
ESCUELA NACIONAL DE ESTUDIOS SUPERIORES UNIDAD
LEÓN

CENTRO DE CIENCIAS MATEMÁTICAS

UGA-LANGEBIO CINVESTAV

Análisis exploratorio de datos de microbiomas

Capítulo 7

Andrés Arredondo Cruz (andresabstract@gmail.com)
Adriana Haydé Contreras Peruyero (haydeeperuyero@gmail.com)
David Alberto García Estrada (david.garcia.e@cinvestav.mx)

Morelia

Septiembre de 2022

Índice

1. Datos de Ratones y Humanos	3
1.1. Conjunto de datos de ratón Vdr—/—	3
1.2. Conjunto de datos de fumadores de cigarro	3
2. Análisis exploratorio como resumen gráfico	3
2.1. Gráficos de riqueza	3
2.2. Gráfico de barras de abundancia	6
2.3. Gráfico de mapa de calor	9
2.4. Gráfico de redes	14
2.5. Gráfica de árbol filogenético	16
3. Agrupamiento (Clústers)	20
3.1. Introducción a la agrupación, la distancia y la ordenación	20
3.2. Distancias y disimilitudes	21
3.2.1. Distancias y disimilitudes entre datos	22
3.3. Agrupamiento (Clústers)	29
3.3.1. Clustering aglomerativo de la liga sencilla	29
3.3.2. Clustering aglomerativo de la liga completa	30
3.3.3. Clustering aglomerativo de la liga promedio	31
3.3.4. Agrupación de varianza mínima de Ward	32
3.3.5. Mismos gráficos usando el paquete factoextra	35
4. Ordenación	39
4.1. Ordenaciones NO restringidas	40
4.1.1. Análisis de componentes principales (PCA)	40
4.1.2. Análisis de coordenadas principales (PCoA)	50
4.1.3. Escalamiento multidimensional no métrico (NMDS)	66
4.1.4. Análisis de correspondencia (AC)	69
4.2. Ordenaciones restringidas	74
4.2.1. Análisis de redundancia (RDA)	74
4.2.2. Análisis de correspondencia restringido (CCA)	83
4.2.3. Análisis restringido de coordenadas principales (CAP)	90
5. Resumen y discusión	94

1. Datos de Ratones y Humanos

1.1. Conjunto de datos de ratón Vdr—/—

Los datos del microbioma intestinal murino (Jin et al. 2015) se obtuvieron de muestras de heces fecales y cecales. Aquí, se utilizan las muestras fecales.

1.2. Conjunto de datos de fumadores de cigarro

El segundo conjunto de datos es de Charlson et al. (2010) y Chen (2012), que incluye estudios sobre el efecto del tabaquismo en el microbioma del tracto respiratorio superior. El conjunto de datos original contiene muestras de microbiomas de garganta y nariz, y de ambos lados del cuerpo. El conjunto de datos utilizado en este capítulo proviene del microbioma de la garganta del lado izquierdo del cuerpo. Contiene 60 sujetos (32 no fumadores y 28 fumadores). El conjunto de datos incluye tres datos: recuento de abundancia, árbol y metadatos. Es adecuado para ilustrar el diagrama de árbol y el análisis de ordenación restringida.

2. Análisis exploratorio como resumen gráfico

Podemos dividir los métodos de estudio de la composición de la comunidad de microbiomas en dos componentes principales: - Análisis de diversidades taxonómicas - Análisis multivariante de la composición del microbioma

El análisis multivariado incluye varias técnicas multivariadas, como el agrupamiento y la ordenación (sin restricciones y con restricciones) y las diferencias de prueba de hipótesis entre los grupos.

Usaremos el paquete de “Phyloseq” que es una herramienta bastante importante, almacena, analiza y muestra gráficamente datos complejos filogenéticos. Los datos de entrada usados en este paquete pueden ser OTUs o datos de abundancia de cuentas. Este paquete usa sistemas gráficos y avanzados (ggplot2) para facilitar gráficas de calidad de datos.

En este capítulo se exploran diferentes gráficos usuales: riqueza, barras de abundancia, mapas de calor, redes y árbol filogenético.

2.1. Gráficos de riqueza

Las diversidades alfa estimadas pueden resumirse mediante un gráfico utilizando la función `plot_richness()` del paquete `phyloseq`. Aunque su nombre sugiere trazar “riqueza”, que normalmente se refiere a trazar el número total de especies/taxa/OTUs en una muestra o entorno, en realidad la función no sólo traza la riqueza, sino que también genera cifras de diversidades observadas y otras estimadas.

En primer lugar, debemos cargar los paquetes `phyloseq` y `ggplot2`, y el conjunto de datos de ratones Vdr—/—.

```
#Para instalar el paquete, usamos Bioconductor
#if (!require("BiocManager", quietly = TRUE))
#install.packages("BiocManager")
#BiocManager::install("phyloseq")
```

```
# IMPORTANTE
# Ajustamos path de trabajo según tu PC
path <- "~/R_sites/"
path <- paste0(path, "Equipo4/")
setwd(path)
```

```
# cargamos librerias
#library("phyloseq")
#library("ggplot2")
#library("igraph")
#library("vegan")
#library("GUniFrac")
#library("pbkrtest")
#library("BiodiversityR")
```

En la carpeta data se encuentra la base de datos a usar VdrFecalGenusCounts.csv.

```
# Mandamos llamar los datos originales
# Fila es el taxón y columna la condición
abund_table=read.csv(paste0(path,"data/VdrFecalGenusCounts.csv"), row.names=1,check.names=FALSE)
# Generamos la transpuesta, fila la condición, columna el taxón
abund_table<-t(abund_table)
```

El paso crítico cuando se utiliza el paquete phyloseq es construir un objeto phyloseq-class. El siguiente código R construye un objeto de clase phyloseq llamado physeq utilizando el comando phyloseq(). El objeto de clase phyloseq se construye a partir de sus componentes datos: - Tabla OTU. - Datos muestra. - Tabla de taxonomía. - Árbol filogenético.

Como objeto a nivel de experimento, deben proporcionarse dos o más objetos de datos componentes. El orden de los argumentos no importa.

```
# Hacemos una tabla que incluye el nombre de las filas y además incluye como dato,
#estos nombres separados por "_"
meta_table <- data.frame(row.names=rownames(abund_table),
                        t(as.data.frame(strsplit(rownames(abund_table),"_"))))
# Hacemos factor la columna "X2" y los que sean 11, 12, 13, 14 y 15 serán "Vdr-/-",
#el resto será "WT"
meta_table$Group <- with(meta_table,ifelse(as.factor(X2) %in%
                                           c(11,12,13,14,15), c("Vdr-/-"), c("WT")))
```

Convertimos los datos al formato phyloseq.

```
# OTUs
OTU <- otu_table(as.matrix(abund_table), taxa_are_rows = FALSE)
# Sample
SAM <- sample_data(meta_table)
# Unimos en un objeto phyloseq
physeq <- merge_phyloseq(phyloseq(OTU),SAM)
physeq
```

```
## phyloseq-class experiment-level object
## otu_table() OTU Table: [ 248 taxa and 8 samples ]
## sample_data() Sample Data: [ 8 samples by 4 sample variables ]
```

Una vez que tenemos nuestro objeto phyloseq, podemos usar la función plot_richness() para construir graficar las diversidades alpha observadas y estimadas.

```
plot_richness(physeq, x = "Group ", color = "Group ")
```

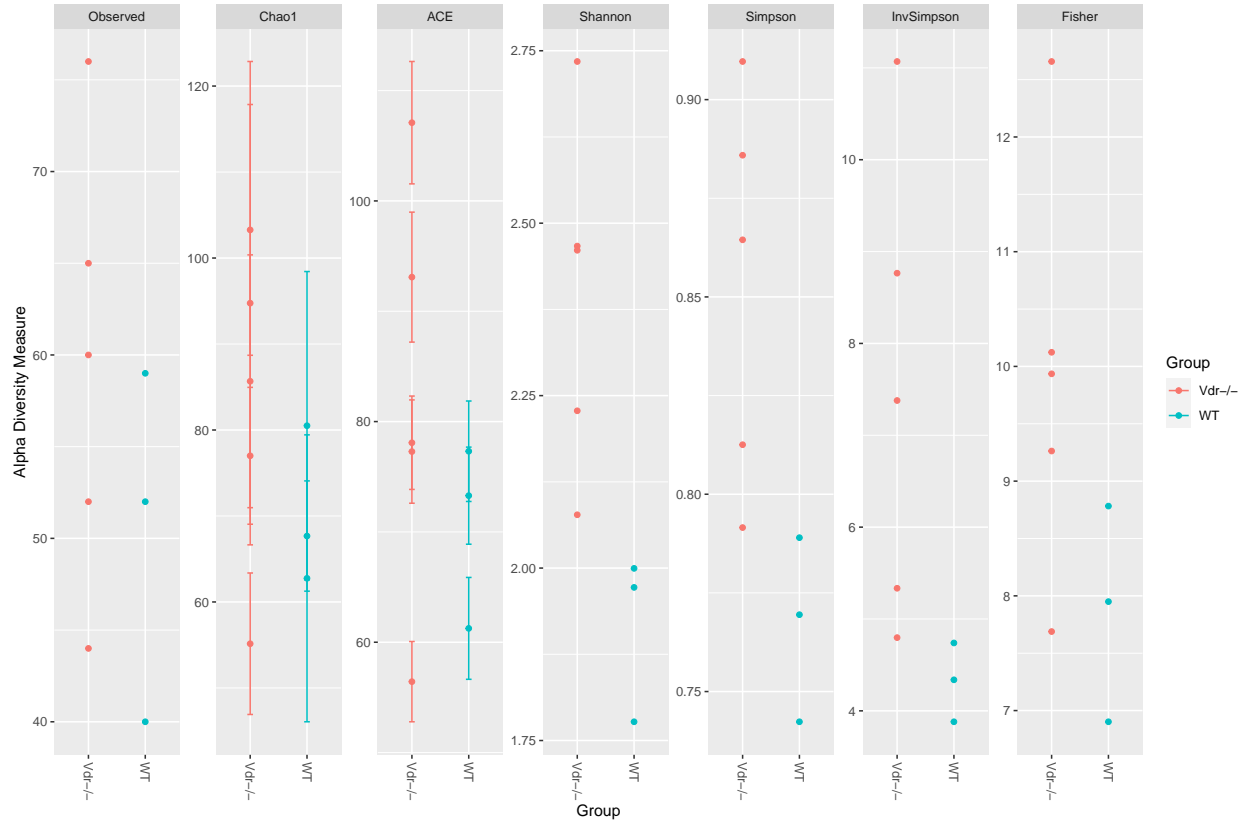


Figura 1: Gráficos de diversidad alpha con Vdr y grupos WT en muestras de heces.

Esta función también nos permite seleccionar solo algunas diversidades. El siguiente es un ejemplo usando solo dos diversidades, la de Chao1 y Shannon.

```
plot_richness(physeq, measures = c("Chao1", "Shannon"), x = "Group ", color = "Group ")
```



Figura 2: Gráficos de diversidad alpha seleccionando las diversidades de Chao y Shannon.

Se requiere el dato de entrada “physeq”, que es de la clase “phyloseq”, o alternatively, un una tabla OTUs. El argumento opcional “x” es una variable que se asigna al eje horizontal; x puede ser una cadena de caracteres o un vector. El valor por defecto es “samples”, que asignará el nombre de cada muestra a una posición horizontal distinta en el gráfico. En este caso, $x = \text{"Group"}$ asignará la pertenencia a un grupo al eje x. El argumento color también es opcional. Especificará la variable de muestra que se asignará a diferentes colores. Al igual que el argumento x, puede ser una cadena de caracteres o un vector. En la estimación de la diversidad alfa, el ruido puede recortarse; sin embargo, dado que muchas estimaciones de riqueza e incluso la riqueza “observada” dependen en gran medida del número de individuos. Por lo tanto, si desea obtener resultados significativos, debe utilizar conjuntos de datos sin recortar (McMurdie y Holmes 2013).

2.2. Gráfico de barras de abundancia

El diagrama de barras por defecto, sin ningún parámetro, trazará con cada muestra individualmente en el eje x, y los valores de abundancia en el eje y. Los valores de abundancia para cada OTU/muestra se apilan en el orden de mayor a menor, separados por una fina línea horizontal.

```
theme_set(theme_bw())
# Gráfico de barras
plot_bar(physeq)
```

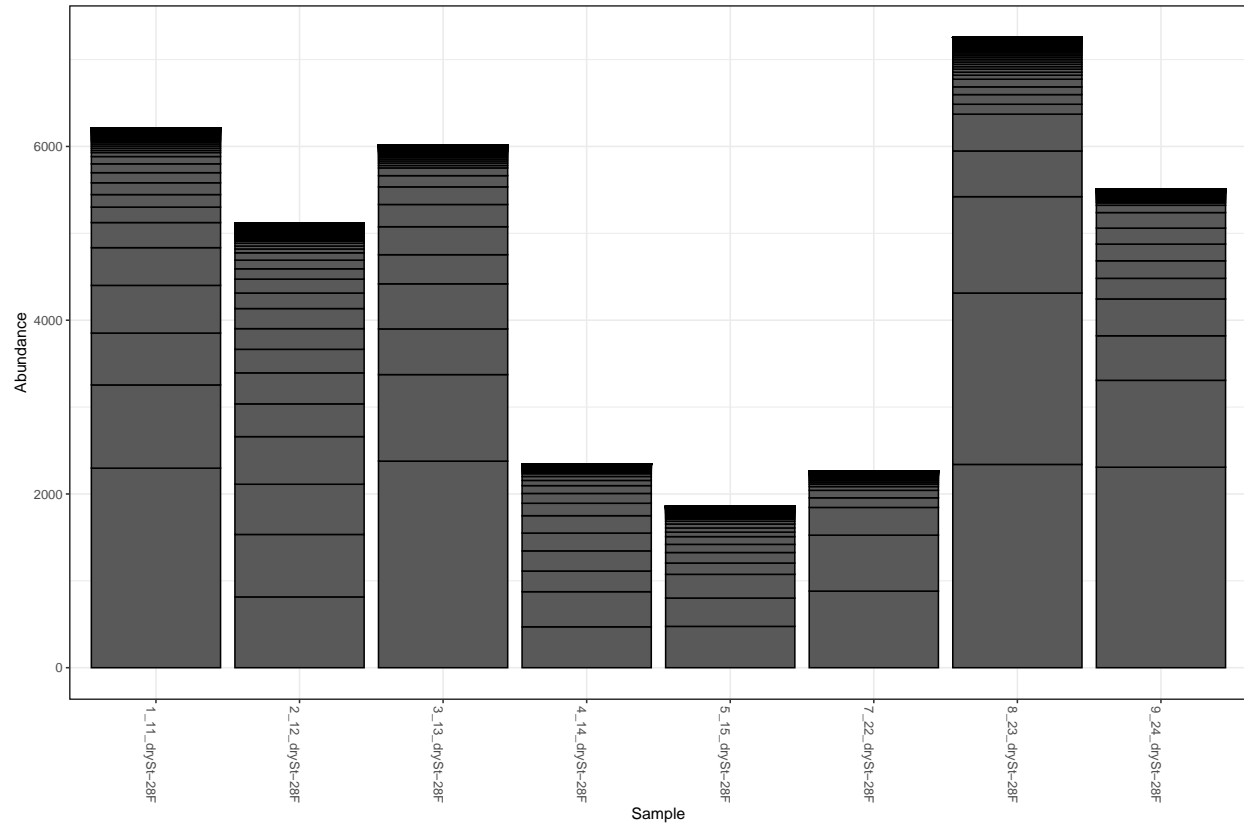


Figura 3: Gráfico de barras de abundancia por defecto.

Otros parámetros que se pueden proporcionar son `fill` y `facet_grid`. El parámetro `fill` es una cadena de caracteres que especifica cual variable muestral puede ser usada para mapearla a los colores con los cuales se llenan las barras.

```
plot_bar(physeq, fill="Group")
```



Figura 4: Gráfico de barras de abundancia indicando cual variable es mapeada al parámetro fill.

El parámetro `facet_grid` es una fórmula que especifica las facetas a mostrar en el gráfico. En el siguiente código, primero encontramos los 5 taxones con más abundancia y después los graficamos añadiendo también el parámetro de `facet_grid`.

```
# Nos da el nombre de los taxones top 5
TopNGenus <- names(sort(taxa_sums(physeq), TRUE)[1:5])
# Generamos un objeto phyloseq con solo los 5 taxa que escogimos
Top5Genus <- prune_taxa(TopNGenus, physeq)
# Graficamos
plot_bar(Top5Genus, fill="Group", facet_grid=~Group)
```

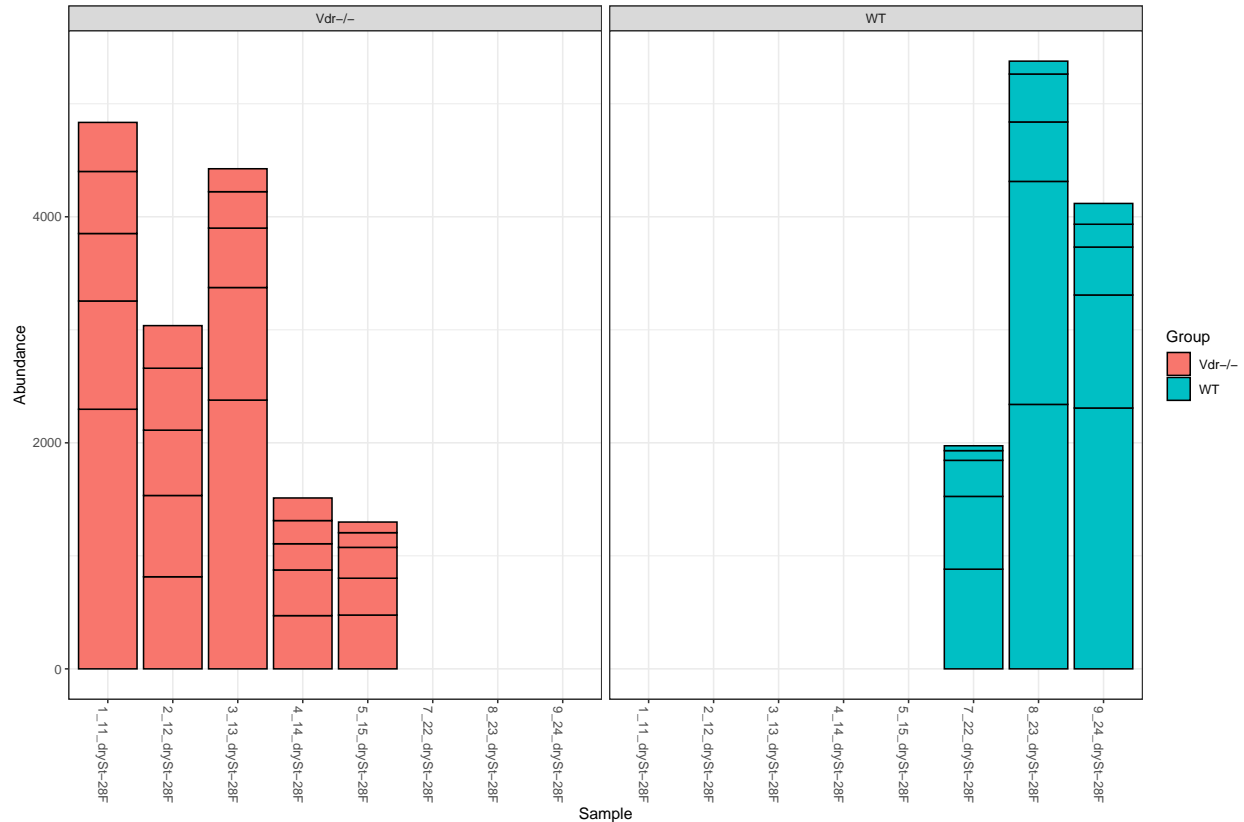



Figura 5: Gráfico de barras de abundancia de los 5 taxones con mayor abundancia.

Otra opción que podríamos especificar dentro del parámetro `fill` es `Genus` pero estos datos no cuentan con ese parámetro.

```
# plot_bar(Top5Genus, fill="Genus", facet_grid=~Group)
```

2.3. Gráfico de mapa de calor

El ordenamiento basado en la ordenación es mucho mejor que el cluster jerárquico para presentar los datos del microbioma. Aquí, nos centramos en ilustrar la función `plot_heatmap()` basada en los métodos de ordenación NMDS y PCA. A continuación se presenta un uso de la función `plot_heatmap()`.

```
plot_heatmap(physeq_object, method = "NMDS", distance = "bray", sample.label = NULL,
             taxa.label = NULL, low = "#000033", high = "#66CCFF", na.value = "black").
```

Se requiere el argumento de datos de entrada “physeq”. Es un objeto de la clase `phyloseq` (`otu_table`). Tanto el método como los aumentos de distancia son opcionales. El método de ordenación es el que se utilizará para organizar el mapa de calor. El método de distancia ecológica es una cadena de caracteres para usar en la ordenación. Tanto “sample.label” como “taxa.label” son cadenas de caracteres y opcionales para usar para etiquetar el eje de la muestra (horizontal) y reetiquetar el eje de taxones/especies/OTU (vertical), respectivamente. Los aumentos bajos y altos son cadenas de caracteres y opcionales. Se utilizan para elegir las opciones de color compatibles con R. R entiende más de 600 colores. Puede escribir `colors()` en R para comprobar los nombres.

En los mapas de calor, los valores 0 son tratados como valores faltantes (NA) y se mapean al color negro. Por lo general, los valores más pequeños se representan por default con el color azul oscuro; mientras que los valores altos con los colores más claros. Para ejemplificar esto, vamos a usar primero solo los cinco taxones con mayor abundancia.

```
# Nos da el nombre de los taxones top 5
TopNGenus <- names(sort(taxa_sums(physeq), TRUE)[1:5])
# Generamos un objeto phyloseq con solo los 5 taxa que escogimos
Top5Genus <- prune_taxa(TopNGenus, physeq)
# Graficamos
plot_heatmap(Top5Genus)
```

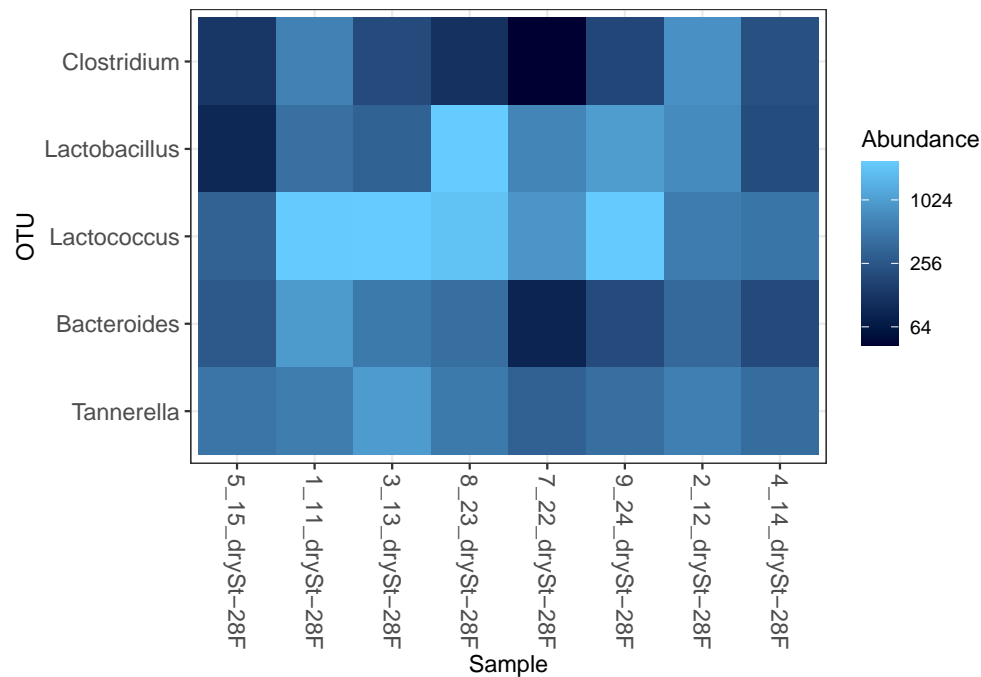


Figura 6: Mapa de calor de los 5 taxones con mayor abundancia.

Vamos a especificar otros parámetros como la distancia y métodos.

```
# Método NMDS y distancia de Bray-Curtis
p <- plot_heatmap(Top5Genus, "NMDS", "bray")
p
```

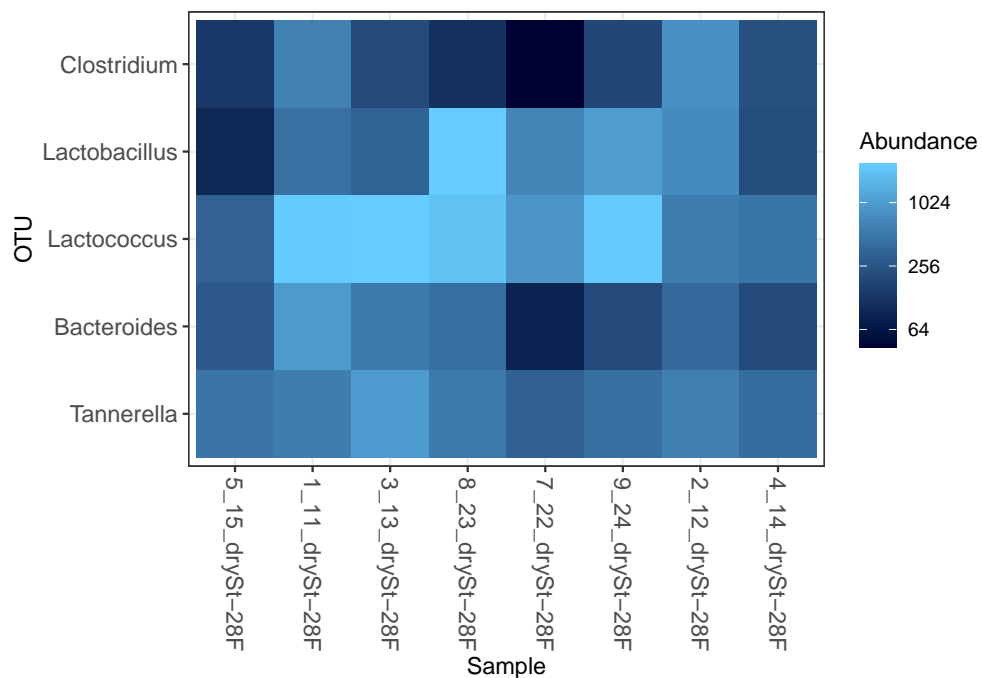


Figura 7: Mapa de calor de los 5 taxones con mayor abundancia usando el método NMDS y la distancia Bray-Curtis.

