

Contents

LIMNOLOGIA CON R

Taller dictado en el marco del **IV Congreso Iberoamericano de Limnología - X Congreso Argentino de Limnología (IV CIL – CAL X)**, 27 y 28 de julio de 2023.

El material se elaboró utilizando el paquete *bookdown* (??), basado en *R Markdown* (?) y *knitr* (?).

Docentes: Natalia Morandeira, María E. Llames, Sofía Carusso, Patricia E. García, Joaquín Cochero, Julieta Capeletti y María Victoria Quiroga.

Coordinadora: María Victoria Quiroga.

0.1 Contenido

Unidad ??: Introducción a R y RStudio. Importación y manejo de datos en R. Gráficos exploratorios básicos.

Unidad ??: Correlaciones y regresiones múltiples.

Unidad ??: Modelos Aditivos Generalizados.

Unidad ??: Análisis multivariados I. PCA, NMDS, PERMANOVA.

Unidad ??: Análisis multivariados II. DCA, RDA, dbRDA.

Unidad ??: Paquete *metrix*. Estimar métricas (individuales y ecológicas) e índices bióticos basados en macroinvertebrados acuáticos para evaluar la calidad del agua.

Unidad ??: Paquete *DiaThor*. Estimación de información ecológica e índices bióticos para diatomeas. Paquete *optimos.prime*. Estimación de óptimo y rango de tolerancia de una especie.

Unidad ??: Buenas prácticas en R y Rstudio

Chapter 1

Introducción a R y RStudio

Sofía Carusso¹

Museo Argentino de Ciencias Naturales Bernardino Rivadavia-CONICET

& María Victoria Quiroga²

Instituto Tecnológico de Chascomús (INTECH, UNSAM-CONICET), Escuela de Bio y Nanotecnologías (UNSAM)

1.1 Instalación

Seguir las indicaciones de la página <https://posit.co/download/rstudio-desktop/> para descargar e instalar **R** y **RStudio**. Es muy importante que lo haga de manera secuencial como se indica, primero **R** y luego **RStudio**.

1.2 RStudio

La primera vez que abrimos RStudio la interfaz nos muestra tres paneles:

- *Panel izquierdo -Consola:* donde corremos el código
- *Panel derecho superior -solapa Entorno:* vemos los datos y funciones que cargamos en la sesión de R.
- *Panel derecho inferior -solapa Files:* directorio de trabajo y archivos dentro de la carpeta.
- *Panel derecho inferior -solapa: Gráficos:* visualización de plots.

¹soficarusso@gmail.com

²mvquirosa@iib.unsam.edu.ar

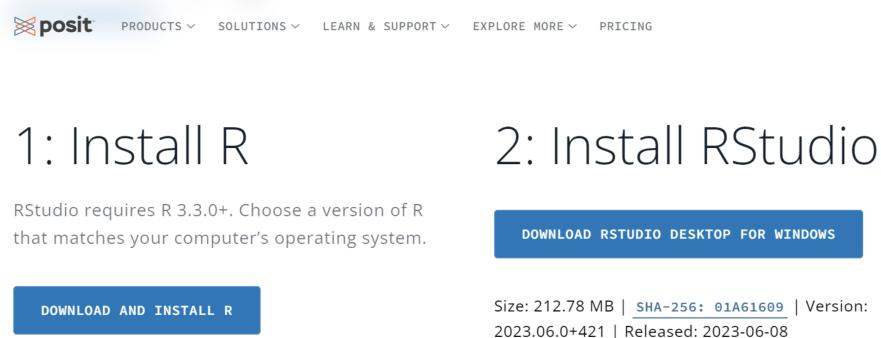


Figure 1.1: Instalación de R y RStudio.

- *Panel derecho inferior -solapa:* **Paquetes:** visualización/carga/actualización de paquetes de R.
- *Panel derecho inferior -solapa:* **Ayuda:** Recuerde que puede buscar ayuda desde esta solapa o tipeado en la consola `?nombre_del_comando`.

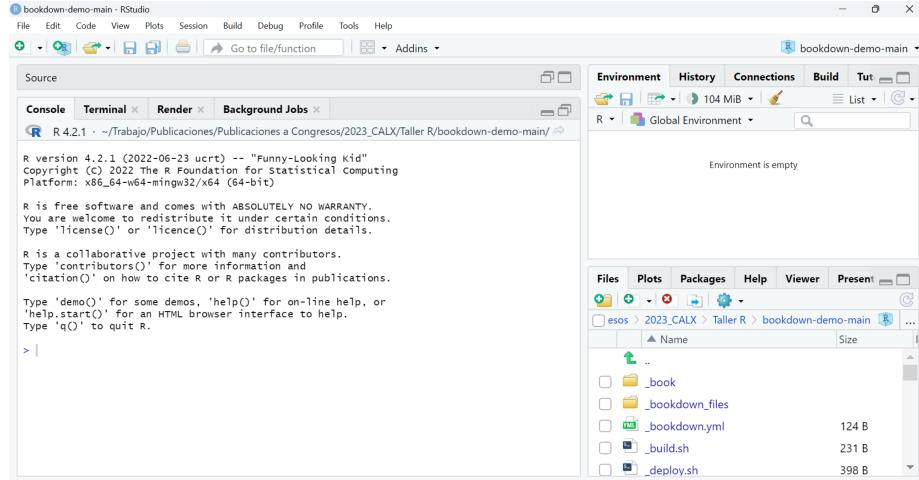


Figure 1.2: Interfaz de RStudio.

1.3 ¡A trabajar!

1.3.1 Crear un Proyecto

1. Click en **File** (*esquina superior izquierda en Figura ??*).

2. Click en **New Project**.
3. Click en **New Directory**.
4. Click en **New Project**.
5. Escribir el nombre de la carpeta, que será el **Directorio de Trabajo** y contendrá el **Proyecto**. Se puede setear la ubicación de la carpeta haciendo click en **Browse**. En la Figura ?? se creó la carpeta LimnologiaR_U01 en el Escritorio.
6. Click en **Create Project**.

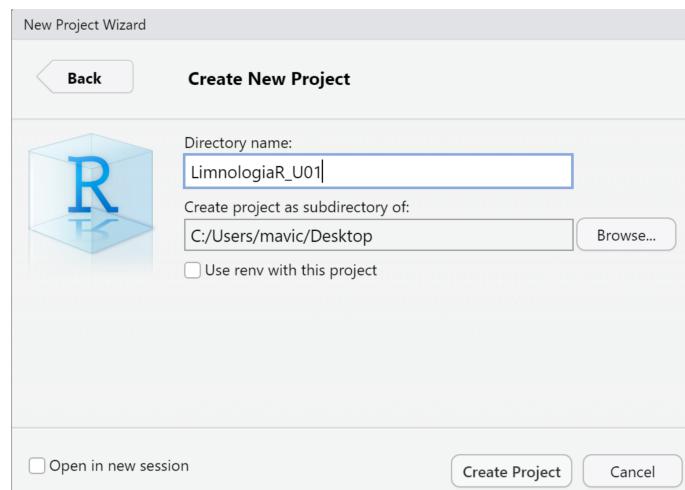


Figure 1.3: Crear un nuevo proyecto para la Unidad 1.

Ver el **Directorio de Trabajo**:

```
getwd()
```

Recomendaciones:

Crear una carpeta para cada unidad, para tener los análisis separados y ordenados.

Generar nombres de carpetas y archivos **sin** espacios, acentos o caracteres especiales.

1.3.2 Crear un R Script

1. Click en **New file**.
 2. Click en **R Script**.
- Se puede utilizar el atajo **Ctrl+Shift+N**.

Recuerde las recomendaciones para nombrar archivos al guardar el script.

El script se va a guardar como archivo **.R** en el **Directorio de Trabajo**.

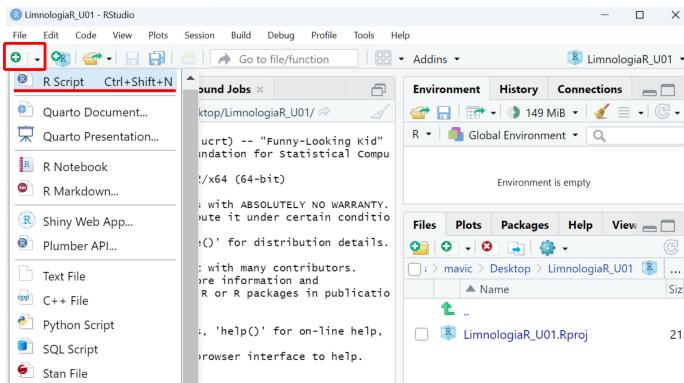


Figure 1.4: Crear un nuevo R Script.

En este **script** se escribe y guarda el código.

Una forma fácil de copiar código es utilizando el botón **Copy to clipboard** que se encuentra en la esquina superior izquierda de los **bloques de código** (Figura ??).



Figure 1.5: Copiar código desde el material del taller.

Se pega o escribe **código** en el **script** y se **guarda** haciendo click como se indica en el *recuadro 1* de la Figura ??.

El código en el **script** se **ejecuta** haciendo click en **Run the current line or selection** (*recuadro 2* en Figura ??). Podemos ejecutar de una línea por vez: nos situamos en una línea y hacemos click; o podemos ejecutar varias líneas juntas: seleccionamos el set de líneas y hacemos click. La última opción *no* se recomienda si es la primera vez que corre el código.

El código se ejecuta en la **consola** Figura ??.**2**.

Se observan los resultados en la **consola** Figura ??.**3**.

Los resultados se pueden copiar de la **consola** y pegar en el **script**. Recuerde marcar los resultados como comentarios (líneas que comienzan con #). *Atajo: seleccionar todas las líneas de resultados y presionar Ctrl+Shift+C.*

R **no** ejecuta lo que se encuentra después de #.

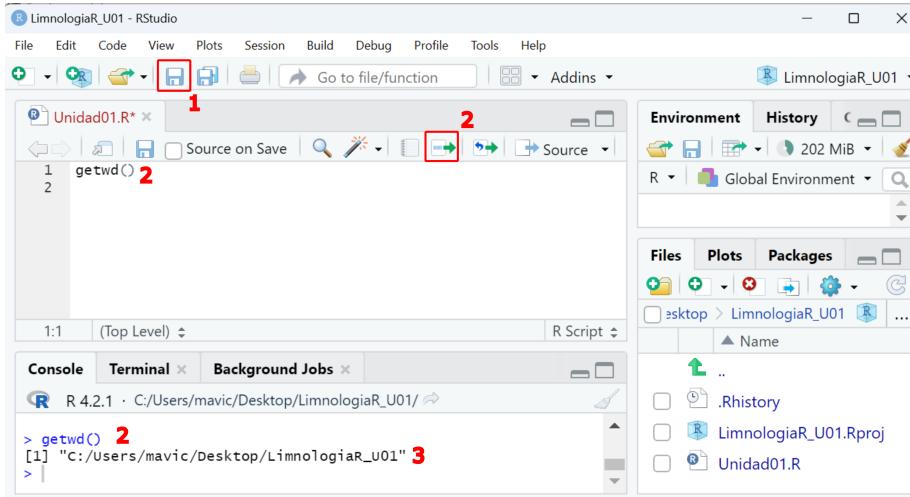


Figure 1.6: Trabajo en la interfaz de RStudio. 1- Guardar Script. 2- Ejecutar código desde Script. 3- Resultados en consola

1.3.3 Instalar y cargar paquetes

Los paquetes se instalan (por única vez) desde la **consola** con `install.packages("nombre")` o desde la interfase de la solapa **Packages**.

Los paquetes se cargan en la sesión con `library("nombre")`.

- **Buenas prácticas!** Conviene poner todas las librerías que se vayan a usar al comienzo.

1.3.4 Importar datos

Hay diversas maneras de leer datos. Podemos leer datos

- `.txt`
 - Conviene guardar la base de datos como txt (Tab delimited); donde las columnas quedan definidas por espacios.
- `.csv` Comma/Separated Values
 - Archivos de texto separados por comas. Se forma una tabla de filas y columnas

- *.xls*

Generalmente se utiliza un comando `read`, por ejemplo `read.csv("Nombre del archivo.csv")`, según el tipo de archivo que se trate (txt, xlsx, etc.)

Una vez que tenemos cargado nuestro conjunto de datos nos puede interesar realizar algunos comandos exploratorios que nos van a dar información acerca de este *data frame* (estructura de datos).

- `summary(objeto)`
 - La salida va a depender de la clase de objeto al cual lo aplicamos.
 - Útil para ver información básica sobre nuestras variables
 - Para *data frames* el summary nos devuelve el valor mínimo, máximo, la mediana y el 1er y 3er cuantil.
 - Si se aplica para salidas de modelos lineales (Ver Unidad 2), nos proporciona información acerca de los residuos, los coeficientes del modelo, el error estándar residual, R^2 , el R ajustado, el estadístico F y el p valor.
- `str(objeto)`
 - Nos devuelve cuántas observaciones tiene el data frame, información acerca de las variables (son numéricas? factores?); como esta estructurado nuestro conjunto de datos.
- `head(objeto)` y `tail(objeto)`
 - Nos da las primeras y últimas filas, respectivamente.
- `class(objeto)`
 - Devuelve qué tipo de objeto es (data frame, matrix, vector, etc).

```
data("iris")
summary(iris)
```

```
##   Sepal.Length   Sepal.Width    Petal.Length   Petal.Width
## Min.   :4.300   Min.   :2.000   Min.   :1.000   Min.   :0.100
## 1st Qu.:5.100   1st Qu.:2.800   1st Qu.:1.600   1st Qu.:0.300
## Median :5.800   Median :3.000   Median :4.350   Median :1.300
## Mean   :5.843   Mean   :3.057   Mean   :3.758   Mean   :1.199
## 3rd Qu.:6.400   3rd Qu.:3.300   3rd Qu.:5.100   3rd Qu.:1.800
## Max.   :7.900   Max.   :4.400   Max.   :6.900   Max.   :2.500
##
##             Species
## setosa      :50
## versicolor:50
```

```

##  virginica :50
##
##
##



str(iris)

## 'data.frame':   150 obs. of  5 variables:
##   $ Sepal.Length: num  5.1 4.9 4.7 4.6 5 5.4 4.6 5 4.4 4.9 ...
##   $ Sepal.Width : num  3.5 3 3.2 3.1 3.6 3.9 3.4 3.4 2.9 3.1 ...
##   $ Petal.Length: num  1.4 1.4 1.3 1.5 1.4 1.7 1.4 1.5 1.4 1.5 ...
##   $ Petal.Width : num  0.2 0.2 0.2 0.2 0.2 0.4 0.3 0.2 0.2 0.1 ...
##   $ Species      : Factor w/ 3 levels "setosa","versicolor",...: 1 1 1 1 1 1 1 1 1 1 ...


head(iris)

##   Sepal.Length Sepal.Width Petal.Length Petal.Width Species
## 1          5.1         3.5          1.4         0.2  setosa
## 2          4.9         3.0          1.4         0.2  setosa
## 3          4.7         3.2          1.3         0.2  setosa
## 4          4.6         3.1          1.5         0.2  setosa
## 5          5.0         3.6          1.4         0.2  setosa
## 6          5.4         3.9          1.7         0.4  setosa


class(iris)

## [1] "data.frame"

```

1.3.5 Visualizar y manipular datos

Qué tipo de variables hay?

Existen variables numéricas (num), factores (que pueden tener varios niveles, factor), char o de carácter. Generalmente R detecta el tipo de variable al cargar los datos. Se puede transformar variables entre sí.

- **Observación** El comando `c()` nos permite concatenar elementos. Siempre que se selecciona más de un elemento se debe concatenar.

Por ejemplo,

```
variable <- c(1,2,1,1,1,2,2,2,1)
class(variable)

## [1] "numeric"

factor <- factor(variable, levels = c(1,2), labels=c("Nivel 1", "Nivel 2"))
```

De esta manera, transformamos una variable numérica en un factor con dos niveles, Nivel 1 y 2. Nos puede ser útil en caso de tener una variable tomada desde campo (o de encuestas) como numérica y queremos que sea categórica.

Podemos acceder a elementos particulares dentro del data frame, ya sea porque nos interesa ver ese elemento individual, para sacarlo del data frame, **para armar filtros** o para realizar ciertos análisis con una parte del data frame. Se trabaja con coordenadas (x,y), donde **x** es la fila e **y** la columna.

Si quiero todos los datos de una fila en particular, por ejemplo la 17 se escribe **iris[17,]** (seleccionamos la fila y las columnas quedan vacías porque queremos verlas todas).

De la misma manera, si queremos ver solamente una columna **iris[,3]**. También podemos llamarlo según el nombre de la misma **iris[, "Species"]**.

Para seleccionar los primeros 10 datos **iris[1:10,]** (o con el comando que ya vimos **Head()**).

Si queremos seleccionar **varias filas** simplemente las concatenamos. Entonces si escribimos **iris[c(1:5), c(2,3)]** Qué seleccionamos en este caso?

Qué pasa si quiero seleccionar una sola variable? Podemos! El signo **\$** indica que se selecciona una columna dentro del data frame que lo precede.

iris\$Species me permite ver toda la columna de la variable Especies. De esta manera, incluso podemos crear variables nuevas, asignandolas de la siguiente manera

```
iris$variable.nueva <- iris$Sepal.Length/iris$Sepal.Width
```

Qué variable nueva acabamos de crear?

- **Ojo!** Si no le asignamos el data frame a la variable nueva con el signo **\$** (**iris\$variablenueva**) el R la va a crear por fuera de nuestro data frame como un vector de los valores.

Subsets

Como lo indica el nombre, nos quedamos con una porción que seleccionemos del data frame, y sobre el cual podremos operar de manera independiente, sin que el data frame original se vea afectado.

En el ejemplo con iris, podemos crear un subset que contenga solamente a la especie iris setosa.

```
setosa <- subset(iris, iris$Species=="setosa")
```

Para aquellos que se sientan más cómodos, la librería **dplyr** permite hacer que las selecciones y filtros sea más fácil. Suma mucho a la facilidad del trabajo aprender a usarla!

```
library(dplyr)

setosa.dplyr <- iris %>%
  filter (Species %in% "setosa")
```

Hay muchas formas de hacer subsets! Es cuestión de usar aquella con la que se sientan más cómodos.

1.3.6 Gráficos exploratorios básicos

La visualización de datos es un punto clave dentro de cualquier análisis. Los gráficos son útiles para explorar datos, interpretarlos, identificar patrones, detectar outliers y constituyen una de las mejores maneras de comunicar los resultados.

¿Qué tipos de gráficos existen?

Hay una enorme diversidad, e incluso se pueden utilizar combinaciones de ellos para aumentar la información comunicada.

- **Gráfico de barras**
 - Muestra una barra para cada categoría de una variable categórica.
El alto muestra el valor observado para cada categoría (Fig 1.A)
- **Gráfico de puntos/dispersión/scatter plot**
 - Muestra la relación entre dos variables numéricas.
- **Histograma**
 - Aplica para *variables cuantitativas*. Permite ver la distribución de la variable.
- **Boxplot**

- Aplica para *variables cuantitativas*. Muestra una serie de medidas de resumen para dicha variable. Se suele graficar junto a una variable categórica para permitir la comparación entre niveles.

```
library(ggplot2)

## Warning: package 'ggplot2' was built under R version 4.2.2

library(ggpubr)

## Warning: package 'ggpubr' was built under R version 4.2.2

library(grid)

barra <- iris %>%
  group_by(Species) %>%
  summarise(n=n()) %>%
  ggplot(aes(Species,n, fill=Species)) +
  geom_bar(stat="identity")

scatter <- iris %>%
  ggplot(aes(Sepal.Length, Petal.Length, color=Species)) +
  geom_point()

histo <- iris %>%
  ggplot(aes(Sepal.Length, fill=Species)) +
  geom_histogram()

box <- iris %>%
  ggplot(aes(Species, Petal.Length, fill=Species)) +
  geom_boxplot()

ggarrange(barra,scatter,histo,box, ncol=2, nrow = 2, widths = c(0.5,0.5), labels=c("A"))

## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```

¿R base o Ggplot2?

La librería en la que se suelen armar los gráficos y que nunca falta en ningún script es **ggplot2**. Sin embargo, R permite realizar gráficos nativamente sin librerías.

R base

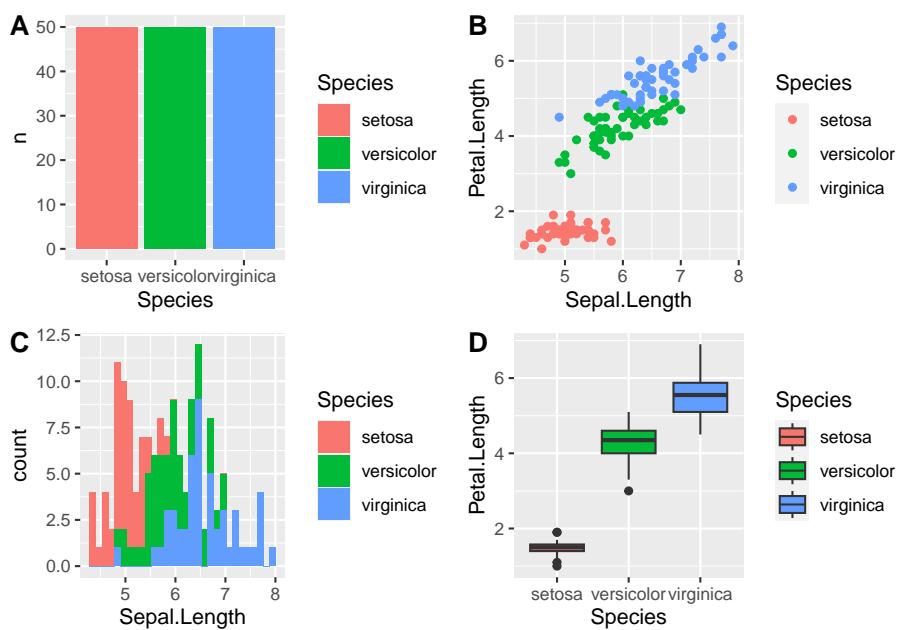
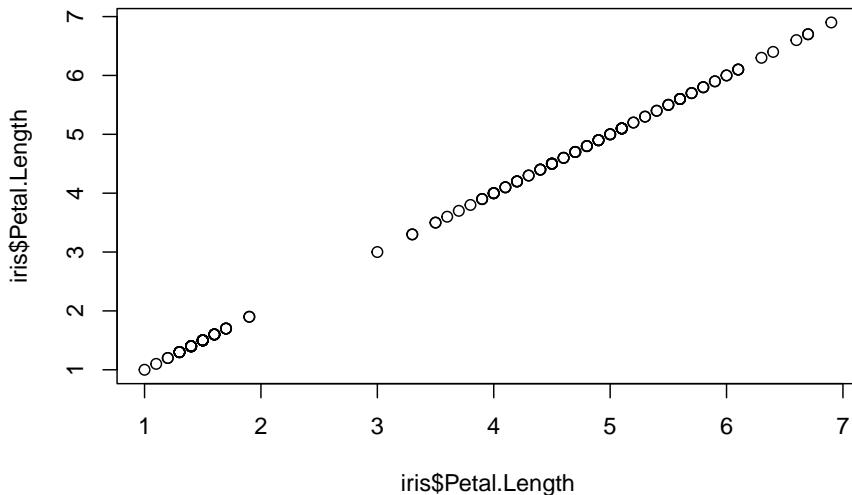


Figure 1.7: Tipos de gráficos. A) Gráfico de barras. B) Scatter plot, gráfico de puntos. C) Histograma. D) Boxplot

Las ventajas de usar R base para graficar es que es su rapidez para visualizar las relaciones entre variables, sin tener que preocuparnos por acordarnos de la sintaxis del script. Sin embargo, ggplot2 ofrece una variedad de combinaciones de customización para presentar los datos difícil de equiparar.

El comando básico es `plot()` entre dos variables. Es decir,

```
plot(iris$Petal.Length,iris$Petal.Length)
```



De ahí en adelante, se pueden agregar distintos parámetros que hagan a la customización. Por ejemplo,

```
plot(iris$Petal.Length,iris$Petal.Width,
     cex=2, # Tamaño de la forma
     pch=16, # Forma (puede ser circulo, triangulo, etc)
     xlab="Petal Length", # Titulo del eje x
     ylab="Petal Width", # Titulo del eje y
     main="Petal Width vs Petal Length of Iris", # Titulo del grafico
     col=iris$Species) # Si quiero agregarle color a los puntos. En ese caso, lo hace .
```

```
legend(x=1, y=2.4, legend=levels(iris$Species), col=c(1:3), pch=16) # Leyenda
```

Veamos como se hace el resto de los gráficos:

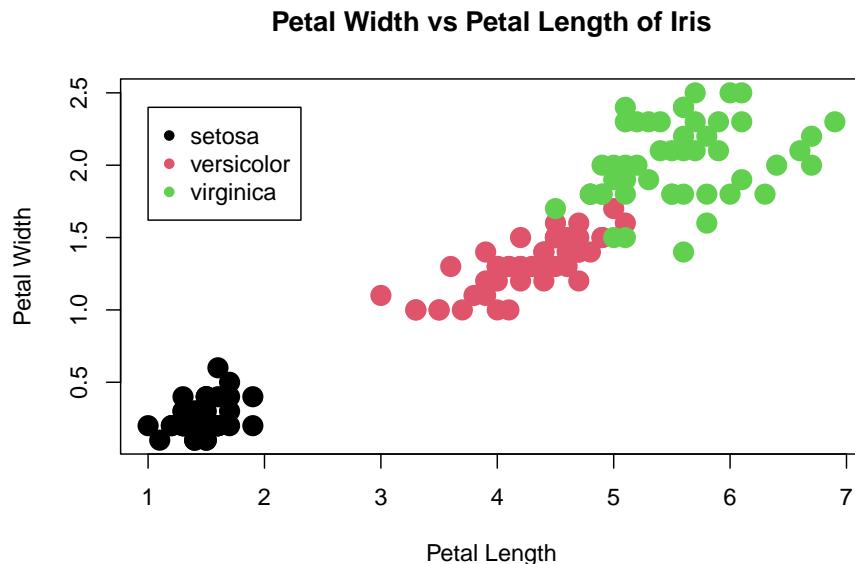


Figure 1.8: Tipos de gráficos

```
# Barra
plot(iris$Species)
```

```
# Boxplot
plot(iris$Species, iris$Sepal.Length,
     xlab="Species",
     ylab="Sepal Length",
     main="Sepal Length by Iris species",
     col="skyblue")
```

```
# Histogram
hist(iris$Sepal.Width,
     col="yellow",
     xlab="Sepal Width",
     main="Histogram of Sepal Width",
     breaks=30)
```

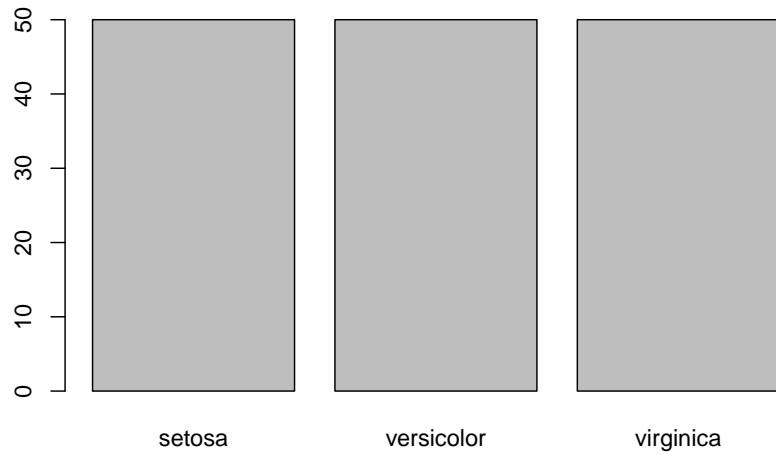


Figure 1.9: Tipos de gráficos

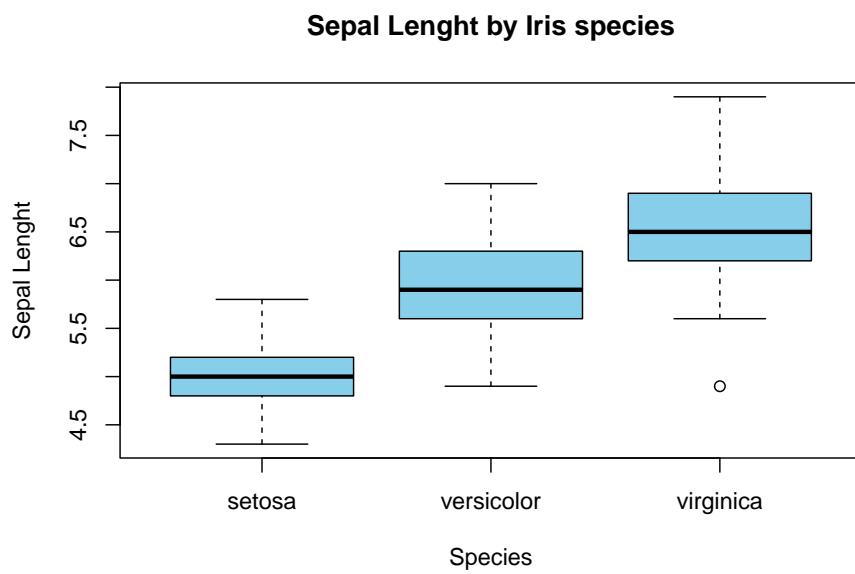


Figure 1.10: Tipos de gráficos

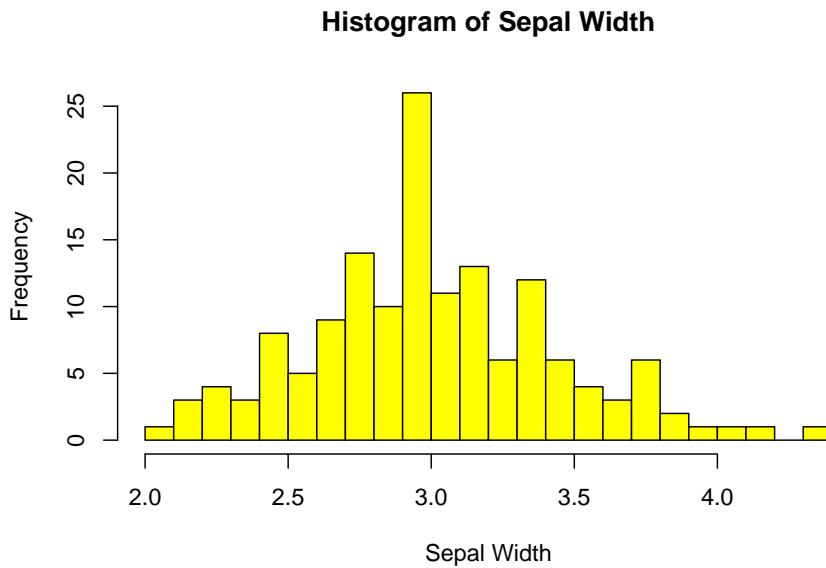


Figure 1.11: Tipos de gráficos

```
# Qué pasa si queremos plotear el data frame entero?
plot(iris)
```

```
ggplot2
```

Este paquete tiene una manera de escribirse particular en capas. Los gráficos de ggplot2 está compuesto por los datos, por un conjunto de características estéticas a tener en cuenta entre los datos junto con los aspectos visuales (*aes*, *aesthetic mapping*) y por al menos una capa que indica cómo se debe manipular cada observación (*geom_*).

```
library(ggplot2)
ggplot(iris, aes(x=Petal.Length, y=Petal.Width))+
  geom_point()
```

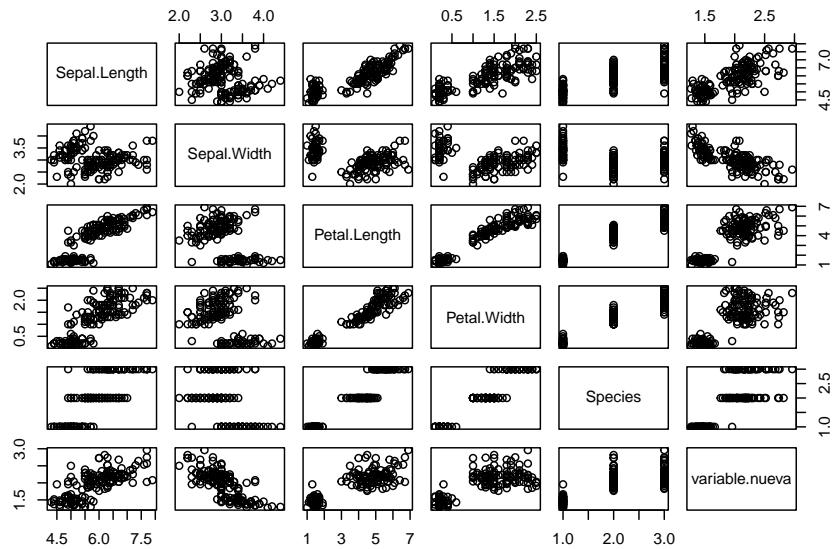
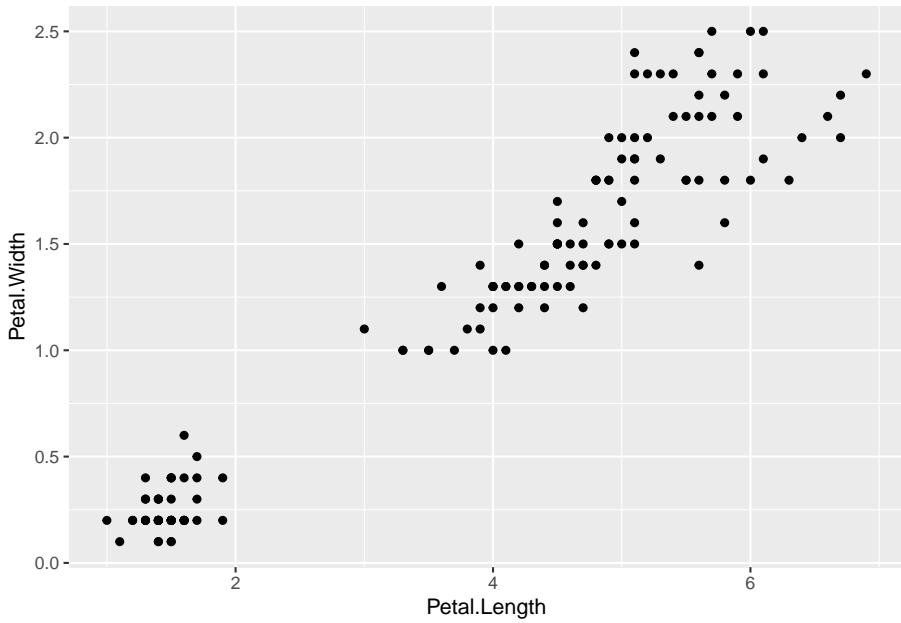


Figure 1.12: Tipos de gráficos

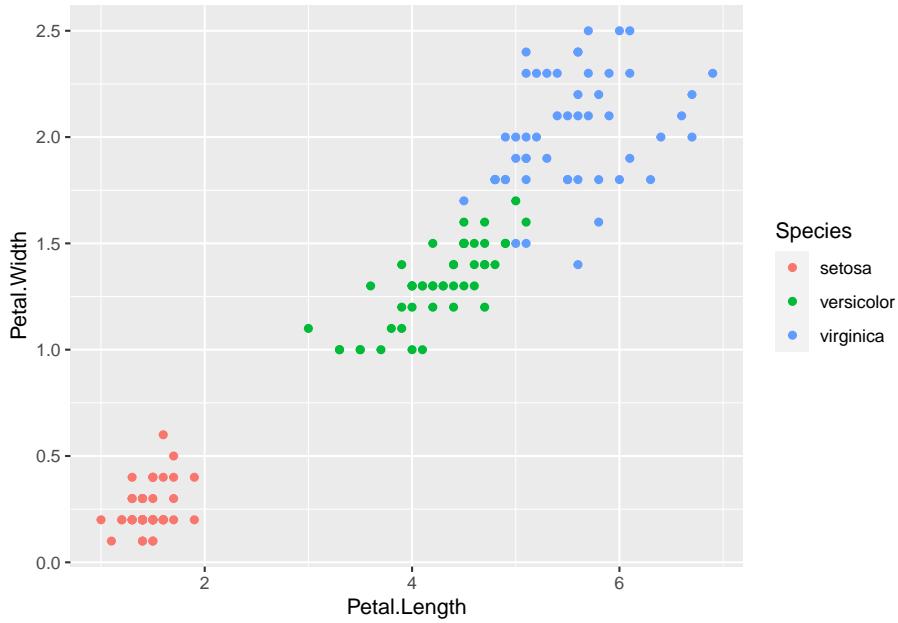


- **Buenas prácticas!** Conviene usar un renglón por capa, de esta manera

es fácil detectar errores y ver cómo se va modificando el gráfico a medida que van agregando capas.

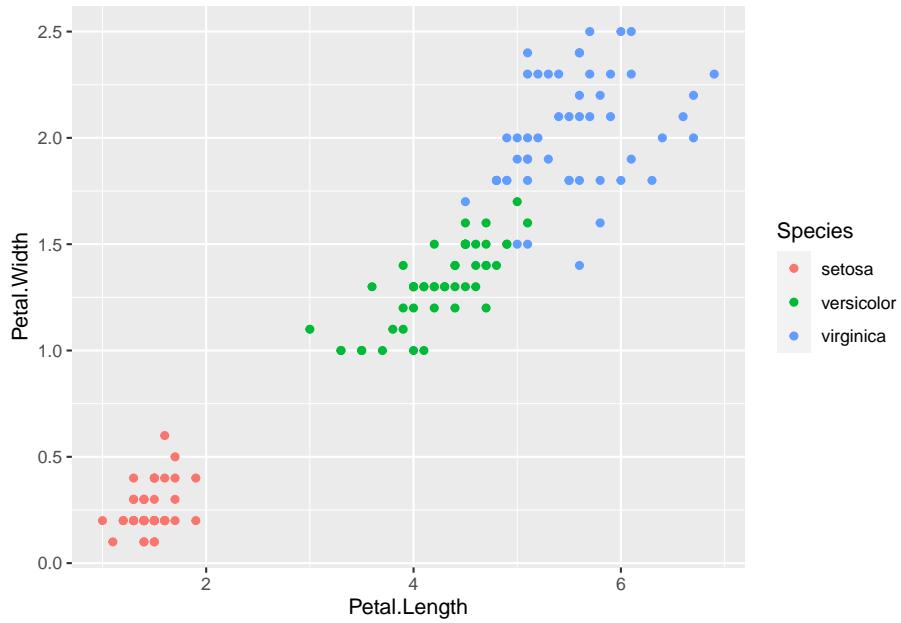
Al igual que en el plot del R base, se pueden cambiar los colores y formas. Sin embargo, esto va a estar atado a qué elemento queremos cambiar.

```
ggplot(iris, aes(x=Petal.Length, y=Petal.Width))+
  geom_point(aes(colour=Species))
```



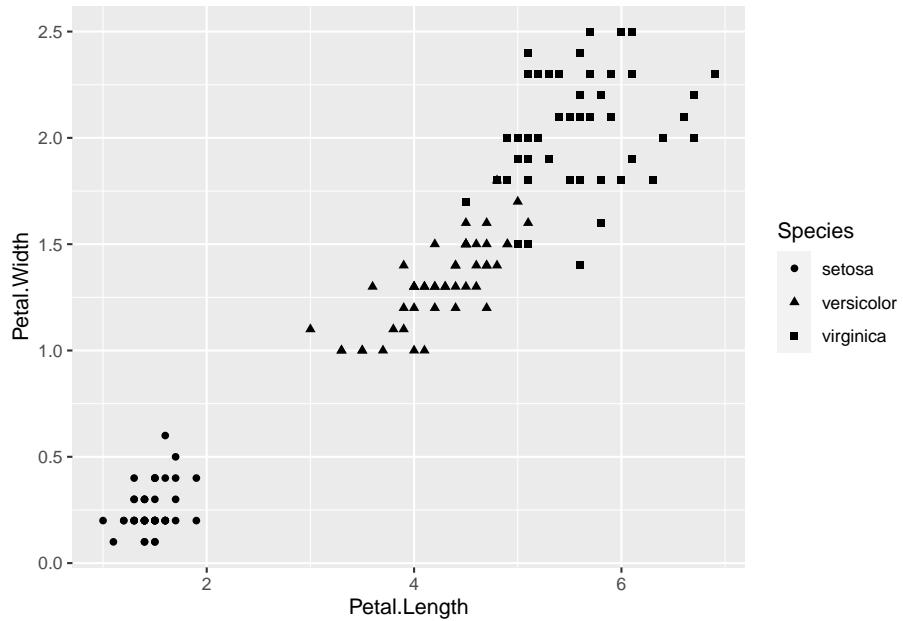
En este caso, quise cambiar el color de los puntos. Para ello, lo tengo que especificar en la capa *aes* del *geom_point*.

```
ggplot(iris, aes(x=Petal.Length, y=Petal.Width, color=Species))+  
  geom_point()
```

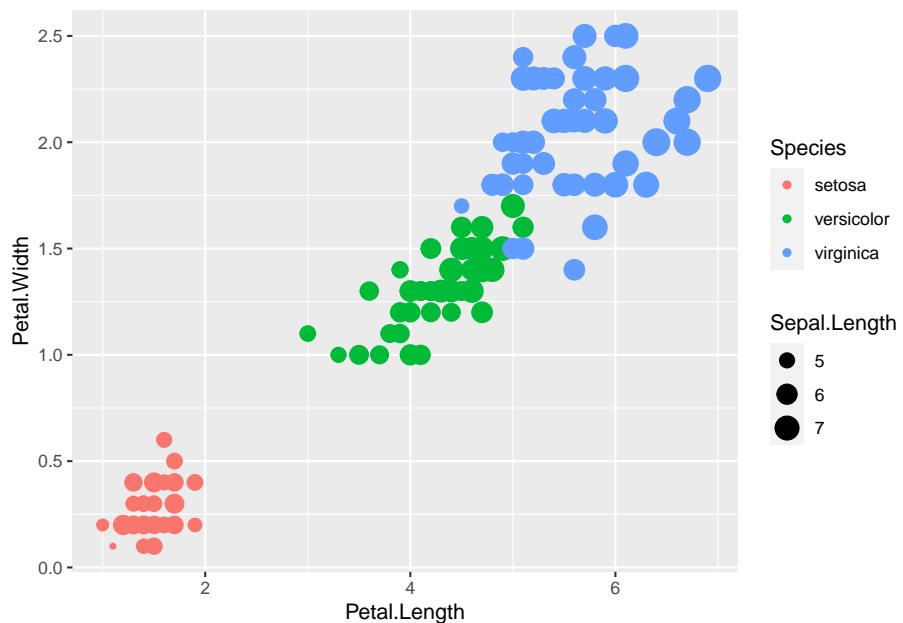


También puede ir en el aes general del gráfico. No solo puedo cambiar el color, sino también formas. O combinar todo:

```
ggplot(iris, aes(x=Petal.Length, y=Petal.Width, shape=Species))+  
  geom_point()
```



```
ggplot(iris, aes(x=Petal.Length, y=Petal.Width, size=Sepal.Length))+  
  geom_point(aes(color=Species))
```



Noten que se generan automáticamente las leyendas, según vamos cambiando

lo estético, a diferencia de R base.

Este es un primer pantallazo el mundo de ggplot. Hay mucho para aprender y en general, todo lo que quieran graficar van a poder, y de muchísimas maneras. Google es su gran amigo!

1.3.7 Recursos extra recomendados

Uso de la librería **ggpubr** para alinear plots <http://www.sthda.com/english/articles/24-ggpubr-publication-ready-plots/81-ggplot2-easy-way-to-mix-multiple-graphs-on-the-same-page/>

Healy K. 2019. Data visualization: A practical Introduction **Versión libre online:**

<https://socviz.co/index.html#preface>

R Charts <https://r-charts.com/es/ggplot2/>

Cheatsheet en Code <http://zevross.com/blog/2014/08/04/beautiful-plotting-in-r-a-ggplot2-cheatsheet-3>

Chapter 2

Correlaciones y regresiones múltiples

Natalia Morandeira¹

Instituto de Investigación e Ingeniería Ambiental (CONICET-UNSAM)

2.1 Introducción

En esta Unidad vamos a abordar análisis de correlación y regresiones múltiples (modelos lineales). Las correlaciones son análisis útiles para explorar nuestros datos y para evaluar si dos variables están asociadas positivamente o negativamente. Por otro lado, en el caso de una regresión simple, tenemos una variable respuesta (o dependiente) y una variable explicativa (o independiente). Un análisis de regresión múltiple es muy adecuado cuando hemos medido muchas variables, por ejemplo, si tenemos una variable respuesta y muchas posibles variables explicativas. Nos interesa conocer cuál o cuáles variables explican la variación de la variable respuesta, y elegir el mejor modelo.

En el marco de análisis limnológicos, una situación común es tener una variable respuesta medida en cuerpos de agua y múltiples variables que hipotetizamos que pueden explicar su variación. Entre las variables respuestas, podríamos tener (de acuerdo al objetivo de nuestro estudio) la abundancia, biomasa o diversidad de especies de un dado taxón o de grupos funcionales; o la concentración de un agroquímico; o incluso variables físico-químicas del cuerpo de agua que podrían depender de otro factor. Entre las variables respuestas, es posible que tengamos variables físico-químicas, características morfométricas de la laguna, usos de suelo del entorno, características de la cuenca, abundancia de otro taxón

¹nmorandeira@unsam.edu.ar

que interactúe con nuestra especie de interés, etc. Nos puede interesar en primer lugar analizar si están correlacionadas nuestras potenciales variables explicativas. Luego, podemos intentar ajustar un buen modelo que explique cómo varía nuestra variable respuesta en función de varias variables explicativas. Ese es el camino que recorreremos en esta Unidad.

2.2 Caso de estudio

El sitio de estudio es la turbera de Rancho Hambre (Tierra del Fuego, Argentina), en donde entre octubre de 2009 y abril de 2010 se realizaron muestreos en lagunas (ver artículo de Quiroga *et al.* 2017, y la tesis doctoral de María Victoria Quiroga (FCEN-UBA)). En cinco lagunas (RH1 a RH5), se realizaron muestreos estacionales (octubre, diciembre, febrero y abril), es decir, tenemos un total de 20 observaciones. Contamos con dos tablas de datos:

- **Datos limnológicos físicoquímicos.** Los datos corresponden a los sitios de muestreo de orilla de los cuerpos de agua. Los parámetros limnológicos fueron estimados según lo descripto en la sección *Sampling regime, physical-chemical analyses* de Materiales y Métodos en Quiroga *et al.* (2017). Los datos físicoquímicos están disponibles en el repositorio digital de CONICET. Consideraremos a las variables medidas como variables explicativas: pH, conductividad, oxígeno disuelto, dureza total, DOC, DIN, nitrógeno total, DRP, fósforo total, ag(440), SUVA254 e índice MS; todas ellas variables numéricas. Por otro lado tenemos variables categóricas que discutiremos cómo considerar: la laguna (cinco niveles) y la fecha de muestreo (cuatro niveles).
- **Biomasa de morfotipos bacterianos.** La biomasa de bacterias heterótrofas se estimó utilizando microscopía de epifluorescencia y análisis de imágenes. Para detalles ver sección *Bacterioplankton morphotypes* de Materiales y Métodos en Quiroga *et al.* (2017). Los datos de morfotipos bacterianos están disponibles en el repositorio digital de CONICET. En la tabla se cuenta con la biomasa de seis morfotipos bacterianos distintos. Podemos tomar a la biomasa de cada morfotipo como una variable respuesta por separado, o bien considerar para un primer análisis a la biomasa total de bacterias heterótrofas como la variable a modelar.

2.3 Leer y emprolijar los datos

Siempre debemos mirar nuestra tabla de datos antes de empezar: detectar posibles errores o datos faltantes, retocar los nombres de las variables, agrupar filas y columnas si corresponde, etc. En nuestro caso además tenemos por un lado

las variables físico-químicas y por el otro lado las variables biológicas, pero para los análisis es conveniente tener todos los datos en un mismo *dataframe*.

Empezamos leyendo las bases de datos en R. Es una buena práctica guardar los archivos en una carpeta llamada *data*, dentro de la carpeta de nuestro proyecto. Tenemos los datos en formato *csv* en GitHub Limno-con-R/CILCAL2023 **RH_abiotic_data.csv** y **RH_morpho_biomass.csv** (son casi iguales a los disponibles en el repositorio CONICET, se cambió un guión bajo por un espacio en una de las tablas). Cargamos los datos indicando que se saltee (*skip*) la primera fila (¿por qué?).

```
library(readr)

datos_fq <- read_csv("data/RH_abiotic_data.csv", skip = 1)

datos_bacterias <- read_csv("data/RH_morpho_biomass.csv", skip = 1)
```

Ahora vamos a ver los datos. Recordar o anotar cuántas filas y columnas tiene cada *dataframe*.

```
datos_fq
```

```
## # A tibble: 20 x 16
##   ID    Pool Site     Date      pH Condu~1 Disso~2 Total~3   DOC   DIN Total~4
##   <chr> <chr> <chr>   <chr> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl>
## 1 10   RH1   shore   Octo~  4.91   13.4  NA    21.2  5.6   83.6  1870 
## 2 20   RH2   shore   Octo~  5.52   8.7   NA    46.2  5.1   41.3  1980 
## 3 30   RH3   shore   Octo~  4.72   15.2  NA    13.8  2.8   10    1980 
## 4 40   RH4   north sh~ Octo~  5.09   11.8  NA    33.2  5.7   55.3  1650 
## 5 50   RH5   shore   Octo~  4.82   5.5   NA    25.8  3.9   54.4  3410 
## 6 1D   RH1   shore   Dece~  6.39   19.1  11.0  38.5  7.2   21.1  1320 
## 7 2D   RH2   shore   Dece~  4.75   22.6  11.0  42.1  7.7   12.1  1870 
## 8 3D   RH3   shore   Dece~  4.67   26.7  11.7  18.9  13.4  22.6  5720 
## 9 4D   RH4   north sh~ Dece~  6.75   25.6  11.7  24.8  5.3   34    7480 
## 10 5D  RH5   shore   Dece~  4.65   23    11.5  20.1  11    11.2  10230 
## 11 1F  RH1   shore   Febr~  5.9    21.4  10.4  39.5  12.7  53    8030 
## 12 2F  RH2   shore   Febr~  5.19   22.6  10.2  34    5.9   30.3  10010 
## 13 3F  RH3   shore   Febr~  4.66   25.7  10.6  32    12.5  45.3  11330 
## 14 4F  RH4   north sh~ Febr~  6.65   26.8  10.9  36.6  4     0     5610 
## 15 5F  RH5   shore   Febr~  4.82   27.2  10.4  24.5  7.1   22.3  8250 
## 16 1A  RH1   shore   Apri~  7.1    21.4  10.7  25.3  7.6   23.1  9790 
## 17 2A  RH2   shore   Apri~  4.88   24    11.4  20.3  7.5   0     11110 
## 18 3A  RH3   shore   Apri~  5.4    28.5  10.4  18.8  12.7  103.  8910 
## 19 4A  RH4   north sh~ Apri~  6.2    28.5  11.2  27.7  5.3   43    3630 
## 20 5A  RH5   shore   Apri~  5.43   27.1  11.2  24.2  11.6  73.2  3740
```

```
## # ... with 5 more variables: DRP <dbl>, `Total phosphorus` <dbl>,
## #   `ag(440)` <dbl>, SUVA254 <dbl>, `MS index` <dbl>, and abbreviated variable
## #   names 1: Conductivity, 2: `Dissolved oxygen`, 3: `Total hardness`,
## #   4: `Total nitrogen`
```

datos_bacterias

```
## # A tibble: 20 x 10
##   ID    Pool Site      Date Filam~1 Large~2 Vibri~3 Large~4 Small~5 Small~6
##   <chr> <chr> <chr>     <chr>   <dbl>   <dbl>   <dbl>   <dbl>   <dbl>   <dbl>
## 1 10   RH1   shore   Octo~       0    16921    5315   12676   1747   25994
## 2 20   RH2   shore   Octo~      1435    5645   2384    7381    888   12075
## 3 30   RH3   shore   Octo~      13307   1597    392    8131    708   13222
## 4 40   RH4   north shore Octo~       0    8085    3499    7534   2028   25353
## 5 50   RH5   shore   Octo~       0    1921    451    15662    666   31621
## 6 1D   RH1   shore   Dece~       0   12374    9228   21906   4503   38421
## 7 2D   RH2   shore   Dece~       0   22924    5933   34663   6687   77581
## 8 3D   RH3   shore   Dece~       0   12309    3471   46807   8694   139006
## 9 4D   RH4   north shore Dece~      44419   76819   19692   44391   24931  156132
## 10 5D  RH5   shore   Dece~       0   66952   23046   134707   2949  102890
## 11 1F  RH1   shore   Febr~      3177    6414    7533   22218   4013   34495
## 12 2F  RH2   shore   Febr~       0    7263    4879   25886   4098  137049
## 13 3F  RH3   shore   Febr~      3886    9283   1294   17402   2522   93008
## 14 4F  RH4   north shore Febr~      5860   54624   20331   33949  12469  58990
## 15 5F  RH5   shore   Febr~       0   21324   17482   24397   7064  106885
## 16 1A  RH1   shore   Apri~      12331   21098   11615   16083   6592  33478
## 17 2A  RH2   shore   Apri~       0   6950    4028   22525   1395  94037
## 18 3A  RH3   shore   Apri~       0   12394   17281   18608   4678  121090
## 19 4A  RH4   north shore Apri~       0    901    2484    3455    642   50957
## 20 5A  RH5   shore   Apri~       0   7626    3338   17231   2016  60287
## # ... with abbreviated variable names 1: Filaments, 2: Large_rods,
## #   3: Vibrio_shaped, 4: Large_cocci, 5: Small_rods, 6: Small_cocci
```

Para unir estas tablas en una sola, debemos buscar una variable o columna que sea un indicador único de cada dato, y que sea equivalente en ambas tablas. ¿Cuál les parece que es?

Con ese indicador como nexo, haremos una **unión exterior completa** o ***full_join***. Esto se debe a que queremos mantener todos los registros de nuestras tablas aunque para alguna laguna –quizás– no hayamos podido medir los parámetros físicoquímicos o no hayamos podido medir la biomasa bacteriana. Para saber más sobre funciones que permiten relacionar conjuntos de datos, recomendamos consultar el capítulo 13. Datos relacionales (en particular 13.4.1. *Entendiendo las uniones*) en el libro traducido a castellano R para Ciencias de Datos, de Hadley Wickham y Garret Grolemund (2017).

Hacemos la unión e inspeccionamos la tabla (¿cuántas filas y columnas tiene?).

