

1er. COLOQUIO DE R

Algunos topicos en R

CONTENIDO

Explorando R

- Instalación del programa.
- La Consola, preparación del ambiente de trabajo
- Creación objetos
- Manejo de datos externos: Texto y Excel
- Creación de vectores y matrices
- Operaciones con vectores y matrices.
- Gráficos de líneas, barras, pie, histogramas,

Aplicación en analisis multivariial

- Análisis multivariial. Componentes principales
- Clasificación no supervisada: cluster análisis, dendrograma y consensus (métodos jerárquicos).
- Clasificación supervisada. Análisis discriminante k vecinos mas cercanos.

Tabla de codigos ASCII

Cód igo	Símbolo
92	\
47	/
91	[
93]
40	(
41)
123	{
125	}
124	
60	<
62	>
61	=

Cód igo	Símbolo
34	"
126	~
58	:
59	;
64	@
94	^
35	#
36	\$
37	%
38	&
39	`

Explorando R

Introducción

R es un lenguaje de programación funcional, de uso público, disponible en Internet en la dirección: <http://www.r-project.org/>

El programa R tiene un módulo BASE, lo necesario para iniciar una sesión, el módulo BASE contiene las herramientas de programación, procedimientos estadísticos y gráficos frecuentemente utilizados. Adicional a este módulo, el proyecto R dispone de una cantidad muy grande de paquetes lo necesario para realizar un análisis estadístico. R también provee toda la documentación por manuales y libros de todos los paquetes propuestos.

El programa permite interactuar con el computador, pasos a paso, según los resultados que son mostrados en la consola, es decir que puede uno monitorear los procesos.

Una lista de órdenes pueden ser escritos en un archivo en ASCII con extensión .R y ser ejecutado mediante la orden: `source("ejemplo.R")`, si el nombre del programa es "ejemplo.R"

EL Programa R opera bajo diferentes plataformas: Windows, Linux, Unix y Mac. Las órdenes de ejecución de los procedimientos son las mismas en cada plataforma, la diferencia está en la instalación y en algunas funciones de menú que presenta cada versión.

Descargar e instalar la base R en la plataforma Windows.

Ingresar a la dirección de Internet:

<http://www.r-project.org/>

En el menú Download, seleccione CRAN y una dirección cerca a su lugar de instalación, por ejemplo:

<http://cran.at.r-project.org/> de Austria

Seleccione Windows (95 and later)

Luego dentro, seleccione BASE, que es la opción para el programa principal de R.

Finalmente encontrará una lista de información, en la cual está el programa instalador; por lo general tiene un nombre específico:

[R-2.6.0-win32.exe](#) (es la versión que en el momento de editar este manual, se dispone en Internet).

Procedimiento a seguir para instalar el programa.

Localizar el programa en Internet:

Pagina web: <http://www.r-project.org/>

Buscar el programa R-2.6.0-win32:

- Seleccione Download **CRAN**
- Seleccione un sitio, ejemplo: Australia. <http://cran.au.r-project.org/>
- La plataforma: **Windows**
- Seleccione: **base**
- Descargue el programa: [R-2.6.0-win32.exe](#)

Una vez que lo tiene en le disco duro, ejecute este programa presionando el Icon:



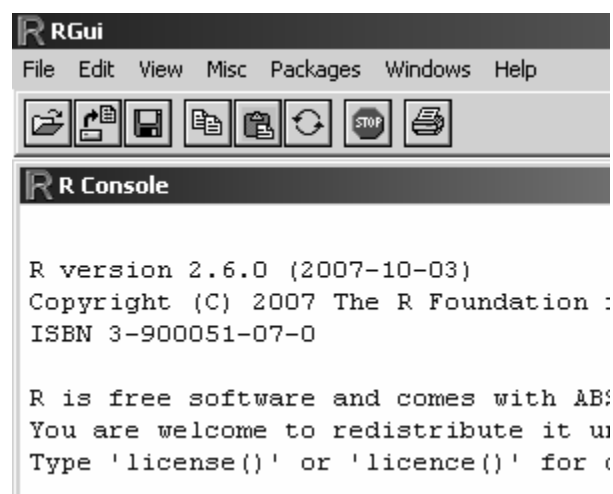
R 2.6.0

El programa es la base del sistema R. Para tareas mas avanzadas requiere bajar las funciones de los contribuyentes (mas de 1200).

Entrada y salida del programa.

Cuando el programa esta bien instalado, se ejecuta el Icon de R que esta en el escritorio. El ambiente de trabajo se llama consola. Para salir del programa, utilice el menú o escriba la orden q() y presione Enter.

Al ejecutar, se ingresa a una ventana de trabajo llamada la consola, que es el lugar para ejecutar ordenes y a la vez recibir los resultados en texto, los cuales pueden ser enviados a un procesador de texto y los gráficos otra ventana que puede ser enviado a un archivo de gráficos o insertar en un documento.



Como se observa las opciones de menú son muy pocas:

File.- para editar o ejecutar un conjunto de ordenes, cambiar el directorio o salir del programa.