Podemos especificar también los rangos de colores.

```
plot_heatmap(Top5Genus, "NMDS", "bray", low="#000033", high="#CCFF66")
```

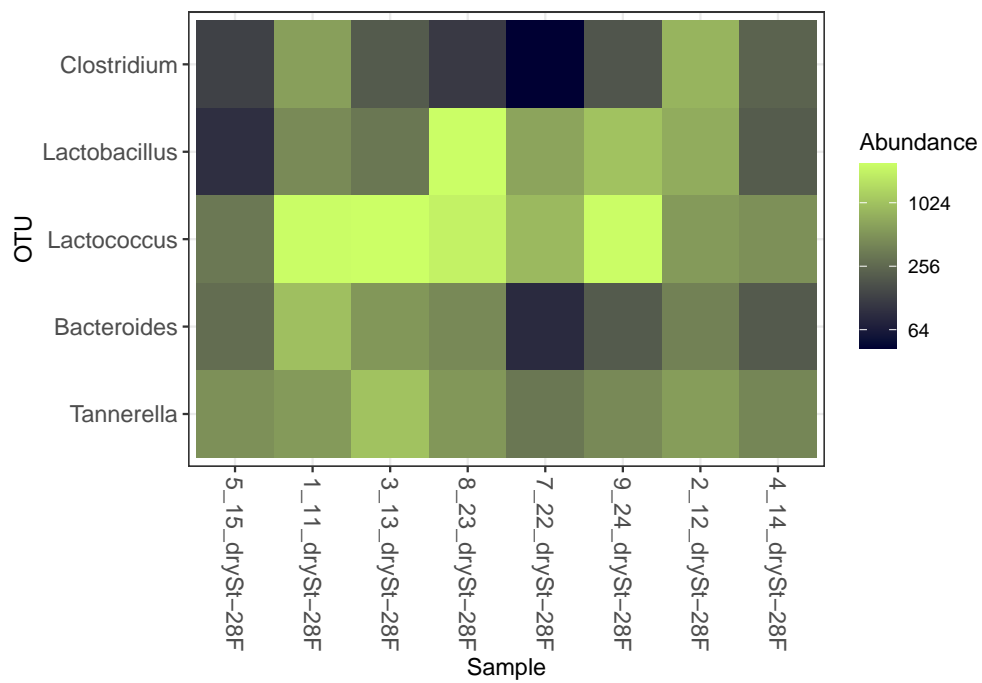


Figura 8: Mapa de calor especificando otras escalas de colores.

```
plot_heatmap(Top5Genus, "NMDS", "bray", low="#000033", high="#FF3300")
```

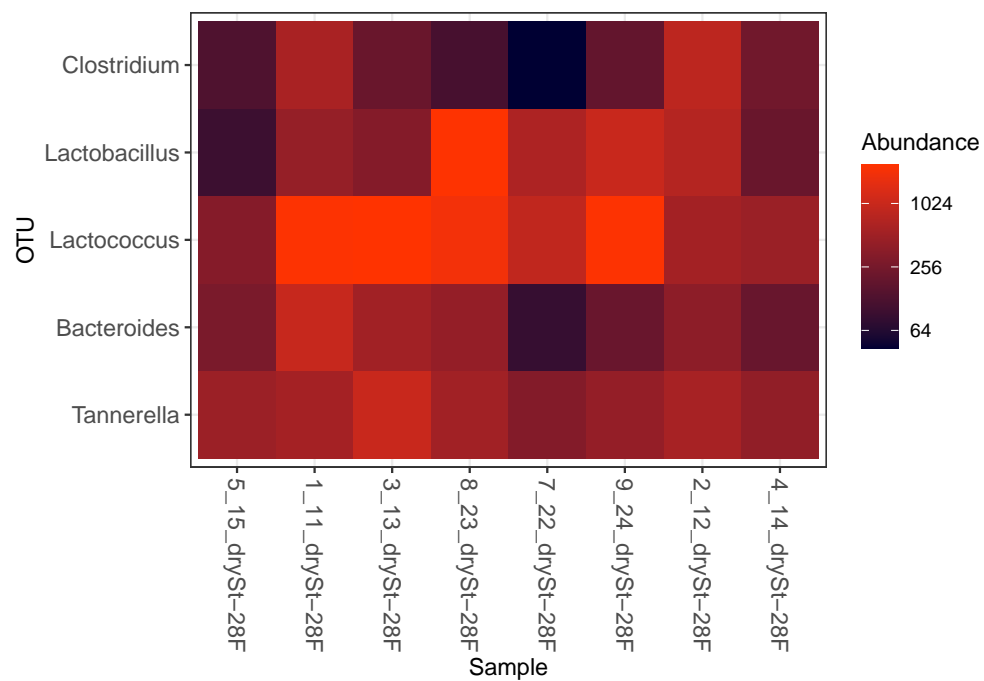


Figura 9: Mapa de calor especificando otras escalas de colores.

```
plot_heatmap(Top5Genus, "NMDS", "bray", low="#000033", high="#66CCFF")
```

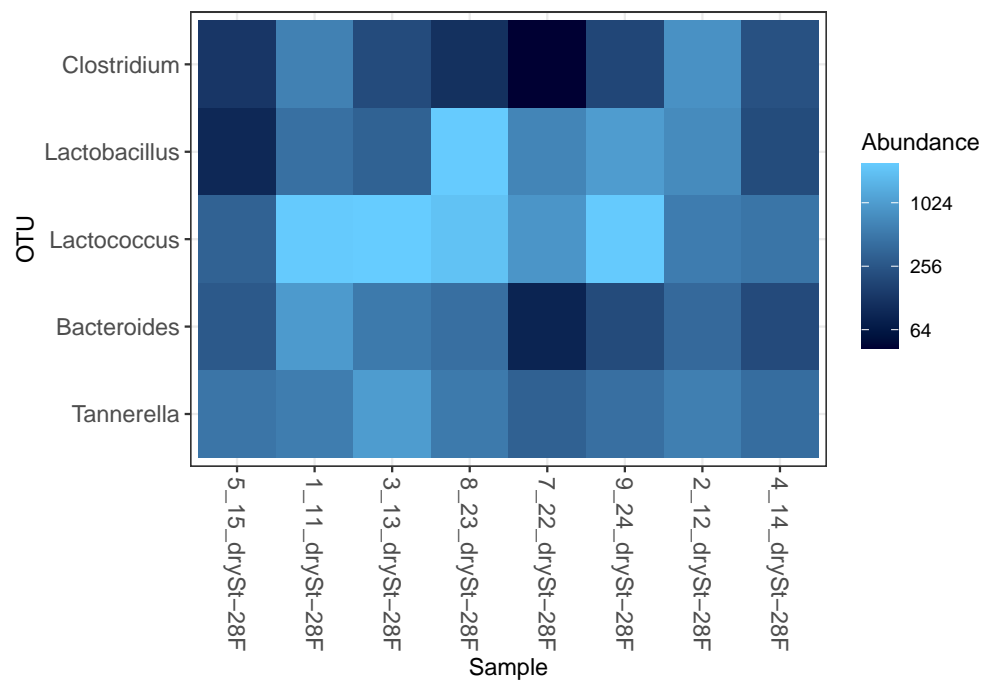


Figura 10: Mapa de calor especificando otras escalas de colores.

```
plot_heatmap(Top5Genus, "NMDS", "bray", low="#66CCFF", high="#000033", na.value="white")
```

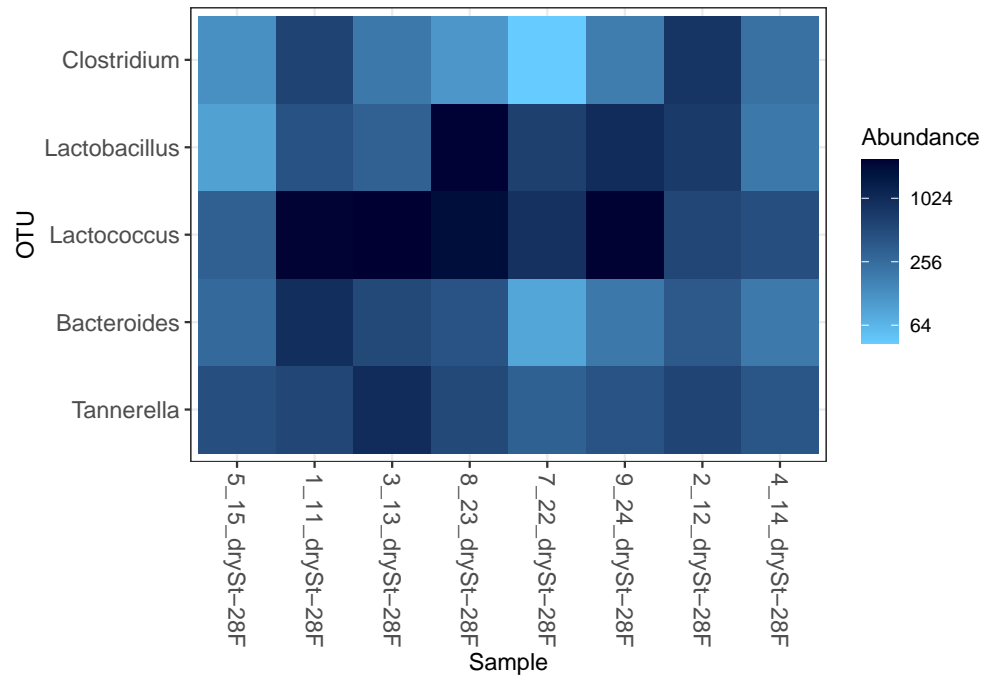


Figura 11: Mapa de calor especificando otras escalas de colores.

El siguiente mapa de calor es un ejemplo usando PCoA.

```
# Método de PCoA y distancia de Bray-Curtis
plot_heatmap(Top5Genus, "PCoA", "bray")
```

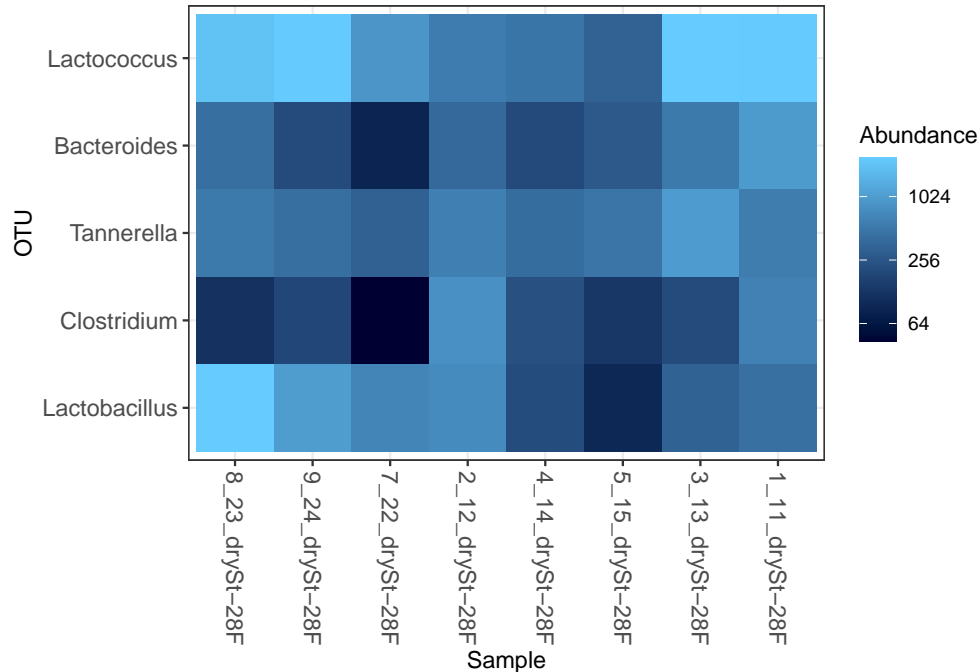


Figura 12: Mapa de calor usando PCoA.

2.4. Gráfico de redes

Hay dos funciones en el paquete `phyloseq` para trazar la red del microbioma usando “`ggplot2`”: `plot_network()` y `plot_net()`. Si se utiliza la función `plot_network()`, su primer argumento debe ser un objeto `igraph`, y la red en sí debe representarse utilizando el paquete `igraph`. El objeto red (argumento `g`) puede ser creado usando la función `make_network()` a través del paquete `phyloseq`.

La función `plot_net()` es una revisión del rendimiento y de la interfaz de `plot_network()`, su primer/principal argumento es una instancia de la clase `phyloseq`. Los ejemplos de uso de `plot_network()` y `plot_net()` se dan a continuación:

- `plot_network(g, physeq=NULL, type="samples", color="Group", shape="Group")`
 - `g` es un objeto de clase `igraph` necesario creado por la función `make_network()`, o directamente por el paquete `igraph`.
 - El argumento opcional `physeq` es un objeto de clase `phyloseq` en el que se basa `g`.
 - La opción `Type` indica si el grafo representado en el argumento primario, `g`, es de muestras o de taxones/OTUs. Por defecto es “`samples`”.
 - Las opciones `color` y `shape` son opcionales. Se utilizan para el mapeo de color y el mapeo de forma de los puntos.
- `plot_net(physeq, distance = "bray", type = "samples", maxdist = 0.7, color = NULL, shape = NULL)`
 - El argumento requerido `physeq`, es el objeto de clase `phyloseq` que se quiere representar como una red.
 - La opción de distancia es un método de distancia o una clase-distancia ya calculada. Por defecto es `bray`.
 - La opción `maxdist` significa el valor máximo de distancia entre dos vértices para conectar con una arista en el grafo. El valor por defecto es 0,7.

Los siguientes códigos de R crean un grafo basado en igraph, basado en el método de distancia por defecto, *Jaccard* y una distancia máxima entre nodos conectados de 0,8. El “Group” se utiliza para los mapeos de color y forma para visualizar la estructura de las muestras de ratones *Vdr*^{-/-} y *WT*.

```
set.seed(123)
library("igraph")
# Hacemos una gráfica a partir de el objeto phyloseq
ig <- make_network(physeq, max.dist=0.8)
# Graficamos
plot_network(ig, physeq, color="Group", shape="Group")
```

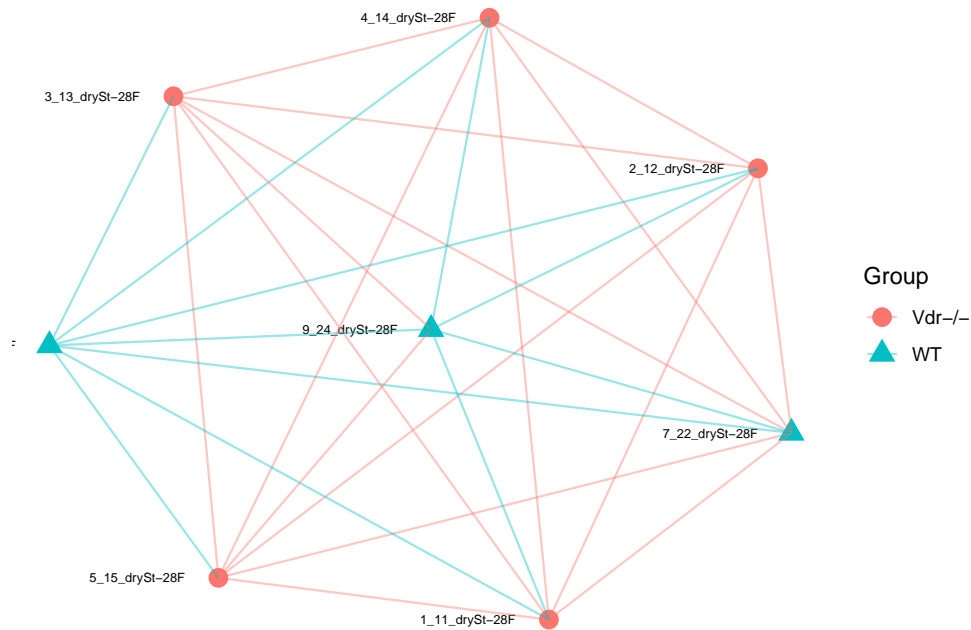


Figura 13: Gráfico usando plot network().

El gráfico anterior muestra una estructura interesante, con dos subgráficos que comprenden las muestras de los ratones *Vdr*^{-/-} y *WT* respectivamente. Además, parece haber una correlación entre las muestras.

En comparación con la función `plot_network()`, la nueva función `plot_net()` no requiere una llamada separada a la función `make_network()`, o un objeto igraph separado. Los códigos siguientes crean una red basada en una distancia máxima entre los nodos conectados de 0,5. Como estamos interesados en cómo la estructura de las muestras de ratones *Vdr*^{-/-} y *WT* muestras, utilizamos el “Group” para el mapeo de color y el mapeo de forma de los puntos.

```
# Graficamos sin necesidad de hacer el objeto anterior
plot_net(physeq, maxdist = 0.5, color = "Group", shape="Group")
```

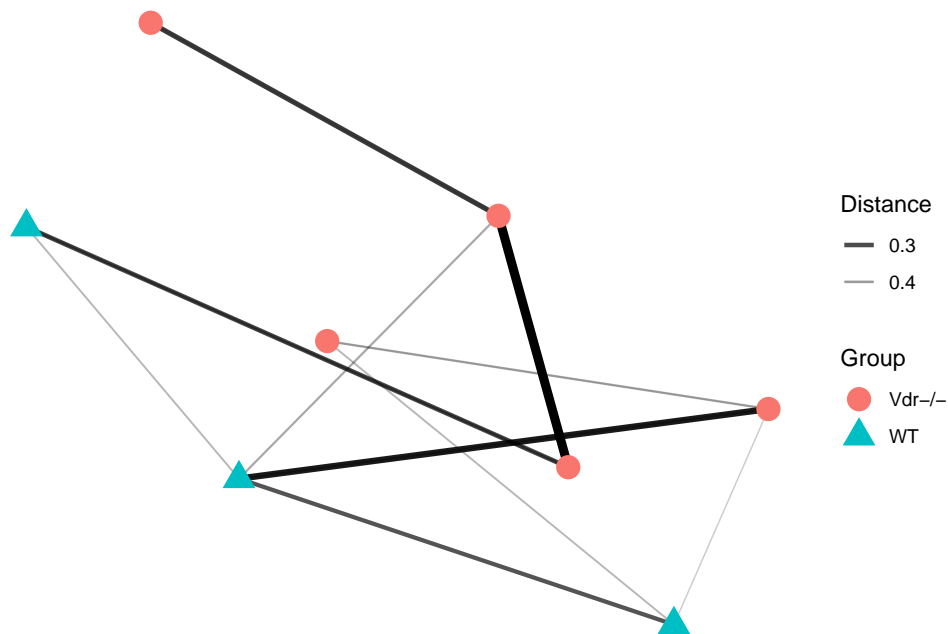


Figura 14: Gráfico de red creado con plot.net().

2.5. Gráfica de árbol filogenético

La función `plot_tree()` del paquete `phyloseq` pretende facilitar la investigación gráfica del árbol filogenético, así como de los datos de la muestra. Para la secuenciación filogenética de muestras con gran riqueza, el árbol generado por esta función será difícil de leer e interpretar. Una “regla general” propuesta por los autores del paquete `phyloseq` es utilizar subconjuntos de datos con no más de 200 OTU por parcela. A continuación se muestra un uso de esta función:

```
■ plot_tree(physeq, method = "sampledodge", color = NULL, shape = NULL, ladderize = FALSE)
```

Los datos de entrada “physeq” son necesarios. Los datos deben ser de la clase `phyloseq` y contener como *mínimo un componente de árbol filogenético*. Son estos datos los que se quieren trazar y anotar un árbol filogenético. La opción “method” es el nombre del método de anotación a utilizar. El método por defecto “sampledodge” el cual dibujará puntos junto a las hojas si se observaron individuos de ese taxón, y un punto separado para cada muestra. Los argumentos “color” y “forma” son opcionales. Proporcionan los nombres de las variables en `physeq` para mapear el color del punto, y mapear la forma del punto, respectivamente. La opción “ladderize” se utiliza para especificar si el árbol se ordena en forma de escalera o no, es decir reordenar los nodos de acuerdo con la profundidad de sus subárboles antes de trazarlos. Por defecto es `FALSE`, no se aplica la jerarquización. Cuando es `TRUE` o “right”, se utiliza la se utiliza la jerarquización. Cuando se establece como “izquierda”, se aplica la escalonación “izquierda”. El conjunto de datos de los fumadores se utiliza para ilustrar el trazado del árbol filogenético. Los datos son del paquete `GUniFrac` (Charlson et al. 2010; Chen 2012). Primero vamos a cargar este paquete y hagamos que los datos estén disponibles para su uso.

```
library("GUniFrac")
data(throat.otu.tab)
```



```
data(throat.tree)
data(throat.meta)
```

Ahora, necesitamos construir nuestro objeto phyloseq.

```
# Convertimos los datos en un objeto phyloseq
# Tabla de OTUs
OTU <- otu_table(as.matrix(throat.otu.tab), taxa_are_rows = FALSE)
# Tabla de Sample
SAM <- sample_data(throat.meta)
# Árbol filogenético
TRE <- throat.tree
# Generación del objeto de phyloseq
physeq <- merge_phyloseq(phyloseq(OTU), SAM, TRE)
```

Verificamos cuantos taxones/OTUs tienen nuestros datos.

```
# Número de taxones
ntaxa(physeq)
```

```
## [1] 856
```

Y seleccionamos solo los primeros 50.

```
# Nos quedamos con los primeros 50
physeq <- prune_taxa(taxa_names(physeq)[1:50], physeq)
```

Ahora, graficamos usando como variable en el color el estado de fumador.

```
plot_tree(physeq, ladderize = "left", color = "SmokingStatus")
```

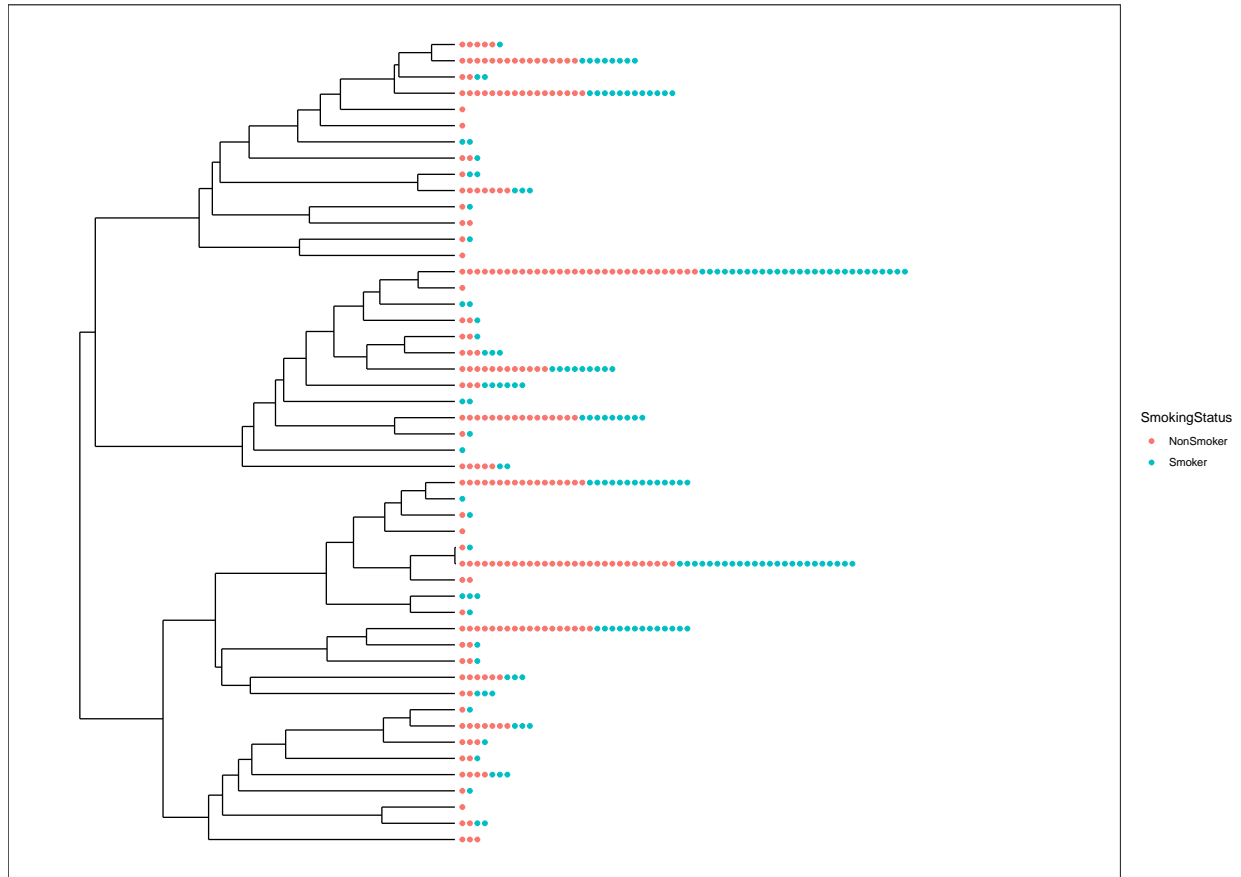


Figura 15: Árbol filogenético usando como color la variable SmokingStatus.

```
plot_tree(physeq, ladderize = "left", color = "SmokingStatus", shape = "SmokingStatus")
```

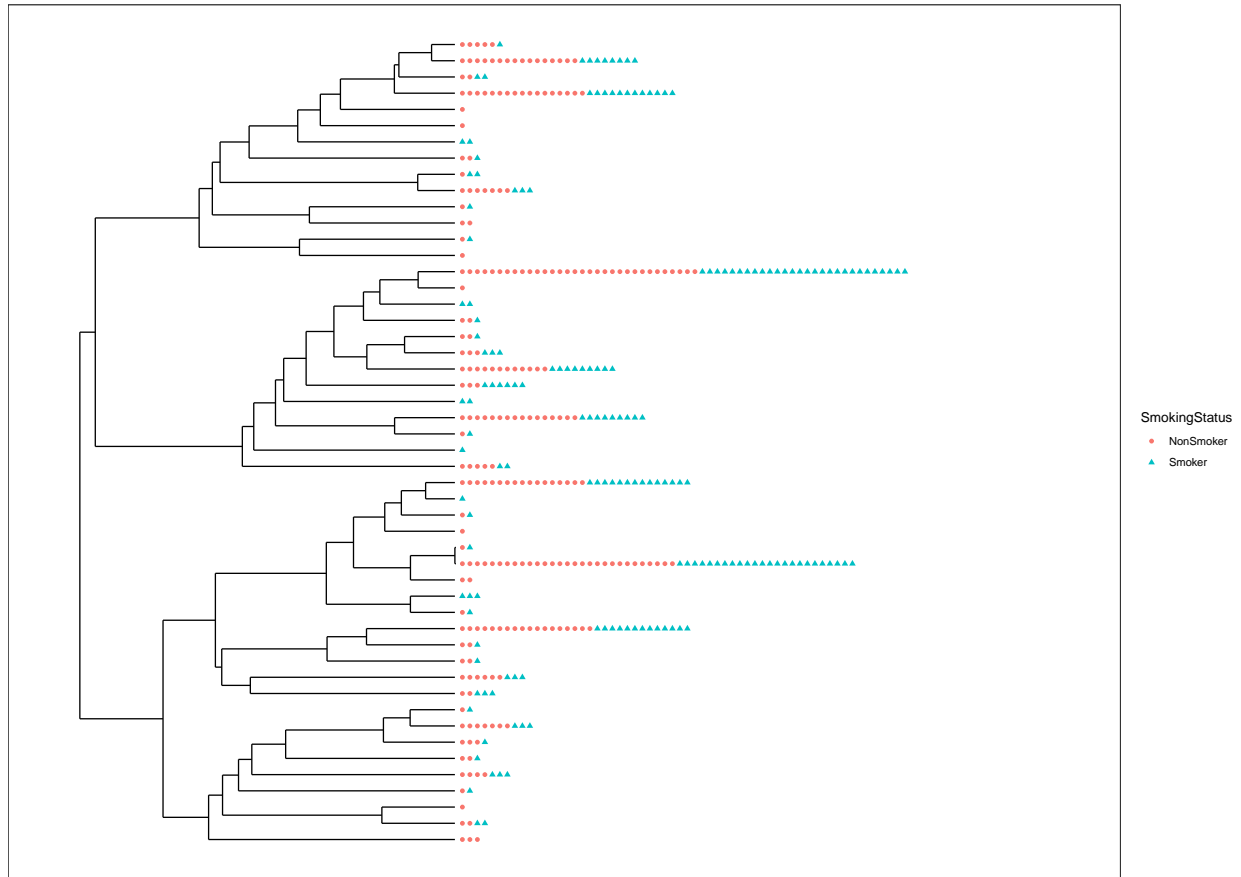


Figura 16: Árbol filogenético usando como color y forma la variable SmokingStatus.

Otro tipo de gráfico que se suele ocupar es al mapear los datos a coordenadas polares.

```
plot_tree(physeq, ladderize = "left", color = "SmokingStatus", shape = "SmokingStatus") +
  coord_polar(theta = "y")
```

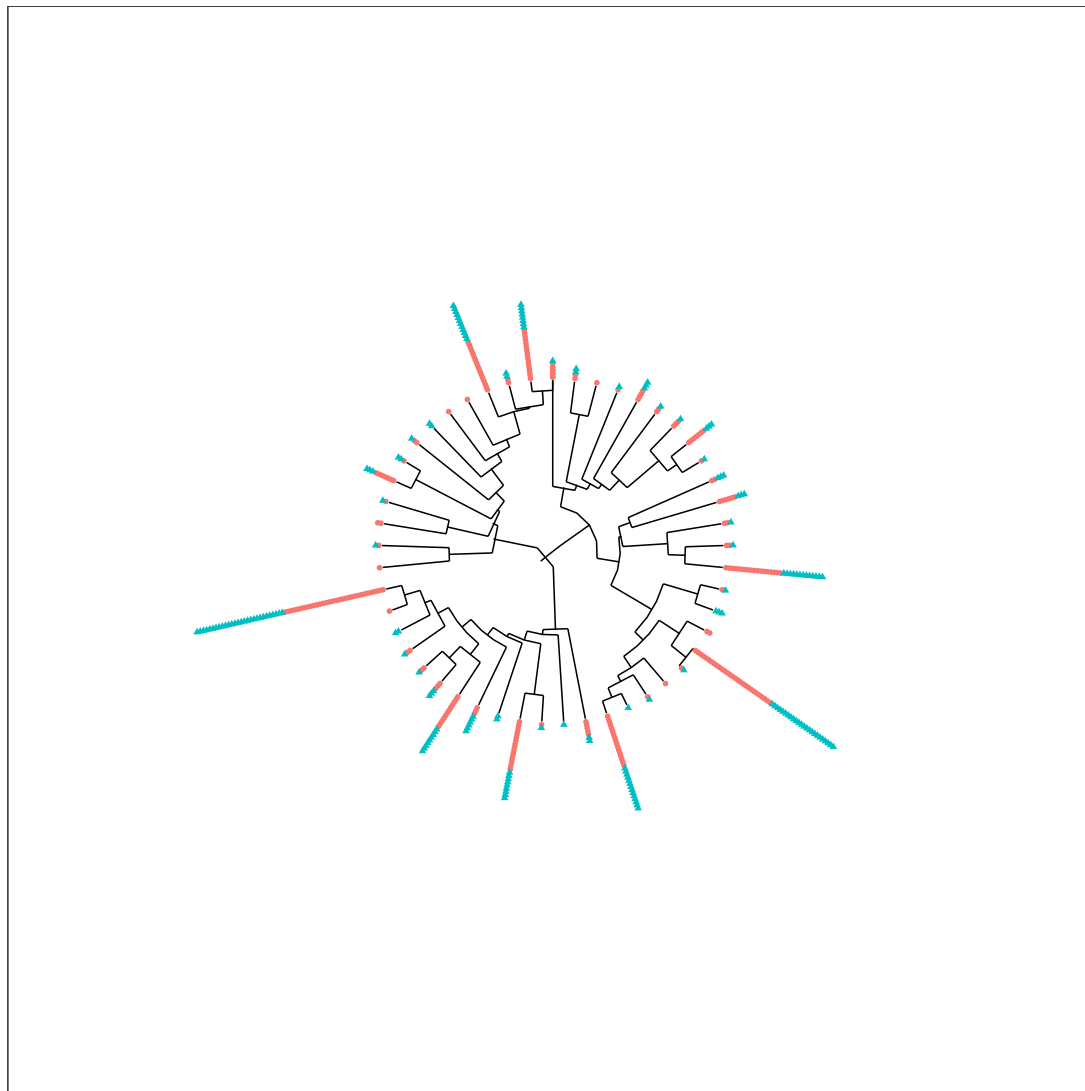


Figura 17: Árbol filogenético con coordenadas polares

3. Agrupamiento (Clústers)

3.1. Introducción a la agrupación, la distancia y la ordenación

La *agrupación* (o *clasificación*) y la *ordenación* son las dos clases principales de métodos multivariantes que suelen emplear los investigadores del microbioma y los ecologistas de comunidades. Hasta cierto punto, estos dos enfoques son complementarios. El objetivo de la agrupación es clasificar las muestras en clases (quizás jerárquicas) para reducir la complejidad (dimensionalidad) de los datos; puede reducir todas las muestras en una dimensión (eje x).