```
library(tidyverse)

datos_RH <- full_join(datos_bacterias, datos_fq, by = "ID")
datos_RH

## # A tibble: 20 x 25
##   ID    Pool.x Site.x Date.x Filam~1 Large~2 Vibri~3 Large~4 Small~5 Small~6
##   <chr> <chr>  <chr>  <chr>    <dbl>    <dbl>    <dbl>    <dbl>    <dbl>    <dbl>
## 1 10   RH1    shore   Octob~     0    16921    5315    12676    1747   25994
## 2 20   RH2    shore   Octob~    1435    5645    2384    7381     888   12075
## 3 30   RH3    shore   Octob~   13307   1597     392    8131     708   13222
## 4 40   RH4    north sh~ Octob~     0    8085    3499    7534    2028   25353
## 5 50   RH5    shore   Octob~     0    1921     451    15662    666   31621
## 6 1D   RH1    shore   Decem~     0   12374    9228    21906    4503   38421
## 7 2D   RH2    shore   Decem~     0   22924    5933    34663    6687   77581
## 8 3D   RH3    shore   Decem~     0   12309    3471    46807    8694   139006
## 9 4D   RH4    north sh~ Decem~   44419   76819   19692    44391   24931  156132
## 10 5D  RH5    shore   Decem~     0   66952    23046   134707    2949   102890
## 11 1F  RH1    shore   Febru~    3177    6414    7533    22218    4013   34495
## 12 2F  RH2    shore   Febru~     0    7263    4879    25886    4098   137049
## 13 3F  RH3    shore   Febru~    3886    9283    1294    17402    2522   93008
## 14 4F  RH4    north sh~ Febru~   5860    54624   20331   33949   12469  58990
## 15 5F  RH5    shore   Febru~     0   21324   17482   24397    7064   106885
## 16 1A  RH1    shore   April~   12331   21098   11615   16083    6592   33478
## 17 2A  RH2    shore   April~     0    6950    4028    22525    1395   94037
## 18 3A  RH3    shore   April~     0   12394   17281   18608    4678   121090
## 19 4A  RH4    north sh~ April~     0    901    2484    3455     642   50957
## 20 5A  RH5    shore   April~     0   7626    3338   17231    2016   60287
## # ... with 15 more variables: Pool.y <chr>, Site.y <chr>, Date.y <chr>,
## # pH <dbl>, Conductivity <dbl>, `Dissolved oxygen` <dbl>,
## # `Total hardness` <dbl>, DOC <dbl>, DIN <dbl>, `Total nitrogen` <dbl>,
## # DRP <dbl>, `Total phosphorus` <dbl>, `ag(440)` <dbl>, SUVA254 <dbl>,
## # `MS index` <dbl>, and abbreviated variable names 1: Filaments,
## # 2: Large_rods, 3: Vibrio_shaped, 4: Large_cocci, 5: Small_rods,
## # 6: Small_cocci
```

Las variables Pool, Site y Date se repiten en ambas tablas, por eso aparecen como Pool.x y Pool.y (por ejemplo). Dado que son idénticas, tenemos la opción de eliminar estas variables de alguna de las dos tablas, o bien incluirlas como parte de la información de nexo.

```

datos_RH <- full_join(datos_bacterias, datos_fq, by = c("ID", "Pool", "Site", "Date"))
datos_RH

## # A tibble: 20 x 22
##   ID    Pool Site Date Filam~1 Large~2 Vibri~3 Large~4 Small~5 Small~6 pH
##   <chr> <chr> <chr> <chr>   <dbl>   <dbl>   <dbl>   <dbl>   <dbl>   <dbl>
## 1 10   RH1  shore Octo~      0    16921    5315   12676   1747  25994  4.91
## 2 20   RH2  shore Octo~     1435    5645   2384    7381    888  12075  5.52
## 3 30   RH3  shore Octo~    13307   1597    392    8131    708  13222  4.72
## 4 40   RH4  nort~ Octo~      0    8085    3499    7534   2028  25353  5.09
## 5 50   RH5  shore Octo~      0    1921    451    15662    666  31621  4.82
## 6 1D   RH1  shore Dece~      0   12374    9228   21906   4503  38421  6.39
## 7 2D   RH2  shore Dece~      0   22924    5933   34663   6687  77581  4.75
## 8 3D   RH3  shore Dece~      0   12309    3471   46807   8694  139006 4.67
## 9 4D   RH4  nort~ Dece~    44419   76819   19692   44391  24931  156132 6.75
## 10 5D  RH5  shore Dece~      0   66952   23046   134707  2949  102890 4.65
## 11 1F  RH1  shore Febr~    3177    6414    7533   22218   4013  34495  5.9
## 12 2F  RH2  shore Febr~      0   7263    4879   25886   4098  137049 5.19
## 13 3F  RH3  shore Febr~    3886    9283    1294   17402   2522  93008  4.66
## 14 4F  RH4  nort~ Febr~    5860    54624   20331   33949  12469  58990  6.65
## 15 5F  RH5  shore Febr~      0   21324   17482   24397  7064  106885 4.82
## 16 1A  RH1  shore Apri~   12331   21098   11615   16083  6592  33478  7.1
## 17 2A  RH2  shore Apri~      0   6950    4028   22525   1395  94037  4.88
## 18 3A  RH3  shore Apri~      0   12394   17281   18608   4678  121090 5.4
## 19 4A  RH4  nort~ Apri~      0    901    2484    3455    642  50957  6.2
## 20 5A  RH5  shore Apri~      0   7626    3338   17231   2016  60287  5.43
## # ... with 11 more variables: Conductivity <dbl>, `Dissolved oxygen` <dbl>,
## #   `Total hardness` <dbl>, DOC <dbl>, DIN <dbl>, `Total nitrogen` <dbl>,
## #   DRP <dbl>, `Total phosphorus` <dbl>, `ag(440)` <dbl>, SUVA254 <dbl>,
## #   `MS index` <dbl>, and abbreviated variable names 1: Filaments,
## #   2: Large_rods, 3: Vibrio_shaped, 4: Large_cocci, 5: Small_rods,
## #   6: Small_cocci

```

Ahora vamos a mejorar los nombres de las columnas. Una buena práctica es elegir siempre el mismo tipo de nomenclatura. Hay varias convenciones pero las más elegidas en el mundo de R son *snake* y *camel*. La nomenclatura *snake* implica usar guiones bajos como separador de palabras y, generalmente, todas las letras en minúscula. La nomenclatura *camel* implica, en vez de usar guiones, usar a las mayúsculas como indicador de que hay una palabra nueva, y siempre dejar a la primera palabra en minúscula. Si nuestra variable se llama “Fecha de muestreo”, la nomenclatura sería *fecha_de_muestreo* o *fechaDeMuestreo*, de acuerdo a la convención elegida.

En la tabla que tenemos, hay una mezcla de criterios: se usan guiones bajos pero a la vez las variables empiezan con mayúscula. Vamos a ajustar. Vamos a pasar

todas las variables al formato *snake* con una función muy útil, que también sirve para nombres de columnas más desprolijos (por ejemplo, si tenemos otros signos de puntuación en los nombres).

```
library(janitor)

datos_RH <- clean_names(datos_RH)
datos_RH

## # A tibble: 20 x 22
##   id    pool  site  date filam~1 large~2 vibri~3 large~4 small~5 small~6 p_h
##   <chr> <chr> <chr> <chr>   <dbl>   <dbl>   <dbl>   <dbl>   <dbl>   <dbl>
## 1 10   RH1   shore Octo~      0     16921    5315   12676   1747   25994  4.91
## 2 20   RH2   shore Octo~     1435    5645   2384    7381    888   12075  5.52
## 3 30   RH3   shore Octo~    13307   1597    392    8131    708   13222  4.72
## 4 40   RH4   nort~ Octo~      0     8085   3499    7534   2028   25353  5.09
## 5 50   RH5   shore Octo~      0     1921    451    15662    666   31621  4.82
## 6 1D   RH1   shore Dece~      0    12374   9228   21906   4503   38421  6.39
## 7 2D   RH2   shore Dece~      0    22924   5933   34663   6687   77581  4.75
## 8 3D   RH3   shore Dece~      0    12309   3471   46807   8694   139006 4.67
## 9 4D   RH4   nort~ Dece~    44419   76819   19692   44391   24931  156132 6.75
## 10 5D  RH5   shore Dece~      0    66952   23046   134707   2949   102890 4.65
## 11 1F  RH1   shore Febr~    3177    6414    7533   22218   4013   34495  5.9
## 12 2F  RH2   shore Febr~      0    7263    4879   25886   4098   137049 5.19
## 13 3F  RH3   shore Febr~    3886    9283    1294   17402   2522   93008  4.66
## 14 4F  RH4   nort~ Febr~    5860    54624   20331   33949   12469  58990  6.65
## 15 5F  RH5   shore Febr~      0    21324   17482   24397   7064   106885 4.82
## 16 1A  RH1   shore Apri~   12331   21098   11615   16083   6592   33478  7.1
## 17 2A  RH2   shore Apri~      0    6950    4028   22525   1395   94037  4.88
## 18 3A  RH3   shore Apri~      0    12394   17281   18608   4678   121090 5.4
## 19 4A  RH4   nort~ Apri~      0     901    2484    3455    642   50957  6.2
## 20 5A  RH5   shore Apri~      0    7626    3338   17231   2016   60287  5.43
## # ... with 11 more variables: conductivity <dbl>, dissolved_oxygen <dbl>,
## #   total_hardness <dbl>, doc <dbl>, din <dbl>, total_nitrogen <dbl>,
## #   drp <dbl>, total_phosphorus <dbl>, ag_440 <dbl>, suva254 <dbl>,
## #   ms_index <dbl>, and abbreviated variable names 1: filaments, 2: large_rods,
## #   3: vibrio_shaped, 4: large_cocci, 5: small_rods, 6: small_cocci
```

El nombre de la variable pH quedó un poco raro, lo vamos a cambiar. A continuación se pide el nombre de las columnas de datos_RH, luego específicamente el de la columna 11, y luego lo cambiamos.

```
colnames(datos_RH)
```

```

## [1] "id"           "pool"          "site"          "date"
## [5] "filaments"    "large_rods"     "vibrio_shaped" "large_cocci"
## [9] "small_rods"   "small_cocci"    "p_h"           "conductivity"
## [13] "dissolved_oxygen" "total_hardness" "doc"           "din"
## [17] "total_nitrogen"  "drp"           "total_phosphorus" "ag_440"
## [21] "suva254"       "ms_index"

colnames(datos_RH) [11]

## [1] "p_h"

colnames(datos_RH) [11] <- "ph"
colnames(datos_RH)

## [1] "id"           "pool"          "site"          "date"
## [5] "filaments"    "large_rods"     "vibrio_shaped" "large_cocci"
## [9] "small_rods"   "small_cocci"    "ph"           "conductivity"
## [13] "dissolved_oxygen" "total_hardness" "doc"           "din"
## [17] "total_nitrogen"  "drp"           "total_phosphorus" "ag_440"
## [21] "suva254"       "ms_index"

```

Finalmente, vamos a agregar una nueva columna con la biomasa total bacteriana. Existen múltiples métodos para hacer esto, vamos a ver dos, uno con la sintaxis base de R y otro con la sintaxis de *tidyverse*. Para conocer más sobre los tipos de sintaxis de R, recomendamos la hoja de referencia Comparación de sintaxis de R, por Amelia McNamara (traducida al castellano por RLadies).

```

#Con la sintaxis base
datos_bio_opcion1 <- datos_RH #duplico el dataframe para no sobreescribirlo y comparar

datos_bio_opcion1$bio_total <- datos_bio_opcion1$filaments + datos_bio_opcion1$large_r

datos_bio_opcion1

## # A tibble: 20 x 23
##      id    pool   site   date filam~1 large~2 vibri~3 large~4 small~5 small~6    ph
##      <chr> <chr> <chr> <chr>    <dbl>    <dbl>    <dbl>    <dbl>    <dbl>    <dbl>    <dbl>
## 1 10   RH1    shore Octo~     0    16921    5315    12676    1747    25994    4.91
## 2 20   RH2    shore Octo~    1435    5645    2384    7381     888    12075    5.52
## 3 30   RH3    shore Octo~   13307    1597     392    8131     708    13222    4.72
## 4 40   RH4    nort~ Octo~     0    8085    3499    7534    2028    25353    5.09
## 5 50   RH5    shore Octo~     0    1921     451    15662     666    31621    4.82
## 6 1D   RH1    shore Dece~     0    12374    9228    21906    4503    38421    6.39

```

```

## 7 2D RH2 shore Dece~ 0 22924 5933 34663 6687 77581 4.75
## 8 3D RH3 shore Dece~ 0 12309 3471 46807 8694 139006 4.67
## 9 4D RH4 nort~ Dece~ 44419 76819 19692 44391 24931 156132 6.75
## 10 5D RH5 shore Dece~ 0 66952 23046 134707 2949 102890 4.65
## 11 1F RH1 shore Febr~ 3177 6414 7533 22218 4013 34495 5.9
## 12 2F RH2 shore Febr~ 0 7263 4879 25886 4098 137049 5.19
## 13 3F RH3 shore Febr~ 3886 9283 1294 17402 2522 93008 4.66
## 14 4F RH4 nort~ Febr~ 5860 54624 20331 33949 12469 58990 6.65
## 15 5F RH5 shore Febr~ 0 21324 17482 24397 7064 106885 4.82
## 16 1A RH1 shore Apri~ 12331 21098 11615 16083 6592 33478 7.1
## 17 2A RH2 shore Apri~ 0 6950 4028 22525 1395 94037 4.88
## 18 3A RH3 shore Apri~ 0 12394 17281 18608 4678 121090 5.4
## 19 4A RH4 nort~ Apri~ 0 901 2484 3455 642 50957 6.2
## 20 5A RH5 shore Apri~ 0 7626 3338 17231 2016 60287 5.43
## # ... with 12 more variables: conductivity <dbl>, dissolved_oxygen <dbl>,
## #   total_hardness <dbl>, doc <dbl>, din <dbl>, total_nitrogen <dbl>,
## #   drp <dbl>, total_phosphorus <dbl>, ag_440 <dbl>, suva254 <dbl>,
## #   ms_index <dbl>, bio_total <dbl>, and abbreviated variable names
## #   1: filaments, 2: large_rods, 3: vibrio_shaped, 4: large_cocci,
## #   5: small_rods, 6: small_cocci

```

#Con la sintaxis tidyverse, la cual usa el operador _pipe_ ("la pipa") %>%

```

datos_bio_opcion2 <- datos_RH %>%
  mutate(bio_total = filaments + large_rods + vibrio_shaped + large_cocci + small_rods + small_cocci)

datos_bio_opcion2

```

```

## # A tibble: 20 x 23
##   id    pool  site      date bio_t~1 filam~2 large~3 vibri~4 large~5 small~6
##   <chr> <chr> <chr>     <chr> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl>
## 1 10   RH1   shore     Octo~  62653    0    16921  5315  12676  1747
## 2 20   RH2   shore     Octo~  29808   1435   5645  2384  7381   888
## 3 30   RH3   shore     Octo~  37357  13307   1597   392   8131   708
## 4 40   RH4   north shore Octo~  46499    0    8085  3499  7534  2028
## 5 50   RH5   shore     Octo~  50321    0    1921   451   15662  666
## 6 1D   RH1   shore     Dece~  86432    0    12374  9228  21906  4503
## 7 2D   RH2   shore     Dece~ 147788   0    22924  5933  34663  6687
## 8 3D   RH3   shore     Dece~ 210287   0    12309  3471  46807  8694
## 9 4D   RH4   north shore Dece~ 366384  44419  76819  19692  44391  24931
## 10 5D  RH5   shore     Dece~ 330544   0    66952  23046  134707 2949
## 11 1F  RH1   shore     Febr~ 77850   3177   6414  7533  22218  4013
## 12 2F  RH2   shore     Febr~ 179175   0    7263  4879  25886  4098
## 13 3F  RH3   shore     Febr~ 127395  3886  9283  1294  17402  2522
## 14 4F  RH4   north shore Febr~ 186223  5860  54624  20331  33949 12469

```

```

## 15 5F    RH5    shore      Febr~ 177152        0   21324  17482  24397  7064
## 16 1A    RH1    shore      Apri~ 101197  12331  21098  11615  16083  6592
## 17 2A    RH2    shore      Apri~ 128935        0   6950   4028  22525  1395
## 18 3A    RH3    shore      Apri~ 174051        0   12394  17281  18608  4678
## 19 4A    RH4    north shore Apri~ 58439         0   901    2484  3455   642
## 20 5A    RH5    shore      Apri~ 90498         0   7626   3338  17231  2016
## # ... with 13 more variables: small_cocci <dbl>, ph <dbl>, conductivity <dbl>,
## # dissolved_oxygen <dbl>, total_hardness <dbl>, doc <dbl>, din <dbl>,
## # total_nitrogen <dbl>, drp <dbl>, total_phosphorus <dbl>, ag_440 <dbl>,
## # suva254 <dbl>, ms_index <dbl>, and abbreviated variable names 1: bio_total,
## # 2: filaments, 3: large_rods, 4: vibrio_shaped, 5: large_cocci,
## # 6: small_rods

```

La ventaja de la segunda opción es que podemos elegir dónde agregar la nueva columna (antes de la columna *filaments*, por ejemplo). Así que nos quedaremos con esta tabla para los siguientes pasos. De paso, borramos los dataframes que ya no necesitamos.

```

datos_RH <- datos_bio_opcion2

rm(datos_bio_opcion2)
rm(datos_bio_opcion1)

```

Un último paso es informar que las variables *pool* y *date* son categóricas (factores). En el caso de *date*, además es adecuado ordenar los niveles de los factores, tanto para algunos análisis que consideren el orden de los meses (aquí nos exceden) como para mejorar la forma en que resumimos los resultados en una tabla o en un gráfico.

```

datos_RH$pool <- factor(datos_RH$pool) #en este caso no indicamos los niveles, ya que ...

datos_RH$date <- factor(datos_RH$date, levels = c("October_2009", "December_2009", "Febr...")

```

¡Listo! Podemos empezar con los análisis.

2.4 Correlaciones

Primero vamos a realizar análisis rápidos de todos los pares de variables con la librería *GGally*.

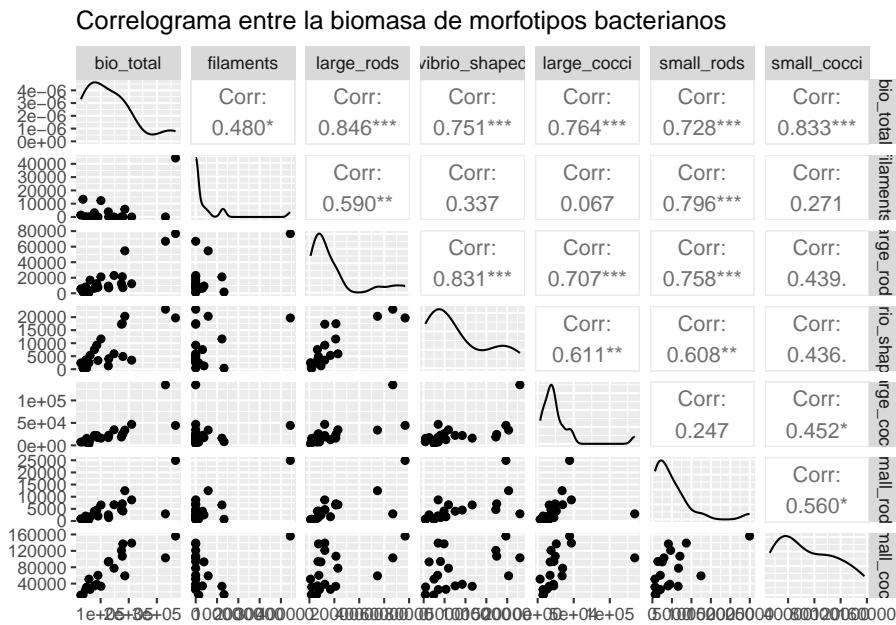
```
library(GGally)
```

2.4.1 Entre biomasa de morfotipos bacterianos

Vamos a analizar en primer lugar las variables biológicas, para lo cual generamos un nuevo dataframe que es un subconjunto de los anteriores.

```
datos_bacterias <- datos_RH[,5:11]

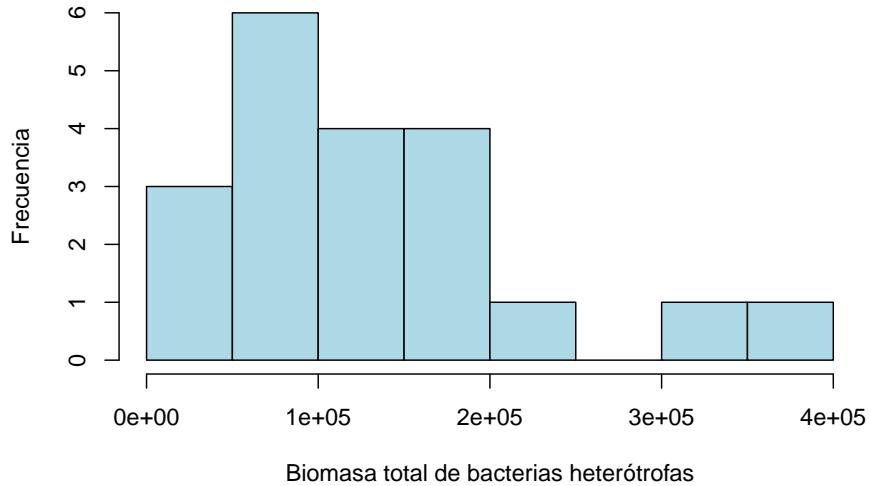
ggpairs(datos_bacterias, title="Correlograma entre la biomasa de morfotipos bacterianos")
```



Obtenemos una matriz de gráficos, en la que cada celda expresa la correlación de un par de variables. En este caso, todas las variables son continuas, por lo que observamos que los gráficos en la parte inferior de la matriz son de dispersión (puntos).

En las diagonales, observamos histogramas suavizados para cada variable. Por ejemplo, comparemos el histograma de la biomasa total.

```
hist(datos_bacterias$bio_total, breaks=10, col="lightblue", xlab="Biomasa total de bacterias hetero")
```



En la parte superior, se muestra el coeficiente de correlación para cada par de variables y asteriscos indicando su nivel de significancia. Podemos evaluar el par que nos interese con una función, para tener información numérica más detallada.

```
cor.test(datos_bacterias$vibrio_shaped, datos_bacterias$large_rods)

##
## Pearson's product-moment correlation
##
## data: datos_bacterias$vibrio_shaped and datos_bacterias$large_rods
## t = 6.3382, df = 18, p-value = 5.68e-06
## alternative hypothesis: true correlation is not equal to 0
## 95 percent confidence interval:
## 0.6144454 0.9311213
## sample estimates:
## cor
## 0.8310104
```

Aquí observamos que el coeficiente de la correlación de Pearson es $r = 0.8310$, igual al que observamos en el correlograma. Además obtenemos un intervalo de confianza y un p-valor. Sin embargo, la correlación de Pearson supone que la distribución de las variables es normal. Evaluemos la normalidad de estas dos variables (deberíamos analizarlo para todas).

```
shapiro.test(datos_bacterias$vibrio_shaped)
```

```
##  
## Shapiro-Wilk normality test  
##  
## data: datos_bacterias$vibrio_shaped  
## W = 0.8493, p-value = 0.005189
```

```
shapiro.test(datos_bacterias$large_rods)
```

```
##  
## Shapiro-Wilk normality test  
##  
## data: datos_bacterias$large_rods  
## W = 0.71437, p-value = 5.866e-05
```

Dado que se rechaza el supuesto de distribución normal, tenemos dos opciones. La primera es hacer una correlación de Spearman, la cual es adecuada si suponemos que la relación entre las variables es monotónica (las variables siempre crecen o decrecen). La desventaja es que, en la fórmula de cómputo del método, las variables se ordenan en un rango y pasan a ser ordinales. Agregamos a la misma función que el método sea Spearman (por defecto es Pearson, ver la ayuda de la función con `?cor.test`). El rho es de 0.8316.

```
cor.test(datos_bacterias$vibrio_shaped, datos_bacterias$large_rods, method = "spearman")
```

```
##  
## Spearman's rank correlation rho  
##  
## data: datos_bacterias$vibrio_shaped and datos_bacterias$large_rods  
## S = 224, p-value = 2.994e-07  
## alternative hypothesis: true rho is not equal to 0  
## sample estimates:  
## rho  
## 0.8315789
```

La segunda opción es hacer una correlación de Pearson por permutaciones. Indicamos entonces el número de permutaciones a generar o dejamos el valor por defecto que es 999. Puede tardar un poco ya que tiene que permutar. El coeficiente de correlación obtenido es de 0.8310.

```
library(RVAideMemoire) #Si no encuentran esta librería en las computadoras del curso, j

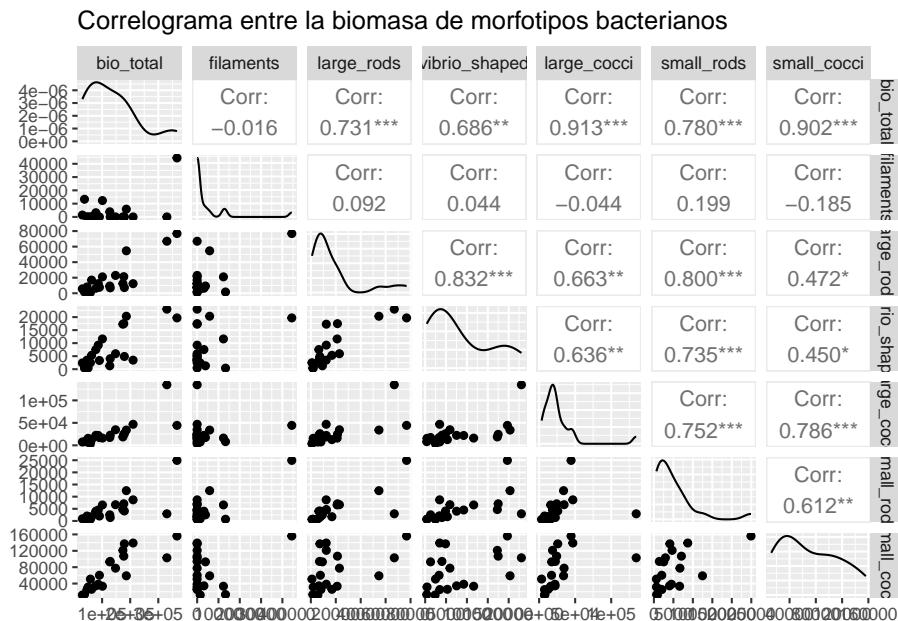
perm.cor.test(datos_bacterias$vibrio_shaped, datos_bacterias$large_rods, progress=FALSE)

## 
## Pearson's product-moment correlation - Permutation test
##
## data: datos_bacterias$vibrio_shaped and datos_bacterias$large_rods
## 999 permutations
## t = 6.3382, p-value = 0.002
## alternative hypothesis: true correlation is not equal to 0
## sample estimates:
##       cor
## 0.8310104
```

En el correlograma, vamos a cambiar qué coeficiente se muestra. Dejaremos el de Spearman. Observar que para algunos de los pares de variables hay diferencias entre los coeficientes de Pearson y Spearman.

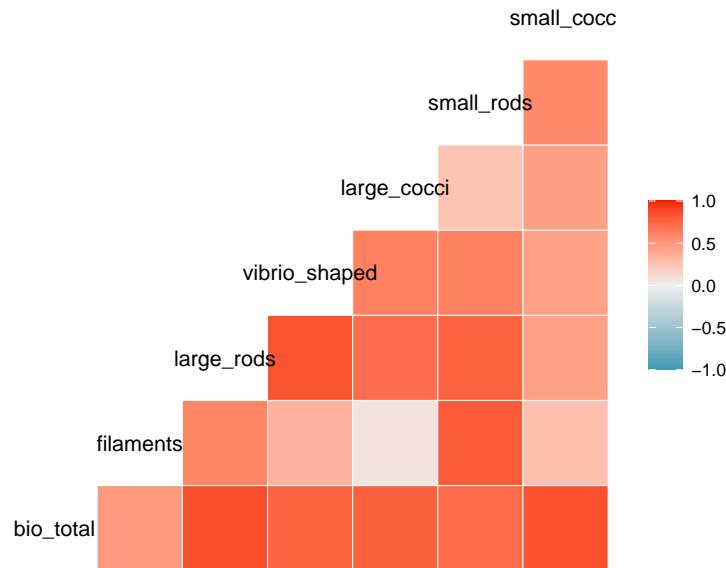
```
datos_bacterias <- datos_RH[,5:11]

ggpairs(datos_bacterias, title="Correlograma entre la biomasa de morfotipos bacterianos")
```

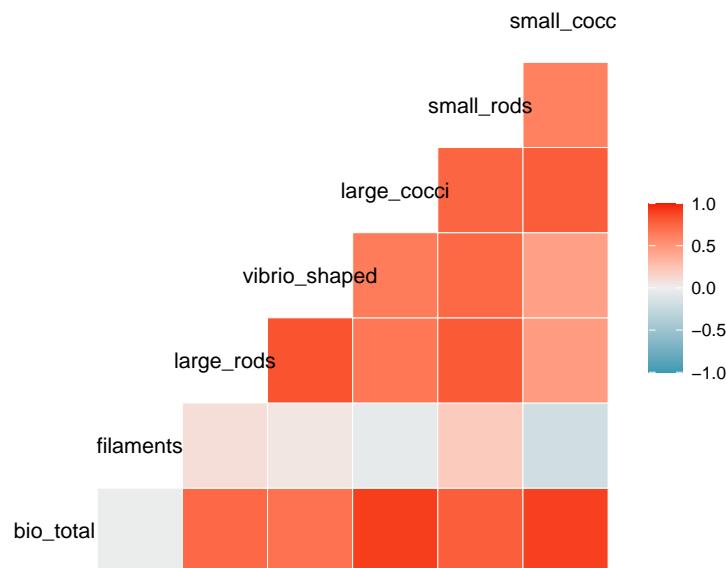


Ahora visualizaremos la correlación como un mapa de calor.

```
ggcorr(datos_bacterias, method = c("everything", "pearson"))
```



```
ggcorr(datos_bacterias, method = c("everything", "spearman"))
```



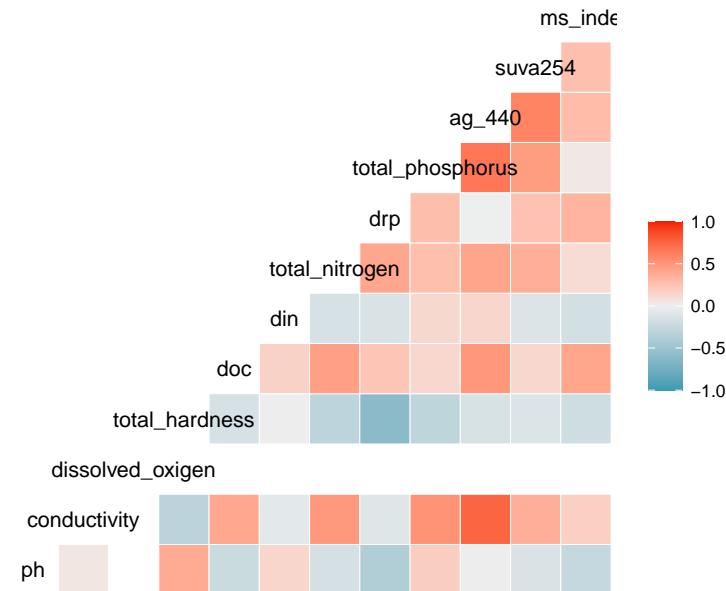
2.4.2 Entre variables abióticas

Ahora analizaremos las variables explicativas. Esto es útil ya que en el siguiente paso de realizar regresiones múltiples evitaremos colocar en el modelo dos variables muy correlacionadas entre sí, ya que son redundantes. Generamos un nuevo dataframe que es un subconjunto de los anteriores.

Podemos repetir el estudio de comparar correlaciones de Pearson, de Pearson por permutaciones y de Spearman. Aquí resumidamente pasaremos a los correlogramas visuales. ¿Qué pasa con el oxígeno disuelto? Tendremos cuidado con el uso de esta variable en los análisis siguientes.

```
datos_fq <- datos_RH[, 12:23]

ggcorr(datos_fq, method = c("everything", "spearman"))
```



Si queremos tener un resumen numérico de los coeficientes de correlación, podemos generar una matriz.

```
library(Hmisc)

## Loading required package: lattice

## Loading required package: survival
```

```

## Loading required package: Formula

##
## Attaching package: 'Hmisc'

## The following objects are masked from 'package:dplyr':
##
##     src, summarize

## The following objects are masked from 'package:base':
##
##     format.pval, units

rcorr(as.matrix(datos_fq), type="spearman")

##          ph conductivity dissolved_oxygen total_hardness   doc   din
## ph            1.00        0.05       -0.03        0.39 -0.22  0.14
## conductivity    0.05        1.00        0.06       -0.31  0.41 -0.07
## dissolved_oxygen -0.03        0.06        1.00       -0.34 -0.03 -0.29
## total_hardness    0.39       -0.31       -0.34        1.00 -0.16  0.01
## doc            -0.22        0.41       -0.03       -0.16  1.00  0.16
## din             0.14       -0.07       -0.29        0.01  0.16  1.00
## total_nitrogen   -0.16        0.48       -0.24       -0.30  0.45 -0.15
## drp            -0.38       -0.09       -0.15       -0.60  0.24 -0.13
## total_phosphorus  0.19        0.52       -0.22       -0.29  0.13  0.13
## ag_440           -0.01        0.75       -0.32       -0.14  0.49  0.14
## suva254          -0.13        0.36       -0.63       -0.12  0.13 -0.12
## ms_index         -0.25        0.17       -0.27       -0.20  0.41 -0.18
##          total_nitrogen    drp total_phosphorus ag_440 suva254 ms_index
## ph            -0.16 -0.38        0.19 -0.01 -0.13 -0.25
## conductivity    0.48 -0.09        0.52  0.75  0.36  0.17
## dissolved_oxygen -0.24 -0.15       -0.22 -0.32 -0.63 -0.27
## total_hardness    -0.30 -0.60       -0.29 -0.14 -0.12 -0.20
## doc            0.45  0.24        0.13  0.49  0.13  0.41
## din             -0.15 -0.13        0.13  0.14 -0.12 -0.18
## total_nitrogen    1.00  0.41        0.27  0.42  0.36  0.10
## drp            0.41  1.00        0.28  0.00  0.25  0.33
## total_phosphorus  0.27  0.28        1.00  0.66  0.46  0.03
## ag_440           0.42  0.00        0.66  1.00  0.59  0.29
## suva254          0.36  0.25        0.46  0.59  1.00  0.27
## ms_index         0.10  0.33        0.03  0.29  0.27  1.00
##
## n
##          ph conductivity dissolved_oxygen total_hardness doc din

```

```

## ph          20          20          15          20  20  20
## conductivity 20          20          15          20  20  20
## dissolved_oxygen 15          15          15          15  15  15
## total_hardness 20          20          15          20  20  20
## doc          20          20          15          20  20  20
## din          20          20          15          20  20  20
## total_nitrogen 20          20          15          20  20  20
## drp          20          20          15          20  20  20
## total_phosphorus 20          20          15          20  20  20
## ag_440        20          20          15          20  20  20
## suva254       20          20          15          20  20  20
## ms_index      20          20          15          20  20  20
##               total_nitrogen drp total_phosphorus ag_440 suva254 ms_index
## ph            20  20          20  20          20  20
## conductivity 20  20          20  20          20  20
## dissolved_oxygen 15  15          15  15          15  15
## total_hardness 20  20          20  20          20  20
## doc            20  20          20  20          20  20
## din            20  20          20  20          20  20
## total_nitrogen 20  20          20  20          20  20
## drp            20  20          20  20          20  20
## total_phosphorus 20  20          20  20          20  20
## ag_440         20  20          20  20          20  20
## suva254        20  20          20  20          20  20
## ms_index       20  20          20  20          20  20
##
## P
##               ph      conductivity dissolved_oxygen total_hardness doc
## ph            0.8450     0.9094          0.0875    0.3417
## conductivity  0.8450           0.8292          0.1788    0.0760
## dissolved_oxygen 0.9094  0.8292           0.2182    0.9142
## total_hardness  0.0875  0.1788          0.2182    0.5120
## doc            0.3417  0.0760          0.9142    0.5120
## din            0.5541  0.7813          0.2979    0.9824    0.4934
## total_nitrogen 0.5036  0.0318          0.3973    0.1926    0.0472
## drp            0.1009  0.6939          0.5901    0.0048    0.3147
## total_phosphorus 0.4266  0.0192          0.4402    0.2116    0.5765
## ag_440         0.9699  0.0001          0.2474    0.5434    0.0271
## suva254        0.5801  0.1200          0.0126    0.6090    0.5735
## ms_index       0.2850  0.4631          0.3369    0.3975    0.0714
##               din      total_nitrogen drp      total_phosphorus ag_440 suva254
## ph            0.5541  0.5036          0.1009  0.4266    0.9699  0.5801
## conductivity  0.7813  0.0318          0.6939  0.0192    0.0001  0.1200
## dissolved_oxygen 0.2979  0.3973          0.5901  0.4402    0.2474  0.0126
## total_hardness  0.9824  0.1926          0.0048  0.2116    0.5434  0.6090
## doc            0.4934  0.0472          0.3147  0.5765    0.0271  0.5735

```

```

## din           0.5348      0.5760 0.5876      0.5560 0.6267
## total_nitrogen 0.5348      0.0740 0.2464      0.0675 0.1193
## drp          0.5760 0.0740                  0.2333      0.9873 0.2787
## total_phosphorus 0.5876 0.2464      0.2333      0.0015 0.0393
## ag_440        0.5560 0.0675      0.9873 0.0015      0.0057
## suva254       0.6267 0.1193      0.2787 0.0393      0.0057
## ms_index      0.4556 0.6686      0.1489 0.8907      0.2144 0.2494
##               ms_index
## ph            0.2850
## conductivity 0.4631
## dissolved_oxygen 0.3369
## total_hardness 0.3975
## doc           0.0714
## din           0.4556
## total_nitrogen 0.6686
## drp           0.1489
## total_phosphorus 0.8907
## ag_440        0.2144
## suva254       0.2494
## ms_index

```

2.5 Regresiones múltiples

Ahora vamos a buscar un modelo que explique la variabilidad de la biomasa total de morfotipos bacterianos en función de una o más variables abióticas medidas en los cuerpos de agua. Para ello, es útil todo el conocimiento que contamos sobre estas variables a partir de los análisis exploratorios previos, incluyendo las relaciones entre variables explicativas.

Un **modelo de regresión múltiple** puede expresarse: $y = \theta + 1.x_1 + 2.x_2 + \dots + n.x_n + \epsilon$, donde y es la variable respuesta; θ es la ordenada al origen; x_i son las variables explicativas, cada una con un coeficiente estimado i ; y ϵ es el error.

¿Hay que estandarizar las variables explicativas? Podemos notar que las variables explicativas están en distintas escalas, por ejemplo con ‘summary(datos_fq)’ vemos que ph varía entre 4.65 y 7.1, mientras que $total_nitrogen$ varía entre 1320 y 11330. No es estrictamente necesario estandarizar las variables. En caso de no estandarizar, hay que tener cuidado con la interpretación de los coeficientes i : si en un modelo final el coeficiente asociado al nitrógeno total es mayor al del pH, no podemos afirmar que el impacto del nitrógeno total sobre la biomasa bacteriana es mayor al del pH. Vamos a ver un ejemplo de interpretación del modelo final más adelante.

Vamos a trabajar con la tabla de datos *datos_RH*. Estandarizamos y centramos los datos para facilitar la interpretación (y luego comparemos los resultados

que se obtienen con las variables originales). La función *scale* permite centrar y/o escalar una o más columnas de una tabla de datos. Por defecto hace las dos operaciones, pero podemos solicitar que sólo centre ó sólo escale los datos. La operación de centrar significa que para cada columna se calcula la media (omitiendo NAs) y luego a cada valor se le resta la media de esa columna. La operación de escalar significa que el valor de cada celda se divide por la desviación estándar. El siguiente script es para analizar los valores originales de la variable pH, los valores centrados y los valores centrados y escalados.

```
datos_RH[,12]
```

```
## # A tibble: 20 x 1
##       ph
##   <dbl>
## 1  4.91
## 2  5.52
## 3  4.72
## 4  5.09
## 5  4.82
## 6  6.39
## 7  4.75
## 8  4.67
## 9  6.75
## 10 4.65
## 11 5.9
## 12 5.19
## 13 4.66
## 14 6.65
## 15 4.82
## 16 7.1
## 17 4.88
## 18 5.4
## 19 6.2
## 20 5.43
```

```
scale(datos_RH[,12], center = TRUE, scale = FALSE)
```

```
##       ph
## [1,] -0.515
## [2,]  0.095
## [3,] -0.705
## [4,] -0.335
## [5,] -0.605
## [6,]  0.965
```

```

## [7,] -0.675
## [8,] -0.755
## [9,]  1.325
## [10,] -0.775
## [11,]  0.475
## [12,] -0.235
## [13,] -0.765
## [14,]  1.225
## [15,] -0.605
## [16,]  1.675
## [17,] -0.545
## [18,] -0.025
## [19,]  0.775
## [20,]  0.005
## attr(,"scaled:center")
##     ph
## 5.425

scale(datos_RH[,12], center = TRUE, scale = TRUE) #estos son los parámetros por defecto de la función scale

##                  ph
## [1,] -0.648046357
## [2,]  0.119542532
## [3,] -0.887131421
## [4,] -0.421544718
## [5,] -0.761297177
## [6,]  1.214300456
## [7,] -0.849381148
## [8,] -0.950048543
## [9,]  1.667303735
## [10,] -0.975215392
## [11,]  0.597712660
## [12,] -0.295710474
## [13,] -0.962631968
## [14,]  1.541469491
## [15,] -0.761297177
## [16,]  2.107723589
## [17,] -0.685796631
## [18,] -0.031458561
## [19,]  0.975215392
## [20,]  0.006291712
## attr(,"scaled:center")
##     ph
## 5.425
## attr(,"scaled:scale")

```

```
##          ph
## 0.7946962
```

Ahora vamos a estandarizar todas las variables explicativas.

```
datos_RH_est <- datos_RH
datos_RH_est[, 12:23] <- scale(datos_RH[, 12:23])
datos_RH_est
```

```
## # A tibble: 20 x 23
##   id    pool site      date bio_t~1 filam~2 large~3 vibri~4 large~5 small~6
##   <chr> <fct> <chr>     <fct> <dbl>   <dbl>   <dbl>   <dbl>   <dbl>   <dbl>
## 1 10   RH1   shore   Octo~  62653     0  16921   5315  12676  1747
## 2 20   RH2   shore   Octo~  29808   1435   5645  2384  7381   888
## 3 30   RH3   shore   Octo~  37357  13307   1597   392   8131   708
## 4 40   RH4   north shore Octo~  46499     0  8085   3499  7534  2028
## 5 50   RH5   shore   Octo~  50321     0  1921   451   15662  666
## 6 1D   RH1   shore   Dece~  86432     0  12374  9228  21906 4503
## 7 2D   RH2   shore   Dece~ 147788    0  22924  5933  34663 6687
## 8 3D   RH3   shore   Dece~ 210287    0  12309  3471  46807 8694
## 9 4D   RH4   north shore Dece~ 366384  44419  76819 19692 44391 24931
## 10 5D  RH5   shore   Dece~ 330544    0  66952  23046 134707 2949
## 11 1F  RH1   shore   Febr~  77850   3177   6414  7533  22218 4013
## 12 2F  RH2   shore   Febr~ 179175    0  7263   4879  25886 4098
## 13 3F  RH3   shore   Febr~ 127395   3886   9283  1294  17402 2522
## 14 4F  RH4   north shore Febr~ 186223   5860  54624 20331 33949 12469
## 15 5F  RH5   shore   Febr~ 177152    0  21324  17482 24397 7064
## 16 1A  RH1   shore   Apri~ 101197  12331  21098 11615 16083 6592
## 17 2A  RH2   shore   Apri~ 128935    0  6950   4028  22525 1395
## 18 3A  RH3   shore   Apri~ 174051    0  12394  17281 18608 4678
## 19 4A  RH4   north shore Apri~ 58439     0  901   2484  3455   642
## 20 5A  RH5   shore   Apri~ 90498     0  7626  3338  17231 2016
## # ... with 13 more variables: small_cocci <dbl>, ph <dbl>, conductivity <dbl>,
## # dissolved_oxygen <dbl>, total_hardness <dbl>, doc <dbl>, din <dbl>,
## # total_nitrogen <dbl>, drp <dbl>, total_phosphorus <dbl>, ag_440 <dbl>,
## # suva254 <dbl>, ms_index <dbl>, and abbreviated variable names 1: bio_total,
## # 2: filaments, 3: large_rods, 4: vibrio_shaped, 5: large_cocci,
## # 6: small_rods
```

Nota. Uno de los supuestos del modelo de regresión múltiple es que las observaciones son independientes entre sí. Dado que cada laguna fue medida repetidamente en cuatro fechas distintas, es muy posible que las cuatro observaciones de estas lagunas no sean totalmente independientes. A los efectos de esta actividad práctica, vamos a suponer que sí hay independencia entre cada una de las observaciones. Para el futuro, recomendamos explorar análisis que

permitan incluir a la identidad de la laguna como factor aleatorio (modelos lineales generalizados mixtos) y/o considerar las medidas repetidas. Todo lo que veamos a continuación sobre selección de variables es aplicable a otros modelos de regresión más complejos.

2.5.1 Selección automática de variables

Contamos con 12 posibles variables explicativas continuas, más la fecha de muestreo. Un buen modelo de regresión múltiple tendrá un compromiso entre el poder explicativo y la cantidad de variables a incluir. Tengamos en cuenta que para cada variable explicativa tenemos que estimar un coeficiente. Además, un modelo muy complejo puede sobre-ajustar a nuestros datos e implica que, para luego predecir la variable respuesta, necesitaremos conocer el valor de muchas variables, algunas quizás difíciles de obtener a campo. Queremos obtener un modelo con buen poder explicativo pero lo más sencillo posible. Además sólo tenemos 20 observaciones, ¿sería lógico explicarlas con 12 o 13 variables? Y por último, ya vimos que las variables están correlacionadas, por lo cual hay redundancia, y que para algunas variables hay NAs.

Hay dos métodos de selección automática de variables. Luego, veremos cómo hacer una selección manual, lo cual muchas veces nos permite obtener mejores resultados o conocer mejor

2.5.2 Modelos nulo y completo

En primer lugar, construimos un **modelo nulo**, es decir, sin variables explicativas. Usamos la notación de R de “fórmula” (ver la hoja de referencia Comparación de sintaxis de R ya citada). La virgulilla o símbolo `~` expresa que una variable está en función de otras. En el caso del modelo nulo, indicamos con `1` que no hay variables explicativas. Vamos a trabajar con los datos estandarizados. La función `lm` significa *linear model* y es la que usaremos para ajustar las regresiones múltiples.

```
modelo_nulo <- lm(bio_total ~ 1, data = datos_RH_est)
summary(modelo_nulo)

##
## Call:
## lm(formula = bio_total ~ 1, data = datos_RH_est)
##
## Residuals:
##      Min       1Q   Median       3Q      Max 
## -103641  -71850  -19153   44208  232935 
##
```

```

## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) 133449     20582   6.484 3.26e-06 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 92050 on 19 degrees of freedom

```

En este modelo nulo, sólo contamos con un coeficiente estimado para la ordenada al origen (en inglés, *intercept*, que es significativamente distinta de cero), tenemos 19 grados de libertad y un error residual de 92050.

Para la construcción del **modelo completo**, es decir, con todas las variables explicativas, vamos a usar todas las variables abióticas excepto el oxígeno disuelto que cuenta con varios NA y la fecha de muestreo.

```

modelo_completo <- lm(bio_total ~ date + ph + conductivity + total_hardness + doc + d
summary(modelo_completo)

```

```

##
## Call:
## lm(formula = bio_total ~ date + ph + conductivity + total_hardness +
##     doc + din + total_nitrogen + drp + total_phosphorus + ag_440 +
##     suva254 + ms_index, data = datos_RH_est)
##
## Residuals:
##      1       2       3       4       5       6       7       8
## -16862.4 15891.9 4256.6 -12144.1 8858.0 -21518.8 8125.8 -18328.7
##      9      10      11      12      13      14      15      16
## -3907.0 35628.7 -2249.8 -7961.8 -14643.3 3655.2 21199.7 10598.8
##      17      18      19      20
## -32983.0 2219.6 -170.4 20334.9
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) 181516     71287   2.546  0.0515 .
## dateDecember_2009 34544     83141   0.415  0.6950
## dateFebruary_2010 -108501    106972  -1.014  0.3570
## dateApril_2010    -118308    101025  -1.171  0.2943
## ph            2814     17824   0.158  0.8807
## conductivity    73801    28259   2.612  0.0476 *
## total_hardness   -15060    15497  -0.972  0.3758
## doc           -11469    17019  -0.674  0.5303
## din            15408    12567   1.226  0.2748
## total_nitrogen   61310    19326   3.172  0.0247 *

```

```

## drp           -31077    20978  -1.481   0.1986
## total_phosphorus 14382    21719   0.662   0.5371
## ag_440        -51591    31988  -1.613   0.1677
## suva254        17420    15006   1.161   0.2981
## ms_index       -4105     15676  -0.262   0.8039
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 32540 on 5 degrees of freedom
## Multiple R-squared:  0.9671, Adjusted R-squared:  0.875
## F-statistic: 10.5 on 14 and 5 DF,  p-value: 0.008489

```

Observemos que el modelo completo tiene un R^2 ajustado de 87,5%, con 5 grados de libertad (más adelante interpretaremos estos resultados).

2.5.2.1 Selección secuencial *Forward*: del modelo nulo, sumando variables explicativas

La regresión Forward inicia con un modelo nulo y en cada paso agrega una variable explicativa significativa hasta llegar al modelo final. La función *step* en primer lugar pide que le indiquemos el modelo nulo. Luego, en scope, podemos indicar con una fórmula el modelo final máximo al que podemos llegar o (como en este ejemplo) un modelo mínimo *lower* y un modelo máximo *upper* (el nulo y el completo, respectivamente). Luego tenemos que indicar en qué dirección se construye el modelo, en este caso es *forward*.

El criterio que se utiliza para agregar o no una variable es el **AIC (Akaike Information Criterion)**. A partir del modelo nulo, se prueba de agregar todas las variables y se analiza si esa variable reduce el valor de AIC se reduce. Si al agregar una variable el AIC no se reduce, esto significa que el modelo más simple es mejor y la selección se detiene. Podemos correr la siguiente función y analizar los resultados.

