Análisis Multivariado Aplicado a Ciencias Ambientales usando R

Actividad 1

Prof. Edlin Guerra Castro

2025-04-06

Trabajando con R

El objetivo principal de este laboratorio es introducirlos a la exploración de datos y estadística descriptiva con R y RStudio, las herramientas computacionales que utilizaremos a lo largo del semestre para aprender a aplicar los conceptos más importantes de *Estadística Aplicada I*, pero en especial, para aprender a procesar y analizar datos reales.

El programa R, sus versiones actualizadas y todos los paquetes con funciones, asi como otra información relevante se encuentre en los repositorios de R conocidos como **CRAN** (Comprehensive R Archive Network). Los distintos servidores distribuidos en todo el mundo, conforman el CRAN y son conocidos como los **CRAN mirrors** (de espejo). Para descargar los paquetes requieres antes escoger un **CRAN mirror**, y la función que te permite escogerlo de una lista que aparece en la consola es `chooseCRANmirror`. Alternativamente, se pueden descargar paquetes desde otros repositorios, uno muy popular y que estaremos usando en esta asignatura es GitHub.

RStudio es un entorno de desarrollo integrado (IDE) para **R**. Incluye una consola, editor de comandos y líneas de programación que admite la ejecución directa de código, así como herramientas para graficar, documentar, registrar el historial de comandos ejecutados, acceder a archivos, y muchas cosas más desde la gestión de un espacio de trabajo. **RStudio** hace que el trabajar con **R** sea más poderoso, y a su vez simple. Para que tengan una idea, esta guía se escribió desde **RStudio**.

Las funciones están organizadas en paquetes. El paquete denominado `base` constituye el núcleo de **R** y contiene las funciones básicas del lenguaje. Otro paquete muy importante es `stats` e incluye las funciones estadísticas más importantes y básicas de **R**. Ambos ya vienen preinstalados en **R**. Existen muchos paquetes, a medida que se requiera el uso de alguno específico se irá indicando para que lo descarguen e instalen. Por ahora les adelanto el uso de un set de paquetes agrupados en una familia de paquetes muy usados para ordenar, limpiar,

modelar, reproducir, comunicar y graficar datos; este grupo de paquetes se les denomina tidyverse. Para instalarlos pueden escribir en la consola:

```
#Para instalar ggplot2 (realizar gráficos de alta calidad)
install.packages("ggplot2")

#Para depurar y reordenar bases de datos, instala: tidyr
install.packages("tidyr")

#Para administrar bases de datos: usa dplyr
install.packages("dplyr")

#Para importar datos desde Excel: readxl
install.packages("readxl")

#Para análisis en ecología de comunidades usa vegan
install.packages("vegan")

#para análisis de diversidad usa iNEXT
install.packages("iNEXT")

#para estimar simetría y curtosis
install.packages("moments")

#Para instalar todos los paquetes del Tidyverse:
install.packages("tidyverse")
```

Parte 1: Objetos, funciones y paquetes

R es un lenguaje orientado a objetos, lo que significa que las variables, datos, funciones, resultados, etc., se guardan en la memoria activa del computador en forma de objetos con un nombre específico dado por el usuario en cada sesión. Los objetos se manipulan mediante funciones (que, a su vez, pueden ser tratados como objetos) y operadores. La ventana de la consola es donde se escriben los comandos, después de un indicador o prompt `>` que notifica cuando R está listo para recibir la siguiente instrucción. La tecla `esc` aborta la tentativa de esa línea de comando y da la señal para que aparezca un nuevo prompt. Dos prompts `> >` seguidos invalida esa línea de comando. Si aparece un signo de `+` es que la linea de comando está incompleta y requiere ser completada ante de devolver un resultado. Si aparece un mensaje de *Error* significa que él comando o instrucción no tuvo efecto. Si aparece un *Warning* significa que **R** efectuó la instrucción anterior, pero tuvo algún obstáculo mismo que es descrito inmediatamente. Con las flechas del arriba y abajo del teclado, aparece la linea de comando inmediata anterior y es una manera de no re-escribir dichas líneas cada vez. El signo

de número **#** indica un comentario que no será tomado en cuenta hasta que aparezca un nuevo prompt.