Sin embargo, los datos con dos o tres dimensiones son más interpretables que los con una dimensión porque la

mayoría de los datos comunitarios son continuos. Con la *ordenación* los datos comunitarios pueden reducirse a dos o tres dimensiones. Por lo tanto, la *ordenación* suele ser deseada por los investigadores del microbioma y los ecologistas de comunidades. Se han desarrollado muchos métodos multivariantes basados en la *ordenación* en el estudio del microbioma y la ecología. Cuando se realiza la *agrupación* y la *ordenación*, es necesario que un método de distancia proporcione una medida de distancia.

3.2. Distancias y disimilitudes

Antes de entrar al tema de análisis de clústers es importante introducir los conceptos de distancias y disimilitudes o similitudes.

Una medida de *disimilitud* o *distancia* entre dos sujetos muy distintos será grande y por el contrario, una medida de *similitud* entre ellos será pequeña.

Una **distancia** es una función $d : \mathcal{X} \times \mathcal{X} \rightarrow \mathbb{R}$ que cumple las siguientes propiedades (ver [?]):

1. Para cualesquiera $x, y \in \mathcal{X}$ se tiene que $d(x, y) \geq 0$.
2. Se cumple que $d(x, y) = 0$ si y sólo si $x = y$.
3. Para cualesquiera $x, y \in \mathcal{X}$ se cumple que $d(x, y) = d(y, x)$ (condición de simetría).
4. Para cualesquiera $x, y, z \in \mathcal{X}$ se cumple la desigualdad del triángulo:

$$d(x, y) \leq d(x, z) + d(z, y).$$

Dos de las distancias más comunes en estadística para variables aleatorias multivariadas cuantitativas son la distancia euclidiana y la distancia de Mahalanobis:

1. Si $\bar{x}, \bar{y} \in \mathbb{R}^n$, entonces la distancia euclidiana d_E entre \bar{x} y \bar{y} está dada por

$$d_E(\bar{x}, \bar{y}) = [(\bar{x} - \bar{y})' \cdot (\bar{x} - \bar{y})]^{\frac{1}{2}}$$

2. Si $\bar{x}, \bar{y} \in \mathbb{R}^n$, entonces la distancia de Mahalanobis d_M entre \bar{x} y \bar{y} está dada por

$$d_M(\bar{x}, \bar{y}) = [(\bar{x} - \bar{y})' \Sigma^{-1} (\bar{x} - \bar{y})]^{\frac{1}{2}}$$

donde Σ^{-1} denota a la matriz inversa de la matriz de covarianzas de \bar{x} y \bar{y} .

Existen otras distancias, como la distancia de Gower que puede ser utilizada para estudiar variables aleatorias con entradas cuantitativas y/o cualitativas.

Informalmente, una **similitud** es una medida entre dos objetos de que tanto se parecen. Por lo cual entre más parecidos sean los dos objetos su medida de similitud será más grande. Usualmente toma valores positivos entre 0 y 1.

Formalmente, una similitud es una función $s : \mathcal{X} \times \mathcal{X} \rightarrow \mathbb{R}$ que típicamente satisface las siguientes dos condiciones:

1. $s(x, y) = 1$ si y sólo si $x = y$. Además $0 \leq s \leq 1$.
2. $s(x, y) = s(y, x)$ para todo x y y (condición de simetría).

Usualmente, la función de similitud satisface $s(x, y) \geq 0$ pero existen ejemplos de similitudes que no lo satisfacen, por ejemplo: el coeficiente de correlación y los productos escalares. Sin embargo, si la función de similitud no es positiva y está acotada siempre es posible transformarla en una función de similitud positiva añadiendo una constante, es decir, podemos escribir nuestra nueva función de similitud \tilde{s} como $\tilde{s}(x, y) = s(x, y) + c$, donde c es una constante positiva lo suficientemente grande (ver [?] y [?]).

A diferencia de una distancia o una disimilitud, no existe un análogo para la desigualdad del triángulo para la similitud.

Informalmente, una **disimilitud** entre dos objetos es una medida del grado en el cual dos objetos son diferentes. Como es de esperarse, una disimilitud debe ser baja si se consideran pares de objetos que son similares.

De manera precisa, una disimilitud es una función $d : \mathcal{X} \times \mathcal{X} \rightarrow \mathbb{R}$ que satisface las siguientes dos condiciones:

1. Para cualquier $x \in \mathcal{X}$ se tiene que $d(x, x) = 0$.
2. Para cualesquiera $x, y \in \mathcal{X}$ se cumple que $d(x, y) \geq 0$ (condición de no negatividad).

Es importante mencionar que, en la literatura de análisis de datos como *data mining* y algunos textos de estadística (ver [?] y [?]), a las disimilitudes también se les nombra *distancias*, lo cual es incorrecto en virtud de la definición dada en la primera pregunta. Lo que sí ocurre es que toda distancia (o métrica) es una disimilitud, pero no toda disimilitud es una distancia.

En [?] se menciona que las disimilitudes usuales de acuerdo al tipo de dato que se esté trabajando son las siguientes:

- **Tipo nominal:** La disimilitud más sencilla es

$$d(x, y) = \begin{cases} 0 & \text{si } x = y \\ 1 & \text{si } x \neq y \end{cases}$$

- **Tipo ordinal:** La disimilitud más sencilla es

$$d(x, y) = \frac{|x - y|}{n - 1}$$

donde suponemos que hay n datos, y a cada uno de ellos se les ha asignado un valor entero de 0 a $n - 1$.

- **Tipo intervalo:** La disimilitud más sencilla es

$$d(x, y) = \|x - y\|$$

donde $\|\cdot\|$ denota la distancia euclidiana usual.

3.2.1. Distancias y disimilitudes entre datos

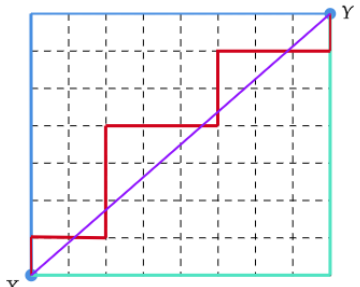
En el Cuadro 1 se muestra cómo se calculan distancias y disimilitudes entre individuos con datos numéricos, binarios, categóricos y mixtos.

Cuadro 1: Distancias y disimilitudes entre distintos tipos de datos

Nombre	Descripción	Tipo de datos	Fórmula y explicación
Distancia euclidiana	<p>Es la distancia usual entre dos puntos en el espacio Euclidiano \mathcal{X}. Cumple las siguientes propiedades:</p> <ol style="list-style-type: none"> 1. $d(X, Y) \geq 0$ para todo $X, Y \in \mathcal{X}$ y $d(X, Y) = 0$ si y sólo si $X = Y$ 2. $d(X, Y) = d(Y, X)$ para todo $X, Y \in \mathcal{X}$ (Simetría) 3. $d(X, Z) \leq d(X, Y) + d(Y, Z)$ para todo $X, Y, Z \in \mathcal{X}$ (Desigualdad del triángulo) 	Numéricos	<p>Sea \mathcal{X} el espacio euclidiano de dimensión n y sean $X, Y \in \mathcal{X}$ dos puntos en el espacio con coordenadas $X = (x_1, \dots, x_n)$ y $Y = (y_1, \dots, y_n)$. Entonces su distancia euclidiana esta dada por:</p> $d(X, Y) = \sqrt{\sum_{i=1}^n (x_i - y_i)^2}$
Distancia de Minkowski	Es una generalización de la distancia euclidiana. Se llama así en honor al matemático Hermann Minkowski.	Numéricos	<p>Sean $X, Y \in \mathcal{X}$ dos puntos en el espacio con coordenadas $X = (x_1, \dots, x_n)$ y $Y = (y_1, \dots, y_n)$ y sea p un entero. Entonces su distancia Minkowski esta dada por:</p> $d(X, Y) = \left(\sum_{i=1}^n x_i - y_i ^p \right)^{\frac{1}{p}}$ <p>Para $p \geq 1$ esta distancia es una métrica. Si $p < 1$ no se satisface la desigualdad del triángulo y por lo tanto no es una métrica. Si $p = 2$ es la distancia euclidiana y si $p = 1$ es la distancia de Manhattan(o city block).</p>

continúa ...

... continuación

Nombre	Descripción	Tipo de datos	Fórmula y explicación
Distancia <i>City block</i>	<p>También se conoce como la distancia del taxista o distancia de Manhattan. Esta distancia no es única en el sentido de que hay más de un camino de distancia mínima para llegar de un punto a otro. Se le conoce como distancia del taxista o distancia de Manhattan porque hace referencia a la forma en que un automóvil puede ir de un punto a otro en un diseño cuadrícula de las calles de Manhattan. En la figura siguiente, podemos la distancia City block entre los puntos X, Y esta representada con los colores rojo, azul y verde, los tres caminos tienen la misma longitud, mientras que la distancia en morado es la distancia euclidiana entre los puntos.</p> 	Numéricos	<p>Sean $X, Y \in \mathcal{X}$ dos puntos en el espacio con coordenadas $X = (x_1, \dots, x_n)$ y $Y = (y_1, \dots, y_n)$. Entonces su distancia city block esta dada por:</p> $d_1(X, Y) = \ X - Y\ = \sum_{i=1}^n x_i - y_i $ <p>En el caso de la figura, la distancia de city block (en color azul, rojo o verde) sería 15 si consideramos que cada cuadrito tiene longitud 1.</p>

continúa ...

... continuación

Nombre	Descripción	Tipo de datos	Fórmula y explicación
Disimilitud calculando <i>Simple Matching</i>	El coeficiente de concordancia simple, llamado coeficiente <i>simple matching</i> (SMC) o <i>coeficiente de Rand</i> se trata de la razón de coincidencias respecto al número total de valores. Se ofrece una ponderación igual a las coincidencias y a las no coincidencias. Toma valores entre 0 y 1. La disimilitud calculada vía simple matching o <i>distancia simple matching</i> , que mide la disimilitud entre las muestras, se calcula restando el SMC a 1.	Binarios	<p>Consideremos A y B dos objetos con n atributos binarios, esto es, cada atributo vale 0 o 1. El número total para cada combinación de atributos para A y B se obtiene como sigue:</p> <ul style="list-style-type: none"> ■ Denotamos M_{11} al número total de atributos donde A y B tienen un valor de 1. ■ Denotamos M_{01} al número total de atributos donde A vale 0 y B vale 1. ■ Denotamos M_{10} al número total de atributos donde A vale 1 y B vale 0. ■ Denotamos M_{00} al número total de atributos donde A y B tienen un valor de 0. <p>Entonces $M_{00} + M_{01} + M_{10} + M_{11} = n$. La distancia <i>simple matching</i> SMD entre A y B se define como</p> $\text{SMD} = 1 - \text{SMC},$ <p>donde SMC es el coeficiente <i>simple matching</i> dado por</p> $\text{SMC} = \frac{M_{00} + M_{11}}{M_{00} + M_{01} + M_{10} + M_{11}}.$

continúa ...

... continuación

Nombre	Descripción	Tipo de datos	Fórmula y explicación
Disimilitud calculando el coeficiente de Jaccard	<p>El coeficiente de Jaccard fue presentado como <i>coefficient de communauté</i> por Paul Jaccard en 1912 y posteriormente fue formulado de manera independiente por T. Tanimoto en 1958, y toma valores entre 0 y 1. Se trata de un índice en el que no se toman en cuenta las ausencias conjuntas. Se ofrece una ponderación igual a las coincidencias y a las no coincidencias. Se conoce también como <i>razón de similaridad</i>.</p> <p>La disimilitud de Jaccard (habitualmente conocida como <i>distancia de Jaccard</i>) se calcula restando el coeficiente de Jaccard a 1. La distancia de Jaccard suele usarse para calcular una matriz $n \times n$ para la agrupación y el escalamiento multidimensional de n conjuntos de muestras.</p>	Binarios	<p>Consideremos A y B dos objetos con n atributos binarios, esto es, cada atributo vale 0 o 1. El número total para cada combinación de atributos para A y B se obtiene como sigue:</p> <ul style="list-style-type: none"> ■ Denotamos M_{11} al número total de atributos donde A y B tienen un valor de 1. ■ Denotamos M_{01} al número total de atributos donde A vale 0 y B vale 1. ■ Denotamos M_{10} al número total de atributos donde A vale 1 y B vale 0. ■ Denotamos M_{00} al número total de atributos donde A y B tienen un valor de 0. <p>Entonces $M_{00} + M_{01} + M_{10} + M_{11} = n$. La distancia de Jaccard d_J se define como</p> $d_J = \frac{M_{01} + M_{10}}{M_{01} + M_{10} + M_{11}}.$ <p>Se cumple que $d_J = 1 - J$ donde J es el índice (o coeficiente) de Jaccard dado por</p> $J = \frac{M_{11}}{M_{01} + M_{10} + M_{11}}.$ <p>También, para fines del diseño, si A y B son conjuntos vacíos, entonces $d_J = 0$ (o equivalentemente, $J(A, B) = 1$).</p>

continúa ...

... continuación

Nombre	Descripción	Tipo de datos	Fórmula y explicación
Disimilitud calculando el coeficiente de Czekanowski	<p>El coeficiente Czekanowski es uno en el que no se toman en cuenta las ausencias conjuntas y donde las coincidencias se ponderan doblemente en datos binarios. También se conoce como coeficiente de Dice, coeficiente de Sørensen o coeficiente de Sørensen–Dice (quienes los derivaron de manera independiente en 1948). Fue desarrollado en 1913 por Jan Czekanowski para establecer relaciones entre dialectos y lenguas, pero realmente se puede aplicar a cualquier comparación entre individuos calificados por múltiples atributos, por ejemplo, razas, especies de plantas o de animales, biocenosis, biotopos y hábitats, culturas, etc. La calificación puede ser cualitativa o cuantitativa y se basa en la comparación atributo por atributo de cada par de individuos de una colección. Este índice toma valores entre 0 y 1.</p> <p>La disimilitud asociada al coeficiente de Czekanowski se obtiene restando dicho índice a 1. Contrario a la distancia de Jaccard, esta disimilitud no cumple la desigualdad del triángulo y por ello no es una distancia verdadera.</p>	Binarios	<p>Consideremos A y B dos objetos con n atributos binarios, esto es, cada atributo vale 0 o 1. El número total para cada combinación de atributos para A y B se obtiene como sigue:</p> <ul style="list-style-type: none"> ■ Denotamos M_{11} al número total de atributos donde A y B tienen un valor de 1. ■ Denotamos M_{01} al número total de atributos donde A vale 0 y B vale 1. ■ Denotamos M_{10} al número total de atributos donde A vale 1 y B vale 0. ■ Denotamos M_{00} al número total de atributos donde A y B tienen un valor de 0. <p>La disimilitud asociada al coeficiente de Czekanowski se calcula como</p> $d_{Cz} = 1 - \frac{2M_{11}}{2M_{11} + M_{10} + M_{01}}$ <p>De manera equivalente, al considerar la formulación dada por Søren y Dice, la disimilitud anterior se puede formular como</p> $d_{Cz} = 1 - \frac{2 A \cap B }{ A + B }$ <p>donde \cdot denota la cardinalidad de los respectivos conjuntos.</p>

continúa ...

... continuación

Nombre	Descripción	Tipo de datos	Fórmula y explicación
Disimilitud de 1- coincidencias de Sneath	Medida de disimilitud propuesta en 1958 por Sneath y Sokal para variables binarias. Cumple las propiedades de simetría entre a y d (atributos en que las unidades coinciden), mientras que su rango está entre 0 y 1.	Binarios	$d_{ij} = 1 - c_{ij}$ <p>con</p> $c_{ij} = \frac{a + d}{a + b + c + d}$ <p>donde a y d son los atributos en que las unidades coinciden mientras el denominador es la suma de todos los valores de la tabla de contingencia.</p>
Distancia de Gower	El coeficiente de similitud de Gower propuesto por Gower en 1971 permite la manipulación simultánea de variables cuantitativas y cualitativas en una base de datos, mediante la aplicación de este coeficiente se logra hallar la similitud entre individuos a los cuales se les han medido una serie de características en común. Una similaridad alta, es decir cercana a 1, indicara gran homogeneidad entre los individuos; por el contrario, una similaridad cercana a cero indica que los individuos son diferentes.	Mixtos: variables cuantitativas y cualitativas	<p>donde</p> $s_{ij} = \frac{\sum_{h=1}^{p_1} \left(1 - \frac{ x_{ih} - x_{jh} }{G_h} \right) + a + \alpha}{p_1 + p_2 - d + p_3}$ <p>con p_1 es el número de variables cuantitativas, a y d son el número de coincidencias en 1 (presencia de la característica) y el número de coincidencias en 0 (ausencia de la característica) respectivamente para las p_2 variables binarias, α es el número de coincidencias para las p_3 variables cualitativas y G_h es el rango de la h-ésima variable cuantitativa.</p>

3.3. Agrupamiento (Clústers)

Existen varias familias de métodos de *agrupamiento* disponibles en la literatura (Legendre y Legendre 2012): **algoritmos secuenciales** o **algoritmos simultáneos**, **aglomerativos** o **divisivos**, **monotéticos** o **politéticos**, **métodos jerárquicos** o **métodos no jerárquicos**, **métodos probabilísticos** frente a **métodos no probabilísticos**. Entre estas categorías, los métodos de *agrupación jerárquica de Ward* y de *partición de k-means* son los más comunes en los estudios sobre el microbioma.

Los métodos jerárquicos aglomerativos, dan origen a un gráfico llamado *dendrogramas*. Estos métodos son iterativos y en cada paso debe recalcularse la matriz de distancias, lo cual para bases de datos muy grandes suele consumir mucho tiempo de cómputo.

En este capítulo se ilustran varios métodos de clustering diferentes, incluyendo:

- *Single linkage agglomerative clustering* o método de la liga sencilla o del vecino más cercano.
- *Complete linkage agglomerative clustering* o método de la liga completa o del vecino más lejano.
- *Average linkage agglomerative clustering* o método de la liga promedio.
- *Ward's minimum variance clustering* o método Ward.

El conjunto de datos de ratones **Vdr**—/— con muestras fecales con el que se ha estado trabajando es el que se utilizará para ilustrar el análisis de conglomerados. Aquí, la distancia **Bray-Curtis** se utilizará para ilustrar la clasificación de las muestras. También se aplican otras distancias.

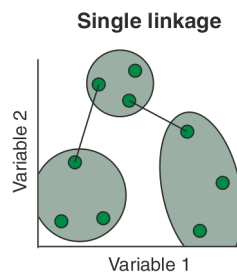
Cargamos el paquete **vegan**. Primero vamos a *normalizar la tabla de abundancia* utilizando la función `decostand()` y calcular las disimilitudes *Bray-Curtis* entre todos los pares de muestras utilizando la función `vegdist()` del paquete **vegan**.

```
library("vegan")
abund_table_norm <- decostand(abund_table, "normalize")
bc_dist <- vegdist(abund_table_norm , method = "bray")
```

3.3.1. Clustering aglomerativo de la liga sencilla

“*Aglomerar*” significa reunir en un clúster; la agrupación aglomerativa de la liga sencilla también se denomina *clasificación de vecinos más cercanos*. En cada paso, la agrupación combina dos muestras que contienen las distancias más cortas entre pares (o la mayor similitud), es decir, se toma la mínima de todas las posibles distancias entre los objetos que pertenecen a distintos conglomerados:

$$d_{AB} = \min_{i \in A, j \in B} (d_{ij}).$$



El resultado de la agrupación puede visualizarse como un *dendrograma*. El inconveniente del dendrograma resultante de un clustering de liga sencilla a menudo muestra el encadenamiento de las muestras. Los clústeres formados se ven forzados a unirse debido a que los elementos individuales están cerca unos de otros, mientras

que muchos elementos de cada clúster pueden estar muy alejados entre sí. Otro inconveniente relevante del clustering de la liga sencilla es que podría ser difícil de interpretar en términos de particiones de los datos. Es una desventaja real del clustering de la liga sencilla porque estamos interesados en las particiones de los datos. Este método se tiende a desempeñar bien cuando hay grupos de forma elongada, conocidos como tipo cadena.

```
cluster_single <- hclust (bc_dist, method = 'single')
plot(cluster_single)
```

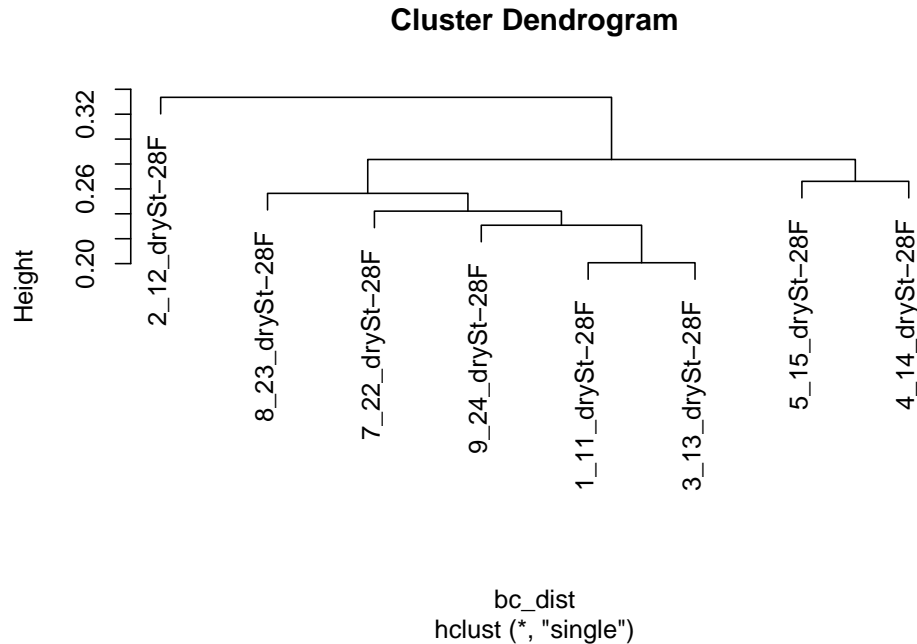
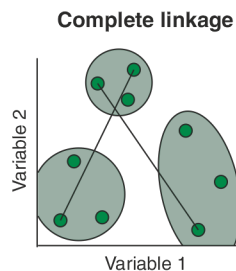


Figura 18: Cluster con el método single.

3.3.2. Clustering aglomerativo de la liga completa

El clustering de la liga completa también se conoce como *clustering del vecino más lejano*. Permite que una muestra (o un grupo) se aglomere con otra muestra (o grupo) sólo a la distancia más lejana entre sí. Así, todos los miembros de ambos grupos están vinculados.

$$d_{AB} = \max_{i \in A, j \in B} (d_{ij}).$$



La agrupación de la liga completa evita el inconveniente de encadenar las muestras por el método de la liga simple. El encadenamiento completo tiende a encontrar muchos pequeños grupos separados. Este método se desempeña bien cuando los conglomerados son de forma circular.

```
cluster_completo <- hclust (bc_dist, method = 'complete')
plot(cluster_completo)
```

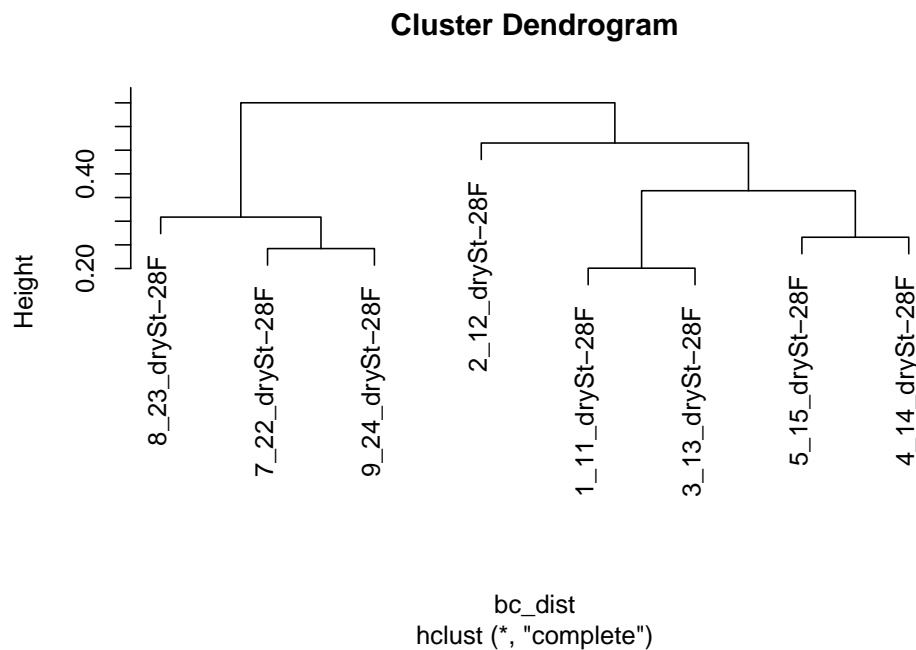
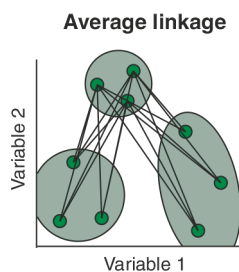


Figura 19: Cluster con el método complete.

3.3.3. Clustering aglomerativo de la liga promedio

La agrupación de liga promedio permite agrupar una muestra en un clúster en la media de las distancias entre esta muestra y todos los miembros del clúster. Los dos clusters se unen a la *media de las distancias entre todos los miembros de un clúster y todos los miembros del otro*.

$$d_{AB} = \frac{1}{n_A n_B} \sum_{i \in A} \sum_{j \in B} (d_{ij}).$$



El clúster resultante del dendrograma tiene un aspecto intermedio entre una agrupación de enlace simple y una completa.

```
cluster_average <- hclust (bc_dist, method = 'average')
plot(cluster_average)
```

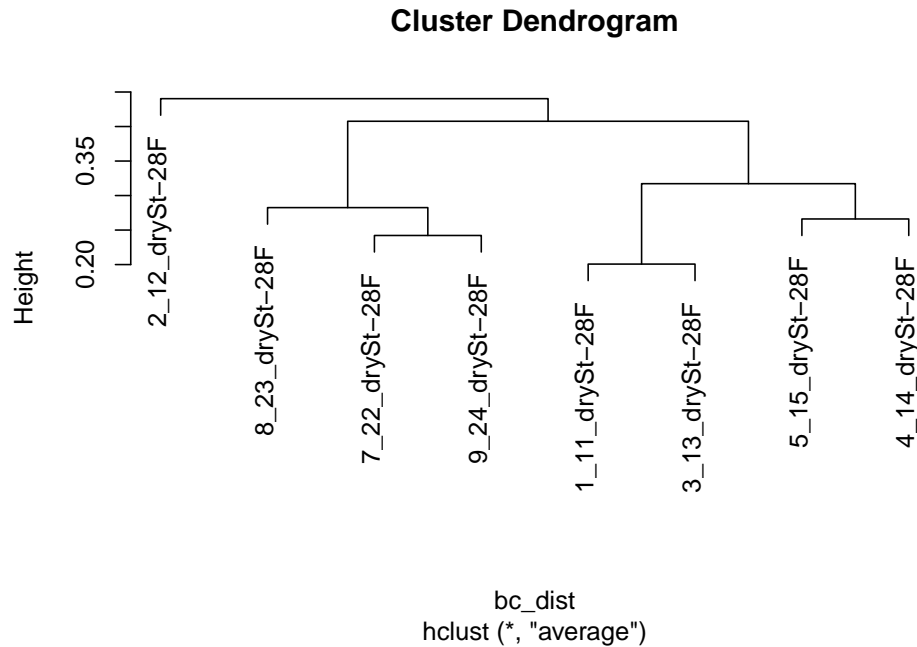
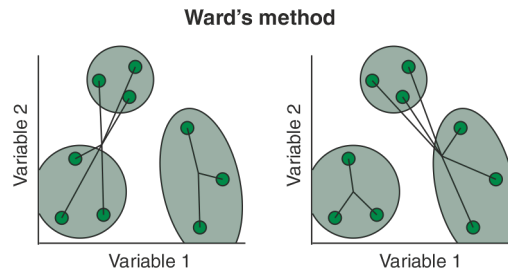


Figura 20: Cluster con el método average.