```

modelo_forward <- step(modelo_nulo, scope = list(lower=modelo_nulo, upper=modelo_completo), direc
## Start: AIC=458.18
## bio_total ~ 1
##
##              Df  Sum of Sq      RSS      AIC
## + date          3 8.7701e+10 7.3277e+10 448.44
## + conductivity  1 4.8091e+10 1.1289e+11 453.08
## + total_nitrogen 1 4.5911e+10 1.1507e+11 453.46
## <none>                   1.6098e+11 458.18
## + ms_index       1 1.3770e+10 1.4721e+11 458.39
## + din            1 1.2006e+10 1.4897e+11 458.63

```

```

## + doc           1 1.1071e+10 1.4991e+11 458.75
## + total_hardness 1 9.3550e+09 1.5162e+11 458.98
## + drp          1 1.8187e+09 1.5916e+11 459.95
## + ph           1 1.0336e+09 1.5994e+11 460.05
## + total_phosphorus 1 4.4439e+08 1.6053e+11 460.12
## + ag_440        1 3.4721e+08 1.6063e+11 460.13
## + suva254       1 1.0572e+08 1.6087e+11 460.16
##
## Step: AIC=448.44
## bio_total ~ date
##
## Df  Sum of Sq      RSS     AIC
## + total_nitrogen 1 3.0178e+10 4.3098e+10 439.82
## + ms_index        1 2.5308e+10 4.7968e+10 441.96
## + total_hardness 1 2.0956e+10 5.2321e+10 443.70
## + conductivity   1 9.0729e+09 6.4204e+10 447.79
## <none>            7.3277e+10 448.44
## + total_phosphorus 1 2.7961e+09 7.0481e+10 449.66
## + ag_440          1 2.3777e+09 7.0899e+10 449.78
## + suva254         1 1.9938e+09 7.1283e+10 449.88
## + drp             1 1.6684e+09 7.1608e+10 449.97
## + doc             1 6.5824e+08 7.2618e+10 450.26
## + din             1 2.9814e+08 7.2979e+10 450.35
## + ph              1 1.3244e+08 7.3144e+10 450.40
##
## Step: AIC=439.82
## bio_total ~ date + total_nitrogen
##
## Df  Sum of Sq      RSS     AIC
## + ms_index        1 1.5933e+10 2.7165e+10 432.59
## + conductivity   1 1.3897e+10 2.9202e+10 434.04
## + total_hardness 1 4.9749e+09 3.8123e+10 439.37
## + doc             1 4.9533e+09 3.8145e+10 439.38
## <none>            4.3098e+10 439.82
## + total_phosphorus 1 2.1160e+09 4.0982e+10 440.81
## + drp             1 1.7711e+09 4.1327e+10 440.98
## + ph              1 1.5312e+09 4.1567e+10 441.10
## + suva254         1 1.1274e+09 4.1971e+10 441.29
## + din             1 9.4483e+08 4.2153e+10 441.38
## + ag_440          1 8.5977e+07 4.3012e+10 441.78
##
## Step: AIC=432.59
## bio_total ~ date + total_nitrogen + ms_index
##
## Df  Sum of Sq      RSS     AIC
## + conductivity   1 1.1302e+10 1.5863e+10 423.83

```

```

## + total_hardness    1 6.2126e+09 2.0953e+10 429.40
## + suva254          1 3.8600e+09 2.3305e+10 431.52
## <none>              2.7165e+10 432.59
## + total_phosphorus 1 2.4124e+09 2.4753e+10 432.73
## + drp               1 1.0825e+09 2.6083e+10 433.78
## + din               1 7.2970e+08 2.6435e+10 434.04
## + doc               1 7.2857e+08 2.6437e+10 434.05
## + ag_440             1 2.5952e+08 2.6906e+10 434.40
## + ph                1 7.3627e+07 2.7092e+10 434.54
##
## Step: AIC=423.83
## bio_total ~ date + total_nitrogen + ms_index + conductivity
##
##                               Df Sum of Sq      RSS      AIC
## + doc                  1 1665616714 1.4197e+10 423.61
## <none>                 1.5863e+10 423.83
## + ag_440               1 1509519938 1.4353e+10 423.83
## + suva254              1 1457149002 1.4406e+10 423.90
## + ph                  1 928283403 1.4935e+10 424.62
## + din                  1 206546528 1.5656e+10 425.57
## + total_hardness       1 95043074 1.5768e+10 425.71
## + total_phosphorus     1 21598302 1.5841e+10 425.80
## + drp                  1 7603750 1.5855e+10 425.82
##
## Step: AIC=423.61
## bio_total ~ date + total_nitrogen + ms_index + conductivity +
##   doc
##
##                               Df Sum of Sq      RSS      AIC
## + din                  1 3065337891 1.1132e+10 420.75
## <none>                 1.4197e+10 423.61
## + suva254              1 711689288 1.3486e+10 424.58
## + ag_440               1 582841690 1.3614e+10 424.77
## + drp                  1 490860741 1.3706e+10 424.91
## + ph                  1 305626182 1.3892e+10 425.18
## + total_hardness       1 23984669 1.4173e+10 425.58
## + total_phosphorus     1 10218359 1.4187e+10 425.60
##
## Step: AIC=420.75
## bio_total ~ date + total_nitrogen + ms_index + conductivity +
##   doc + din
##
##                               Df Sum of Sq      RSS      AIC
## <none>                 1.1132e+10 420.75
## + ag_440               1 861844423 1.0270e+10 421.14
## + suva254              1 856008929 1.0276e+10 421.15

```

```
## + drp           1 102714121 1.1029e+10 422.56
## + total_phosphorus 1 65934861 1.1066e+10 422.63
## + ph            1 25518439 1.1106e+10 422.70
## + total_hardness 1 14969693 1.1117e+10 422.72
```

```
summary(modelo_forward)
```

```
##
## Call:
## lm(formula = bio_total ~ date + total_nitrogen + ms_index + conductivity +
##     doc + din, data = datos_RH_est)
##
## Residuals:
##   Min     1Q Median     3Q    Max
## -43732 -15210  -1318    9721   56450
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) 161030     34616    4.652 0.000703 ***
## dateDecember_2009 82352     44747    1.840 0.092823 .
## dateFebruary_2010 -83583     48224   -1.733 0.110955
## dateApril_2010   -109091    48920   -2.230 0.047528 *
## total_nitrogen    64465     11963    5.389 0.000220 ***
## ms_index          -19291     8911   -2.165 0.053256 .
## conductivity      59452     17532    3.391 0.006024 **
## doc              -27533     13022   -2.114 0.058129 .
## din               19263     11068    1.740 0.109649
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 31810 on 11 degrees of freedom
## Multiple R-squared:  0.9308, Adjusted R-squared:  0.8806
## F-statistic: 18.51 on 8 and 11 DF,  p-value: 2.397e-05
```

El modelo completo elegido por el método automático Forward incluye a las variables: date (notar que al ser una variable categórica con cuatro niveles se generan tres variables dummy) y a cinco variables abióticas. De ellas, algunas no tienen un efecto significativo según el p-valor a pesar de haber reducido el AIC (veremos cómo mejorar este modelo con el método de selección manual). El R^2 ajustado es de 88,06%, con 11 grados de libertad.

2.5.2.2 Selección secuencial *Backward*: del modelo completo, quitando variables explicativas

La regresión Backward inicia con un modelo completo y en cada paso va quitando la variable explicativa menos significativa. Modificamos la fórmula del modelo Forward: ahora iniciamos con el modelo completo.

```
modelo_backward <- step(modelo_completo, scope = list(lower=modelo_nulo, upper=modelo_completo),
```

```
## Start: AIC=417.89
## bio_total ~ date + ph + conductivity + total_hardness + doc +
##      din + total_nitrogen + drp + total_phosphorus + ag_440 +
##      suva254 + ms_index
##
##              Df  Sum of Sq      RSS      AIC
## - ph          1 2.6397e+07 5.3214e+09 415.99
## - ms_index    1 7.2630e+07 5.3676e+09 416.16
## - total_phosphorus 1 4.6437e+08 5.7594e+09 417.57
## - doc         1 4.8093e+08 5.7759e+09 417.62
## <none>           5.2950e+09 417.89
## - total_hardness 1 1.0002e+09 6.2951e+09 419.35
## - suva254     1 1.4271e+09 6.7221e+09 420.66
## - din          1 1.5919e+09 6.8869e+09 421.14
## - drp          1 2.3241e+09 7.6191e+09 423.16
## - ag_440       1 2.7546e+09 8.0496e+09 424.26
## - conductivity 1 7.2228e+09 1.2518e+10 433.09
## - date         3 1.3281e+10 1.8576e+10 436.99
## - total_nitrogen 1 1.0658e+10 1.5953e+10 437.94
##
## Step: AIC=415.99
## bio_total ~ date + conductivity + total_hardness + doc + din +
##      total_nitrogen + drp + total_phosphorus + ag_440 + suva254 +
##      ms_index
##
##              Df  Sum of Sq      RSS      AIC
## - ms_index    1 6.6515e+07 5.3879e+09 414.23
## <none>           5.3214e+09 415.99
## - doc          1 7.3119e+08 6.0526e+09 416.56
## - total_hardness 1 1.0013e+09 6.3227e+09 417.43
## - total_phosphorus 1 1.0863e+09 6.4077e+09 417.70
## - din          1 1.8299e+09 7.1513e+09 419.90
## - suva254     1 1.8543e+09 7.1757e+09 419.96
## - drp          1 2.4459e+09 7.7673e+09 421.55
## - ag_440       1 4.4059e+09 9.7272e+09 426.05
## - conductivity 1 7.4832e+09 1.2805e+10 431.55
```

```

## - total_nitrogen      1 1.1671e+10 1.6993e+10 437.21
## - date                  3 1.8187e+10 2.3508e+10 439.70
##
## Step: AIC=414.23
## bio_total ~ date + conductivity + total_hardness + doc + din +
##   total_nitrogen + drp + total_phosphorus + ag_440 + suva254
##
##                               Df  Sum of Sq      RSS      AIC
## <none>                      5.3879e+09 414.23
## - doc                     1 8.5179e+08 6.2397e+09 415.17
## - total_hardness          1 1.3588e+09 6.7467e+09 416.73
## - din                     1 1.7839e+09 7.1718e+09 417.95
## - suva254                 1 1.7935e+09 7.1814e+09 417.98
## - total_phosphorus         1 1.8814e+09 7.2693e+09 418.22
## - ag_440                   1 7.5077e+09 1.2896e+10 429.69
## - drp                      1 8.0232e+09 1.3411e+10 430.47
## - conductivity             1 8.6662e+09 1.4054e+10 431.41
## - date                     3 1.8637e+10 2.4024e+10 438.13
## - total_nitrogen            1 1.5758e+10 2.1145e+10 439.58

summary(modelo_backward)

##
## Call:
## lm(formula = bio_total ~ date + conductivity + total_hardness +
##   doc + din + total_nitrogen + drp + total_phosphorus + ag_440 +
##   suva254, data = datos_RH_est)
##
## Residuals:
##    Min     1Q Median     3Q    Max 
## -35553 -11521   1893  12213  34533 
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)    
## (Intercept) 179438     44004   4.078  0.00470 **  
## dateDecember_2009 33089     54723   0.605  0.56447    
## dateFebruary_2010 -103636    67833  -1.528  0.17040    
## dateApril_2010   -113409    58328  -1.944  0.09294 .    
## conductivity     74699     22262   3.355  0.01216 *   
## total_hardness   -16434     12369  -1.329  0.22563    
## doc              -13408     12745  -1.052  0.32777    
## din               15674     10295   1.522  0.17173    
## total_nitrogen    62335     13777   4.525  0.00272 **  
## drp              -35621     11033  -3.229  0.01448 *   
## total_phosphorus  18749     11992   1.563  0.16193    

```

```

## ag_440           -58207      18637   -3.123  0.01677 *
## suva254          15282      10011    1.526  0.17073
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 27740 on 7 degrees of freedom
## Multiple R-squared:  0.9665, Adjusted R-squared:  0.9092
## F-statistic: 16.85 on 12 and 7 DF,  p-value: 0.000514

```

El mejor modelo según el criterio AIC incluye la fecha *date* y nueve variables abióticas, algunas no significativas según el p-valor. El R^2 ajustado es de 90,92%, con 7 grados de libertad.

2.5.2.3 Selección *Stepwise*: combinación secuencial de Forward y Backward

La regresión Stepwise inicia con un modelo nulo, en cada paso agrega una variable explicativa significativa (similar a Forward), pero revisa el modelo y si hay una variable no significativa la quita y repite el paso.

```

modelo_stepwise <- step(modelo_nulo, scope = list(lower=modelo_nulo, upper=modelo_completo), direc

```

	Df	Sum of Sq	RSS	AIC
## Start: AIC=458.18				
## bio_total ~ 1				
##				
##	Df	Sum of Sq	RSS	AIC
## + date	3	8.7701e+10	7.3277e+10	448.44
## + conductivity	1	4.8091e+10	1.1289e+11	453.08
## + total_nitrogen	1	4.5911e+10	1.1507e+11	453.46
## <none>		1.6098e+11	458.18	
## + ms_index	1	1.3770e+10	1.4721e+11	458.39
## + din	1	1.2006e+10	1.4897e+11	458.63
## + doc	1	1.1071e+10	1.4991e+11	458.75
## + total_hardness	1	9.3550e+09	1.5162e+11	458.98
## + drp	1	1.8187e+09	1.5916e+11	459.95
## + ph	1	1.0336e+09	1.5994e+11	460.05
## + total_phosphorus	1	4.4439e+08	1.6053e+11	460.12
## + ag_440	1	3.4721e+08	1.6063e+11	460.13
## + suva254	1	1.0572e+08	1.6087e+11	460.16
##				
## Step: AIC=448.44				
## bio_total ~ date				
##	Df	Sum of Sq	RSS	AIC

```

## + total_nitrogen      1 3.0178e+10 4.3098e+10 439.82
## + ms_index            1 2.5308e+10 4.7968e+10 441.96
## + total_hardness       1 2.0956e+10 5.2321e+10 443.70
## + conductivity         1 9.0729e+09 6.4204e+10 447.79
## <none>                7.3277e+10 448.44
## + total_phosphorus     1 2.7961e+09 7.0481e+10 449.66
## + ag_440               1 2.3777e+09 7.0899e+10 449.78
## + suva254              1 1.9938e+09 7.1283e+10 449.88
## + drp                  1 1.6684e+09 7.1608e+10 449.97
## + doc                  1 6.5824e+08 7.2618e+10 450.26
## + din                  1 2.9814e+08 7.2979e+10 450.35
## + ph                   1 1.3244e+08 7.3144e+10 450.40
## - date                 3 8.7701e+10 1.6098e+11 458.18
##
## Step: AIC=439.82
## bio_total ~ date + total_nitrogen
##
##                               Df  Sum of Sq      RSS      AIC
## + ms_index            1 1.5933e+10 2.7165e+10 432.59
## + conductivity        1 1.3897e+10 2.9202e+10 434.04
## + total_hardness       1 4.9749e+09 3.8123e+10 439.37
## + doc                 1 4.9533e+09 3.8145e+10 439.38
## <none>                4.3098e+10 439.82
## + total_phosphorus    1 2.1160e+09 4.0982e+10 440.81
## + drp                 1 1.7711e+09 4.1327e+10 440.98
## + ph                  1 1.5312e+09 4.1567e+10 441.10
## + suva254             1 1.1274e+09 4.1971e+10 441.29
## + din                 1 9.4483e+08 4.2153e+10 441.38
## + ag_440              1 8.5977e+07 4.3012e+10 441.78
## - total_nitrogen       1 3.0178e+10 7.3277e+10 448.44
## - date                3 7.1968e+10 1.1507e+11 453.46
##
## Step: AIC=432.59
## bio_total ~ date + total_nitrogen + ms_index
##
##                               Df  Sum of Sq      RSS      AIC
## + conductivity         1 1.1302e+10 1.5863e+10 423.83
## + total_hardness        1 6.2126e+09 2.0953e+10 429.40
## + suva254              1 3.8600e+09 2.3305e+10 431.52
## <none>                2.7165e+10 432.59
## + total_phosphorus     1 2.4124e+09 2.4753e+10 432.73
## + drp                  1 1.0825e+09 2.6083e+10 433.78
## + din                  1 7.2970e+08 2.6435e+10 434.04
## + doc                  1 7.2857e+08 2.6437e+10 434.05
## + ag_440               1 2.5952e+08 2.6906e+10 434.40
## + ph                   1 7.3627e+07 2.7092e+10 434.54

```



```

## <none>          1.1132e+10 420.75
## + ag_440        1 8.6184e+08 1.0270e+10 421.14
## + suva254       1 8.5601e+08 1.0276e+10 421.15
## + drp           1 1.0271e+08 1.1029e+10 422.56
## + total_phosphorus 1 6.5935e+07 1.1066e+10 422.63
## + ph            1 2.5518e+07 1.1106e+10 422.70
## + total_hardness 1 1.4970e+07 1.1117e+10 422.72
## - din           1 3.0653e+09 1.4197e+10 423.61
## - doc            1 4.5244e+09 1.5656e+10 425.57
## - ms_index       1 4.7426e+09 1.5875e+10 425.84
## - conductivity   1 1.1637e+10 2.2769e+10 433.06
## - total_nitrogen 1 2.9385e+10 4.0517e+10 444.59
## - date          3 5.9982e+10 7.1114e+10 451.84

summary(modelo_stepwise)

##
## Call:
## lm(formula = bio_total ~ date + total_nitrogen + ms_index + conductivity +
##     doc + din, data = datos_RH_est)
##
## Residuals:
##    Min      1Q Median      3Q      Max
## -43732 -15210 -1318   9721  56450
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) 161030     34616   4.652 0.000703 ***
## dateDecember_2009 82352     44747   1.840 0.092823 .
## dateFebruary_2010 -83583     48224  -1.733 0.110955
## dateApril_2010    -109091    48920  -2.230 0.047528 *
## total_nitrogen    64465     11963   5.389 0.000220 ***
## ms_index         -19291     8911  -2.165 0.053256 .
## conductivity     59452     17532   3.391 0.006024 **
## doc              -27533     13022  -2.114 0.058129 .
## din              19263     11068   1.740 0.109649
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 31810 on 11 degrees of freedom
## Multiple R-squared:  0.9308, Adjusted R-squared:  0.8806
## F-statistic: 18.51 on 8 and 11 DF,  p-value: 2.397e-05

```

Comparen el modelo final según este método con los obtenidos con los métodos Forward y Backward.

2.5.3 Selección manual de variables

Vamos a realizar una selección manual similar a Forward: partimos de un modelo nulo y agregamos variables de a una. Los criterios para agregar o no una variable serán:

- AIC: al agregar una variable, el AIC del modelo se reduce en 2 o más respecto al AIC del modelo más sencillo.
- Significancia: el efecto de la variable es significativo, con un nivel de significancia del 5% ($p < 0.05$).
- Redundancia: evitaremos agregar variables muy correlacionadas entre sí.

Algunas razones por las cuales es recomendable hacer una selección manual: a) entenderemos mejor nuestros modelos; b) podemos ajustar uno o más modelos; c) podemos incluir más fácilmente las interacciones entre variables; d) podemos tener en cuenta múltiples criterios para elegir si agregamos o no variables al modelo; e) vamos a incluir consideraciones sobre la redundancia de variables, según lo que vimos en los análisis de regresiones múltiples.

2.5.3.1 Modelos univariados

En primer lugar, ajustamos todos los modelos univariados y los comparamos con el modelo nulo. El AIC del modelo nulo es:

```
AIC(modelo_nulo)
```

```
## [1] 516.9335
```

Corremos todos los modelos nulos. En este caso no los guardamos (pero podríamos hacerlo), sólo buscamos el resumen.

```
summary(lm(bio_total ~ date, data = datos_RH_est))

##
## Call:
## lm(formula = bio_total ~ date, data = datos_RH_est)
##
## Residuals:
##      Min       1Q   Median       3Q      Max 
## -141855  -20636   -3400   28099  138097 
## 
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)    
## (Intercept)  141855    138097     1.03    0.305    
## date        -20636     28099    -0.73    0.465    
##
```

```

## (Intercept)      45328      30265    1.498 0.153679
## dateDecember_2009 182959      42801    4.275 0.000581 ***
## dateFebruary_2010 104231      42801    2.435 0.026956 *
## dateApril_2010    65296       42801    1.526 0.146636
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 67670 on 16 degrees of freedom
## Multiple R-squared:  0.5448, Adjusted R-squared:  0.4595
## F-statistic: 6.383 on 3 and 16 DF,  p-value: 0.004745

summary(lm(bio_total ~ ph, data = datos_RH_est))

##
## Call:
## lm(formula = bio_total ~ ph, data = datos_RH_est)
##
## Residuals:
##     Min      1Q  Median      3Q      Max
## -104523 -68891 -21227  43030  220637
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) 133449     21078    6.331 5.76e-06 ***
## ph          7376      21626    0.341    0.737
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 94260 on 18 degrees of freedom
## Multiple R-squared:  0.006421, Adjusted R-squared:  -0.04878
## F-statistic: 0.1163 on 1 and 18 DF,  p-value: 0.737

summary(lm(bio_total ~ conductivity, data = datos_RH_est))

##
## Call:
## lm(formula = bio_total ~ conductivity, data = datos_RH_est)
##
## Residuals:
##     Min      1Q  Median      3Q      Max
## -128497 -34802 -12961   17067  200813
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
```

```

## (Intercept) 133449      17708    7.536 5.68e-07 ***
## conductivity 50310       18168    2.769   0.0126 *
## ---
## Signif. codes: 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 79190 on 18 degrees of freedom
## Multiple R-squared: 0.2987, Adjusted R-squared: 0.2598
## F-statistic: 7.668 on 1 and 18 DF, p-value: 0.01264

summary(lm(bio_total ~ total_hardness, data = datos_RH_est))

##
## Call:
## lm(formula = bio_total ~ total_hardness, data = datos_RH_est)
##
## Residuals:
##    Min     1Q Median     3Q    Max
## -132383 -63178 -23214  49696 224033
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) 133449     20523    6.503 4.09e-06 ***
## total_hardness -22189     21056   -1.054    0.306
## ---
## Signif. codes: 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 91780 on 18 degrees of freedom
## Multiple R-squared: 0.05811, Adjusted R-squared: 0.005787
## F-statistic: 1.111 on 1 and 18 DF, p-value: 0.3059

summary(lm(bio_total ~ doc, data = datos_RH_est))

##
## Call:
## lm(formula = bio_total ~ doc, data = datos_RH_est)
##
## Residuals:
##    Min     1Q Median     3Q    Max
## -91318 -58325 -35827  39124 250399
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) 133449     20406    6.540 3.8e-06 ***
## doc         24139      20936    1.153    0.264

```

```
## ---
## Signif. codes: 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 91260 on 18 degrees of freedom
## Multiple R-squared: 0.06877, Adjusted R-squared: 0.01704
## F-statistic: 1.329 on 1 and 18 DF, p-value: 0.264
```

```
summary(lm(bio_total ~ din, data = datos_RH_est))
```

```
##
## Call:
## lm(formula = bio_total ~ din, data = datos_RH_est)
##
## Residuals:
##    Min      1Q  Median      3Q     Max
## -120561 -62881 -19252  32721 230252
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) 133449     20342   6.560 3.65e-06 ***
## din         -25138     20871  -1.204   0.244
## ---
## Signif. codes: 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 90970 on 18 degrees of freedom
## Multiple R-squared: 0.07458, Adjusted R-squared: 0.02317
## F-statistic: 1.451 on 1 and 18 DF, p-value: 0.244
```

```
summary(lm(bio_total ~ total_nitrogen, data = datos_RH_est))
```

```
##
## Call:
## lm(formula = bio_total ~ total_nitrogen, data = datos_RH_est)
##
## Residuals:
##    Min      1Q  Median      3Q     Max
## -85298 -49521 -14767  25657 211357
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) 133449     17878   7.464 6.48e-07 ***
## total_nitrogen 49157     18343   2.680  0.0153 *
## ---
## Signif. codes: 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```

## 
## Residual standard error: 79950 on 18 degrees of freedom
## Multiple R-squared:  0.2852, Adjusted R-squared:  0.2455
## F-statistic: 7.182 on 1 and 18 DF,  p-value: 0.01529

summary(lm(bio_total ~ drp, data = datos_RH_est))

## 
## Call:
## lm(formula = bio_total ~ drp, data = datos_RH_est)
## 
## Residuals:
##    Min     1Q   Median     3Q    Max 
## -107917 -65468 -10016  43286 228659 
## 
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)    
## (Intercept) 133449     21026   6.347 5.58e-06 ***
## drp        -9784      21573  -0.454   0.656    
## ---        
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1 

## 
## Residual standard error: 94030 on 18 degrees of freedom
## Multiple R-squared:  0.0113, Adjusted R-squared:  -0.04363
## F-statistic: 0.2057 on 1 and 18 DF,  p-value: 0.6556

summary(lm(bio_total ~ total_phosphorus, data = datos_RH_est))

## 
## Call:
## lm(formula = bio_total ~ total_phosphorus, data = datos_RH_est)
## 
## Residuals:
##    Min     1Q   Median     3Q    Max 
## -104702 -73112 -20945  36201 234799 
## 
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)    
## (Intercept) 133449     21117   6.320 5.9e-06 ***
## total_phosphorus 4836      21666   0.223   0.826  
## ---        
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1 

## 
## Residual standard error: 94440 on 18 degrees of freedom

```

```
## Multiple R-squared:  0.002761,  Adjusted R-squared: -0.05264
## F-statistic: 0.04983 on 1 and 18 DF,  p-value: 0.8259
```

```
summary(lm(bio_total ~ ag_440, data = datos_RH_est))
```

```
##
## Call:
## lm(formula = bio_total ~ ag_440, data = datos_RH_est)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -107303  -68950  -18224   48291  226609
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) 133449     21123    6.318 5.92e-06 ***
## ag_440      -4275     21672   -0.197    0.846
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 94470 on 18 degrees of freedom
## Multiple R-squared:  0.002157,  Adjusted R-squared: -0.05328
## F-statistic: 0.03891 on 1 and 18 DF,  p-value: 0.8458
```

```
summary(lm(bio_total ~ suva254, data = datos_RH_est))
```

```
##
## Call:
## lm(formula = bio_total ~ suva254, data = datos_RH_est)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -104804  -72849  -18281   49172  228928
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) 133449     21139    6.313 5.98e-06 ***
## suva254     -2359     21688   -0.109    0.915
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 94540 on 18 degrees of freedom
## Multiple R-squared:  0.0006568,  Adjusted R-squared: -0.05486
## F-statistic: 0.01183 on 1 and 18 DF,  p-value: 0.9146
```

```
summary(lm(bio_total ~ ms_index, data = datos_RH_est))

##
## Call:
## lm(formula = bio_total ~ ms_index, data = datos_RH_est)
##
## Residuals:
##    Min     1Q Median     3Q    Max
## -126832 -71941 -1492  46065 198950
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) 133449     20222   6.599 3.38e-06 ***
## ms_index    -26921     20747  -1.298    0.211
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 90430 on 18 degrees of freedom
## Multiple R-squared:  0.08554, Adjusted R-squared:  0.03473
## F-statistic: 1.684 on 1 and 18 DF,  p-value: 0.2108
```

Observar que el modelo univariado con *date* es significativo: la biomasa bacteriana es distinta en diciembre de 2009 y febrero de 2010 ($p < 0.05$) respecto al nivel base de octubre de 2009. También son significativos los modelos de *conductivity* y de *total_nitrogen*. Los R^2 de estos modelos son 0.4595, 0.2598 y 0.2455, respectivamente. Podemos obtener sus AIC con la función AIC como hicimos con el modelo nulo y vemos que sus AIC son 507.19, 511.84 y 512.22. En todos los casos, el AIC es menor al del modelo nulo. Haremos entonces dos ramas de modelos de regresión múltiple, uno iniciando con *date* y otro iniciando con *conductivity* (podríamos hacer también la rama de *total_nitrogen*).

2.5.3.2 Modelos de regresión múltiple

Partimos de los dos mejores modelos univariados y agregamos de a una a todas las variables.

2.5.3.3 Iniciando con la fecha de muestreo

```
modelo_fecha <- lm(bio_total ~ date, data = datos_RH_est) ## AIC 507.19
```

Los modelos bivariados son (analicemos los p-valores de las variables incluidas y luego el AIC de los modelos):

```

summary(lm(bio_total ~ date + ph, data = datos_RH_est))

##
## Call:
## lm(formula = bio_total ~ date + ph, data = datos_RH_est)
##
## Residuals:
##    Min     1Q   Median     3Q    Max
## -138477 -22328   -1676   26205 142757
##
## Coefficients:
##                               Estimate Std. Error t value Pr(>|t|)
## (Intercept)             43856     32481    1.350  0.19696
## dateDecember_2009      184492     45132    4.088  0.00097 ***
## dateFebruary_2010      105771     45141    2.343  0.03332 *
## dateApril_2010          68111     47352    1.438  0.17086
## ph                     -2832     17181   -0.165  0.87130
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 69830 on 15 degrees of freedom
## Multiple R-squared:  0.5456, Adjusted R-squared:  0.4245
## F-statistic: 4.503 on 4 and 15 DF,  p-value: 0.01373

summary(lm(bio_total ~ date + conductivity, data = datos_RH_est))

##
## Call:
## lm(formula = bio_total ~ date + conductivity, data = datos_RH_est)
##
## Residuals:
##    Min     1Q   Median     3Q    Max
## -109509 -40830    -75   35420 121548
##
## Coefficients:
##                               Estimate Std. Error t value Pr(>|t|)
## (Intercept)           122957.8     60820.4    2.022  0.0614 .
## dateDecember_2009    89081.0     76614.9    1.163  0.2631
## dateFebruary_2010     273.1     82526.5    0.003  0.9974
## dateApril_2010        -47387.8    87763.6   -0.540  0.5972
## conductivity         51368.1     35282.2    1.456  0.1660
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##

```

```

## Residual standard error: 65420 on 15 degrees of freedom
## Multiple R-squared:  0.6012, Adjusted R-squared:  0.4948
## F-statistic: 5.652 on 4 and 15 DF,  p-value: 0.005585

summary(lm(bio_total ~ date + total_hardness, data = datos_RH_est)) ## AIC 502.45

## 
## Call:
## lm(formula = bio_total ~ date + total_hardness, data = datos_RH_est)
##
## Residuals:
##    Min      1Q  Median      3Q     Max
## -102529 -29179   -6313   35596  121418
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) 43958     26418   1.664 0.116872
## dateDecember_2009 186393     37379   4.987 0.000163 ***
## dateFebruary_2010 125816     38377   3.278 0.005079 **
## dateApril_2010    45756     38194   1.198 0.249507
## total_hardness   -36430     14863  -2.451 0.026984 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

## Residual standard error: 59060 on 15 degrees of freedom
## Multiple R-squared:  0.675, Adjusted R-squared:  0.5883
## F-statistic: 7.788 on 4 and 15 DF,  p-value: 0.001324

summary(lm(bio_total ~ date + doc, data = datos_RH_est))

## 
## Call:
## lm(formula = bio_total ~ date + doc, data = datos_RH_est)
##
## Residuals:
##    Min      1Q  Median      3Q     Max
## -145469 -14517   -2572   24403  130490
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) 38792     35810   1.083 0.2958
## dateDecember_2009 191996     50369   3.812 0.0017 **
## dateFebruary_2010 112259     49096   2.287 0.0372 *
## dateApril_2010    74375     50424   1.475 0.1609

```

```

## doc              -7058      19141   -0.369   0.7175
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 69580 on 15 degrees of freedom
## Multiple R-squared:  0.5489, Adjusted R-squared:  0.4286
## F-statistic: 4.563 on 4 and 15 DF, p-value: 0.01306

summary(lm(bio_total ~ date + din, data = datos_RH_est))

##
## Call:
## lm(formula = bio_total ~ date + din, data = datos_RH_est)
##
## Residuals:
##    Min     1Q Median     3Q    Max
## -142000 -24216   -786  29043 135880
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) 43406     32146   1.350  0.19695
## dateDecember_2009 187573     47890   3.917  0.00137 **
## dateFebruary_2010 107242     45760   2.344  0.03329 *
## dateApril_2010   65361     44115   1.482  0.15915
## din          4449      17971   0.248  0.80784
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 69750 on 15 degrees of freedom
## Multiple R-squared:  0.5467, Adjusted R-squared:  0.4258
## F-statistic: 4.522 on 4 and 15 DF, p-value: 0.01351

summary(lm(bio_total ~ date + total_nitrogen, data = datos_RH_est)) ## AIC 498.5779

##
## Call:
## lm(formula = bio_total ~ date + total_nitrogen, data = datos_RH_est)
##
## Residuals:
##    Min     1Q Median     3Q    Max
## -80129 -30021    785  28393 104860
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)

```

```

## (Intercept)      102645     29790    3.446  0.00360 **
## dateDecember_2009 134460     37057    3.628  0.00248 **
## dateFebruary_2010   4520     45781    0.099  0.92267
## dateApril_2010     -15762    42129   -0.374  0.71354
## total_nitrogen      55630     17165    3.241  0.00548 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 53600 on 15 degrees of freedom
## Multiple R-squared:  0.7323, Adjusted R-squared:  0.6609
## F-statistic: 10.26 on 4 and 15 DF,  p-value: 0.0003312

summary(lm(bio_total ~ date + drp, data = datos_RH_est))

## 
## Call:
## lm(formula = bio_total ~ date + drp, data = datos_RH_est)
## 
## Residuals:
##    Min     1Q Median     3Q    Max
## -138451 -28295    995  25376 139799
## 
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  41498     31571    1.314 0.208444
## dateDecember_2009 189427     45047    4.205 0.000765 ***
## dateFebruary_2010 111720     45497    2.456 0.026749 *
## dateApril_2010    66658     43759    1.523 0.148488
## drp          9931      16799    0.591 0.563205
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 69090 on 15 degrees of freedom
## Multiple R-squared:  0.5552, Adjusted R-squared:  0.4365
## F-statistic:  4.68 on 4 and 15 DF,  p-value: 0.01187

summary(lm(bio_total ~ date + total_phosphorus, data = datos_RH_est))

## 
## Call:
## lm(formula = bio_total ~ date + total_phosphorus, data = datos_RH_est)
## 
## Residuals:
##    Min     1Q Median     3Q    Max
## 
```

```

## -134792 -28395 -3691 29135 136851
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) 50625     31415   1.611 0.127911
## dateDecember_2009 184206     43383   4.246 0.000704 ***
## dateFebruary_2010 95922      44671   2.147 0.048522 *
## dateApril_2010    51171      47061   1.087 0.294056
## total_phosphorus 13737      17808   0.771 0.452444
## ---
## Signif. codes: 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 68550 on 15 degrees of freedom
## Multiple R-squared: 0.5622, Adjusted R-squared: 0.4454
## F-statistic: 4.815 on 4 and 15 DF, p-value: 0.01065

summary(lm(bio_total ~ date + ag_440, data = datos_RH_est))

##
## Call:
## lm(formula = bio_total ~ date + ag_440, data = datos_RH_est)
##
## Residuals:
##    Min     1Q Median     3Q    Max
## -143531 -20154   3670  23616 122559
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) 32541     35642   0.913 0.375674
## dateDecember_2009 188239     44114   4.267 0.000675 ***
## dateFebruary_2010 123976     51630   2.401 0.029752 *
## dateApril_2010    91417      56982   1.604 0.129489
## ag_440        -15573     21957  -0.709 0.489043
## ---
## Signif. codes: 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 68750 on 15 degrees of freedom
## Multiple R-squared: 0.5596, Adjusted R-squared: 0.4421
## F-statistic: 4.764 on 4 and 15 DF, p-value: 0.01109

summary(lm(bio_total ~ date + suva254, data = datos_RH_est))

##
## Call:

```

```

## lm(formula = bio_total ~ date + suva254, data = datos_RH_est)
##
## Residuals:
##   Min     1Q Median     3Q    Max
## -141744 -20603 -5302 18056 150149
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) 48331     31176  1.550 0.14191
## dateDecember_2009 189410     44722  4.235 0.00072 ***
## dateFebruary_2010 90850      48246  1.883 0.07923 .
## dateApril_2010    60212     44300  1.359 0.19417
## suva254         12662     19548  0.648 0.52695
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 68940 on 15 degrees of freedom
## Multiple R-squared:  0.5572, Adjusted R-squared:  0.4391
## F-statistic: 4.719 on 4 and 15 DF,  p-value: 0.01151

summary(lm(bio_total ~ date + ms_index, data = datos_RH_est)) ## AIC 500.7191

##
## Call:
## lm(formula = bio_total ~ date + ms_index, data = datos_RH_est)
##
## Residuals:
##   Min     1Q Median     3Q    Max
## -79021 -30748  1850 37503 100396
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) 32000     25730  1.244 0.23271
## dateDecember_2009 200774     36322  5.528 5.8e-05 ***
## dateFebruary_2010 111544     35860  3.111 0.00716 **
## dateApril_2010    93481     37142  2.517 0.02370 *
## ms_index       -38101     13544 -2.813 0.01311 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 56550 on 15 degrees of freedom
## Multiple R-squared:  0.702, Adjusted R-squared:  0.6226
## F-statistic: 8.835 on 4 and 15 DF,  p-value: 0.0007134

```

Entre todos estos modelos, el mejor es el que incluye la fecha y el nitrógeno total, con AIC menor al modelo univariado. Vamos a analizar si hay una interacción

significativa entre la fecha y el nitrógeno total, para lo cual reemplazamos el símbolo + por un *.

```
summary(lm(bio_total ~ date * total_nitrogen, data = datos_RH_est))

##
## Call:
## lm(formula = bio_total ~ date * total_nitrogen, data = datos_RH_est)
##
## Residuals:
##    Min     1Q Median     3Q    Max
## -74889 -21946 -2761 19545 75950
##
## Coefficients:
##                               Estimate Std. Error t value Pr(>|t|)
## (Intercept)                  54478     115308   0.472   0.645
## dateDecember_2009             190297     116940   1.627   0.130
## dateFebruary_2010            109279     120020   0.911   0.380
## dateApril_2010                 43304     117273   0.369   0.718
## total_nitrogen                  8881     110351   0.080   0.937
## dateDecember_2009:total_nitrogen  95135     112247   0.848   0.413
## dateFebruary_2010:total_nitrogen -27512     115984  -0.237   0.816
## dateApril_2010:total_nitrogen      21209     112532   0.188   0.854
##
## Residual standard error: 42920 on 12 degrees of freedom
## Multiple R-squared:  0.8627, Adjusted R-squared:  0.7826
## F-statistic: 10.77 on 7 and 12 DF,  p-value: 0.0002447
```

Vemos que ninguno de los términos de la interacción es significativo.

Podemos también realizar un análisis de la varianza para evaluar si los tres modelos ajustados difieren significativamente (y en ese caso, vale la pena quedarnos con un modelo más complejo).

```
modelo_fecha2 <- lm(bio_total ~ date + total_nitrogen, data = datos_RH_est)
anova(modelo_fecha, modelo_fecha2)

## Analysis of Variance Table
##
## Model 1: bio_total ~ date
## Model 2: bio_total ~ date + total_nitrogen
##   Res.Df       RSS Df Sum of Sq    F    Pr(>F)
## 1     16 7.3277e+10
## 2     15 4.3098e+10  1 3.0178e+10 10.503 0.005485 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Ahora seguimos con modelos con tres variables explicativas.

```
summary(lm(bio_total ~ date + total_nitrogen + ph, data = datos_RH_est))

##
## Call:
## lm(formula = bio_total ~ date + total_nitrogen + ph, data = datos_RH_est)
##
## Residuals:
##   Min     1Q Median     3Q    Max 
## -88009 -21620   2162  34206  86138 
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)    
## (Intercept) 111660     32782   3.406  0.00426 ***
## dateDecember_2009 125815     39547   3.181  0.00666 ** 
## dateFebruary_2010 -7546     49478  -0.153  0.88095  
## dateApril_2010    -31112     47864  -0.650  0.52621  
## total_nitrogen    59317     18189   3.261  0.00568 ** 
## ph              10036     13975   0.718  0.48448  
## ---            
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 54490 on 14 degrees of freedom
## Multiple R-squared:  0.7418, Adjusted R-squared:  0.6496 
## F-statistic: 8.044 on 5 and 14 DF,  p-value: 0.0009286

summary(lm(bio_total ~ date + total_nitrogen + conductivity, data = datos_RH_est)) # AIC 492.7927

##
## Call:
## lm(formula = bio_total ~ date + total_nitrogen + conductivity,
##      data = datos_RH_est)
##
## Residuals:
##   Min     1Q Median     3Q    Max 
## -76085 -26521    479  24654  81376 
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)    
## (Intercept) 204379     46880   4.360 0.000654 ***
## dateDecember_2009 13212     56599   0.233 0.818802  
## dateFebruary_2010 -133679     66243  -2.018 0.063178 . 
## dateApril_2010    -163251     67480  -2.419 0.029750 *
```

```

## total_nitrogen      60384     14741    4.096 0.001090 **
## conductivity       64076     24824    2.581 0.021760 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 45670 on 14 degrees of freedom
## Multiple R-squared:  0.8186, Adjusted R-squared:  0.7538
## F-statistic: 12.64 on 5 and 14 DF,  p-value: 8.883e-05

summary(lm(bio_total ~ date + total_nitrogen + total_hardness, data = datos_RH_est))

##
## Call:
## lm(formula = bio_total ~ date + total_nitrogen + total_hardness,
##      data = datos_RH_est)
##
## Residuals:
##   Min     1Q Median     3Q    Max
## -71913 -34297   4589  24140 102928
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  89280     30640    2.914  0.01133 *
## dateDecember_2009 147030     37255    3.947  0.00146 **
## dateFebruary_2010  38410     51138    0.751  0.46502
## dateApril_2010    -8762     41339   -0.212  0.83520
## total_nitrogen   43395     19005    2.283  0.03855 *
## total_hardness   -20187     14935   -1.352  0.19793
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 52180 on 14 degrees of freedom
## Multiple R-squared:  0.7632, Adjusted R-squared:  0.6786
## F-statistic: 9.023 on 5 and 14 DF,  p-value: 0.0005253

summary(lm(bio_total ~ date + total_nitrogen + doc, data = datos_RH_est))

##
## Call:
## lm(formula = bio_total ~ date + total_nitrogen + doc, data = datos_RH_est)
##
## Residuals:
##   Min     1Q Median     3Q    Max
## -84034 -30767  -8906  29102  79931

```

```

## 
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)    
## (Intercept) 90024     30482   2.953  0.01048 *  
## dateDecember_2009 155060    39187   3.957  0.00143 ** 
## dateFebruary_2010 17021     45535   0.374  0.71416    
## dateApril_2010    1623      43004   0.038  0.97043    
## total_nitrogen   61327     17241   3.557  0.00316 ** 
## doc            -19971     14812  -1.348  0.19897    
## --- 
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1 

## 
## Residual standard error: 52200 on 14 degrees of freedom
## Multiple R-squared:  0.763, Adjusted R-squared:  0.6784 
## F-statistic: 9.016 on 5 and 14 DF,  p-value: 0.0005273

summary(lm(bio_total ~ date + total_nitrogen + din, data = datos_RH_est))

## 
## Call:
## lm(formula = bio_total ~ date + total_nitrogen + din, data = datos_RH_est)
## 
## Residuals:
##    Min     1Q Median     3Q    Max 
## -79543 -24954   6940  25554 100447 

## 
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)    
## (Intercept) 99997     30860   3.240  0.00593 ** 
## dateDecember_2009 142035    40273   3.527  0.00335 ** 
## dateFebruary_2010 8532      47410   0.180  0.85976    
## dateApril_2010    -16755    43163  -0.388  0.70372    
## total_nitrogen   56390     17624   3.200  0.00643 ** 
## din           7943      14180   0.560  0.58421    
## --- 
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1 

## 
## Residual standard error: 54870 on 14 degrees of freedom
## Multiple R-squared:  0.7381, Adjusted R-squared:  0.6446 
## F-statistic: 7.893 on 5 and 14 DF,  p-value: 0.001018

summary(lm(bio_total ~ date + total_nitrogen + drp, data = datos_RH_est))

## 
```

```

## Call:
## lm(formula = bio_total ~ date + total_nitrogen + drp, data = datos_RH_est)
##
## Residuals:
##   Min     1Q Median     3Q    Max 
## -76496 -29895   -583  27004  98822 
##
## Coefficients:
##                               Estimate Std. Error t value Pr(>|t|)    
## (Intercept)             114089     33616   3.394  0.00437 ** 
## dateDecember_2009       121051     41359   2.927  0.01104 *  
## dateFebruary_2010      -16343     53654  -0.305  0.76515    
## dateApril_2010          -27265     45211  -0.603  0.55611    
## total_nitrogen          62445     19497   3.203  0.00638 ** 
## drp                     -11466     14803  -0.775  0.45147    
## ---                     
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1 
##
## Residual standard error: 54330 on 14 degrees of freedom
## Multiple R-squared:  0.7433, Adjusted R-squared:  0.6516 
## F-statistic: 8.107 on 5 and 14 DF,  p-value: 0.000894 

summary(lm(bio_total ~ date + total_nitrogen + total_phosphorus, data = datos_RH_est))

##
## Call:
## lm(formula = bio_total ~ date + total_nitrogen + total_phosphorus,
##      data = datos_RH_est)
##
## Residuals:
##   Min     1Q Median     3Q    Max 
## -74631 -34398   1280  18561 104125 
##
## Coefficients:
##                               Estimate Std. Error t value Pr(>|t|)    
## (Intercept)             106652     30436   3.504  0.00351 ** 
## dateDecember_2009       136057     37452   3.633  0.00272 ** 
## dateFebruary_2010      -1662     46778  -0.036  0.97215    
## dateApril_2010          -27204     44603  -0.610  0.55168    
## total_nitrogen          55043     17340   3.174  0.00676 ** 
## total_phosphorus        11960     14067   0.850  0.40952    
## ---                     
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1 
##
## Residual standard error: 54100 on 14 degrees of freedom

```

```

## Multiple R-squared:  0.7454, Adjusted R-squared:  0.6545
## F-statistic: 8.198 on 5 and 14 DF,  p-value: 0.0008461

summary(lm(bio_total ~ date + total_nitrogen + ag_440, data = datos_RH_est))

## 
## Call:
## lm(formula = bio_total ~ date + total_nitrogen + ag_440, data = datos_RH_est)
## 
## Residuals:
##    Min      1Q Median      3Q     Max 
## -81230 -30738     188  28914 102242 
## 
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)    
## (Intercept) 99429     36313   2.738  0.01601 *  
## dateDecember_2009 136100    39553   3.441  0.00398 ** 
## dateFebruary_2010  9627    56331   0.171  0.86675  
## dateApril_2010    -9643    56884  -0.170  0.86781  
## total_nitrogen   54932    18233   3.013  0.00931 ** 
## ag_440          -3042    18185  -0.167  0.86954  
## ---            
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1 

## 
## Residual standard error: 55430 on 14 degrees of freedom
## Multiple R-squared:  0.7328, Adjusted R-squared:  0.6374 
## F-statistic: 7.679 on 5 and 14 DF,  p-value: 0.001162

summary(lm(bio_total ~ date + total_nitrogen + suva254, data = datos_RH_est))

## 
## Call:
## lm(formula = bio_total ~ date + total_nitrogen + suva254, data = datos_RH_est)
## 
## Residuals:
##    Min      1Q Median      3Q     Max 
## -80812 -35249  -2144  24093 114354 
## 
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)    
## (Intercept) 104196     30534   3.412  0.00421 ** 
## dateDecember_2009 139924    38887   3.598  0.00291 ** 
## dateFebruary_2010  -4325    48937  -0.088  0.93083  
## dateApril_2010    -18586    43279  -0.429  0.67414 

```

```

## total_nitrogen      54938     17570    3.127  0.00742 **
## suva254            9541     15558    0.613  0.54956
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 54750 on 14 degrees of freedom
## Multiple R-squared:  0.7393, Adjusted R-squared:  0.6462
## F-statistic: 7.939 on 5 and 14 DF,  p-value: 0.0009894

summary(lm(bio_total ~ date + total_nitrogen + ms_index, data = datos_RH_est)) # AIC 4

##
## Call:
## lm(formula = bio_total ~ date + total_nitrogen + ms_index, data = datos_RH_est)
##
## Residuals:
##   Min     1Q Median     3Q    Max
## -69581 -27527  -3505  36231  54390
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) 83140     25409    3.272 0.005563 **
## dateDecember_2009 156263     31389    4.978 0.000203 ***
## dateFebruary_2010 25584     38333    0.667 0.515370
## dateApril_2010    19390     36730    0.528 0.605838
## total_nitrogen    47185     14410    3.274 0.005538 **
## ms_index        -30884     10778   -2.866 0.012462 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 44050 on 14 degrees of freedom
## Multiple R-squared:  0.8312, Adjusted R-squared:  0.771
## F-statistic: 13.79 on 5 and 14 DF,  p-value: 5.461e-05

```

Hay dos modelos significativos, ambos con menor AIC que el modelo bivariado. Entre ellos, el AIC es muy similar (diferencias menores de 2) y el R^2 también. ¿Con qué modelo de regresión múltiple seguir?

Si volvemos a los análisis de correlación vemos que el nitrógeno total está algo correlacionado con la conductividad (pero con rho relativamente bajo, de 0.48) mientras que no está significativamente correlacionado con ms_index. Ese puede ser un criterio para elegir un modelo (el que incluye ms_index). O bien, podemos elegir al modelo con conductivity porque el R^2 es mayor. Alternativamente, podemos seguir dos nuevas ramas con ambos modelos. O bien, ante dos modelos prácticamente equivalentes podemos escoger el que más sentido

tenga de acuerdo a nuestro conocimiento biológico (o presentar ambos modelos y fundamentar cuál nos parece más adecuado).

