Computación y Programación Básica

2020

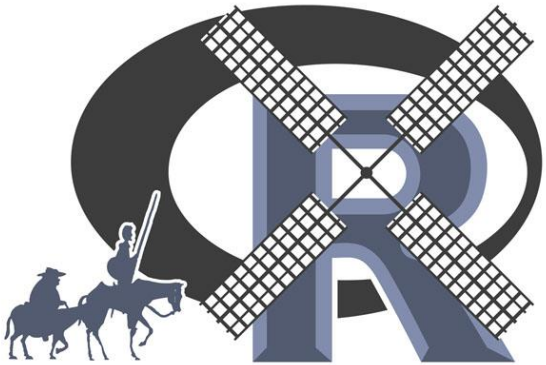
Billy Ernst y Diego Narváez

Departamento de Oceanografía

Cabina 10

biernst@udec.cl

Fono: 4012



Contenido

- Almacenamiento de datos
- Dataframes (crear, concatenar)
- Listas (crear, concatenar)
- Indexación
- Filtrar por varios criterios
- Ordenar

Tipos de estructuras para almacenar datos en R

list

vector

```
1  3  10 -2  7
```

matrix

```
1   3  10 -2  7
2   1  3  10 -2
3   1  3  10 -2
4   1  3  10 -2
```

dataframe

```
> worms[order(Slope),]
          Field.Name Area Slope Vegetation Soil.pH  Damp Worm.density
5    Gunness.Thicket  3.8     0      Scrub     4.2 FALSE            6
8            Ashurst  2.1     0     Arable     4.8 FALSE            4
9        The.Orchard  1.9     0    Orchard     5.7 FALSE            9
15        Pond.Field  4.1     0     Meadow     5.0  TRUE            6
16      Water.Meadow  3.9     0     Meadow     4.9  TRUE            8
12      North.Gravel  3.3     1  Grassland     4.1 FALSE            1
19        Gravel.Pit  2.9     1  Grassland     3.5 FALSE            1
2     Silwood.Bottom  5.1     2     Arable     5.2 FALSE            7
6           Oak.Mead  3.1     2  Grassland     3.9 FALSE            2
13      South.Gravel  3.7     2  Grassland     4.0 FALSE            2
18        Pound.Hill  4.4     2     Arable     4.5 FALSE            5
3      Nursery.Field  2.8     3  Grassland     4.3 FALSE            2
7       Church.Field  3.5     3  Grassland     4.2 FALSE            3
10     Rookery.Slope  1.5     4  Grassland     5.0  TRUE            7
4        Rush.Meadow  2.4     5     Meadow     4.9  TRUE            5
14 Observatory.Ridge  1.8     6  Grassland     3.8 FALSE            0
17         Cheapside  2.2     8      Scrub     4.7  TRUE            4
11       Garden.Wood  2.9    10      Scrub     5.2 FALSE            8
20         Farm.Wood  0.8    10      Scrub     5.1  TRUE            3
1       Nashs.Field  3.6    11  Grassland     4.1 FALSE            4
```

