadenas de texto entrecomilladas, o lógicos TRUE/FALSE).
* Ideal para albergar muestras univariantes (cuantitativas o cualitativas).
* Se pueden leer desde ficheros con la función scan() (no la vamos a trabajar, pide ayuda help(scan) si necesitas saber más).
* Si se invoca en la línea de comandos, se imprime en pantalla, por filas, con un indicador de la posición de la componente más a la izquierda de cada fila.

## **2.1 Creación de vectores**

* La función c() concatena constantes (y vectores) en un vector.
* La función : crea un vector de números consecutivos.
* Ejemplo: copia y pega el siguente bloque en la consola de R:

*# Vectores de constantes numéricas*

v1 = 5

v2 = 1:10

v3 = c(v1, v2, pi, -5, 0, 1/3)

v1

v2

v3

*# Vectores de constantes tipo texto*

*# Ojo: siempre entrecomilladas, o parecerán nombres de variables*

v4 = 'a'

v5 = c('b','c','d')

v6 = c(v4, v5, v5, v4, 'hola')

v4

v5

v6

*# Vectores de constantes lógicas*

*# Ojo: sólo TRUE y FALSE*

v7 = TRUE

v8 = c(TRUE,FALSE,FALSE)

v9 = c(v7, v8)

v7

v8

v9

## **2.2 Llamadas a funciones en R**

* Las funciones constan de un nombre seguido de paréntesis que encierran a ninguno, o uno o más argumentos separados por comas. A veces aparece el argumento ... (puntos suspensivos), que indica una cantidad indeterminada de argumentos.
* Ejemplo:

*# Función 'sample()' simula lanzamientos aleatorios de un dado abstracto*

*# Argumentos:*

*# x: número de caras del dado*

*# size: número de lanzamientos*

*# replace: ¿se puede repetir resultado?*

*# prob: probabilidad de cada cara (todas iguales si vale NULL)*

sample(x=6, size=100, replace=TRUE, prob=NULL) *# llamada completa*

sample(6, 100, TRUE, NULL) *# llamada sin nombres de argumento*

sample(TRUE, 6, NULL, 100) *# llamada errando el orden de los argumentos*

sample(100, 6, TRUE, NULL) *# otra llamada errando el orden de los argumentos*

sample(replace=TRUE, x=6, prob=NULL, size=100) *# llamada ok*

## **2.3 Algunas funciones y operaciones importantes con vectores**

Primeras funciones importantes de R:

* ls() sin argumentos, da el vector de nombres de variables definidas.
* a:b crea vector de enteros consecutivos, de a a b.
* c(...) da un vector con los argumentos ... concatenados.
* length(x) da la longitud del vector argumento x.
* sum(...) SUMa las componentes de los argumentos.
* prod(...) PRODucto de las componentes de los argumentos.
* sort(x) da el vector argumento x reordenado de manera creciente.
* sort(x,decreasing=TRUE) lo mismo (pero decreciente).
* min(...) da el valor MÍNimo de los argumentos.
* max(...) da el valor MÁXimo de los argumentos.
* which(x) da el vector de “posiciones” donde x es TRUE. El argumento suele ser una comparación.

Las siguientes funciones/operadores actúan **componente por componente**, devolviendo otro vector de igual tamaño:

* +, -, \*, /, ^, … operaciones matemáticas
* sqrt(), log(), exp(), sin(), abs(), … otras funciones matemáticas
* <, ==, >, <=, >=, != operadores de comparación
* &, |, xor(), ! operadores lógicos (and, or, or exclusivo, not)
* Ejemplo: copia y pega el siguente bloque en la consola de R

ls()

1:10

10:1

v3

length(v3)

sum(v3)

prod(v3)

sort(v3)

sort(v3, decreasing=TRUE)

sqrt(v3)

v3 < 3

which(v3<3)

v3 == 5

which(v3==5)

**EJERCICIO 2:** Realiza las siguientes sumas:

1. 1+2+3+⋯+10001+2+3+⋯+1000
   * Sol.: 500500500500
2. 1+2+4+8+16+⋯+10241+2+4+8+16+⋯+1024
   * Sol.: 20472047

**FIN EJERCICIO 2**

**EJERCICIO 3:** El vector grupo representa el grupo al que pertenece una serie de alumnos.

1. Visualízalo en pantalla
   * Sol.: -
2. ¿Cuántas componentes tiene?
   * Sol.: 192192
3. ¿En qué posiciones del vector está la letra ‘A’?
   * Sol.: 2,8,17,21,28,84,101,108,111,115,123,136,190,1922,8,17,21,28,84,101,108,111,115,123,136,190,192

**FIN EJERCICIO 3**

**EJERCICIO 4:** El vector nota representa la nota de un examen, de los mismos alumnos cuyo grupo se ha guardado en el vector grupo, y en el mismo orden.