3.3.4. Agrupación de varianza mínima de Ward

La agrupación de varianza mínima de Ward (también conocida como agrupación de Ward) presentada originalmente por Ward (1963) se basa en el criterio del *modelo lineal de mínimos cuadrados*. Este método minimiza las sumas de las distancias al cuadrado (es decir, el error al cuadrado de ANOVA) entre las muestras. Básicamente, considera el análisis de conglomerados como un análisis de la varianza, en lugar de utilizar métricas de distancia o medidas de asociación. Este método es más apropiado para las variables cuantitativas, y no variables binarias. El clustering de Ward se implementa mediante la búsqueda del par de clusters en cada paso que conduce al mínimo incremento de la varianza total dentro del cluster después de la fusión. Este incremento es una distancia cuadrada ponderada entre los centros de los clusters. Aunque las distancias iniciales de los clusters se definen como la distancia euclidiana al cuadrado entre puntos en el clustering de Ward, otros métodos de distancia pueden producir resultados significativos.




```
cluster_ward <- hclust (bc_dist, method = 'ward.D2')
plot(cluster_ward)
```

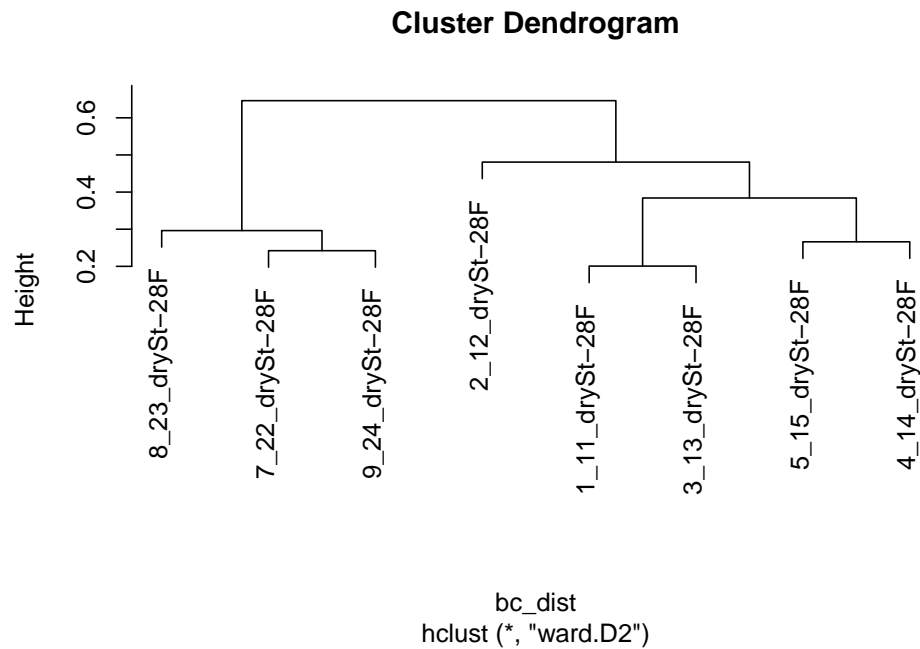


Figura 21: Cluster con el método Ward.

```
# Correrlo en consola para visualizarlo mejor
par (mfrow = c(2,2))
plot(cluster_single)
plot(cluster_complete)
plot(cluster_average)
plot(cluster_ward)
```

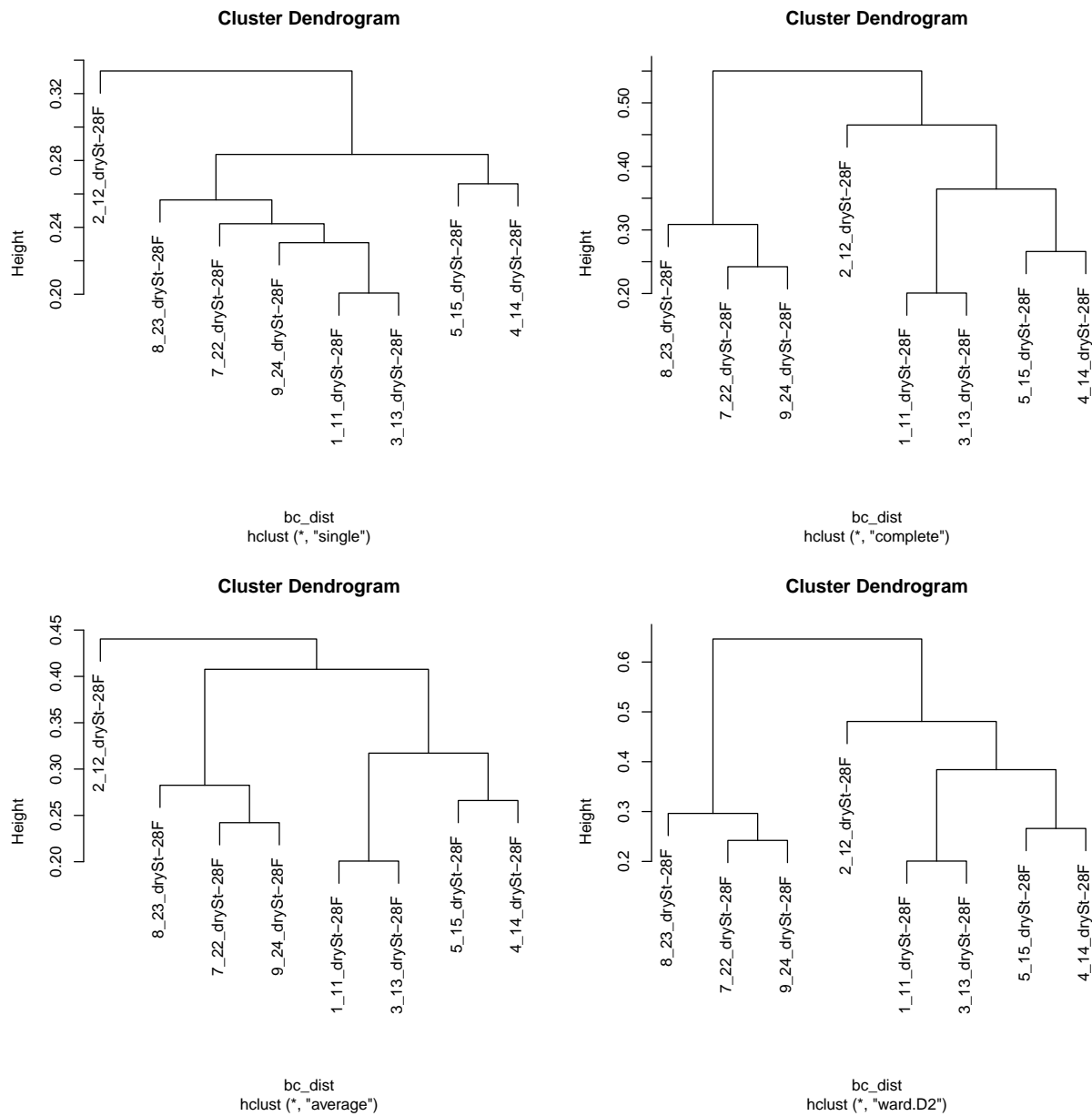


Figura 22: Comparación de los clusters con los 4 métodos.

```
par (mfrow = c(1,1))
```

La comparación entre estos cuatro dedrogramas muestra que la vinculación completa y Ward generan los mismos clusters de partición y tienen el mejor rendimiento en términos de partición de datos en la dirección de nuestro interés (las muestras 11, 12 13, 14 y 15 son de ratones Vdr $^{-/-}$, y 22, 23 y 24 de ratones de tipo salvaje).

3.3.5. Mismos gráficos usando el paquete factoextra

```
library(factoextra)
fviz_dist(dist.obj = bc_dist, lab_size = 8)
```

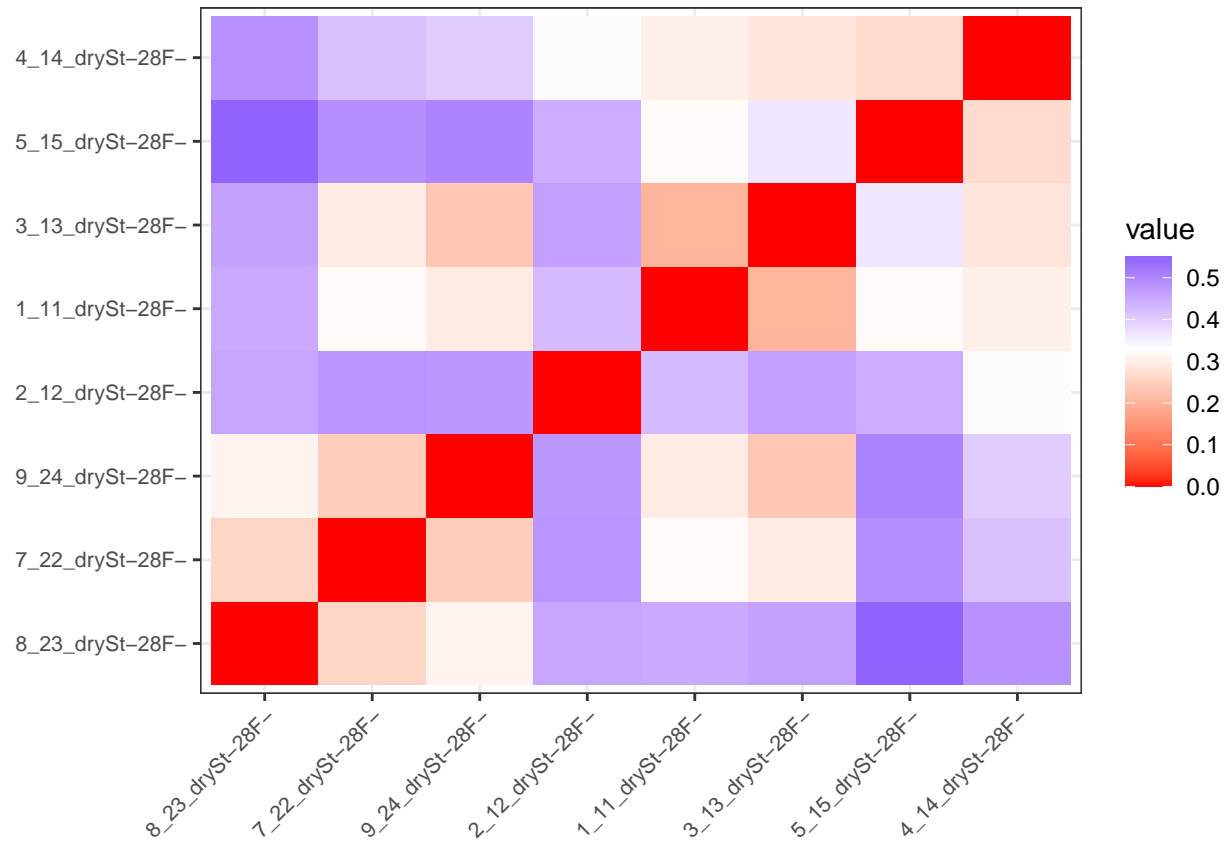


Figura 23: Heatmap de la matriz de distancias

```
set.seed(12345)
hc_single <- hclust(d=bc_dist, method = "single")
hc_average <- hclust(d=bc_dist, method = "average")
hc_complete <- hclust(d=bc_dist, method = "complete")
hc_ward <- hclust(d=bc_dist, method = "ward.D2")
```

```
fviz_dend(x = hc_average, k=2,
  cex = 0.7,
  main = "",
  xlab = "Samples",
  ylab = "Distance",
  sub = "",
  horiz = TRUE) +
  geom_hline(yintercept = 0.41, linetype = "dashed")
```

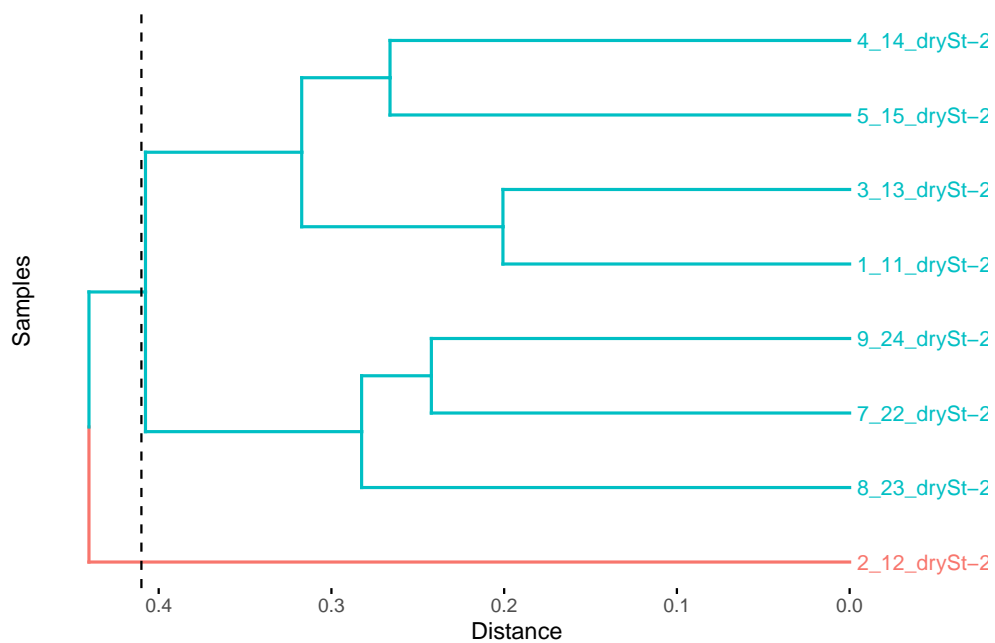


Figura 24: con el método del promedio.

```
fviz_dend(x = hc_ward, k=2,
  cex = 0.7,
  main = "",
  xlab = "Samples",
  ylab = "Distance",
  sub = "",
  horiz = TRUE)+
geom_hline(yintercept = 0.5, linetype = "dashed")
```

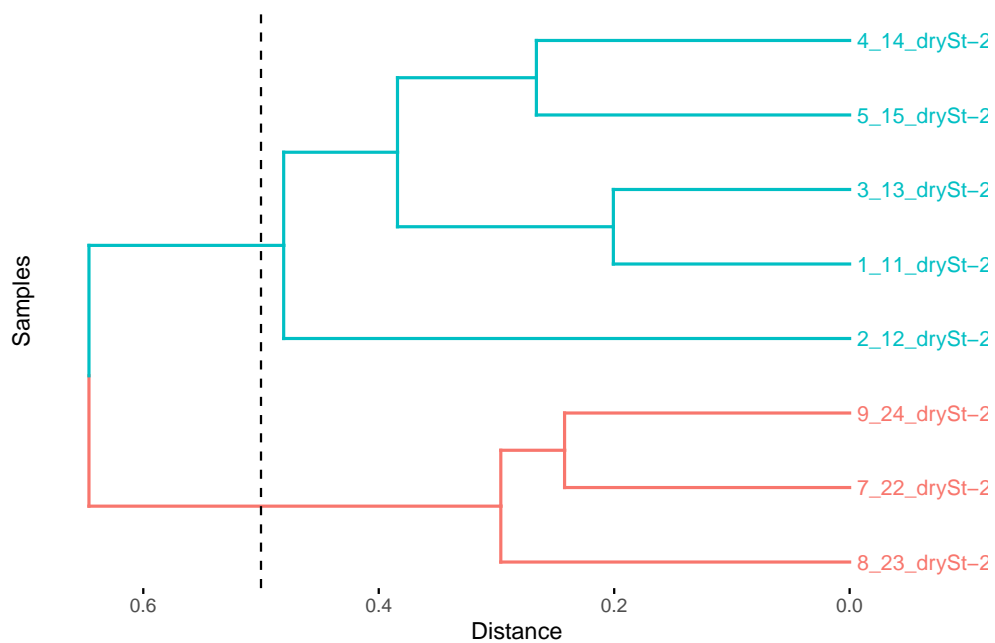


Figura 25: con el método Ward.

```
fviz_dend(x = hc_single, k=2,
  cex = 0.7,
  main = "",
  xlab = "Samples",
  ylab = "Distance",
  sub = "",
  horiz = TRUE)+
  geom_hline(yintercept = 0.3, linetype = "dashed")
```

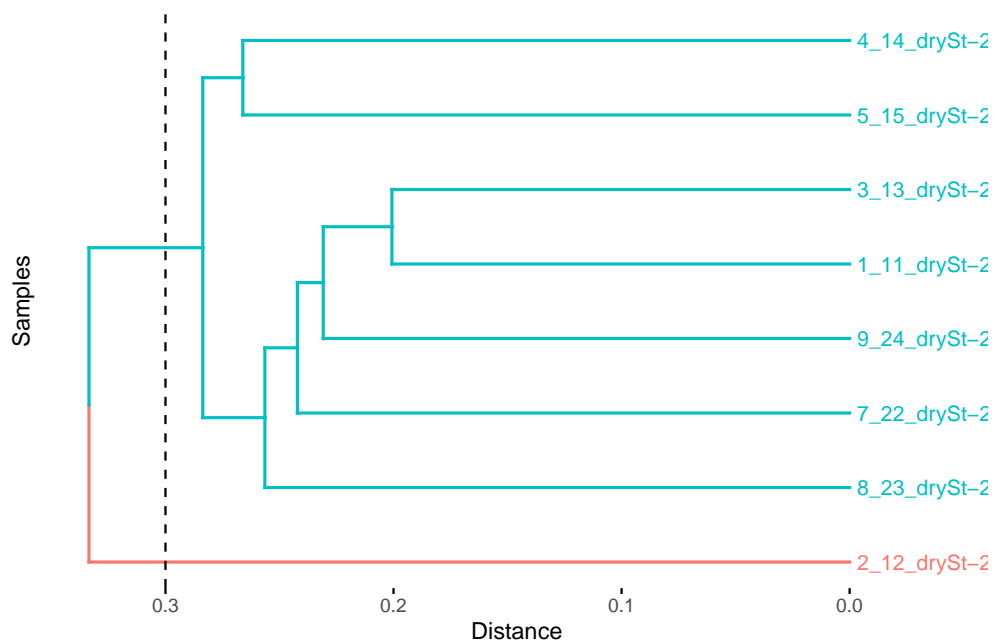


Figura 26: con el método de la liga más sencilla.

```
fviz_dend(x = hc_complete, k=2,
  cex = 0.7,
  main = "",
  xlab = "Samples",
  ylab = "Distance",
  sub = "",horiz = TRUE)+
  geom_hline(yintercept = 0.5, linetype = "dashed")
```

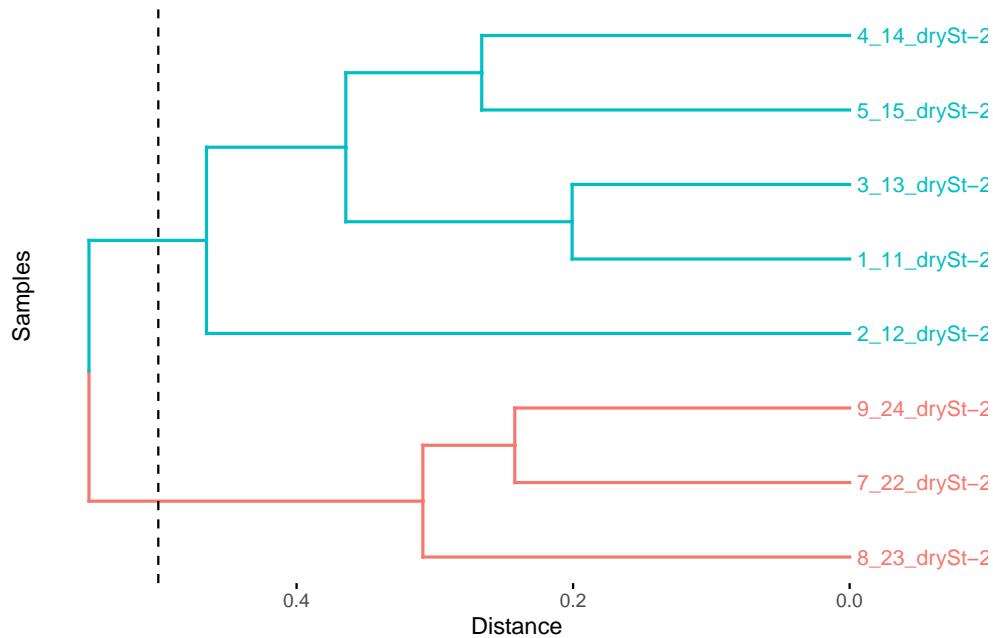


Figura 27: con el método de la liga completa.

4. Ordenación

El principal objetivo de la ordenación fue la “exploración” con la introducción de un análisis canónico de correspondencia (CCA), la ordenación ha ido más allá de meramente un análisis “exploratorio” y se ha convertido en una prueba de hipótesis.

La estructura de los datos de dos variables suele revelarse mediante un gráfico de dispersión de las muestras. Los datos multivariantes del microbioma son multidimensionales, generalmente tienen más de dos variables. El conjunto de datos del microbioma puede verse como una colección de muestras (sujetos) situadas en un espacio en el que *cada variable o especie (o OTU/taxa) define una dimensión*. Por lo tanto, *hay tantas dimensiones como variables o especies (u OTU/taxa)*. Por ejemplo, en nuestro conjunto de datos fecales de ratón Vdr^{-/-} hay 8 muestras y 248 géneros. Por tanto, la dimensión de los datos es 248. Para un conjunto de datos con n variables, el número de gráficos de dispersión que hay que dibujar sería $n(n-1)/2$. En nuestro caso, los 248 géneros necesitarán $(248 * 247)/2 = 30628$ gráficos de dispersión. Un número tan grande de gráficos de dispersión no es informativo para conocer la estructura de los datos; también es tedioso trabajar con ellos.

La ordenación trata principalmente de representar las relaciones entre las muestras y las especies (o OTUs/taxa) lo más fielmente posible en un espacio de baja dimensión (Gauch 1982a, b). Este objetivo es deseable, ya que los datos de la comunidad tienen múltiples dimensiones mezcladas con ruido, las dimensiones bajas pueden representar idealmente y de forma típica interpretaciones importantes e intuitivas de las relaciones especie (o OTUs/taxa)-ambiente. Para un conjunto de datos $n \times p$ que contiene n sujetos y p variables, la estructura de los datos puede revisarse como n sujetos (representados como un grupo de puntos) en el espacio p -dimensional.

El **objetivo principal** de la ordenación es representar múltiples muestras (sujetos) en un número reducido de ejes ortogonales (es decir, independientes), donde el número total de ejes es menor o igual al número de muestras (sujetos). La importancia de los ejes de ordenación disminuye por orden. El **primer eje** de una ordenación explica la mayor parte de la variación en el conjunto de datos, seguido por el segundo eje, luego el

tercero, y así sucesivamente. Los gráficos de ordenación son especialmente útiles para visualizar la similitud entre muestras (sujetos). Por ejemplo, en el contexto de la diversidad beta, las muestras que están más cerca en el espacio de ordenación tienen conjuntos de especies que son más similares entre sí que las muestras que están más alejadas en el espacio de ordenación.

Los métodos de ordenación incluyen *ordenaciones restringidas* y *no restringidas*, dependiendo de si los ejes de ordenación están limitados por factores ambientales (variables). Como su nombre indica, en la **ordenación restringida**, los ejes de ordenación están restringidos por factores ambientales: las posiciones de las muestras en la ordenación están limitadas por las variables del entorno. En la **ordenación no restringida**, los ejes de ordenación no están limitados por factores ambientales. En otras palabras, la ordenación restringida utiliza directamente las variables ambientales en la construcción de la ordenación, mientras que en la ordenación no restringida los ejes no están limitados por factores ambientales; mientras que en la ordenación sin restricciones, las variables ambientales se introducen en los análisis post hoc.

Desde el punto de vista de la *comprobación de hipótesis*, el análisis de ordenación sin restricciones per se es un método principalmente descriptivo y no implica realmente la comprobación de hipótesis en datos multivariados. Aunque implica la formulación de hipótesis sobre la explicación de los ejes mediante variables ambientales, como la función `envfit()` del paquete **vegan**, la ordenación sin restricciones es sencilla. Analiza una matriz de datos; su objetivo es revelar la estructura principal de los datos en un gráfico a partir de un conjunto reducido de ejes ortogonales. Por el contrario, la ordenación restringida es una ordenación “impulsada por la hipótesis”: un método de comprobación de hipótesis, que pone a prueba directamente la hipótesis sobre la influencia de los factores ambientales en la composición de las especies (o OTU/taxa).

La ordenación restringida está relacionada con los modelos lineales multivariantes, de esta manera con las variables “dependientes” (o la comunidad) en el lado izquierdo como respuestas, las “independientes” (o restricciones) en el lado derecho como factores explicados. Así, la ordenación restringida no es simétrica.

En este apartado se tratan las ordenaciones no restringidas más comunes:

- Análisis de componentes principales (PCA).
- Análisis de coordenadas principales (PCoA).
- Escalamiento multidimensional no métrico (NMDS).
- Análisis de correspondencia (AC).

Y las ordenaciones restringidas:

- Análisis de redundancia (RDA).
- Análisis de correspondencia restringido (CCA).
- Análisis restringido de coordenadas principales (CAP)*.

4.1. Ordenaciones NO restringidas

4.1.1. Análisis de componentes principales (PCA)

En cuanto al paquete **vegan**, la *variable ambiental* en nuestro caso es la condición genética (Vdr-/- y WT). **Queremos saber si la deficiencia del gen vdr interpreta la diversidad beta en la composición del género.** Realizamos un PCA para explorar si los cambios en la composición de géneros de las comunidades (diversidades beta) son causados por el factor genético. Con el PCA, las muestras se trazan en base a las abundancias del género A en el eje 1 el género B en el eje 2, el género C en el eje 3, y así sucesivamente hasta que se trazan n muestras en un espacio de muy alta dimensión. Las n muestras crean $(n - 1)$ número de componentes principales (PC):

- PC1: es la primera línea recta que atraviesa el espacio creado por todas estas muestras.
- PC2: es la segunda línea, perpendicular a PC1.

y así la tercera PC3, hasta PC($n - 1$). La importancia de las PC disminuye por orden. **PC1 es la PC más importante** y explica la mayor parte de las variaciones entre todas las muestras. La **PC2** explica la segunda mayor parte de las variaciones, y PC3 explica la tercera más, y así sucesivamente la ($n - 1$) PC explica la menor cantidad de variaciones de las muestras. Varias funciones de R, como `prcomp()` en el paquete preinstalado de estadísticas, `rda()` en el paquete `vegan`, `pca()` en el paquete `labdsv` pueden utilizarse para llevar a cabo el PCA.