Para poder ver los objetos que se encuentran en una sesión activa de **R**, se puede escribir la función de enlistar `ls`, o si estás en **R-Studio**, verifica directamente la pestaña **Environment** en el panel superior derecho.

```
ls()
```

El nombre de un objeto se asigna con el operador '<-', '->' o '=', y puede estar hecho de letras, números y puntuación. Nota: Usar el mismo nombre para dos objetos distintos implica perder la asignación del primer objeto

```
y.y <- 10 * 10 #operación matemática

fecha <- date() #función para generar información


xx <- 4 #número guardado como xx
xx
xx <- "xx ya no es el mismo" # texto guerdado como xx, se reescribió sobre 4
xx
```

R es sensible a mayúsculas, pero no a los espacios:

```
compar
Compar
sum    (3+2)
sum(3+2)
```

En **R** se usan tres tipos de lementos: números (*numeric*), letras (*character*, siempre entre comillas), lógicos (*logical*). Estos elementos son usados para generar objetos. Los objetos pueden clasificarse como:

A) *Vector*: una columna o una fila de elementos, que pueden ser numéricos, de caracteres de texto, de operadores lógicos, etc. Cuando se trata de una variable categórica, el vector puede ser tratado como un factor, y los niveles del factor corresponden a las categorías de dicha variable. Un vector se crea con la funcion `c`, deguido de paréntesis `()` que incluyen todos los elementos del vector separados por coma.

```r
vect1 <- c(2,4,6,3,7,8,9,2)
vect1

vect2 <-c("esp", "ing", "por")
vect2

#Puedes preguntar si un vector tiene elementos de un tipo en particular:
is.numeric(vect1)
is.numeric(vect2)
is.character(vect1)
is.character(vect2)

#Qué hace esto:
vect1[5]
vect2[2]
```

Una de las grandes fortalezas de **R** es que permite el acceso a los elementos de un objeto a través de una selección de subconjuntos de éstos. El *sub-setting* es una manera eficiente y flexible de acceder selectivamente a los elementos de un objeto, y se hace mediante el uso de corchetes `[]`.

B) *Matrix*: es un arreglo bidimensional de columnas y renglones, sobre el cual se pueden aplicar operaciones algebraicas. Cada elemento de una matriz puede ser accedido con `[,]`, delante de la coma iría el número de la o las filas, luego de la coma, el número de la o las columnas. La combinación específica de una fila y columna lleva al valor de la celda.

```r
#creando una matriz combinando tres vectores
matr1 <- rbind(c(1,2,3),c(4,5,6),c(7,8,9))
matr1
is.factor(matr1)
is.numeric(matr1)

#creando una matriz con una función

matr2 <- matrix(data = seq(1:9), nrow = 3, ncol = 3, byrow = TRUE)

#Note la diferencia entre matr2 y matr3 si cambiamos el argumento byrow a FALSE
matr3 <- matrix(data = seq(1:9), nrow = 3, ncol = 3, byrow = FALSE)

#Selección de segunda y tercera columna de matr2
matr2[,c(2,3)]
```

```r
#Selección de primera fila de matr2
matr2[1,]

#Selección el valor de la primera fila y segunda columna de matr2
matr2[1,2]
```

C) *Array*: es un arreglo de dimensiones k>2.

```r
n <- 3
k <- 2
j <- 4
samp <- array(dim = c(n,k,j))
samp
```

Los vectores, matrices y arreglos solo pueden tener elementos del mismo tipo (e.g. numéricos, lógicos, letras)

D) *Dataframe*: es una tabla compuesta de uno o más vectores de la misma longitud, pero con elementos que pueden ser de diferentes tipos. Es el formato ideal para bases de datos, ya que las variables suelen ser de diferente naturaleza (i.e. continuas, nominales, etc.). Se puede acceder a ellas usando la sintaxis de matrices, pero también son el signo `$` para identificar a la columna por su nombre.

```r
iris
data(iris)

#haga click sobre <promise> de iris en su ambiente, luego explore visualmente.
# ¿Qué es iris?

dim(iris)    #Pide las dimensiones de la tabla iris
names(iris)  #Pide los nombres de las columnas en iris
iris[,"Species"] # Selecciona la columna por su nombre
iris[,5] # Selecciona la columna por su número de columna
iris$Species     # Selecciona la columna por su asignación en la tabla 'iris'
```

E) *List*: Este objeto puede ser visto como un estante, ya que agrupa ordenadamente objetos de diferente tipo (e.g. vectores, arreglos, tablas, otras listas, etc). Se usa mucho para devolver los resultados de una función que se encuentran en la forma de una colección de objetos:

```r
mi_lista <- vector(mode = "list")

mi_lista[[1]] <- iris
mi_lista[[2]] <- y.y
mi_lista[[3]] <- fecha

mi_lista
```

Parte 2: Estadística descriptiva

Alternativamente, puedes usar del menú superior la opción *Tools/install packages...*, se desplegará una ventana para que escribas el nombre del paquete a instalar. Los paquetes se instalan una sola vez, siempre que estes en el mismo computador. Para usarlos debes incluirlos en tu sesión de trabajo cada vez que se inicia la sesión. Esto se logra con la función `library`:

```r
library("tidyverse")
```