Evaluemos estos modelos con las interacciones. En lugar de usar * (que implica analizar las variables separadas y la interacción), vamos a usar específicamente : para indicar qué interacciones queremos considerar.

```
summary(lm(bio_total ~ date + total_nitrogen + conductivity, data = datos_RH_est)) # AIC 492.792

##
## Call:
## lm(formula = bio_total ~ date + total_nitrogen + conductivity,
##      data = datos_RH_est)
##
## Residuals:
##   Min     1Q Median     3Q    Max
## -76085 -26521     479  24654  81376
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) 204379     46880    4.360 0.000654 ***
## dateDecember_2009 13212     56599    0.233 0.818802
## dateFebruary_2010 -133679     66243   -2.018 0.063178 .
## dateApril_2010    -163251     67480   -2.419 0.029750 *
## total_nitrogen    60384     14741    4.096 0.001090 **
## conductivity      64076     24824    2.581 0.021760 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 45670 on 14 degrees of freedom
## Multiple R-squared:  0.8186, Adjusted R-squared:  0.7538
## F-statistic: 12.64 on 5 and 14 DF,  p-value: 8.883e-05

summary(lm(bio_total ~ date + total_nitrogen + conductivity + date:conductivity, data = datos_RH_
```

```
##
## Call:
## lm(formula = bio_total ~ date + total_nitrogen + conductivity +
##      date:conductivity, data = datos_RH_est)
##
## Residuals:
##   Min     1Q Median     3Q    Max
## -79569 -16033 -1263  18343  72487
##
## Coefficients:
```

```

##                                     Estimate Std. Error t value Pr(>|t|) 
## (Intercept)                  132108     71856   1.839  0.09312 .
## dateDecember_2009            75433      74016   1.019  0.33003 
## dateFebruary_2010           -77572     88675  -0.875  0.40038 
## dateApril_2010                -102751    93765  -1.096  0.29656 
## total_nitrogen                 58607     18275   3.207  0.00835 ** 
## conductivity                   17465     42283   0.413  0.68750 
## dateDecember_2009:conductivity 77496     74717   1.037  0.32191 
## dateFebruary_2010:conductivity 80791     75287   1.073  0.30620 
## dateApril_2010:conductivity    64972     69846   0.930  0.37222 
## --- 
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1 
## 
## Residual standard error: 47470 on 11 degrees of freedom 
## Multiple R-squared:  0.846,  Adjusted R-squared:  0.734 
## F-statistic: 7.555 on 8 and 11 DF,  p-value: 0.001545 

summary(lm(bio_total ~ date + total_nitrogen + conductivity + total_nitrogen:conductiv

## 
## Call: 
## lm(formula = bio_total ~ date + total_nitrogen + conductivity + 
##     total_nitrogen:conductivity, data = datos_RH_est) 
## 
## Residuals: 
##    Min     1Q Median     3Q    Max 
## -71931 -25219     876  22428  81171 
## 
## Coefficients: 
##                                     Estimate Std. Error t value Pr(>|t|) 
## (Intercept)                  220911.9    70763.7   3.122  0.00810 ** 
## dateDecember_2009             -823.7     73071.7  -0.011  0.99118 
## dateFebruary_2010            -148551.2    82707.3  -1.796  0.09574 . 
## dateApril_2010                -179599.8    86404.6  -2.079  0.05802 . 
## total_nitrogen                  63833.5    18652.6   3.422  0.00454 ** 
## conductivity                   62597.7    26071.1   2.401  0.03202 * 
## total_nitrogen:conductivity   -10212.6    31854.1  -0.321  0.75360 
## --- 
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1 
## 
## Residual standard error: 47210 on 13 degrees of freedom 
## Multiple R-squared:  0.82,  Adjusted R-squared:  0.737 
## F-statistic: 9.872 on 6 and 13 DF,  p-value: 0.0003276

```

```

summary(lm(bio_total ~ date + total_nitrogen + conductivity + date:conductivity + total_nitrogen))

##
## Call:
## lm(formula = bio_total ~ date + total_nitrogen + conductivity +
##     date:conductivity + total_nitrogen:conductivity, data = datos_RH_est)
##
## Residuals:
##   Min     1Q Median     3Q    Max 
## -73199 -13340   2523  17166  76595 
##
## Coefficients:
##                               Estimate Std. Error t value Pr(>|t|)    
## (Intercept)                 180913    94397   1.917   0.0843 .  
## dateDecember_2009            43420     84812   0.512   0.6198    
## dateFebruary_2010           -137948    116607  -1.183   0.2642    
## dateApril_2010              -166781    123447  -1.351   0.2065    
## total_nitrogen                75160    27511   2.732   0.0211 *  
## conductivity                  5072     45555   0.111   0.9136    
## dateDecember_2009:conductivity 60650    78648   0.771   0.4584    
## dateFebruary_2010:conductivity 114014    86647   1.316   0.2176    
## dateApril_2010:conductivity   92472    78551   1.177   0.2664    
## total_nitrogen:conductivity   -33888    41575  -0.815   0.4340    
## ---                        
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1 
##
## Residual standard error: 48210 on 10 degrees of freedom
## Multiple R-squared:  0.8556, Adjusted R-squared:  0.7257 
## F-statistic: 6.585 on 9 and 10 DF,  p-value: 0.003431

summary(lm(bio_total ~ date + total_nitrogen + ms_index, data = datos_RH_est)) # AIC 491.3469

##
## Call:
## lm(formula = bio_total ~ date + total_nitrogen + ms_index, data = datos_RH_est)
##
## Residuals:
##   Min     1Q Median     3Q    Max 
## -69581 -27527  -3505  36231  54390 
##
## Coefficients:
##                               Estimate Std. Error t value Pr(>|t|)    
## (Intercept)                 83140     25409   3.272  0.005563 ** 
## dateDecember_2009            156263    31389   4.978  0.000203 *** 

```

```

## dateFebruary_2010      25584      38333   0.667 0.515370
## dateApril_2010         19390      36730   0.528 0.605838
## total_nitrogen        47185      14410   3.274 0.005538 **
## ms_index                -30884     10778  -2.866 0.012462 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 44050 on 14 degrees of freedom
## Multiple R-squared:  0.8312, Adjusted R-squared:  0.771
## F-statistic: 13.79 on 5 and 14 DF,  p-value: 5.461e-05

summary(lm(bio_total ~ date + total_nitrogen + ms_index + date:ms_index, data = datos_L))

##
## Call:
## lm(formula = bio_total ~ date + total_nitrogen + ms_index + date:ms_index,
##     data = datos_RH_est)
##
## Residuals:
##    Min     1Q Median     3Q    Max 
## -70539 -20560   2474  18498  54609 
##
## Coefficients:
##                               Estimate Std. Error t value Pr(>|t|)    
## (Intercept)                 90225.35  25628.26   3.521 0.004794 ***
## dateDecember_2009            151094.81  30468.42   4.959 0.000429 ***
## dateFebruary_2010             20957.54  39972.34   0.524 0.610471  
## dateApril_2010                -2955.24  39305.28  -0.075 0.941416  
## total_nitrogen                  43586.88  15566.88   2.800 0.017276 *  
## ms_index                         -33.12  21304.05  -0.002 0.998787  
## dateDecember_2009:ms_index       -51968.26  25915.38  -2.005 0.070163 .  
## dateFebruary_2010:ms_index       -32641.50  30352.87  -1.075 0.305217  
## dateApril_2010:ms_index          12220.67  40131.37   0.305 0.766418  
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 40040 on 11 degrees of freedom
## Multiple R-squared:  0.8905, Adjusted R-squared:  0.8108
## F-statistic: 11.18 on 8 and 11 DF,  p-value: 0.0002695

summary(lm(bio_total ~ date + total_nitrogen + ms_index + total_nitrogen:ms_index, data = datos_R))

##
## Call:

```

```

## lm(formula = bio_total ~ date + total_nitrogen + ms_index + total_nitrogen:ms_index,
##     data = datos_RH_est)
##
## Residuals:
##   Min     1Q Median     3Q    Max
## -76031 -18967   -597  26222  57057
##
## Coefficients:
##                               Estimate Std. Error t value Pr(>|t|)
## (Intercept)                 91800     28294   3.245  0.00639 **
## dateDecember_2009            138778     39538   3.510  0.00384 **
## dateFebruary_2010             15876     41054   0.387  0.70522
## dateApril_2010                  10019     39366   0.255  0.80307
## total_nitrogen                53302     16770   3.178  0.00726 **
## ms_index                      -34156     11792  -2.897  0.01249 *
## total_nitrogen:ms_index      -13217     17663  -0.748  0.46761
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 44760 on 13 degrees of freedom
## Multiple R-squared:  0.8382, Adjusted R-squared:  0.7635
## F-statistic: 11.23 on 6 and 13 DF,  p-value: 0.00017

summary(lm(bio_total ~ date + total_nitrogen + ms_index + date:ms_index + total_nitrogen:ms_index))

##
## Call:
## lm(formula = bio_total ~ date + total_nitrogen + ms_index + date:ms_index +
##     total_nitrogen:ms_index, data = datos_RH_est)
##
## Residuals:
##   Min     1Q Median     3Q    Max
## -72808 -18730   1179  17115  54492
##
## Coefficients:
##                               Estimate Std. Error t value Pr(>|t|)
## (Intercept)                 92037     27592   3.336  0.00755 **
## dateDecember_2009            145705     37472   3.888  0.00302 **
## dateFebruary_2010              19474     42121   0.462  0.65373
## dateApril_2010                  -5056     41788  -0.121  0.90609
## total_nitrogen                45444     17635   2.577  0.02756 *
## ms_index                      -4286     27180  -0.158  0.87784
## dateDecember_2009:ms_index     -49057     29108  -1.685  0.12282
## dateFebruary_2010:ms_index     -27700     36529  -0.758  0.46577
## dateApril_2010:ms_index        15816     43958   0.360  0.72649

```

```

## total_nitrogen:ms_index      -5232      19185 -0.273  0.79063
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 41840 on 10 degrees of freedom
## Multiple R-squared:  0.8913, Adjusted R-squared:  0.7934
## F-statistic: 9.107 on 9 and 10 DF,  p-value: 0.0009328

```

Las interacciones no son significativas. Opto por continuar agregando variables a partir del modelo de *date + total_nitrogen + ms_index* (pueden tomar otro criterio).

Al probar todos los modelos con cuatro variables (pueden hacerlo), observamos que al añadir la conductividad hay un efecto significativo y el AIC disminuye. También llama la atención que los p-valores de las variables que ya habíamos incluido disminuyen.

```

modelo_fecha4 <- lm(bio_total ~ date + total_nitrogen + ms_index + conductivity, data =
summary(modelo_fecha4)

```

```

##
## Call:
## lm(formula = bio_total ~ date + total_nitrogen + ms_index + conductivity,
##     data = datos_RH_est)
##
## Residuals:
##    Min     1Q Median     3Q    Max
## -55147 -19486   2598  19344  47183
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) 176886    36808   4.806 0.000343 ***
## dateDecember_2009 44653    44322   1.007 0.332101
## dateFebruary_2010 -101324   51603  -1.964 0.071338 .
## dateApril_2010   -117074   53468  -2.190 0.047395 *
## total_nitrogen    52175    11545   4.519 0.000576 ***
## ms_index        -28388     8586  -3.306 0.005679 **
## conductivity     58052    19074   3.043 0.009418 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 34930 on 13 degrees of freedom
## Multiple R-squared:  0.9015, Adjusted R-squared:  0.856
## F-statistic: 19.82 on 6 and 13 DF,  p-value: 7.665e-06

```

```
AIC(modelo_fecha4)
```

```
## [1] 482.5878
```

Analizamos las interacciones y vemos que ninguna es significativa.

Cuando hay muchas variables, puede haber efectos de colinealidad, que quizás no observamos en las correlaciones entre pares de variables. Un criterio sugerido por Zuur et al. 2010 en su artículo A protocol for data exploration to avoid common statistical problems es calcular el valor **VIF** (*Variance Inflation Factor*) y sugiere quitar las variables con $VIF > 3$ (aunque señala que puede haber colinealidad con $VIF > 2$).

```
library(car)
```

```
modelo_fecha4 <- lm(bio_total ~ date + total_nitrogen + ms_index + conductivity, data = datos_RH)
AIC(modelo_fecha4)
```

```
## [1] 482.5878
```

```
vif(modelo_fecha4)
```

	GVIF	Df	GVIF ^{(1/(2*Df))}
## date	8.451357	3	1.427210
## total_nitrogen	2.075258	1	1.440576
## ms_index	1.147875	1	1.071389
## conductivity	5.665164	1	2.380160

Si el criterio (fijado a priori) es que VIF debe ser menor a 3, entonces estamos en condiciones de dejar las cuatro variables respuestas.

A continuación, probamos de agregar una quinta variable, pero ninguna tiene un efecto significativo adicional. Guardamos este modelo para continuar con el análisis de la otra rama. Ojo, nos falta evaluar los supuestos.

El modelo **modelo_fecha4** presenta un R^2 ajustado de 85,6%, un AIC de 482.5878 y 13 grados de libertad. Si observamos los coeficientes estimados, interpretamos que la biomasa bacteriana disminuye en febrero 2010 y en abril de 2010, aumenta en lagunas con mayor nitrógeno total y con mayor conductividad, y disminuye en lagunas con mayor ms_index. Dado que las variables están estandarizadas, podemos comparar los coeficientes: el efecto de la fecha es mayor al de las demás variables; y el efecto de ms_index es menor al de las demás variables.

2.5.3.4 Iniciando con la conductividad

Iniciaremos a partir del modelo univariado de conductividad, agregando variables de a una por vez, evaluando también las interacciones y utilizando como criterios: AIC (disminución de AIC en 2 unidades), p-valores (menor a 0.05) y vif (menor a 3)..., es muy importante informar estos criterios en un artículo. En este caso, opto por no incluir la variable categórica *date*. Pueden repetir el paso a paso, aquí un resumen de los mejores modelos.

```
#univariado
summary(lm(bio_total ~ conductivity, data = datos_RH_est)) # AIC 511.8359

##
## Call:
## lm(formula = bio_total ~ conductivity, data = datos_RH_est)
##
## Residuals:
##    Min      1Q  Median      3Q     Max
## -128497 -34802 -12961   17067  200813
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) 133449     17708    7.536 5.68e-07 ***
## conductivity  50310     18168    2.769   0.0126 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 79190 on 18 degrees of freedom
## Multiple R-squared:  0.2987, Adjusted R-squared:  0.2598
## F-statistic: 7.668 on 1 and 18 DF,  p-value: 0.01264

#bivariados significativos
summary(lm(bio_total ~ conductivity + ag_440, data = datos_RH_est)) # AIC 504.3061, R2
```

```
##
## Call:
## lm(formula = bio_total ~ conductivity + ag_440, data = datos_RH_est)
##
## Residuals:
##    Min      1Q  Median      3Q     Max
## -97027 -41980 -1633  46214 121043
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
```

```

## (Intercept) 133449      14359    9.294 4.48e-08 ***
## conductivity 89826       19170    4.686 0.000213 ***
## ag_440        -61754      19170   -3.221 0.005012 **
## ---
## Signif. codes: 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 64210 on 17 degrees of freedom
## Multiple R-squared: 0.5645, Adjusted R-squared: 0.5133
## F-statistic: 11.02 on 2 and 17 DF, p-value: 0.0008531

summary(lm(bio_total ~ conductivity * ag_440, data = datos_RH_est)) # interacción NS

## 
## Call:
## lm(formula = bio_total ~ conductivity * ag_440, data = datos_RH_est)
## 
## Residuals:
##     Min      1Q Median      3Q     Max
## -94420 -43144 -2424  47154 121339
## 
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)    
## (Intercept) 135032    28398    4.755 0.000215 ***
## conductivity 87839    36285    2.421 0.027746 *  
## ag_440      -59488    39936   -1.490 0.155787    
## conductivity:ag_440 -2604    39870   -0.065 0.948745  
## ---
## Signif. codes: 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 66180 on 16 degrees of freedom
## Multiple R-squared: 0.5647, Adjusted R-squared: 0.483
## F-statistic: 6.918 on 3 and 16 DF, p-value: 0.00337

summary(lm(bio_total ~ conductivity + ms_index, data = datos_RH_est)) # AIC 508.9108 , R2_adj = 0.5647

## 
## Call:
## lm(formula = bio_total ~ conductivity + ms_index, data = datos_RH_est)
## 
## Residuals:
##     Min      1Q Median      3Q     Max
## -141521 -31043 -10192  21664 185043
## 
## Coefficients:

```

```

##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) 133449     16110   8.283 2.26e-07 ***
## conductivity    56532     16774   3.370  0.00363 **
## ms_index      -36546     16774  -2.179  0.04371 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 72050 on 17 degrees of freedom
## Multiple R-squared:  0.4518, Adjusted R-squared:  0.3873
## F-statistic: 7.006 on 2 and 17 DF,  p-value: 0.006038

summary(lm(bio_total ~ conductivity * ms_index, data = datos_RH_est)) # interacción NS

##
## Call:
## lm(formula = bio_total ~ conductivity * ms_index, data = datos_RH_est)
##
## Residuals:
##    Min      1Q  Median      3Q     Max
## -138975 -30031  -5627   25350  184943
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) 135400     16654   8.130 4.5e-07 ***
## conductivity      50187     19610   2.559  0.0210 *
## ms_index      -36556     17062  -2.143  0.0479 *
## conductivity:ms_index -12059     18370  -0.656  0.5209
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 73290 on 16 degrees of freedom
## Multiple R-squared:  0.4662, Adjusted R-squared:  0.3661
## F-statistic: 4.658 on 3 and 16 DF,  p-value: 0.01594

#No hay trivariados significativos

#modelo final seleccionado en esta rama
modelo_conduct2 <- lm(bio_total ~ conductivity + ag_440, data = datos_RH_est)

# Ojo. Las variables conductivity y ag440 están correlacionadas, mientras que conductivity
modelo_conduct2bis <- lm(bio_total ~ conductivity + ms_index, data = datos_RH_est)

```

El modelo final de esta rama `modelo_conduct2` presenta un R^2 ajustado de 51,33%, un AIC de 504.3061 y 17 grados de libertad. La biomasa bacteriana

aumenta con la conductividad y disminuye con ag_440. Ojo: estas variables están algo correlacionadas entre sí (ver análisis de la primera parte). El *modelo_conduct2bis* presenta un R^2 ajustado de 38.73%, un AIC de 508.9188 y 17 grados de libertad

2.5.4 Evaluación de supuestos

Los supuestos se evalúan con un modelo final, ya ajustado. Lamentablemente, si los supuestos no se cumplen, el arduo trabajo que hicimos hasta ahora de seleccionar variables debe ser repetido con otros modelos o estructuras.

El primer supuesto es el de normalidad de los residuos. Podemos ponerlo a prueba con el test de Shapiro-Wilks. Dado que $p > 0.05$, no rechazamos el supuesto de distribución normal. Si este supuesto se rechaza, podríamos probar (por ejemplo) realizando transformaciones a la variable respuesta.

```
shapiro.test(residuals(modelo_fecha4))

##
##  Shapiro-Wilk normality test
##
## data: residuals(modelo_fecha4)
## W = 0.95639, p-value = 0.4744

shapiro.test(residuals(modelo_conduct2))

##
##  Shapiro-Wilk normality test
##
## data: residuals(modelo_conduct2)
## W = 0.96083, p-value = 0.5606
```

El segundo supuesto es el de heterocedasticidad. Lo podemos evaluar con el test de Breusch-Pagan. Dado que $p > 0.05$, no rechazamos el supuesto heterocedasticidad. Si este supuesto se rechaza, existen modelos que permiten incluir una estructura de varianzas particular.

```
library(lmtest)

bptest(modelo_fecha4)

##
## studentized Breusch-Pagan test
```

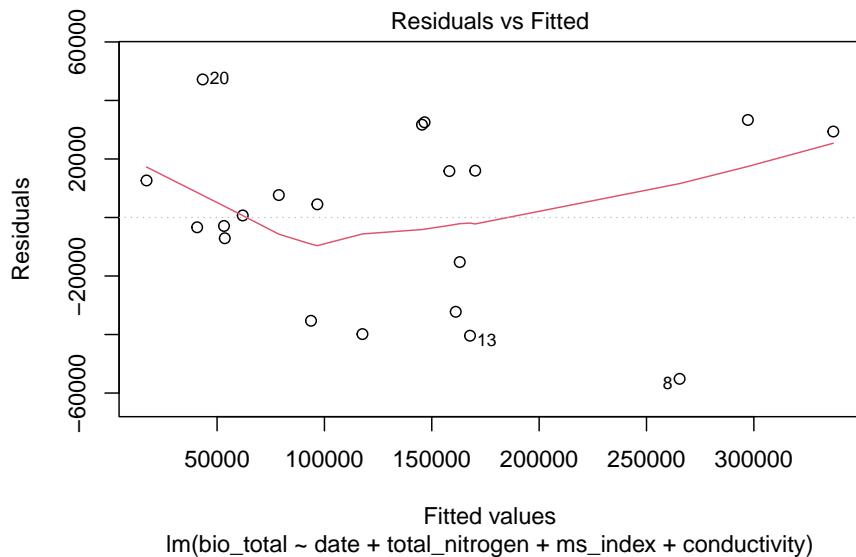
```
##  
## data: modelo_fecha4  
## BP = 7.097, df = 6, p-value = 0.312
```

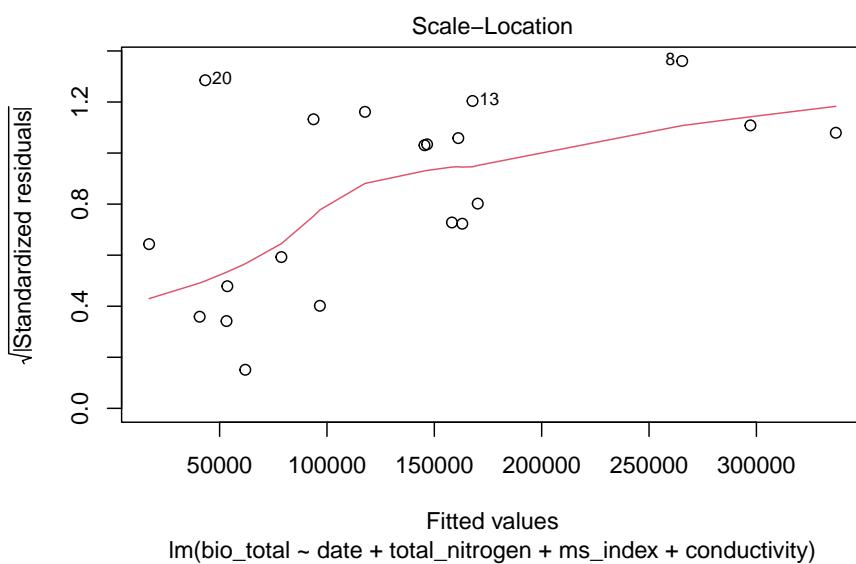
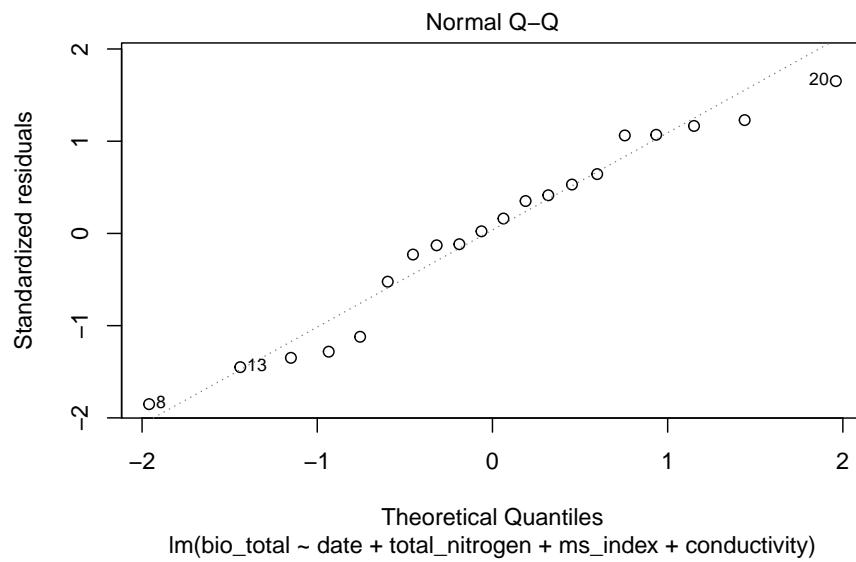
```
bptest(modelo_conduct2)
```

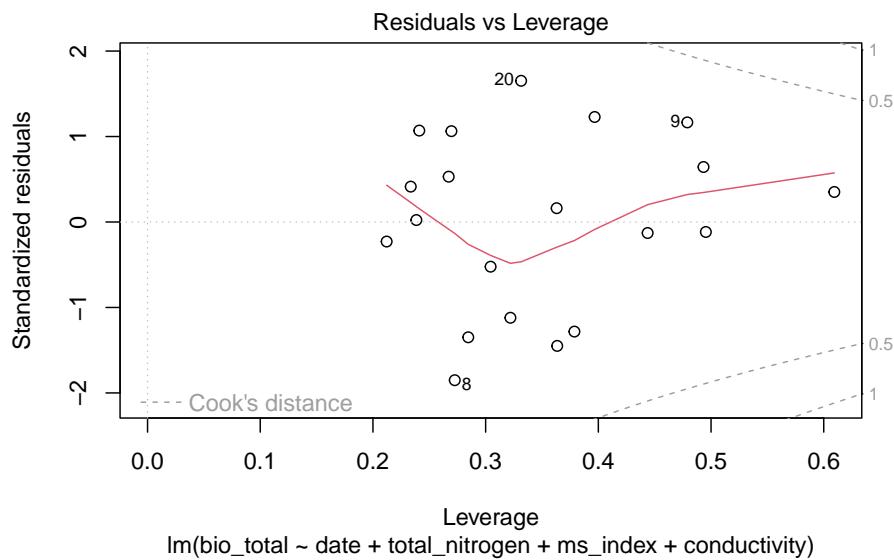
```
##  
## studentized Breusch-Pagan test  
##  
## data: modelo_conduct2  
## BP = 2.4517, df = 2, p-value = 0.2935
```

También podemos obtener algunos gráficos diagnósticos (teclar Enter para pasar de un gráfico al siguiente).

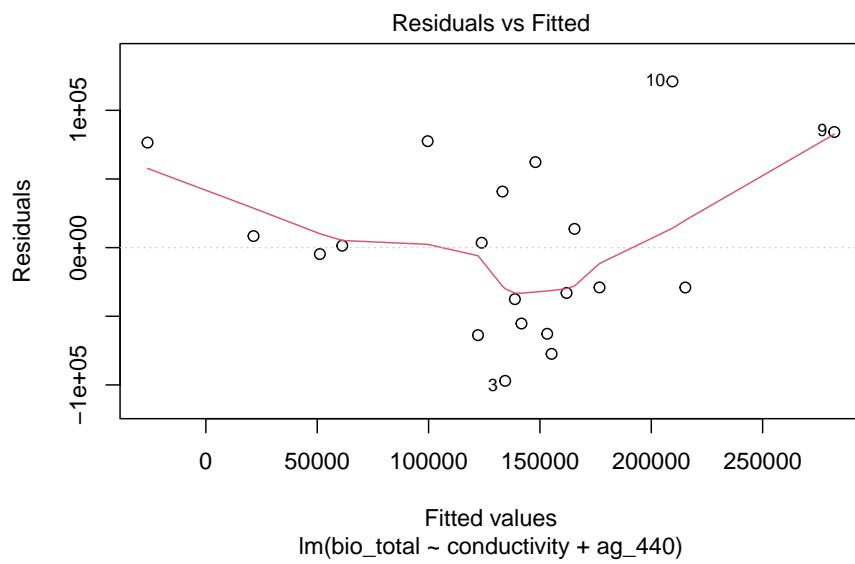
```
plot(modelo_fecha4)
```

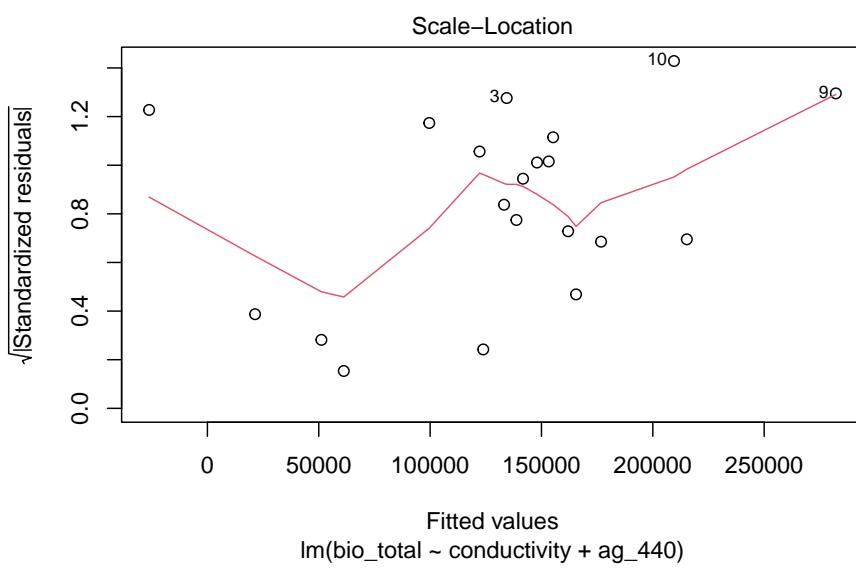
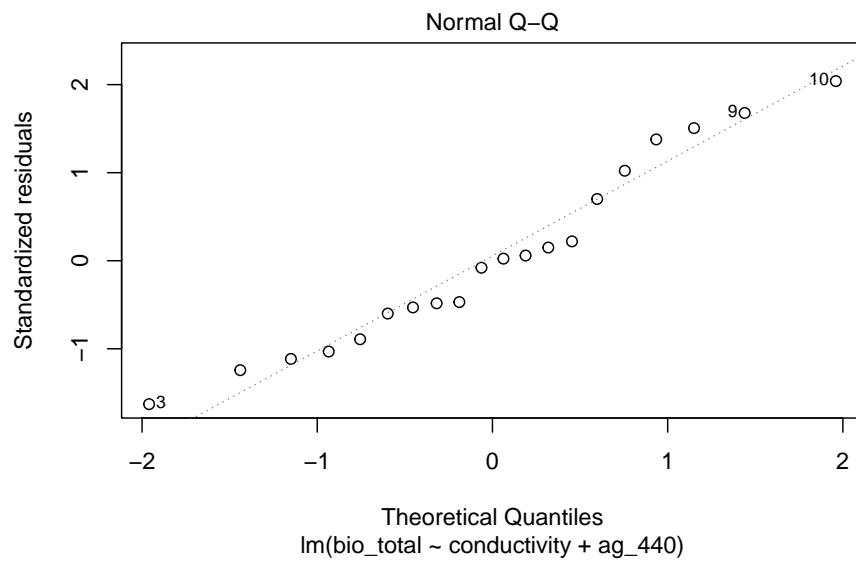


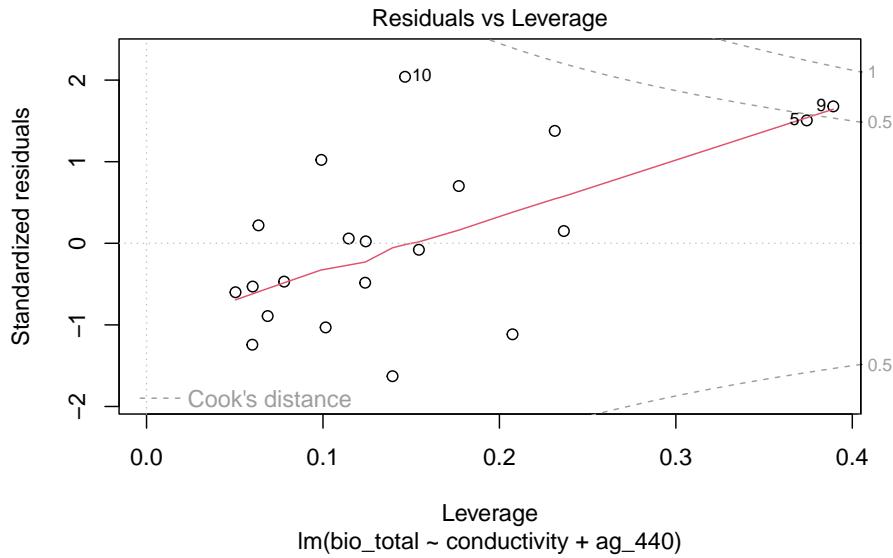




```
plot(modelo_conduct2)
```







2.6 A modo de conclusión

¿Cuál es el mejor modelo? El *modelo_fecha4* tiene mayor poder explicativo, pero incluye la fecha de muestreo (que quizás deberíamos incluir en otro tipo de modelo que considere que hay lagunas repetidas). Sin considerar la fecha de muestreo, un buen modelo es el *modelo_conduct2*: explica más del 50% de la variabilidad en un estudio no experimental, limnológico, a campo (aunque deberíamos discutir si estas variables están muy correlacionadas entre sí). En cualquier caso, el método de selección manual nos dio un conocimiento importante de nuestros datos y nos permitió evaluar las interacciones paso a paso. El procedimiento de selección de variables que vimos en esta Unidad puede ser aplicado a otros tipos de modelos lineales (GLM, GLMM, modelos con estructura de varianzas o modelos no paramétricos).

Como se señaló anteriormente, en este caso de estudio hay medidas repetidas, que podríamos (o deberíamos) considerar en el análisis, incluyendo factores aleatorios como por ejemplo la identidad de las lagunas. Además –y especialmente si tenemos un conjunto grande de cuerpos de agua muestreados–, podemos considerar modelos más complejos que incluyan un componente espacial: ¿la biomasa de las bacterias heterótrofas de las lagunas más cercanas se parece más entre sí?

La mejor manera de entrenarse en este análisis es probando con datos propios.
¿El o los modelos que ajusté tienen sentido biológico?

Chapter 3

Modelos Aditivos Generalizados

María Victoria Quiroga¹

Instituto Tecnológico de Chascomús (INTECH, UNSAM-CONICET), Escuela de Bio y Nanotecnologías (UNSAM)

3.0.1 Aclaración

En esta unidad se utiliza el paquete *ggplot2* (?) para generar algunos gráficos. Para comprender el código empleado se recomienda leer el *Capítulo 3: Data visualization* del libro *R for Data Science* (?).

3.1 ¿Por qué usar Modelos Aditivos Generalizados (GAMs)?

1. La relación entre variables predictoras (i.e., independientes) y la variable respuesta (i.e., dependiente) **no** necesita ser **lineal**.
2. **No** necesitamos conocer de antemano la forma funcional de la relación.
3. Son modelos muy flexibles que permiten la interpretación (de manera gráfica) de los efectos parciales de cada variable independiente.
4. Podemos:
 - Incluir predictores categóricos e interacciones.
 - Usar distribuciones diferentes a la normal para la variable dependiente.

¹mvquirosa@iib.unsam.edu.ar

- Incluir correlaciones entre observaciones (e.g., medidas repetidas, diseños anidados) -modelos mixtos.

3.2 ¿Qué son los GAMs?

Los GAM permiten incorporar los efectos additivos de distintas variables independientes (i.e., explicativas) utilizando funciones suaves (*smooth functions*).

$$g(\mu) = \beta_0 + f_1(x_1) + f_2(x_2) + \dots + f_n(x_n) + \varepsilon$$

Donde μ es el valor esperado de la variable dependiente y que puede presentar distribuciones de la familia exponencial, g es la función de enlace, β_0 es el intercepto, x_n son las n variables independientes y cada f_n es una función suave que se estima de manera no paramétrica. Se pueden incluir funciones suaves para interacciones, por ejemplo entre las variables independientes 1 y 2 $f(x_1, x_2)$. Si se incluyen términos aditivos aleatorios, por ejemplo la autocorrelación en series de tiempo, se generan modelos mixtos (GAMM, *generalized additive mixed model*).

Existen diferentes estrategias para ajustar una curva suave y continua a los datos. Aquí vamos a utilizar *splines*, y en particular *spline* cúbico y cúbico cíclico. De manera muy breve, un *spline* cúbico es una curva suave cúbica por tramos (Figura ??). El *spline* cúbico cíclico empieza y termina en el mismo punto. El paquete con el que vamos a trabajar *mgcv* (?) usa splines penalizados (*conventional integrated square second derivative cubic spline penalty*), donde la penalización será más grande cuanto menos suave sea la curva.

3.3 ¡Manos a la obra!

Utilizaremos un set de datos del trabajo *The dynamics of picocyanobacteria from a hypereutrophic shallow lake is affected by light-climate and small-bodied zooplankton: a 10-year cytometric time-series analysis* publicado en *FEMS Microbiology Ecology* (?), disponibles en el Repositorio Institucional CONICET Digital.

Descargar el set de datos **data_gam.csv** de GitHub Limno-con-R/CILCAL2023. Guardar el archivo en una carpeta llamada *data*, dentro del **Directorio de Trabajo del Proyecto** que creamos para esta Unidad (ver cómo hacerlo en la Unidad ??).

Instalar los paquetes como se indica en la Unidad ?. Luego, cargarlos en la sesión

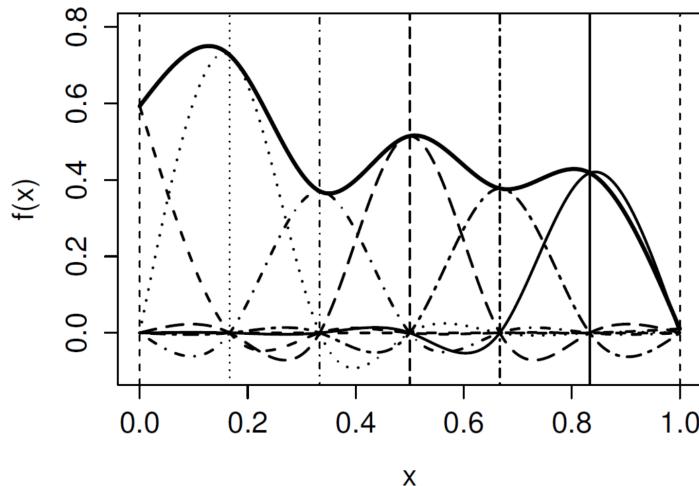


Figure 3.1: Construcción de un spline cúbico. La curva suave (línea continua gruesa) es la suma de las 5 funciones basis (líneas finas). Las líneas verticales muestran los nodos equiespaciados. Extraído de Wood (2017).

```
library(mgcv)
library(ggplot2)
library(itsadug)
library(data.table)
library(car)
library(grid)
library(lubridate)
```

Leer datos, dar formato `as.Date` a la fecha y generar las variables `Mes` y `Dias`.

```
base <- read.csv("data/data_gam.csv")
base$Fecha <- as.Date(base$Fecha, "%m/%d/%Y") # formato de fecha en inglés
base$Mes <- as.numeric(format(base$Fecha, '%m')) # generar variable Mes
base$Dia1 = rep(base$Fecha[1], nrow(base)) # fecha inicial de la serie
base$Dias <- (interval(base$Dia1, base$Fecha) %/ days(1))+1 # generar variable Dias
```

Inspección visual de primeras filas de la tabla. Para ver toda la tabla se puede utilizar `View(base)`.

```
head(base)
```

```
##           Fecha Pcy_orgml Mes      Dia1 Dias
## 1 2007-05-22    6660000   5 2007-05-22     1
```

```
## 2 2007-06-05 6830000 6 2007-05-22 15
## 3 2007-06-21 4700000 6 2007-05-22 31
## 4 2007-07-03 4680000 7 2007-05-22 43
## 5 2007-07-17 6500000 7 2007-05-22 57
## 6 2007-07-31 8110000 7 2007-05-22 71
```

Ver la estructura de los datos

```
str(base)
```

```
## 'data.frame': 225 obs. of 5 variables:
## $ Fecha : Date, format: "2007-05-22" "2007-06-05" ...
## $ Pcy_orgml: num 6660000 6830000 4700000 4680000 6500000 ...
## $ Mes : num 5 6 6 7 7 7 8 8 9 9 ...
## $ Dia1 : Date, format: "2007-05-22" "2007-05-22" ...
## $ Dias : num 1 15 31 43 57 71 85 99 113 127 ...
```

`base` es un objeto `data.frame` con 225 `obs.` observaciones o filas y 5 `variables` o columnas. Variables: `Fecha` con formato `Date`. `Pcy_orgml`, abundancia de picocianobacterias (organismos/ml) en la laguna Chascomús, como valores numéricos `num`.

Se generaron las variables `Mes` y `Dias` como números `num`. `Mes` indica el #mes (1-12) de la fecha de muestreo y `Dias` indica el #días transcurridos desde la primer fecha (considerando la primer fecha como día #1).

Grafico exploratorio de la serie temporal

```
ggplot(base, aes(x= Dias, y= Pcy_orgml)) +
  geom_point() +
  geom_line()+
  theme(legend.position = "none")
```

No se observan outliers en el gráfico.

Para datos de conteo (e.g., organismos/ml) generalmente se utiliza la distribución de Poisson, pero las series temporales de abundancia de microorganismos suelen presentar sobredispersión (Figura ??). Para considerar una varianza mayor que la media se podría utilizar la distribución binomial negativa o la *quasi-familia* quasipoisson. En este ejemplo se implementará `family=quasipoisson`.

La autocorrelación temporal de los datos se incluye utilizando una estructura autorregresiva de primer orden continua `correlation = corCAR1(form = ~ Dias)` en un modelo mixto `gamm()`.

Como primer modelado se generan funciones suaves `s()` para el efecto estacional `Mes` y el efecto interanual (*trend*) `Dias`. Se utiliza *spline* cúbico cíclico para

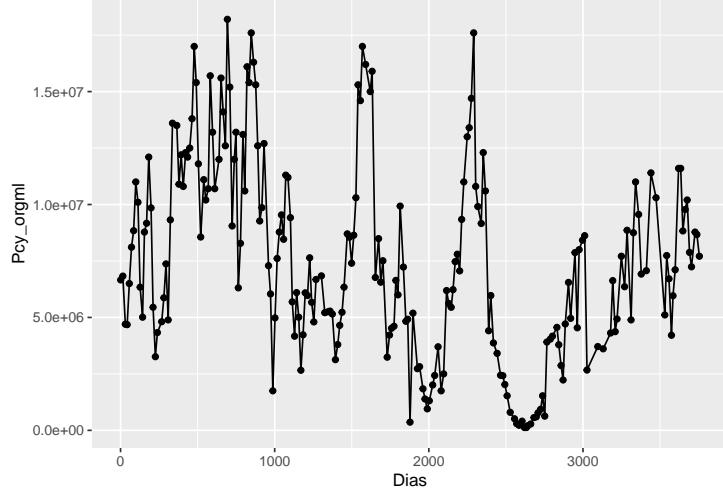


Figure 3.2: Serie temporal de abundancia (organismos/ml) de picocianobacterias.

el primer efecto `s(Mes, bs="cc")` y cúbico para el segundo efecto `s(Dias, bs="cr")`.

```
modelo1<-gamm(Pcy_orgml ~ s(Mes, bs = "cc") + s(Dias, bs="cr"),
family=quasipoisson, data = base,
correlation = corCAR1(form = ~ Dias))
```

```
##
## Maximum number of PQL iterations: 20

## iteration 1

## iteration 2

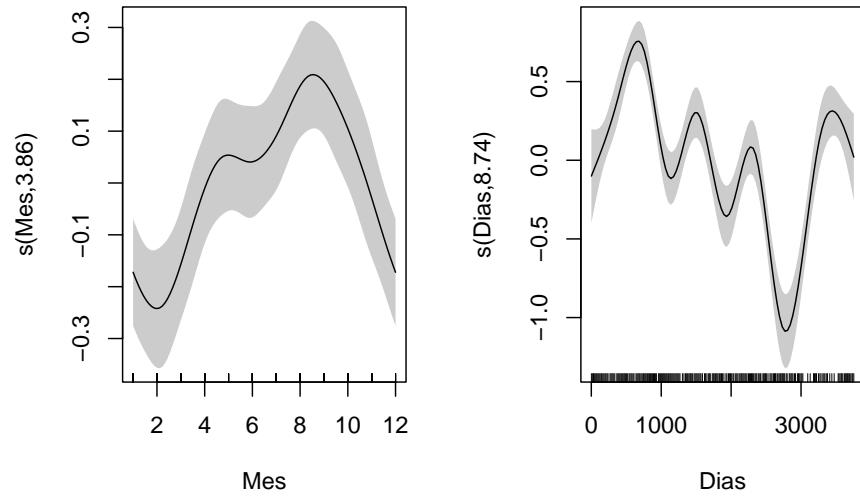
## iteration 3

## iteration 4
```

Por *default* el máximo número de iteraciones está seteado en 20 `niterPQL=20`. Si el modelo no converge se puede incrementar el número de interacciones, por ejemplo: `gamm(..., niterPQL=40)`.

Interpretación visual de los efectos parciales: curvas suaves `s()` (*smooth functions*)

```
plot(modelo1$gam, scale=0, scheme=1, pages=1)
```



\begin{figure}
\caption{Efectos parciales del modelo 1. Las curvas suaves se centraron en cero, se indican los intervalos de confianza de 95% en gris. Las líneas internas en los ejes x (Mes y Dias) representan los datos.} \end{figure}

Información del modelo

```
summary(modelo1$gam)
```

```
##  

## Family: quasipoisson  

## Link function: log  

##  

## Formula:  

## Pcy_orgml ~ s(Mes, bs = "cc") + s(Dias, bs = "cr")  

##  

## Parametric coefficients:  

##             Estimate Std. Error t value Pr(>|t|)  

## (Intercept) 15.72716    0.02977  528.2   <2e-16 ***  

## ---  

## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1  

##  

## Approximate significance of smooth terms:  

##             edf Ref.df      F  p-value
```

```

## s(Mes) 3.858 8.000 3.674 3.46e-06 ***
## s(Dias) 8.743 8.743 23.752 < 2e-16 ***
## ---
## Signif. codes: 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## R-sq.(adj) = 0.564
## Scale est. = 1.209e+06 n = 225

```

Se especifica la familia que se utilizó `## Family: quasipoisson` y la función de enlace `## Link function: log`. `?family` muestra las familias y funciones de enlace que se utilizan por *default*. Para cambiar la función de enlace solo hay que especificarlo en el código, ejemplo `family = Gamma(link = "log")`. Si no se especifica una familia, `gamm` usa distribución gaussiana con función de enlace identidad. La ayuda `?gamm` muestra todos los argumentos y los *defaults*.

Se observa que ambos términos de suavizado son significativos ($p\text{-value} < 0.05$), y el R^2_{adj} del modelo es 0.5640034.

Evaluación del modelo

```

windows()
par(mfrow=c(2,2))
gam.check(modelo1$gam, type="pearson")

##
## 'gamm' based fit - care required with interpretation.
## Checks based on working residuals may be misleading.
## Basis dimension (k) checking results. Low p-value (k-index<1) may
## indicate that k is too low, especially if edf is close to k'.
##
##          k'  edf k-index p-value
## s(Mes) 8.00 3.86    0.93    0.16
## s(Dias) 9.00 8.74    0.31 <2e-16 ***
## ---
## Signif. codes: 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

```

Los gráficos de la izquierda de la Figura ?? muestran que es correcto utilizar la *quasi familia* quasipoisson. El gráfico superior derecho muestra una dispersión bastante homogénea de los residuos, por lo que no habría mayores problemas con la varianza. El gráfico inferior derecho muestra una relación lineal positiva entre los valores observados y los predichos por el modelo. Si se logra mejorar el modelo, se observará una mejoría en este gráfico y en el R^2_{adj} del modelo.

En el modelo 1 no se especificó el `k` en los términos de suavizado `s()`, se utilizó la opción `default k = 10`. Este argumento especifica la dimensión de las funciones

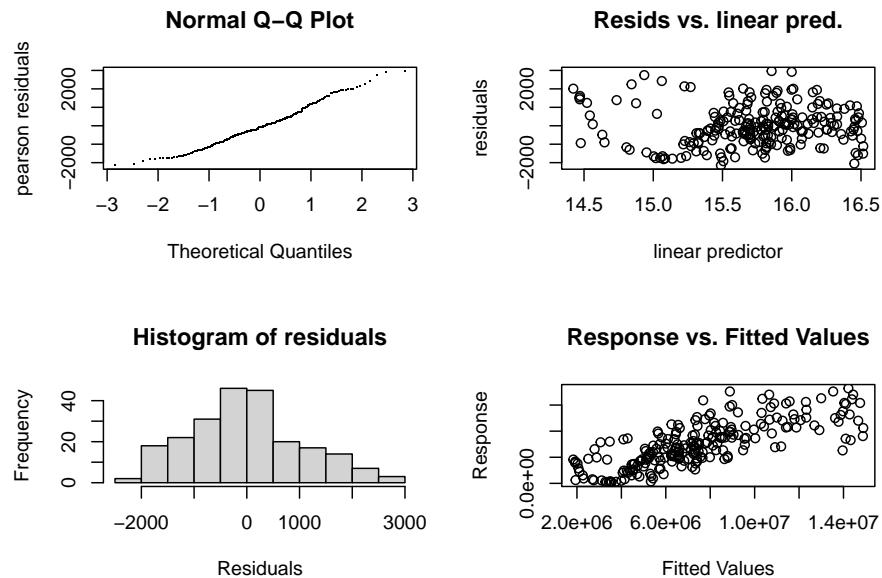


Figure 3.3: gam.check del modelo 1.

basis del *spline*. Cuando se indica un valor de *k*, este determina el máximo grado de libertad permitida para ese término del modelo. Sin embargo, los grados de libertad efectivos *edf* para cada término los estima el modelo a través de la penalización, siendo el límite superior $k' = k - 1$. No especificar el argumento *k* equivale a indicar *k* = 10.

El *gam.check()* nos indica que el *k* del término *s(Dias)* del modelo 1 es muy bajo (valor de *p-value* bajo y *edf* cercano a *k'*). Ver *?choose.k* para mas detalles.

Para mejorar el modelo se modifica *k*. Se disminuye *k* = 6 en *s(Mes)* para evitar que incrementen los *edf* de ese término, y se aumenta *k* = 20 en *s(Dias)*. Además, se agrega un término de interacción *ti()* adecuando para trabajar con variables que tienen diferentes unidades (meses *versus* días).

```
modelo2<-gamm(Pcy_orgml ~ s(Mes, bs = "cc", k=6) + s(Dias, bs="cr", k=20)
                + ti(Mes,Dias, bs=c("cc","cr")), family=quasipoisson,
                data = base, correlation = corCAR1(form = ~ Dias))

## 
## Maximum number of PQL iterations: 20

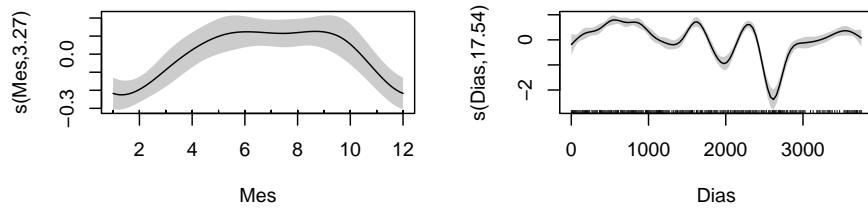
## iteration 1
```

```
## iteration 2
```

```
## iteration 3
```

Interpretación visual de los efectos parciales: curvas suaves `s()` (*smooth functions*)

```
plot(modelo2$gam, scale=0, scheme=1, pages=1)
```



```
\begin{figure}
\caption{Efectos parciales del modelo 2. Las curvas suaves se centraron en cero, se indican los intervalos de confianza de 95% en gris. Las líneas internas en los ejes x (Mes y Dias) representan los datos.} \end{figure}
```

La interacción se muestra en 3D, y es difícil de interpretar visualmente. Aquí nos centramos en la interpretación visual de los efectos parciales.

Información del nuevo modelo

```
summary(modelo2$gam)
```

```
##
## Family: quasipoisson
## Link function: log
##
## Formula:
```

```

## Pcy_orgml ~ s(Mes, bs = "cc", k = 6) + s(Dias, bs = "cr", k = 20) +
##      ti(Mes, Dias, bs = c("cc", "cr"))
##
## Parametric coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) 15.64156   0.02398  652.3 <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Approximate significance of smooth terms:
##             edf Ref.df      F p-value
## s(Mes)       3.268   4.00 9.085 <2e-16 ***
## s(Dias)      17.539  17.54 28.940 <2e-16 ***
## ti(Mes,Dias) 3.718  12.00  0.595  0.0395 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## R-sq.(adj) =  0.753
## Scale est. = 5.8774e+05 n = 225

```

Los efectos parciales de `s(Mes)` y `s(Dias)` son significativos, y la interacción `ti(Mes,Dias)` es significativa. El R^2_{adj} del modelo aumentó a 0.7532274.

Evaluación del modelo

```
windows()
par(mfrow=c(2,2))
gam.check(modelo2$gam, type="pearson")
```

```

##
## 'gamm' based fit - care required with interpretation.
## Checks based on working residuals may be misleading.
## Basis dimension (k) checking results. Low p-value (k-index<1) may
## indicate that k is too low, especially if edf is close to k'.
##
##          k'    edf k-index p-value
## s(Mes)     4.00  3.27    0.84   0.010 **
## s(Dias)    19.00 17.54    0.66 <2e-16 ***
## ti(Mes,Dias) 12.00  3.72    0.89   0.025 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

```

De nuevo, parece correcto utilizar la *quasi familia* quasipoisson, sin mayores problemas con la varianza. La relación lineal positiva entre los valores observados y los predichos (gráfico inferior derecho) ha mejorado, coincidiendo con el incremento del R^2_{adj} del modelo.

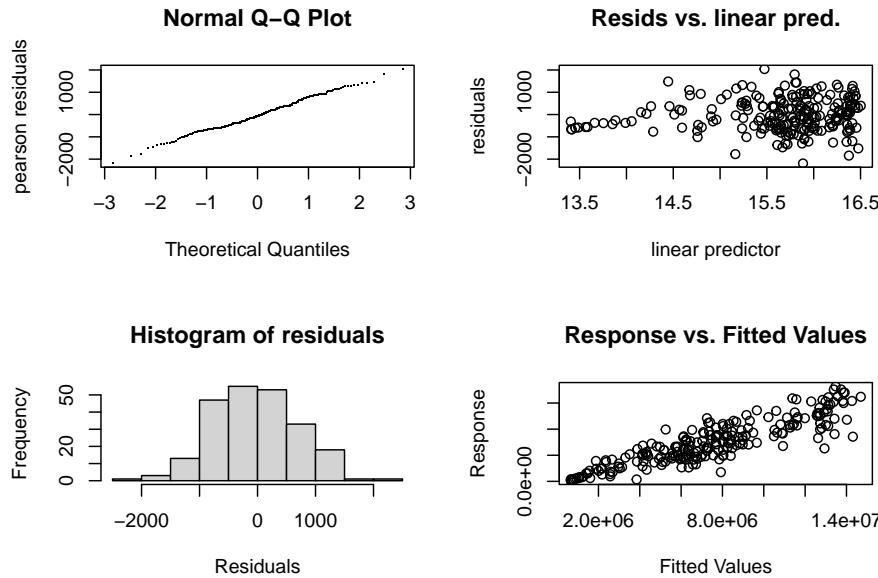


Figure 3.4: gam.check del modelo 2.