En este caso, utilizamos la función `rda()`. Las dos funciones de extensión: `evplot()` (Borcard et al. 2011) y `PCAsignificance()` en el paquete `BiodiversityR` pueden mejorar los gráficos. La función `evplot()` proporciona métodos visuales para decidir la importancia de los ejes de ordenación mediante el uso de el criterio de *Keiser-Guttman* y el modelo de broken-stick. La función `PCAsignificance()` calcula el modelo de broken-stick para los ejes PCA. Cuando se utiliza la función `rda()` para realizar el PCA, al no especificar la matriz de datos ambientales (es decir, la variable de grupo), la función realiza un ordenación PCA. En el análisis de datos del microbioma, los recuentos de abundancia absoluta no son apropiados porque los valores grandes tendrán una influencia demasiado alta en el análisis. Por lo tanto, **necesitamos estandarizar los datos de lectura de abundancia antes del análisis**. En este caso, utilizamos la función `decostand()` con el método total para estandarizar la lectura.

```
# Llamamos a nuestra tabla
abund_table=read.csv(paste0(path,"data/VdrFecalGenusCounts.csv"),row.names=1,check.names=FALSE)
# Transpuesta de la tabla
abund_table<-t(abund_table)
#head(abund_table)
# Tabla de metadatos
meta_table <- data.frame(row.names=rownames(abund_table),
                        t(as.data.frame(strsplit(rownames(abund_table),"_"))))
# Agregamos el factor de grupo
meta_table$Group <- with(meta_table,ifelse(as.factor(X2)%in% c(11,12,13,14,15),
                                           c("Vdr-/-"), c("WT")))
# Estandarizamos la abundancia de los datos
stand_abund_table <- decostand(abund_table, method = "total")
PCA <-rda(stand_abund_table)
PCA
```

```
## Call: rda(X = stand_abund_table)
##
##              Inertia Rank
## Total          0.04082
## Unconstrained 0.04082    7
## Inertia is variance
##
## Eigenvalues for unconstrained axes:
##      PC1      PC2      PC3      PC4      PC5      PC6      PC7
## 0.020288 0.012792 0.003712 0.001730 0.001425 0.000774 0.000098
```

Con el siguiente código podemos pedir los valores de los eigenvalores.

```
eig <- PCA$CA$eig
```

En lenguaje `vegan`, “**Inertia**” es el término general de “**variación**” en los datos. El total de variación de todo el conjunto de datos es de 0.0408196 en este caso, y el primer eje explica el 49.7017233 % de la variación total ($0.02029/0.0408 = 0.4973$). La **variación total** es la suma de las variaciones de cada género en la matriz analizada. Los siguientes códigos R comprueban la variación total:

```
# Variación total
sum(apply(stand_abund_table, 2, var))
```

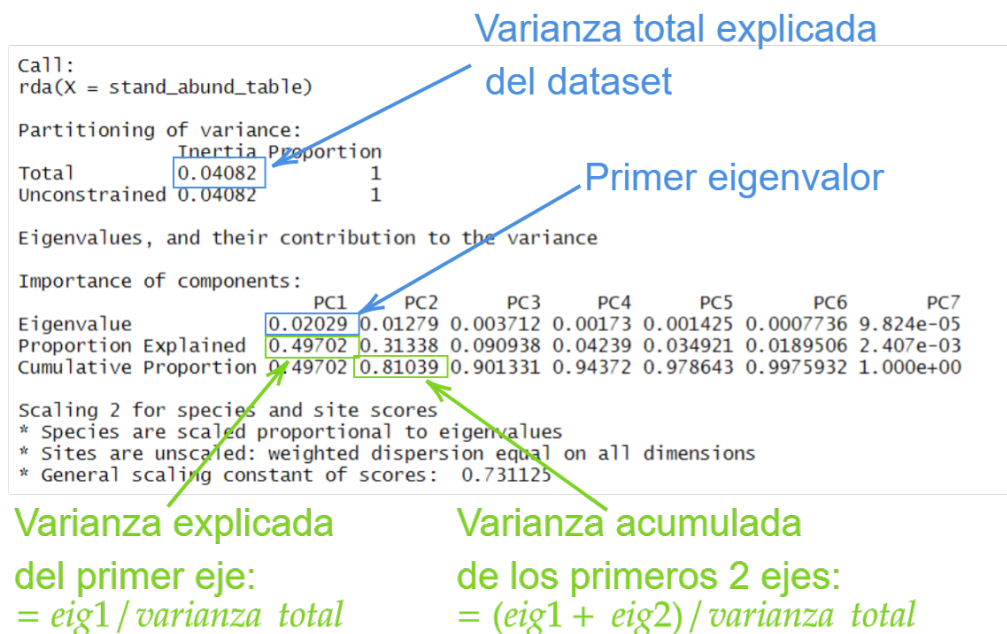
```
## [1] 0.04081957
```

Otra forma de obtener este análisis es llamando la función `summary`.

```
head(summary(PCA))
```

```
##
## Call:
## rda(X = stand_abund_table)
##
## Partitioning of variance:
##           Inertia Proportion
## Total      0.04082      1
## Unconstrained 0.04082      1
##
## Eigenvalues, and their contribution to the variance
##
## Importance of components:
##           PC1      PC2      PC3      PC4      PC5      PC6
## Eigenvalue    0.02029 0.01279 0.003712 0.00173 0.001425 0.0007736
## Proportion Explained 0.49702 0.31338 0.090938 0.04239 0.034921 0.0189506
## Cumulative Proportion 0.49702 0.81039 0.901331 0.94372 0.978643 0.9975932
##           PC7
## Eigenvalue    9.824e-05
## Proportion Explained 2.407e-03
## Cumulative Proportion 1.000e+00
##
## 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: 0.731125
##
##
## Species scores
##
##           PC1      PC2      PC3      PC4
## Tannerella    -0.1460060 0.059004 -0.1581134 0.0263299
## Lactococcus     0.3347414 0.268815 0.0199084 -0.0094492
## Lactobacillus    0.2884642 -0.241439 -0.0576255 0.0148015
## Lactobacillus::Lactococcus 0.0047630 0.002077 -0.0002720 -0.0008367
## Parasutterella 0.0002188 -0.001427 -0.0000296 -0.0008465
## Helicobacter   -0.0390189 0.005225 -0.0389887 -0.0101868
## ....
##           PC5      PC6
## Tannerella    -0.0074351 -0.0108844
## Lactococcus     0.0133844 -0.0002257
## Lactobacillus    0.0420286 0.0006774
## Lactobacillus::Lactococcus 0.0037431 -0.0004606
## Parasutterella -0.0009601 0.0001548
```

```
## Helicobacter          -0.0106939 -0.0092049
## ....
##
##
## Site scores (weighted sums of species scores)
##
##          PC1      PC2      PC3      PC4      PC5      PC6
## 5_15_drySt-28F -0.35858  0.04972 -0.443749 -0.2395 -0.12631 -0.21700
## 1_11_drySt-28F -0.01508  0.25950  0.308436 -0.4460  0.28194  0.14170
## 2_12_drySt-28F -0.28343 -0.36080  0.318538  0.1051  0.10647 -0.28342
## 3_13_drySt-28F  0.02043  0.37303 -0.007325  0.1292 -0.26490  0.07505
## 4_14_drySt-28F -0.24988 -0.01016  0.007908  0.3703  0.03704  0.44259
## 7_22_drySt-28F  0.31464 -0.02094 -0.313787  0.1650  0.47544 -0.09180
## 8_23_drySt-28F  0.28787 -0.42150 -0.065406 -0.2454 -0.26694  0.26212
## 9_24_drySt-28F  0.28403  0.13115  0.195385  0.1613 -0.24274 -0.32924
```



También podemos obtener cuales variables (species) tienen la mayor correlación absoluta al primer y segundo eje:

```
head(sort(abs(PCA$CA$v[,1]),decreasing = TRUE))
```

```
## Lactococcus Lactobacillus Tannerella Clostridium Bacteroides
## 0.6494296 0.5596474 0.2832653 0.2751724 0.2046105
## Allobaculum
## 0.1487113
```

```
head(sort(abs(PCA$CA$v[,2]),decreasing = TRUE))
```

```
## Lactococcus Lactobacillus Allobaculum Akkermansia Bacteroides
```

```
##      0.6567934      0.5899062      0.3501612      0.1680896      0.1457728
##      Tannerella
##      0.1441641
```

A continuación, vamos a dibujar los diagramas utilizando la función `biplot()`. La opción de visualización “species” es la etiqueta del paquete `vegan` para OTUs/taxa. Por defecto es “sites” (etiqueta para muestras).

```
biplot(PCA, display = 'species')
```

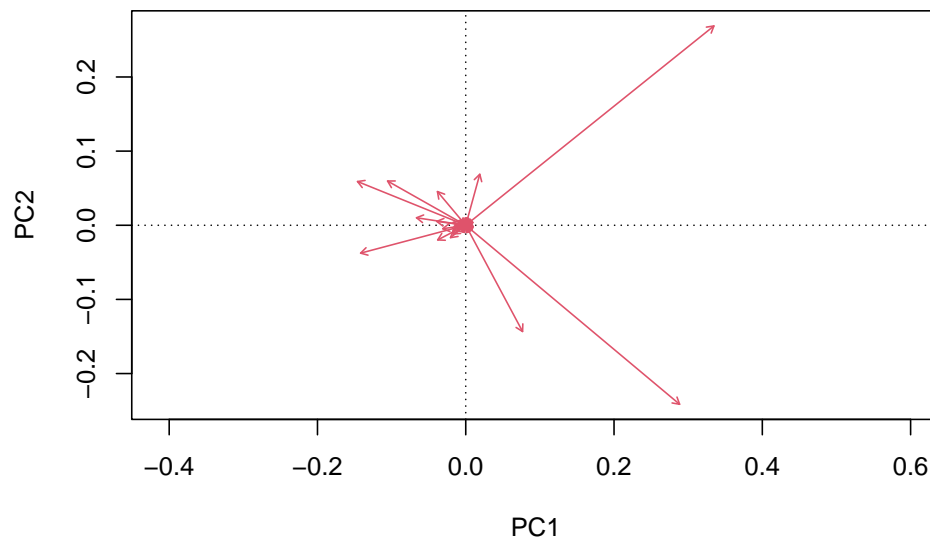


Figura 28: Biplot de las primeras dos componentes principales de los datos de materia fecal de ratones Vdr^{-/-}

Otros argumentos que uno puede incluir en el biplot son por ejemplo mostrar también los `sites` y agregar etiquetas a los datos, (Ver `vegan` pag. 37)

```
#Mismo biplot con etiquetas
biplot(PCA, display = c('sites','species'), type=c("text","points"))
```

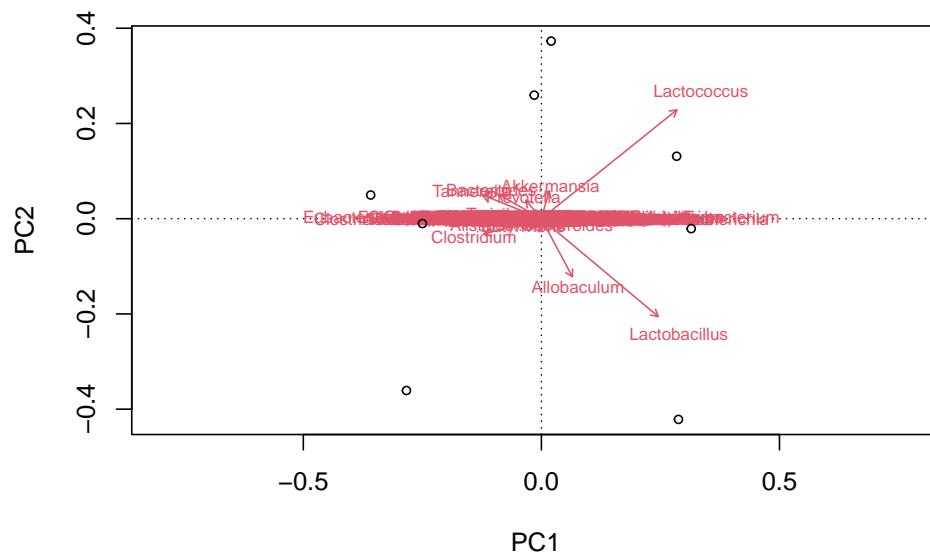


Figura 29: Biplot con etiquetas

Los diagramas anteriores trazados por `biplot()` sólo dibujan flechas para el género, lo que no es informativo. El diagrama más informativo es utilizar la función `ordiplot()` para dibujar tanto el género como las puntuaciones de la muestra como centroides, como se muestra a continuación:

```
ordiplot(PCA, display = "sites", type = "text")
```

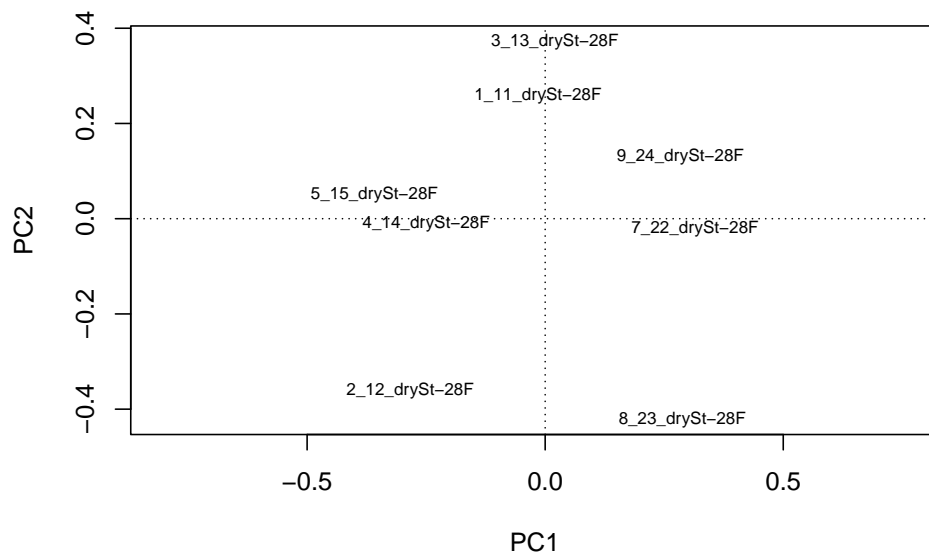


Figura 30: Ordiploot de las primeras dos componentes principales de los datos de materia fecal de ratones Vdr-/- con etiquetas

En los argumentos anteriores, `type="text"` o `t` añade etiquetas de texto a la figura (la configuración por defecto sólo añade puntos). Como alternativa, podemos utilizar la función `cleanplot.pca()` escrita por François Gillet & Daniel Borcard para intentar dibujar los resultados del PCA en dos diagramas y diferenciarlos por su escala. Esta función sirve para dibujar dos biplots (escalado 1 y escalado 2) a partir de un objeto de clase "rda" en PCA o RDA resultado de la función `rda()` de `vegan`. Primero ejecutamos la función `cleanplot.pca()` como se indica a continuación.

```
"cleanplot.pca" <- function(res.pca, ax1=1, ax2=2, point=FALSE,
                             ahead=0.07, cex=0.5)
{
  # A function to draw two biplots (scaling 1 and scaling 2) from an object
  # of class "rda" (PCA or RDA result from vegan's rda() function)
  #
  # License: GPL-2
  # Authors: Francois Gillet & Daniel Borcard, 24 August 2012

  require("vegan")

  par(mfrow=c(1,2))
  p <- length(res.pca$CA$eig)

  # Scaling 1: "species" scores scaled to relative eigenvalues
  sit.sc1 <- scores(res.pca, display="wa", scaling=1, choices=c(1:p))
  spe.sc1 <- scores(res.pca, display="sp", scaling=1, choices=c(1:p))
  plot(res.pca, choices=c(ax1, ax2), display=c("wa", "sp"), type="n",
       main="PCA - scaling 1", scaling=1)
  if (point)
  {
```

```

points(sit.sc1[,ax1], sit.sc1[,ax2], pch=20)
text(res.pca, display="wa", choices=c(ax1, ax2), cex=cex, pos=1, scaling=1)
}
else
{
  text(res.pca, display="wa", choices=c(ax1, ax2), cex=cex, scaling=1)
}
text(res.pca, display="sp", choices=c(ax1, ax2), cex=cex, pos=1, col="blue", scaling=1)
arrows(0, 0, spe.sc1[,ax1], spe.sc1[,ax2], length=ahead, angle=20, col="red")
pcacircle(res.pca)

# Scaling 2: site scores scaled to relative eigenvalues
sit.sc2 <- scores(res.pca, display="wa", choices=c(1:p))
spe.sc2 <- scores(res.pca, display="sp", choices=c(1:p))
plot(res.pca, choices=c(ax1,ax2), display=c("wa","sp"), type="n",
      main="PCA - scaling 2")
if (point) {
  points(sit.sc2[,ax1], sit.sc2[,ax2], pch=20)
  text(res.pca, display="wa", choices=c(ax1 ,ax2), cex=cex, pos=1)
}
else
{
  text(res.pca, display="wa", choices=c(ax1, ax2), cex=cex)
}
text(res.pca, display="sp", choices=c(ax1, ax2), cex=cex, pos=1, col="blue")
arrows(0, 0, spe.sc2[,ax1], spe.sc2[,ax2], length=ahead, angle=20, col="red")
}

"pcacircle" <- function (pca)
{
  # Draws a circle of equilibrium contribution on a PCA plot
  # generated from a vegan analysis.
  # vegan uses special constants for its outputs, hence
  # the 'const' value below.

  eigenv <- pca$CA$eig
  p <- length(eigenv)
  n <- nrow(pca$CA$u)
  tot <- sum(eigenv)
  const <- ((n - 1) * tot)^0.25
  radius <- (2/p)^0.5
  radius <- radius * const
  symbols(0, 0, circles=radius, inches=FALSE, add=TRUE, fg=2)
}

```

```
cleanplot.pca(PCA)
```

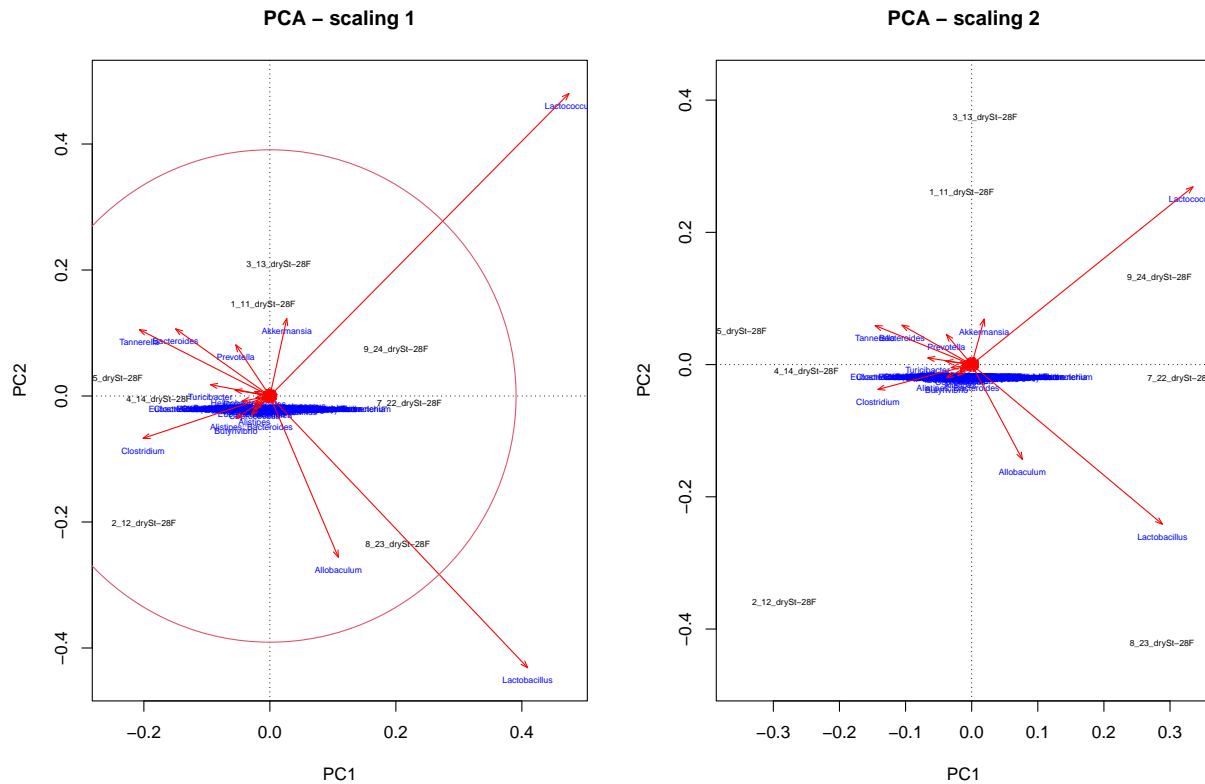


Figura 31: PCA plots de las primeras dos componentes principales de los datos de materia fecal de ratones Vdr^{-/-} usando la función `cleant.pca()`

El “*escalado*” es la forma en que los resultados de la ordenación se proyectan en el espacio reducido para su visualización gráfica (Borcard et al. 2011). La función `cleanplot.pca()` genera dos escalas de PCA. El escalado *PCA 1* en la figura de la izquierda es el biplot de distancia que se centra en las distancias entre las muestras. Si está interesado principalmente en interpretar las relaciones entre las muestras, elija el escalado 1. Las características esenciales de Scaling 1 son: Los vectores propios se escalan a la unidad de longitud y las distancias entre las muestras (sujetos) en el biplot son aproximaciones de sus distancias euclidianas en el espacio multidimensional. Sin embargo, los ángulos entre los vectores de las variables (bacterias en este caso) no tienen sentido. El círculo se llama *círculo de contribución de equilibrio*, que representa la contribución de equilibrio de las variables. Para una combinación dada de ejes, las variables con *vectores más largos* que el radio del círculo pueden ser interpretadas con confianza como las *bacterias más importantes*, mientras que las variables con *vectores más cortos* que el radio del círculo de contribución de equilibrio *contribuyen poco* a un espacio reducido dado.

El *PCA 2* de la figura de la derecha es un biplot de correlación. Traza la correlación entre variables (bacterias en este caso). Si su interés principal se centra en *las relaciones entre variables*, elija el escalamiento 2. Las características esenciales del escalamiento 2 son: Cada vector propio se escala a la raíz cuadrada de su valor propio; la longitud del vector se aproxima a la desviación estándar de las variables (bacterias); los ángulos entre variables (bacterias en este caso) reflejan sus correlaciones: el coseno del ángulo aproxima la correlación entre las variables (bacterias); sin embargo, las distancias entre muestras (sujetos) en el biplot no son aproximaciones de sus distancias euclidianas en el espacio multidimensional.

4.1.1.1. Cálculo del PC1 El análisis de componentes principales tiene como objetivo reducir la dimensión y conservar en lo posible la estructura de estos, la cual no depende de los ejes utilizados para las

coordenadas. Dadas n observaciones de p variables, se analiza si es posible representar adecuadamente la información con un número menor de variables construidas como combinaciones lineales de las originales. El *primer componente principal* es la combinación lineal de las variables originales que tienen varianza máxima.

Supongamos que tenemos datos en un espacio de p variables en n elementos de una población en una matriz X de tamaño $n \times p$, donde las columnas son las variables y las filas los elementos. Vamos a suponer que hemos restado a cada variable su media, de tal forma que las variables de la matriz X tienen media cero y su matriz de covarianzas está dada por $\frac{1}{n} X'X$. Los valores del primer componente de los n individuos se representará por un vector F_1 que está dado por:

$$F_1 = Xa_1.$$

Donde $a_1 = (a_{11}, a_{12}, \dots, a_{1p})$. Como las variables originales se supuso tenían media cero, entonces también F_1 tendrá media nula y su varianza será:

$$\frac{1}{n} F_1' F_1 = \frac{1}{n} a_1' X' X a_1 = a_1' S a_1.$$

donde S es la matriz de varianzas y covarianzas de las observaciones. Si se elige un a_1 que tenga una norma muy grande, entonces la varianza de F_1 podría resultar muy grande también. Entonces es necesario imponer condiciones sobre a_1 para acotar el tamaño de la varianza de F_1 y no hacerla muy grande de manera arbitraria. La condición que se impone es:

$$\|a_1\| = a_1' \cdot a_1 = 1.$$

Esta restricción se introduce mediante el multiplicador de Lagrange:

$$M = a_1' S a_1 - \lambda(a_1' a_1 - 1).$$

Después, se maximizará esta expresión derivando con respecto a los componentes de a_1 e igualando a 0:

$$\frac{\partial M}{\partial a_1} = 2S a_1 - 2\lambda a_1 = 0$$

cuya solución es:

$$S a_1 = \lambda a_1$$

entonces a_1 es un vector propio de la matriz S y λ será su correspondiente valor propio.

Para determinar que valor propio de S es la solución de la ecuación anterior, multiplicamos por a_1 a la izquierda:

$$a_1' S a_1 = \lambda a_1' a_1 = \lambda$$

entonces, λ es la varianza de F_1 . Como esta es la cantidad que queríamos maximizar, entonces λ será el valor propio mayor de la matriz S y su vector asociado a_1 serán los coeficientes de cada variable de la primera componente.

4.1.2. Análisis de coordenadas principales (PCoA)

El PCoA también se denomina *escalamiento multidimensional métrico*. PCoA es una técnica de ordenación que permite al usuario elegir prácticamente cualquier métrica de distancia (por ejemplo, *Jaccard*, *Bray-Curtis*, *Euclides*, etc.). Al igual que PCA, PCoA utiliza los valores propios para medir la importancia de un conjunto de ejes ortogonales devueltos. La dimensionalidad de la matriz se reduce determinando cada vector propio y cada valor propio. Las coordenadas principales se obtienen escalando cada vector propio. El PCoA cuando se calcula sobre distancias euclidianas entre las muestras produce los mismos resultados que el PCA calculado sobre matriz de covarianza del mismo conjunto de datos (si se utiliza el escalado 1). Las funciones de R, incluyendo `cmdscale()` en el paquete `vegan` y `pcoa()` en el paquete `ape` pueden realizar el PCoA. Con `vegan`, los datos de entrada pueden ser calculados por la función `vegdist()` (por defecto es la disimilitud de *Bray-Curtis*), y el diagrama de ordenación puede ser dibujado con la función `ordiplot()`. El diagrama de ordenación también puede ser con la función `biplot.pcoa()` del paquete `ape`. En este caso, utilizamos la función `cmdscale()` y los mismos datos fecales de ratones Vdr—/— para realizar un PCoA. Esta función necesita una matriz de semejanza como datos de entrada. Primero, vamos a calcular la *disimilitud de Bray-Curtis* utilizando la función `vegdist()`.

```
options(digits=4)
options(width=78, digits=4)
library("vegan")

bc_dist <- vegdist(abund_table, "bray")
```

Cuadro 2: Índices de disimilitud de Bray-Curtis

	5_15_	1_11_	2_12_	3_13_	4_14_	7_22_	8_23_	9_24_dry
5_15_drySt-28F	0.0000	0.5873	0.5293	0.5786	0.2765	0.4756	0.6487	0.5984
1_11_drySt-28F	0.5873	0.0000	0.3742	0.2041	0.4896	0.5208	0.4084	0.2921
2_12_drySt-28F	0.5293	0.3742	0.0000	0.4861	0.4114	0.4955	0.5119	0.4368
3_13_drySt-28F	0.5786	0.2041	0.4861	0.0000	0.4645	0.5436	0.4217	0.2362
4_14_drySt-28F	0.2765	0.4896	0.4114	0.4645	0.0000	0.4172	0.6220	0.5210
7_22_drySt-28F	0.4756	0.5208	0.4955	0.5436	0.4172	0.0000	0.5356	0.4359
8_23_drySt-28F	0.6487	0.4084	0.5119	0.4217	0.6220	0.5356	0.0000	0.3105
9_24_drySt-28F	0.5984	0.2921	0.4368	0.2362	0.5210	0.4359	0.3105	0.0000

A continuación, vamos a establecer explícitamente $k = 2$ (los valores por defecto para el número de dimensiones a devolver) y `eig = TRUE` (que guarda los valores propios).

```
PCoA <- cmdscale(bc_dist, eig = TRUE, k = 2)
PCoA
```

```
## $points
##           [,1]      [,2]
## 5_15_drySt-28F  0.35060  0.007035
## 1_11_drySt-28F -0.17119  0.137244
## 2_12_drySt-28F  0.02928  0.115931
## 3_13_drySt-28F -0.17013  0.126964
## 4_14_drySt-28F  0.27190  0.088394
## 7_22_drySt-28F  0.13943 -0.258936
## 8_23_drySt-28F -0.25091 -0.157891
## 9_24_drySt-28F -0.19898 -0.058740
##
## $eig
```

```
## [1] 3.779e-01 1.517e-01 1.170e-01 8.855e-02 2.771e-02 2.167e-02
## [7] -2.515e-17 -4.493e-03
##
## $x
## NULL
##
## $ac
## [1] 0
##
## $GOF
## [1] 0.6713 0.6751
```

La función `cmdscale()` produce una lista de salida. Los primeros **puntos** de salida contienen las coordenadas de cada muestra en cada dimensión reducida. La segunda salida **eig** contiene los valores propios. Las últimas tres salidas pertenecen a otras opciones del análisis que no cubriremos aquí. El siguiente fragmento de códigos R se utiliza para examinar el porcentaje de variación en el conjunto de datos que se explica por los dos primeros ejes del PCoA.

```
explainedvar1 <- round(PCoA$eig[1] / sum(PCoA$eig), 2) * 100
explainedvar1
```

```
## [1] 48
```

```
explainedvar2 <- round(PCoA$eig[2] / sum(PCoA$eig), 2) * 100
explainedvar2
```

```
## [1] 19
```

```
sum_eig <- sum(explainedvar1, explainedvar2)
sum_eig
```

```
## [1] 67
```

El primer eje explica el 48% de las variaciones de los datos, y el segundo el 19%. Por lo tanto, una gran cantidad de variaciones de los datos (un total del 67%) ha sido explicada por estos dos ejes.