Hagamos un análisis exploratorio a los datos de los pinguinos de la Antártida del género *Pygoscelis*. Lo primero que debe hacer es instalar el paquete de datos `palmerpenguins` y cárguelo en su sesión. Luego haga lo siguiente:

1. Busque en la pestaña *Help* qué es *palmerpenguin*.
2. Identifique la base de datos `penguins` y cárguela en su ambiente global con `data(penguins)`.
3. Identifique cuántas variables hay, cuál es la naturaleza de cada una de ellas (tipo de variable, escala), así como cuáles pueden ser consideradas causales y cuáles respuesta.
4. Efectue un gráfico de dispersión entre las variables `body_mass_g` y `flipper_lemgth_mm`. Mida el grado de asociación ¿cómo lo haría? Una forma de hacer estas cosas es graficando la asociación, y estimando la correlación (método que se desarrollará en otro laboratorio, pero acá es pide con fines demostrativos)

```r
plot(penguins$body_mass_g, penguins$flipper_length_mm)

#¿qué hace cor()?
cor(penguins$body_mass_g, penguins$flipper_length_mm)

#Si no obtuvo resultado, trate de resulverlo con el argumento "use"
```

Usemos el paquete `ggplot2` para mejorar el gráfico:

```r
pp <- ggplot(data = penguins, aes(x = flipper_length_mm,
                                  y = body_mass_g,
                                  colour = species))+
      geom_point()

pp

#Mejoremos con capas
pp +  theme_bw()+
      xlab("Largo de aleta (mm)")+
      ylab("Masa corporal (g)")+
      scale_y_continuous(breaks = seq(2600,6400,400))+
      scale_x_continuous(breaks = seq(170,240,5))

#¿cuál gráfico le gustó más?
```

Llegados a este punto, vamos hacer algo de estadística descriptiva. Calculen promedio, varianza, desviación estandar, valor mínimo y máximo, cuartiles, simetría y curtosis a la variable masa corporal. Usen para ello las funciones recomendadas y responda las siguientes preguntas:

```r
# Para facilitar cálculos, vamos a remover los datos sin registro
# (identificados con NA), usando el siguiente código:

penguins2 <- penguins |>
  na.omit()
```

PREGUNTAS

1. Copia el comando que sigue. ¿Qué se calculó?

```r
xx <- penguins2$body_mass_g
sum(xx, na.rm = T)/length(xx)
```

2. Busca y aplica una función que ejecute la linea de comando anterior. PISTA: escribe `?mean` en la cónsola y enter para ampliar tu búsqueda.
3. Calcula la mediana de la masa corporal usando la función correspondiente.
4. Calcula la varianza y desviación estándar de la masa corporal usando el comando `var()`
5. ¿Cuál es la diferencia entre estas dos fórmulas? ¿Representan lo mismo?

```r
sum((xx-mean(xx))^2)/length(xx)

sum((xx-mean(xx))^2)/(length(xx)-1)
```

6. Con base en el valor de la varianza y usando operadores aritméticos, calcula la desviación estándar de la masa corporal. Confirma tu resultado usando la función `sd()`.
7. Explore el rango de la masa corporal identificando mínimos y máximos con la función `min()` y `max()`, respectivamente.
8. Ahora estime los cuartiles de la masa corporal con la función `quantile()`
9. Describa la forma de la distribución de la masa corporal usando la simetría y curtosis con las funciones `skewness()` y `kurtosis()`.
10. Todas estas estimaciones ignoran las posibles diferencias en la masa corporal entre las especies. ¿Qué le dice este gráfico?

```r
boxplot(body_mass_g~species, data = penguins2)
```

11. Calcule estos estimadores para cada especie usando el paquete **dplyr** y sus funciones `group_by()` y `summarize()`. Estas líneas de comando lo ayudarán (interprete los resultados):

```r
library(dplyr)

penguins |>
  group_by(species) |>
  summarise(media = mean(body_mass_g, na.rm =T),
            desviacion = sd(body_mass_g, na.rm =T),
            simetria = skewness(body_mass_g, na.rm =T),
            curtosis = kurtosis(body_mass_g, na.rm =T))
```

12. Usando como guía el libro digital R Graphics Cookbook, genere: (i) una distribución de frecuencias con histograma, (ii) una distribución de frecuencias basada en densidad, (iii) un diagrama de cajas que incluya promedio. En los tres casos la masa corporal debe distinguirse por especie.