Edit.- para editar datos existentes creados por R

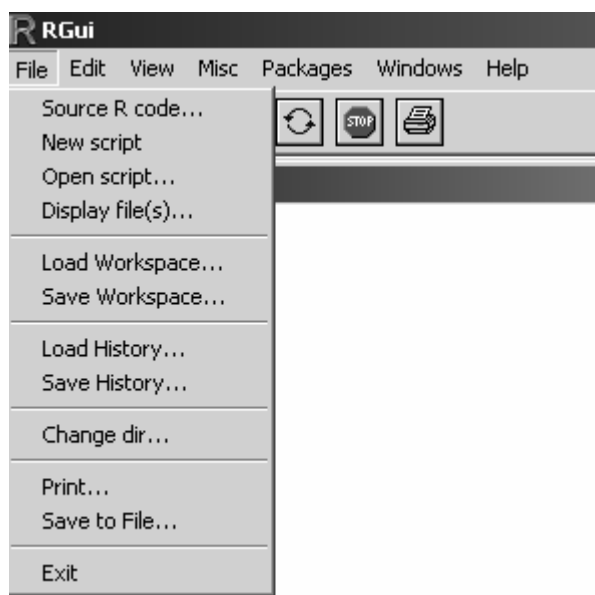
Misc.- para listar o remover objetos.

Packages.- es la opción mas importante, nos permite cargar un modulo de la biblioteca existente o cargar un modulo externo a nuestra base para trabajos futuros.

Windows.- manejar las ventanas activas.

Help.- Ayuda del programa.

Por ejemplo la opción File:



Las opciones New Script y Open Script son funciones para escribir y ejecutar procedimientos en R.

En general son pocas las opciones que ofrece el menú, esto significa que uno debe realizar todos los procesos en línea de comandos:

>

Este es el símbolo o señal para escribir una orden.

La Consola

Es el área de trabajo de R, lugar donde el usuario de R escribe las instrucciones necesarias para comunicarse con el sistema.

La señal de la línea de ordenes esta identificada con el símbolo “>”, si la orden es demasiado grande, presión enter y el sistema le indica con un ”+” para continuar escribiendo.

Es importante leer por lo menos una vez lo que aparece en la pantalla cuando ingresa al sistema “R”.

Es importante diferenciar entre ventana activa y no activa, la barra de titulo aparece iluminada si la ventana esta activa.

Escriba demo(graphics) y presione ENTER y continúe presionando ENTER según lo solicite:

```
> demo(graphics)
```

Para visualizar mejor, ir a la opción del menú, seleccione windows y ejecute **Tile** para observar los gráficos paralelo a la ventana de la consola.

NOTA.- Usted observa dos ventanas; una es la consola y la otra es la ventana de gráficos. Estos resultados pueden ser exportados para cualquier sistema.

```
> help.start()
```

Es la ayuda en formato HTML de R. Los sistemas de ayuda son amplios en el sistema R. (Ayuda por manuales, Ayuda por Internet, Ayuda interactiva)

Como Crear Objetos

Los objetos en R esta formado por vectores, matrices, textos, tablas y todo elemento que puede ser manipulado por R. Un conjunto de objetos puede ser integrado en una lista como objeto de R.

El nombre de los objetos es muy sensible a mayúsculas y minúsculas, así como cualquier orden en R. Por ejemplo “print” existe pero “PRINT” no existe. Por lo general son minúsculas, en pocos casos se utiliza mayúsculas, para esto es necesario conocer la sintaxis apropiadamente.

Nuestro primer objeto: Rdto de papa en Kg. en 10 parcelas.

```
rdto <- c(12, 16, 9, 14, 8, 12, 11, 7, 15, 13)
rdto = c(12, 16, 9, 14, 8, 12, 11, 7, 15, 13)
c(12, 16, 9, 14, 8, 12, 11, 7, 15, 13) -> rdto
```

Creado el objeto, se puede hacer algunas estadísticas:

```
min(rdto)
max(rdto)
mean(rdto)
sd(rdto)
summary(rdto)
```

Los datos pueden ser impresos y ordenados.

```
sort(rdto)
sort(rdto, decreasing = TRUE)
print(rdto)
```

Los datos pueden ser convertidos a matrices

```
fila <- rbind(rdto)
columna <- cbind(rdto)
fila
columna
```

Lista de objetos creados:

```
objects()
[1] "columna" "fila"      "rdto"
ls()
[1] "columna" "fila"      "rdto"
```

Remover objetos: rm

```
rm(fila)
rm(list=ls())
```

Limpiar la pantalla

Presione las teclas ctrl y “L” simultáneamente

Preparación del ambiente de trabajo: Crear el folder C:\COLOQUIO-R

Cada vez que ingrese a una nueva sesión de trabajo, localizar el lugar (fólder) para el manejo de su información.

Ubicación del área de trabajo

Get Working Directory: getwd()

```
getwd()
[1] "C:/Program Files/R/R-2.6.0"
```

Ir al folder c:\COLOQUIO-R

```
setwd("c:/COLOQUIO-R")
getwd()
[1] "c:/COLOQUIO-R"
```

O también:

```
setwd("c:\\COLOQUIO-R")
```

Algunas ordenes básicas:

```
list.files()      # Lista de archivos del área de trabajos
ls()              # Lista de Objetos: > ls()
x<-c("A","B","C") # crea el objeto x con elementos "A","B","C"
x                # Muestra el contenido del objeto x
rm(x)            # Borra el objeto "x"
```

Revisión de los operadores aritméticos, lógicos, de relación y matriciales.