Existen dos criterios para evaluar si los primeros ejes del PCoA captan o no una cantidad desproporcionadamente grande de la variación total explicada: (1) el *criterio de Kaiser-Guttman* y (2) el *modelo broken-stick*. El *criterio de Kaiser-Guttman* establece que “los valores propios asociados a los primeros ejes deben ser mayores que la media de todos los valores propios”. Según el *criterio del modelo broken-stick*, los valores propios asociados a los primeros ejes se comparan con las expectativas del modelo broken-stick. El *modelo de broken-stick* supone que la suma total de los valores propios disminuye secuencialmente con los ejes PCoA ordenados. Evaluamos el rendimiento de PCoA utilizando estos dos criterios con los siguientes gráficos:

```
# Correr todo junto
# Define Plot Parameters
par(mar = c(5, 5, 1, 2) + 0.1)
# Plot Eigenvalues
plot(PCoA$eig, xlab = "PCoA", ylab = "Eigenvalue",
     las = 1, cex.lab = 1.5, pch = 16)
# Add Expectation based on Kaiser-Guttman criterion and Broken Stick Model
```

```

abline(h = mean(PCoA$eig), lty = 2, lwd = 2, col = "blue")
b_stick <- bstick(8, sum(PCoA$eig))
lines(1:8, b_stick, type = "l", lty = 4, lwd = 2, col = "red")
# Add Legend
legend("topright", legend = c("Avg Eigenvalue", "Broken-Stick"),
      lty = c(2, 4), bty = "n", col = c("blue", "red"))

```

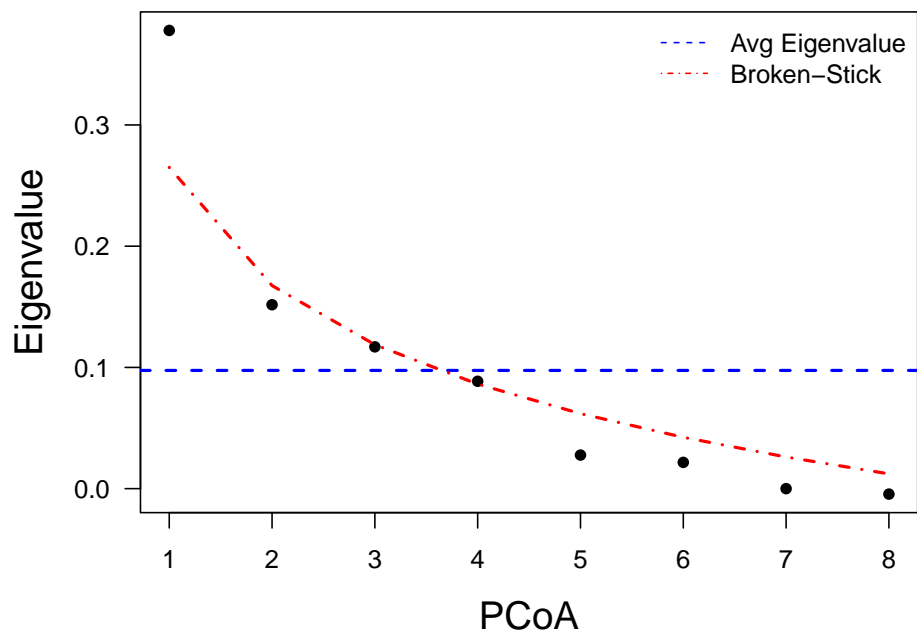


Figura 32: PCoA con el criterio de Kaiser-Guttman y el modelo de broken-stick

La figura anterior muestra que los valores propios asociados a los tres primeros ejes son mayores que la media de todos los valores propios y son mayores que las expectativas del el modelo Broken-Stick. Después de evaluar el resultado del PCoA, crearemos un de ordenación para los dos ejes del PCoA.

```

# Correr todo junto
# Define Plot Parameters
par(mar = c(5, 5, 1, 2) + 0.1)
# Initiate Plot
plot(PCoA$points[,1], PCoA$points[,2], ylim = c(-0.5, 0.5),
     xlab = paste("PCoA 1 (", explainedvar1, "%)", sep = ""),
     ylab = paste("PCoA 2 (", explainedvar2, "%)", sep = ""),
     pch = 5, cex = 1.0, type = "n", cex.lab = 1.0, cex.axis = 1.2, axes = FALSE)
# Add Axes
axis(side = 1, labels = T, lwd.ticks = 2, cex.axis = 1.2, las = 1)
axis(side = 2, labels = T, lwd.ticks = 2, cex.axis = 1.2, las = 1)
abline(h = 0, v = 0, lty = 3)
box(lwd = 2)
# Add Points & Labels
points(PCoA$points[,1], PCoA$points[,2],
       pch = 19, cex = 3, bg = "blue", col = "blue")

```

```
text(PCoA$points[,1], PCoA$points[,2],
     labels = row.names(PCoA$points))
```

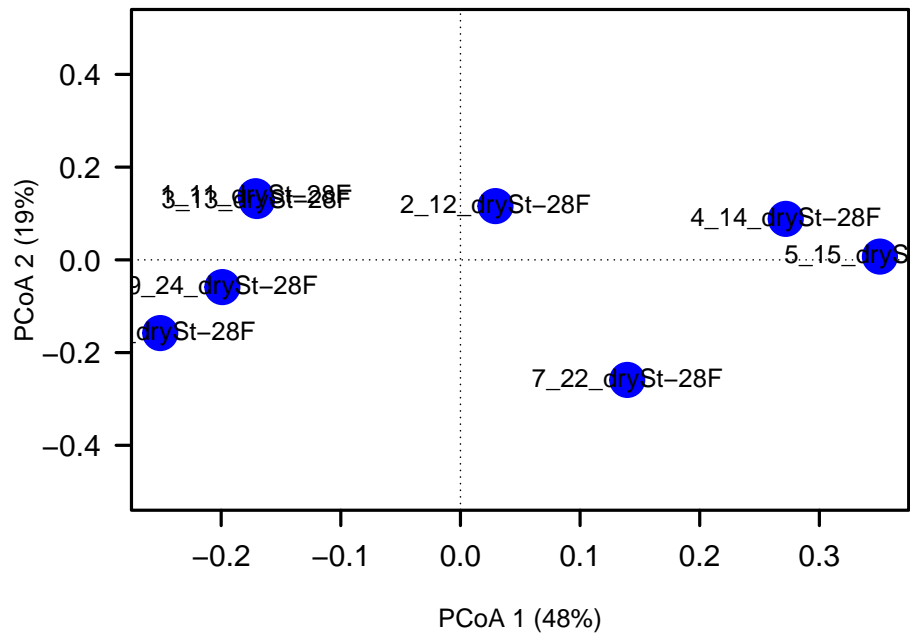


Figura 33: Ordination plot para los dos ejes de PCoA

Los gráficos de ordenación básicos nos permiten ver cómo se separan las muestras entre sí. En nuestro ejemplo, las muestras se separan a lo largo de los ejes PCoA debido a la variación en la abundancia de los diferentes géneros de ratones. Una pregunta lógica de seguimiento es preguntar qué géneros del conjunto de datos están impulsando la divergencia observada entre los puntos. ¿Podemos identificar y visualizar estos géneros influyentes en el PCoA? Podemos obtener esta información utilizando la función `add.spec.scores()` del paquete BiodiversityR. En primer lugar, la abundancia relativa se calcula como sigue:

```
# Correr todo junto
# Define Plot Parameters
par(mar = c(5, 5, 1, 2) + 0.1)
# Initiate Plot
plot(PCoA$points[,1], PCoA$points[,2], ylim = c(-0.5, 0.5),
     xlab = paste("PCoA 1 (", explainedvar1, "%)", sep = ""),
     ylab = paste("PCoA 2 (", explainedvar2, "%)", sep = ""),
     pch = 5, cex = 1.0, type = "n", cex.lab = 1.0, cex.axis = 1.2, axes = FALSE)
# Add Axes
axis(side = 1, labels = T, lwd.ticks = 2, cex.axis = 1.2, las = 1)
axis(side = 2, labels = T, lwd.ticks = 2, cex.axis = 1.2, las = 1)
abline(h = 0, v = 0, lty = 3)
box(lwd = 2)
# Add Points & Labels
points(PCoA$points[,1], PCoA$points[,2],
       pch = 19, cex = 3, bg = "blue", col = "blue")
text(PCoA$points[,1], PCoA$points[,2],
     labels = row.names(PCoA$points))
```

```
fecalREL <- abund_table
for(i in 1:nrow(abund_table)){
  fecalREL[i, ] = abund_table[i, ] / sum(abund_table[i, ])
}

#install.packages("BiodiversityR")
library("pbrtest")
library("BiodiversityR")

# Calculate and Add Species Scores
PCoA <- add.spec.scores(PCoA,fecalREL,method = "pcoa.scores",Rscale=TRUE,scaling=1, multi=1)
text(PCoA$cproj[,1], PCoA $cproj[,2],
     labels = row.names(PCoA$cproj),cex=0.5, col = "blue")
```

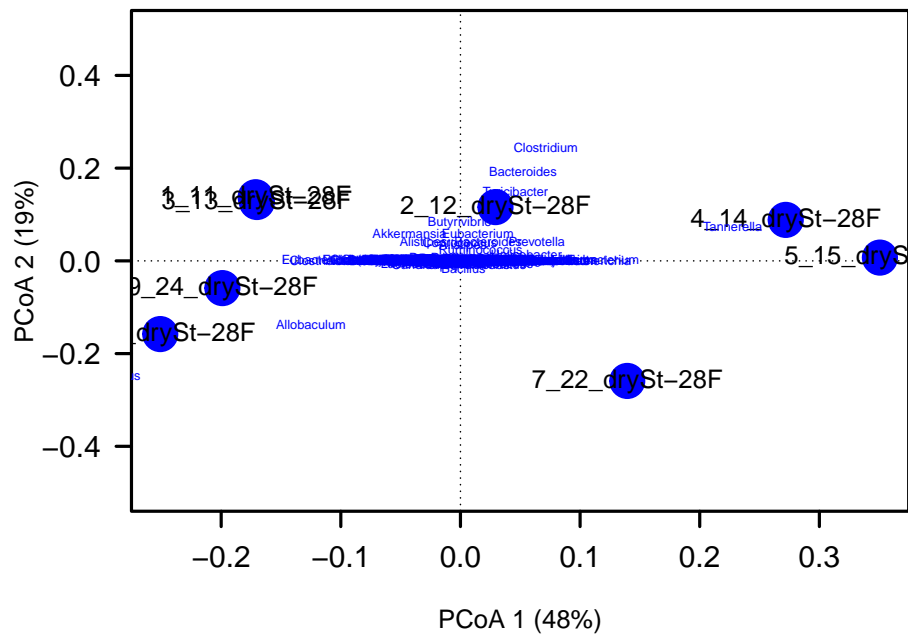


Figura 34: Ordination plot para los dos ejes de PCoA con influencia de genero

La función `add.spec.scores()` también puede utilizarse para determinar la correlación de cada taxón a lo largo de los ejes PCoA. Este es un enfoque más efectivamente cuantitativo de identificar los taxones influyentes. Además, para identificar y obtener el taxón más importante, es necesario definir un coeficiente de correlación de corte, en este caso, vamos a tomar 0,7.

```
genus_corr <- add.spec.scores(PCoA, fecalREL, method = "cor.scores")$cproj
corrcut <- 0.7 # user defined cutoff
import_genus <- genus_corr[abs(genus_corr[, 1]) >= corrcut | abs(genus_corr[, 2]) >= corrcut, ]
```

Los 12 géneros importantes con correlación mayor o igual a 0,7 a lo largo de los ejes PCoA se imprimen a continuación:

```
import_genus[complete.cases(import_genus),] %>%
  kbl(booktabs = TRUE, align = "c",
      caption = "Géneros más importantes con correlación mayor o igual a 0.7" )%>%
  kable_styling(position = "center",
      latex_options = c("hold_position", "striped"),
      font_size = 7)
```

Cuadro 3: Géneros más importantes con correlación mayor o igual a 0.7

	Dim1	Dim2
Tannerella	0.8205	0.1685
Lactobacillus	-0.3064	-0.8665
Helicobacter	0.7488	0.1176
Paraprevotella	0.7463	-0.0858
Bacillus	0.1980	-0.7727
Pedobacter	-0.2461	0.8703
Limibacter	-0.0824	0.7036
Mycoplasma	0.7227	0.2295
Slackia	0.0977	-0.7577
Fluviicola	0.2425	-0.7107
Caldicellulosiruptor	0.2425	-0.7107
Anaerophaga	0.2425	-0.7107

Por último, utilizamos la función `envfit()` del paquete **vegan** para realizar una prueba de permutación para las abundancias generales a través de los ejes en estas correlaciones.

```
# Permutation Test for Species Abundances Across Axes
envfit(PCoA, fecalREL, perm = 999)
```

```
##
## ***VECTORS
##
##
## Dim1 Dim2
## Tannerella 0.951 0.308
## Lactococcus -0.769 -0.639
## Lactobacillus -0.219 -0.976
## Lactobacillus::Lactococcus -0.428 -0.904
## Parasutterella -0.754 -0.657
## Helicobacter 0.971 0.241
## Prevotella 0.842 0.540
## Bacteroides 0.262 0.965
## Barnesiella 0.979 0.204
## Odoribacter -0.537 -0.843
## Eubacterium 0.238 0.971
## Allobaculum -0.676 -0.737
## Roseburia -0.044 0.999
## Clostridium 0.280 0.960
## Porphyromonas 0.359 -0.933
## Butyrivibrio -0.006 1.000
## Ruminococcus 0.557 0.830
## Acholeplasma 0.327 0.945
## Alistipes 0.071 0.997
## Clostridium::Coprococcus 0.000 0.000
## Eubacterium (Erysipelotrichaceae)::Eubacterium -0.820 0.573
```

## Hydrogenoanaerobacterium	0.037	-0.999
## Paraprevotella	0.984	-0.179
## Blautia	-0.087	-0.996
## Adlercreutzia::Asaccharobacter	-0.140	0.990
## Coprococcus	-0.011	1.000
## Parabacteroides	0.978	0.211
## Eubacterium (Erysipelotrichaceae)	-0.770	-0.639
## Oscillibacter	0.273	0.962
## Acinetobacter	0.999	0.050
## Alkalitalea	-0.622	-0.783
## Sporobacter	0.000	0.000
## Oscillospira	0.481	0.876
## Blautia::Clostridium	0.107	-0.994
## Planctomyces	-0.474	0.881
## Akkermansia	-0.587	0.809
## Ruminofilibacter	-0.538	-0.843
## Pseudoflavonifractor	0.639	0.769
## Butyricimonas	0.293	-0.956
## Desulfovibrio	0.971	0.237
## Anaerotruncus	-0.448	0.894
## Alistipes::Bacteroides	0.009	1.000
## Turicibacter	0.293	0.956
## Mucispirillum	0.999	0.050
## Lachnospira	0.494	0.869
## Catabacter	0.000	0.000
## Desulfosporosinus	0.000	0.000
## Bacillus	0.160	-0.987
## Sporanaerobacter	-0.585	-0.811
## Streptomyces	-0.677	0.736
## Butyrivibrio::Clostridium	0.922	-0.388
## Candidatus Arthromitus	-0.185	-0.983
## Enterorhabdus	0.366	0.931
## Kurthia	0.000	0.000
## Eggerthella	0.974	0.227
## Actinocorallia	0.000	0.000
## Caldanaerocella	0.999	0.050
## Dorea	0.391	-0.921
## Streptococcus	-0.370	0.929
## Adlercreutzia	-0.791	0.612
## Paraeggerthella	0.962	0.273
## Lachnobacterium	-0.538	-0.843
## TM7 (genus)	0.101	0.995
## Clostridium::Anaerostipes	0.999	0.050
## Erysipelothrix	-0.208	-0.978
## Formosa	0.000	0.000
## Rikenella	-0.521	-0.854
## Dysgonomonas	-0.456	0.890
## Desulfitobacterium	0.999	0.050
## Oribacterium	-0.474	0.881
## Syntrophococcus	0.000	0.000
## Ruminococcus::Clostridium	-0.538	-0.843
## Rhodospirillum	0.999	0.050
## Pedobacter	-0.176	0.984
## Acetivibrio	0.996	-0.085

## Clostridium::Eubacterium	-0.997	0.078
## Limibacter	-0.074	0.997
## Clostridium::Ruminococcus	0.000	0.000
## Coprobacillus	0.990	0.142
## Cytophaga	-0.403	-0.915
## Denitrobacterium	-0.382	-0.924
## Mycoplasma	0.894	0.448
## Roseburia::Clostridium	-0.448	0.894
## Sporobacterium	0.583	0.813
## Eubacterium::Ruminococcus	0.000	0.000
## Pseudobutyrvibrio	0.101	0.995
## Robinsoniella	0.000	0.000
## Brevibacterium	-0.192	0.981
## Blautia::Ruminococcus	0.961	-0.278
## Pseudomonas	-0.405	0.914
## Clostridium::Roseburia	0.000	0.000
## Desulfotomaculum	-0.991	0.134
## Clostridium (Erysipelotrichaceae)	0.764	0.645
## Olsenella	-0.714	0.700
## Azospirillum	0.000	0.000
## Oxobacter	0.000	0.000
## Asaccharobacter::Adlercreutzia	0.000	0.000
## Desulfocurvus	0.101	0.995
## Anaerostipes	0.101	0.995
## Clostridium::Ruminococcus::Desulfotomaculum::Escherichia	0.000	0.000
## Halanaerobacter	-0.999	0.034
## Slackia	0.081	-0.997
## Anaplasma	0.000	0.000
## Syntrophomonas	0.000	0.000
## Clostridium::Blautia	0.000	0.000
## Anaerosporebacter::Clostridium	0.000	0.000
## Clostridium::Bacteroides	0.000	0.000
## Blautia::Lactonifactor	0.000	0.000
## Parabacteroides::Bacteroides	-0.474	0.881
## Enterorhabdus::Adlercreutzia	0.000	0.000
## Flavobacterium	-0.266	0.964
## Clostridium::Desulfotomaculum	0.000	0.000
## Parasporobacterium	0.999	0.050
## Coprococcus::Clostridium	0.000	0.000
## Fusobacterium	0.854	0.521
## Ruminococcus::Escherichia	-0.448	0.894
## Clostridium::Ruminococcus::Coprococcus	0.000	0.000
## Lactonifactor	0.000	0.000
## Haemophilus	0.309	0.951
## Sporichthya	-0.849	0.529
## Cytophaga::Flavobacterium	-0.768	-0.640
## Clostridium::Dorea	0.000	0.000
## Ruminococcus::Blautia	0.000	0.000
## Frigoribacterium	-0.474	0.881
## Paenibacillus	-0.738	-0.675
## Johnsonella	0.101	0.995
## Pseudoflavonifractor::Clostridium	0.000	0.000
## Adlercreutzia::Enterorhabdus::Asaccharobacter	0.000	0.000
## Nocardiosis	-0.474	0.881

## Pedobacter::Pseudomonas	0.000	0.000
## Flexibacter	-0.688	-0.726
## Catabacter::Ruminococcus	0.000	0.000
## Butyricimonas::Bacteroides	-0.474	0.881
## Staphylococcus	-0.253	0.968
## Alkalitalea::Prevotella	0.000	0.000
## Lachnospira::Anaerostipes	0.000	0.000
## Flavobacterium::Cytophaga	-0.538	-0.843
## Gelidibacter	0.817	0.577
## Treponema	-0.474	0.881
## Pontibacter	0.000	0.000
## Desulfuromonas	0.000	0.000
## Butyricicoccus	0.000	0.000
## Clostridium::Butyrivibrio	-0.448	0.894
## Coprobacillus::Clostridium	0.000	0.000
## Porphyromonas::Prevotella	0.000	0.000
## Effluviibacter	-0.580	0.815
## Caminicella	0.101	0.995
## Prevotella::Bacteroides	0.000	0.000
## Pseudoalteromonas	0.000	0.000
## Butyrivibrio::Blautia	0.000	0.000
## Lachnospira::Clostridium	-0.806	-0.593
## Alicyclobacillus	0.000	0.000
## Lachnospira::Robinsoniella	0.000	0.000
## Subdoligranulum	0.000	0.000
## Anaerofilum	0.000	0.000
## Sporosarcina	0.000	0.000
## Alkalitalea::Sphingobacterium	0.000	0.000
## Koproimonas	0.777	0.629
## Olivibacter	0.000	0.000
## Lachnobacterium::Coprococcus	0.000	0.000
## Fluviicola	0.211	-0.977
## Peptococcus	0.000	0.000
## Ruminococcus::Roseburia	0.000	0.000
## Marinilabilia	0.000	0.000
## Catonella	0.000	0.000
## Sphingobium	0.101	0.995
## Olsenella::Streptomyces	0.000	0.000
## Terasakiella	0.000	0.000
## Roseospirillum	-0.152	-0.988
## Clostridium::Ruminococcus::Blautia	0.000	0.000
## Lachnospira::Roseburia	0.000	0.000
## Thalassospira	0.000	0.000
## Eubacterium::Clostridium	0.101	0.995
## Allobaculum::Eubacterium	0.000	0.000
## Blautia::Lachnospira	0.000	0.000
## Cryptobacterium	0.000	0.000
## Atopobium	0.000	0.000
## Eubacterium::Lachnospira	0.000	0.000
## Bacteroides::Alistipes	0.000	0.000
## Faecalibacterium	-0.192	0.981
## Lachnospira::Coprococcus	0.000	0.000
## Microbacterium	0.000	0.000
## Zhangella::Stella	0.777	0.629

## Paludibacter	0.000	0.000
## Butyrivibrio::Ruminococcus	0.000	0.000
## Bacteroides::Lactobacillus	0.000	0.000
## Prevotella::Flavobacterium	-0.806	-0.593
## Ruminococcus::Dorea	0.000	0.000
## Slackia::Asaccharobacter::Adlercreutzia	0.000	0.000
## Caldicellulosiruptor	0.211	-0.977
## Kordia	0.000	0.000
## Bacteroides::Lactobacillus::Lachnospira	-0.448	0.894
## Sphingobacterium	0.000	0.000
## Anaeroplasma	-0.448	0.894
## Atopostipes	0.000	0.000
## Enterococcus	0.000	0.000
## Insolitispirillum	0.000	0.000
## Clostridium::Acetivibrio	0.000	0.000
## Plantactinospira	0.000	0.000
## Stappia	0.000	0.000
## Anaplasma::Clostridium	0.000	0.000
## Clostridium (Erysipelotrichaceae)::Clostridium	0.000	0.000
## Algoriphagus	0.000	0.000
## Dorea::Ruminococcus	0.000	0.000
## Roseburia::Lachnospira	0.000	0.000
## Rhizobium	0.000	0.000
## Anaerofustis	0.000	0.000
## Echinicola	0.000	0.000
## Anaerostipes::Clostridium	0.000	0.000
## Lachnospira::Blautia	0.000	0.000
## Aestuariimicrobium	0.000	0.000
## Gelidibacter::Flavobacterium	0.000	0.000
## Helicobacter::Flexispira	0.101	0.995
## Eubacterium (Erysipelotrichaceae)::Paenibacillus::Eubacterium	0.000	0.000
## Pelospora	0.000	0.000
## Stella	0.000	0.000
## Methylobacterium	0.000	0.000
## Rickettsia	-0.448	0.894
## Porphyromonas::Flavobacterium	0.000	0.000
## Adlercreutzia::Enterorhabdus	0.000	0.000
## Ruminococcus::Escherichia::Parabacteroides	0.000	0.000
## Limibacter::Bacteroides	0.000	0.000
## Paraprevotella::Prevotella	0.000	0.000
## Janthinobacterium::Zoogloea::Duganella	0.000	0.000
## Flavobacterium::Gelidibacter	0.000	0.000
## Barnesiella::Bacteroides	0.000	0.000
## Porphyromonas::Gelidibacter	0.000	0.000
## Bacilloplasma	0.000	0.000
## Natronincola	0.000	0.000
## Lumbricincola	0.000	0.000
## Desulfotomaculum::Clostridium	0.000	0.000
## Roseburia::Ruminococcus	0.000	0.000
## Bifidobacterium	0.000	0.000
## Streptacidiphilus	0.000	0.000
## Butyrivibrio::Pseudobutyrvibrio	0.101	0.995
## Aeromicrobium	0.000	0.000
## Proteus	0.000	0.000

## Sporobacterium::Clostridium	0.000	0.000
## Butyrivibrio::Roseburia	0.000	0.000
## Luteococcus	0.000	0.000
## Clostridium::Blautia::Desulfotomaculum	0.101	0.995
## Lachnospira::Ruminococcus	0.101	0.995
## Ornithinimicrobium	0.000	0.000
## Persicivirga	0.000	0.000
## Lachnospira::Ruminococcus::Escherichia	0.000	0.000
## Anaerophaga	0.211	-0.977
## Dysgonomonas::Flavobacterium	0.000	0.000
## Bizionia	0.101	0.995
##	r2	Pr(>r)
## Tannerella	0.70	0.041 *
## Lactococcus	0.40	0.264
## Lactobacillus	0.84	0.012 *
## Lactobacillus::Lactococcus	0.35	0.327
## Parasutterella	0.17	0.674
## Helicobacter	0.57	0.050 *
## Prevotella	0.52	0.162
## Bacteroides	0.44	0.262
## Barnesiella	0.44	0.158
## Odoribacter	0.39	0.293
## Eubacterium	0.48	0.201
## Allobaculum	0.41	0.241
## Roseburia	0.15	0.807
## Clostridium	0.58	0.120
## Porphyromonas	0.63	0.040 *
## Butyrivibrio	0.30	0.432
## Ruminococcus	0.38	0.277
## Acholeplasma	0.46	0.205
## Alistipes	0.23	0.579
## Clostridium::Coprococcus	0.00	1.000
## Eubacterium (Erysipelotrichaceae)::Eubacterium	0.14	0.680
## Hydrogenoanaerobacterium	0.14	0.711
## Paraprevotella	0.56	0.081 .
## Blautia	0.03	0.939
## Adlercreutzia::Asaccharobacter	0.06	0.978
## Coprococcus	0.37	0.312
## Parabacteroides	0.50	0.097 .
## Eubacterium (Erysipelotrichaceae)	0.25	0.543
## Oscillibacter	0.34	0.356
## Acinetobacter	0.37	0.361
## Alkalitalea	0.35	0.282
## Sporobacter	0.00	1.000
## Oscillospira	0.17	0.588
## Blautia::Clostridium	0.08	0.803
## Planctomyces	0.21	0.752
## Akkermansia	0.15	0.648
## Ruminofilibacter	0.38	0.246
## Pseudoflavonifractor	0.37	0.299
## Butyricimonas	0.45	0.212
## Desulfovibrio	0.40	0.217
## Anaerotruncus	0.23	0.603
## Alistipes::Bacteroides	0.22	0.615

## Turicibacter	0.46	0.191
## Mucispirillum	0.37	0.361
## Lachnospira	0.11	0.865
## Catabacter	0.00	1.000
## Desulfosporosinus	0.00	1.000
## Bacillus	0.64	0.054
## Sporanaerobacter	0.22	0.629
## Streptomyces	0.10	0.868
## Butyrivibrio::Clostridium	0.39	0.242
## Candidatus Arthromitus	0.51	0.103
## Enterorhabdus	0.15	0.713
## Kurthia	0.00	1.000
## Eggerthella	0.38	0.268
## Actinocorallia	0.00	1.000
## Caldanaerocella	0.37	0.361
## Dorea	0.17	0.621
## Streptococcus	0.07	0.872
## Adlercreutzia	0.30	0.443
## Paraeggerthella	0.17	0.685
## Lachnobacterium	0.38	0.246
## TM7 (genus)	0.10	1.000
## Clostridium::Anaerostipes	0.37	0.361
## Erysipelothrix	0.25	0.475
## Formosa	0.00	1.000
## Rikenella	0.12	0.703
## Dysgonomonas	0.46	0.167
## Desulfitobacterium	0.37	0.361
## Oribacterium	0.21	0.752
## Syntrophococcus	0.00	1.000
## Ruminococcus::Clostridium	0.38	0.246
## Rhodospirillum	0.37	0.361
## Pedobacter	0.82	0.014 *
## Acetivibrio	0.46	0.242
## Clostridium::Eubacterium	0.05	0.918
## Limibacter	0.50	0.197
## Clostridium::Ruminococcus	0.00	1.000
## Coprobacillus	0.27	0.447
## Cytophaga	0.47	0.206
## Denitrobacterium	0.04	0.870
## Mycoplasma	0.57	0.118
## Roseburia::Clostridium	0.23	0.603
## Sporobacterium	0.37	0.251
## Eubacterium::Ruminococcus	0.00	1.000
## Pseudobutyrvibrio	0.10	1.000
## Robinsoniella	0.00	1.000
## Brevibacterium	0.30	0.387
## Blautia::Ruminococcus	0.26	0.526
## Pseudomonas	0.58	0.108
## Clostridium::Roseburia	0.00	1.000
## Desulfotomaculum	0.21	0.605
## Clostridium (Erysipelotrichaceae)	0.13	0.824
## Olsenella	0.06	0.897
## Azospirillum	0.00	1.000
## Oxobacter	0.00	1.000

## Asaccharobacter::Adlercreutzia	0.00	1.000
## Desulfocurvus	0.10	1.000
## Anaerostipes	0.10	1.000
## Clostridium::Ruminococcus::Desulfotomaculum::Escherichia	0.00	1.000
## Halanaerobacter	0.24	0.532
## Slackia	0.58	0.101
## Anaplasma	0.00	1.000
## Syntrophomonas	0.00	1.000
## Clostridium::Blautia	0.00	1.000
## Anaerosporobacter::Clostridium	0.00	1.000
## Clostridium::Bacteroides	0.00	1.000
## Blautia::Lactonifactor	0.00	1.000
## Parabacteroides::Bacteroides	0.21	0.752
## Enterorhabdus::Adlercreutzia	0.00	1.000
## Flavobacterium	0.43	0.233
## Clostridium::Desulfotomaculum	0.00	1.000
## Parasporobacterium	0.37	0.361
## Coprococcus::Clostridium	0.00	1.000
## Fusobacterium	0.36	0.281
## Ruminococcus::Escherichia	0.23	0.603
## Clostridium::Ruminococcus::Coprococcus	0.00	1.000
## Lactonifactor	0.00	1.000
## Haemophilus	0.06	0.832
## Sporichthya	0.26	0.513
## Cytophaga::Flavobacterium	0.38	0.268
## Clostridium::Dorea	0.00	1.000
## Ruminococcus::Blautia	0.00	1.000
## Frigoribacterium	0.21	0.752
## Paenibacillus	0.29	0.450
## Johnsonella	0.10	1.000
## Pseudoflavonifractor::Clostridium	0.00	1.000
## Adlercreutzia::Enterorhabdus::Asaccharobacter	0.00	1.000
## Nocardiosis	0.21	0.752
## Pedobacter::Pseudomonas	0.00	1.000
## Flexibacter	0.42	0.245
## Catabacter::Ruminococcus	0.00	1.000
## Butyricimonas::Bacteroides	0.21	0.752
## Staphylococcus	0.50	0.162
## Alkalitalea::Prevotella	0.00	1.000
## Lachnospira::Anaerostipes	0.00	1.000
## Flavobacterium::Cytophaga	0.38	0.246
## Gelidibacter	0.46	0.197
## Treponema	0.21	0.752
## Pontibacter	0.00	1.000
## Desulfuromonas	0.00	1.000
## Butyricicoccus	0.00	1.000
## Clostridium::Butyrivibrio	0.23	0.603
## Coprobacillus::Clostridium	0.00	1.000
## Porphyromonas::Prevotella	0.00	1.000
## Effluviibacter	0.26	0.487
## Caminicella	0.10	1.000
## Prevotella::Bacteroides	0.00	1.000
## Pseudoalteromonas	0.00	1.000
## Butyrivibrio::Blautia	0.00	1.000

