Formatos de bases de datos

Andrea P. Goijman, PhD

Actualizado 6 Diciembre de 2018

1) Formato de matriz

Existen varias formas de ingresar los datos luego que fueron colectados en el campo.

La primer opción que vamos a ver se ve como el archivo DATOS_MATRIZ.CSV

```
# remueve datos viejos
rm(list=ls(all=TRUE))

# indicar directorio de trabajo
setwd('C:\\Users\\goijman.andrea\\Dropbox\\Materiales interesantes\\Scripts_Modelos\\Formatos
 BD')

#leer el csv
data<-read.csv('datos_MATRIZ.csv', sep = ";", header = T)

#explorar datos
head(data, 10)
```

```
##    PREDIO PUNTO   ID ID.REP REP COLME PSELO TYRSA PYRRU MIMPA LEILO SICLU
## 1      CZ    B1 CZB1  CZB11   1     3    NA    NA    NA    NA    NA     3
## 2      CZ    B1 CZB1  CZB12   2    NA    NA     1    NA    NA    NA     8
## 3      CZ    B2 CZB2  CZB21   1    NA     2     1    NA    NA    NA     1
## 4      CZ    B2 CZB2  CZB22   2    NA    NA    NA    NA    NA    NA     7
## 5      CZ    B3 CZB3  CZB31   1    NA    NA    NA    NA    NA     1     2
## 6      CZ    B3 CZB3  CZB32   2     1    NA    NA     2    NA    NA    NA
## 7      CZ    B4 CZB4  CZB41   1    NA    NA     1    NA    NA    NA     4
## 8      CZ    B4 CZB4  CZB42   2    NA    NA     1    NA    NA    NA     1
## 9      CZ    B5 CZB5  CZB51   1    NA    NA     2    NA    NA     2     1
## 10     CZ    B5 CZB5  CZB52   2    NA    NA    NA    NA    NA    NA    NA
##    ZONCA RHOFR SPIBA PASDO ZENAU PATMA
## 1      1    NA    NA    NA    NA    NA
## 2      1    NA    NA    NA    NA    NA
## 3      1     1    NA    NA    NA    NA
## 4     NA    NA     3     1    NA    NA
## 5     NA    NA    NA    NA    NA    NA
## 6     NA    NA     1    NA    NA    NA
## 7      2    NA    NA    NA    NA    NA
## 8      2    NA    NA     1    NA    NA
## 9      1    NA     1    NA    NA    NA
## 10    NA    NA    NA    NA    NA    NA
```

En este caso hay PUNTOS muestreados dentro de PREDIOS, con 2 REPETICIONES. Se deben respetar los nombres de los puntos para cada predio y repeticion, y se puede armar un ID unico que identifique a cada punto. Luego, los datos de Cada especie se ponen en cada columna. De esta manera todas las especies estan representadas con deteccion/no deteccion en todos los puntos y repeticiones.

Chequeo los datos.

```
str(data)
```

```
## 'data.frame':    72 obs. of  18 variables:
##  $ PREDIO: Factor w/ 3 levels "CZ","TW","ZU": 1 1 1 1 1 1 1 1 1 1 ...
##  $ PUNTO : Factor w/ 12 levels "B1","B2","B3",..: 1 1 2 2 3 3 4 4 5 5 ...
##  $ ID    : Factor w/ 36 levels "CZB1","CZB2",..: 1 1 2 2 3 3 4 4 5 5 ...
##  $ ID.REP: Factor w/ 72 levels "CZB11","CZB12",..: 1 2 3 4 5 6 7 8 9 10 ...
##  $ REP   : int  1 2 1 2 1 2 1 2 1 2 ...
##  $ COLME : int  3 NA NA NA NA 1 NA NA NA NA ...
##  $ PSELO : int  NA NA 2 NA NA NA NA NA NA NA ...
##  $ TYRSA : int  NA 1 1 NA NA NA 1 1 2 NA ...
##  $ PYRRU : int  NA NA NA NA NA 2 NA NA NA NA ...
##  $ MIMPA : int  NA NA NA NA NA NA NA NA NA NA ...
##  $ LEILO : int  NA NA NA NA 1 NA NA NA 2 NA ...
##  $ SICLU : int  3 8 1 7 2 NA 4 1 1 NA ...
##  $ ZONCA : int  1 1 1 NA NA NA 2 2 1 NA ...
##  $ RHOFR : int  NA NA 1 NA NA NA NA NA NA NA ...
##  $ SPIBA : int  NA NA NA 3 NA 1 NA NA 1 NA ...
##  $ PASDO : int  NA NA NA 1 NA NA NA 1 NA NA ...
##  $ ZENAU : int  NA NA NA NA NA NA NA NA NA NA ...
##  $ PATMA : int  NA NA NA NA NA NA NA NA NA NA ...
```