Los `edf` del término `s(Mes)` son cercanos a 3, muy similares al primer modelo, lo cual corrobora que la especificación de `k` en el modelo no es crítica (modelo 1 $k = 10$, modelo 2 $k = 6$), siempre que se considere un límite superior adecuado. Para `s(Dias)` el `p-value` sigue siendo bajo y `edf` sigue cercano a k' , pero `k-index` aumentó a 0.66. Se considera que no es necesario aumentar el `k` porque complejizaría el modelo, y consecuentemente la interpretación visual de la tendencia (*trend*, efecto `Dias`).

Se pueden comparar ambos modelos estimando su AIC

```
AIC(modelo1$lme, modelo2$lme)
```

```
##          df      AIC
## modelo1$lme  6 311.8362
## modelo2$lme  8 228.2012
```

Efectivamente el modelo 2 presenta un AIC más bajo.

Otros gráficos

Se puede graficar con `vis.gam()`.

```
vis.gam(modelo2$gam, type="response",
        view=c("Mes","Dias"), main="Pcy (org/ml)",
        plot.type = "contour", contour.col="black",color="terrain")
points(base$Mes,base$Dias,pch = 20) #agregar puntos de muestreo
```

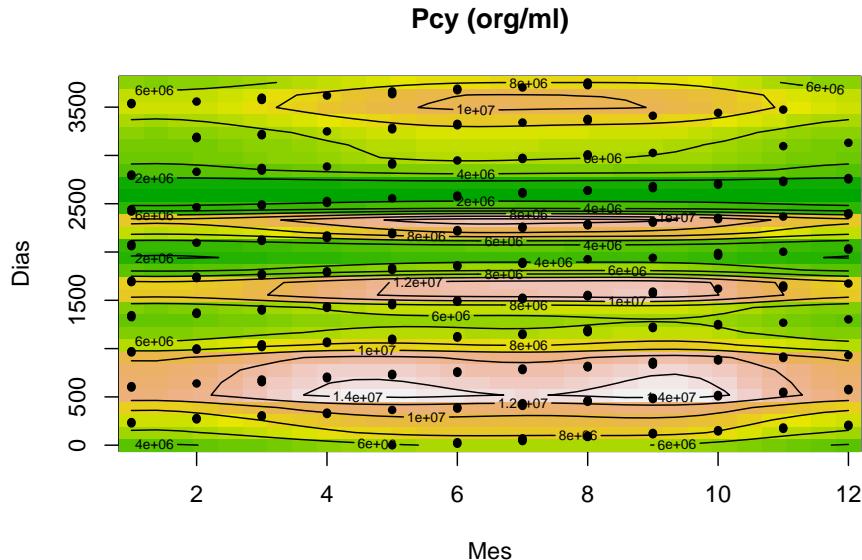


Figure 3.5: vis.gam plot del modelo 2.

Se pueden graficar los valores observados a campo y los predichos por el modelo. Cuanto mas alto sea el R^2_{adj} , mas similares serán ambos valores.

```
# generar datos para el gráfico
Base<-as.data.table(base)
datas <- rbindlist(list(Base[, .(Pcy_orgml, Fecha)],
                        data.table(Pcy_orgml = (modelo2$gam)$fitted.values,
                                   Fecha = Base[, Fecha])))
datas[, datos := c(rep("Observados", nrow(Base)),
                  rep("Predichos", nrow(Base)))]
```

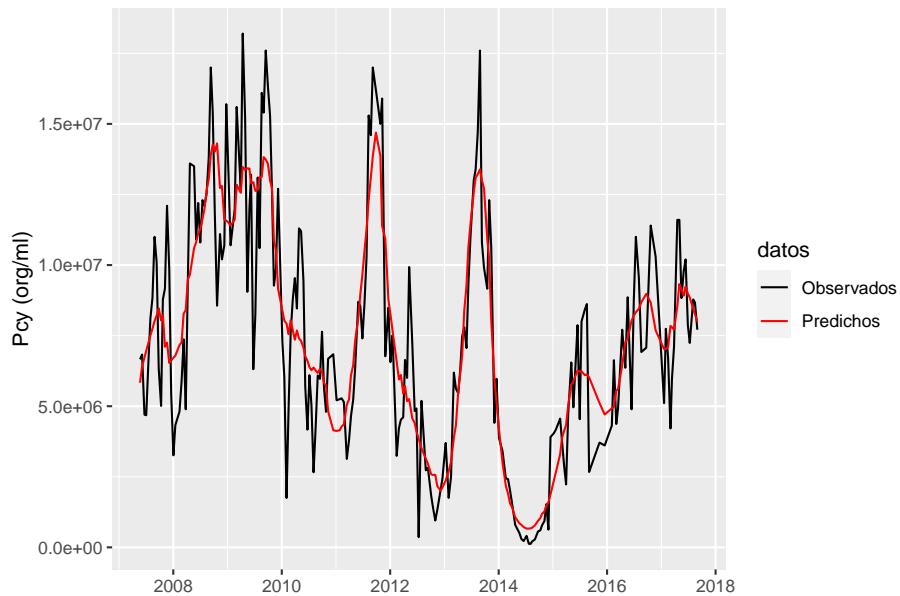


```
# generar el gráfico
ggplot(data = datas, aes(Fecha,Pcy_orgml, group = datos))+
```

```
geom_line(aes(colour = datos))+
```

```
scale_color_manual(values=c("black","red"))+
```

```
labs(x = "", y = "Pcy (org/ml)")
```



3.4 Agradecimientos

Un agradecimiento muy especial para los Profesores **Gerardo Cueto y Adriana Pérez** de la FCEN-UBA.

Chapter 4

Análisis multivariados I

Patricia E. García¹

Instituto de Investigaciones en Biodiversidad y Medioambiente (INIBIOMA,
CONICET-UNComahue)

En esta unidad se verán ejemplos de ACP, NMDS, PERMANOVA

4.1 Análisis de Componentes Principales

Introducción: Algunos sistemas no pueden definirse por sus características individuales, sino que deben verse en conjunto. Un ejemplo muy sencillo para entender este concepto es intentar describir una cara humana con un solo rasgo: la nariz.

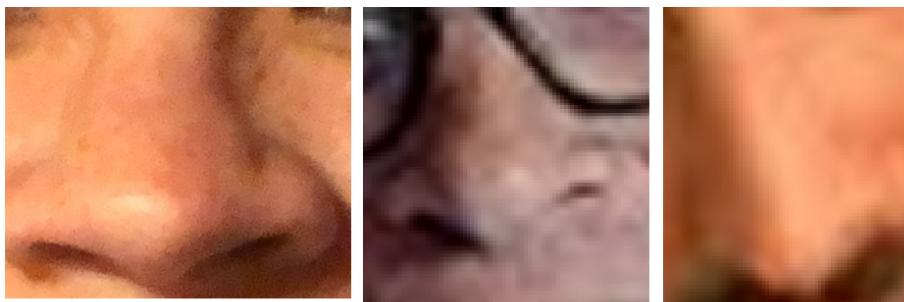


Figure 4.1: Se puede identificar una cara solo describiendo la nariz?

¹garciapatriciaelizabeth@gmail.com

Para describir y reconocer una cara es necesario observar todas sus características: la nariz, los ojos, la boca, las cejas, etc. Es decir, múltiples dimensiones.

El **análisis de componentes principales (ACP)** también conocido por siglas en inglés Principal Component Analysis PCA, es una poderosa herramienta estadística para describir un conjunto de datos.

Se basa en la reducción de la multidimensionalidad de una manera “*no supervisada*”.

Su principal objetivo es proyectar datos en las direcciones de **máxima varianza** y de este modo eliminar aquellas direcciones o planos que aporten menos información.

El ACP puede aplicarse antes de realizar un análisis de regresión o simplemente con fines exploratorios para ayudar a los investigadores a comprender relaciones entre las variables o descubrir patrones presentes en los datos.

4.1.1 ¿Qué tipos de datos usa una ACP?

Utiliza principalmente variable *numéricas continuas* que son las que efectivamente entran en el análisis. Estás variables se organizan en **columnas**, mientras que en las **filas** se organizan los individuos.

Individuos	Variable 1	Variable 2	Variable 3
1	15	0.6	70
2	25	0.1	50
3	24	0.2	30
4	13	0.3	55
5	23	0.5	54
6	22	0.4	31
7	19	0.1	87

Figure 4.2: Organización de los datos

Estas bases de datos son de doble entrada, por lo que es posible analizarla desde el punto de vista de los individuos, de tal manera que *individuos* que tiene características similares estarán más cerca.

Desde el punto de las *variables*, se pueden analizar las relaciones entre las mismas. El análisis considera relaciones lineales entre las variables y usa coeficientes de correlación. Es importante utilizar la matriz de correlación para observar las relaciones más importantes entre las variables.

4.1.2 ¿Cómo funciona el análisis?

El funcionamiento del análisis es muy sencillo, trata de buscar la *dirección o componente* que maximice la variación de datos. Es decir que el primer plano/componente trata de capturar la mayor dispersión de los datos.

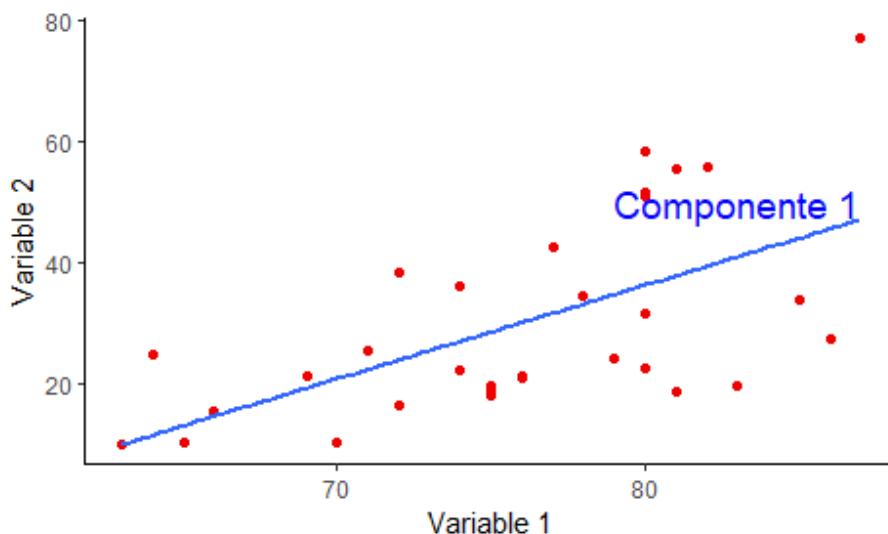
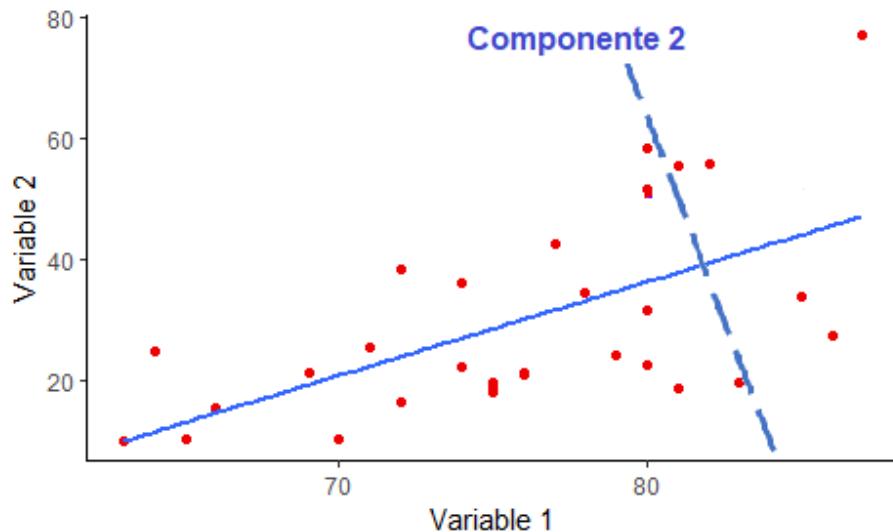


Figure 4.3: El primer componente

En el ACP, luego que se encontró el primer componente que maximizó la dispersión de los datos, el segundo componente es ortogonal, es decir a 90° .



4.1.3 Ahora manos a la obra: miremos datos

En RStudio importamos la base de datos ambientales de Castro Berman et al., 2018, disponible como **data_cursoR.xlsx** en GitHub Limno-con-R/CILCAL2023. Recuerden guardar el archivo en una carpeta llamada ***data***, dentro del **Directorio de Trabajo del Proyecto** que creamos para esta Unidad (ver cómo hacerlo en la Unidad ??).

```
library(readxl)
datos <- read_excel("data/data_cursoR.xlsx",
  col_types = c("text", "numeric", "numeric",
    "numeric", "numeric", "numeric",
    "numeric", "numeric", "numeric",
    "numeric", "numeric", "numeric",
    "numeric", "numeric", "numeric"),
  na = "NA")

## New names:
## * `` -> `...1` 

datos <- data.frame(datos, row.names = 1)
head(datos)

##          ELEVACION LATITUD LONGITUD Temp Secch
```

```

## SL_13      62 -38.4184 -61.7594   17    44 11.8 8.55      0.32 15.7 3438
## SL_17      20 -38.9263 -61.3687    7     8 8.0 8.53      3.43 363.0 4010
## SL_18     119 -38.3474 -60.4527   14    12 9.0 8.84      4.05 110.0 4200
## SL_19      89 -38.4273 -60.3181   14    24 10.8 8.54      1.10 54.4 4738
## SL_20      64 -38.5216 -60.1404   10     9 9.0 8.44      3.25 206.0 7470
## SL_21     194 -38.0508 -60.0340   11    16 10.0 8.30      3.35 129.0 4256
##      TDON    SPM    POM     TP     TDP   Chl.a     DOC
## SL_13 2565 11.7   9.2   124    0.1   3.79   4.55
## SL_17 2890 377.5 103.4  272  114.0  68.63 10.98
## SL_18 3909 71.0   23.4  1009  833.0  54.36 253.50
## SL_19 4424 30.9   23.4   162    4.0   66.75 10.12
## SL_20 5107 186.6  62.5   727  456.0  475.36 95.43
## SL_21 3237 111.0  33.4  1365 1363.0  61.81 177.70

```

La base de datos que importamos contiene información acerca de la localización geográfica de los sitios y las variables físico-químicas.

```
summary(datos)
```

```

##      ELEVACION        LATITUD        LONGITUD       Temp
##  Min.   :-16.00   Min.   :-38.93   Min.   :-63.09   Min.   : 7.00
##  1st Qu.: 10.00  1st Qu.:-37.49  1st Qu.:-62.25  1st Qu.:15.00
##  Median : 58.50  Median :-36.84  Median :-60.23  Median :17.50
##  Mean   : 73.65  Mean   :-36.62  Mean   :-60.13  Mean   :18.52
##  3rd Qu.:104.00 3rd Qu.:-35.69  3rd Qu.:-57.98  3rd Qu.:22.25
##  Max.   :246.00  Max.   :-34.48  Max.   :-56.98  Max.   :30.00
##
##      Secchi         DO          pH      Conductivity
##  Min.   : 7.00  Min.   : 5.000  Min.   :8.000  Min.   : 0.320
##  1st Qu.:12.00 1st Qu.: 8.725  1st Qu.:8.557  1st Qu.: 1.215
##  Median :18.00  Median :10.300  Median :8.795  Median : 2.615
##  Mean   :29.44  Mean   :10.238  Mean   :8.762  Mean   : 7.968
##  3rd Qu.:37.75 3rd Qu.:11.050  3rd Qu.:8.982  3rd Qu.: 5.513
##  Max.   :140.00  Max.   :20.000  Max.   :9.400  Max.   :202.100
##
##      NTU          TON        TDON        SPM
##  Min.   : 2.70  Min.   : 2856  Min.   :1926  Min.   : 6.70
##  1st Qu.:27.90 1st Qu.: 4104  1st Qu.:3052  1st Qu.: 36.20
##  Median :83.25  Median : 4872  Median :3478  Median : 75.45
##  Mean   :88.67  Mean   : 5063  Mean   :3790  Mean   :116.83
##  3rd Qu.:110.00 3rd Qu.: 5323  3rd Qu.:4356  3rd Qu.:128.53
##  Max.   :363.00  Max.   :10830  Max.   :7235  Max.   :788.40
##
##      POM          TP        TDP        Chl.a
##  Min.   : 6.70  Min.   : 46.0  Min.   : 0.1  Min.   : 1.58

```

```

## 1st Qu.: 20.00 1st Qu.: 324.0 1st Qu.: 114.0 1st Qu.: 15.80
## Median : 29.60 Median : 533.0 Median : 302.0 Median : 52.05
## Mean    : 40.80 Mean   : 766.1 Mean   : 560.3 Mean   : 87.80
## 3rd Qu.: 47.92 3rd Qu.: 820.5 3rd Qu.: 624.0 3rd Qu.: 89.13
## Max.    :344.00 Max.   :4538.0 Max.   :4140.0 Max.   :981.06
##
##          DOC
## Min.    : 1.26
## 1st Qu.: 12.36
## Median : 19.45
## Mean   : 93.22
## 3rd Qu.: 97.58
## Max.   :1010.00
## NA's   :1

##      ELEVACION      LATITUD      LONGITUD      Temp
## Min.   :-16.00  Min.   :-38.93  Min.   :-63.09  Min.   : 7.00
## 1st Qu.: 10.00  1st Qu.: -37.49  1st Qu.: -62.25  1st Qu.:15.00
## Median : 58.50  Median : -36.84  Median : -60.23  Median :17.50
## Mean   : 73.65  Mean   : -36.62  Mean   : -60.13  Mean   :18.52
## 3rd Qu.:104.00  3rd Qu.: -35.69  3rd Qu.: -57.98  3rd Qu.:22.25
## Max.   :246.00  Max.   : -34.48  Max.   : -56.98  Max.   :30.00
##
##          Secchi        DO        pH      Conductivity
## Min.    : 7.00  Min.   : 5.000  Min.   :8.000  Min.   : 0.320
## 1st Qu.: 12.00  1st Qu.: 8.725  1st Qu.:8.557  1st Qu.: 1.215
## Median : 18.00  Median :10.300  Median :8.795  Median : 2.615
## Mean   : 29.44  Mean   :10.238  Mean   :8.762  Mean   : 7.968
## 3rd Qu.: 37.75  3rd Qu.:11.050  3rd Qu.:8.982  3rd Qu.: 5.513
## Max.   :140.00  Max.   :20.000  Max.   :9.400  Max.   :202.100
##
##      NTU         TON       TDON      SPM
## Min.   : 2.70  Min.   :2856  Min.   :1926  Min.   : 6.70
## 1st Qu.: 27.90 1st Qu.:4104  1st Qu.:3052  1st Qu.: 36.20
## Median : 83.25 Median :4872  Median :3478  Median : 75.45
## Mean   : 88.67 Mean   :5063  Mean   :3790  Mean   :116.83
## 3rd Qu.:110.00 3rd Qu.:5323  3rd Qu.:4356  3rd Qu.:128.53
## Max.   :363.00 Max.   :10830  Max.   :7235  Max.   :788.40
##
##      POM         TP       TDP      Chl.a
## Min.   : 6.70  Min.   : 46.0  Min.   : 0.1  Min.   : 1.58
## 1st Qu.: 20.00 1st Qu.: 324.0 1st Qu.: 114.0 1st Qu.: 15.80
## Median : 29.60 Median : 533.0 Median : 302.0 Median : 52.05
## Mean   : 40.80 Mean   : 766.1 Mean   : 560.3 Mean   : 87.80
## 3rd Qu.: 47.92 3rd Qu.: 820.5 3rd Qu.: 624.0 3rd Qu.: 89.13
## Max.   :344.00 Max.   :4538.0 Max.   :4140.0 Max.   :981.06

```

```
##      DOC
##  Min.   :  1.26
##  1st Qu.: 12.36
##  Median : 19.45
##  Mean   : 93.22
##  3rd Qu.: 97.58
##  Max.   :1010.00
##  NA's    :1
```

En resumen, tenemos 52 observaciones (individuos) y 17 variables. De las variables presentes las primeras 3 (Elevación, Longitud y Latitud) corresponden a la localización de los ambientes.

Las variables son continuas y cada una está medida de manera distintas por lo que cada una tiene su unidad y a su vez su variación. Debido a esto es necesario **estandarizar** y **centrar** las variables. La estandarización simplemente le resta a cada observación el promedio de la variable y la divide por la desviación estandar de la variable. Mientras que centrar significa que se mueve la nube de puntos de los individuos al centro de gravedad.

```
?scale # el comando para estandarizar los datos
```

```
## starting httpd help server ... done
```

```
datos.stand <- data.frame(scale(datos [4:17], center = T))
```

Para este ejemplo solo usé las variables físico-químicas, para ello utilicé los corchetes e indiqué el número de columnas que quería se estandarizaran, de la columna 4 a la columna 17: [4:17].

El comando para realizar el análisis de componentes principales , está en el paquete FactoMiner. Este paquete es muy versatil y se mantiene actualizado. Aquí adjunto el link:

FactoMineR

```
library(FactoMineR)
```

```
## Warning: package 'FactoMineR' was built under R version 4.2.3
```

```
?PCA ## ayuda del comando PCA
```

```
# Realizar el análisis solo en las variables numéricas continuas
```

```
res.pca <- PCA(datos.stand [-2], graph=FALSE) # se guarda en el objeto res.pca (uso res para indi
```

```
## Warning in PCA(datos.stand[-2], graph = FALSE): Missing values are imputed by
## the mean of the variable: you should use the imputePCA function of the missMDA
## package
```

¡Recibimos un *mensaje de advertencia!* Este mensaje nos indica que hay valores faltantes y que el comando va a usar el promedio de la variable en las celdas donde faltan datos.

Importante: algunos análisis son muy sensibles a la falta de datos.

¡Listo hemos realizado el análisis de ACP! El objeto “res.pca” es una lista que tiene toda la información.

```
print(res.pca)
```

```
## **Results for the Principal Component Analysis (PCA)**
## The analysis was performed on 52 individuals, described by 13 variables
## *The results are available in the following objects:
##
##      name           description
## 1  "$eig"          "eigenvalues"
## 2  "$var"          "results for the variables"
## 3  "$var$coord"    "coord. for the variables"
## 4  "$var$cor"      "correlations variables - dimensions"
## 5  "$var$cos2"     "cos2 for the variables"
## 6  "$var$contrib"  "contributions of the variables"
## 7  "$ind"          "results for the individuals"
## 8  "$ind$coord"    "coord. for the individuals"
## 9  "$ind$cos2"     "cos2 for the individuals"
## 10 "$ind$contrib"  "contributions of the individuals"
## 11 "$call"          "summary statistics"
## 12 "$call$centre"   "mean of the variables"
## 13 "$call$ecart.type" "standard error of the variables"
## 14 "$call$row.w"    "weights for the individuals"
## 15 "$call$col.w"    "weights for the variables"
```

4.1.3.1 Análisis de los resultados

El ACP calcula los autovalores (eigenvalues) y autovectores propios a partir de la matriz de covarianzas. El cálculo de los vectores depende de la cantidad de dimensiones de los datos. Los autovalores no son más que la magnitud de los autovectores, ambos ayudan a calcular los Componentes principales.

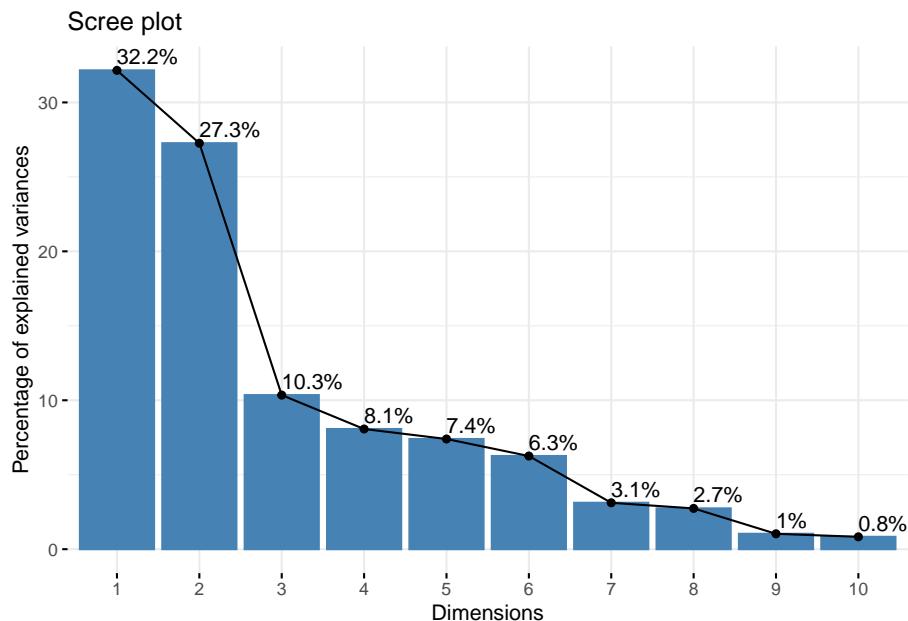
```
eigenvalues <- res.pca$eig# eigenvalues
head(eigenvalues[, 1:3])
```

```
##          eigenvalue percentage of variance cumulative percentage of variance
## comp 1  4.1797364           32.151818                  32.15182
## comp 2  3.5439409           27.261084                  59.41290
## comp 3  1.3449952           10.346117                 69.75902
## comp 4  1.0490855            8.069888                 77.82891
## comp 5  0.9618192            7.398609                 85.22752
## comp 6  0.8132415            6.255704                 91.48322
```

En la primer columna se observa el valor del eigenvalue, en la segunda columna se observa el porcentaje de variación explicada. En la tercera columna se muestra el porcentaje de variación acumulada.

Un paquete útil para mejorar los gráficos de los análisis multivariados es **factoextra**.

```
library(factoextra)
fviz_eig(res.pca, addlabels = TRUE) # scree plot de los eigenvalue
```



Los **eigenvalues** se utilizan para determinar el número de componentes que deben conservarse. Existen dos maneras frecuentes de analizar estos eigenvalues, por un lado muchos investigadores utilizan los eigenvalues > 1 . Otra manera de determinar el número de componentes es por la cantidad de variación explicada, muchos usan $> 70\%$ de la variabilidad de los datos.

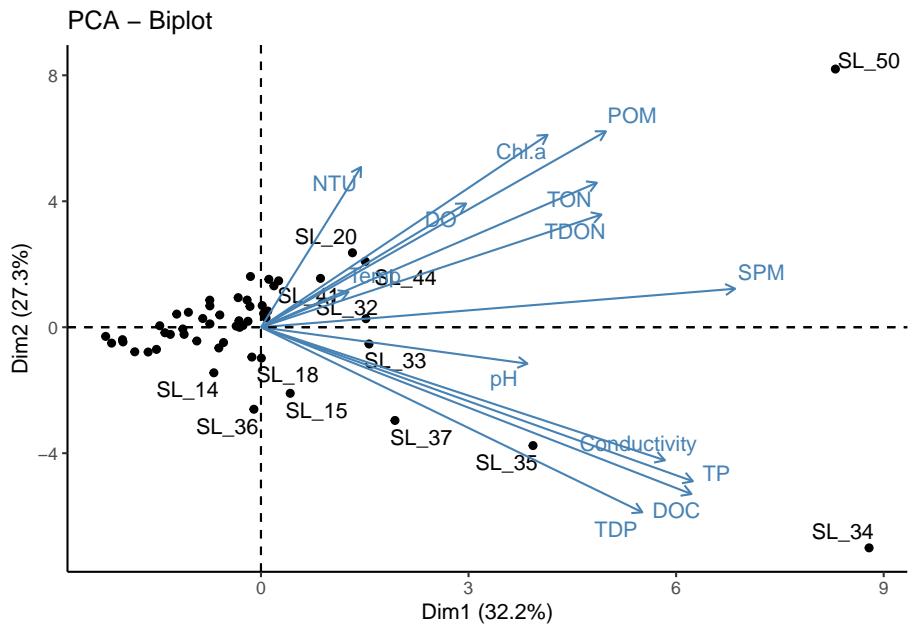
En este ejemplo, los primeros dos planos solo explican un 59.5% de la variabilidad de los datos y al usar un 3 eje se explica 69.8%. Decido usar 3 ejes para describir estos datos.

4.1.3.2 Visualizar los datos

Ahora vamos a visualizar los tres ejes. Como habíamos dicho previamente, el ACP genera gráficos de los individuos y de las variables. Actualmente es más frecuente usar gráficos en los que se observen ambos: los individuos y las variables. Este tipo de gráficos se denomina biplots.

```
?fviz_pca_biplot
fviz_pca_biplot(res.pca, repel=TRUE, invisible = "quali")+theme_classic()

## Warning: ggrepel: 39 unlabeled data points (too many overlaps). Consider
## increasing max.overlaps
```



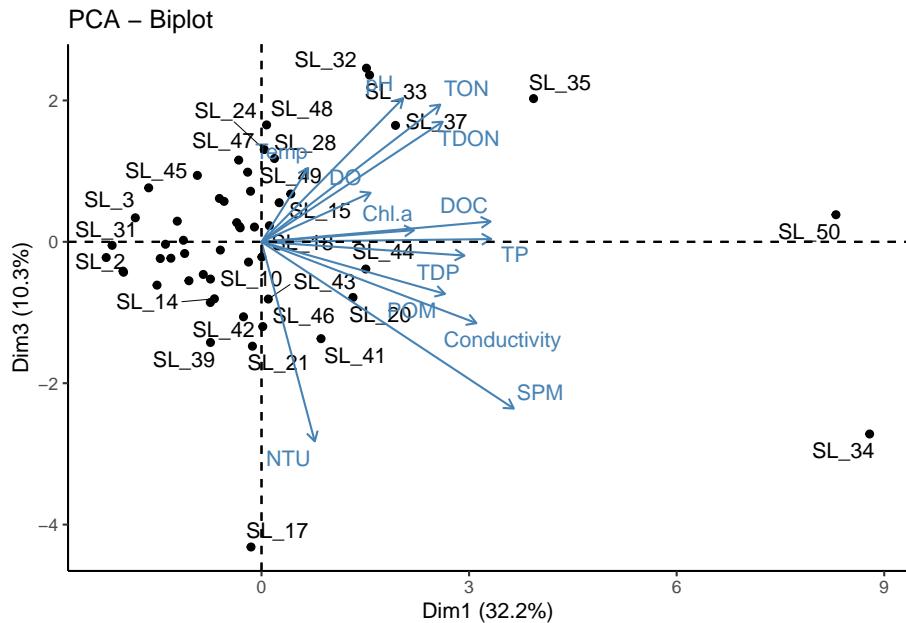
El gráfico muestra los ejes o componentes 1 y 2. A simple vista se observa que algunos individuos, como SL_34 por ejemplo es la muestra que tiene mas DOC (carbono orgánico disuelto), mientras que la muestra SL_50 es la que presentó la mayor cantidad de POM (materia orgánica particulada).

Interpretación: el primer componente (Dim1 32.%) divide a las muestras que tienen más DOC, más conductividad, más TP hacia la derecha del gráfico. El componente 2, parece que divide a las muestras de acuerdo a la POM, la clorofila a, la salinidad.

Ahora vamos a ver el componente 1 vs el componente 3.

```
fviz_pca_biplot(res.pca, axes=c(1,3), repel=TRUE, invisible = "quali")+theme_classic()
```

```
## Warning: ggrepel: 24 unlabeled data points (too many overlaps). Consider
## increasing max.overlaps
```



Una función importante del paquete FactoMiner, es el comando dimdesc. El mismo se utiliza para identificar las variables más significativamente asociadas a un componente.

```
dimdesc(res.pca, axes = c(1:3), proba = 0.05)
```

```
## $Dim.1
##
## Link between the variable and the continuous variables (R-square)
## =====
##          correlation      p.value
## SPM        0.8005755 1.052112e-12
## TP         0.7292919 8.700526e-10
## DOC        0.7269277 1.048135e-09
## Conductivity 0.6820821 2.578475e-08
## TDP        0.6440372 2.589279e-07
## POM        0.5824318 5.906561e-06
## TDON       0.5749478 8.275646e-06
```

```

## TON          0.5672584 1.160335e-05
## Chl.a       0.4838790 2.787599e-04
## pH          0.4499121 8.187113e-04
## DO          0.3463183 1.190357e-02
##
## $Dim.2
##
## Link between the variable and the continuous variables (R-square)
## =====
##           correlation      p.value
## POM        0.7266597 1.070368e-09
## Chl.a     0.7136585 2.879319e-09
## NTU        0.5945254 3.363780e-06
## TON        0.5361705 4.189718e-05
## DO         0.4590453 6.194896e-04
## TDON       0.4178871 2.052244e-03
## Conductivity -0.4936215 2.003631e-04
## TP         -0.5713221 9.715629e-06
## DOC        -0.6187953 1.011371e-06
## TDP        -0.6865882 1.917673e-08
##
## $Dim.3
##
## Link between the variable and the continuous variables (R-square)
## =====
##           correlation      p.value
## pH         0.4460824 9.181604e-04
## TON        0.4261880 1.631046e-03
## TDON       0.3729687 6.466085e-03
## SPM        -0.5178939 8.417748e-05
## NTU        -0.6197587 9.622302e-07

```

Estos resultados nos permiten asociar variables a los distintos ejes/componentes. Se puede inclusive establecer un criterio ya que usa coeficientes de correlación. Para este ejemplo, se puede usar las variables que tienen una correlación >0.6 con el eje. Para el componente principal 1, que explica (32.2%), las variables asociadas a este componente son DOC, TP, TDP, conductivity y SPM. Para el componente principal 2 (27.3%), está asociado a POM, clorofila a y NTU (turbidez). Finalmente, el componente principal 3 (10.3%) estaría asociado de manera negativa solo a la salinidad (NTU). El signo de la correlación indica la dirección de la correlación.

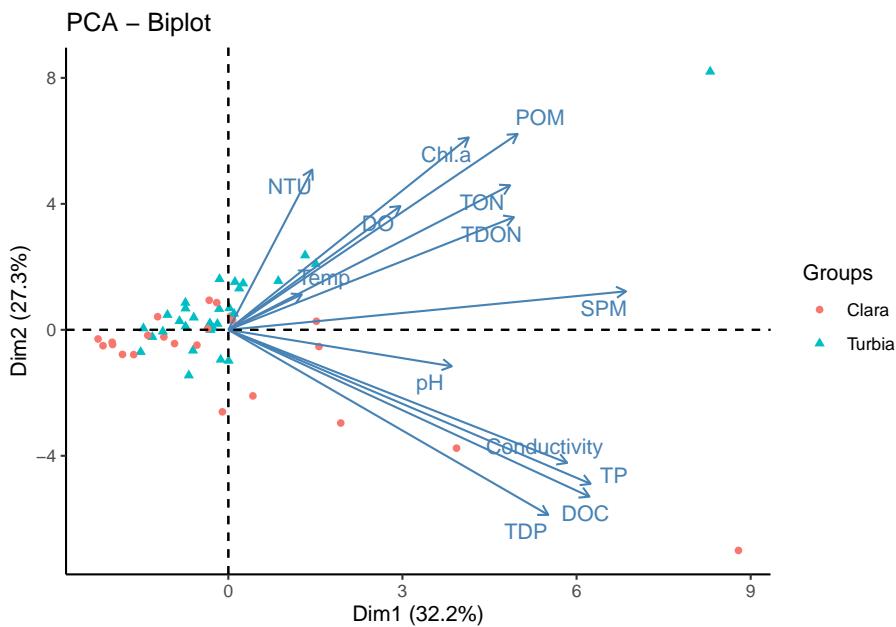
4.1.4 Crear una condición para colorear los individuos

A veces es útil crear alguna condición (variable categórica) para colorear los individuos y quizás empezar a ver patrones más claros. En este voy a usar la variable “secchi” para colorear a los individuos. Si la variable secchi es menor o igual a 20 entonces voy a llamar a la laguna como “turbia” mientras que si es mayor a 20, la voy a denominar laguna “clara”.

```
datos$condicion <- as.factor(ifelse(datos$Secchi<=20, "Turbia", "Clara"))
```

Cree una nueva variable llamada condicion

```
fviz_pca_biplot(res.pca, repel=TRUE, invisible = "quali", habillage = datos$condicion, geom = (
```



4.1.5 Refinando el modelo

El APC explica aproximadamente un 69% usando tres ejes. En el próximo paso vamos a seleccionar solo las variables disueltas: DOC, pH, TDON, DO, TDP, chla. y conductivity.

```
data.stan.sel <- datos.stand [,c(3:5,8,12:14)]
```

Vuelvo a realizar el análisis de ACP:

```
res.pca2 <- PCA(data.stan.sel, graph=FALSE)
```

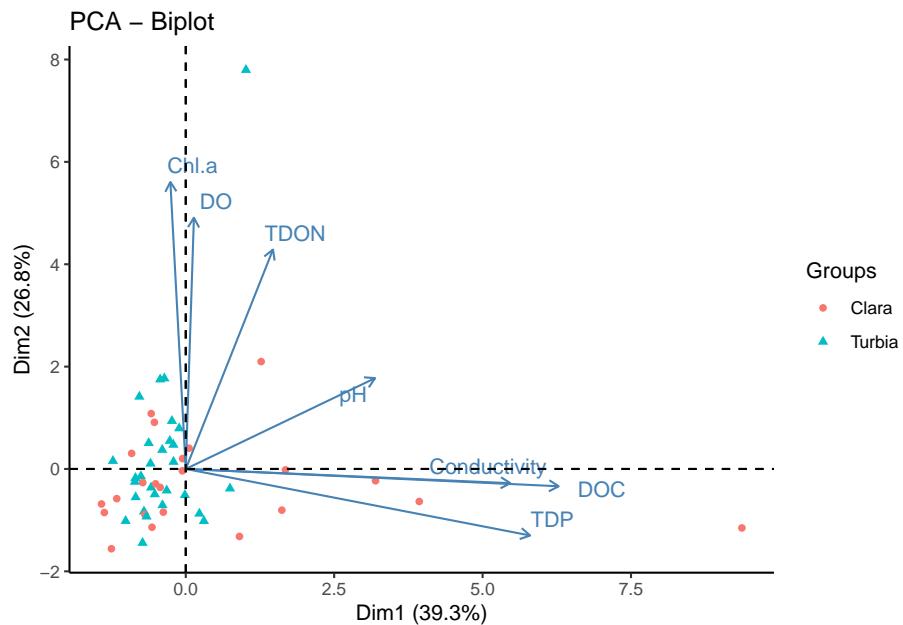
```
## Warning in PCA(data.stan.sel, graph = FALSE): Missing values are imputed by the
## mean of the variable: you should use the imputePCA function of the missMDA
## package
```

```
eigenvalues2 <- res.pca2$eig# eigenvalues
head(eigenvalues2[, 1:3])
```

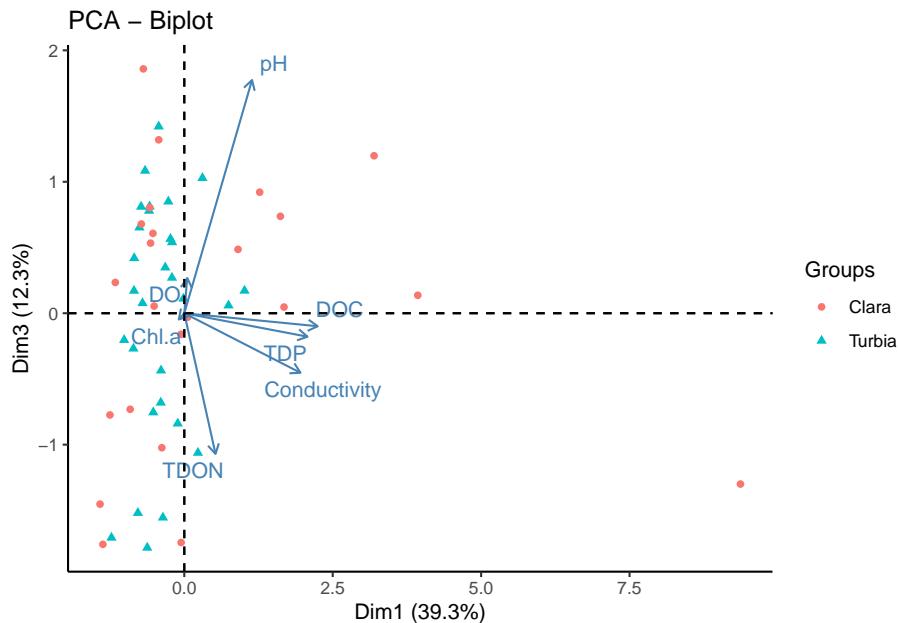
	eigenvalue	percentage of variance	cumulative percentage of variance
## comp 1	2.7477587	39.253696	39.25370
## comp 2	1.8792045	26.845778	66.09947
## comp 3	0.8578634	12.255191	78.35467
## comp 4	0.7497849	10.711212	89.06588
## comp 5	0.4279332	6.113331	95.17921
## comp 6	0.2531477	3.616395	98.79560

Biplot:

```
fviz_pca_biplot(res.pca2, repel=TRUE, invisible = "quali", habillage = datos$condicion)
```



```
fviz_pca_biplot(res.pca2, axes= c(1,3), repel=TRUE, invisible = "quali", habillage = datos$condi
```



El nuevo ACP con las variables seleccionadas, explica un 78% de la variabilidad total de los datos, teniendo en cuenta los primeros 3 ejes. Considero que este modelo refinado es mucho mejor que el anterior.

4.2 NMDS

El escalamiento multidimensional no métrico, mejor conocido por sus siglas en inglés (**Non-metric Multidimensional Scaling**), es un método de análisis estadístico multivariado que representa mediciones de *similaridad* (o disimilaridad) entre pares de objetos como distancias entre puntos de un espacio de dimensión reducida. El objetivo fundamental del **NMDS** es generar una representación gráfica de los objetos en un espacio de modo que sus posiciones relativas sean el reflejo de su proximidad. A diferencia de otros métodos de escalamientos, el NMDS utiliza ordenes de rango, por lo que es una técnica extremadamente flexible que puede adaptarse a una gran variedad de datos.

```
library(vegan)
```

Vamos a usar las variables seleccionadas en el segundo ACP, es decir las variables más relacionadas con la fracción disuelta.

```
disueltos <- datos[,c(6:8,11,15:17)]
summary(disueltos)
```

```
##          DO              pH      Conductivity      TDON
##  Min.   : 5.000   Min.   :8.000   Min.   : 0.320   Min.   :1926
##  1st Qu.: 8.725   1st Qu.:8.557   1st Qu.: 1.215   1st Qu.:3052
##  Median :10.300   Median :8.795   Median : 2.615   Median :3478
##  Mean   :10.238   Mean   :8.762   Mean   : 7.968   Mean   :3790
##  3rd Qu.:11.050   3rd Qu.:8.982   3rd Qu.: 5.513   3rd Qu.:4356
##  Max.   :20.000   Max.   :9.400   Max.   :202.100  Max.   :7235
##
##          TDP             Chl.a        DOC
##  Min.   : 0.1   Min.   : 1.58   Min.   : 1.26
##  1st Qu.:114.0   1st Qu.: 15.80   1st Qu.: 12.36
##  Median :302.0   Median : 52.05   Median : 19.45
##  Mean   :560.3   Mean   : 87.80   Mean   : 93.22
##  3rd Qu.:624.0   3rd Qu.: 89.13   3rd Qu.: 97.58
##  Max.   :4140.0  Max.   :981.06  Max.   :1010.00
##                  NA's   :1
```

Algunos análisis exploratorios multivariados son sensibles a los datos faltantes. Como se puede observar, la variable DOC tiene un dato faltante. En este caso, lo mejor es sacar esa fila del análisis.

```
disueltos <- disueltos[-17,]
condicion <- data.frame(datos$condicion)
condicion <- condicion [-c(17),]## hay que sacar la misma fila de la variable condicion
```

Ahora se estandarizan los datos y luego se los convierte a distancias euclidianas para luego realizar el análisis.

```
set.seed(2306)## generar resultados reproducibles
dis.stan <- decostand(disueltos, "stand") # estandarizar: variables ambientales en diferentes unidades
dis.dist <- vegdist(dis.stan, "euc") # distancia euclidiana
res.nmds <- metaMDS(dis.dist, trymax = 500) # NMDS

## Run 0 stress 0.08172658
## Run 1 stress 0.08190544
## ... Procrustes: rmse 0.01185868 max resid 0.06039221
## Run 2 stress 0.08129024
## ... New best solution
## ... Procrustes: rmse 0.01390953 max resid 0.05964353
## Run 3 stress 0.08801438
```

```
## Run 4 stress 0.08189141
## Run 5 stress 0.0846288
## Run 6 stress 0.08169404
## ... Procrustes: rmse 0.02381188 max resid 0.116934
## Run 7 stress 0.08336852
## Run 8 stress 0.08169061
## ... Procrustes: rmse 0.0190035 max resid 0.08992999
## Run 9 stress 0.09152335
## Run 10 stress 0.08174482
## ... Procrustes: rmse 0.01178546 max resid 0.04261408
## Run 11 stress 0.08704684
## Run 12 stress 0.08073759
## ... New best solution
## ... Procrustes: rmse 0.007340069 max resid 0.03921827
## Run 13 stress 0.08074268
## ... Procrustes: rmse 0.02379549 max resid 0.1167022
## Run 14 stress 0.08214195
## Run 15 stress 0.0819048
## Run 16 stress 0.08173675
## Run 17 stress 0.08722591
## Run 18 stress 0.08190564
## Run 19 stress 0.0877217
## Run 20 stress 0.08117769
## ... Procrustes: rmse 0.008500534 max resid 0.04099969
## Run 21 stress 0.08109837
## ... Procrustes: rmse 0.04191297 max resid 0.2040888
## Run 22 stress 0.08142552
## Run 23 stress 0.08713932
## Run 24 stress 0.08215738
## Run 25 stress 0.08117084
## ... Procrustes: rmse 0.01652584 max resid 0.07695317
## Run 26 stress 0.08088512
## ... Procrustes: rmse 0.009314285 max resid 0.04562758
## Run 27 stress 0.08161162
## Run 28 stress 0.08276532
## Run 29 stress 0.08138922
## Run 30 stress 0.08708527
## Run 31 stress 0.0814658
## Run 32 stress 0.08666174
## Run 33 stress 0.08186204
## Run 34 stress 0.08801345
## Run 35 stress 0.08087082
## ... Procrustes: rmse 0.008549763 max resid 0.04176883
## Run 36 stress 0.08143206
## Run 37 stress 0.08068018
## ... New best solution
```

```

## ... Procrustes: rmse 0.007515726 max resid 0.03743025
## Run 38 stress 0.08102709
## ... Procrustes: rmse 0.03204386 max resid 0.1556518
## Run 39 stress 0.08473582
## Run 40 stress 0.09152295
## Run 41 stress 0.08068146
## ... Procrustes: rmse 0.00815311 max resid 0.04025691
## Run 42 stress 0.09152335
## Run 43 stress 0.08245253
## Run 44 stress 0.08143243
## Run 45 stress 0.08718439
## Run 46 stress 0.08137234
## Run 47 stress 0.08720687
## Run 48 stress 0.08085884
## ... Procrustes: rmse 0.01546736 max resid 0.07652651
## Run 49 stress 0.08111344
## ... Procrustes: rmse 0.03515309 max resid 0.1708126
## Run 50 stress 0.08179221
## Run 51 stress 0.08169834
## Run 52 stress 0.08675948
## Run 53 stress 0.08797076
## Run 54 stress 0.09155446
## Run 55 stress 0.08774277
## Run 56 stress 0.08068016
## ... New best solution
## ... Procrustes: rmse 0.00136907 max resid 0.008143052
## ... Similar to previous best
## *** Best solution repeated 1 times

```

Una regla general: si el stress < 0,05 proporciona una excelente representación en dimensiones reducidas, < 0,1 es genial, < 0,2 es bueno/ok, y un stress < 0,3 proporciona una mala representación. **Para recordar:** ¡un stress alto es malo, un *stress bajo es bueno!*

res.nmds

```

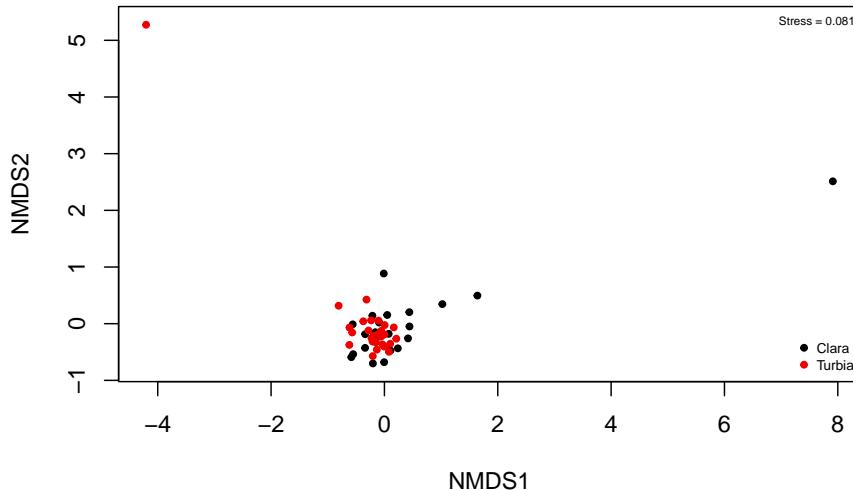
##
## Call:
## metaMDS(comm = dis.dist, trymax = 500)
##
## global Multidimensional Scaling using monoMDS
##
## Data:      dis.dist
## Distance: euclidean
##
```

```
## Dimensions: 2
## Stress: 0.08068016
## Stress type 1, weak ties
## Best solution was repeated 1 time in 56 tries
## The best solution was from try 56 (random start)
## Scaling: centring, PC rotation
## Species: scores missing
```

El stress es de 0.08 que es menor que 0.1 con lo cual la representación está bastante bien. Ahora vamos a graficar estos resultados:

```
{plot(res.nmds, type="n")
points(res.nmds, display = "site", cex=0.6, select=which(condicion=="Clara"), pch = 19, col="black")
points(res.nmds, display = "site", cex=0.6, select=which(condicion=="Turbia"), pch = 19, col="red2")
legend("topright", cex=0.5, box.col=NA,legend=paste("Stress =",round(res.nmds$stress, 3)))
legend("bottomright", cex=0.6, box.col=NA,
       legend=c("Clara", "Turbia"), pch=19, col=c("black","red2"))}

## species scores not available
```



El escalamiento es parecido al resultado del ACP, en donde se pueden observar que hay un solapamiento importante en el tipo de lagunas.

4.3 PERMANOVA

El análisis multivariante de permutaciones de la varianza o PerMANOVA, es una alternativa no paramétrica a la prueba de ANOVA multivariada. Es apropiado con conjuntos de múltiples variables que no cumplen los supuestos, por ejemplo, el de normalidad. Además, se puede utilizar para datos muy sesgados, ordinales o cualitativos, y en datos de comunidades ecológicas, datos de comunidades microbianas o en datos genéticos. Su funcionamiento incluye una matriz de distancias construida a partir de cualquier medida de disimilitud. Este análisis se utiliza para comparar grupos de objetos y probar con la hipótesis nula de que los centroides y la dispersión de los grupos son equivalentes. Este análisis suele acompañar a los gráficos de ordenamiento tales como el NMDS.

```
res.perma <- adonis2(dis.dist~condicion, method = "euclidean", permutations = 599)
res.perma

## Permutation test for adonis under reduced model
## Terms added sequentially (first to last)
## Permutation: free
## Number of permutations: 599
##
## adonis2(formula = dis.dist ~ condicion, permutations = 599, method = "euclidean")
##          Df SumOfSqs      R2      F Pr(>F)
## condicion  1    14.03 0.04009 2.0462   0.06 .
## Residual  49   335.97 0.95991
## Total     50   350.00 1.00000
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Para comparar los distintos grupos se pueden hacer análisis a posteriori, de comparaciones entre los grupos.