1. Visualízalo en pantalla
   * Sol.: -
2. ¿Cuántas componentes tiene?
   * Sol.: 192192
3. ¿Cuánto suman todas las notas?
   * Sol.: 962962
4. ¿Cuál es la media aritmética de todas las notas?
   * Sol.: 5.01041675.0104167
5. ¿En qué posiciones están las notas mayores de 7.07.0?
   * Sol.: 81,103,120,15181,103,120,151
6. Visualiza las notas ordenadas de menor a mayor
   * Sol.: -
7. Visualiza las notas ordenadas de mayor a menor
   * Sol.: -
8. ¿Cuál ha sido la nota máxima?
   * Sol.: 7.77.7
9. ¿En qué posición del vector está esa nota máxima?
   * Sol.: 120120

**FIN EJERCICIO 4**

# 3. Subvectores

* A veces interesa una parte de la muestra, por lo que es muy habitual usar este operador.
* El operador corchete [] selecciona componentes de un vector.
* Ejemplo: copia y pega el siguente bloque en la consola de R

v3

v3[2] *# la segunda componente*

v3[-2] *# todo menos la segunda componente*

v3[1:3] *# la 1a, 2a y 3a comp.*

v3[ c(2,5) ] *# la 2a y la 5a componente ( ojo a la c() necesaria )*

v3[ c(TRUE,TRUE,FALSE)] *# 1a y 2a sí, 3a no, etc.*

v3[ v3<4 ] *# las comp. que cumplen la condición v3<4 (los TRUEs)*

v3[ v3>1 & v3<4 ] *# las comp. que cumplen v3>1 y v3<4*

*# Utilidad: elegir parte de una muestra*

*# Suma los 3 primeros datos de v3*

v3[1:3] *# da los 3 primeros datos de v3*

sum(v3[1:3]) *# ahora los suma*

*# ¿Cuántos datos de v3 son menores que 4?*

v3[ v3<4 ] *# da los datos de v3 que cumplen v3<4*

length(v3[ v3<4 ]) *# ahora los cuenta*

*# ¿Cuántos alumnos son del grupo A?*

grupo[ grupo=='A' ] *# da los datos del grupo A*

length(grupo[ grupo=='A' ]) *# ahora los cuenta*

*# ¿Cuál ha sido la máxima nota del grupo A?*

nota[ grupo=='A' ] *# da las notas del grupo A*

max(nota[ grupo=='A' ]) *# ahora da la máxima*

*# ¿Cuántos alumnos del A tienen menos de 3 en la nota?*

nota[ grupo=='A' & nota<3 ] *# da las notas de los del grupo A y con menos de un 3*

length(nota[ grupo=='A' & nota<3 ]) *# ahora los cuenta*

**EJERCICIO 5:** A partir de los vectores grupo y nota definidos:

1. Visualiza las notas de los 10 primeros alumnos
   * Sol.: -
2. Suma las notas de los 10 primeros alumnos del vector
   * Sol.: 51.851.8
3. ¿Cuántos alumnos hay del grupo ‘C’?
   * Sol.: 3939.
4. Suma las notas de los alumnos del grupo ‘A’
   * Sol.: 63.663.6.
5. ¿Cuántos alumnos han aprobado?
   * Sol.: 102102.
6. ¿Cuántos alumnos del grupo ‘B’ han aprobado?
   * Sol.: 1212
7. ¿Qué porcentaje de alumnos del grupo ‘C’ han aprobado?
   * Sol.: 58.97435958.974359
8. ¿Cuáles han sido las notas máxima y mínima del grupo ‘D’?
   * Sol.: 6.8 y 2.5. respectivamente.
9. ¿De qué grupos son la máxima y mínima notas de toda la muestra?
   * Sol.: E� y B� respectivamente.
10. Nota media de alumnos de los grupos ‘A’ y ‘B’, reunidos, teniendo en cuenta sólo a los que han aprobado.
    * Sol.: 5.8255.825.

**FIN EJERCICIO 5**

* También se pueden modificar ciertas componentes de un vector o definir vectores a partir de otros por condiciones
* Ejemplo: copia y pega el siguente bloque en la consola de R

nota2 = nota

nota

nota2

nota2[ nota < 5 ] = 'Suspenso'

nota2

nota2[ nota >= 5 ] = 'Aprobado'

nota2

# 4. Estadística descriptiva de una variable con R

## **4.1 La muestra**

* Los datos a analizar deben estar almacenados en una variable de tipo vector.
* Llamaremos x al vector con los datos de la muestra.
* **Pero puede tener cualquier nombre, o estar indexado con el operador corchete para usar parte de un vector**.