Ingresa help, html y en packages localice el modulo BASE.

Estudie los operadores, tome nota y realice operaciones creando objetos.

Operaciones Aritméticas:

a) Localice la simbología. > help("Arithmetic")

```
x <- 2*5/3; y <- -2*2*2 ; v <- 2^3
z <- 13/%4 ; w <- 13%4 ; s <- c(1,2,3,4,5,6,7,8,9,10)
```

b) Revise el contenido de cada variable creada.

```
x ; y ; v ; z ; w ; s
```

c) Halle un nuevo objeto "p" de elementos {0,1} que represente a los valores pares e impares del objeto "s" ya creado.

```
p <- s%%2
p
```

Operadores logicos.

Revise la simbología y su uso > help("Logic")

```
x > y ; z < w ; y==v
```

Problema: Se tiene una lista numerada de personas que practican diferentes deportes; b=basket, f=football y v=voley. Se quiere saber quienes y cuantos practican cada uno de los deportes.

```
Id <- c(1,2,3,4,5,6,7,8,9,10,11,12)
deporte <- c("f","b","f","v","b","v","b","f","v","b","v","b")

# formar una tabla y listar los que juegan basket

participan<-data.frame(Id=1:12,deporte)
selecto <- participan[,2]=="b"
participan[selecto,]
```

```
# Contabilizar  
nrow(paricipan[selecto,])
```

Ejercicio: Contabilizar para los casos voley y fútbol

Secuencia de números.

Ejecute y describa cada resultado:

```
> 1:20  
> seq(1,20,1)  
> seq(from=1, to = 20, by=1)  
> seq(0,1,length=20)  
> seq(length=10, to = 20, from=5)
```

Halle las siguientes secuencias de números:

```
-  -3  -4  -5  -6  -7  -8  -9 -10 -11  
-  -3 -1  1  3  5  7  9 11  
-   3.0 3.2 3.4 3.6 3.8 4.0  
-   20 18 16 14 12 10  8  6
```

Funciones estadística y matemática:

sum(x) : suma de los elementos del objeto “x”
mean(x) : media de los elementos del objeto “x”
length(x): longitud de los elementos del objeto “x”
var(x) : variancia muestral de los elementos del objeto “x”
sd(x) : desviacion estandar
min(x) : mínimo valor de “x”
max(x) : máximo valor de “x”
log(x) : logaritmo neperiano
log(x,2) : logaritmo base 2
exp(x) : exponente

Ejercicios desarrollados:

1. crear dos objetos y determinar cuantos y cuales son los valores comunes.

```
“x” contiene { 5, 2, 7, 90, 24, 3, 6, 8,2, 5, 2}  
“y” contiene { 10, 7, 4, 7, 2, 10, 4, 3, 3, 1, 1, 3, 5}  
  
x <- c(5, 2, 7, 90, 24, 3, 6, 8,2, 5, 2)  
y <- c(10, 7, 4, 7, 2, 10, 4, 3, 3, 1, 1, 3, 5)
```

2. Identificar con verdadero o falso que elementos de “x” están en “y”

La orden `x%in%y`, indica cada elemento de “x” si esta en “y” con una respuesta TRUE o FALSE (verdadero o falso)


```
x%in%y
```

3. Identificar que elementos son:

`x[x%in%y]`, con esta instrucción se obtiene los elementos verdaderos, es decir los elementos de “x” que están en “y”

```
x[x%in%y]
```

4. Cuantos elementos son:

```
length( x[x%in%y] )    # cuenta los elementos.
```

5. Cuántos y cuáles son los valores únicos de ambos conjuntos.

`unique(objeto)` es la función para hallar los únicos valores que son distintos entre ellos.

```
unique(x[x%in%y])  
length(unique(x[x%in%y]))
```

Cada resultado puede ser almacenado en un objeto

Ayudas a las órdenes estudiadas:

```
help(seq)  
help(sum)  
help(unique)  
? mean  
? var
```

Manejo de datos.

Los datos pueden ser creados en el ambiente R o en forma externa por Tinn-R, notepad, word, excel, etc u otros sistemas estadísticos como SAS, MiniTab, SPSS, etc. R puede importar y exportar de otros ambientes.

En el ambiente R, nosotros asignamos un nombre al objeto de datos, para luego registrar los datos según como deseamos.

Creando datos en R

Objeto: camote.

Campos: variedad, días acumulados y evaluación en materia seca.

```
camote <- edit(data.frame())
```

Información: Materia seca del follaje en camote cortes cada 90 días

variedad	días	MS
ARB265	90	5
ARB265	180	13.1
ARB265	270	15.5
HELENA	90	2.3
HELENA	180	9.7
HELENA	270	12.8

Para modificar su contenido

```
> fix(camote)
```

Agregar los siguientes registros:

DLP3548	90	4.4
DLP3548	180	12.7
DLP3548	270	15.4

Observe los datos con las siguientes instrucciones:

```
camote
print(camote)
camote$variedad
camote$MS
camote[,1]
camote[,3]
```

Algunas estadísticas de Materia seca

```
mean(camote$MS)
mean(camote[,3])
```

Formas directas para acceder a un archivo externo

Caso texto: Tinn-R, notepad, wordpad, word

```
azucar <- read.table("azucares en yacon.txt", header=T)
```