```
library(pairwiseAdonis)

## Loading required package: cluster

res.pos<- pairwise.adonis(dis.dist, condicion, sim.method = "euclidean", p.adjust.m =
res.pos

##           pairs Df SumsOfSqs  F.Model      R2 p.value p.adjusted sig
## 1 Clara vs Turbia  1  14.03004 2.046231 0.04008584  0.048      0.048 .
```

Nota: para instalar la librería *pairwiseAdonis* desde el repositorio github del autor, seguir las instrucciones (de acuerdo al Sistema Operativo) del repositorio.

Chapter 5

Análisis multivariados II

TECNICAS DE ORDENACION CANONICA

Maria Eugenia del R. Llames¹

Instituto Tecnológico de Chascomús (INTECH, UNSAM-CONICET), Escuela de Bio y Nanotecnologías (UNSAM)

5.1 Introducción

El supuesto principal de cualquier técnica de ordenación es que los datos analizados son redundantes, es decir, contienen más variables (y dimensiones) de las necesarias para describir la información subyacente, y podemos reducir el número de estas dimensiones sin perder demasiada información. Por ejemplo, en el caso de los datos de composición de especies, algunas de las especies suelen ser ecológicamente similares (por ejemplo, especies que prefieren crecer en un hábitat húmedo en lugar de seco), lo que significa que el conjunto de datos contiene varias variables redundantes (especies) que cuentan la misma historia. O, para explicar la redundancia de otra manera, a partir de la presencia de una especie, a menudo podemos predecir la presencia de varias otras especies. En el caso de la ordenación aplicada en la matriz de variables ambientales, estas a menudo se correlacionan entre sí (por ejemplo, las mediciones del temperatura del agua a menudo se relacionan con las concentraciones de oxígeno disuelto), lo que también permite la reducción de la dimensión.

Dado que el espacio multidimensional no es fácil de mostrar, describir o simplemente imaginar, vale la pena reducirlo a unas pocas dimensiones principales, conservando al máximo la información. Esto también significa que si las variables individuales son completamente independientes entre sí (por ejemplo, cada

¹mariaellames@intech.gov.ar

especie tiene preferencias completamente diferentes), entonces es probable que la ordenación no encuentre una reducción razonable del espacio multidimensional.

Lo que hace el método de ordenación se puede formular de dos maneras alternativas:(i) busca gradientes en la composición de especies (representados generalmente por ejes de ordenación) e intenta explicar estos gradientes por variables ambientales; y/o (ii) busca la distribución de muestras en un espacio de ordenación reducido que refleje al máximo la disimilitud (= distancia) entre muestras en términos de su composición de especies.

De acuerdo al apartado anterior (**Unidad 4**), las técnicas allí presentadas permiten el análisis de una única matriz de datos de forma tal de revelar su estructura a través de un grafo construido con un conjunto reducido de ejes ortogonales (*i.e* independientes) Las variables externas que pueden estar influenciando esta estructura sólo podrán ser consideradas después del cómputo de la ordenación. De acuerdo con esta característica, se las conocen como “técnicas de ordenación indirecta” en donde uno deja que la matriz de datos exprese las relaciones entre objetos y variables sin restricción (“*unconstrained analyses*”). Por lo tanto es una forma pasiva de análisis, y el usuario interpreta los resultados de la ordenación *a posteriori*.

Las técnicas de **ordenación canónica**, por el contrario, permiten asociar dos o más conjuntos de datos en el propio proceso de ordenación. Típicamente, en ecología, estas matrices la constituyen una matriz de datos biológicos, como por ejemplo, relevamiento de especies en diferentes sitios (matriz de “respuesta”) y su matriz ambiental asociada (matriz “explicativa”) (Figura 1).

Figura 1: Esquema de las matrices utilizadas comúnmente en estudios ecológicos

En este contexto, las técnicas de ordenación canónicas se engloban dentro de las “técnicas de ordenación directa” y permiten analizar patrones entre un conjunto de datos que están relacionadas con (o pueden ser interpretadas por) otro conjunto de datos, y/o probar formalmente hipótesis estadísticas sobre la importancia de estas relaciones. En otras palabras, las técnicas de ordenación canónica exploran explícitamente las relaciones entre dos matrices: una matriz de respuesta y una matriz explicativa y ambas matrices son utilizadas en la producción de la ordenación. Son análisis “restringidos” o “*constrained analyses*” (Figura 2).

Figura 2: Esquema de “funcionamiento” de las técnicas de ordenación canónicas.

5.2. LA LÓGICA DE LAS TÉCNICAS DE ORDENACIÓN RESTRINGIDA O CANÓNICA (“CONSTRAINED”, A)

5.2 La lógica de las técnicas de ordenación restringida o canónica (“constrained”, análisis de gradiente directo)

Los ejes de ordenación están limitados/restringidos por factores ambientales. Relaciona la composición de especies directamente con las variables ambientales y extrae la variación en la composición de especies que está directamente relacionada con el medio ambiente. Las variables ambientales ingresan directamente al algoritmo, y los ejes de ordenación restringidos corresponden a las direcciones de la variabilidad en los datos que se explica por estas variables ambientales. El método generalmente se usa como análisis confirmatorio, es decir, puede probar las hipótesis sobre la relación entre los factores ambientales en la composición de especies (a diferencia de la ordenación sin restricciones, que es exploratoria). Descompone la varianza total en los datos de composición de especies en una fracción explicada por variables ambientales (relacionadas con ejes de ordenación restringidos) y no explicada por variables ambientales (relacionadas con ejes de ordenación no restringidos). Ofrece varias oportunidades interesantes cuando se trata de variables explicativas: permite selección directa (la selección de variables ambientales importantes mediante la exclusión de aquellas que no son relevantes para la composición de especies), habilita la prueba de permutación de Monte Carlo (una prueba de significancia de la varianza explicada por factores ambientales) y permite el análisis de la partición de la varianza (partición de la varianza explicada por diferentes grupos de variables ambientales).

5.2.1 ¿Qué tipo de datos de composición de especies se utilizan para el análisis?

5.2.1.1 (a) Métodos basados en datos brutos (enfoque clásico)

Métodos basados en el análisis de matrices crudas de muestras-especies con datos de abundancia o presencia/ausencia. Dentro de estos métodos, se reconocen tradicionalmente dos categorías, que se diferencian por la suposición de la respuesta de las especies a lo largo del gradiente ambiental:

1. **lineal** (Figura 3, panel izquierdo): supone que las especies responden linealmente a lo largo del gradiente ambiental, lo que podría ser cierto para datos ecológicos bastante homogéneos, y esto se da cuando donde los gradientes ecológicos considerados son bastante cortos;
2. **unimodal** (Figura 3, panel derecho): respuesta de la especie unimodalmente a lo largo del gradiente, con su punto óptimo en una determinada posición del gradiente; este modelo se acerca más a la realidad de los datos ecológicos y es más

adecuado para conjuntos de datos heterogéneos (estructurados por un gradiente ecológico fuerte o largo, con un alto recambio de especies y muchos ceros en la matriz de especies), es decir, análisis de gradientes ambientales bastante largos.

Figura 3: Supuesto de respuesta lineal (izquierda) frente a unimodal (derecha) del “fitness”/abundancia de las especies a lo largo del medio ambiente. Fuente <https://www.davidzeleny.net/anadat-r/doku.php/en:ordination>

5.2.1.2 (b) Métodos aplicados sobre datos transformados

Todos los métodos de ordenación tienen implícito un paso de cálculo de índices de distancia en su algoritmo. En algunos casos, tenemos la opción de indicarle al algoritmo qué índice utilizar (ejemplo: PCoA o NMDS), mientras que en otros el índice utilizado es fijo (ejemplo: PCA, RDA). Lo que tenemos que tener en cuenta es que, entre todos los índices posibles, existen índices “simétricos” (incluyen el doble cero en el cálculo del índice), e índices “asimétricos” (no incluyen al doble cero en el cálculo del índice). La utilización de índices simétricos en análisis de ordenación que suponen respuesta lineal pueden llevar a la paradoja de que dos sitios se “ordenen cercanos” en base a la ausencia de especies en ambos sitios (para más detalle sobre este punto, referirse al Capítulo 7 de libro “Numerical Ecology” de Legendre & Legendre, 2012). En aquellos casos en que el modelo a aplicar es de respuesta lineal y se base en el cálculo de índices simétricos, se recomienda transformar los datos de composición de especies de forma tal de adecuar los datos ecológicos y evitar caer en la paradoja del doble-cero. En este sentido, Legendre y Gallagher (2001) detallan diversas transformaciones adecuadas para la aplicación de métodos de ordenación que asumen respuesta lineal.

5.2.1.3 (c) Métodos basados en la distancia

Métodos que utilizan la matriz de distancias entre muestras medidas por coeficientes de distancia y que proyectan estas distancias en diagramas de ordenación de dos o más dimensiones. El método se conoce como db-RDA (RDA basado en la distancia) y consiste en una combinación de PCoA, aplicada en datos sin procesar utilizando una medida de distancia seleccionada, y RDA aplicada en los autovectores resultantes de PCoA (ver más abajo). Ofrece una alternativa a RDA (basada en distancias euclidianas) y al RDA basado en datos transformados (por ejemplo, transformados por Hellinger), con la libertad de elegir la medida de distancia adecuada para los datos investigados.

5.2.2 ¿Cómo saber si debo utilizar un método de ordenación de respuesta lineal o unimodal?

Para decidir si aplicar el método de ordenación de respuesta lineal o unimodal en los datos, podemos basarnos en la regla general introducida por Lepš & Šmilauer (2003): primero, calcule DCA (“Detrended correspondence analysis”-análisis sin tendencia por segmentos) en sus datos y verifique la longitud del primer Eje del DCA (que está escalado en unidades de desviación estándar, S.D.). Una longitud del primer eje DCA > 4 S.D. indica un conjunto de datos heterogéneo en el que se deben utilizar métodos unimodales, mientras que la longitud < 3 S.D. indica un conjunto de datos homogéneo para el cual los métodos lineales son adecuados. En la zona gris entre 3 y 4 S.D., tanto los métodos lineales como los unimodales son adecuados. Sin embargo, tengan en cuenta que si bien **los métodos lineales no deben usarse para datos heterogéneos** (*i.e.* para datos de respuesta unimodal), los métodos unimodales pueden ser usados para datos homogéneos, no obstante, los métodos lineales, en este caso, son más poderosos y deberían ser preferidos. Alternativamente, si tus datos de composición de especies son heterogéneos, pero aún querés utilizar algún método de ordenación lineal (PCA, RDA), recordá trasnformar estos datos de composición de especies de forma tal de evitar el problema del doble-cero.

5.3 Análisis de Redundancia (RDA)

El RDA es un método que combina las técnicas de regresión lineal con el análisis de componentes principales (PCA). Por un lado, es una extensión directa del análisis de regresión múltiple para modelar la respuesta multivariada de datos. *Redundancia* es sinónimo de “variación explicada” y se interpreta de la misma manera que como lo aprendieron en sus cursos introductorios de Estadística al estudiar los modelos de análisis univariados. El análisis se dice *asimétrico* (*i.e.* se establece una matriz de respuesta y una matriz explicativa), en donde la matriz “Y” (biológica) corresponde a la variable multidimensional de respuesta y “X” (o “E”, de acuerdo a la Figura 1), corresponde a la matriz multidimensional de variables explicativas. Desde una perspectiva descriptiva, uno diría que la ordenación de “Y” está *restringida* de forma tal que los vectores de ordenación resultantes son combinaciones lineales de las variables en “X”.

Por otro lado, RDA también puede verse como una extensión del análisis de componentes principales (PCA), porque los vectores de ordenación canónica que resultan constituyen combinaciones lineales de las variables de respuesta “Y”. Esto significa que cada vector de ordenación es una proyección unidimensional de la distribución de los objetos en un espacio que conserva las distancias euclidianas entre ellos.

Rápidamente, este método busca, en orden sucesivo, una serie de *combinaciones lineales* de las *variables explicativas* que *mejor explican la variación* de los datos de la matriz de *variable respuesta*. Los *ejes* definidos en el espacio a partir de la matriz de las variables explicativas son *ortogonales* entre sí (*i.e.* independientes). RDA es por lo tanto un *procedimiento de ordenación restringida*. La diferencia con la ordenación sin restricciones es importante: la matriz de variables explicativas condiciona los “pesos” (*i.e.* autovalores, valores propios o *eigenvalues*) y las direcciones de los ejes de ordenación. En RDA, uno puede decir verdaderamente que los ejes explican o modelan (en el sentido estadístico) la variación de la matriz respuesta o dependiente.

Como resultado, la variación en la composición de especies se descompone en variación relacionada con variables ambientales (representadas por ejes restringidos/canónicos, autovectores o *eigenvectors*) y no relacionada con variables ambientales (ejes no restringidos). El número total de ejes canónicos generados corresponde a la cantidad $\min [p, m, n-1]$. Es decir, la cantidad de ejes será el número más bajo entre los parámetros p , m , y $(n-1)$, talque (*i*) la cantidad de ejes generados no puede exceder p que es el tamaño del espacio de referencia de la matriz “Y” (*i.e.* número de variables respuesta consideradas, de especies para nuestro caso); (*ii*) no puede exceder m que es el número de variables en X (cantidad de variables ambientales consideradas) y (*iii*) no puede exceder $(n - 1)$, que es el número máximo de dimensiones requeridas para representar n puntos (*i.e.* sitios) en el espacio euclíadiano. Finalmente, la significancia estadística del modelo de RDA (modelo global) y la de los ejes canónicos se pueden probar mediante análisis permutacionales (no paramétricos).

5.4 Caso de estudio

5.4.1 Analicemos estos aspectos a través de un ejemplo práctico...

Utilizaremos un set de datos del trabajo *Bacterioplankton morphotypes structure and cytometric fingerprint rely on environmental conditions in a sub-Antarctic peatland* publicado en *Hydrobiologia* (Quiroga *et al.* 2017), disponibles en el Repositorio Institucional CONICET Digital. Para ello, descargar el set de datos **morpho_biomass.csv** y **chem.csv** de GitHub Limno-con-R/CILCAL2023. Guardar los archivos en una carpeta llamada *data*, dentro del **Directorio de Trabajo del Proyecto** que creamos para esta Unidad (ver cómo hacerlo en la Unidad ??). Finalmente, instalar los paquetes como se indica en la Unidad ?. Luego, cargarlos en la sesión.

5.4.2 RDA utilizando el paquete vegan (Oksanen *et al.*, 2007)

```

library (vegan)

## Warning: package 'vegan' was built under R version 4.2.3

## Loading required package: permute

## Loading required package: lattice

## This is vegan 2.6-4

data_biomass <- read.csv("./data/morpho_biomass.csv", row.names=1) # biomasa [pgC/mL]
biomass <- data_biomass[,4:9] #selecciono solo los datos de biomasa de los diferentes morfotipos
bio.trans <- log10(biomass+1) # transformacion log10(x+1). Esta transformación se utiliza para est

# Cómputo del DCA para verificar la longitud del gradiente
#----

vare.dca <- decorana(bio.trans)
vare.dca

## 
## Call:
## decorana(veg = bio.trans)
##
## Detrended correspondence analysis with 26 segments.
## Rescaling of axes with 4 iterations.
## Total inertia (scaled Chi-square): 0.109
##
##          DCA1      DCA2      DCA3      DCA4
## Eigenvalues   0.07139  0.005130  0.0020470  0.0012492
## Additive Eigenvalues 0.07139  0.008798  0.0117827  0.0154503
## Decorana values  0.10595  0.001355  0.0005076  0.0002615
## Axis lengths   0.60193  0.239784  0.1647616  0.1281498

```

La última línea de la primer columna (DCA1) indica *Axis lengths*= 0.60193. Ese valor corresponde a la longitud del primer eje de ordenamiento del DCA, expresado en unidades de desvio estándar (S.D.) de recambio de especies. Un gradiente mayor a 4 S.D. indica que algunas especies presentan respuesta unimodal. En el caso de nuestro ejemplo, la longitud del gradiente es < 4, indicando

un gradiente homogéneo o corto y, por lo tanto, el modelo de respuesta lineal es el adecuado para aplicar. Procedemos, entonces, al modelado a través de un RDA:

```
# RDA
#----
# Seleccion de variables explicativas
ambiental <- read.csv("./data/chem.csv", row.names=1) #cargo las variables ambientales
env <- ambiental[,c(4:5,7:15)] # saco una de ellas: "DO" (oxígeno disuelto) ya que tie
```

Hasta acá, hemos (i) verificado el tipo de respuesta en nuestros datos de composición de especie para elegir el modelo adecuado y (ii) preparamos los datos de composición y de datos ambientales para modelar. Además del tipo de respuesta, otro requisito necesario es la “multinormalidad” de los datos de la matriz ambiental. En el siguiente paso, lo que vamos a hacer es chequear que este supuesto se cumpla:

```
#Chequeo de normalidad de los datos en la amtriz ambiental
library(MVN)
mvn.env <- mvn(env, mvnTest = "mardia")
mvn.env$Descriptives$Skew # skwe < 2

## [1] 0.7514272 -0.8963630 0.3566289 0.4353447 0.6865369 0.1131861
## [7] 1.5255447 0.7839304 0.4965284 0.3858427 -0.2629160

mvn.env$multivariateNormality$p.value #p-valores > 0.05, no tenemos evidencia para re
```

```
## [1] 0.854080999478208 0.186705671286224 <NA>
## Levels: 0.854080999478208 0.186705671286224
```

En este caso, elegimos analizar la asimetría de los datos (“*skewness*”) a través del estudio de la significancia de los coeficiente de asimetría y curtosis multivariado de Mardia. Para la normalidad multivariada, los valores *p* de las estadísticas de asimetría y curtosis deben ser superiores a 0,05. Si el tamaño de la muestra es inferior a 20, se debe utilizar *p.value.small* como valor significativo de asimetría en lugar de *p.value.skew*.

Con los supuestos del modelo revisados y comprobados, avanzamos con la aplicación del modelo a los datos. En el caso del paquete **vegan**, permite el cálculo de una RDA de dos maneras diferentes. La sintaxis más simple es listar los nombres de los marcos de datos involucrados separados por comas:

```
simpleRDA <- rda(bio.trans, env)
```

Esta manera, si bien es sencilla, tiene algunas limitaciones. Su principal inconveniente es que no permite incluir variables cualitativas en la matriz explicativa. Por lo tanto, en todas las aplicaciones excepto en las más simples, es mejor usar la interfaz de la fórmula:

```
biomass.rda<-rda(bio.trans ~ ., data= env, scale=TRUE )#dado que las variables #ambientales son categóricas  
#Observación: Observá el atajo(.) para indicarle a la función que use todas las #variables presentes  
summary (biomass.rda)
```

```

## Call:
## rda(formula = bio.trans ~ pH + EC + TH + DOC + DIN + TN + DRP +      TP + a440 + SUVA254 + E2_
##
## Partitioning of correlations:
##           Inertia Proportion
## Total       6.000   1.0000
## Constrained 4.657   0.7762
## Unconstrained 1.343   0.2238
##
## Eigenvalues, and their contribution to the correlations
##
## Importance of components:
##           RDA1    RDA2    RDA3    RDA4    RDA5    RDA6    PC1
## Eigenvalue     3.1710  0.8844  0.33988  0.21332  0.033907  0.014628  0.6473
## Proportion Explained  0.5285  0.1474  0.05665  0.03555  0.005651  0.002438  0.1079
## Cumulative Proportion 0.5285  0.6759  0.73256  0.76811  0.773762  0.776200  0.8841
##           PC2    PC3    PC4    PC5    PC6
## Eigenvalue     0.36127 0.23664 0.058178 0.035203 0.0042519
## Proportion Explained  0.06021 0.03944 0.009696 0.005867 0.0007087
## Cumulative Proportion 0.94429 0.98373 0.993424 0.999291 1.0000000
##
## Accumulated constrained eigenvalues
## Importance of components:
##           RDA1    RDA2    RDA3    RDA4    RDA5    RDA6
## Eigenvalue     3.1710  0.8844  0.33988  0.2133  0.033907  0.014628
## Proportion Explained  0.6809  0.1899  0.07298  0.0458  0.007281  0.003141
## Cumulative Proportion 0.6809  0.8708  0.94377  0.9896  0.996859  1.0000000
##
## Scaling 2 for species and site scores
## * Species are scaled proportional to eigenvalues
## * Sites are unscaled: weighted dispersion equal on all dimensions
## * General scaling constant of scores: 3.26758

```

```

## 
## 
## Species scores
## 
##      RDA1     RDA2     RDA3     RDA4     RDA5     RDA6
## Fil    0.1640   0.9800   0.1229 -0.36682  0.01563  0.006191
## Lrods  1.0807   0.1722   0.1353   0.21391  0.04241  0.124300
## Vibrio 1.0330   0.1990  -0.3880   0.10220  0.14806 -0.061658
## Lcocci  1.0518  -0.2601   0.5923 -0.01995  0.02843 -0.065814
## Srods   1.1491   0.2899  -0.1576   0.12975 -0.18794 -0.032996
## Scocci  0.9769  -0.6264  -0.2126 -0.41426 -0.01565  0.036324
## 
## 
## Site scores (weighted sums of species scores)
## 
##      RDA1     RDA2     RDA3     RDA4     RDA5     RDA6
## 10 -0.315023 -0.01150 -0.135348  1.83690  1.989508  2.36638
## 20 -1.007735  1.21914  0.282566  0.40541  2.088730  1.04932
## 30 -1.484836  1.11482  1.265037 -1.32407 -1.594090 -0.24504
## 40 -0.600298  0.02235 -0.719724  1.50242 -0.090550  1.60198
## 50 -1.158091  -0.80220  1.212587 -0.40025 -1.467472 -0.72606
## 1D  0.125647  -0.12541 -0.241248  1.54171  0.244519 -2.25780
## 2D  0.503364  -0.50115  0.265594  0.84928 -1.481579  0.48143
## 3D  0.571223  -0.93323  0.477569 -0.33140 -3.801358 -1.22805
## 4D  1.440168  1.09255 -0.009618 -1.21473 -1.730754  2.44805
## 5D  1.110690  -0.83129  1.491267  0.91641  5.229787 -0.65145
## 1F  -0.004785  0.97072  0.213935 -0.49315  0.297806 -4.20914
## 2F  0.243819  -0.95176 -0.343644 -0.70321 -1.449543 -1.28073
## 3F  -0.145507  0.29996  0.622884 -2.30409 -2.203613  4.03065
## 4F  0.955108  1.20742  0.102013  0.23464  0.302300  0.83304
## 5F  0.660170  -0.47342 -0.933551  0.65107  0.249799 -0.36565
## 1A  0.284992  1.49562 -0.243656  0.11548  0.300082  0.24773
## 2A  -0.126578  -0.94884  0.028396 -0.58969  1.471159  0.03331
## 3A  0.450376  -0.60836 -1.317999  0.08787  0.958064 -0.96982
## 4A  -1.273194  -0.62696 -1.942517 -0.96279  0.691910 -1.54544
## 5A  -0.229511  -0.60848 -0.074542  0.18221 -0.004706  0.38729
## 
## 
## Site constraints (linear combinations of constraining variables)
## 
##      RDA1     RDA2     RDA3     RDA4     RDA5     RDA6
## 10 -0.666050 -0.03263 -0.20456  0.67067  0.30973  1.13043
## 20 -0.589321  0.88879 -0.19113  1.06969  0.83522  0.07651
## 30 -1.147534  0.65602  0.97820 -0.32919 -0.63364  0.64437
## 40 -0.541377  0.06784  0.30905  0.83005 -0.34576  0.49226
## 50 -1.367806 -0.60884  0.89670  0.09781  0.20458  0.29427

```

```

## 1D  0.009829  0.05750 -0.08803  1.53774  0.90167 -1.38061
## 2D -0.051073  0.14523  0.39871  0.09523 -1.36220  0.16740
## 3D  0.791745 -0.67157  0.49123  0.56383 -1.01736 -1.11073
## 4D  1.402806  0.76775 -0.20109 -0.26664  1.09996  0.76278
## 5D  0.750646 -0.79242  1.44929 -0.54727  0.84634 -0.33252
## 1F  0.188119  0.86592  0.35285 -0.64724  0.05688 -0.77830
## 2F  0.543423 -0.25688 -0.09422 -0.22587  0.52622  1.29148
## 3F -0.217915 -0.65881  0.07894 -1.19060  0.10541 -0.28021
## 4F  1.014708  0.75753 -0.24087  0.08763 -1.74999  0.36433
## 5F  0.318650 -0.94261 -1.09939  0.29612  0.09228  0.31852
## 1A -0.073515  1.77741 -0.15026 -0.49096  0.29268 -0.96792
## 2A  0.288515 -0.54700  0.59304 -0.89605  0.35509  0.05141
## 3A  0.419385 -0.12031 -1.15996 -0.05880 -0.12945  0.59023
## 4A -1.205438 -0.29740 -1.70082 -1.29957 -0.08222 -0.77949
## 5A  0.132203 -1.05551 -0.41768  0.70342 -0.30544 -0.55421
##
##
## Biplot scores for constraining variables
##
##          RDA1      RDA2      RDA3      RDA4      RDA5      RDA6
## pH       0.24581  0.67115 -0.41787 -0.003153  0.13921 -0.281524
## EC       0.62762 -0.23182 -0.44280 -0.411541 -0.22997 -0.241235
## TH      -0.01424  0.32740 -0.10029  0.266745 -0.02467 -0.130398
## DOC      0.39463 -0.34017  0.01318 -0.137529 -0.02881 -0.475663
## DIN      -0.25130 -0.14701 -0.40052  0.175035  0.16676  0.188011
## TN       0.53545 -0.12080  0.01779 -0.644442  0.25086 -0.011189
## DRP     -0.17603  0.32493  0.29212 -0.201619  0.04890  0.152063
## TP       0.16442 -0.03934 -0.66674 -0.027193 -0.13282 -0.005328
## a440     0.03653 -0.48845 -0.66818 -0.233564 -0.24154 -0.272376
## SUVA254  0.13008 -0.29171 -0.27511  0.019930 -0.30331  0.359999
## E2_E3    -0.17459 -0.17869 -0.06789  0.162049  0.09327 -0.451209

```

Coeficientes canónicos
coef(biomass.rda)

```

##          RDA1      RDA2      RDA3      RDA4      RDA5
## pH       2.152488e-02 1.592068e-01 -1.762663e-02 6.615200e-02 -6.710379e-02
## EC       2.531088e-02 7.981753e-04 -1.679251e-02 -1.075705e-02 -1.256239e-02
## TH      -6.176966e-03 1.827047e-02 -6.753879e-03 -1.236731e-02 -9.362529e-03
## DOC      3.715987e-02 1.313701e-02 5.364947e-02 4.154162e-02 -7.030470e-02
## DIN     -4.111146e-04 -1.983296e-04 -4.529877e-03 1.006686e-04 4.049021e-03
## TN       7.788896e-06 -1.510186e-05 -8.793536e-06 -4.098084e-05 9.646049e-05
## DRP     -5.590659e-03 8.149917e-03 1.649684e-03 -8.282799e-03 -9.950571e-03
## TP       2.128018e-03 1.801153e-04 -1.587443e-03 2.700815e-03 2.960609e-03
## a440    -1.459099e-01 2.398457e-03 -1.904511e-02 -1.356861e-01 -1.003461e-01

```

```

## SUVA254  4.905131e-02 -9.475625e-03  2.433850e-02  7.493413e-02 -8.659162e-02
## E2_E3    4.729434e-02 -2.018131e-01 -1.164860e-01  2.563431e-01  5.309020e-01
##                      RDA6
## pH      -2.039089e-01
## EC      2.933565e-02
## TH      7.262698e-03
## DOC     -5.807067e-02
## DIN     6.838714e-03
## TN      1.385058e-05
## DRP     1.061373e-03
## TP      6.682286e-04
## a440    -8.433608e-02
## SUVA254  3.630886e-02
## E2_E3    -8.524522e-02

## Test Global para el RDA con todas las variables

anova(biomass.rda, permutations = how(nperm = 999))#nada tiene que ver este análisis #

## Permutation test for rda under reduced model
## Permutation: free
## Number of permutations: 999
##
## Model: rda(formula = bio.trans ~ pH + EC + TH + DOC + DIN + TN + DRP + TP + a440 + S
##              Df Variance   F Pr(>F)
## Model     11  4.6572 2.5224  0.013 *
## Residual  8   1.3428
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

##R^2
(R2 <- RsquareAdj(biomass.rda)$r.squared) # 0.78

## [1] 0.7762

##el R^2, al igual que ocurre en la regresión múltiple, está "inflado" por la cantidad de variables incluidas en el modelo

## R^2 ajustado (penalizado por la cantidad de variables incluidas en el modelo, mide la calidad del modelo)
(R2adj <- RsquareAdj(biomass.rda)$adj.r.squared)

## [1] 0.4684749

```

#0.47

Expliquemos un poco esta salida... En primer lugar, se revisa la prueba global del modelo utilizando todas las variables explicativas (resultado de la función *anova*). Si, y solo si, la prueba global es significativa, se puede proceder con la selección de variables (más abajo). La salida anterior porporciona información útil: “*Inertia*” (Inercia) es otro nombre para variación o varianza en este caso. “*Total*” se refiere a la varianza total, “*Constrained*” (Restringida) se refiere a la cantidad de varianza explicada por las variables explicativas, “*Unconstrained*” (Sin restricciones) se refiere a la varianza residual. Restringido + Sin restricciones = Total. Se deriva, además, un estadístico R^2 que se interpreta de manera similar al R^2 de regresión: representa la variación explicada por el modelo en relación a la varianza total (Restringido/Total). Los autovalores se muestran tanto para los ejes restringidos como para los no restringidos. En este contexto, estos autovalores indican qué con cantidad de varianza contribuye cada uno de los ejes.

Podemos graficar el resultado de nuestro modelo (con todas las variables explicativas) para tener una idea de qué variables se correlacionan con especies a lo largo de qué ejes.

```
ordiplot(biomass.rda, scaling = 1, type = "text")
```

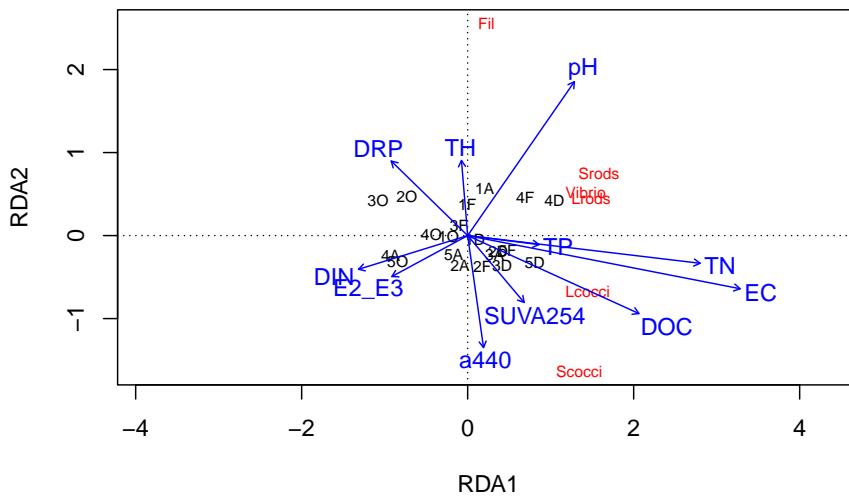


Figura 4: El “escalado” 1 refleja en el gráfico las similitudes entre objetos en la matriz de respuesta. Es decir, los sitios (en negro) que están más cerca entre

sí tienen comunidades más similares. Las especies que están más juntas ocupan más sitios en común.

```
ordiplot(biomass.rda, scaling = 2, type = "text")
```

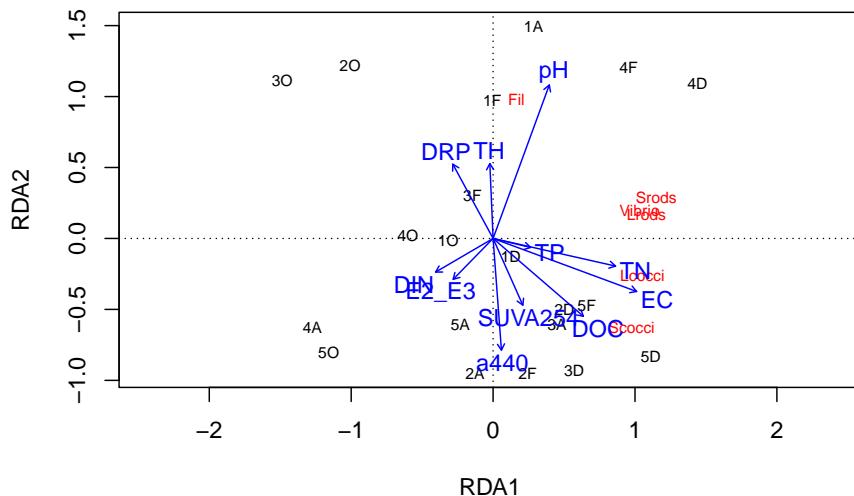


Figura 5: El “escalado” 2 muestra los efectos de las variables explicativas. Las flechas más largas significan que esta variable impulsa fuertemente la variación en la matriz de la comunidad. Las flechas que apuntan en direcciones opuestas tienen una relación negativa. Las flechas que apuntan en la misma dirección tienen una relación positiva. Cuanto más superpuestas las flechas, más correlacionadas resultan esas variables (ya sea positiva o negativamente).

5.4.3 Seleccionando las variables explicativas relevantes

Si queremos simplificar este modelo, podemos realizar una selección hacia adelante (*forward*, van ingresando de a una por vez al modelo), o hacia atrás (*backward*, inicia con todas las variables involucradas y va eliminando de a una), o paso a paso (*stepwise*, proceso iterativo de inclusión y exclusión de variables). Estos tipos de selecciones nos ayudan a seleccionar variables que son estadísticamente importantes. Sin embargo, es importante notar que seleccionar variables de relevancia ecológica es mucho más importante que realizar la selección de variables “estadísticamente significativas”. Si una variable relevante desde el punto de vista ecológico no se selecciona “estadísticamente”, esto no significa que deba eliminarse del RDA.

```

# Forward selection of variables:
fwd.sel <- ordistep(rda(bio.trans ~ 1, data = env, scale= TRUE), # modelo mínimo con una sola variable
                      scope = formula(biomass.rda), #modelo "de máxima", con todas las variables incluidas
                      direction = "forward",
                      R2scope = TRUE, # no puede sobrepasar el R^2 del modelo completo
                      pstep = 1000,
                      trace = TRUE) # cambiar a FALSE para no ver todo el proceso de selección

## 
## Start: bio.trans ~ 1
##
##          Df      AIC      F Pr(>F)
## + EC     1 33.486 5.4886  0.020 *
## + TN     1 35.111 3.6559  0.035 *
## + DOC    1 36.689 2.0134  0.105
## + pH     1 36.512 2.1912  0.125
## + a440   1 37.495 1.2229  0.275
## + DIN    1 37.847 0.8874  0.440
## + TP     1 37.997 0.7466  0.470
## + DRP    1 38.029 0.7166  0.480
## + SUVA254 1 38.270 0.4923  0.675
## + E2_E3   1 38.353 0.4154  0.780
## + TH     1 38.425 0.3496  0.825
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Step: bio.trans ~ EC
##
##          Df      AIC      F Pr(>F)
## + a440   1 30.718 4.5770  0.005 **
## + pH     1 32.870 2.3759  0.080 .
## + E2_E3   1 34.177 1.1501  0.285
## + TN     1 34.291 1.0469  0.410
## + DIN    1 34.608 0.7632  0.535
## + TP     1 34.803 0.5910  0.595
## + DRP    1 34.851 0.5485  0.640
## + TH     1 34.873 0.5294  0.695
## + DOC    1 34.986 0.4307  0.725
## + SUVA254 1 35.170 0.2709  0.895
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Step: bio.trans ~ EC + a440
##
##          Df      AIC      F Pr(>F)

```

```

## + pH      1 30.098 2.2393  0.060 .
## + TP     1 30.590 1.7964  0.185
## + TN     1 31.422 1.0710  0.330
## + DOC    1 31.460 1.0392  0.370
## + DRP    1 31.766 0.7802  0.460
## + SUVA254 1 31.840 0.7185  0.585
## + TH     1 32.114 0.4904  0.705
## + E2_E3   1 32.045 0.5477  0.715
## + DIN    1 32.500 0.1756  0.945
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

```

Para evitar la sobreestimación de la varianza explicada, la selección de variables debe realizarse con dos criterios de parada: (1) el nivel de significación alfa habitual y (2) el coeficiente de determinación múltiple ajustado ($R^2_{adj.}$) calculado utilizando todas las variables explicativas. Cuando la selección hacia adelante identifica una variable que hace que uno u otro criterio supere el umbral fijado, se rechaza esa variable y se detiene el procedimiento (referirse a Blanchet, F. G., Legendre, P., & Borcard, D. 2008. *Forward selection of explanatory variables*. Ecology, 89(9): 2623-2632, para más detalles).

Qué variables son retenidas por la selección *forward*?

```
fwd.sel$call
```

```
## rda(formula = bio.trans ~ EC + a440, data = env, scale = TRUE)
```

Cuál es el valor del R^2 ajustado de este modelo?

```
# Escribir el modelo
biomass.rda.signif <- rda(bio.trans ~ EC + a440, data = env, scale= TRUE)

RsquareAdj(biomass.rda.signif)
```

```
## $r.squared
## [1] 0.3962273
##
## $adj.r.squared
## [1] 0.3251952
```

Tests de significación

```

anova.cca(biomass.rda.signif, step = 1000) #Significación global del modelo final

## Permutation test for rda under reduced model
## Permutation: free
## Number of permutations: 999
##
## Model: rda(formula = bio.trans ~ EC + a440, data = env, scale = TRUE)
##          Df Variance      F Pr(>F)
## Model     2   2.3774 5.5781  0.001 ***
## Residual 17   3.6226
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

anova.cca(biomass.rda.signif, step = 1000, by = "term") #significación de variables

## Permutation test for rda under reduced model
## Terms added sequentially (first to last)
## Permutation: free
## Number of permutations: 999
##
## Model: rda(formula = bio.trans ~ EC + a440, data = env, scale = TRUE)
##          Df Variance      F Pr(>F)
## EC        1   1.4020 6.5793  0.001 ***
## a440      1   0.9753 4.5770  0.008 **
## Residual 17   3.6226
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

anova.cca(biomass.rda.signif, step = 1000, by = "axis") #significación de los ejes

## Permutation test for rda under reduced model
## Forward tests for axes
## Permutation: free
## Number of permutations: 999
##
## Model: rda(formula = bio.trans ~ EC + a440, data = env, scale = TRUE)
##          Df Variance      F Pr(>F)
## RDA1      1   1.9986 9.3787  0.002 **
## RDA2      1   0.3788 1.7776  0.143
## Residual 17   3.6226
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

```

Colinealidad entre variables explicativas

```
vif.cca(biomass.rda.signif) #Sii vif ("variable inflation factor") <5 no hay colinealidad
##          EC      a440
## 1.693375 1.693375
```

5.4.4 RDA basado en matriz de distancia (“Distance-based RDA”, db-RDA)

La diferencia en el análisis entre RDA y dbRDA es simplemente el paso de expresar una matriz de disimilitud no euclidiana en un espacio euclidiano. ¿Les suena familiar?... Sí! eso es exactamente lo que hace un PCoA. Los datos de especies crudos, sin procesar, se transforman primero en una matriz de disimilitud utilizando alguna métrica de disimilitud seleccionada, y con esta matriz se modela un PCoA. La matriz resultante de *scores* de sitios sobre todos los ejes de ordenación de PCoA se usa luego en RDA junto con las variables explicativas. El beneficio de dbRDA es que se puede aplicar cualquier métrica de distancia a los datos (es decir, no solo euclidiana como en RDA, Hellinger (o algunas otras) como en tbRDA o chi-cuadrado como en CCA). Se debe tener cuidado y evitar los autovalores negativos obtenidos en el PCoA, que se omitirían de los análisis. Para que ellos no ocurra, la solución es usar solo distancias métricas (euclidianas), o aplicar una transformación a la matriz de distancia de forma tal de convertir esa distancia “no métrica” en una métrica (por ejemplo, la transformación de raíz cuadrada aplicada a la matriz de distancia obtenida por Bray-Curtis), o usar algunas de las correcciones propuestas en las diferentes funciones.

Notas: 1. Si se aplica dbRDA a una matriz de distancia euclidiana, el dbRDA resulta idéntico a RDA.

2. Si no se proporcionan variables explicativas, un dbRDA resulta idéntico a un PCoA (porque el primer paso da como resultado un PCoA regular y el segundo paso no tiene sentido).

Estos pasos son bastante simples de realizar y se pueden ejecutar uno por uno en R con unas pocas líneas de código. Sin embargo, **vegan** propone la función **capscale()** para este análisis. Esta función permite el trazado directo de las puntuaciones medias ponderadas de las especies si el usuario proporciona la matriz de especies en el argumento *com*.

```
#El primer paso es decidir qué medida de distancia usar. Una forma en que podemos hacerlo es
rankindex(env, biomass, indices = c("euc", "man", "gow", "bra", "kul"), stepacross= FALSE)
```

```
##      euc      man      gow      bra      kul
## 0.2327555 0.2591706 0.3040157 0.3376863 0.3214870
```

La distancia de Bray Curtis parece ser el mejor índice para aplicar.

```
# 1. capscale() en base a datos crudos (sitios x especies)
bray.env.cap<- capscale (biomass ~ 1, env, add = "lingoes")#modelo mínimo
bray.env.cap.all <- capscale(biomass ~ . , env, add = "lingoes")# modelo completo
# Existen dos maneras de evitar autovalores negativos: una es agregando una constante #al análisis

fwd.sel.dbRDA<-ordistep (bray.env.cap, scope = formula (bray.env.cap.all), direction = "forward",
pstep = 1000,
trace = TRUE)

## 
## Start: biomass ~ 1
##
##          Df     AIC      F Pr(>F)
## + EC      1 435.62 7.3760  0.005 **
## + TN      1 436.35 6.4767  0.010 **
## + DOC     1 439.94 2.4469  0.080 .
## + a440    1 440.76 1.6257  0.180
## + TP      1 441.11 1.2859  0.295
## + TH      1 441.34 1.0725  0.345
## + SUVA254 1 441.40 1.0155  0.355
## + DIN     1 441.51 0.9092  0.370
## + E2_E3   1 441.53 0.8848  0.390
## + pH      1 441.68 0.7432  0.525
## + DRP     1 442.18 0.2880  0.735
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Step: biomass ~ EC
##
##          Df     AIC      F Pr(>F)
## + a440    1 433.14 4.2685  0.030 *
## + E2_E3   1 435.03 2.3593  0.070 .
## + TN      1 435.40 1.9954  0.155
## + pH      1 435.57 1.8349  0.160
## + TP      1 436.22 1.2399  0.345
## + SUVA254 1 436.60 0.8973  0.410
## + DIN     1 436.64 0.8577  0.475
## + DOC     1 436.87 0.6540  0.540
## + TH      1 437.12 0.4337  0.690
## + DRP     1 437.50 0.1099  0.935
```

```

## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Step: biomass ~ EC + a440
##
##          Df      AIC      F Pr(>F)
## + pH     1 431.31 3.3846 0.045 *
## + TN    1 432.60 2.1715 0.115
## + E2_E3 1 433.25 1.5858 0.210
## + DOC   1 433.79 1.1182 0.385
## + TP    1 434.29 0.6956 0.505
## + TH    1 434.31 0.6796 0.550
## + SUVA254 1 434.40 0.6032 0.570
## + DRP   1 434.72 0.3418 0.755
## + DIN   1 434.90 0.1953 0.910
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Step: biomass ~ EC + a440 + pH
##
##          Df      AIC      F Pr(>F)
## + E2_E3 1 430.27 2.4564 0.105
## + TN    1 431.11 1.7398 0.155
## + TP    1 431.80 1.1744 0.330
## + DRP   1 432.24 0.8201 0.480
## + SUVA254 1 432.73 0.4406 0.685
## + DOC   1 432.71 0.4525 0.690
## + DIN   1 432.94 0.2786 0.780
## + TH    1 433.05 0.1947 0.850

```

```
ordiplot(bray.env.cap.all, scaling = 2, type = "text")
```

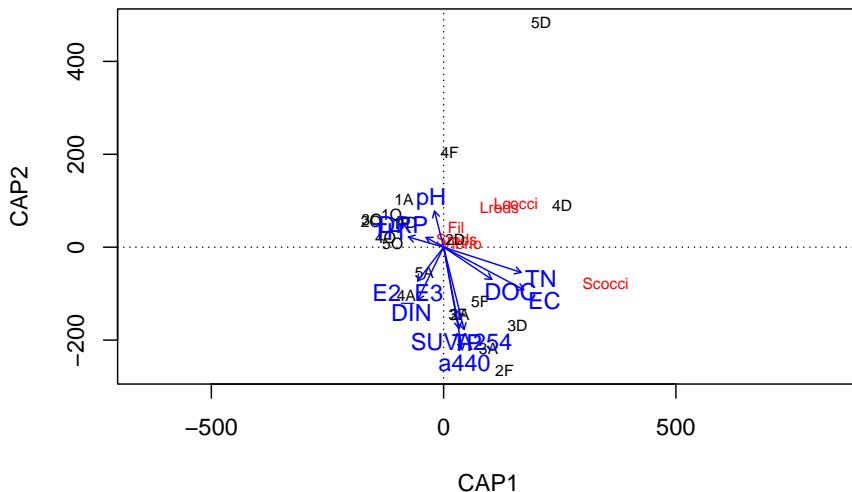


Figura 6: Ordenamiento obtenido a partir del dbRDA en base a la distancia de Bray Curtis. Escalamiento tipo 2.

Qué variables son retenidas por la selección *forward*?

fwd.sel.dbRDA\$call

```
## capscale(formula = biomass ~ EC + a440 + pH, data = env, add = "lingoes")
```

Cuál es el valor del R^2 ajustado de este modelo?

```
# Escribir el modelo  
biomass.dbrda.signif <- capscale(formula = biomass ~ EC + a440 + pH, data = env, add = "lingoes")
```

```
RsquareAdj(biomass.dbrda.signif)
```

```
## $r.squared  
## [1] 0.5320242  
##  
## $adj.r.squared  
## [1] 0.4442787
```

Test de significación

```

anova.cca(biomass.dbrda.signif, step = 1000) #Significación global del modelo final

## Permutation test for capscale under reduced model
## Permutation: free
## Number of permutations: 999
##
## Model: capscale(formula = biomass ~ EC + a440 + pH, data = env, add = "lingoes")
##          Df  Variance   F Pr(>F)
## Model      3 1861913203 6.0633 0.001 ***
## Residual 16 1637764593
## ---
## Signif. codes: 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

anova.cca(biomass.dbrda.signif, step = 1000, by = "term") #significación de variables

## Permutation test for capscale under reduced model
## Terms added sequentially (first to last)
## Permutation: free
## Number of permutations: 999
##
## Model: capscale(formula = biomass ~ EC + a440 + pH, data = env, add = "lingoes")
##          Df  Variance   F Pr(>F)
## EC         1 1017246043 9.9379 0.001 ***
## a440       1 498217165 4.8673 0.016 *
## pH         1 346449995 3.3846 0.046 *
## Residual 16 1637764593
## ---
## Signif. codes: 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

anova.cca(biomass.dbrda.signif, step = 1000, by = "axis") #significación de los ejes

## Permutation test for capscale under reduced model
## Forward tests for axes
## Permutation: free
## Number of permutations: 999
##
## Model: capscale(formula = biomass ~ EC + a440 + pH, data = env, add = "lingoes")
##          Df  Variance   F Pr(>F)
## CAP1       1 1503918383 14.6924 0.001 ***
## CAP2       1 278096049  2.7168 0.164
## CAP3       1  79898771  0.7806 0.512
## Residual 16 1637764593
## ---
## Signif. codes: 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

```

Colinealidad entre variables explicativas

```
vif.cca(biomass.dbrda.signif) #Si vif ("variable inflation factor") <5 no hay colinealidad

##          EC      a440      pH
## 1.932725 1.873615 1.153775
```

Existe otra manera de calcular dbRDA que fue propuesta por McArdle y Anderson (2001). Este método alternativo ejecuta el análisis directamente sobre la matriz de similitud de respuesta sin tener que pasar por un PCoA. Esta forma alternativa es proporcionada en el paquete **vegan** en la función **dbrda()**.