Parte 3: Análisis univariados a datos de estructura multivariada

Caso hipotético: Se requiere identificar si existen cambios en la composición de especies bentónicas en una línea de costa en cuyo extremo oriental se están vertiendo aguas residuales de una planta de tratamiento de un gran hotel. Estas aguas, aun siendo tratadas, siguen siendo ricas en materia orgánica, y se dispersan según el patrón de corrientes en sentido Este-Oeste. En principio, la línea de costa es bastante homogénea en características ambientales, siendo la contaminación orgánica la única variable que genera gradiente ambiental. Se presume que el aporte orgánico puede estar modificando la estructura trófica del ecosistema marino-costero según el gradiente de dispersión. Para identificar si esto ocurre, se decidió evaluar la composición de especies y las abundancias de los poliquetos de la macrofauna, ya que es bien conocido que son excelentes bioindicadores de contaminación orgánica. Se tomaron 10 muestras de sedimento en seis localidades a lo largo de la bahía, desde el punto de vertido hasta 30 km alejados de la fuente de contaminación. Las abundancias por especies fueron agrupadas por localidad y se encuentran en el archivo "gradiente.csv". La actividad consiste en identificar el gradiente biológico. NOTA: Sí existe gradiente biológico, fallar en detectarlo implica librar al hotel de responsabilidades ambientales y administrativas ante el gobierno.

1. Importe la matriz de datos "gradiente.csv" y nómbrela *gradiente*. Examine el objeto y responda:

a) ¿Qué tipo de objeto se creó? Si usaste la función `read_csv` del paquete `readr`, asegura de transformar el objeto resultante a *data frame* con la función `as.data.frame`.
b) ¿Cuáles son las dimensiones del objeto?
c) ¿qué representan las filas y qué representan las columnas?
d) ¿Con base en las dimensiones, se puede inferir cuántas especies hay en la matriz y en cada localidad?
e) ¿Son similares las magnitudes de abundancia de las especies en cada localidad?
f) Considerando que esta es una matriz pequeña ¿puede apreciar algún patrón biológico viendo solo la tabla?

2. Genere una tabla de nombre "uni" que describa las propiedades UNIVARIADAS bióticas de las localidades, incluya: la riqueza de especies, la abundancia de individuos y el índice de diversidad de Simpson. Le recomiendo buscar ayuda sobre las funciones `specnumber`, `diversity`. Acá un **script** que le ayudará a avanzar rápido. Nota: ninguna de estas funciones filtran texto

```
uni <-
  data.frame(
    "S" = specnumber(gradiente),
    "N" = apply(gradiente, 1, sum),
    "Simpson" = diversity(gradiente, "simpson")
  )
```

a) ¿Obtuvo un error? ¿qué cree que pudo ocurrir?

b) Seleccione sólo las columnas con información de las abundancias de las especies y llámelas "dat".

c) Repita el **script** para generar la tabla de descriptores univariados.

d) ¿volvió a obtener un problema? ¿qué cree que pudo ocurrir ahora?

e) ¿qué significan los NA? ¿cómo convertimos los NA en números? ¿Qué números le asignamos?

f) Calculados los descriptores univariados ¿puede apreciar algún patrón biológico viendo solo la tabla? Explique el patrón. Si considera necesario realizar algún gráfico, hágalo. El siguiente **script** puede ser útil, pero no es la única forma.

```
dat.s <- data.frame(stack(dat), loc = row.names(dat))
qplot(
  data = dat.s,
  x = loc,
  y = values,
  group = ind,
  col = ind,
  shape = ind,
  geom = c("point", "line"),
  xlab = "Localidad",
  ylab = "Abundancia"
)
```

g) Aparentemente no hay un patrón biológico según el gradiente, sin embargo, la ausencia de patrón puede ser producto del reduccionismo de los descriptores univariados. Por ello, usaremos métodos multivariados para apreciar mejor la informacion de la matriz.

Parte 4: Análisis en modo Q

1. Hemos identificado que nuestra matriz tiene 6 muestras (localidades) y 10 especies. Además, no todas las especies están en todas las muestras, hay especies con abundancias muy elevadas y otras muy bajas. ¿qué representa esto para iniciar un análisis multivariado? Exploremos esto usando varios métodos. Sugiero leer el material de ayuda `vegdist` del paquete `vegan`, tipeando `??vegdist` en la cónsola. Para esta práctica compararemos el desempeño del índice Bray-Curtis y la distancia Euclideana.

a) Iniciemos con las distancias Euclideanas. Use la función `vegdist`, y llame al resultado *euc*.

b) ¿cuántos valores se estimaron teniendo en cuenta que hay 6 localidades?