Caso dBase

Archivo: Bolivia.dbf

```
canal<-odbcConnect(dsn="dBase files")
bolivia <- sqlFetch(canal, "Bolivia")
odbcCloseAll()
```

Caso Excel

Archivo: Labranza.xls

Tabla de datos: datos

```
canal <- odbcConnectExcel("Labranza.xls")
labranza <- sqlFetch(canal, "datos")
odbcCloseAll()
```

Si en excel se dispone de una tabla de datos en una hoja, este se puede grabar en el disco duro como un archivo de tipo CSV, en R podemos leer con la opción:

```
labranza <- read.csv("Labranza.csv", header=TRUE)
```

Caso Access

Proyecto: papacip.mdb

Tabla de datos: Morfologia

```
canal <- odbcConnectAccess("papacip.mdb")
morfologia <- sqlFetch(canal, "Morfologia")
odbcCloseAll()
```

Listado y creación de nuevas tablas en Access.

Conectar R al sistema ACCESS, a la base de datos papacip.mdb

```
canal <- odbcConnectAccess("papacip.mdb")
```

Lista las tablas que existen

```
sqlTables(canal)[, c(4, 3)]
```

La columna 4 es el tipo y la columna 3 es el nombre de la tabla.

Si en R se tiene algunas tablas (data.frame), estas pueden ser agregadas a la base de datos de access, activas en “canal”.

Por ejemplo “bolivia” es una tabla existente.

```
sqlSave(canal, bolivia)
```

Liste nuevamente las tablas de la base de datos activa.

```
sqlTables(canal)[, c(4, 3)]
```

Cierre todas las comunicaciones:

```
> odbcCloseAll()
```

Caso Minitab

Los datos almacenados por minitab deben ser en formato: Minitab Portable

Y su estructura debe ser numérica (no texto)

Para este ejercicio, considere el archivo "camote raices.MTP"

El procedimiento en R es:

```
> library(foreign)
> minitab<-read.mtp("camote raices.MTP")
```

Los datos leídos están en el objeto “minitab”, cuya estructura se puede visualizar con la orden:

```
> str(minitab)

> datos<-data.frame(minitab[1],minitab[2],minitab[3], minitab[4], minitab[5],
minitab[6], minitab[7], minitab[8])

> fix(datos)
```

Porque son 8 columnas

Caso SAS

Cuando uno dispone datos en SAS, estos deben ser almacenados en un formato de SAS, para esto se dispone del siguiente procedimiento en SAS para crear un archivo de datos portable.

```
/*
En SAS se prepara datos para R
*/
libname R sasv5xpt 'C:\COLOQUIO-R\maca.dat';
libname sitio 'C:\COLOQUIO-R';
DATA R.temp ;
set sitio.maca;
run;
quit;
```

Al correr este procedimiento en SAS, se genera automáticamente un archivo portable de nombre: "maca.dat", a partir del archivo de origen: maca.sas7bdat

Para leer en R, utilice el siguiente procedimiento:

```
> # Carga el paquete para la importación
> library(foreign)
> # lectura de los datos
> maca <- read.xport("maca.dat")
```

Caso SPSS

Cuando uno dispone datos creados por SPSS, estos pueden ser importados por R. El siguiente es un script escrito en Tinn-R:

```
# Lectura de un archivo SPSS
# archivo cip.sav
datos<-read.spss("Mouse survival.sav")
nuevo<-data.frame(datos)
# "nuevo" es la tabla que ahora contiene los datos del archivo SPSS
# para grabar el archivo en modo texto.
write.table(nuevo,"Mouse survival.txt", row.names=F, sep="\t")
# Este archivo puede ser leído desde Excel o de un editor de texto.
```

Algunas funciones que permiten resúmenes

Resumen estadístico de MS

```
resumen<-by(camote[,3], camote[,1], summary)
names(resumen)
resumen$"ARB265"
```

Promedio de MS por variedad

```
medias <- by(camote[,3], camote[,1], function(x) mean(x))
acciones <-as.matrix(medias)
colnames(acciones)<-c("promedio")
acciones
```

```
library(agricolae)
attach(camote)
tapply.stat(variedad, MS, mean)
detach(camote)
```

Hallar la media, variancia, el mínimo y máximo valor por variedad. Usar Tinn-R

```
media <-tapply.stat(variedad, MS, mean)
variancia <-tapply.stat(variedad, MS, var)
minimo <-tapply.stat(variedad, MS, min)
maximo <-tapply.stat(variedad, MS, max)
estadisticas <- data.frame(variedad= media[,1],media=media[,2],
variancia=variancia[,2],minimo=minimo[,2],maximo=maximo[,2])
```

Grabación del objeto en archivo permanente en el disco duro

```
write.table(camote, file = "camote1.txt", sep = ",", row.name=FALSE)
write.table(camote, file = "camote2.txt", sep = "\t", row.name=FALSE)
```

Es equivalente a:

```
write.table(camote, "camote1.txt", sep = ",", row.name=F)
```

```
write.table(camote, "camote2.txt", sep = "\t", row.name=F)
```

La orden `file.show` permite visualizar los archivos grabados:

```
file.show("camote1.txt")
file.show("camote2.txt")
```

Creación de vectores y matrices

Borre todos los objetos

```
> rm(list=ls())
```

Crear un vector numérico de 10 elementos con ceros

```
> v<- rep(0,10)
```

Crear un vector de elementos $\{2, 4, 3, 6, 5, 8\}$

```
> x <- c(2,4,3,6,5,8)
```