```
# dbRDA
#-----
# generar la matriz de distancia
biomass.dist <- vegdist(biomass, method="bray") # bray curtis

# Modelado
dbrda.0 <- dbrda(biomass.dist ~ 1, env, add=TRUE) #modelo mínimo
dbrda.all <- dbrda(biomass.dist ~ ., env, add=TRUE) #modelo completo
fwd.sel.dbRDA2<-ordistep (dbrda.0, scope = formula (dbrda.all), add=TRUE)

##
## Start: biomass.dist ~ 1
##
##          Df      AIC      F Pr(>F)
## + EC     1 18.715 6.6542  0.005 ***
## + TN     1 20.679 4.3484  0.005 ***
## + a440   1 22.817 2.0827  0.070 .
## + DOC    1 22.618 2.2839  0.075 .
## + DRP    1 23.525 1.3845  0.175
## + DIN    1 23.810 1.1098  0.270
## + TP     1 24.087 0.8474  0.435
## + TH     1 24.081 0.8529  0.485
## + pH     1 24.092 0.8425  0.505
## + SUVA254 1 24.037 0.8944  0.510
## + E2_E3   1 24.382 0.5712  0.760
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Step: biomass.dist ~ EC
##
##          Df      AIC      F Pr(>F)
## - EC    1 23.007 6.6542  0.005 **
```

```

## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
##          Df      AIC      F Pr(>F)
## + a440    1 18.018 2.4546  0.015 *
## + TN      1 19.061 1.4660  0.120
## + pH      1 19.078 1.4506  0.145
## + DRP     1 19.089 1.4399  0.200
## + E2_E3   1 19.284 1.2616  0.215
## + DIN     1 19.478 1.0851  0.330
## + DOC     1 19.703 0.8828  0.535
## + TH      1 19.807 0.7900  0.660
## + TP      1 19.883 0.7230  0.690
## + SUVA254 1 20.175 0.4660  0.965
##
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Step: biomass.dist ~ EC + a440
##
##          Df      AIC      F Pr(>F)
## - a440   1 18.715 2.4546  0.050 *
## - EC     1 22.817 6.8831  0.005 **
##
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
##          Df      AIC      F Pr(>F)
## + pH      1 17.824 1.8552  0.045 *
## + TN      1 18.218 1.5069  0.100 .
## + TP      1 18.175 1.5445  0.145
## + DRP     1 18.257 1.4724  0.185
## + E2_E3   1 18.937 0.8890  0.500
## + TH      1 18.889 0.9289  0.505
## + DOC     1 18.935 0.8905  0.570
## + SUVA254 1 19.014 0.8242  0.605
## + DIN     1 19.423 0.4831  0.960
##
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Step: biomass.dist ~ EC + a440 + pH
##
##          Df      AIC      F Pr(>F)
## - pH     1 18.018 1.8552  0.080 .
## - a440   1 19.078 2.8267  0.030 *
## - EC     1 23.836 7.8833  0.005 **
##
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

```

```

##          Df   AIC      F Pr(>F)
## + TP     1 17.506 1.8428  0.045 *
## + DRP    1 17.560 1.7980  0.070 .
## + TN     1 18.201 1.2684  0.270
## + E2_E3   1 18.352 1.1453  0.275
## + TH     1 18.613 0.9364  0.500
## + DOC    1 18.917 0.6956  0.765
## + SUVA254 1 18.944 0.6748  0.855
## + DIN    1 19.177 0.4931  0.930
## ---
## Signif. codes: 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Step: biomass.dist ~ EC + a440 + pH + TP
##
##          Df   AIC      F Pr(>F)
## - pH     1 18.175 2.1410  0.115
## - TP     1 17.824 1.8428  0.100 .
## - a440   1 20.167 3.9360  0.010 **
## - EC     1 24.511 8.5293  0.005 **
## ---
## Signif. codes: 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Step: biomass.dist ~ EC + a440 + TP
##
##          Df   AIC      F Pr(>F)
## + pH     1 17.506 2.1410  0.025 *
## + TN     1 18.222 1.5383  0.135
## + DRP    1 18.515 1.2980  0.210
## + DOC    1 18.855 1.0235  0.370
## + E2_E3   1 18.929 0.9646  0.450
## + TH     1 19.095 0.8322  0.525
## + SUVA254 1 19.184 0.7621  0.655
## + DIN    1 19.519 0.5004  0.940
## ---
## Signif. codes: 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Step: biomass.dist ~ EC + a440 + TP + pH
##
##          Df   AIC      F Pr(>F)
## - TP     1 17.824 1.8428  0.070 .
## - pH     1 18.175 2.1410  0.045 *
## - a440   1 20.167 3.9360  0.005 **
## - EC     1 24.511 8.5293  0.005 **
## ---
## Signif. codes: 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

```

```

##          Df      AIC      F Pr(>F)
## + DRP     1 16.699 2.1102  0.050 *
## + E2_E3   1 17.878 1.1876  0.220
## + TN      1 17.810 1.2392  0.310
## + TH      1 18.280 0.8858  0.550
## + DOC     1 18.389 0.8046  0.615
## + SUVA254 1 18.726 0.5574  0.860
## + DIN     1 18.762 0.5308  0.935
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Step: biomass.dist ~ EC + a440 + TP + pH + DRP
##
##          Df      AIC      F Pr(>F)
## - DRP    1 17.506 2.1102  0.065 .
## - TP     1 17.560 2.1532  0.060 .
## - pH     1 18.515 2.9436  0.030 *
## - a440   1 20.239 4.4688  0.010 **
## - EC     1 25.377 9.8785  0.005 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
##          Df      AIC      F Pr(>F)
## + TN     1 16.046 1.8438  0.025 *
## + DOC    1 17.329 0.9210  0.495
## + TH     1 17.397 0.8739  0.545
## + E2_E3   1 17.506 0.7989  0.690
## + SUVA254 1 17.757 0.6268  0.845
## + DIN    1 17.854 0.5611  0.915
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Step: biomass.dist ~ EC + a440 + TP + pH + DRP + TN
##
##          Df      AIC      F Pr(>F)
## - TN     1 16.699 1.8438  0.070 .
## - DRP    1 17.810 2.6923  0.035 *
## - TP     1 17.282 2.2832  0.030 *
## - pH     1 17.936 2.7909  0.030 *
## - a440   1 20.157 4.6463  0.005 **
## - EC     1 21.367 5.7462  0.005 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
##          Df      AIC      F Pr(>F)

```

```

## + E2_E3      1 16.474 0.9813  0.460
## + TH         1 16.532 0.9435  0.515
## + DOC        1 16.732 0.8149  0.640
## + DIN        1 17.065 0.6033  0.865
## + SUVA254   1 17.247 0.4891  0.965

summary(fwd.sel.dbRDA2)

##
## Call:
## dbrda(formula = biomass.dist ~ EC + a440 + TP + pH + DRP + TN,      data = env, add = TRUE)
##
## Partitioning of Lingoes adjusted squared Bray distance:
##           Inertia Proportion
## Total      3.009    1.0000
## Constrained 1.843    0.6125
## Unconstrained 1.166    0.3875
##
## Eigenvalues, and their contribution to the Lingoes adjusted squared Bray distance
##
## Importance of components:
##           dbRDA1 dbRDA2 dbRDA3 dbRDA4 dbRDA5 dbRDA6 MDS1
## Eigenvalue      1.1754 0.3222 0.13988 0.09198 0.06892 0.04464 0.30067
## Proportion Explained 0.3906 0.1071 0.04648 0.03057 0.02290 0.01484 0.09992
## Cumulative Proportion 0.3906 0.4977 0.54420 0.57477 0.59767 0.61251 0.71243
##           MDS2   MDS3   MDS4   MDS5   MDS6   MDS7   MDS8
## Eigenvalue      0.18028 0.16887 0.09947 0.08289 0.06645 0.05467 0.04915
## Proportion Explained 0.05991 0.05612 0.03306 0.02755 0.02208 0.01817 0.01633
## Cumulative Proportion 0.77234 0.82846 0.86151 0.88906 0.91115 0.92931 0.94565
##           MDS9   MDS10  MDS11  MDS12  MDS13
## Eigenvalue      0.04332 0.03912 0.03779 0.028308 0.015003
## Proportion Explained 0.01440 0.01300 0.01256 0.009408 0.004986
## Cumulative Proportion 0.96005 0.97305 0.98561 0.995014 1.000000
##
## Accumulated constrained eigenvalues
## Importance of components:
##           dbRDA1 dbRDA2 dbRDA3 dbRDA4 dbRDA5 dbRDA6
## Eigenvalue      1.1754 0.3222 0.13988 0.09198 0.06892 0.04464
## Proportion Explained 0.6378 0.1748 0.07589 0.04990 0.03739 0.02422
## Cumulative Proportion 0.6378 0.8126 0.88848 0.93839 0.97578 1.000000
##
## Scaling 2 for species and site scores
## * Species are scaled proportional to eigenvalues
## * Sites are unscaled: weighted dispersion equal on all dimensions
## * General scaling constant of scores: 2.749779

```

```

##  

##  

## Site scores (weighted sums of species scores)  

##  

##      dbRDA1   dbRDA2   dbRDA3   dbRDA4   dbRDA5   dbRDA6  

## 10 -0.6376 -0.12804 -0.03876 -0.70720  0.94168  0.36537  

## 20 -1.0783 -0.69197 -0.97096 -1.34720  0.71191 -0.47784  

## 30 -1.0880 -1.38047 -0.59586  1.40988 -0.69450 -1.79001  

## 40 -0.8487  0.10292 -0.28861 -0.92574  0.42778  0.03716  

## 50 -0.7566  0.50129 -0.26091  0.09517 -1.15429  0.92082  

## 1D -0.2764  0.32189  0.83648 -0.29089  0.08027  1.14611  

## 2D  0.3804  0.23554  0.42590 -0.31229 -0.26397 -1.15533  

## 3D  0.7486 -0.01581 -0.61532 -0.46519  0.90639 -0.95405  

## 4D  0.9702 -1.35302  0.40833 -0.12695  0.09472 -0.02752  

## 5D  0.9299 -0.68352 -0.45440 -0.57097 -1.55774 -0.23787  

## 1F -0.3959  0.17973  0.37789  0.03529 -0.71031  1.86816  

## 2F  0.6261  0.26268 -0.74995 -0.15072  0.10641  0.39856  

## 3F  0.2837  0.61556 -0.84461  0.93364 -0.32083 -0.36893  

## 4F  0.4400 -0.45454  1.88536 -0.37386 -0.86192 -0.15695  

## 5F  0.5894  0.10828 -0.07868  0.57239  1.12282  1.09516  

## 1A -0.3431 -0.58540  1.06498  1.46516  0.19695  0.82287  

## 2A  0.3467  0.61790 -0.80926  0.10200 -0.22206  0.05421  

## 3A  0.5703  0.22123 -0.25632  0.75529  1.13742  0.82965  

## 4A -0.3562  1.23612  0.69753  0.02256  0.08127 -1.68110  

## 5A -0.1043  0.88964  0.26718 -0.12037 -0.02199 -0.68847  

##  

##  

## Site constraints (linear combinations of constraining variables)  

##  

##      dbRDA1   dbRDA2   dbRDA3   dbRDA4   dbRDA5   dbRDA6  

## 10 -0.65387 -0.40828 -0.42250 -0.259400  0.49969 -0.28818  

## 20 -0.78702 -0.42489 -0.22204 -0.899555  0.63246  0.70496  

## 30 -0.87594 -1.11224 -0.44687  1.005626 -0.22788 -1.36697  

## 40 -0.68672 -0.01060 -0.24895 -0.689544  0.05814  0.03217  

## 50 -1.12487  0.57403 -0.68306 -0.638230 -0.53740  0.83041  

## 1D -0.59364  0.13277  1.00816 -0.222758 -0.42824 -0.28122  

## 2D  0.17076  0.03359 -0.01648 -0.571875 -0.35579 -1.22769  

## 3D  0.61036 -0.02806 -0.41357 -0.275512  0.92014 -0.56471  

## 4D  0.92221 -1.27397  0.85630 -0.632025 -0.21743  0.31357  

## 5D  0.76516 -0.28788 -0.67397 -0.015505 -1.06698 -0.07148  

## 1F  0.11361  0.13108  0.24730  0.011029 -0.60317  0.52747  

## 2F  0.53002 -0.06287 -0.37766 -0.002092 -0.46726  0.44806  

## 3F  0.37249  1.01009 -0.61368  0.636949 -0.62408  0.28100  

## 4F  0.51331 -0.19979  1.02982 -0.509022  0.09064  0.17621  

## 5F  0.56656  0.29261 -0.54748  0.036357  1.42845  0.15755  

## 1A -0.56450 -0.61460  0.73390  1.597780  0.14837  1.00394

```

```

## 2A  0.56228 -0.09108 -0.64944  0.424847 -0.29278  0.30415
## 3A  0.35973  0.19731 -0.10291  0.694853  0.93742  0.09024
## 4A -0.19700  1.20222  1.02435  0.219285  0.37901 -0.29066
## 5A -0.00293  0.94059  0.51877  0.088795 -0.27329 -0.77881
##
##
## Biplot scores for constraining variables
##
##      dbRDA1   dbRDA2   dbRDA3 dbRDA4   dbRDA5   dbRDA6
## EC    0.80555  0.24747  0.333697 0.3423  0.15567 -0.19257
## a440  0.31896  0.71101  0.014811 0.3302  0.53142 -0.03271
## TP    0.23805  0.07677  0.001941 0.3340  0.89976  0.12779
## pH    -0.02237 -0.19882  0.904131 0.1029  0.01728  0.36280
## DRP   -0.21460 -0.55134 -0.303238 0.7271  0.04008 -0.16661
## TN    0.68113  0.03486 -0.212372 0.4854 -0.13647  0.48527

```

5.4.4.0.1 Desafío: aplicar los códigos correspondientes para para graficar, enumerar las variables significativas, solicitar el R^2 , evaluar la colinealidad y las distintas significaciones del modelo aquí utilizado.

5.5 Análisis de Correspondencia Canónica (CCA)

El Análisis de Correspondencia Canónica (CCA) es apropiado para modelar respuestas unimodales o en forma de joroba a variables explicativas (en lugar de lineales como vimos con RDA). En este caso, el modelo maximiza la correlación entre especies y los *scores* de las muestras (*i.e.* sitios). Esto lo hace después de aplicar una regresión lineal múltiple para “restringir” cada muestra a los ejes de ordenación que constituyen combinaciones lineales de las variables ambientales medidas.

Podemos pensar al CCA com un RDA “ponderado” que preserva la distancia chi-cuadrado entre sitios (la distancia de chi-cuadrado es, en sí, un ejemplo de distancia euclídea ponderada). En el triplot (gráficos que incluyen sitios, variables ambientales y especies) del CCA, las especies se ordenan a lo largo de ejes canónicos siguiendo sus óptimos ecológicos, lo que permite una interpretación ecológica relativamente fácil. Sin embargo, se debe tener en cuenta que, dada las características de la distancia de chi-cuadrado, el algoritmo aumenta el “peso” de las especies de baja frecuencia en relación con las de mayor frecuencia. En consecuencia, se recomienda aplicar el modelo en los casos en que las especies raras están bien representadas y se consideran indicadores potenciales de características particulares del ecosistema; o bien, las especies “raras” deben eliminarse antes de modelar los datos con CCA. Otro punto a tener en

cuenta es que el modelo genera una medida sesgada de variación explicada. Estas debilidades parecen que han llevado a muchxs ecológxs a indicar que “la distancia chi-cuadrado fue una de las peores distancias utilizadas para analizar para los datos de composición de la comunidad”. Lo que esxs ecológxs no tenían presente es que estos algoritmos son sólo modelos es decir, formas sintéticas de representar una realidad compleja. La complejidad de la realidad hace que no exista el modelo perfecto ni único, sino que tenemos varios modelos posibles e imperfectos para tratar de sintetizar y explicar el mundo... Compartimos estas ideas para que, entre café y café, piensen y discutan entre ustedes :).

Pongamos manos a la obra y veamos cómo funciona trabajando con un ejemplo: en esta oportunidad, utilizaremos un set de datos del trabajo *Alternative states drive the patterns in the bacterioplankton composition in shallow Pampean lakes (Argentina)* publicado en *Environmental Microbiology Reports* (Llames *et al.* 2013). Para ello, descargar el set de datos **DGGE_cca.csv** y **DGGE_fq_cca_tunned.csv** de GitHub Limno-con-R/CILCAL2023. Guardar los archivos en una carpeta llamada *data*, dentro del **Directorio de Trabajo del Proyecto** que creamos para esta Unidad (ver cómo hacerlo en la Unidad ??). Finalmente, instalar los paquetes como se indica en la Unidad ??.

Luego, cargarlos en la sesión.

Una vez descargados los archivo, generamos el objeto en nuestro entorno de la matriz de composición de especies (en este caso, relevamiento de OTU's ("unidades taxonómicas operativas") por técnica de DGGE) y la matriz de datos limnológicos. Seguidamente, evaluamos el tipo de respuesta en la matriz de composición bacteriana:

```

library (vegan)
data_cca <- read.csv("./data/DGGE_cca.csv", row.names=1)
env_cca<-read.csv("./data/DGGE_fq_cca_tunned.csv", row.names =1)

# Cómputo del DCA para verificar la longitud del gradiente
#----

vare.dca_cca <- decorana(data_cca)
vare.dca_cca

## 
## Call:
## decorana(veg = data_cca)
##
## Detrended correspondence analysis with 26 segments.
## Rescaling of axes with 4 iterations.
## Total inertia (scaled Chi-square): 5.6709
##
##          DCA1     DCA2     DCA3     DCA4

```

```
## Eigenvalues      0.5616 0.4329 0.3330 0.2931
## Additive Eigenvalues 0.5616 0.4309 0.3338 0.2920
## Decorana values    0.6237 0.3876 0.2863 0.1524
## Axis lengths       4.1986 3.7111 2.8788 2.3732
```

Como podemos observar, la longitud del gradiente es mayor a 4 S.D.. Por lo tanto, el modelo de CCA de respuesta unimodal es el modelo adecuado para este set de datos.

5.5.1 CCA utilizando la función cca() del paquete vegan

Luego de corroborar el tipo de respuesta de la matriz de composición, procedemos a la estandarización de las variables ambientales.

```
#Estandarización de matriz ambiental
env.r<-decostand (env_cca, method = "standardize") #N~(0, 1)
```

Ejecutemos un CCA utilizando la función **cca()** de **vegan** y la expresión de fórmula. Es importante señalar que, en un CCA, no es recomendable la transformación de los datos de composición tal como se puede realizar en RDA ya que al transformar, la distancia preservada ya no sería la distancia chi-cuadrado y los resultados no podrían ser interpretados.

```
cca_DGGE <- cca(data_cca ~ ., data=env.r)#modelo de máxima, con todas las variables

RsquareAdj(cca_DGGE)

## $r.squared
## [1] 0.4143396
##
## $adj.r.squared
## [1] 0.1432949

cca_DGGE

## Call: cca(formula = data_cca ~ Turb + DO + T + Cond + pH + Chl.a + TN +
##           TP + SS + OM + IM + Alk, data = env.r)
##
##                  Inertia Proportion Rank
## Total          5.6709     1.0000
## Constrained   2.3497     0.4143    12
## Unconstrained 3.3212     0.5857    26
## Inertia is scaled Chi-square
```

```

## 
## Eigenvalues for constrained axes:
##   CCA1   CCA2   CCA3   CCA4   CCA5   CCA6   CCA7   CCA8   CCA9   CCA10  CCA11
## 0.5366 0.3764 0.3502 0.2387 0.1981 0.1458 0.1356 0.1155 0.0908 0.0767 0.0517
##   CCA12
## 0.0336
##
## Eigenvalues for unconstrained axes:
##   CA1    CA2    CA3    CA4    CA5    CA6    CA7    CA8
## 0.4053 0.3675 0.3326 0.2724 0.2434 0.2090 0.1670 0.1579
## (Showing 8 of 26 unconstrained eigenvalues)

summary(cca_DGGE)

## 
## Call:
## cca(formula = data_cca ~ Turb + DO + T + Cond + pH + Chl.a + TN + TP + SS + OM
##
## Partitioning of scaled Chi-square:
##           Inertia Proportion
## Total      5.671     1.0000
## Constrained 2.350     0.4143
## Unconstrained 3.321     0.5857
##
## Eigenvalues, and their contribution to the scaled Chi-square
##
## Importance of components:
##          CCA1   CCA2   CCA3   CCA4   CCA5   CCA6   CCA7
## Eigenvalue 0.53664 0.37637 0.35022 0.23869 0.19810 0.14584 0.13557
## Proportion Explained 0.09463 0.06637 0.06176 0.04209 0.03493 0.02572 0.02391
## Cumulative Proportion 0.09463 0.16100 0.22276 0.26485 0.29978 0.32550 0.34941
##          CCA8   CCA9   CCA10  CCA11  CCA12   CA1    CA2
## Eigenvalue 0.11545 0.09075 0.07672 0.051728 0.033583 0.40530 0.3675
## Proportion Explained 0.02036 0.01600 0.01353 0.009122 0.005922 0.07147 0.0648
## Cumulative Proportion 0.36976 0.38577 0.39930 0.408418 0.414340 0.48581 0.5506
##          CA3    CA4    CA5    CA6    CA7    CA8    CA9
## Eigenvalue 0.33257 0.27241 0.24336 0.20904 0.16698 0.15786 0.14827
## Proportion Explained 0.05865 0.04804 0.04291 0.03686 0.02944 0.02784 0.02615
## Cumulative Proportion 0.60926 0.65729 0.70021 0.73707 0.76651 0.79435 0.82050
##          CA10   CA11   CA12   CA13   CA14   CA15   CA16
## Eigenvalue 0.13252 0.12870 0.10553 0.09177 0.08762 0.08385 0.07123
## Proportion Explained 0.02337 0.02269 0.01861 0.01618 0.01545 0.01479 0.01256
## Cumulative Proportion 0.84386 0.86656 0.88517 0.90135 0.91680 0.93159 0.94415
##          CA17   CA18   CA19   CA20   CA21   CA22
## Eigenvalue 0.05733 0.052902 0.041940 0.034947 0.03425 0.02501

```

```

## Proportion Explained 0.01011 0.009329 0.007396 0.006163 0.00604 0.00441
## Cumulative Proportion 0.95426 0.963588 0.970984 0.977146 0.98319 0.98760
## CA23 CA24 CA25 CA26
## Eigenvalue 0.022464 0.021070 0.019127 0.007679
## Proportion Explained 0.003961 0.003716 0.003373 0.001354
## Cumulative Proportion 0.991557 0.995273 0.998646 1.000000
##
## Accumulated constrained eigenvalues
## Importance of components:
## CCA1 CCA2 CCA3 CCA4 CCA5 CCA6 CCA7
## Eigenvalue 0.5366 0.3764 0.3502 0.2387 0.19810 0.14584 0.1356
## Proportion Explained 0.2284 0.1602 0.1491 0.1016 0.08431 0.06207 0.0577
## Cumulative Proportion 0.2284 0.3886 0.5376 0.6392 0.72352 0.78559 0.8433
## CCA8 CCA9 CCA10 CCA11 CCA12
## Eigenvalue 0.11545 0.09075 0.07672 0.05173 0.03358
## Proportion Explained 0.04914 0.03862 0.03265 0.02202 0.01429
## Cumulative Proportion 0.89242 0.93104 0.96369 0.98571 1.000000
##
## Scaling 2 for species and site scores
## * Species are scaled proportional to eigenvalues
## * Sites are unscaled: weighted dispersion equal on all dimensions
##
## Species scores
##
## CCA1 CCA2 CCA3 CCA4 CCA5 CCA6
## otu_1 -0.403249 -0.955155 0.763847 -1.40800 0.749197 -0.91516
## otu_2 -0.802117 -0.469852 0.256538 -0.05351 0.367509 -1.20707
## otu_3 -0.249106 -0.263044 0.146922 -0.07675 -0.400698 0.01643
## otu_4 -0.375787 -0.725437 0.006064 0.15550 0.435117 0.22965
## otu_5 -1.000683 -0.690005 0.582915 -0.11725 -0.259836 0.56259
## otu_6 -1.785162 0.840869 0.106715 -0.34910 0.466184 -0.94946
## otu_7 -0.817495 0.732667 -0.250358 0.54115 -0.579294 -0.23965
## otu_8 -0.067389 -0.809976 -0.190006 -0.29167 -0.570023 0.35885
## otu_9 -0.242941 -0.469628 -0.892187 0.07382 0.170892 -0.06897
## otu_10 0.549038 0.329635 0.316105 -0.46102 0.043947 0.12611
## otu_11 0.522378 -0.471719 0.321634 0.52502 -0.585695 1.06369
## otu_12 0.732276 -0.420509 0.362197 0.20898 -0.430340 0.59674
## otu_13 0.425700 0.208830 0.313627 -0.12682 0.146167 0.17270
## otu_14 1.041453 0.994741 0.357850 0.17837 -0.015604 0.15856
## otu_15 0.986178 1.128844 0.923651 -0.24931 0.383242 -0.04810
## otu_16 0.536141 -0.132200 0.267277 -0.27604 -0.073920 0.28362
## otu_17 0.015017 0.278275 1.326819 -0.46991 0.459386 -0.54252
## otu_18 0.486071 -0.158579 0.627933 -0.47796 0.320389 -0.28083
## otu_19 0.101239 -0.687806 -0.693236 -0.15157 0.007127 0.32022
## otu_20 -0.690556 0.287948 -0.158349 0.94271 1.286098 1.24778

```

```

## otu_21 -0.297528  0.073599  0.348741  0.52100  0.646365 -0.08760
## otu_22 -1.527500  0.343887 -0.227218  0.43527  1.378051  0.22051
## otu_23 -0.439724  0.482016 -0.317722  0.63857 -0.017165 -0.99339
## otu_24 -0.009932 -0.582498 -0.212126 -0.19477 -0.233910 -0.54177
## otu_25  0.279766 -0.707598  0.099509 -0.61727  0.149779 -0.32265
## otu_26 -0.287915 -0.785217  0.160233 -0.39889  0.891188 -0.13645
## otu_27  0.058509 -0.884015 -0.318393 -0.02105  0.380620  0.35233
## otu_28 -0.269128  0.120909 -0.158513 -0.24697 -0.016550 -0.14433
## otu_29 -0.564916  0.155731 -0.389144  0.13397 -0.068322 -0.24544
## otu_30 -0.172715  0.790504  0.580147  0.11936 -0.236480  0.22945
## otu_31 -0.150320 -0.371168 -0.917728  0.45088  0.195741  0.03331
## otu_32 -0.048324  0.002044 -0.466621  0.63247 -0.042630 -1.05791
## otu_33 -1.992309  2.022882 -1.943319 -1.74227 -0.031000  0.56095
## otu_34  0.561034  0.579849  0.158790 -0.26669  0.599610 -0.60735
## otu_35  0.453956 -0.287475  0.271138 -0.36441  0.175697  0.07431
## otu_36 -0.514859  0.299528 -0.126874  1.43021  0.507622 -0.25276
## otu_37  0.054169 -1.016254 -0.180183  0.02552  0.675691  0.04312
## otu_38 -0.385252 -0.020055 -0.530828  0.59703  0.737244 -0.14189
## otu_39 -0.354540  0.394670 -0.060948 -0.00325  0.162600 -0.45938
## otu_40 -0.283637  0.209192 -0.400436  0.61968  0.741545 -0.07761
## otu_41  0.562247  0.204309  0.426973 -0.42377  0.544602 -0.51472
## otu_42  1.036005  1.005882  0.323829  0.86793  0.056946 -0.82951
## otu_43  0.489902  0.583573  0.200813  0.27384 -0.173743  0.01413
## otu_44  0.313163 -0.718301  0.130694 -0.66577  0.420362 -0.36566
## otu_45 -0.292266 -0.166925 -0.343672  0.23232 -0.416471  0.01441
## otu_46  0.979977  0.881874  0.439167 -0.41424  0.236389  0.17093
## otu_47  0.009391 -0.076228 -0.596509  0.39290  0.032468 -0.77207
## otu_48  0.335868 -0.581428  0.158930 -1.01640  0.308862 -0.26091
## otu_49 -0.321809 -0.078012 -0.209954  0.55103 -0.293267 -0.66216
## otu_50  0.211536 -0.919157 -0.006439 -0.67871  0.142491 -0.09779
## otu_51  0.135106 -0.917477  0.096837  0.09523 -0.776378  0.82080
## otu_52  0.098428 -1.089263  0.083656 -0.53499 -0.086324 -0.14263
## otu_53 -0.281993 -0.377860  0.144662 -0.70469 -0.037245 -0.17924
## otu_54 -0.067520 -0.548279 -0.145160  0.40939  0.481432  0.27200
## otu_55 -0.435988 -0.512198 -0.301248  0.48363 -0.896383  0.43581
## otu_56 -0.523627 -0.280158 -0.805672  2.22615  4.115503  2.20437
## otu_57 -0.539141  0.827100 -0.093621  0.76073  0.019996 -0.17229
## otu_58 -4.211443 -0.016031  3.130961  0.06535 -0.052594  0.19969
##
##
## Site scores (weighted averages of species scores)
##
##          CCA1      CCA2      CCA3      CCA4      CCA5      CCA6
## Ch1   -0.48091  0.14793 -0.72179  0.967598  0.71012 -2.014562
## Ch2   -0.02767  0.42956 -0.80076  1.841698  0.27701 -3.048024
## Ch3   -0.54718 -0.45035 -1.15634  1.335847 -1.07744 -0.292007

```

```

## Ch4 -0.22676 0.40502 -0.34890 1.302204 0.05435 -0.904778
## Ch5 -0.34683 0.52599 -0.29463 1.208809 0.13836 -1.820431
## Ch6 -0.44878 0.28019 -0.54669 0.968681 0.11377 -1.619152
## Ch7 0.11436 0.60145 -0.31167 1.016114 0.40502 -2.119718
## Sj1 -0.58539 -0.59687 -0.99574 1.032564 -2.30883 0.524776
## Sj2 -0.47505 0.21451 -0.12785 0.779782 0.47234 -0.008115
## Sj3 0.04097 0.28496 -0.51683 0.352077 -0.02758 -1.407831
## Sj4 -0.73046 0.50013 -0.72143 0.959823 -0.54320 -0.493739
## Sj5 -0.48070 0.47360 -0.06971 0.846169 -1.06375 -0.442025
## Sj6 -2.39149 2.98782 -3.40268 -4.038014 -0.11222 1.904247
## Sj7 -4.85594 0.41664 4.18175 -0.138159 -0.06739 0.751706
## Li1 1.19879 1.75563 0.78942 0.183581 -0.23759 0.466829
## Li2 1.19484 1.42234 0.83039 -0.251825 0.27663 0.640686
## Li3 1.25456 1.00370 1.14757 -0.814304 0.72710 -0.057785
## Li4 1.22912 1.14540 1.17642 -1.094040 0.81625 0.259254
## Li5 1.14399 1.56904 0.90853 -0.064662 -0.06647 0.359280
## Li6 0.66985 0.59059 1.55112 -1.230534 1.35906 -0.874507
## Li7 0.97626 1.74412 0.77911 0.962003 -0.67934 0.230903
## Lac1 0.96710 -0.60034 0.70855 0.244492 -1.17189 2.123951
## Lac2 0.85680 -0.68278 0.59459 0.528395 -1.38373 2.868070
## Lac3 0.42783 0.02065 0.08703 0.500905 -1.08345 1.301757
## Lac6 0.73740 0.45566 0.20920 -0.663942 -0.11125 0.917789
## Lac7 0.22137 0.03881 -0.12714 0.009287 -0.14123 0.102701
## K1 -0.03765 -0.91452 -0.28883 0.376583 -1.13953 1.756458
## K3 -0.13132 -1.56226 -0.39158 -0.913921 0.69263 -0.103756
## K4 0.34312 -1.63530 -0.13629 -1.273456 1.85320 -1.085206
## K5 0.51953 -1.62785 0.21899 -2.375443 1.10828 -1.051285
## K6 0.33029 -1.64605 0.11098 -1.999444 1.28244 -0.694926
## K7 0.12477 -0.72355 -1.31292 0.135426 0.31204 0.800331
## T1 -0.52157 -0.42682 -0.89998 2.420251 4.65795 2.519162
## T2 -0.33979 -1.05034 -1.03857 0.520391 -0.53674 0.482352
## T3 -0.06759 -0.96145 -0.27962 0.224592 -0.93565 -0.830518
## T4 0.26769 -1.14757 0.18755 -0.737555 -0.85175 0.668473
## T5 0.13253 -0.82928 0.19881 -0.781146 -0.72115 0.414811
## T6 0.04019 -0.87695 0.20887 -0.482980 -1.07860 0.834443
## T7 -0.09623 -1.28122 0.60018 -1.856533 0.08275 -1.060346
##
##
## Site constraints (linear combinations of constraining variables)
##
##          CCA1      CCA2      CCA3      CCA4      CCA5      CCA6
## Ch1 -0.11497 -0.561320 -0.32280 -0.07746  0.45218 -1.490609
## Ch2  0.19921  0.001145 -1.02810  0.92143 -0.20477 -0.976480
## Ch3 -0.05885 -0.133435 -0.56208  0.65543 -0.08223 -0.482436
## Ch4 -0.53914  0.827100 -0.09362  0.76073  0.02000 -0.172291
## Ch5 -0.18998  0.327785  0.04802  0.66021  1.00885 -2.665591

```

```

## Ch6 -0.70354 0.325544 -0.64083 1.14286 0.01094 -1.877311
## Ch7 0.25248 0.859022 -0.45437 0.69229 0.04675 -1.054281
## Sj1 -0.99048 -0.541330 -0.37653 0.35628 -1.19988 0.142165
## Sj2 -1.34644 0.494263 0.46828 -0.61544 -0.46903 1.479350
## Sj3 -0.33066 -0.090220 -0.56893 -0.04976 -0.13875 -0.038845
## Sj4 -0.57695 0.475621 0.15127 1.45084 -0.84077 -0.132499
## Sj5 -0.85663 1.251094 -0.41979 0.97224 -0.80434 -0.730943
## Sj6 -1.97484 2.680997 -3.07550 -2.58762 0.06848 0.842234
## Sj7 -4.21144 -0.016031 3.13096 0.06535 -0.05259 0.199689
## Li1 1.24243 1.078357 -0.08415 -0.75353 -0.11049 0.352234
## Li2 1.23009 1.345893 0.52021 -0.45519 0.58094 -0.273697
## Li3 1.03600 1.005882 0.32383 0.86793 0.05695 -0.829511
## Li4 1.44902 1.468429 1.09153 -1.18106 1.16376 0.586290
## Li5 1.41212 1.487502 1.37592 -0.43910 0.22383 0.227352
## Li6 0.31871 0.617924 2.33598 -0.82550 0.62355 -0.254105
## Li7 0.82629 1.917932 0.78517 1.54294 -1.01372 0.247933
## Lac1 0.75541 -0.757728 0.31997 0.17677 -0.57608 0.723587
## Lac2 0.54736 -0.622072 0.53684 0.49536 -0.71470 1.163215
## Lac3 0.76406 -0.355896 -0.07772 0.75115 -0.37755 0.825887
## Lac6 0.56588 -0.456085 -0.22960 0.02063 -0.43787 0.797464
## Lac7 0.41968 0.022614 -0.10481 -1.78301 0.04780 0.670512
## K1 -0.09996 -0.534380 -0.08935 0.95218 -1.53355 1.810454
## K3 0.23873 -1.355406 -0.31829 -0.68292 0.49798 -0.256958
## K4 0.20171 -1.612935 0.30825 -0.40511 1.12413 -1.367122
## K5 0.58826 -1.053849 -0.05872 -0.70543 0.02761 -0.548415
## K6 -0.20230 -0.815308 -0.31469 -1.69726 0.74179 0.005042
## K7 -0.17989 -0.541524 -1.82766 -0.13037 -0.05125 0.190001
## T1 -0.52157 -0.426815 -0.89998 2.42025 4.65795 2.519160
## T2 0.29273 -1.053246 -0.54412 0.22782 -0.94081 0.239311
## T3 0.31231 -0.978487 -0.66909 0.14810 -1.25561 0.380140
## T4 -0.09293 -1.364876 -0.38312 -0.48768 0.04336 -0.708574
## T5 0.10643 -0.790164 0.09642 -0.58304 -0.80361 -0.166382
## T6 0.59634 -1.006587 0.70708 -0.08594 -0.63231 1.415544
## T7 -0.36478 -1.119223 0.94380 -1.73478 0.84377 -0.791904
##
##
## Biplot scores for constraining variables
##
##          CCA1      CCA2      CCA3      CCA4      CCA5      CCA6
## Turb -0.25564 0.68668 0.200499 0.140198 0.05446 -0.42916
## DO   -0.22050 0.05761 0.004259 -0.423606 0.10599 -0.35042
## T    0.02446 0.06865 -0.061948 -0.207869 0.01437 -0.08840
## Cond -0.08886 -0.25025 -0.123892 0.175101 -0.16592 0.10994
## pH   -0.31139 0.29588 0.055242 -0.041248 -0.09994 0.01941
## Chl.a -0.84661 0.42182 0.035439 0.004208 -0.18410 0.05524
## TN   -0.57544 0.23569 0.205206 -0.289341 0.05494 0.13753

```

```

## TP      0.22105  0.81562  0.252643  0.210932  0.03693 -0.33729
## SS      0.03714  0.74664  0.091780 -0.003759  0.12249 -0.33827
## OM     -0.59797  0.55302 -0.231386  0.079057 -0.10699 -0.26828
## IM      0.28466  0.66859  0.202521 -0.036285  0.18911 -0.29580
## Alk    -0.14734  0.17538  0.088390 -0.003335 -0.24797  0.35383

```

Hay que tener presente que en un CCA se calculan dos conjuntos de “*scores*”: puntajes “**LC**” y puntajes “**WA**”. Los scores LC (“*linear constraints*”) corresponden a los *scores* de los objetos en función de la combinación lineal de las variables ambientales, mientras que los puntajes de WA (“*weigthed averaged*”) corresponden a los *scores* de los objetos en función del “peso” de las especies. Para tener en cuenta, por *default* el paquete **vegan** las gráficas de ordenación CCA de los sitios están utilizando *scores* WA (revisar salida anterior en “*Sites scores*”), mientras que en CANOCO 5 se grafican usando los *scores* LC. El uso de cada método de puntuación tiene sus defensores y detractores. Dado que pueden existir diferencias entre ambos tipos de gráficos, no hay que olvidar informar qué *score* se ha elegido mostrar, ya sea LC o WA.

Revisemos este punto gráficamente: generemos la figura correspondiente a la ordenación de los sitios “*default*” de **vegan** y solicitemos, además, el *score* de los sitios en función de LC. Luego, para que resulte más evidente, vamos a trazar una línea de unión que marque la posición de un mismo sitio en cada caso:

```

plot(cca_DGGE, dis=c("wa","lc"))
ordispider(cca_DGGE)

```

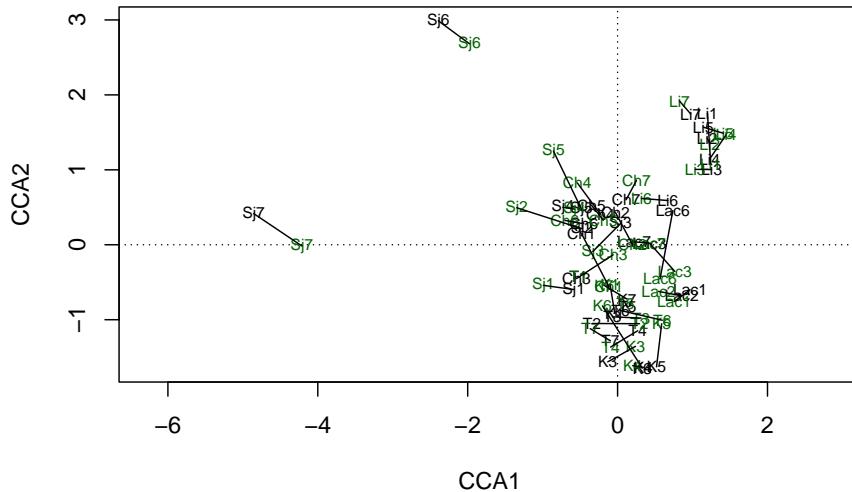


Figura 7: Diferencias entre *scores* de los sitios con LC y *scores* de los sitios con WA en CCA.

5.5.1.1 Triplot del CCA

El código para producir triplots es similar al utilizado para RDA, excepto que las variables de respuesta (especies) están representadas por puntos y, por lo tanto, las flechas no están disponibles.

```
# scaling=1: foco en la relación de distancias entre sitios
# scores de sitios como promedios ponderados de la especie (WA)
plot(cca_DGGE, scaling = 1, display = c("sp", "lc", "cn"), main = "Triplot CCA DGGE -scal")
```

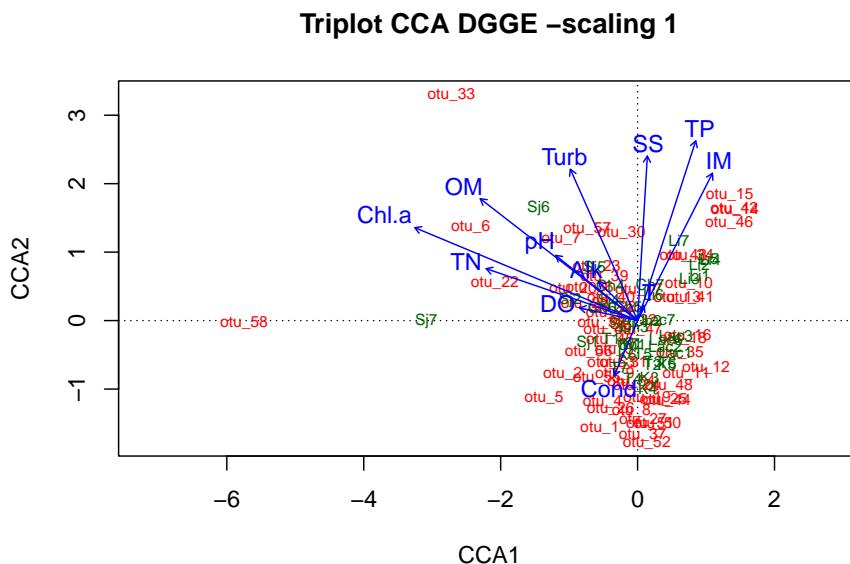


Figura 8: Triplot del modelo completo de CCA con escalamiento 1.

```
# Default scaling 2: foco en la relación de distancias entre otu's,
#scores de otu's como promedio ponderado de los sitios (WA)
plot(cca_DGGE, display = c("sp", "lc", "cn"), main = "Triplot CCA DGGE - scaling 2")
```

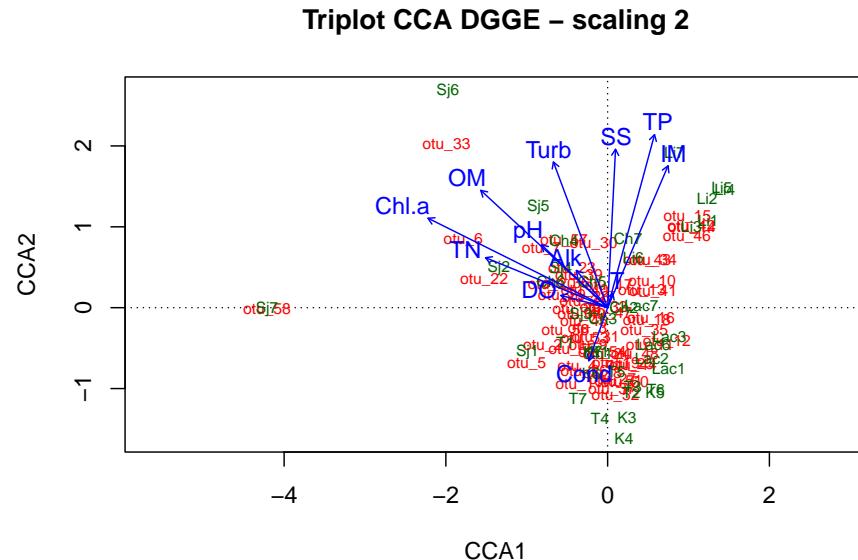


Figura 9: Triplot del modelo completo de CCA con escalamiento 2.

5.5.1.2 Test permutacionales y selección de variables

La significación del modelo puede analizarse de la misma manera que para el RDA

```
anova(cca_DGGE, permutations = how(nperm = 999)) #Modelo completo
```

```
## Permutation test for cca under reduced model
## Permutation: free
## Number of permutations: 999
##
## Model: cca(formula = data_cca ~ Turb + DO + T + Cond + pH + Chl.a + TN + TP + SS + IM + Cond + DO + pH + Turb + Chl.a + TN + TP + SS + IM)
##          Df ChiSquare      F Pr(>F)
## Model     12    2.3497 1.5329  0.001 ***
## Residual 26    3.3212
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
anova(cca_DGGE, by = "axis", permutations = how(nperm = 999)) #Significación de los ejes
```

```
## Permutation test for cca under reduced model
```

```

## Forward tests for axes
## Permutation: free
## Number of permutations: 999
##
## Model: cca(formula = data_cca ~ Turb + DO + T + Cond + pH + Chl.a + TN + TP + SS + OM + IM + Alk)
##          Df ChiSquare      F Pr(>F)
## CCA1      1   0.5366 4.2011  0.011 *
## CCA2      1   0.3764 2.9464  0.014 *
## CCA3      1   0.3502 2.7417  0.043 *
## CCA4      1   0.2387 1.8686  0.723
## CCA5      1   0.1981 1.5508  0.922
## CCA6      1   0.1458 1.1417  1.000
## CCA7      1   0.1356 1.0613  0.998
## CCA8      1   0.1155 0.9038  1.000
## CCA9      1   0.0908 0.7105  1.000
## CCA10     1   0.0767 0.6006  1.000
## CCA11     1   0.0517 0.4050  1.000
## CCA12     1   0.0336 0.2629  1.000
## Residual 26   3.3212
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

#Selección forward de variables
cca.step.forward <- ordistep(cca(data_cca ~ 1, data = env.r), scope = formula(cca_DGGE),
direction = "forward", permutations = how(nperm = 199))

##
## Start: data_cca ~ 1
##
##          Df      AIC      F Pr(>F)
## + Chl.a  1 247.92 3.3236  0.005 **
## + OM    1 248.75 2.4791  0.005 **
## + TP    1 248.92 2.3023  0.005 **
## + TN    1 249.23 1.9917  0.005 **
## + SS    1 249.43 1.7943  0.005 **
## + Turb  1 249.31 1.9118  0.010 **
## + IM    1 249.34 1.8855  0.025 *
## + pH    1 250.04 1.1904  0.260
## + DO    1 250.33 0.9122  0.535
## + Alk   1 250.54 0.7069  0.820
## + Cond  1 250.57 0.6747  0.885
## + T     1 250.57 0.6751  0.945
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##

```

```

## Step: data_cca ~ Chl.a
##
##          Df      AIC      F Pr(>F)
## + TP     1 247.31 2.4923  0.005 **
## + IM     1 247.82 1.9958  0.005 **
## + SS     1 247.96 1.8591  0.010 **
## + Turb   1 248.23 1.5960  0.015 *
## + OM     1 248.57 1.2757  0.125
## + TN     1 248.69 1.1618  0.220
## + pH     1 248.96 0.8984  0.570
## + DO     1 249.01 0.8576  0.705
## + T      1 249.16 0.7173  0.865
## + Cond   1 249.16 0.7171  0.880
## + Alk    1 249.24 0.6383  0.905
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Step: data_cca ~ Chl.a + TP
##
##          Df      AIC      F Pr(>F)
## + OM     1 247.69 1.4913  0.030 *
## + TN     1 247.95 1.2471  0.135
## + Turb   1 248.03 1.1693  0.240
## + SS     1 248.15 1.0624  0.395
## + IM     1 248.28 0.9348  0.535
## + pH     1 248.31 0.9083  0.545
## + DO     1 248.34 0.8866  0.645
## + T      1 248.49 0.7454  0.890
## + Alk    1 248.59 0.6523  0.890
## + Cond   1 248.64 0.6067  0.955
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Step: data_cca ~ Chl.a + TP + OM
##
##          Df      AIC      F Pr(>F)
## + TN     1 248.10 1.4079  0.080 .
## + Turb   1 248.23 1.2906  0.145
## + SS     1 248.62 0.9453  0.490
## + IM     1 248.62 0.9454  0.540
## + T      1 248.62 0.9430  0.605
## + Cond   1 248.68 0.8907  0.610
## + Alk    1 248.69 0.8760  0.610
## + pH     1 248.64 0.9217  0.620
## + DO     1 248.64 0.9231  0.645
## ---

```

```
## Signif. codes: 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

5.5.1.3 Elección del modelo

Con la expresión en base a la fórmula, observamos que tenemos control del modelo pero nos enfrentamos al dilema de elegir el “mejor” de los modelos. Los modelos deben construirse con cuidado y, preferiblemente, utilizarse para probar hipótesis específicas. En la construcción automática de modelos, generalmente necesitamos dos modelos extremos: el mínimo y el modelo completo. Con esos dos modelos, podemos implementar la función **step()** para seleccionar el mejor modelo. Esta función utiliza el criterio de información de Akaike (AIC) en la elección del modelo. El AIC se basa en la bondad de ajuste (alta inercia restringida/ variabilidad explicada), pero es penalizado por el número de parámetros estimados (rango restringido/ criterio de parsimonia). Los modelos alternativos se ordenan por AIC. En cada caso, “+” indica el efecto de agregar un término, y “-” el efecto de eliminar un término, mientras que el modelo que se evalúa está marcado como “”. Se debe tener mucho cuidado porque realmente no hay AIC para ordenaciones canónicas (jaunque está calculada!), y siempre debemos inspeccionar la validez de elección del modelo.

```
final_cca <- step(cca.step.forward, scope=formula(cca_DGGE), test="perm")

## Start: AIC=247.69
## data_cca ~ Chl.a + TP + OM
##
##          Df      AIC      F Pr(>F)
## - OM     1 247.31 1.4913  0.030 *
## <none>   247.69
## - Chl.a  1 247.81 1.9596  0.010 **
## + TN    1 248.10 1.4079  0.070 .
## + Turb   1 248.23 1.2906  0.130
## - TP    1 248.57 2.6824  0.005 **
## + IM    1 248.62 0.9454  0.535
## + SS    1 248.62 0.9453  0.500
## + T     1 248.62 0.9430  0.600
## + DO    1 248.64 0.9231  0.545
## + pH    1 248.64 0.9217  0.590
## + Cond   1 248.68 0.8907  0.670
## + Alk   1 248.69 0.8760  0.605
## ---
## Signif. codes: 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Step: AIC=247.31
## data_cca ~ Chl.a + TP
```

```
##
##          Df      AIC      F Pr(>F)
## <none>    247.31
## + OM     1 247.69 1.4913  0.045 *
## - TP     1 247.92 2.4923  0.005 **
## + TN     1 247.95 1.2471  0.130
## + Turb   1 248.03 1.1693  0.285
## + SS     1 248.15 1.0624  0.305
## + IM     1 248.28 0.9348  0.550
## + pH     1 248.31 0.9083  0.610
## + DO     1 248.34 0.8866  0.580
## + T      1 248.49 0.7454  0.865
## + Alk    1 248.59 0.6523  0.920
## + Cond   1 248.64 0.6067  0.965
## - Chl.a  1 248.92 3.4926  0.005 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Criterio AIC: cuanto más bajo el valor, mejor. Sin embargo, modelos con diferencias de AIC < 2 son igualmente válidos. En este caso, ambos modelos son válidos.

```
#Colinealidad de las variables
vif.cca(cca_DGGE)
```

```
##      Turb      DO       T      Cond      pH      Chl.a
## 1.624592e+01 1.675050e+00 1.726303e+00 3.892686e+00 1.846872e+00 5.074104e+00
##      TN       TP      SS      OM       IM      Alk
## 2.052634e+00 5.642662e+00 4.859386e+08 5.514569e+07 3.413327e+08 3.391515e+00
```

```
vif.cca(cca.step.forward)
```

```
##      Chl.a      TP      OM
## 2.340084 1.390174 2.865323
```

```
vif.cca(final_cca)
```

```
##      Chl.a      TP
## 1.013204 1.013204
```

Notar que en el modelo completo Chl.a y TP presentan VIF > 5.

5.6 Bibliografía citada y de lectura recomendada

- Blanchet, F. G., P. Legendre, & D. Borcard, 2008. Forward selection of explanatory variables. *Ecology* 89: 2623–2632.
- Borcard, D., F. Gillet, & P. Legendre, 2018. Numerical Ecology with R. Dairy Science & Technology, CRC Taylor & Francis Group. Springer International Publishing, Cham, <http://link.springer.com/10.1007/978-3-319-71404-2>.
- Greenacre, M., 2008. Representación gráfica de distancias ji-cuadrado. La práctica del análisis de correspondencias 1–11.
- Legendre, P., & E. D. Gallagher, 2001. Ecologically meaningful transformations for ordination of species data. *Oecologia* 129: 271–280.
- Lepš, J., & P. Šmilauer, 2003. Multivariate Analysis of Ecological Data using CANOCO. . Mcardle, B. H., M. J. Anderson, S. Ecology, & N. Jan, 2014. Fitting Multivariate Models to Community Data: A Comment on Distance-Based Redundancy Analysis. *Ecological Society of America* 82: 290–297.
- Llames M.E., P. A. del Giorgio, H. Zagarese, M. Ferraro & I. Izaguirre, 2013. Alternative states drive the patterns in the bacterioplankton composition in shallow Pampean lakes (Argentina). *Environmental Microbiology Reports* 5(2): 310-321, <https://doi.org/10.1111/1758-2229.12020>.
- Oksanen, J., 2008. Vegan: an introduction to ordination. R- packace Vegan 1: 1–10, <http://doi.acm.org/10.1145/2037556.2037605%5Cnftp://ftp3.ie.freebsd.org/pub/cran.r-project.org/web/packages/vegan/vignettes/intro-vegan.pdf>.
- Oksanen, J., 2009. Design decisions and implementation details in vegan. R- packace Vegan 2: 1–11.
- Oksanen, J., 2012. Constrained ordination: tutorial with R and vegan. R- packace Vegan 1–10.
- Peres-Neto, P. R., P. Legendre, S. Dray, & D. Borcard, 2006. VARIATION PARTITIONING OF SPECIES DATA MATRICES: ESTIMATION AND COMPARISON OF FRACTIONS PEDRO. 87: 2614–2625.
- Quiroga, M. V., G. Mataloni, B. M. S. Wanderley, A. M. Amado & F. Unrein, 2017. Bacterioplankton morphotypes structure and cytometric fingerprint rely on environmental conditions in a sub-Antarctic peatland. *Hydrobiologia* 787: 255–268, <https://doi.org/10.1007/s10750-016-2969-2>.
- Zuur, A. F., E. N. Ieno, & G. M. Smith, 2007. Analysing Ecological Data. Springer New York, New York, NY, <http://link.springer.com/10.1007/978-0-387-45972-1>.

Chapter 6

Paquete *metrix*

Julieta Capeletti¹

Instituto Nacional de Limnología (INALI, UNL-CONICET), Laboratorio de Bentos

Juan Manuel Cabrera²

Instituto de Investigación y Desarrollo en Bioingeniería y Bioinformática (IBB, UNER-CONICET)

6.1 ¿Que es Metrix?

metrix es un paquete de R de código abierto diseñado específicamente para evaluar la calidad del agua utilizando datos de densidad de macroinvertebrados (?). Este paquete nos permite calcular una extensa cantidad de métricas e índices bióticos que evalúan la calidad del ambiente de manera rápida y sencilla. Admite una resolución taxonómica heterogénea que permite cálculos en varios niveles taxonómicos y es capaz de leer hojas de cálculo biológicas a partir de grandes volúmenes de datos. Este paquete facilita la manipulación de datos, simplificando el proceso mediante el uso de hojas de cálculo predefinidas en formato .csv. Esto reduce la necesidad de software o paquetes adicionales, haciéndolo accesible y fácil de usar, incluso para usuarios sin conocimientos de programación. Las futuras versiones del paquete pueden incluir otros índices métricos y bióticos, mejorando aún más sus capacidades. Además, se puede desarrollar una interfaz fácil de usar para mejorar la usabilidad de la herramienta para usuarios no científicos, como las agencias gubernamentales responsables de evaluar el impacto ambiental de las actividades relacionadas con los seres humanos.

¹julieta.capeletti@hotmail.com

²juan.cabrera@uner.edu.ar

6.2 Conjunto de datos de ejemplo

Antes de comenzar a probar *metrix*, vamos a descargar el conjunto de datos perteneciente a muestreos en la cuenca Cañada Carrizales localizada en la ecorregión Pampeana de Argentina ($32^{\circ}26'23.53\text{ ``S}$, $61^{\circ}18'10.69\text{ ``W}$), perteneciente al trabajo *Métricas basadas en macroinvertebrados como monitores de ambientes con uso de suelo agrícola: estudio preliminar en una cuenca pampeana* (?). El conjunto de datos está disponible en la carpeta *data* del repositorio de GitHub de *metrix* <https://github.com/Dvelop-R/metrix>.

6.3 Instalar *metrix*

metrix se encuentra disponible en el repositorio CRAN. Primeramente se debe descargar e instalar utilizando el código `install.packages("metrix")`. Seguidamente se debe cargar el paquete en el entorno de trabajo:

```
library(metrix)
```

6.4 ¿Que tipos de datos analiza *metrix*?

metrix utiliza un formato específico de tabla de datos. La misma consiste en 8 columnas que representen la clasificación científica y funcional de los taxones:

- Las primeras 7 columnas corresponden a *Clase*, *Orden*, *Familia*, *Subfamilia*, *Tribu*, *Género* y *Especie*.
- La columna 8 indica el *grupo funcional* (FG) de los taxones, que puede ser *recolectores filtradores* (GF), *colectores recolectores* (GC), *depredadores* (P), *raspadores* (SCR), o *trituradores* (SHR).
- Después de estas columnas, se debe incluir los *datos de densidad* (*individuos/m²*) de los macroinvertebrados para cada muestra de sitio.

Durante el proceso de carga de datos es muy común que se cometan errores de tipeo que pueden llevar a que no se pueda identificar de manera correcta a los taxa incluidos en el conjunto de datos. *metrix* posee un sistema de autocorrección que puede asisitir al usuario en la identificación de estas fallas en la carga de datos. Para ello utiliza una base de datos de nombres científicos utilizados en la descripción de los taxones que son tenidos en cuenta a la hora de calcular las métricas e índices bióticos calculados por el paquete. Este sistema de autocorrección informa al usuario sobre posibles errores de carga y, si el usuario lo indica, puede corregirlo de manera automática (con un criterio de similitud de palabra).

6.5. CARGAR DATOS DE DENSIDAD DE MACROINVERTEBRADOS USANDO METRIX179

6.5 Cargar datos de densidad de macroinvertebrados usando *metrix*

Para cargar los datos en el entorno de trabajo de R utilizamos la función `read_data()`. Esta función recibe 3 parámetros:

- *file_name*: dirección/nombre del archivo .csv que contiene los datos de densidad de macroinvertebrados.
- *correct*: valor lógico que indica si se va a utilizar el sistema de autocorrección de datos.
- *verbose*: valor lógico que indica si se debe mostrar mensajes descriptivos del proceso de carga.