Para trabajar con Occupancy tengo que convertir los NA en ceros. Existen casos donde el punto o repeticion no fue muestreado, y alli si hay que dejar un “NA”, pero en los casos donde hubo muestreo y no hubo registro, necesito que haya un CERO.

```
# remplazar NA por cero
data[is.na(data)] <- 0
head(data)
```

```
##   PREDIO PUNTO   ID ID.REP REP COLME PSELO TYRSA PYRRU MIMPA LEILO SICLU
## 1     CZ    B1 CZB1  CZB11   1     3     0     0     0     0     0     3
## 2     CZ    B1 CZB1  CZB12   2     0     0     1     0     0     0     8
## 3     CZ    B2 CZB2  CZB21   1     0     2     1     0     0     0     1
## 4     CZ    B2 CZB2  CZB22   2     0     0     0     0     0     0     7
## 5     CZ    B3 CZB3  CZB31   1     0     0     0     0     0     1     2
## 6     CZ    B3 CZB3  CZB32   2     1     0     0     2     0     0     0
##   ZONCA RHOFR SPIBA PASDO ZENAU PATMA
## 1     1     0     0     0     0     0
## 2     1     0     0     0     0     0
## 3     1     1     0     0     0     0
## 4     0     0     3     1     0     0
## 5     0     0     0     0     0     0
## 6     0     0     1     0     0     0
```

Ahora vamos a reacomodar los datos para obtener los datos de forma longitudinal. Los datos pueden ser ingresados desde el campo de esta manera, vamos a ver un ejemplo luego con el archivo “datos_LONG.csv”. Pero ahora queremos llegar a esta forma:

```
datasample<-read.csv('datOS_LONG_COMPLETOS.csv', sep = ";", header = T)
head(data, 10)
```

```
##    PREDIO PUNTO   ID ID.REP REP COLME PSELO TYRSA PYRRU MIMPA LEILO SICLU
## 1      CZ    B1 CZB1  CZB11   1     3     0     0     0     0     0     3
## 2      CZ    B1 CZB1  CZB12   2     0     0     1     0     0     0     8
## 3      CZ    B2 CZB2  CZB21   1     0     2     1     0     0     0     1
## 4      CZ    B2 CZB2  CZB22   2     0     0     0     0     0     0     7
## 5      CZ    B3 CZB3  CZB31   1     0     0     0     0     0     1     2
## 6      CZ    B3 CZB3  CZB32   2     1     0     0     2     0     0     0
## 7      CZ    B4 CZB4  CZB41   1     0     0     1     0     0     0     4
## 8      CZ    B4 CZB4  CZB42   2     0     0     1     0     0     0     1
## 9      CZ    B5 CZB5  CZB51   1     0     0     2     0     0     2     1
## 10     CZ    B5 CZB5  CZB52   2     0     0     0     0     0     0     0
##    ZONCA RHOFR SPIBA PASDO ZENAU PATMA
## 1      1     0     0     0     0     0
## 2      1     0     0     0     0     0
## 3      1     1     0     0     0     0
## 4      0     0     3     1     0     0
## 5      0     0     0     0     0     0
## 6      0     0     1     0     0     0
## 7      2     0     0     0     0     0
## 8      2     0     0     1     0     0
## 9      1     0     1     0     0     0
## 10     0     0     0     0     0     0
```

Para acomodar los datos vamos a necesitar algunos paquetes

```
#cargo los paquetes que voy a usar
library(reshape)
library(plyr)
```

```
##
## Attaching package: 'plyr'
```

```
## The following objects are masked from 'package:reshape':
##
##     rename, round_any
```

Ahora si, comenzamos.