Crear una matriz de 4 filas y 5 columnas con ceros

```
> m <- rep(0,20)
> dim(m)<-c(4,5)
```

O también

```
> m<-matrix(m,4,5)
```

Crear la matriz A:

$$A = \begin{pmatrix} 3 & 1 & 5 & 4 & 2 \\ 1 & 6 & 7 & 5 & 3 \\ 5 & 3 & 8 & 4 & 1 \\ 2 & 1 & 4 & 9 & 2 \end{pmatrix}$$

Esta matriz tiene 4 filas y 5 columnas

Una forma puede ser:

```
> A <- c(3,1,5,2,1,6,3,1,5,7,8,4,4,5,4,9,2,3,1,2)
> dim(A)<-c(4,5)
```

También

```
> A<-matrix(A,4,5)
```

Juntando filas:

```
> A<-rbind(c(3,1,5,4,2),c(1,6,7,5,3),c(5,3,8,4,1),c(2,1,4,9,2))
```

Columnas

```
> A<-cbind(c(3,1,5,2),c(1,6,3,1),c(5,7,8,4),c(5,5,4,9),c(2,3,1,2))
```

Operaciones con vectores y matrices.

Considere el siguiente sistema de ecuaciones:

$$\begin{array}{rcrcrcrcrl} 2a & + & 3b & + & 3c & = & 10 \\ 4a & + & b & + & 2c & = & 11 \\ a & + & 6b & + & c & = & 9 \end{array}$$

Para resolver el sistema de ecuaciones, escribimos el sistema en forma matricial: $A \cdot x = b$. Y mediante R, utilizando la función solve() podemos resolver.

A son los coeficientes del sistema, “x” el vector incógnita y “b” el vector del lado derecho.

```
> A <- rbind(c(2,3,3), c(4,1,2), c(1,6,1))
> b <- c(10, 11,9)
> x <- solve(A,b)
> x
```

Graficos

Funciones principales en r:

```
plot() generico
barplot (categorica) Barras en vertical y horizontal
pie() (categorica) distribucion en torta
hist() (continua)
pairs() (continua)
matplot() (continua)
boxplot() (continua por categorias)
```

ejemplo: Produccion de papa 2000-2006

Fuente: MINAG - DGIA

INDICADORES	2000	2001	2002	2003	2004	2005	2006
Producción (miles t)	3.274	2.680	3.297	3.141	2.918	3.290	3.224
Sup Cos (miles Has.)	284	234	271	275	235	264	259
Rendimiento (t/ha.)	11,5	11,5	12,2	11,4	12,4	12,5	12,4

```
year <- 2000:2006
produccion <- c(3.274, 2.680, 3.297, 3.141, 2.918, 3.290, 3.224)
cosecha <- c(284, 234, 271, 275, 235, 264, 259)
names(produccion)<- year
names(cosecha)<- year
```

```
barplot(produccion,col="yellow",ylim=c(0,3.5))
plot(year,produccion,type="b", col="blue", lwd=1.5, lty=4, ylim=c(0,4))
grid(col="brown")

indice<-barplot(produccion,col="yellow", ylim=c(0,4), axes=F,las=2)
axis(2,seq(0,4,0.5))
par(new=T)
barplot(cosecha,col=0, ylim=c(0,350),axes=F,border=0,las=2)
axis(4,seq(0,350,50))
points(indice,cosecha,pch=17,col="blue",cex=1)
lines(indice,cosecha,cex=1,col="blue")
legend("top",c("produccion (miles ton.)", "cosecha(miles has.)"),
pch=c(22,17), col=c("black","blue"),lty=c(0,1), pt.bg=c("yellow","red"),
cex=1)
```

Produccion de papa 2006 por Dpto.

```
dpto <- c("Puno","Huanuco","Junin","La libertad", "Cusco", "Cajamarca",
"Ayacucho", "Lima", "Arequipa", "Apurimac", "Huancaavelica", "otros")
prod<-c(14,13,10,9,9,9,6,5,5,5,4,11)
names(prod) <-dpto
pie(prod, main="Gráfico N°-: Producción de papa\npor Departamentos 2006" ,
cex.main=0.9 )
pie(prod,col=colors()[sample(1:657,12)])
title(main="Gráfico N°-: Producción de papa\npor Departamentos 2006" ,
cex.main=0.9 )

library(plotrix)
pie3D(prod,border=0,col=colors(),explode=0.1,labels=names(prod),labelcex=1)
title(main="Gráfico N°-: Producción de papa\npor Departamentos 2006" ,
cex.main=0.9 )
pie3D(prod,border=0,col=colors(),labels=names(prod),labelcex=1)
title(main="Gráfico N°-: Producción de papa\npor Departamentos 2006" ,
cex.main=0.9 )
```

Ejercicio 1

Leer el archivo “camote lamolina.xls” a R y seleccionar una nueva tabla correspondiente a los extremos de la medida Beta caroteno formado por los cuantiles 10 y 90.

```
setwd("C:/COLOQUIO-R")
library(RODBC)
canal <-odbcConnectExcel("camote lamolina.xls")
camote<- sqlFetch(canal, "camote")
odbcCloseAll()
str(camote)
minimo<-quantile(camote$BC,0.10,na.rm=T)
maximo<-quantile(camote$BC,0.90,na.rm=T)
A<-na.omit(camote[ camote$BC <= minimo, ])
B<-na.omit(camote[ camote$BC >= maximo, ])
selecto<-rbind(A,B)
```