Escalar

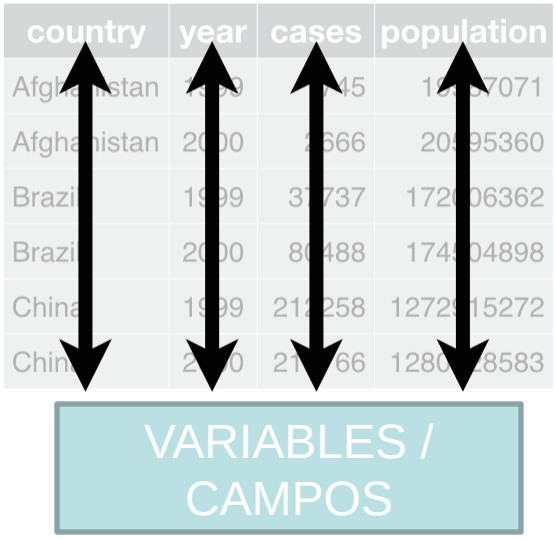
Vector

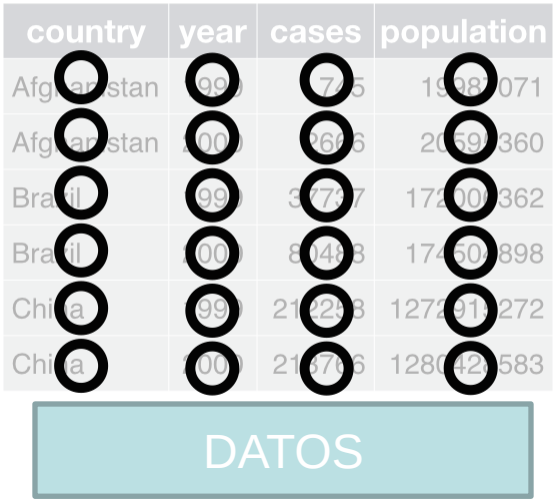
Matriz

Dataframe

Lista

Estructura de datos





Procedimientos

- Conocer o inspeccionar nuestra data
  - Que tipo de variables son? (categóricas, numéricas)
  - Eliminar registros (erróneos, Na’s)
  - Que estructura tiene (cuan balanceada esta la data)
  - Necesitamos analizar un subconjunto de la data (variables o campos)
  - Filtrar/Indexar registros por criterios varios

Dataframes

- Un dataframe es un objeto que permite albergar distintas clases de variables al mismo tiempo (diferencia con vectores y matrices)
  - Numéricas
  - Lógicas
  - Enteras
  - Texto

Ejemplo Dataframe

Numérica

Lógica

Categórica

Field Name	Area	Slope	Vegetation	Soil pH	Damp	Worm Density
Nash's Field	3.6	11	Grassland	4.1	F	4
Silwood Bottom	5.1	2	Arable	5.2	F	7
Nursery Field	2.8	3	Grassland	4.3	F	2
Rush Meadow	2.4	5	Meadow	4.9	T	5
Gunness' Thicket	3.8	0	Scrub	4.2	F	6
Oak Mead	3.1	2	Grassland	3.9	F	2
Church Field	3.5	3	Grassland	4.2	F	3
Ashurst	2.1	0	Arable	4.8	F	4
The Orchard	1.9	0	Orchard	5.7	F	9
Rookery Slope	1.5	4	Grassland	5	T	7
Garden Wood	2.9	10	Scrub	5.2	F	8
North Gravel	3.3	1	Grassland	4.1	F	1
South Gravel	3.7	2	Grassland	4	F	2
Observatory Ridge	1.8	6	Grassland	3.8	F	0
Pond Field	4.1	0	Meadow	5	T	6
Water Meadow	3.9	0	Meadow	4.9	T	8
Cheapside	2.2	8	Scrub	4.7	T	4

Como creamos un dataframe?

1a ←read.table() Procedencia externa, archivos separados por espacio o tabulador

1b ← read.csv() Procedenia externa, archivos separados por coma.

2 data.frame()

Procedencia "interna". Registros son creados dentro de R a partir de otros objetos o funciones.

```
emp.data <- data.frame(
   emp_id = c (1:5),
   emp_name = c("Rick","Dan","Michelle","Ryan","Gary"),
   salary = c(623.3,515.2,611.0,729.0,843.25),

   start_date = as.Date(c("2012-01-01", "2013-09-23", "2014-11-15", "2014-05-11",
      "2015-03-27")),
   stringsAsFactors = FALSE
)
```

Como se ve el ejemplo?

```
# Print the data frame.
print(emp.data)
```

```
  emp_id    emp_name     salary     start_date
1      1      Rick        623.30     2012-01-01
2      2      Dan         515.20     2013-09-23
3      3      Michelle    611.00     2014-11-15
4      4      Ryan        729.00     2014-05-11
5      5      Gary        843.25     2015-03-27
```

Como creamos una lista?