```
#IMPORTANTE: revisar la dirección del archivo ecological_data.csv. En este ejemplo estaría dentro de los datos_cargados<-read_data(file_name = "ecological_data.csv", correct = TRUE, verbose = FALSE)
```

```
## The word ampullaridae was not found in internal Family word database. Please check that entry.  
## The autocorrect system replace the word ampullaridae for Ampullariidae.
```

En este ejemplo, los datos ecológicos utilizados contienen un error en la columna de *Familia*. El sistema de autocorrección notificará al usuario sobre este error y, si el parámetro `correct` está seteado con el valor `TRUE`, reemplazará la palabra presuntamente errónea por otra similar hallada en la base de datos interna.

6.5.1 ¿QUE PASA SI QUIERO CREAR MI PROPIO CONJUNTO DE DATOS?

metrix contiene una función que genera un archivo .CSV con una tabla cuyas columnas están debidamente etiquetadas para que pueda ser utilizada por el paquete. Para generar este archivo en el espacio de trabajo, se debe correr el siguiente código:

```
metrix_table_template()  
  
##      Class    Order       Family      Subfamily        Tribe        Genus Species  
## 1 Insecta Diptera Chironomidae Chironominae Tanytarsini Paratanytarsus  
##   FG Sample1 Sample2  
## 1 GC      1      0
```

Una vez generado el archivo se puede editar el mismo utilizando un bloc de notas o una editor de hojas de cálculo. Luego se puede cargar al entorno de trabajo de igual manera que en el paso anterior.

6.6 Cálculo de metricas individuales y ecológicas

Para evaluar la calidad biológica del agua, es necesario examinar valores específicos que midan características clave de la comunidad de macroinvertebrados, las cuales son indicativas de su respuesta a degradaciones ambientales. *metrix* incluye una amplia gama de cálculos tanto para métricas individuales como ecológicas: * *Métricas individuales*: abarcan la riqueza, que indica la diversidad del conjunto de macroinvertebrados; las métricas de composición, que calculan la abundancia relativa de taxones específicos dentro de la comunidad en términos de porcentajes; y las métricas de densidad, que sirven como medidas universales utilizadas en todo tipo de estudios biológicos. * *Métricas ecológicas*: incluyen métricas tróficas (como la riqueza de depredadores, la riqueza de colectores recolectores, la densidad de trituradores y la densidad de raspadores), que actúan como indicadores de procesos complejos como las interacciones tróficas, la producción y la disponibilidad de fuentes de alimento. Además, se incluyen medidas de tolerancia para indicar la sensibilidad de la comunidad y especies individuales a varios tipos de perturbaciones.

En este ejemplo vamos a calcular metricas de riqueza de nuestro conjunto de datos previamente cargados. Para ello vamos a utilizar la función `rich_metrics()`, la cual recibe 4 parámetros:

- *dataset*: conjunto de datos cargados con la función `read_data()`.
- *store*: valor lógico que indica si el resultado debe ser almacenado en un archivo.
- *dec_c*: carácter utilizado como separador de decimales.
- *verbose*: valor lógico que indica si se debe mostrar mensajes descriptivos del cálculo realizado.

```
#Calculation of richness measures
rich_metrics(dataset = datos_cargados, store = FALSE, dec_c = ".", verbose = TRUE)

## Checking table format for Richness measures calculation...

##          P1 P2 P3 P4 P5 P6 P7 P8 P9 P10
## n_taxa      4  7  7  7  6  6  4  6  8   4
## n_fam       4  3  6  5  6  6  4  6  7   3
## n_gen       4  6  4  5  5  6  4  6  6   4
## n_insec_fam 2  1  4  2  3  2  2  2  4   1
## n_non_insec_order 2  2  2  2  3  3  2  3  3   2
## n_dip_fam    1  1  3  2  2  1  1  1  3   1
## n_dip_gen    1  4  1  1  1  1  1  1  2   1
## n_dip_chir_gen 1  4  1  1  1  1  1  1  2   1
## n_chir_tax   1  4  1  1  1  1  1  1  2   1
## n_tany_tax   0  0  0  0  0  0  0  0  0   0
```

```

## n_stemp_tax      0  0  0  0  0  0  0  0  0  0
## n_non_chir_dip_tax 0  0  2  1  1  0  0  0  2  0
## n_mol_tax        1  1  1  2  1  2  1  2  1  1
## n_gastr_tax       1  1  1  2  1  2  1  2  1  1
## n_biv_tax         0  0  0  0  0  0  0  0  0  0
## n_crus_tax        0  0  0  0  1  1  1  1  1  0
## n_crusmol         1  1  1  2  2  3  2  3  2  1
## n_oligo_tax        1  2  2  3  1  1  0  1  1  2
## n_ephetrich        0  0  0  0  0  0  0  0  1  0

```

Los sitios P1, P7 y P10 son los que menor riqueza presentan, a diferencia de los demás sitios. En estos sitios la riqueza se da principalmente por la presencia de los órdenes de Diptera, Mollusca, Crustacea y Oligochaeta.

También podemos calcular métricas de tolerancia utilizando la función `tol_metrics()`. Esta función recibe 4 parámetros:

- `dataset`: conjunto de datos cargados con la función `read_data()`.
- `store`: valor lógico que indica si el resultado debe ser almacenado en un archivo.
- `dec_c`: carácter utilizado como separador de decimales.
- `verbose`: valor lógico que indica si se debe mostrar mensajes descriptivos del cálculo realizado.

```

#Calculo de metricas de tolerancia
tol_metrics(dataset = datos_cargados, store = FALSE, dec_c = ".", verbose = TRUE)

## Checking table format for Tolerance measures calculation...

##          P1      P2      P3      P4      P5      P6      P7      P8      P9      P10
## r_oligochir 0.0066 0.0054 0.0278 0.6667 0.0357 0.0085 0.0000 0.0313 0.100 3.3333
## r_oligoset  0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.000 0.0000
## r_tanychir  0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.000 0.0000
## den_t_lhoff  0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.000 0.0000
## den_t_bothr  0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.000 0.0000
## den_t_tubi   0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.000 0.0000
## den_t_dero   0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.000 0.0000
## den_t_prist  0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.000 0.0000
## den_t_chiro  0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.000 0.0000
## den_t_hele   0.5740 0.5740 0.5740 0.5740 0.5740 0.5740 0.574 0.5740 0.574 0.5740

```

La relación de la densidad de Oligochaeta/Chironomidae nos permitiría diferenciar los sitios de acuerdo a su gradiente de impacto. Esta métrica aumenta a medida que aumenta la degradación, por lo tanto los sitios con los valores

más altos serían los más contaminados a diferencia de los sitios con los valores más bajo los cuales presentarían condiciones de referencia. La métrica Heleobia/DT presenta el mismo valor en todos los sitios por lo tanto no nos estaría diferenciando los sitios de acuerdo a su gradiente de impacto antrópico.

6.7 Cálculo de indices bióticos

Los índices bióticos comprenden una combinación de dos o tres métricas individuales o ecológicas, que se condensan en un único valor que resume las características biológicas de todos los organismos presentes. Para los índices cualitativos, estas métricas incluyen la riqueza de taxones y la sensibilidad a la contaminación. Para los índices cuantitativos se incorpora la abundancia (absoluta o relativa). *metrix* incluye cálculos para los siguientes índices bióticos:

- *BMWP* (Biological Monitoring Working Party, (?)), *ASPT* (Average Score per Taxon, (?)) y sus adaptaciones: *BMWP'* (?) y *BMWP''* (?). Estos índices utilizan la presencia y abundancia de ciertas familias de macroinvertebrados. A cada familia se le asigna un valor de puntuación basado en su tolerancia a la contaminación (de 0 a 10). Las familias más sensibles a la contaminación reciben puntuaciones más altas, mientras que las familias más tolerantes reciben puntuaciones más bajas. Las puntuaciones de todas las familias de organismos presentes en una muestra de agua se suman para obtener el valor total del índice y determinar la calidad del agua.
- *IMRP* (Índice de Macroinvertebrados en Ríos Pampeanos, (?)). Este índice se basa en la presencia y abundancia de ciertos órdenes y familias de macroinvertebrados para determinar la calidad del agua. Al igual que los índices BMWP, a cada grupo se le asigna un valor de puntuación basado en su tolerancia a la contaminación (en este caso, de 0 a 1), siendo el valor final del índice la suma de todos los órdenes o familias presentes en la muestra.
- *ICBrio* (Índice das Comunidades Benticas em Ríos, (?)). Este índice utiliza los valores de densidad total de la muestra y se basa en la tolerancia de los organismos a la contaminación y su respuesta a las perturbaciones ambientales. El índice asigna valores de puntuación a diferentes grupos taxonómicos (a nivel de orden, familia y género) de macroinvertebrados presentes en las muestras. Estas puntuaciones se suman luego para obtener un valor total del índice y determinar la calidad del agua.

En este ejemplo vamos a calcular los índices *BMWP* e *ICBrio*. Para ello vamos a utilizar las funciones `bmwp_ind()` e `icbrio_ind()`. Ambas funciones reciben 4 parámetros:

- *dataset*: conjunto de datos cargados con la función `read_data()`.
- *store*: valor lógico que indica si el resultado debe ser almacenado en un archivo.
- *dec_c*: carácter utilizado como separador de decimales.
- *verbose*: valor lógico que indica si se debe mostrar mensajes descriptivos del cálculo realizado.

```
#Calculo de BMWP
bmwp_ind(dataset = datos_cargados, store = FALSE, dec_c = ".", verbose = TRUE)

## Checking table format for BMWP and ASPT index calculation...

## $Ibmwp_n
##          P1  P2  P3  P4  P5  P6  P7  P8  P9  P10
## ind_bmwp 9 3.0 9 3.0 9 12 8 12 3.0 3.0
## ind_aspt 3 1.5 3 1.5 3  3  4  3 1.5 1.5
##
## $Ibmwp_c
##          P1          P2          P3
## ind_bmwp_class Class VII Very Bad Class VII Very Bad Class VII Very Bad
## ind_aspt_class      Class VI Bad Class VII Very Bad      Class VI Bad
##          P4          P5          P6
## ind_bmwp_class Class VII Very Bad Class VII Very Bad Class VII Very Bad
## ind_aspt_class Class VII Very Bad      Class VI Bad      Class VI Bad
##          P7          P8          P9
## ind_bmwp_class Class VII Very Bad Class VII Very Bad Class VII Very Bad
## ind_aspt_class   Class V Moderate      Class VI Bad Class VII Very Bad
##          P10
## ind_bmwp_class Class VII Very Bad
## ind_aspt_class Class VII Very Bad
```

Los valores de BMWP y ASPT nos indican condiciones de mala calidad en los 10 puntos de muestreo.

```
#Calculo de ICBrio
icbrio_ind(dataset = datos_cargados, store = FALSE, dec_c = ".", verbose = TRUE)

## Checking table format for ICbrio index calculation...

## $Icbrio_n
##          P1  P2  P3  P4          P5  P6  P7          P8  P9  P10
## ind_icbrio 4.333333 4  4  4 3.666667 4  6 3.666667 2.75 4.333333
##
## $Icbrio_c
##          P1  P2  P3  P4  P5  P6  P7  P8          P9  P10
## ind_icbrio_class Bad Bad Bad Bad Bad Bad Bad Regular Bad
```

De igual manera el indice ICBRio nos indica que 9 sitios presentan una condición biológica mala, mientras que 1 sitio (P9) presenta regulares condiciones.

6.8 FUTURO DEL PAQUETE

Las futuras versiones del paquete podrían incluir otras métricas e índices bióticos, mejorando aún más sus capacidades. Además, puede que se desarrolle una interfaz amigable para el usuario, mejorando la usabilidad de la herramienta para usuarios no científicos, como agencias gubernamentales encargadas de evaluar el impacto ambiental de actividades relacionadas con el ser humano.

6.9 AGRADECIMIENTOS

Agradecemos a la Dra. Florencia Zilli, Dra. Mercedes Marchese y Dr. Joaquin Cochero por su ayuda y apoyo en la creación de esta herramienta.

Chapter 7

Paquetes *DiaThor* y *optimos.prime*

Joaquín Cochero¹

7.1 Paquete *DiaThor*

Autores: María Mercedes Nicolosi Gelis², María Belén Sathicq³, Joaquín Cochero

7.1.1 ¿Qué hace?

El paquete calcula múltiples **índices bióticos y ecológicos basados en diatomeas** a partir de datos de abundancia en muestras ambientales.

7.1.2 ¿Cómo funciona?

El paquete obtiene las **preferencias ecológicas** de los taxones de diatomeas vinculandolos por nombre a las bases de datos de cada índice biótico y ecológico. Realiza una búsqueda exacta y luego heurística del nombre del taxón comparándola contra distintas Fuentes de información ecológica de las especies.

Una vez que encuentra la especie en cada alguna de las bases de datos de los distintos índices, realiza los cálculos de los índices en base a su abundancia en

¹jcochero@ilpla.edu.ar

²mercedesnicolosi@ilpla.edu.ar

³mbelen@ilpla.edu.ar

las muestras, y devuelve una tabla con los valores de cada índice por muestra ingresada.

7.1.3 ¡Manos a la obra!

Vamos a utilizar el set de datos del trabajo “*Exploring the use of nuclear alterations, motility and ecological guilds in epipelagic diatoms as biomonitoring tools for water quality improvement in urban impacted lowland streams*”, de Nicolosi Gelis et al. (2020) (?) publicado en Ecological Indicators y disponible en el Repositorio Institucional CONICET Digital.

7.1.3.1 Descargar los datos

Descargar los datos de ejemplo desde el repositorio del curso y guardar el archivo en el Directorio de Trabajo del Proyecto que creamos para esta Unidad (ver cómo hacerlo en la Unidad ??).

Datos de ejemplo aquí: https://github.com/Limno-con-R/CILCAL2023/blob/main/datasets/sampleData_diaThor.csv

7.1.3.2 Instalar el paquete

Instalaremos el paquete `diaThor`(?) como se indica en la Unidad ?? y los cargaremos a nuestro espacio de trabajo actual:

```
library(diaThor)
```

7.1.3.3 Cargar y previsualizar nuestro archivo de datos

Cargaremos nuestro archivo de ejemplo (`sampleData_diaThor.csv`) a un dataframe llamado “base”:

```
base <- read.csv("sampleData_diaThor.csv") #Corroborar que el nombre del archivo coincide
```

Veamos brevemente cómo está organizada la información de nuestras especies:

```
str(base)
```

`base` es un objeto `data.frame` con 164 obs. o filas (especies) y 110 variables o columnas (muestras).

Nuestra planilla de datos además contiene una **columna obligatoria** llamada “*species*”, la primera, adonde debremos ingresar los nombres de las especies lo mejor que podamos, sin datos sobre autorías de las mismas.

Ejemplo de cómo llamar a las especies:

Hippodonta capitata —————> (ASI SI!)

Hippodonta capitata (Ehrenb.) Lange-Bert., Metzeltin and Witkowski 1996 —————> (ASI NO!)

Veamos las primeras filas y las primeras columnas:

```
head(base[1:10, 1:5])
```

	X	species	S1_LI1_DAY0	S1_LI2_DAY0	S1_LI3_DAY0
## 1	1	Achnanthidium minutissimum	0	0	0
## 2	2	Achnanthes exigua var. exigua	0	0	0
## 3	3	Amphora libyca	0	0	0
## 4	4	Anomoeoneis sphaerophora	0	0	0
## 5	5	Craticula ambigua	0	0	1
## 6	6	Caloneis bacillum	0	0	0

Cada columna representa **a una muestra**.

Debe contener un nombre de muestra (*por ejemplo, se llaman S1_LI1_DAY0, S1_LI2_DAY0, etc.*).

NOTA: Los valores en las celdas pueden ser valores de densidad absoluta (como es en este ejemplo) o abundancia relativa. Por defecto, *diaThor* asume que las densidades son absolutas; si queremos cambiar esta opción, debemos pasar el parámetro `isRelAb=FALSE` a `isRelAb=TRUE` cuando usamos las funciones.

7.1.3.4 ¡A la magia!

Vamos a ejecutar la función `diaThorAll()`, que se encargará de realizar **todos los análisis que puede realizar el paquete**, una atrás de otra. Y dispondremos de los resultados en un único objeto de tipo `data.frame`, que aquí lo llamaremos **resultados**.

Más adelante veremos que podemos solicitar cosas específicas si sólo necesitamos un tipo de resultado, para ahorrar tiempo y reducir la cantidad de salidas.

```
resultados <- diaThorAll(base)
```

El paquete nos irá mostrando información en la consola de RStudio, y nos solicitará que le indiquemos una carpeta adonde guardar todas las posibles salidas

que vaya generando. Exploraremos estas salidas luego para tratar de mejorar los resultados.

```
[1] "Select Results folder"
```

Seleccionemos un directorio (preferentemente vacío!) para guardar las salidas, y el paquete continuará reconociendo las especies, y buscando información ecológica sobre ellas para calcular los distintos índices bióticos.

Aclaración: Si estamos trabajando en sistema operativo Linux, es posible que este cuadro de dialogo no aparezca en la pantalla y obtengamos un error en la consola. Para evitar esto, podemos manualmente indicar el directorio adonde se guardaran los resultaos como un parámetro de la función `diaThorAll`, de la siguiente manera:

```
resultados <- diaThorAll(base, resultsPath="/Resultados") #Por ejemplo, aquí debemos hacerlo
```

7.1.3.5 Los resultados

Si todo ha salido bien, `diaThor` debería haber calculado todos los índices posibles.

Cuando los índices bióticos se publican, no necesariamente se le asigna un valor ecológico a todas las especies de diatomeas (*hay unas 200.000 especies!*). Hay índices que sólo contemplan menos de 300 especies (ejemplos: *IDP*, *SPEAR* o los de Lobo et al.), y otros incluyen más de 6000 (ejemplo: *IPS*).

En la consola veremos que porcentaje del listado de especies que ingresamos fue efectivamente utilizado para calcular cada índice. Por ejemplo -> “*Taxa recognized to be used in DISP index: 22.6 %*”, nos indica que de nuestras 164 especies, sólo utilizó 22.6% para calcular el índice DISP, mientras que dejó afuera el 77.4%. Es importante entonces delimitar qué índices fueron calculados sobre una base sólida de nuestros datos al momento de interpretar los resultados.

El final de la secuencia de cálculo en la consola está delimitado por la exportación de gráficos ([1] “*Plots exported!*”).

En el objeto `resultados` veremos una tabla completa con todos los resultados obtenidos.

```
str(resultados)
```

Es un objeto `data.table` con 109 `obs.` (nuestras muestras, filas) y 124 `vars.` (los índices calculados para cada muestra).

Veamos los resultados completos!

`View(resultados)`

	Richness	Shannon.H	Evenness	chloroplasts...1	chloroplasts...2	chlorop
X	164	4.9097182	0.9627151	35.23	53.94	
S1_LI1_DAY0	32	1.6000228	0.4616690	14.39	82.48	
S1_LI2_DAY0	17	1.1219849	0.3960114	47.87	51.38	
S1_LI3_DAY0	34	1.9917677	0.5648225	17.30	82.69	
S1_X1_DAY0	22	1.4950968	0.4836869	56.57	43.21	
S1_X2_DAY0	14	0.8618036	0.3265574	21.91	77.85	
S1_X3_DAY0	25	1.3347881	0.4146752	13.87	85.33	
S1_HI1_DAY0	30	1.1668271	0.3430636	6.50	92.22	
S1_HI2_DAY0	22	1.4267034	0.4615606	51.44	48.28	
S1_HI3_DAY0	36	1.6623785	0.4638956	8.49	91.21	
S1_LI1_DAY15	47	3.0250414	0.7856949	40.12	48.86	
S1_LI2_DAY15	40	2.1617052	0.5860059	17.87	77.72	

Figure 7.1: Figura 7.1. Primeras filas y columnas de la tabla de resultados de DiaThor

Veremos que cada muestra (fila) tiene sus valores de diversidad (*Richness*, *Shannon.H*, *Evenness*), porcentajes de número y forma de cloroplastos (*chloroplasts*, *shape.chloroplasts*), porcentaje de diatomeas de cada clase de tamaño (*size.class*), porcentaje de diatomeas de cada gremio ecológico (*guilds*), los valores de cada índice de Van Dam (*VD.Salinity*, *VD.NHET*, *VD.Oxygen*, *VD.Saprobity*, *VD.Aero*, *VD.Trophic*) y luego todos los índices bióticos.

Luego de cada uno de los índices, se encuentra una columna que indica cuántas especies **no fueron** determinadas en cada muestra (terminan en “*.Indet*”) y cuantas fueron usadas (terminan en “*.Taxa.Used*”).

7.1.3.6 La sintonía fina: veamos las salidas!

Una vez que el paquete termine el análisis, vaya a la carpeta adonde se guardaron los resultados. Se encontrará con varios archivos:

- ***Diato_results - Results.csv***: Este archivo contiene la tabla principal de resultados, con cada muestra (fila) y sus valores de los diversos índices que si se pudieron calcular (columnas).
- ***Plots.pdf***: Se incluyen los gráficos de cada uno de los índices bióticos calculados (eje X) en cada uno de las muestras analizadas (eje Y) para poder visualizarlo. Son realizados con los mismos datos que se encuentran en la tabla de resultados.

- **Taxa excluded** y **Taxa included**: Es MUY importante que se revisen estos dos archivos. En estos archivos veremos que taxa de nuestra lista de especies original fueron **incluidos** o **excluidos** en el cálculo de cada uno de los índices. También nos ayudará a reconocer potenciales errores de tipeo o de nombramiento de especies.

VanDam Taxa used.txt: En particular, los valores ecológicos de Van Dam son separados en un archivo de texto al resto de los resultados, porque suelen ser muy amplios. Aquí se indica en cada una de nuestras muestras los taxones que fueron incluidos en el cálculo de cada una de las tolerancias (*salinidad, N-heterotrofía, requisitos de oxígeno, saprobiedad, humedad, estado trófico*).

Es **muy importante** que antes de reportar los resultados del análisis, se revisen cuidadosamente las salidas del paquete. Nos pueden indicar que hay especies con el nombre mal escrito, o que para el cálculo de algún índice en particular se utilizó poca cantidad de especies presentes, cuestionando así la precisión del mismo.

7.1.3.7 Algunas aclaraciones relevantes:

- Sobre el reconocimiento de los nombres de las especies:
 - DiaThor realiza **dos búsquedas** automáticamente de cada especie en nuestro archivo.
 - * Primero realiza una **búsqueda exacta** de cada especie comparándola contra todas las especies utilizadas en todos los índices bióticos y ecológicos.
 - * Aquellas especies que no son encontradas por este medio, las separa y las vuelve a buscar automáticamente por una **búsqueda heurística**, para tratar de contemplar aquellos errores en la escritura de los nombres (Quien no escribió incorrectamente ‘*Nitzschia fasciculata*’ alguna vez, que tire la primer piedra!).

Este tipo de búsqueda permite que haya un número determinado de caracteres distintos entre dos palabras para considerarlas iguales. Por defecto, este valor en las funciones de *diaThor* es igual a 2. Por ejemplo, la búsqueda heurística encontraría a la especie ‘*Nitzschia fasciculata*’ aunque la hubieramos escrito como ‘*Nitzschia fasiculata*’, porque hay sólo 1 carácter distinto. Este valor de distancia se puede cambiar con el parámetro `maxDistTaxa`.

¡Cuidado! Mientras más alto el valor de este parámetro, más chances de encontrar la especie que buscamos, pero más posibilidad de que confunda nuestra especie con otra de nombre similar, y se incrementa notablemente el tiempo que tarda en buscarlas.

- Las especies que tienen **categorías subespecíficas** (variedades y/o formas) se deben indicar con “var.” y “fo.”.

* Ejemplo: *Achnanthes exigua* var. *exigua*” o “*Gomphonema parvulum* var. *parvulum* fo. *parvulum*”. El texto no busca formatos, no es necesario poner las especies en itálicas o subrayadas.

El paquete va a tratar de buscarlas de cualquier manera aunque no contengan estos acrónimos, pero las búsquedas son más precisas y rápidas cuando estan nombradas así.

7.1.3.8 Preguntas frecuentes

- ¿Hay problemas encontrando el archivo de entrada CSV?

La consola nos indica este error:

```
*cannot open file 'nuestro_archivo.csv': No such file or
directory
```

Si corremos la función `diaThorAll()` directamente, sin agregar argumentos o parámetros, **un cuadro de diálogo** nos pedirá que seleccionemos el archivo con nuestros datos.

```
resultados <- diaThorAll() #sin argumentos
```

- ¿Nuestros datos están en abundancia relativa?

Hay que indicarselo, cambiando el argumento `isRelAb` a `TRUE`.

```
resultados <- diaThorAll(isRelAb=TRUE)
```

- ¿No encuentra las especies, y quiero hacer una búsqueda heurística más amplia?

Cambiemos el argumento `maxDistTaxa`.

```
resultados <- diaThorAll(maxDistTaxa=4) #Ahora buscará nombres de taxa en las bases de datos internas
```

- ¿Y si quiero sólo calcular un índice, en vez de todo? Así es más rápido.

Podemos usar funciones para cada índice biótico o ecológico.

```
#Primero, cargamos nuestros datos, como siempre
```

```
base <- read.csv("sampleData_diaThor.csv") #Corroborar que el nombre del archivo coincida con el nombre de la función
```

```
#Luego, necesitamos que diaThor reconozca las especies por su nombre. Para esto existe la función
#A esta función se le pueden pasar los mismos argumentos que a diaThorAll(). Es decir, podemos reutilizarlos
```

```

resultados <- diat_loadData(base, isRelAb=TRUE, maxDistTaxa = 2) #Ejemplo usando abund
#Deberemos indicar una carpeta para los resultados, en el cuadro de diálogo

#Y luego, por ejemplo, podemos calcular sólo el índice IDP

resultadosIDP <- diat_idp(resultados) #Grabamos los resultados en el objeto resultados

#o el índice IPS

resultadosIPS <- diat_ips(resultados) #Grabamos los resultados en el objeto resultados

#Hay funciones separadas para todos los índices del paquete; en la documentación se in

```

7.1.3.9 Shiny!

Hay una versión online para utilizar el paquete `diaThor` a través de un entorno Shiny, en:

<https://limnolab.shinyapps.io/diathor-shiny/>

7.1.4 Fuentes de información ecológica de las especies

La información morfológica se busca en la base de datos del proyecto ‘Diat.Barcode’:

- Rimet F., Gusev E., Kahlert M., Kelly M., Kulikovskiy M., Maltsev Y., Mann D., Pfannkuchen M., Trobajo R., Vasselon V., Zimmermann J., Bouchez A., 2019. Diat.barcode, an open-access curated barcode library for diatoms. *Scientific Reports*. <https://www.nature.com/articles/s41598-019-51500-6>

La clasificación de las especies por su clase de tamaño se obtiene de:

- Rimet F. & Bouchez A., 2012. Life-forms, cell-sizes and ecological guilds of diatoms in European rivers. *Knowledge and management of aquatic ecosystems*, 406: 1-14. <https://www.kmae-journal.org/kmae/abs/2012/03/kmae120025/kmae120025.html>

La clasificación de las especies en gremios ecológicos se obtiene de:

- Rimet F. & Bouchez A., 2012. Life-forms, cell-sizes and ecological guilds of diatoms in European rivers. *Knowledge and management of aquatic ecosystems*, 406: 1-14. <https://www.kmae-journal.org/kmae/abs/2012/03/kmae120025/kmae120025.html>

La clasificación combinada de gremios y clases de tamaño (más nueva) se obtiene de:

- B-Béres, V., Török, P., Kókai, Z., Lukács, Á., Enikő, T., Tóthmérész, B., & Bácsi, I. (2017). Ecological background of diatom functional groups: Comparability of classification systems. *Ecological Indicators*, 82, 183-188. <https://www.sciencedirect.com/science/article/abs/pii/S1470160X1730420X>

Preferencias ecológicas para la clasificación de Van Dam se obtienen de:

- Van Dam, H., Mertens, A., & Sinkeldam, J. (1994). A coded checklist and ecological indicator values of freshwater diatoms from the Netherlands. *Netherland Journal of Aquatic Ecology*, 28(1), 117-133.

Los índices de diversidad (Shannon H') se calcula usando internamente el paquete *vegan* (?):

- Shannon, C. E., and Weaver, W. (1949). 'The Mathematical Theory of Communication.' (University of Illinios Press: Urbana, IL, USA.)

Los valores de tolerancia y valencia de cada especie para cada índice biótico se obtiene de sus fuentes publicadas originales:

- **IPS:** Coste, M. (1982). Étude des méthodes biologiques d'appréciation quantitative de la qualité des eaux. Rapport Cemagref QE Lyon-AF Bassin Rhône Méditerranée Corse.
- **TDI:** Kelly, M. G., & Whitton, B. A. (1995). The trophic diatom index: a new index for monitoring eutrophication in rivers. *Journal of Applied Phycology*, 7(4), 433-444.
- **IDP:** Gómez, N., & Licursi, M. (2001). The Pampean Diatom Index (IDP) for assessment of rivers and streams in Argentina. *Aquatic Ecology*, 35(2), 173-181.
- **DES:** Descy, J. P. 1979. A new approach to water quality estimation using diatom. *Beih. Nov Hedw.* 64:305-323
- **EPID:** Dell'Uomo, A. (1996). Assessment of water quality of an Apennine river as a pilot study for diatom-based monitoring of Italian watercourses. Use of algae for monitoring rivers, 65-72.
- **IDAP:** Prygiel, J., & Coste, M. (1993). The assessment of water quality in the Artois-Picardie water basin (France) by the use of diatom indices. *Hydrobiologia*, 269(1), 343-349.

- **ID-CH:** Hürlimann J., Niederhauser P. 2007: Méthodes d'analyse et d'appréciation des cours d'eau. Diatomées Niveau R (région). État de l'environnement n° 0740. Office fédéral de l'environnement, Berne. 132 p
 - **ILM:** Leclercq, L., & Maquet, B. (1987). Deux nouveaux indices diatomique et de qualité chimique des eaux courantes. Comparaison avec différents indices existants. Cahier de Biology Marine, 28, 303-310.
 - **LOBO:**
 - Lobo, E. A., Callegaro, V. L. M., & Bender, E. P. (2002). Utilização de algas diatomáceas epilíticas como indicadoras da qualidade da água em rios e arroios da Região Hidrográfica do Guaíba, RS, Brasil. Edunisc.
 - Lobo, E. A., Bes, D., Tudesque, L., & Ector, L. (2004). Water quality assessment of the Pardinho River, RS, Brazil, using epilithic diatom assemblages and faecal coliforms as biological indicators. Vie et Milieu, 54(2-3), 115-126.
 - **SLA:** Sládeček, V. (1986). Diatoms as indicators of organic pollution. Acta hydrochimica et hydrobiologica, 14(5), 555-566.
 - **SPEAR(herbicides):** Wood, R. J., Mitrovic, S. M., Lim, R. P., Warne, M. S. J., Dunlop, J., & Kefford, B. J. (2019). Benthic diatoms as indicators of herbicide toxicity in rivers—A new SPEcies At Risk (SPEARherbicides) index. Ecological Indicators, 99, 203-213.
 - **PBIDW:** Castro-Roa, D., & Pinilla-Agudelo, G. (2014). Periphytic diatom index for assessing the ecological quality of the Colombian Andean urban wetlands of Bogotá. Limnetica, 33(2), 297-312.
 - **DISP:** Stenger-Kovács, C., Körmendi, K., Lengyel, E., Abonyi, A., Hajnal, É., Szabó, B., Buczkó, K. & Padisák, J. (2018). Expanding the trait-based concept of benthic diatoms: Development of trait-and species-based indices for conductivity as the master variable of ecological status in continental saline lakes. Ecological Indicators, 95, 63-74.
-

7.2 Paquete *optimos.prime*

Autores: Belén Sathicq, María Mercedes Nicolosi Gelis, Joaquín Cochero

7.2.1 ¿Qué hace?

El paquete calcula los valores **óptimos y rangos de tolerancia ecológicos** de los taxones a las variables ambientales que proveamos.

7.2.2 ¿Cómo lo hace?

El enfoque más común utilizado para el cálculo de los óptimos y los rangos de tolerancia ecológicos es calcular la **media ponderada** (Birks et al. 1990; Potapova y Charles, 2003).

La **media ponderada** u_k de las estimaciones de los óptimos de las especies se calcula como:

$$u_k = \frac{\sum_{i=1}^n y_{ik}x_i}{\sum_{i=1}^n y_{ik}}$$

y la **tolerancia**, o desviación estándar ponderada t_k , se calcula como:

$$t_k = \sqrt{\frac{\sum_{i=1}^n y_{ik}(x_i - u_k)^2}{\sum_{i=1}^n y_{ik}}}$$

adonde y_{ik} es la abundancia relativa de la especie k en la muestra i ; x_i es el valor \log_{10} del parámetro ambiental en la muestra i ; y n es el número total de muestras del conjunto de datos.

7.2.3 ¡Manos a la obra!

Como ejemplo, vamos a utilizar parte del set de datos de la tesis doctoral “*Empleo de descriptores fitoplanctónicos como biomonitoros en la evaluacion de la calidad del agua en la costa del río de la Plata (Franja Costera Sur)*” de M.B. Sathicq (2017) (?), pública en el repositorio SEDICI de la UNLP.

7.2.3.1 Descargar los datos

Descargaremos dos sets de datos desde el repositorio del curso, y los debemos guardar en el Directorio de Trabajo del Proyecto que creamos para esta Unidad (ver cómo hacerlo en la Unidad ??).

Descargar el set de datos ambiental desde aquí: https://github.com/Limno-con-R/CILCAL2023/blob/main/datasets/environmental_data_OptimosPrime.csv

Descargar el set de dato de especies por muestras desde aquí: https://github.com/Limno-con-R/CILCAL2023/blob/main/datasets/species_data_OptimosPrime.csv

7.2.3.2 Instalar el paquete

Instalaremos el paquete *optimos.prime* (?) como se indica en la Unidad ?? y los cargaremos a nuestro espacio de trabajo actual:

```
library(optimos.prime)
```

7.2.3.3 Cargar y previsualizar nuestro archivo de datos

Debemos cargar dos sets de datos. El primero contiene la información de las variables ambientales en cada sitio o muestra (“*environmental_data_OptimosPrime*”). El segundo contiene la información de abundancia relativa de los taxa en cada sitio o muestra (“*species_data_OptimosPrime*”).

```
env_data <- read.csv("environmental_data_OptimosPrime.csv") #Corroborar que el nombre de este archivo es correcto

species_data <- read.csv("species_data_OptimosPrime.csv") #Corroborar que el nombre de este archivo es correcto

#Alternativamente, para seleccionar un archivo con un cuadro de diálogo, se puede usar
#file.choose()

env_data <- read.csv(file.choose()) #y seleccionar el CSV con los datos ambientales
species_data <- read.csv(file.choose()) #y seleccionar el CSV con los datos de abundancia

#Cualquiera de estas dos opciones cargará nuestros datos a los objetos "env_data" y "species_data"
```

Veamos brevemente cómo está organizada la información de nuestras especies:

```
str(env_data)
```

env_data es un objeto `data.frame` con 11 obs. o filas (*parámetros ambientales*) y 51 variables o columnas (*sitios muestreados, o muestras*).

```
str(species_data)
```

species_data es un objeto `data.frame` con 57 obs. o filas (*especies*) y 51 variables o columnas (*sitios muestreados, o muestras*).

¡Es importante que la cantidad de variables o columnas (*sitios muestreados, o muestras*) sea igual en ambos sets de datos!

7.2.3.4 ¡A la magia!

Vamos a ejecutar la función `op_calculate()`, que se encargará de realizar los análisis de óptimos ecológicos y rangos de tolerancia de los taxones que ingresamos en `species_data` a las variables ambientales que ingresamos en `env_data`.

. Y dispondremos de los resultados en un único objeto de tipo `data.frame`, que aquí lo llamaremos `resultados`.

```
resultados <- op_calculate(env_data, species_data)
```

Cuando terminen los cálculos, en la consola nos indicará que ya podemos ver los resultados.

```
[1] "Optimum values and tolerance range calculated and placed in  
the final data frame"  
[1] "Use View() to view data frame with results"  
  
View(resultados)
```

7.2.3.5 Los resultados

Si todo ha salido bien, debemos ver que el objeto `resultados` es una nueva matriz de 57 obs. o filas (*especies*) y ahora 34 variables o columnas (*parámetros ambientales*). Es decir, por cada uno de los parámetros ambientales que ingresamos (eran 11), vamos a tener una columna con su valor de óptimo ecológico y otras dos columnas con su rango ecológico máximo y su ecológico rango mínimo (33 columnas). La primera columna contendrá los nombres de las especies.

Veamos por ejemplo las columnas 2, 3 y 4.

La **columna 2** (“*Conductivity uS/cm*”) indica el valor óptimo de conductividad eléctrica para las especies, y el nombre con las unidades fueron ingresadas por el usuario en la matriz de datos ambientales.

La **columna 3** (“*Conductivity uS/cm - HIGH*”) indica el rango de tolerancia máximo, y la **columna 4** (“*Conductivity uS/cm - LOW*”), el rango de tolerancia mínimo.

Es importante entender que los valores obtenidos responden a los óptimos, máximos y mínimos de los taxones en base a los valores de los parámetros ambientales ingresamos. Si luego incorporamos nuevas muestras (por ejemplo en otros ambientes) y encontramos las mismas especies en otras condiciones ambientales, los óptimos y rangos de tolerancia de esas especies van a cambiar. Es por eso que se les llama *rangos ecológicos*.

7.2.3.6 Graficar e interpretar nuestros resultados

El paquete `optimos.prime` nos puede generar gráficos de tipo caterpillar con nuestros resultados para cada variable ambiental, con la función `op_plot()`. A esta función le tenemos que indicar la matriz de resultados de la función `op_calculate()`, y le podemos indicar los argumentos `label`, para personalizar

la etiqueta del gráfico, y/o el argumento `html`. Éste último indica si el gráfico producido se realizará en formato HTML, que permite interactuar con los datos usando el mouse, y es el valor predeterminado. Si `html=FALSE`, nuestro gráfico será una imagen plana.



Figure 7.2: Figura 7.2. Óptimos ecológicos y rangos de variación para las especies de algas en referencia al pH.

En el gráfico, se verán las especies (*en el eje Y*) y la variable ambiental elegida (pH, *en el eje X*). Los puntos verdes indican el **valor óptimo de cada especie**, y los bigotes indican los **rangos de tolerancia** de cada especie a esa variable.

Aquellas especies con rangos de tolerancia más bajos (ejs. *Actinastrum fluviale*, o *Euglena acus*), son consideradas especies ‘sensibles’ a esa variable ambiental, y aquellas con grandes rangos de tolerancia (ejs. *S. nanus* o *Tetrastrum glabrum*) son consideradas especies ‘tolerantes’. A su vez en este ejemplo, aquellas con valores óptimos más altos (más arriba en el gráfico) son consideradas mas alcaló-

filas y las que se encuentran más abajo en el gráfico son especies más acidófilas. Es importante revisar aquellas especies que tienen rangos de tolerancia muy bajos (ej. *A. distans*), ya que puede ser el resultado de una baja cantidad de presencias de esa especie en las muestras.

```
op_plot(resultados)
```

#En la consola nos preguntará que variable queremos graficar, y deberemos ingresar el número de

7.2.3.7 Algunas aclaraciones relevantes:

- ¿Nuestros datos no se encuentran en abundancia relativa?
Hay que indicarselo, cambiando el argumento `isRelAb` a `FALSE`.

```
resultados <- op_calculate(isRelAb=FALSE)
```

- ¿Nuestros datos ya fueron convertidos a `log10`?
Hay que indicarselo, cambiando el argumento `islog10` a `TRUE`.

```
resultados <- op_calculate(islog10=TRUE)
```

7.2.3.8 Shiny!

Hay una versión online para utilizar el paquete `optimos.prime` a través de un entorno Shiny, en:

<https://limnolab.shinyapps.io/OptimosPrime-Shiny/>

Chapter 8

Buenas prácticas en R y Rstudio

Joaquín Cochero¹

8.1 Proyectos de RStudio

8.1.1 ¿Qué son?

Los proyectos de RStudio son un sistema para organizar nuestro trabajo en RStudio. Una de las ventajas principales de los proyectos es que organizan lo que vamos a utilizar dentro de un único directorio de trabajo.

Al mantener todos los archivos fuente de un proyecto en el mismo directorio, podemos utilizar rutas relativas cuando sea necesario para acceder a ellos. Por ejemplo, puede utilizar:

```
my_data <- read.csv("data/example_data.csv")
```

8.1.2 ¿Cuándo usarlos?

Casi siempre. Se sugiere hacer un proyecto nuevo para cada actividad que se va a realizar. Por ejemplo, un proyecto que contenga todos los datos y scripts que van a ser utilizados en un mismo artículo o informe técnico.

Es más fácil decir ejemplos de cuando NO usar un proyecto de RStudio:

¹jcochero@ilpla.edu.ar

- Cuando quieran hacer una figura a último minuto.
- Necesitan una vista rápida de los datos antes de hacer el proyecto .
- Necesitan transformar o modificar datos para otra persona .
- Si pueden hacer todo lo que necesitan en unas 100 líneas de código o menos.

8.1.3 Estructura de directorios

A menos que sea un análisis corto que pueda hacerse en unas pocas líneas de código (incluyendo cargar archivos, limpieza, transformación, visualización, etc.), conviene crear *scripts* separados.

No hay una metodología estandarizada para la organización de los directorios en los proyectos, aunque se sugiere una estructura sencilla, por ejemplo:

- -> **mi-proyecto** (*carpeta principal del proyecto, es el directorio de trabajo*)
 - -> **data** (*se guardarán los archivos de datos, CSV, XLSX, etc.*)
 - -> **results** (*para ir exportando los resultados*)
 - -> **scripts** (*cada script es una etapa en nuestro trabajo*)
 - * *script_principal.R*
 - * *01_funciones.R*
 - * *02_limpieza_datos.R*
 - * *03_analisis.R*
 - * *04_figuras.R*

8.1.4 Nombres de archivos

Los nombres de los archivos deben ser explicativos del contenido. En el caso de los *scripts*, se sugiere numerarlos para que otros usuarios del proyecto sepan el orden de las instrucciones.

Buenas prácticas:

“01_carga_datos.R”
 “02_limpieza_datos.R”
 “03_analisis_descriptivos.R”

Malas prácticas:

“Mi archivo.R”
 “untitled.R”
 “script.R”

8.1.5 Otras consideraciones en RStudio

No guarde un historial de la sesión (la opción por defecto en R, cuando le pregunta si desea un archivo RData). En su lugar, comience en un entorno limpio para que los objetos antiguos no permanezcan en su entorno más tiempo del necesario.

Colabora. Practica revisión de código con un colega. Nuestro código también es un logro científico y el producto de mucho trabajo. Utiliza un sistema de control de versiones, como las de GitHub.

8.2 Buenas prácticas dentro de los scripts

8.2.1 Lleva un registro de quién escribió el código y de su finalidad

Empieza tu código con una descripción anotada de lo que hace el código, para cuando tengas que mirarlo o cambiarlo en el futuro.

```
# Este es el código para replicar los análisis y las cifras del paper titulado "La corchea y el  
# Código desarrollado por Juan Martínez.
```

8.2.2 Identifique y segregue distintos componentes en su código

Es fácil anotar y marcar tu código usando # o #- para separar secciones de tu código y facilitar la búsqueda de partes específicas de tu código. Por ejemplo, al escribir código suele ser útil separar las definiciones de las funciones.

Rstudio le ofrece la posibilidad de crear una sección pulsando (*Ctrl + Shift + R*), o puede crear una añadiendo 4 guiones (-) después de un comentario o 4 símbolos de numeral (#) después de un comentario.

```
# algún comentario ----  
# algún comentario #####
```

Esto hace que el código tenga un aspecto prolífico y que sea fácil de mantener a largo plazo.

También puede crear subsecciones en R añadiendo símbolos numeral delante de una sección.

```
## algún comentario ----

### algún comentario ----

#### algún comentario ----
```

8.2.3 Orden del código

Para ordenar las secciones en tu código se sugiere:

1. Incuir todas las librerías en la parte superior del código.
2. Establecer todas las variables por defecto u opciones globales y todas las variables de rutas en la parte superior del código.
3. Fuentes del código y créditos, en la parte superior.
4. Llamar a todos los archivos de datos en la parte superior (CSV, XLS, etc.).

Si creas sólo una o unas pocas funciones personalizadas en tu script, colocalas hacia la parte superior de tu código. Si has escrito muchas funciones, es mejor separarlas en su propio archivo .R y luego crear un script principal que las llame.

También, se sugiere:

- Dividir el código en trozos del tamaño de un bocado. Si una función o un bucle resulta demasiado largo, busca formas de dividirlo en trozos más pequeños.
- Utilizar un estilo coherente en su código. Por ejemplo, nombra todas las matrices con algo que termine en _mat. La coherencia facilita la lectura del código y la detección de problemas.
- No te repitas a ti mismo: ¡automatiza! Si estás repitiendo el mismo código una y otra vez, utiliza un bucle o una función para que repita ese código por ti. La repetición innecesaria no sólo te hace perder tiempo, sino que también aumenta la probabilidad de que cometas un error costoso.

8.2.4 Sea explícito sobre los requisitos y dependencias de su código

Cargar todos los paquetes que serán necesarios para ejecutar tu código (usando las librerías) es una buena forma de indicar qué paquetes son necesarios para ejecutar tu código. Puede ser frustrante llegar a dos tercios del camino a través de un script de larga ejecución sólo para descubrir que una dependencia no ha sido instalada.

Ejemplo:

```
library(ggplot2)
library(reshape)
library(vegan)
```

Otra forma de ser explícito sobre los requisitos de tu código y mejorar su reproducibilidad es limitar la “codificación” de los archivos de entrada y salida de tu script. Si tu código va a leer datos de un archivo, es mejor definir una variable al principio del código que almacene la ruta a ese archivo.

Por ejemplo, esta manera es recomendada:

```
# Definir rutas

input_file <- "data/data.csv"

output_file <- "data/results.csv"

# Leer entrada

input_data <- read.csv(input_file)

# obtener el número de muestras en los datos

sample_number <- nrow(input_data)

# generar resultados

results <- otra_funcion(input_file, sample_number)

# escribir resultados

write.table(results, output_file)
```

Antes que su alternativa:

```
# Leer entrada

input_data <- read.csv("data/data.csv")

# obtener el número de muestras en los datos

sample_number <- nrow(input_data)
```

```
# generar resultados

results <- otra_funcion("data/data.csv", sample_number)

# escribir resultados

write.table("data/results.csv", output_file)
```

8.2.5 Objetos y funciones

Para que el código sea más legible y explícito, trata de que los nombres de las variables sean bien descriptivas. Se suele usar el estilo “*snake_case*” (*todas las letras en minúscula con una guión bajo para separar las palabras*) para **objetos**, y el estilo “*PascalCase*” o “*camelCase*” (*cada palabra comienza con una mayúscula, o excepto la primera palabra*) para **funciones**.

Además, los nombres de los objetos y funciones deben ser cortos.

Un ejemplo de buena práctica:

```
# Buena práctica
vector_123 <- c(1,2,3)
FuncionExponencial(vector_123, 2)

#función FuncionExponencial
NuestraFuncion <- function (base, exponent) {
  return (base**exponent)
}
```

Mala práctica:

```
# Mala Práctica
mi_vector <- c(1,2,3)

mifuncionqueconvierteaExponentes()
mifuncionqueconvierteaExponentes <- function (base, exponent) {
  return (base**exponent)
}
```

Tip: tenga cuidado con la función setwd()

`setwd()` es la función para establecer la ruta del directorio de trabajo, y hay que tener cuidado al usarla. Cambiar directorios en un archivo de script puede limitar la reproducibilidad.

`setwd()` devolverá un error si el directorio al que se intenta cambiar no existe o si el usuario no tiene los permisos correctos para acceder a ese directorio. Esto se convierte en un problema cuando se comparten scripts entre usuarios que han organizado sus directorios de forma diferente.

Si/cuando su script termina con un error, puede dejar al usuario en un directorio diferente al que empezó, y si luego llama al script de nuevo, esto causará más problemas. Si debe usar `setwd()`, es mejor ponerlo al principio del script para evitar estos problemas.

8.2.6 Indentación y espacios

Trate de mantener el mismo número de espacios en todo el código. El editor de RStudio ayudará con ello, pero si eliges 2 espacios o 4 espacios como equivalente de tabuladores debes mantener esa regla en todo el archivo. No se sugiere usar tabuladores como indentaciones.

La única excepción es cuando la definición de una función se extiende por multiples líneas. En ese caso, la segunda línea debe ser indentada adonde comienza la definición. Por ejemplo:

```
funcion_larga <- function(a = "argumento 1",
                           b = "otro argumento",
                           c = "otro argumento largo") {  
}
```

8.2.6.1 Dale un respiro a tu código!

Los espacios ayudan a que el código sea más fácil de leer. Veamos lo feo que se ve este código sin espacios:

```
y=ts(data=c(23,391,728,512,10),start=2010)
```

En comparación con este:

```
y = ts(data = c(23, 391, 728, 512, 10), start = 2010)
```

En Rstudio puedes simplemente pulsar *Ctrl + Shift + A* para autoformatear tu código, y agregar espacios entre las palabras adonde deben estar.