Ejercicio 2

Construir un histograma, polígono de frecuencia, la tabla de frecuencia, La Ojiva y estadísticas de los datos agrupados de la variable Materia seca MS (7)

```
library(agricolae)
attach(camote)
H <- hist(MS, col="yellow", border="blue")
polygon.freq(H,col="brown")
table.freq(H)
ojiva.freq(H,type="b",col="blue")
stat.freq(H)
```

Ejercicio 3

Probar la normalidad de la respuesta Protein y graficar mediante cuantiles de la normalidad.

```
shapiro.test(Protein)
qqnorm(Protein)
qqline(Protein)
# probar si la materia seca (MS) sigue una distribución normal
```

Ejercicio 4

Hallar las correlaciones de beta caroteno con las otras medidas en la población y en las subpoblaciones de bajo y alto valor de caroteno.

```
# Poblacion
correlation(BC, camote[,c(-1,-6)])
correlation(A$BC, A[,c(-1,-6)])
correlation(B$BC, B[,c(-1,-6)])
```

Ejercicio 5

Seleccione una muestra aleatoria de 30 clones y halle las estadísticas de la materia seca y compare con las estadísticas de la población.

Medidas: Media, Mediana, Simetría, curtosis, desviación estándar, desviación media, rango y la correlación con el Beta caroteno.

```
# Población camote de 206 clones.
# Muestra de 30 clones
detach(camote)
muestra <- sample(1:206, 30)
selecto <- camote[muestra,]
attach(selecto)
media <- mean(MS, na.rm=TRUE)
mediana <- median(MS, na.rm=TRUE)
skewness(MS)
kurtosis(MS)
sd(MS, na.rm=TRUE)
sqrt(sum(abs(MS - media),na.rm=TRUE)/30)
rango <- max(MS)-min(MS)
correlation(MS, BC)
detach(selecto)
```

Ejercicio 6

Con la población halle el **Scatterplot** para mostrar la asociación y el histograma de las medias BC, Protein, Fe, Zn, Ca que corresponden a las columnas: 6, 2, 3, 4 y 5.

```
# Población camote
source("panel.R")
camote1<- na.omit(camote)
pairs(camote1[,c(6,2:5)], panel=panel.smooth, cex = 0.8, pch = 19,
bg="light blue", diag.panel=panel.hist, cex.labels = 1, font.labels=1)
pairs(camote1[,c(6,2:5)], lower.panel=panel.smooth, upper.panel=panel.cor)
```

Ejercicio 7

Con la población de camote1 asigne clases a la proteína y hierro según los percentiles, ejemplo: para proteína: p25 "Bajo", p75 "Alto", en otro caso "Medio"

```
#proteina
camote2 <-data.frame(camote1, clase.Protein="medio", clase.Fe="moderado",
clase.BC="regular")
P25 <- quantile(camote2[,2],0.25)
P75 <- quantile(camote2[,2],0.75)
F25 <- quantile(camote2[,3],0.25)
F75 <- quantile(camote2[,3],0.75)
B25 <- quantile(camote2[,6],0.25)
B75 <- quantile(camote2[,6],0.75)

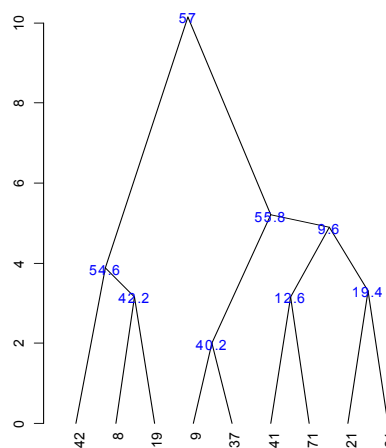
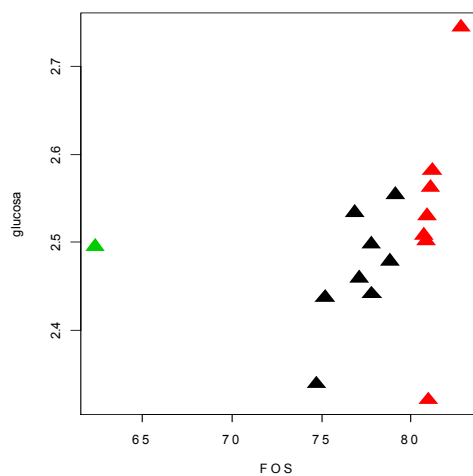
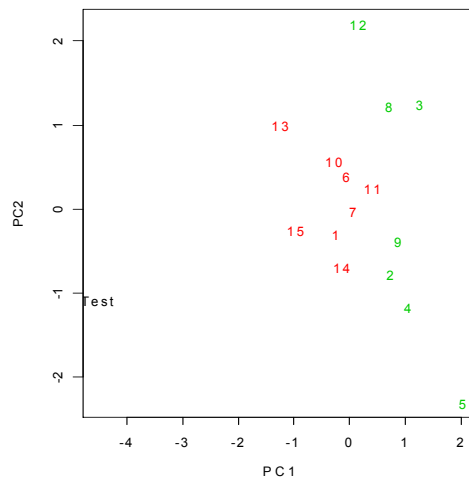
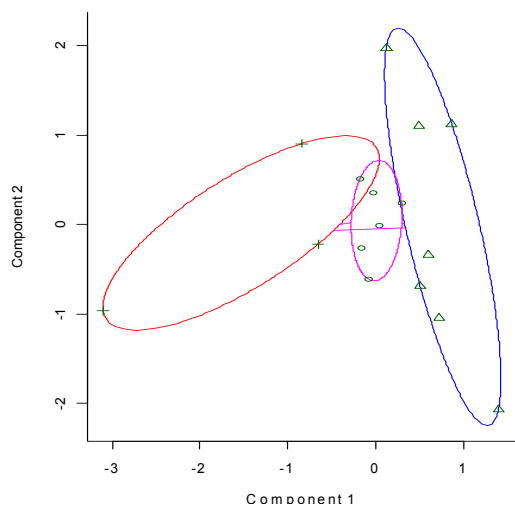
camote2[,8]<-as.character(camote2[,8])
camote2[,9]<-as.character(camote2[,9])
camote2[,10]<-as.character(camote2[,10])
camote2[camote2[,2] <= P25,8]<-"bajo"
camote2[camote2[,2] >= P75,8]<-"alto"
camote2[camote2[,3] <= F25,9]<-"poco"
camote2[camote2[,3] >= F75,9]<-"suficiente"
camote2[camote2[,6] <= B25,10]<-"malo"
camote2[camote2[,6] >= B75,10]<-"bueno"
matriz<- table(camote2[,10], camote2[,8])
matriz
matriz<-matriz[,c(2,3,1)]
matriz<-matriz[c(2,3,1),]

barplot(matriz, xlab="Proteina",col= c(3,5,7))
legend("topleft",c("malo", "regular", "bueno"), pch=22, pt.bg=c(3,5,7),
title="BC",cex=0.8)
title(main="Relacion del Betacaroteno y Proteina\nen clones de
camote",cex.main=0.8)

barplot(matriz, xlab="Proteina",col= c(3,5,7), beside=T)
legend("topleft",c("malo", "regular", "bueno"), pch=22, pt.bg=c(3,5,7),
title="BC", cex=0.8)
title(main="Relacion del Betacaroteno y Proteina\nen clones de
camote",cex.main=0.8)
```