- Función **list**(elem1,elem2,…., elemn)

```
10 list_data <- list(c("Jan","Feb","Mar"), matrix(c(3,9,5,1,-2,8), nrow = 2),
11                   matrix(c("rojo","verde","azul","cafe"),nrow=2,ncol=2))
12
13 # Give names to the elements in the list.
14 names(list_data) <- c("Vector_Meses", "Matriz_A", "Matriz_B")
15
16 # Show the list.
17 print(list_data)
18
```

```
$`Vector_Meses`
[1] "Jan" "Feb" "Mar"

$Matriz_A
     [,1] [,2] [,3]
[1,]    3    5   -2
[2,]    9    1    8

$Matriz_B
     [,1]    [,2]
[1,] "rojo"  "azul"
[2,] "verde" "cafe"
```

Inspeccionar

- Funciones de R:
  - summary()
  - head(), tail()
  - str()

Valores fuera de rango y Na

- Funciones de R:
  - is.na(crabs) -> indica cuantos NA tiene la base de datos
  - na.omit(crabs) -> Elimina todos los registros con NA.
  - Eliminar puntos atípicos? Error digitación?

Subíndice - Índice

- Para Indexar en R se utilizan paréntesis cuadrados [ ], al igual que para vectores y matrices

worms[3,5]

[1] 4.3

worms[14:19,7]

```
> worms[1:5,2:3]
  Area Slope
1  3.6    11
2  5.1     2
3  2.8     3
4  2.4     5
5  3.8     0
```

Operadores lógicos

Operator	Description
<	less than
<=	less than or equal to
>	greater than
>=	greater than or equal to
==	exactly equal to
!=	not equal to
!x	Not x
x | y	x OR y
x & y	x AND y

Indexar un Dataframe

Comando	Significado
data[n,]	Selecciona todas las columnas de la fila n del dataframe
data[-n,]	Descarta la fila n del dataframe
data[1:n,]	Selecciona todas las columnas de las filas 1 a la n del dataframe
data[-(1:n),]	Descarta las filas 1 a n del dataframe
data[c(i,j,k),]	Selecciona todas las columnas de las filas i,j,k del dataframe
data[x>y,]	Selecciona todas las columnas de las filas mayores a y
data[,m]	Selecciona todas las filas de la columna m
data[,1:m]	Selecciona todas las filas de las columnas 1 a la m
data[,-(1:m)]	Descarta las columnas 1 a la m
data[,c(i,j,k)]	Selecciona todas las filas de las columnas i, j, k
data[,x>y]	Seleccopma todas las filas de las columnas mayores a y
data[,c(1:m,i,j,k)]	Agrega columnas duplicadas i,j,k al dataframe
data[x>y,a!=b]	Extrae ciertas filas y columnas
data[c(1:n,i,j,k),]	Agrega filas duplicadas i,j,k al dataframe

Formas alternativas de seleccionar - VARIABLES

– Seleccionar variables (Mantener) o Excluir variables (Eliminar)

Quiero trabajar con un subconjunto de variables o continuar con todo el dataframe?

de “worms” nos interesa **Soil.pH** y **Worm.density**

```
worms2<- cbind(worms$Soil.pH,worms$Worm.density)

worms2<- worms[,c("Soil.pH","Worm.density")]

worms2<- worms[,c(5,7)]
```

Formas alternativas de seleccionar - VARIABLES

– Seleccionar rango de variables

```
worms2<- worms[,1:4]
```

– Eliminar un rango de variables

```
worms2<- worms[,-(1:4)]
```

Formas alternativas de seleccionar - OBSERVACIONES

- Mantener o excluir observaciones

  Las primeras 5 observaciones

  worms[1:5,]

  En la mayoría de los casos queremos utilizar algunos operadores lógicos para filtrar registros u observaciones.

crabs3 <- crabs2[(crabs2$Sex=="F" & crabs2$Month > 9),]

Usando expresiones lógicas para seleccionar filas de dataframes

worms[Damp ==T,]

worms[Worm.density>median(Worm.density) & Soil.pH<5.2,]

worms[-(6:15),] Extrae filas

worms[!(Vegetation==“Grassland”),] Extrae por selección lógica

Función subset

- subset(x, y, select)

x: dataframe a ser subdividido

y: expresión lógica que determina los elementos o filas que se deben mantener

select: expresión que indica que variables

```
subset(crabs2,(crabs2$Sex=="F" & crabs2$Month > 9),select=c(CL))
```