Extraemos los nombres de las especies. Modificar segun las columnas de su propio archivo! En este caso, los nombres estan en las lineas 6 a 18

```
# extraigo el nombre de las especies
especies.names<-as.vector(names(data[6:18]))
especies.names
```

```
##  [1] "COLME" "PSELO" "TYRSA" "PYRRU" "MIMPA" "LEILO" "SICLU" "ZONCA"
##  [9] "RHOFR" "SPIBA" "PASDO" "ZENAU" "PATMA"
```

Voy a necesitar que TODAS las especies esten en TODOS los puntos. Asique necesito repetir el nombre de cada especie N cantidad de veces, segun el numero de registros (numero de sitios x repeticiones).

```
#armar el vector del nombre de las especies para cada uno de los 72 registros (36 sitios x 2
 repeticiones) y lo ordeno alfabeticamente
sp.names<-sort(factor(rep(especies.names,72)))
class(sp.names)
```

```
## [1] "factor"
```

Por otro lado trabajo con los registros, que de nuevo, depende de las columnas de las sp.

```
# extraigo los datos de registros y los pongo en una sola fila de registros
sp.obs<-data[6:18]

# los ordeno alfabeticamente de modo q cada especie aparezca junta
sp.obs<-sp.obs[ , order(names(sp.obs))]
head(sp.obs)
```

```
##   COLME LEILO MIMPA PASDO PATMA PSELO PYRRU RHOFR SICLU SPIBA TYRSA ZENAU
## 1     3     0     0     0     0     0     0     0     3     0     0     0
## 2     0     0     0     0     0     0     0     0     8     0     1     0
## 3     0     0     0     0     0     2     0     1     1     0     1     0
## 4     0     0     0     1     0     0     0     0     7     3     0     0
## 5     0     1     0     0     0     0     0     0     2     0     0     0
## 6     1     0     0     0     0     0     2     0     0     1     0     0
##   ZONCA
## 1     1
## 2     1
## 3     1
## 4     0
## 5     0
## 6     0
```

```
# asi paso los datos de columna a un solo vector con "unlist"
especies.obs<-unlist(sp.obs,use.names = FALSE)
class(especies.obs)
```

```
## [1] "numeric"
```

Chequeo que el numero de registros x en numero de especies este bien

```
#72 registros * 13 especies = 936 registros para cehquear q esta bien
nreg<-length(especies.obs)
nreg
```

```
## [1] 936
```

Ahora puedo unir los datos de las especies con sus registros

```
# unir las especies con sus registros
sp<-data.frame(sp.names, especies.obs)
```

Luego recopilo la informacion de cada sitio y lo repito para cada especie.

```
# colectar informacion de cada punto
data.info<-data[1:72,1:5]
head(data.info)
```

```
##   PREDIO PUNTO   ID ID.REP REP
## 1     CZ    B1 CZB1  CZB11   1
## 2     CZ    B1 CZB1  CZB12   2
## 3     CZ    B2 CZB2  CZB21   1
## 4     CZ    B2 CZB2  CZB22   2
## 5     CZ    B3 CZB3  CZB31   1
## 6     CZ    B3 CZB3  CZB32   2
```

```
# repetirla para cada una de las 13 especcies
rep.data.info<-do.call("rbind", replicate(13, data.info, simplify = FALSE))
```

Ahora puedo unir la informacion de las especies por sitio, con los sitios

```
#armar la planilla longitudinal con los registros para cada especie x sitio
data.aves.long<-data.frame(rep.data.info, sp)
str(data.aves.long)
```

```
## 'data.frame':    936 obs. of  7 variables:
##  $ PREDIO      : Factor w/ 3 levels "CZ","TW","ZU": 1 1 1 1 1 1 1 1 1 1 ...
##  $ PUNTO       : Factor w/ 12 levels "B1","B2","B3",..: 1 1 2 2 3 3 4 4 5 5 ...
##  $ ID          : Factor w/ 36 levels "CZB1","CZB2",..: 1 1 2 2 3 3 4 4 5 5 ...
##  $ ID.REP      : Factor w/ 72 levels "CZB11","CZB12",..: 1 2 3 4 5 6 7 8 9 10 ...
##  $ REP         : int  1 2 1 2 1 2 1 2 1 2 ...
##  $ sp.names    : Factor w/ 13 levels "COLME","LEILO",..: 1 1 1 1 1 1 1 1 1 1 ...
##  $ especies.obs: num  3 0 0 0 0 1 0 0 0 0 ...
```

```
#exportar tabla
write.csv(data.aves.long, file="data.aves.long.csv")
```

Ahora ya estan los datos completos organizados de manera longitudinal. Siempre tener cuidado si hay algun punto o sitio que no fue muestreado. En ese caso poner NA en lugar de cero.