Analisis Multivarial

- Analisis multivarial. Componentes principales
- Clasificacion no supervisada: cluster analysis, dendrograma y consensus (metodos jerarquicos).
- Clasificacion supervisada. Analisis discriminante.y k vecinos mas cercanos.



Componentes principales

R dispone de funciones para hallar las componentes principales a partir de correlaciones o covariancias de las variables observadas.

Para este estudio considere el siguiente archivo: “azucares en yacon.txt”

FOS, glucosa, fructosa y sacarosa

Una variable adicional que es la identificación de cada registro (ID)

```
> azucar <- read.table("azucares en yacon.txt",header=TRUE)
> datos <- azucar[,-1]
> correl <- cor(datos)
> valores <- eigen(correl)
> valores
```

\$values

```
[1] 2.14206323 1.24405952 0.59276138 0.02111586
```

\$vectors

	[,1]	[,2]	[,3]	[,4]
[1,]	0.6688031	0.10715436	0.1680706	0.71622111
[2,]	0.3107702	-0.62950240	-0.7115513	-0.02904034
[3,]	-0.1326303	-0.76885749	0.6184480	0.09375192
[4,]	-0.6622186	-0.03320897	-0.2880435	0.69093745

Para hallar los valores correspondientes a cada componente principal, es necesario primero estandarizar los datos, en este caso “datos”. Si usamos la función estándar descrita anteriormente (Pág. 34). Las 4 columnas se pueden estandarizar

```
> a1 <- estandar(datos,1)
> a2 <- estandar(a1,2)
> a3 <- estandar(a2,3)
> a4 <- estandar(a3,4)
```

Entonces se forma una matriz A.

```
> A <- as.matrix(a4)
```

Las ponderaciones de las componentes se encuentran en el objeto valores, entonces obtenemos la matriz X:

```
> X <- valores$vectors
```

Las componentes principales se obtienen multiplicando la matriz A por X.

```
> CP1 <- A%*%X
```

Utilizando la matriz variancia-covariancia

```
> covar <- cov(datos)
> valores <- eigen(covar)
> valores

$values
[1] 31.054475402  0.400136559  0.096671548  0.007448548

$vectors
      [,1]      [,2]      [,3]      [,4]
[1,]  0.854321447  0.47752777 -0.2042005 -0.02010622
[2,]  0.005110203 -0.01540644 -0.1124739  0.99352209
[3,] -0.013639922 -0.37397769 -0.9207939 -0.10996962
[4,] -0.519540858  0.79490286 -0.3127147 -0.02040282
```

El aporte de cada componente se puede hallar según la proporción respecto al total.