Ordenar un dataframe

- Función ***order***

```
> worms[order(Slope),]
          Field.Name Area Slope Vegetation Soil.pH  Damp Worm.density
5    Gunness.Thicket  3.8     0      Scrub     4.2 FALSE            6
8            Ashurst  2.1     0     Arable     4.8 FALSE            4
9        The.Orchard  1.9     0    Orchard     5.7 FALSE            9
15        Pond.Field  4.1     0     Meadow     5.0  TRUE            6
16      Water.Meadow  3.9     0     Meadow     4.9  TRUE            8
12      North.Gravel  3.3     1  Grassland     4.1 FALSE            1
19        Gravel.Pit  2.9     1  Grassland     3.5 FALSE            1
2     Silwood.Bottom  5.1     2     Arable     5.2 FALSE            7
6           Oak.Mead  3.1     2  Grassland     3.9 FALSE            2
13      South.Gravel  3.7     2  Grassland     4.0 FALSE            2
18        Pound.Hill  4.4     2     Arable     4.5 FALSE            5
3      Nursery.Field  2.8     3  Grassland     4.3 FALSE            2
7       Church.Field  3.5     3  Grassland     4.2 FALSE            3
10     Rookery.Slope  1.5     4  Grassland     5.0  TRUE            7
4        Rush.Meadow  2.4     5     Meadow     4.9  TRUE            5
14 Observatory.Ridge  1.8     6  Grassland     3.8 FALSE            0
17         Cheapside  2.2     8      Scrub     4.7  TRUE            4
11       Garden.Wood  2.9    10      Scrub     5.2 FALSE            8
20         Farm.Wood  0.8    10      Scrub     5.1  TRUE            3
1        Nashs.Field  3.6    11  Grassland     4.1 FALSE            4
```

Ordenar – función ***order***

- Orden inverso

```
> worms[rev(order(Slope)),]
```

- Mayor número de niveles de orden

```
> worms[order(Vegetation,Worm.density),]
```

- Seleccionar y ordenar

```
> worms[order(Vegetation,Worm.density),c(4,7,5,3)]
```

Función table()

– A menudo queremos agrupar la data en forma tabular

```
tb1<-table(crabs$Year,crabs$Month)
```

```
      1    2    3    4    5    6    7    8    9   10   11   12
2008 4255 1149  915 1712    0    0    0    0    0 1511 2205 2111
2009    0    0    0    0    0    0    0    0    0 1593 1674  364
2010    0    0    0    0    0    0    0    0    0 1647  423 1603
2011 5408 4779 1423  543    0    0    0    0    0 1684 3632 2985
2012  876 1179 1473  980    0    0    1   18   14 1900 2908 1114
2013 4377 2585 1583 2041  303    5   23    5    0 2447 2440 3123
2014    2    0    0    0    0    0    0    0    0    0    0    0
> 
```

Como modificar datos Funciones trunc(),round()

- A menudo necesitamos redondear o truncar valores numéricos.
- Recuerde la variable CL (longitud del ejemplar). Generar categorías.