## Lachnospira::Clostridium	0.15	0.884
## Alicyclobacillus	0.00	1.000
## Lachnospira::Robinsoniella	0.00	1.000
## Subdoligranulum	0.00	1.000
## Anaerofilum	0.00	1.000
## Sporosarcina	0.00	1.000
## Alkalitalea::Sphingobacterium	0.00	1.000
## Koproimonas	0.28	0.525
## Olivibacter	0.00	1.000
## Lachnobacterium::Coprococcus	0.00	1.000
## Fluviicola	0.56	0.122
## Peptococcus	0.00	1.000
## Ruminococcus::Roseburia	0.00	1.000
## Marinilabilia	0.00	1.000
## Catonella	0.00	1.000
## Sphingobium	0.10	1.000
## Olsenella::Streptomyces	0.00	1.000
## Terasakiella	0.00	1.000
## Roseospirillum	0.14	0.685
## Clostridium::Ruminococcus::Blautia	0.00	1.000
## Lachnospira::Roseburia	0.00	1.000
## Thalassospira	0.00	1.000
## Eubacterium::Clostridium	0.10	1.000
## Allobaculum::Eubacterium	0.00	1.000
## Blautia::Lachnospira	0.00	1.000
## Cryptobacterium	0.00	1.000
## Atopobium	0.00	1.000
## Eubacterium::Lachnospira	0.00	1.000
## Bacteroides::Alistipes	0.00	1.000
## Faecalibacterium	0.30	0.387
## Lachnospira::Coprococcus	0.00	1.000
## Microbacterium	0.00	1.000
## Zhangella::Stella	0.28	0.525
## Paludibacter	0.00	1.000
## Butyrivibrio::Ruminococcus	0.00	1.000
## Bacteroides::Lactobacillus	0.00	1.000
## Prevotella::Flavobacterium	0.15	0.884
## Ruminococcus::Dorea	0.00	1.000
## Slackia::Asaccharobacter::Adlercreutzia	0.00	1.000
## Caldicellulosiruptor	0.56	0.122
## Kordia	0.00	1.000
## Bacteroides::Lactobacillus::Lachnospira	0.23	0.603
## Sphingobacterium	0.00	1.000
## Anaeroplasma	0.23	0.603
## Atopostipes	0.00	1.000
## Enterococcus	0.00	1.000
## Insolitispirillum	0.00	1.000
## Clostridium::Acetivibrio	0.00	1.000
## Plantactinospora	0.00	1.000
## Stappia	0.00	1.000
## Anaplasma::Clostridium	0.00	1.000
## Clostridium (Erysipelotrichaceae)::Clostridium	0.00	1.000
## Algoriphagus	0.00	1.000
## Dorea::Ruminococcus	0.00	1.000

```

## Roseburia::Lachnospira                0.00 1.000
## Rhizobium                             0.00 1.000
## Anaerofustis                          0.00 1.000
## Echinicola                            0.00 1.000
## Anaerostipes::Clostridium              0.00 1.000
## Lachnospira::Blautia                   0.00 1.000
## Aestuariimicrobium                    0.00 1.000
## Gelidibacter::Flavobacterium           0.00 1.000
## Helicobacter::Flexispira               0.10 1.000
## Eubacterium (Erysipelotrichaceae)::Paenibacillus::Eubacterium 0.00 1.000
## Pelospora                             0.00 1.000
## Stella                                0.00 1.000
## Methylobacterium                       0.00 1.000
## Rickettsia                             0.23 0.603
## Porphyromonas::Flavobacterium          0.00 1.000
## Adlercreutzia::Enterorhabdus          0.00 1.000
## Ruminococcus::Escherichia::Parabacteroides 0.00 1.000
## Limibacter::Bacteroides               0.00 1.000
## Paraprevotella::Prevotella            0.00 1.000
## Janthinobacterium::Zoogloea::Duganella 0.00 1.000
## Flavobacterium::Gelidibacter          0.00 1.000
## Barnesiella::Bacteroides              0.00 1.000
## Porphyromonas::Gelidibacter            0.00 1.000
## Bacilloplasma                         0.00 1.000
## Natronincola                           0.00 1.000
## Lumbricincola                          0.00 1.000
## Desulfotomaculum::Clostridium          0.00 1.000
## Roseburia::Ruminococcus               0.00 1.000
## Bifidobacterium                       0.00 1.000
## Streptacidiphilus                     0.00 1.000
## Butyrivibrio::Pseudobutyrvibrio       0.10 1.000
## Aeromicrobium                         0.00 1.000
## Proteus                               0.00 1.000
## Sporobacterium::Clostridium            0.00 1.000
## Butyrivibrio::Roseburia               0.00 1.000
## Luteococcus                           0.00 1.000
## Clostridium::Blautia::Desulfotomaculum 0.10 1.000
## Lachnospira::Ruminococcus             0.10 1.000
## Ornithinimicrobium                    0.00 1.000
## Persicivirga                          0.00 1.000
## Lachnospira::Ruminococcus::Escherichia 0.00 1.000
## Anaerophaga                           0.56 0.122
## Dysgonomonas::Flavobacterium          0.00 1.000
## Bizionia                              0.10 1.000
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
## Permutation: free
## Number of permutations: 999

```

Los géneros con un p -valor menor a 0,05 se muestran con el siguiente gráfico.

```

# Correr todo junto
# Define Plot Parameters
par(mar = c(5, 5, 1, 2) + 0.1)

```



```

# Initiate Plot
plot(PCoA$points[,1], PCoA$points[,2], ylim = c(-0.5, 0.5),
     xlab = paste("PCoA 1 (", explainedvar1, "%)", sep = ""),
     ylab = paste("PCoA 2 (", explainedvar2, "%)", sep = ""),
     pch = 5, cex = 1.0, type = "n", cex.lab = 1.0, cex.axis = 1.2, axes = FALSE)

# Add Axes
axis(side = 1, labels = T, lwd.ticks = 2, cex.axis = 1.2, las = 1)
axis(side = 2, labels = T, lwd.ticks = 2, cex.axis = 1.2, las = 1)
abline(h = 0, v = 0, lty = 3)
box(lwd = 2)

# Add Points & Labels
points(PCoA$points[,1], PCoA$points[,2],
       pch = 19, cex = 3, bg = "blue", col = "blue")
text(PCoA$points[,1], PCoA$points[,2],
     labels = row.names(PCoA$points))

fecalREL <- abund_table
for(i in 1:nrow(abund_table)){
  fecalREL[i, ] = abund_table[i, ] / sum(abund_table[i, ])
}

#install.packages("BiodiversityR")
library("pbkrtest")
library("BiodiversityR")

# Calculate and Add Species Scores
PCoA <- add.spec.scores(PCoA,fecalREL,method = "pcoa.scores",Rscale=TRUE,scaling=1, multi=1)
text(PCoA$cproj[,1], PCoA $cproj[,2],
     labels = row.names(PCoA$cproj),cex=0.5, col = "blue")

genus_corr <- add.spec.scores(PCoA, fecalREL, method = "cor.scores")$cproj
corrcut <- 0.7 # user defined cutoff
import_genus <- genus_corr[abs(genus_corr[, 1]) >= corrcut | abs(genus_corr[, 2]) >= corrcut, ]

# Código para los generos con un p-valor<0.05

fit <- envfit(PCoA, fecalREL, perm = 999)
plot(fit, p.max = 0.05, cex=0.5, col = "red")

```

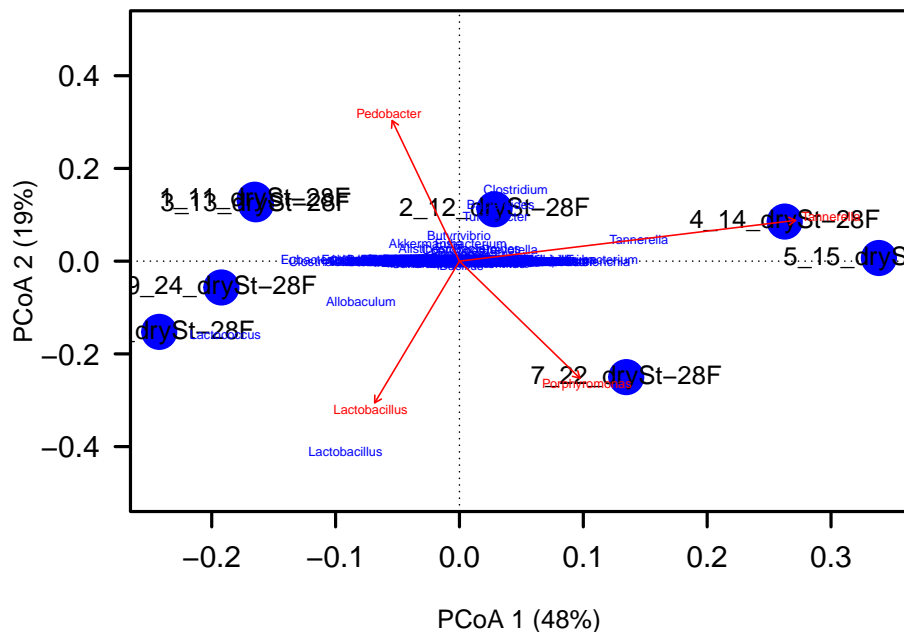


Figura 35: Ordination plot para los dos ejes de PCoA de influencia de genero con un p -valor $< 0,05$

4.1.3. Escalamiento multidimensional no métrico (NMDS)

El *NMDS* es la alternativa no métrica al análisis PCoA. El NMDS ha sido reconocido como un buen método de ordenación porque utiliza formas ecológicamente significativas para medir las *disimilitudes* de las comunidades y cualquier medida de distancia (disimilitudes) entre las muestras como entrada. Por ello, es el método recomendado en la ordenación de comunidades. El objetivo principal del análisis NMDS es *proyectar la posición relativa de los puntos de la muestra en un espacio de ordenación de baja dimensión* (dos o tres ejes).

La función `metaMDS()` del paquete `vegan` realiza el análisis NMDS. Para simplificar, el algoritmo del análisis NMDS se resume como sigue: en primer lugar, utiliza la función `vegdist()` para obtener las medidas de disimilitud adecuadas; a continuación, ejecuta NMDS varias veces con configuraciones iniciales aleatorias y compara los resultados mediante la función `procrustes()`, y se detiene tras encontrar dos veces una solución mínima similar. Por último, escala y rota la solución, y añade las puntuaciones de las especies (o OTUs/taxa) a la configuración como promedios ponderados utilizando la función `wascores()`. Una vez finalizado el algoritmo, la solución final se rota utilizando PCA para facilitar su interpretación.

En este caso, utilizamos el paquete `vegan` y los mismos datos fecales de ratones Vdr/- para ilustrar análisis NMDS. En primer lugar, utilizamos la función `vegdist()` para obtener las medidas de disimilitud de *Bray-Curtis* (el método por defecto) utilizando la configuración por defecto de la función `metaMDS()`. Esta función transforma automáticamente los datos y comprueba la solidez de la solución. La doble estandarización de **Wisconsin** y la transformación **sqrt** se utilizaron conjuntamente en la llamada a la función `metaMDS()`. La llamada a la función produce un valor de tensión del 7,57 %.

```
bc_nmds <- metaMDS(abund_table, dist = "bray")
```

```
## Square root transformation
## Wisconsin double standardization
## Run 0 stress 0.07574
```

```
## Run 1 stress 0.07574
## ... New best solution
## ... Procrustes: rmse 6.721e-07  max resid 1.058e-06
## ... Similar to previous best
## Run 2 stress 0.1192
## Run 3 stress 0.1284
## Run 4 stress 0.07574
## ... Procrustes: rmse 3.801e-07  max resid 6.028e-07
## ... Similar to previous best
## Run 5 stress 0.1758
## Run 6 stress 0.1162
## Run 7 stress 0.1192
## Run 8 stress 0.07574
## ... Procrustes: rmse 1.075e-06  max resid 1.813e-06
## ... Similar to previous best
## Run 9 stress 0.1284
## Run 10 stress 0.1284
## Run 11 stress 0.1899
## Run 12 stress 0.1393
## Run 13 stress 0.1562
## Run 14 stress 0.1801
## Run 15 stress 0.229
## Run 16 stress 0.1562
## Run 17 stress 0.1192
## Run 18 stress 0.1284
## Run 19 stress 0.1192
## Run 20 stress 0.1162
## *** Solution reached
```

```
bc_nmds
```

```
##
## Call:
## metaMDS(comm = abund_table, distance = "bray")
##
## global Multidimensional Scaling using monoMDS
##
## Data:      wisconsin(sqrt(abund_table))
## Distance: bray
##
## Dimensions: 2
## Stress:      0.07574
## Stress type 1, weak ties
## Two convergent solutions found after 20 tries
## Scaling: centring, PC rotation, halfchange scaling
## Species: expanded scores based on 'wisconsin(sqrt(abund_table))'
```

En segundo lugar, utilizamos la función `ordiplot()` para dibujar los resultados del NMDS. La configuración por defecto sólo añade puntos a la figura, utilizamos `type = 't'` o `type = 'text'` para añadir etiquetas de texto en esta figura.

```
ordiplot (bc_nmds, type = 't')
```

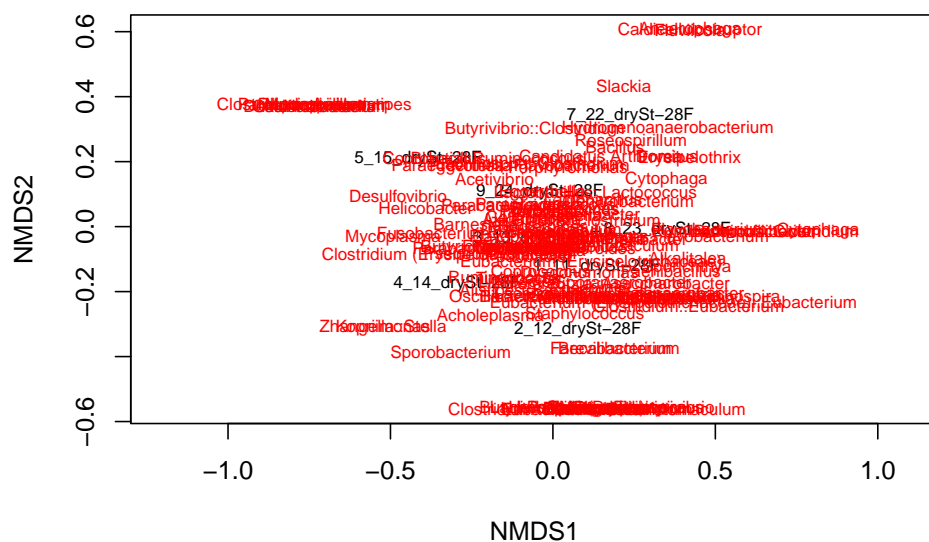


Figura 36: Ordiploot de dos ejes de NMDS con etiquetas

Para trazar las puntuaciones del sitio (muestra) como texto:

```
ordiploot(bc_nmds, display = "sites", type = "text")
```

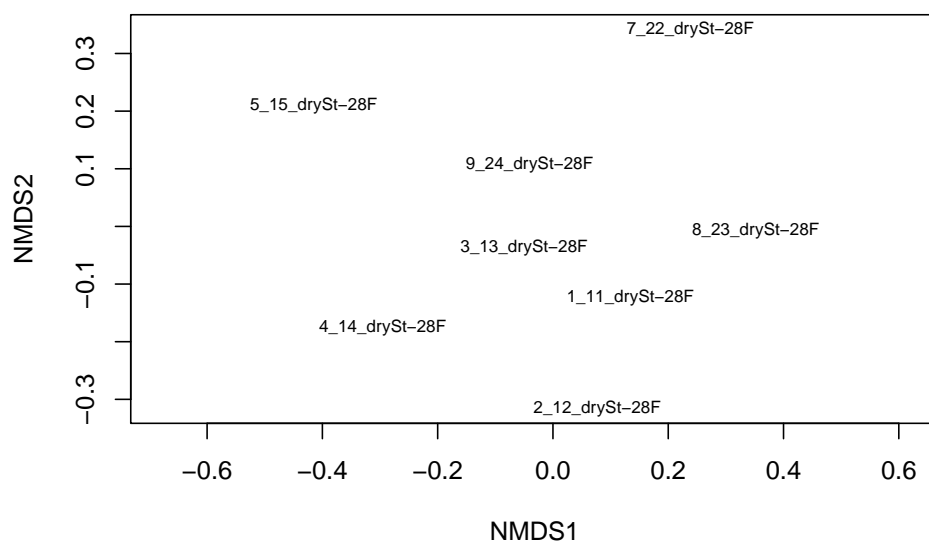


Figura 37: Ordiploot de dos ejes de NMDS con etiquetas de las muestras

Finalmente, la función `stressplot()` se utiliza para dibujar el gráfico de *estrés de Shepard*. La función `stressplot()` genera dos figuras: una traza las distancias de ordenación frente a la disimilitud observada (las disimilitudes de la comunidad elegida), junto con una línea de paso monótona para mostrar el ajuste; otra traza la bondad del ajuste para evaluar la bondad de la ordenación de NMDS2 frente a NMDS1 de muestras concretas. Primero, dividimos la ventana de trazado en dos paneles. La función `plot()` dibuja el diagrama de ordenación NMDS con los sitios (muestras). La función `points()` añade los puntos con un tamaño que refleja la bondad del ajuste (mayor = peor ajuste).

```
par (mfrow = c(1,2))
stressplot (bc_nmds)
plot (bc_nmds, display = 'sites', type = 't', main = 'Goodness of fit')
points (bc_nmds, display = 'sites', cex = goodness (bc_nmds)*300)
```

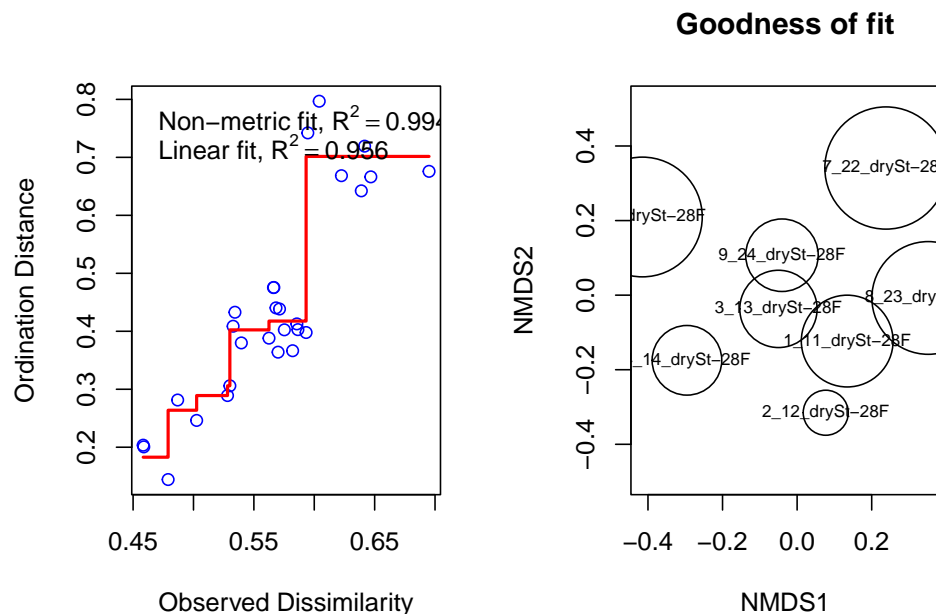


Figura 38: Gráfico del estrés de Shepard y de la bondad de ajuste

El stressplot muestra la relación entre las distancias reales entre las muestras en solución de ordenación de dimensión m , y sus disimilitudes composicionales particulares expresadas por la medida de disimilitud de *Bray-Curtis*. Existen dos estadísticas de bondad de ajuste similares a la correlación; la correlación basada en la tensión $R^2 = 1 - S^2$ (ajuste no métrico = 0,994) y la correlación entre los valores ajustados y las distancias de ordenación, o entre la línea de paso y los puntos: “ R^2 basado en el ajuste” (ajuste lineal = 0,956).

4.1.4. Análisis de correspondencia (AC)

CA es otro método de ordenación sin restricciones. Utiliza las *distancias chi-cuadrado* entre las muestras en el espacio multidimensional de todos los ejes de ordenación y da un alto peso a las especies raras (por ejemplo, especies de baja ocurrencia con muchos ceros). Al igual que PCA y PCoA, CA utiliza los valores propios para medir la importancia de los ejes ortogonales devueltos.

En el diagrama de ordenación del PCA, los taxones son vectores y las muestras son puntos, mientras que en el CA, los taxones y las muestras están representados por puntos. Al igual que PCA, CA tiene dos tipos

de escalas: Escalas 1 y 2. En el espacio de ordenación reducido, *las distancias entre las muestras* (Escala 1) se aproximan a su *distancia chi-cuadrado*. Por ejemplo, cualquier muestra cercana al punto que representa un taxón probablemente contiene una alta contribución a ese taxón. Las *distancias entre taxones* (Escala 2) también se aproximan a sus *distancias chi-cuadrado*. Por ejemplo, cualquier taxón cercano al punto que representa una muestra probablemente tenga una mayor frecuencia en esa muestra. La función `cca()` del paquete **vegan** puede utilizarse para realizar un análisis de correspondencia sin restricciones. Utilizamos la función `cca()` para realizar el CCA y la función `evplot()` para seleccionar ejes de ordenación importantes basados en el criterio de *Kaiser-Guttman* o modelo de *broken-stick*. La función `evplot()`, escrita por Borcard et al. (2011), se utiliza aquí para trazar los valores propios y los porcentajes de variación de un objeto de ordenación. Cuando se utiliza la función `cca()` (`cca` = análisis canónico compoente) para realizar CA sin restricciones, no se especifica la matriz ambiental ni la información de agrupación.

```
fecal_genus_cca=cca(abund_table)
fecal_genus_cca
```

```
## Call: cca(X = abund_table)
##
##              Inertia Rank
## Total              0.502
## Unconstrained    0.502    7
## Inertia is scaled Chi-square
## 122 species (variables) deleted due to missingness
##
## Eigenvalues for unconstrained axes:
##   CA1   CA2   CA3   CA4   CA5   CA6   CA7
## 0.2036 0.1256 0.0747 0.0374 0.0322 0.0200 0.0090
```

La heterogeneidad total de los datos (inercia) es de 0.502, y el primer eje capta 40.5638% de la variación total en la composición del género ($0.2036 / 0.502 = 0.4056$, donde 0.2036 es el valor propio del primer eje CA1, y 0.502 es la heterogeneidad total de los datos).

Los siguientes códigos de R se utilizan para trazar la ordenación y mostrar los nombres de las muestras en la figura:

```
plot(fecal_genus_cca, display="sites")
```

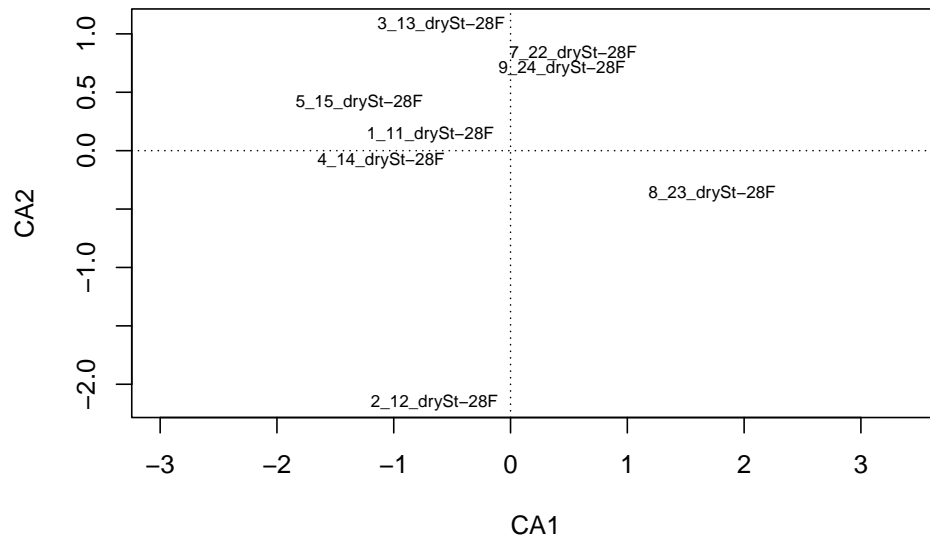


Figura 39: Plot de ordenación con muestras.

Si no quiere que el gráfico esté sobrecargado, utilice los siguientes códigos de R para sólo mostrar puntos en lugar de nombres de muestras en la figura:

```
plot(fecal_genus_cca, display="sites", type="p")
```

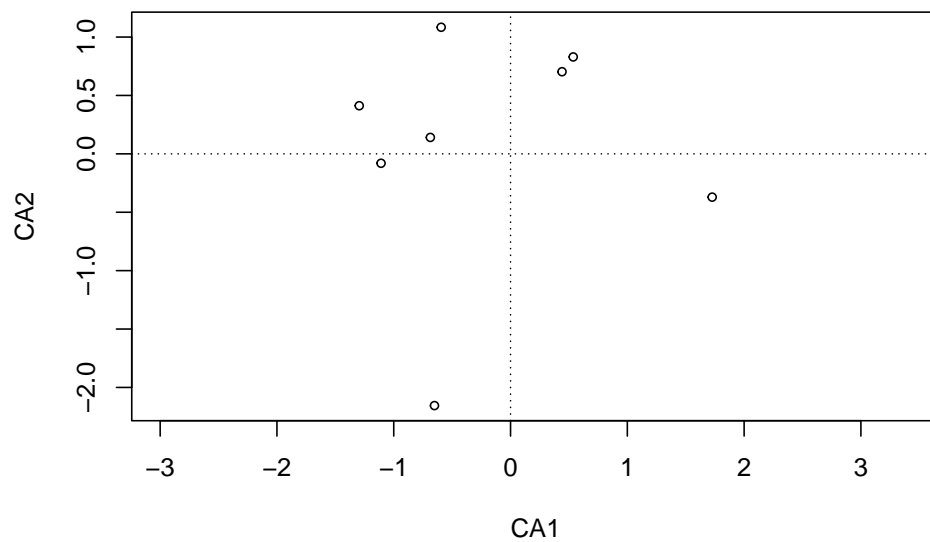


Figura 40: Plot de ordenación sin etiquetas de las muestras

El siguiente diagrama de ordenación revela el patrón de muestras y géneros en diagrama de ordenación:

```
ordiplot(fecal_genus_cca)
```

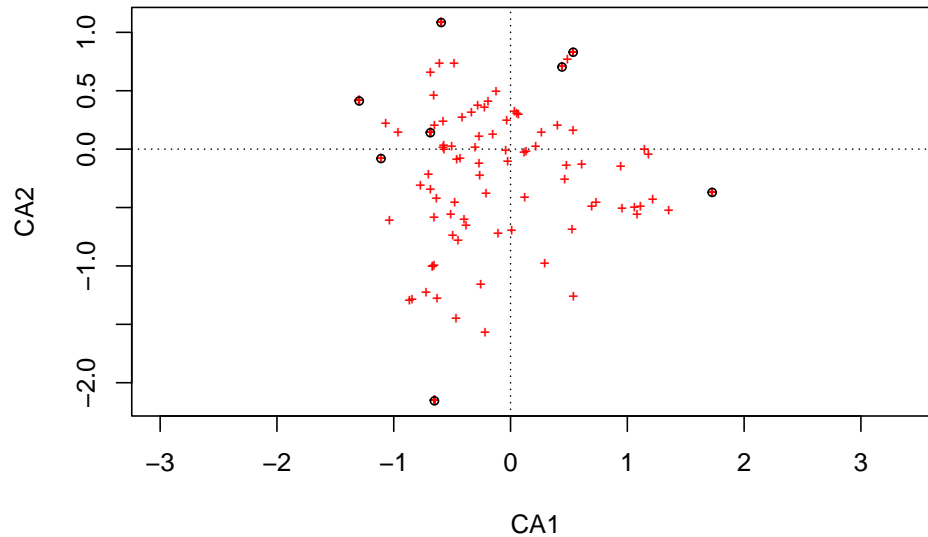


Figura 41: Plot de ordenación, muestra los patrones de las muestras y generos en el diagrama de ordenación. Los círculos representan las muestras y los signos + representan los géneros.

El siguiente análisis posterior a la CA utilizando la función `evplot()` es para ilustrar cómo decidir qué eje de CA debe utilizarse para la interpretación de los resultados. La sintaxis de la función `evplot()` es `evplot(ev)`, donde `ev` es un vector de valores propios. En primer lugar, ejecutamos la función.

```
evplot <- function(ev)
{# Broken stick model (MacArthur 1957)
  n <- length(ev)
  bsm <- data.frame(j=seq(1:n), p=0)
  bsm$p[1] <- 1/n
  for (i in 2:n) bsm$p[i] <- bsm$p[i-1] + (1/(n + 1 - i))
  bsm$p <- 100*bsm$p/n
  # Plot eigenvalues and% of variation for each axis
  op <- par(mfrow=c(2,1))
  barplot(ev, main="Eigenvalues", col="bisque", las=2)
  abline(h=mean(ev), col="red")
  legend("topright", "Average eigenvalue", lwd=1, col=2, bty="n")
  barplot(t(cbind(100*ev/sum(ev), bsm$p[n:1])), beside=TRUE,
          main="% variation", col=c("bisque",2), las=2)
  legend("topright", c("% eigenvalue", "Broken stick model"),
          pch=15, col=c("bisque",2), bty="n")
  par(op)
}
```



```
# Plot eigenvalues and% of variance for each axis
ev <- fecal_genus_cca$CA$eig

windows(title="CA eigenvalues")
evplot(ev)
```

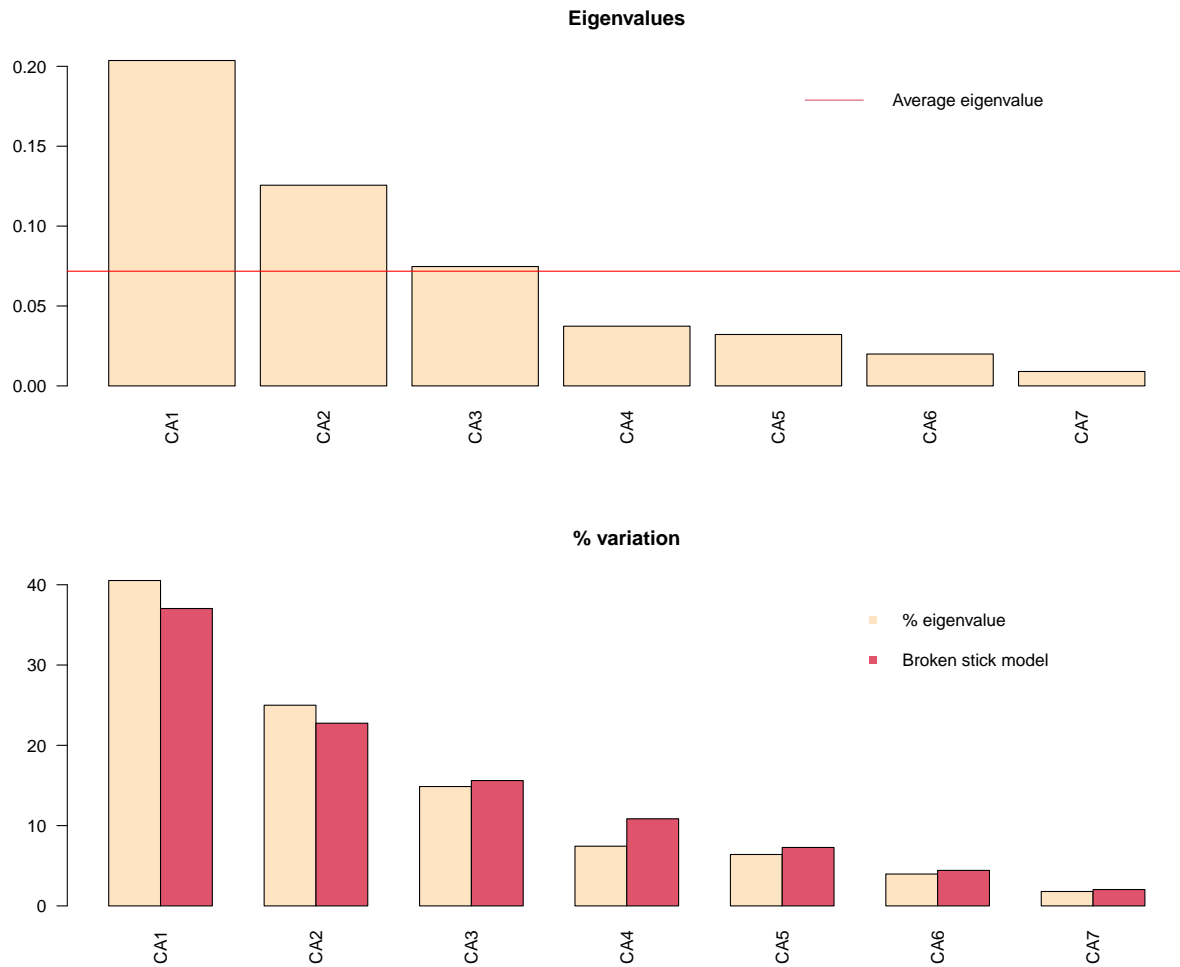


Figura 42: Eigenvalores y porcentajes de variación del criterio Keiser-Guttman y el modelo broken-stick

Como se presenta en la sección de PCoA, para evaluar si los primeros ejes de CA del análisis capturan una cantidad desproporcionadamente grande de la variación total explicada, dos vías adicionales de uso de gráficos son el *criterio de Keiser-Guttman* y ***criterio de broken-stick*. Según el criterio de Kaiser-Guttman, los valores propios asociados a los primeros ejes deben ser mayores que la media de todos los valores propios; con el modelo de broken-stick, los valores propios asociados a los primeros ejes se comparan con las expectativas. Tanto el criterio de Keiser-Guttman* como el modelo de broken-stick muestran que los tres primeros ejes son importantes.