En el caso de correlación:

```
> valores$value*100/sum(valores$value)
[1] 53.5515808 31.1014881 14.8190346  0.5278965
```

En covariancia:

```
> valores$value*100/sum(valores$value)
[1] 98.40216440 1.26791076 0.30632266 0.02360218
```

Y las componentes pueden ser calculadas como:

```
> X <- valores$vectors
```

Las componentes principales se obtienen multiplicando la matriz A por X.

```
> CP2 <- A%%X
```

Otra función que permite hallar las componentes principales es: `princomp()`. Si usa correlaciones el parámetro `cor=TRUE`, si usa covariancias es `cor=FALSE`.

```
> componentes<-princomp(datos, cor=TRUE)
> summary(componentes)
```

Importance of components:

	Comp.1	Comp.2	Comp.3	Comp.4
Standard deviation	1.4635789	1.1153742	0.7699100	0.145312978
Proportion of Variance	0.5355158	0.3110149	0.1481903	0.005278965
Cumulative Proportion	0.5355158	0.8465307	0.9947210	1.000000000

```
> names(componentes)
[1] "sdev"      "loadings" "center"    "scale"     "n.obs"     "scores"    "call"
```

Cada objeto proporciona información sobre las componentes principales.

La matriz de componentes esta en: `componentes$scores` que tiene la misma información obtenida en CP para correlaciones. Para obtener el mismo valor y signo, se debe particularizar cada matriz; por ejemplo si se normaliza ambas matrices CP1 y CP2 para la primera componente se tiene:

```
> CP1[,1]/CP1[1,1]
> componentes$scores[,1]/ componentes$scores[1,1]
```

Ambos vectores son iguales en signo y magnitud.

De igual forma para covariancias.

```
> componentes<-princomp(datos, cor=FALSE)
> summary(componentes)
```

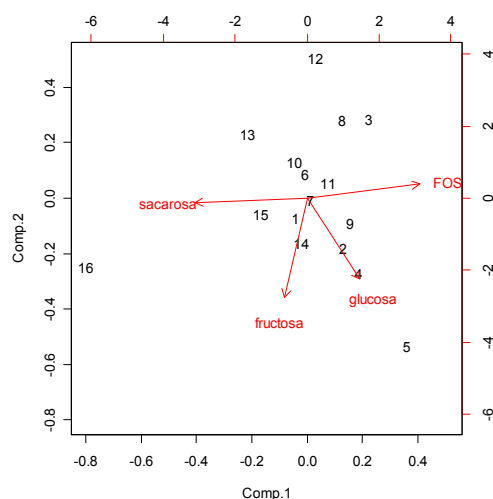
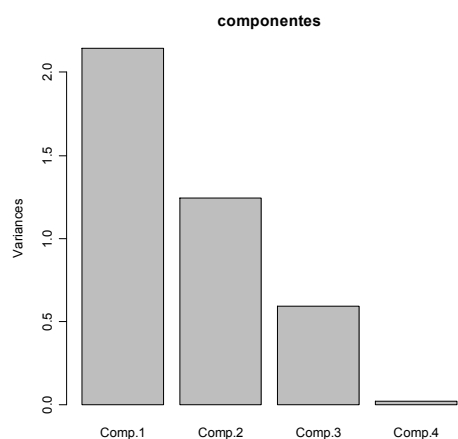
Importance of components:

	Comp.1	Comp.2	Comp.3	Comp.4
Standard deviation	5.3956993	0.61247696	0.301047465	0.0835644307
Proportion of Variance	0.9840216	0.01267911	0.003063227	0.0002360218
Cumulative Proportion	0.9840216	0.99670075	0.999763978	1.0000000000

Las proporciones acumulativas son las mismas encontradas con el procedimiento anterior.

Otra información es obtenida de:

```
> plot(componentes)
> biplot(componentes)
```



Clasificación no supervisada. Conglomerados (cluster).

Permite formar grupos similares según las medidas que uno considere para el agrupamiento.

Hay muchas funciones para ello, las principales son:

dist() para el cálculo de la distancia entre individuos
hclust() procedimiento para los agrupamientos.
cutree() para hacer los cortes del dendrograma.

y otras funciones de la librería cluster y agricolae como se indica

```
plclust()  
rect.hclust()  
daisy()  
pam()  
clusplot()  
as.dendrogram()  
cut()  
consensus()  
hcut()
```

Subir los paquetes

```
> library(cluster)  
> library(agricolae)
```

Análisis cluster

Métodos para calcular distancia.

“euclidian”.- la distancia entre los vectores x e y
“maximum”.- la distancia Máxima entre dos componentes x,y
“manhattan”: la distancia Absoluta entre los dos vectores
“canberra”.- $\text{sum}(|x_i - y_i| / |x_i + y_i|)$.
“minkowski”.- la norma de p componentes
“binary”.- distancia como una proporción de frecuencias.

“**binary**”.- La distancia es determinada como una proporción entre el total de celdas de presencia del marcador (una sola vez) entre el total de celdas con presencia del marcador.

Ejemplo: Para hallar la distancia entre A y B, se tiene 5 celdas con presencia del marcador { (1,1), (1,1), (1,0), (1,1), (1,1) } y celdas con una sola vez { (1,0) }, entonces la distancia es $1/5 = 0.2$. En otro caso, Distancia (B, D), total con presencia del marcador { (1,1), (1,0), (0,1), (1,0), (1,1) } y celdas con una sola vez { (1,0), (0,1), (1,0) }, entonces la distancia (B, D) es igual a $3/5 = 0.6$

Métodos para hallar dendrogramas.