c) ¿Qué tipo de objeto es euc?

d) ¿cómo se interpretan las unidades medidas?

e) ¿cuáles son las localidades más parecidas? ¿qué puede decir de la relación entre la localidad 1 y 6?

f) ¿Aprecia el gradiente? Si no, genere un dendograma que nos ayude a visualizar el patrón de cambio

```
plot(hclust(euc, "average"))
```

g) Interprete el gráfico. ¿Hay patrón de cambio según el gradiente ambiental?

h) ¿Por qué no se aprecia el patrón? ¿qué puede estar ocurriendo?

i) Usemo ahora Bray-Curtis. Use la función vegdist, y llame al resultado bray1.

j) ¿cuántos valores se estimaron teniendo en cuenta que hay 6 localidades?

k) ¿Qué tipo de objeto es bray1?

l) ¿qué fue lo que se estimaron, similitudes o dismilitudes? ¿cómo se interpretan estas unidades? ¿cuál es su escala? Para obtener las similitudes e interpretarlas con mayor facilidad aplique el siguiente comando

```
1-bray1
```

n) ¿cuáles son las localidades más parecidas? ¿qué puede decir de la similitud entre la localidad 1 y 6? ñ) ¿Aprecia el gradiente? Si no, genere un dendograma que nos ayude a visualizar el patrón de cambio

```
plot(hclust(bray1, "average"))
```

o) Este gráfico le permitirá identificar el par de localidades, o subgrupos de localidades, más semejantes en forma jerárquica y pareada ¿Aprecia el patrón?

p) El patrón aun no se aprecia, probablemente se deba al peso de la especie 5. Aplique transformación raíz cuadrada a las abundancias y llame a esa matriz nueva "dat2". Calcúle nuevamente el índice Bray-Curtis y llámelo bray2.

q) ¿cuáles con las localidades más parecidas? ¿qué puede decir de la similitud entre la localidad 1 y 6? ¿Aprecia el gradiente? Genere un dendograma que nos ayude a visualizar el patrón de cambio

```
plot(hclust(bray2, "average"))
```

r) ¿Ahora sí aprecia el patrón de cambio con el gradiente ambiental?
s) Compare los tres dendogramas
t) Genere un nuevo gráfico multivariado, es un nMDS (escalamiento multidimensional no métrico - tema de la Unidad 5). Por ahora solo importa indicar que la distancia de los puntos proyectados en el nMDS refleja las similitudes expresadas en la matriz. En este sentido ¿aprecia mejor el gradiente de cambio en la composición de especies?

```
mds <- metaMDS(dat2, "bray")
plot(mds, type = "n")
text(mds,
     display = "sites",
     cex = 1,
     col = "blue")
```

2. El enunciado hace referencia a la composición de especies, explícitamente indica que solo importa el cambio en la identidad de las especies, no en sus abundancias. Esto implica que a pesar de la transformación raíz cuadrada, las diferencias en abundancias pueden estar influyendo en el patrón observado. Para satisfacer plenamente la pregunta enunciada, se aplicará un índice que sólo considere la presencia de la especie. Hay más de 25 índices para evaluar matrices de incidencia, uno de los más populares es Jaccard. Aplique este índice con la función `vegdist` y llame al objeto "jac".

a) ¿Qué puede inferir ahora del patrón y el gradiente? ¿cuál es la similitud en composición de especies entre las localidades 1 y 6?

b) ¿Cuál es la similtud en la composición de especies entre las localidades 6 y 5?

c) Genere un dendograma para proyectar la nueva matriz. Interprete el agrupamiento del dendograma

d) Veamos ahora globalmente los cuatro dendogramas generados. Para ello use el siguiente código

```
par(mfrow = c(2, 2), mar = c(2, 4, 2, 2)) # este script indica a R que se
#proyectarán cuatro gráficos en uno solo
plot(hclust(euc, "average"), main = "Dist. Euclideana")
plot(hclust(bray1, "average"), main = "Bray-curtis sin transformación")
plot(hclust(bray2, "average"), main = "Bray-curtis con transformación")
plot(hclust(jac, "average"), main = "Jaccard")
```

e) Existes métodos cuantitativos para comparar matrices. Uno de ellos es el método de Mantel, y se base en correlacionar los rangos de las jerarquias de las similitudes en cada matríz. Por ahora solo interesa conocer la correlación, y no la probabilidad asociada. Como reto para usted, busque en el paquete `vegan` la función `mantel`, y correlacione la matríz de similitud en composición de especies respecto a las matrices Bray-Curtis y la Matriz de distancias Eucliedanas.