## **4.2 Argumentos especiales de las funciones gráficas**

* Las funciones gráficas de R tienen argumentos comunes para especificar:
  + Título superior: main. Igualar a una cadena de texto.
  + Título inferior: sub. Igualar a una cadena de texto.
  + Etiqueta eje X: xlab. Igualar a una cadena de texto.
  + Etiqueta eje Y: ylab. Igualar a una cadena de texto.
  + Color o colores: col. Igualar a un número o nombre de colores (o un vector de números o nombres de colores). Para conocer los nombres de colores escribe colors() en la consola.
  + Forma del punto: pch. Igualar a un número entre 1 y 25. Sólo cuando la figura dibuje puntos.
* Ejemplo: copia y pega el siguente bloque en la consola de R

plot(x=1:25, y=1:25, main='Puntos: formas y colores',

sub='Cada punto una forma y un color',

xlab='Número de punto y color',

ylab='Etiqueta eje Y', col=1:25, pch=1:25)

## **4.3 Estadísticas de datos cualitativos**

Suponemos que x es la variable que contiene los datos.

### 4.3.1 Tabla de frecuencias:

* **Absolutas:** table(x)
* **Relativas:** table(x)/length(x)
* **Porcentajes:** table(x)/length(x) \* 100

### 4.3.2 Gráficas:

* **Diagrama de barras:** barplot(height=table(x))
* **Diagrama de sectores (tarta):** pie(x=table(x))

**EJERCICIO 6:** Escribe la tabla de frecuencias absolutas y relativas de la variable grupo (para saber cuántos datos hay de cada grupo), y haz un diagrama de barras y otro de sectores, cambiando los colores y añadiendo títulos para practicar.

**FIN EJERCICIO 6**

## **4.4 Estadísticas de datos cuantitativos**

Suponemos que x es la variable que contiene los datos:

### 4.4.1 Tabla de frecuencias:

* Primero hay que “cortar” la recta en intervalos, y luego se pide la tabla de frecuencias de los intervalos.
* table(cut(x, breaks))
  + x los datos
  + breaks el número de intervalos, o el vector con los extremos sucesivos
* Ejemplo: copia y pega el siguente bloque en la consola de R

nota *# los datos*

table(nota) *# ¡no lo hace bien!*

table( cut(nota, breaks=0:10) ) *# la tabla de frecuencias con intervalos*

### 4.4.2 Estadísticos:

* **Media:** mean(x)
* **Mediana:** median(x)
* **Mínimo:** min(x)
* **Máximo:** max(x)
* **Cuantil de orden**p�**:** quantile(x, prob=p�)
* **Recorrido:** hay que calcularlo con su definición (max(x)-min(x)).
* **Recorrido intercuartílico:** IQR(x)
* **Cuasivarianza:** var(x)
* **Cuasidesviación típica:** sd(x)
* **Varianza:** ¡No está programada! Se puede conseguir con la operación var(x)\*(length(x)-1)/length(x)
* **Desviación típica:** ¡No está programada! Se puede conseguir con la operación sd(x)\*sqrt((length(x)-1)/length(x))
* **Coeficiente de variación**: ¡No está programado! Se puede conseguir una versión “cuasi” con la operación sd(x)/abs(mean(x)), y la versión “sin cuasi” con la operación (sd(x)\*sqrt((length(x)-1)/length(x)))/abs(mean(x)).

### 4.4.3 Gráficos

* **Diagrama de puntos:** stripchart(x) (para pocos datos)
* **Histograma:** hist(x, breaks, ...)
  + x los datos
  + breaks el número de intervalos, o el vector con los extremos sucesivos
* **Diagrama de caja:** boxplot(...) (se pueden poner varias muestras para comparar)

**EJERCICIO 7:** Escribe la tabla de frecuencias absolutas y relativas de la variable nota donde los intervalos correspondan a números enteros. Dibuja el histograma con esa misma partición en intervalos. Calcula la nota media y la cuasidesviación típica, así como la mínima y máxima.

* Sol.: Media = 5.01041675.0104167, cuasidesviación típica = 1.07020371.0702037, mínima = 1.71.7 y máxima = 7.77.7.

**FIN EJERCICIO 7**

**EJERCICIO 8:** Calcula la nota media y la cuasidesviación típica de cada grupo.

* Sol.: A = 4.54285714.5428571 y 0.73139940.7313994; B = 5.12173915.1217391 y 1.1855021.185502; C = 5.22820515.2282051 y 1.01746561.0174656; D = 4.77962964.7796296 y 1.03475081.0347508; E = 5.13870975.1387097 y 1.11404671.1140467.