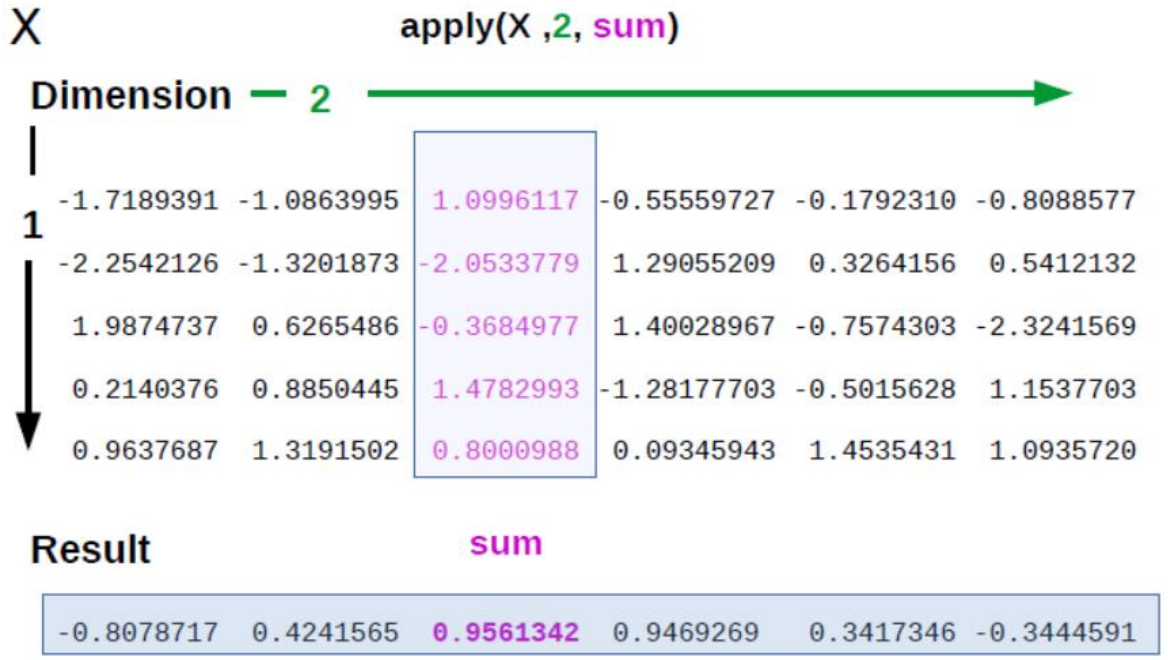
```
t.size<-trunc(crabs2$CL)
```

Agregar esa nueva variable al dataframe crabs2.

```
crabs2$t.size<-t.size
```

Función apply()

- Una función muy útil para hacer operaciones por fila o columna de una matriz
-



apply

```
(X <- matrix(rep(1:6,each=5), nrow=5, ncol=6))

# Suma las columnas`
apply(X, 1, sum)
# Adjunta el vector columna de la suma a la matriz de datos
X.all<-cbind(X,apply(X, 1, sum))
# Adjunta el vector fila de la suma de la matriz de datos
X2.all<-rbind(X.all,apply(X.all, 2, sum))
```

Uniendo 2 dataframes

- Al trabajar con bases de datos es común tener una estructura jerárquica.
- Dos tablas se pueden vincular a través de algunas variables o campos comunes.

Veamos el siguiente ejemplo

```
> producers
   director nacionalidad
1 Spielberg          US
2  Scorsese          US
3 Hitchcock          UK
4 Tarantino          US
5  Polanski      Poland
```

```
> movies
   director                titulo
1 Spielberg               Super 8
2  Scorsese           Taxi Driver
3 Hitchcock                Psycho
4 Hitchcock  North by Northwest
5 Spielberg Catch Me If You Can
6 Tarantino      Reservoir Dogs
7  Polanski             Chinatown
```

```
   director nacionalidad                titulo
1 Hitchcock          UK                Psycho
2 Hitchcock          UK  North by Northwest
3  Polanski      Poland              Chinatown
4  Scorsese          US           Taxi Driver
5 Spielberg          US                Super 8
6 Spielberg          US Catch Me If You Can
7 Tarantino          US      Reservoir Dogs
```

Código

```r
producers <- data.frame(
  director =  c("Spielberg","Scorsese","Hitchcock","Tarantino","Polanski"),
  nacionalidad = c("US","US","UK","US","Poland"),
  stringsAsFactors=FALSE)

# Create destination dataframe
movies <- data.frame(
  director = c("Spielberg",
               "Scorsese",
               "Hitchcock",
               "Hitchcock",
               "Spielberg",
               "Tarantino",
               "Polanski"),
  titulo = c("Super 8",
             "Taxi Driver",
             "Psycho",
             "North by Northwest",
             "Catch Me If You Can",
             "Reservoir Dogs","Chinatown"),
  stringsAsFactors=FALSE)

# Merge two datasets
m1 <- merge(producers, movies, by.x = "director")
m1
dim(m1)
```