2) Formato longitudinal incompleto

Otra opcion cuando se ingresan los datos, es ingresar por cada punto y repeticion, las especies observadas.

```
data2<-read.csv("datos_LONG_INCOMPLETOS.csv",header = T, sep = ";")
head(data2, 10)
```

```
##      site point                         species n.ind
## 1  A1.AFS     1                            <NA>     0
## 2  A1.AFS     2                            <NA>     0
## 3  A1.AFS     3             Leptotila verreauxi     1
## 4  A1.AFS     4            Patagioenas picazuro     1
## 5    A1.F     1           Chaetura meridionalis     1
## 6    A1.F     1 Hemitriccus margaritaceiventer     1
## 7    A1.F     1           Turdus amaurochalinus     1
## 8    A1.F     1              Zenaida auriculata     1
## 9    A1.F     2                    Taraba major     1
## 10   A1.F     2         Tarphonomus certhioides     1
```

En este caso es fundamental que, si en algun sitio no hubo ninguna especie, haya al menos un “NA”.

Algunas observaciones

Para analisis de ocupacion de una sola especie, podemos seleccionar un subset de especies

```
data.LEILO <- subset(data.aves.long,data.aves.long$sp.names == "LEILO")
head(data.LEILO)
```

```
##    PREDIO PUNTO   ID ID.REP REP sp.names especies.obs
## 73     CZ    B1 CZB1  CZB11   1    LEILO            0
## 74     CZ    B1 CZB1  CZB12   2    LEILO            0
## 75     CZ    B2 CZB2  CZB21   1    LEILO            0
## 76     CZ    B2 CZB2  CZB22   2    LEILO            0
## 77     CZ    B3 CZB3  CZB31   1    LEILO            1
## 78     CZ    B3 CZB3  CZB32   2    LEILO            0
```

CONSTRUCCION DE LA MATRIZ Y

Como construir la matriz “y” para analizar con Occupancy

Reordeno los datos con el paquete “reshape”. Los datos de deteccion/no deteccion son reordenados en una matriz de N dimensiones,

En el ejemplo que estabmos trabajando tenemos 4 dimensiones: El predio l es la 1er dimension, luego el punto i, la repeticion j y la especie k

```
data.melt=melt(data.aves.long,id.var=c("PREDIO", "PUNTO", "REP", "sp.names"), measure.var="es
pecies.obs")
y1=cast(data.melt, PREDIO ~ PUNTO ~ REP ~ sp.names)
# Verifico las dimensiones
dim(y1)
```

```
## [1]  3 12  2 13
```

Es importante tener en cuenta que cuando se reordenan los datos, se reordenan por orden alfabetico. Igual SIEMPRE hay que chequear que asi sea, ya que luego, para interpretar los datos o agregar una matriz de covariables, tenemos q saber en que orden se agregaron.

Para chequear puedo mirar los datos ingresados originalmente a ver si coincide con la matriz construida.

```
#chequeo con datos originales
y1[,,,1]
```

```
## , , REP = 1
##
##       PUNTO
## PREDIO B1 B2 B3 B4 B5 B6 C1 C2 C3 C4 C5 C6
##     CZ  3  0  0  0  0  0  0  0  1  0  0  0
##     TW  0  0  0  0  0  0  0  0  0  0  0  0
##     ZU  0  0  0  0  0  0  0  0  0  0  0  0
##
## , , REP = 2
##
##       PUNTO
## PREDIO B1 B2 B3 B4 B5 B6 C1 C2 C3 C4 C5 C6
##     CZ  0  0  1  0  0  0  1  1  2  0  0  0
##     TW  0  0  0  0  0  0  0  0  0  0  0  0
##     ZU  0  0  0  0  0  0  0  0  0  0  0  0
```

A veces cuando se arma la matriz hay datos faltantes y se rellenan con “NA”

```
# si hay algun NA le pone cero (esto no hay que hacerlo siempre)
y1[is.na(y1)] <- 0

# Si trabajo con occupancy paso todos los numeros a 0 o 1
y<-aaply(y1,1,function(x) {x[x>1]<-1;x})
dim(y)
```

```
## [1]  3 12  2 13
```

Miscelaneas

```
# Nombres de cada dimension
lista.predios<-as.vector(names(y[,1,1,1]))
lista.especies<-as.vector(names(y[1,1,1,]))
lista.sitios<-as.vector(names(y[1,,1,1]))
lista.reps<-as.vector(names(y[1,1,,1]))

# Dimensiones de cada elemento
npredios<-length(lista.predios)
nspec<-length(lista.especies)
nsites<-length(lista.sitios)
nreps<-length(lista.reps)

#si quiero reordenar las dimensiones
#predio, punto, rep, sp
#sp, predio, punto, rep
y2<-aperm(y,c(4,1,2,3))
y<-y2
```