**FIN EJERCICIO 8**

**EJERCICIO 9:** Calcula el percentil 66 de las notas de todos los alumnos, y también de los alumnos del grupo C.

* Sol.: 5.55.5 y 5.8085.808.

**FIN EJERCICIO 9**

**EJERCICIO 10:** Un alumno tiene una nota de 4.9. ¿Qué porcentaje, del total de alumnos, tiene una nota menor o igual que la suya? ¿Y qué porcentaje tiene una nota mayor o igual que la suya?

* Sol.: 46.87546.875% y 56.2556.25%

**FIN EJERCICIO 10**

**EJERCICIO 11:** Dibuja los diagramas de caja de las notas de cada grupo, para comparar el nivel de los grupos.

**FIN EJERCICIO 11**

# 5. Ejercicios de entrenamiento

**EJERCICIO 12:** Los siguientes vectores contienen datos sobre una muestra de coches americanos. Cada vector representa:

* coche: modelo de coche
* efi: eficiencia, en millas recorridas por galón de gasolina.
* ncil: número de cilindros del motor.
* cil: cilindrada total, en pulgadas cúbicas.
* pot: potencia, en CV (caballos de vapor).
* peso: peso, en miles de libras.
* tiempo: tiempo, en segundos, en recorrer un cuarto de milla, desde parado.
* tipo: tipo, automático o manual.
* marchas: número de marchas (hacia adelante).
* carbur: número de carburadores.

de modo que el primer datos de cada vector es del mismo coche, y así sucesivamente. Realiza las siguientes tareas de análisis de datos:

1. ¿Cuántos coches hay en la muestra?
   * Sol.: 3232.
2. ¿Cuántos de los coches son automáticos?
   * Sol.: 1313.
3. Escribe la tabla de frecuencias del tipo de coche. ¿Qué tipo es más abundante en la muestra?
   * Sol.: Autom=13;Manual=19�����=13;������=19. Es más abundante el Manual.
4. Calcula la media aritmética y la cuasidesviación típica de la eficiencia de los coches automáticos, y de los manuales, por separado. ¿Qué tipo es más eficiente?
   * Sol.: 24.392307724.3923077 y 6.16650386.1665038, para los automáticos, y 17.147368417.1473684 y 3.83396643.8339664, para los manuales. El más eficiente es el Automático, por tener la media más alta.
5. Dibuja un diagrama de sectores que informa sobre el número de cilindros de los coches de la muestra.
   * Sol.: -
6. Calcula la media de la potencia de los coches automáticos de 8 cilindros.
   * Sol.: 299.5299.5 CV.
7. Calcula la media de tiempos, en recorrer el cuarto de milla, para los coches manuales de 44 marchas delanteras.
   * Sol.: 20.02520.025 seg.
8. ¿Qué modelo de coche tiene la mayor cilindrada de la muestra? Escribe los cinco modelos de mayor cilindrada.
   * Sol.: Cadillac Fleetwood es el de mayor cilindrada, y los otros son, Hornet Sportabout, Duster 360, Lincoln Continental, Chrysler Imperial, Pontiac Firebird.
9. Calcula la media de cilindrada **por cilindro (no por motor)** de toda la muestra de cilindros de todos los coches.
   * Sol.: 35.028645835.0286458 pulgadas cúbicas.

**FIN EJERCICIO 12**

**EJERCICIO 13:** Si la variable conc recoge la concentración de plomo (en ppm) en el aire de cierta zona durante un día completo (1 muestreo cada 5 minutos):

1. ¿Cuántas observaciones se han recogido?
   * Sol.: 288288.
2. ¿Cuál ha sido la concentración máxima?
   * Sol.: 47.3447.34 ppm.
3. ¿En cuántos de los muestreos se ha superado la concentración de 40.040.0 ppm?
   * Sol.: 6161.
4. ¿Cuál ha sido la concentración media del día?
   * Sol.: 24.072291724.0722917 ppm.
5. ¿Cuáles fueron las 10 mediciones más bajas del día?
   * Sol.: 0.93,1.07,1.77,2.03,2.58,2.73,2.75,2.88,2.88,2.910.93,1.07,1.77,2.03,2.58,2.73,2.75,2.88,2.88,2.91.
6. Si la primera medida fue a las 00:00, ¿a qué hora del día se alcanzó la concentración máxima?
   * Sol.: A las 11:45.

**FIN EJERCICIO 13**