4.2. Ordenaciones restringidas

Como contrapartida a los tres métodos básicos de ordenación, el paquete **vegan** cuenta con tres versiones de ordenación restringida:

- Análisis de redundancia (RDA, relacionado con el análisis de componentes principales)
- Análisis restringido de coordenadas principales (CAP, relacionado con el escalado métrico)
- Análisis restringido de correspondencia (CCA, relacionado con el análisis de correspondencia).

4.2.1. Análisis de redundancia (RDA)

El RDA en la función `rda()` se basa en las distancias euclidianas y combina la regresión múltiple con el PCA. Con RDA, cada eje canónico es una combinación lineal de todas las variables explicativas. El número de ejes canónicos corresponde al número de variables explicativas, o más exactamente al número de grados de libertades (Borcard et al. 2011).

Puede utilizar dos sintaxis diferentes para calcular una RDA mediante la función `rda()` de **vegan**: sintaxis matricial y sintaxis de fórmula. La sintaxis matricial es la más sencilla: basta con enumerar los nombres de los objetos separados por comas como se indica a continuación. `RDA = rda(Y, X, W)`, donde `Y` es la matriz de respuesta (composición de taxones), `X` es la matriz explicativa (factores ambientales) y `W` es la matriz opcional de covariables.

La sintaxis de fórmula esta dada como:

```
RDA = rda(Y ~ var1 + factor A + var2 * var3 + condition(var4), data = XW)
```

donde `Y` es la matriz de respuesta (composición de taxones); y las variables del lado derecho de la fórmula son las variables explicatorias o restringidas. El objeto `XW` es un dataframe con las variables explicatorias o covariables.

La prueba de hipótesis no es el caso en PCA. Sin embargo, con dos conjuntos de datos `Y` y `X`, en RDA podemos probar una hipótesis nula de ausencia de relación lineal entre ellos. Utilizamos el conjunto de datos de fumadores para ilustrar el uso de RDA. Primero, cargue los datos del paquete **GUniFrac** y utilice la función `select()` del paquete *dplyr* para crear un subconjunto de variables explicativas.

```
library("GUniFrac")
data(throat.otu.tab)
data(throat.meta)

library("dplyr")
throat_meta <- select(throat.meta, SmokingStatus, Age, Sex, PackYears)
```

A efectos de RDA, transformamos los datos de los taxones utilizando la transformación de Hellinger:

```
library("vegan") # if you haven't use it up to now
abund_hell <- decostand(throat.otu.tab, 'hell')
```

La RDA se calcula con la función `rda()` si se suministra la matriz de variables ambientales (si no, se calculará el ACP)

```
rda_hell <- rda(abund_hell ~ ., throat_meta)
head(summary(rda_hell))
```

```

##
## Call:
## rda(formula = abund_hell ~ SmokingStatus + Age + Sex + PackYears,      data = throat_meta)
##
## Partitioning of variance:
##           Inertia Proportion
## Total           0.4627      1.000
## Constrained      0.0482      0.104
## Unconstrained    0.4145      0.896
##
## Eigenvalues, and their contribution to the variance
##
## Importance of components:
##           RDA1  RDA2  RDA3  RDA4  PC1  PC2  PC3
## Eigenvalue      0.0239 0.0133 0.00597 0.00504 0.0683 0.0595 0.0355
## Proportion Explained 0.0517 0.0287 0.01290 0.01089 0.1476 0.1285 0.0768
## Cumulative Proportion 0.0517 0.0805 0.09335 0.10424 0.2518 0.3803 0.4572
##           PC4  PC5  PC6  PC7  PC8  PC9  PC10
## Eigenvalue      0.024 0.0174 0.0156 0.0149 0.0122 0.0103 0.00965
## Proportion Explained 0.052 0.0376 0.0337 0.0322 0.0263 0.0223 0.02086
## Cumulative Proportion 0.509 0.5467 0.5804 0.6126 0.6390 0.6613 0.68212
##           PC11  PC12  PC13  PC14  PC15  PC16  PC17
## Eigenvalue      0.00924 0.00876 0.00775 0.00671 0.00648 0.00626 0.00597
## Proportion Explained 0.01998 0.01893 0.01674 0.01450 0.01399 0.01352 0.01291
## Cumulative Proportion 0.70210 0.72103 0.73777 0.75226 0.76626 0.77978 0.79269
##           PC18  PC19  PC20  PC21  PC22  PC23  PC24
## Eigenvalue      0.00549 0.00529 0.0050 0.00459 0.00435 0.00400 0.00372
## Proportion Explained 0.01187 0.01144 0.0108 0.00993 0.00939 0.00865 0.00804
## Cumulative Proportion 0.80455 0.81599 0.8268 0.83673 0.84612 0.85477 0.86281
##           PC25  PC26  PC27  PC28  PC29  PC30  PC31
## Eigenvalue      0.00361 0.00357 0.00338 0.00323 0.00314 0.00307 0.00296
## Proportion Explained 0.00780 0.00771 0.00730 0.00698 0.00678 0.00663 0.00639
## Cumulative Proportion 0.87062 0.87832 0.88562 0.89260 0.89938 0.90601 0.91240
##           PC32  PC33  PC34  PC35  PC36  PC37  PC38
## Eigenvalue      0.00285 0.00268 0.00249 0.00232 0.00225 0.00218 0.00213
## Proportion Explained 0.00617 0.00579 0.00537 0.00502 0.00486 0.00471 0.00460
## Cumulative Proportion 0.91857 0.92435 0.92973 0.93475 0.93961 0.94431 0.94891
##           PC39  PC40  PC41  PC42  PC43  PC44  PC45
## Eigenvalue      0.00198 0.00186 0.00176 0.00168 0.00163 0.00155 0.00151
## Proportion Explained 0.00427 0.00403 0.00380 0.00364 0.00353 0.00335 0.00327
## Cumulative Proportion 0.95319 0.95721 0.96101 0.96465 0.96818 0.97153 0.97480
##           PC46  PC47  PC48  PC49  PC50  PC51  PC52
## Eigenvalue      0.00145 0.00139 0.00135 0.00123 0.00121 0.00117 0.00103
## Proportion Explained 0.00313 0.00301 0.00292 0.00266 0.00261 0.00253 0.00222
## Cumulative Proportion 0.97793 0.98094 0.98386 0.98652 0.98913 0.99165 0.99387
##           PC53  PC54  PC55
## Eigenvalue      0.00100 0.000943 0.000892
## Proportion Explained 0.00216 0.002037 0.001927
## Cumulative Proportion 0.99604 0.998073 1.000000
##
## Accumulated constrained eigenvalues
## Importance of components:
##           RDA1  RDA2  RDA3  RDA4
## Eigenvalue      0.0239 0.0133 0.00597 0.00504

```

```

## Proportion Explained  0.4961 0.2757 0.12374 0.10446
## Cumulative Proportion 0.4961 0.7718 0.89554 1.00000
##
## 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:  2.286
##
##
## Species scores
##
##          RDA1      RDA2      RDA3      RDA4      PC1      PC2
## 4695  0.001921 -0.011290 -0.007088 -0.002362 -0.000786 -0.000549
## 2983  0.002385 -0.001850 -0.004702  0.004402  0.000896 -0.003710
## 2554  0.001181 -0.005402 -0.004723 -0.001745 -0.005643 -0.004346
## 3315 -0.000150 -0.000566 -0.001440 -0.000609 -0.000684 -0.007786
## 879   -0.003149 -0.002243  0.000376 -0.002867 -0.000290  0.003117
## 1313  0.000261  0.003148  0.003128  0.000805  0.000496 -0.009699
## ....
##
##
## Site scores (weighted sums of species scores)
##
##          RDA1      RDA2      RDA3      RDA4      PC1      PC2
## ESC_1.1_OPL -0.444 -0.4795 -0.1746 -0.3520  0.0816 -0.1354
## ESC_1.3_OPL  0.601 -0.2263 -0.1055 -0.2759  0.1837 -0.0773
## ESC_1.4_OPL  0.657  0.1918 -0.1318  0.2111  0.3182  0.1287
## ESC_1.5_OPL  0.821  0.0243 -0.0463  0.5609  0.4416  0.3979
## ESC_1.6_OPL  0.980 -0.3117  0.4864 -0.0419  0.3708 -0.0187
## ESC_1.10_OPL 0.117  0.5722  0.8475 -0.2146 -0.3048  0.3063
## ....
##
##
## Site constraints (linear combinations of constraining variables)
##
##          RDA1      RDA2      RDA3      RDA4      PC1      PC2
## ESC_1.1_OPL -0.3062 -0.22035  0.0309 -0.3282  0.0816 -0.1354
## ESC_1.3_OPL  0.3242  0.05576 -0.1321  0.1074  0.1837 -0.0773
## ESC_1.4_OPL  0.3073  0.06338 -0.1988  0.1493  0.3182  0.1287
## ESC_1.5_OPL  0.0932 -0.50693 -0.3499  0.0537  0.4416  0.3979
## ESC_1.6_OPL  0.3660  0.00406  0.1952  0.0548  0.3708 -0.0187
## ESC_1.10_OPL 0.4675 -0.10112  0.8899 -0.1050 -0.3048  0.3063
## ....
##
##
## Biplot scores for constraining variables
##
##          RDA1      RDA2      RDA3      RDA4 PC1 PC2
## SmokingStatusSmoker 0.906 -0.3399 -0.135 0.2110  0  0
## Age                 0.228 -0.0442  0.748 0.6217  0  0
## SexMale             0.496  0.8360  0.103 0.2110  0  0
## PackYears           0.658 -0.1515  0.738 0.0222  0  0
##
##

```

```
## Centroids for factor constraints
##
##           RDA1    RDA2    RDA3    RDA4 PC1 PC2
## SmokingStatusNonSmoker -0.250  0.0938  0.0374 -0.0582  0  0
## SmokingStatusSmoker    0.286 -0.1072 -0.0427  0.0665  0  0
## SexFemale              -0.199 -0.3362 -0.0413 -0.0848  0  0
## SexMale                 0.107  0.1810  0.0222  0.0457  0  0
```

Los cuatro ejes de ordenación restringidos (denominados RDA1 a RDA4) están relacionados con las 4 variables ambientales (SmokingStatus, Age, Sex, PackYears). Los ejes no restringidos se denominan ejes PC.

Las varianzas totales se dividen en varianza restringida y varianza no restringida. La varianza restringida se explica por los ejes restringidos (es decir, las variables ambientales); mientras que la varianza no restringida se explica por los ejes no restringidos (es decir, la varianza no explicada por factores ambientales). La varianza restringida es la cantidad de varianza que la matriz Y es explicada por las variables explicativas. Es un equivalente a un R^2 sesgado y no ajustado en la regresión múltiple. La tabla de partición de la varianza muestra que el 4.8 % (0.0482) de la proporción de la varianza total es explicada por estos 4 factores ambientales.

La función `coef()` recupera los coeficientes canónicos (el equivalente a coeficientes de regresión) para cada variable explicativa en cada eje canónico.

```
coef(rda_hell)
```

```
##           RDA1    RDA2    RDA3    RDA4
## SmokingStatusSmoker 0.177040 -0.120402 -0.187134  0.15184
## Age                 -0.004452 -0.002161  0.002997  0.01757
## SexMale              0.092005  0.267873 -0.027579  0.01856
## PackYears            0.005257 -0.000520  0.011633 -0.01596
```

Podemos utilizar la función `RsquareAdj()` para extraer el valor de R^2 y el ajustado R^2 de los resultados de la ordenación.

```
RsquareAdj(rda_hell)
```

```
## $r.squared
## [1] 0.1042
##
## $adj.r.squared
## [1] 0.03909
```

La función devuelve dos R^2 : uno es el R^2 ordinario (r.al cuadrado), otro es el R^2_{adj} ajustado. Tenga en cuenta que R^2_{adj} es siempre menor que el R^2 , y la diferencia aumenta con el aumento del número de variables explicativas. El R^2_{adj} puede ser negativo, lo que significa que las variables explicativas explican menos variación que el mismo número de variables generadas al azar.

Aplicamos el criterio de Kaiser-Guttman a los ejes residuales.

```
rda_hell$CA$eig[rda_hell$CA$eig > mean(rda_hell$CA$eig)]
```

```
##      PC1      PC2      PC3      PC4      PC5      PC6      PC7      PC8
## 0.068299 0.059456 0.035542 0.024040 0.017406 0.015601 0.014898 0.012182
##      PC9      PC10     PC11     PC12     PC13
## 0.010314 0.009654 0.009245 0.008760 0.007745
```

A continuación, trazamos los resultados de la RDA. En función de la ordenación que le interese, puede elegir trazar el escalado tipo 1 o el escalado tipo 2. Para la *ordenación de las muestras*, el escalamiento 1 es la opción más adecuada. Para la *ordenación de los taxones*, entonces, el escalamiento 2 es el más apropiado.

El *escalamiento de tipo 1* hace hincapié en las relaciones entre las muestras. Las características esenciales (Borcard et al. 2011; Legendre y Legendre 2012) son que las muestras actúan como los centroides de las variables de respuesta (columnas) y las distancias entre los puntos de las muestras indican sus distancias χ^2 . La interpretación se basa en que:

- 1) Los puntos de muestra que están cerca unos de otros probablemente sean relativamente similares en cuanto a sus frecuencias relativas.
- 2) Las muestras cercanas a los centroides que representan estados de variables categóricas o cualitativas tienen más probabilidades de poseer el estado para esa especie/taxón.
- 3) Una proyección en ángulo recto de un punto de muestra sobre un vector que representa una variable cuantitativa se aproxima al valor de la variable realizada para esa muestra.

El *escalamiento de tipo 2* hace hincapié en las relaciones entre las variables de respuesta. Las características esenciales son que las variables de respuesta (columnas) actúan como los centroides de las muestras y las distancias entre los puntos de las variables de respuesta indican sus distancias χ^2 . La interpretación se basa en que:

- 1) Los puntos de especies/taxones que están cerca unos de otros entre sí es probable que tengan frecuencias relativas similares a lo largo de las muestras.
- 2) Cuanto más cerca una variable de respuesta (una especie/taxón) al centroide que representa un estado de una categórica, es más probable que esa variable de respuesta tenga valores más altos en ese estado.
- 3) Una proyección en ángulo recto de un punto que representa una respuesta (una especie/taxón) sobre una flecha que representa una variable explicativa indica la posición del valor máximo (el óptimo) de la variable de respuesta a lo largo de esa variable explicativa.

En la ordenación restringida, se dispone de una fuente de datos adicional para el trazado: *las variables explicativas (ambientales)*. Así, se dispone de tres datos diferentes para muestras, variables de respuesta y variables explicativas. Si se muestran todos los datos de las tres fuentes, se tiene un **“triplot”**. Si muestra los datos de dos fuentes, tiene **biplot**. Los siguientes códigos de R generan un triplot de escala de tipo 2:

```
plot(rda_hell, display=c("sp", "lc", "cn"), main="Triplot RDA - scaling 2")
taxa_scores <- scores(rda_hell, choices=c(1,2), display="sp")
arrows(0, 0, taxa_scores[,1], taxa_scores[,2], length=0, lty=1, col="red")
```

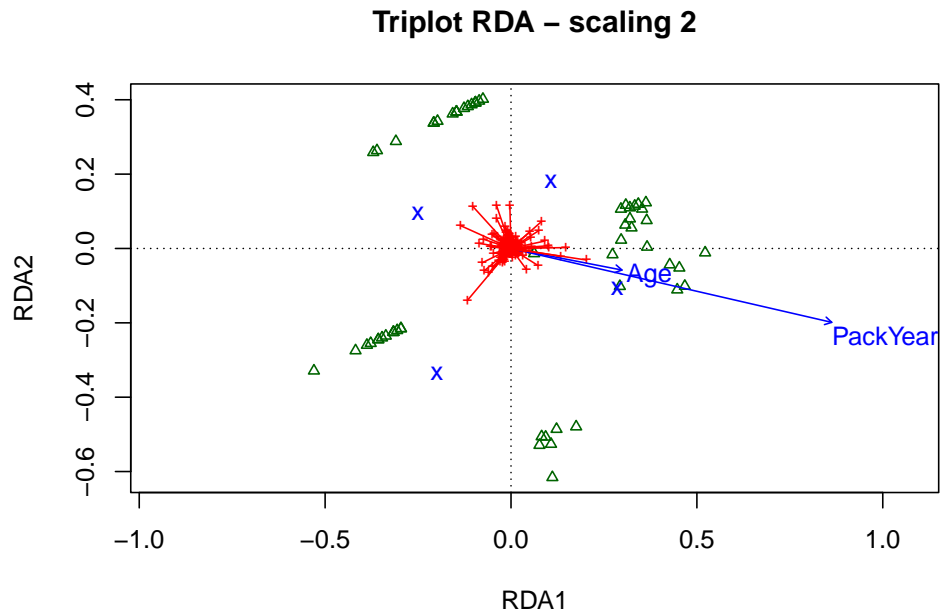


Figura 43: Triplot RDA con escalado tipo 2

En los códigos anteriores, “sp” representa las especies (`display = "sp"`), y “cn” las restricciones o las variables explicativas (`display = "cn"`). Hay dos puntuaciones muestrales en el paquete `vegan`: sumas ponderadas de especies/taxa (`display = "wa"`), y puntuaciones muestrales ajustadas o “LC scores” (`display = "lc"`), es decir, combinaciones lineales de variables explicativas. Se puede elegir una de ellas para mostrarla en un gráfico. La segunda línea de código de R se utilizan para añadir flechas para mostrar especies/taxa. El argumento de la función `scores()` `choices=c(1,2)` es los ejes a trazar. Los valores por defecto trazan los ejes 1 y 2. Por lo tanto, podemos omitir la opción `choices=c(1,2)`. En el gráfico 43, los triángulos huecos verdes representan muestras, las cruces azules representan los estados de una variable explicativa categórica (por ejemplo, hombre, mujer o fumador, no fumador), y las flechas azules para las variables explicativas cuantitativas (en este caso, PackYears y Edad) con puntas de flecha que indican su dirección de aumento, y las especies/taxas se muestran en rojo.

Para comprobar la importancia de la variación de los datos de la comunidad de fumadores explicada por las variables explicativas, podemos realizar una prueba de permutación de Monte Carlo mediante la función `anova.cca()` del paquete `vegan`. Esta función puede probar la significancia del modelo global (por defecto), todos los ejes (`by = ".axis"`), las variables explicativas individuales (`by = "terms"`), el primer eje restringido (`first = TRUE`), o la variación explicada por variables explicativas individuales después de eliminar la variación de todas las demás variables en el modelo (`by = "margin"`). Ilustramos su capacidad probando el modelo global, cada eje y cada variable explicativa, respectivamente.

Para asegurarnos de obtener el mismo resultado de la prueba de permutación cada vez que se ejecute, establecemos primero la misma semilla. Los siguientes códigos R prueban la significación del modelo global. El argumento “paso” especifica el número mínimo de permutaciones.

```
set.seed(123)
anova(rda_hell, step=1000) %>%
  kbl(booktabs = TRUE, align = "c",
      caption = "ANOVA global") %>%
  kable_styling(position = "center",
```

```
latex_options = c("hold_position", "striped"),
font_size = 9)
```

Cuadro 4: ANOVA global

	Df	Variance	F	Pr(>F)
Model	4	0.0482	1.6	0.004
Residual	55	0.4145	NA	NA

Podemos observar que el test global es estadísticamente significativo. Ahora, vamos a ver el test en cada eje.

```
set.seed (123)
anova(rda_hell, by="axis", step=1000)%>%
  kbl(booktabs = TRUE, align = "c",
      caption = "ANOVA test para todos los ejes")%>%
  kable_styling(position = "center",
                latex_options = c("hold_position", "striped"),
                font_size = 9)
```

Cuadro 5: ANOVA test para todos los ejes

	Df	Variance	F	Pr(>F)
RDA1	1	0.0239	3.1754	0.013
RDA2	1	0.0133	1.7644	0.256
RDA3	1	0.0060	0.7920	0.942
RDA4	1	0.0050	0.6686	0.876
Residual	55	0.4145	NA	NA

Podemos observar que el primer y segundo eje son significativos. El siguiente código es el test de significancia para cada variable explicativa.

```
set.seed (123)
anova(rda_hell, by="terms", step=1000)%>%
  kbl(booktabs = TRUE, align = "c",
      caption = "ANOVA test para variables explicativas")%>%
  kable_styling(position = "center",
                latex_options = c("hold_position", "striped"),
                font_size = 9)
```

Cuadro 6: ANOVA test para variables explicativas

	Df	Variance	F	Pr(>F)
SmokingStatus	1	0.0215	2.8571	0.002
Age	1	0.0057	0.7512	0.773
Sex	1	0.0144	1.9171	0.019
PackYears	1	0.0066	0.8751	0.588
Residual	55	0.4145	NA	NA

Aquí podemos observar que tanto **SmokingStatus** como **Sex** son significativas.

Por último, utilizamos la selección directa para reducir el número de variables explicativas que entran en el análisis, mientras se optimiza la variación explicada por ellas.

Actualmente, hay tres funciones disponibles en RDA para la selección hacia adelante: `ordistep()`, `ordiR2step()`, y `forward.sel()`. Las dos primeras funciones son del paquete `vegan`. La tercera, está disponible en el paquete `adespatial`. Estas funciones utilizan diferentes criterios para la selección de variables. La función `ordistep()` utiliza el criterio AIC y los p -valores de la prueba de permutación de Monte Carlo para la comparación de variables. La función `ordiR2step()` utiliza R^2_{adj} . La función `forward.sel()` utiliza el nivel de significación preseleccionado de α como criterio de selección de variables.

Aunque la función `forward.sel()` tiene una lógica diferente para establecer los argumentos comparada con las otras dos funciones, básicamente los resultados devueltos son los mismos que `ordiR2step()`. Por lo tanto, sólo ilustramos las dos primeras funciones. El procedimiento `ordistep()` es aplicable con las funciones `rda()`, `cca()` o `cmdscale()`. El procedimiento `ordiR2step()` sólo puede aplicarse a `rda()` y `capscale()`, pero no para `cca()` porque `cca()` no devuelve R^2_{adj} .

Los siguientes códigos de R utilizan la función `ordistep()` para ejecutar la selección hacia adelante. También se dispone de opciones para la selección por pasos y hacia atrás. Esta función permite el uso de factores.

```
step_forward <- ordistep(rda(abund_hell ~ 1, data=throat_meta),
                        scope=formula(rda_hell), direction="forward", pstep=1000)
```

```
##
## Start: abund_hell ~ 1
##
##              Df    AIC      F Pr(>F)
## + SmokingStatus 1 -46.1 2.83 0.005 **
## + Sex            1 -45.3 2.01 0.015 *
## + PackYears      1 -45.1 1.80 0.050 *
## + Age            1 -44.1 0.83 0.570
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Step: abund_hell ~ SmokingStatus
##
##              Df    AIC      F Pr(>F)
## + Sex            1 -46.0 1.87 0.04 *
## + PackYears      1 -45.0 0.87 0.63
## + Age            1 -44.9 0.74 0.78
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Step: abund_hell ~ SmokingStatus + Sex
##
##              Df    AIC      F Pr(>F)
## + PackYears      1 -45.0 0.85 0.66
## + Age            1 -44.9 0.81 0.66
```

Este procedimiento elige las variables `SmokingStatus` y `Sex`. La siguiente selección utiliza la función `ordiR2step()`. La configuración por defecto de este procedimiento para incluir una nueva variable se basa en R^2_{adj} y su comparación con la del modelo global (con todas las variables). La selección se detendrá si la nueva variable no es significativa o el R^2_{adj} del modelo que incluye esta nueva variable superaría el R^2_{adj} del modelo global.

```
step_forward <- ordiR2step(rda(abund_hell ~ 1, data=throat_meta),
                           scope=formula(rda_hell), direction="forward", pstep=1000)
```

```
## Step: R2.adj= 0
## Call: abund_hell ~ 1
##
##               R2.adjusted
## <All variables> 0.039094
## + SmokingStatus 0.030093
## + Sex           0.016765
## + PackYears     0.013326
## <none>          0.000000
## + Age           -0.002834
##
##           Df   AIC    F Pr(>F)
## + SmokingStatus 1 -46.1 2.83 0.004 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Step: R2.adj= 0.03009
## Call: abund_hell ~ SmokingStatus
##
##               R2.adjusted
## + Sex           0.04449
## <All variables> 0.03909
## <none>          0.03009
## + PackYears     0.02786
## + Age           0.02574
```

La variable estado de fumador se selecciona mediante la selección de avance utilizando la función `ordiR2step()`. Después de ejecutar el procedimiento de selección hacia adelante, ajustamos el modelo final parsimonioso de RDA y probamos con el modelo global, cada eje y cada variable:

```
rda_final<- rda(abund_hell ~ SmokingStatus + Sex, data=throat_meta)
```

```
anova(rda_final, step=1000)%>%
  kbl(booktabs = TRUE, align = "c",
      caption = "ANOVA test global")%>%
  kable_styling(position = "center",
                latex_options = c("hold_position", "striped"),
                font_size = 9)
```

Cuadro 7: ANOVA test global

	Df	Variance	F	Pr(>F)
Model	2	0.0356	2.374	0.001
Residual	57	0.4271	NA	NA

```
anova(rda_final, by="axis", step=1000)%>%
  kbl(booktabs = TRUE, align = "c",
      caption = "ANOVA test para ejes")%>%
  kable_styling(position = "center",
               latex_options = c("hold_position", "striped"),
               font_size = 9)
```

Cuadro 8: ANOVA test para ejes

	Df	Variance	F	Pr(>F)
RDA1	1	0.0226	3.010	0.002
RDA2	1	0.0130	1.737	0.041
Residual	57	0.4271	NA	NA

```
anova(rda_final, by="terms", step=1000)%>%
  kbl(booktabs = TRUE, align = "c",
      caption = "ANOVA test para variables explicativas")%>%
  kable_styling(position = "center",
               latex_options = c("hold_position", "striped"),
               font_size = 9)
```

Cuadro 9: ANOVA test para variables explicativas

	Df	Variance	F	Pr(>F)
SmokingStatus	1	0.0215	2.873	0.001
Sex	1	0.0140	1.874	0.024
Residual	57	0.4271	NA	NA

4.2.2. Análisis de correspondencia restringido (CCA)

El CCA también se conoce como análisis de correspondencia canónica. Desde su introducción en 1986, ha sido uno de los métodos de ordenación más populares en ecología de comunidades y aceptado por los investigadores del microbioma (ter Braak 1986). Al igual que el **RDA se relaciona con el PCA, el CAP se relaciona con el PCoA, y el CCA se relaciona con el CA**. CCA comparte las propiedades básicas de CA y las combina en una ordenación restringida. CCA se realiza mediante la función `cca()`. Su algoritmo se basa en el de Legendre y Legendre (1998): preserva la distancia χ^2 entre muestras, y los taxones se representan como puntos en los triplots (Borcard et al. 2011). La distancia χ^2 calculada se somete a una regresión lineal ponderada sobre las variables de restricción, y los valores ajustados se pasan al análisis de correspondencia realizado mediante la descomposición de valores singulares (svd). Por lo tanto, es una forma ponderada de RDA.

Al igual que RDA, hay dos tipos de sintaxis de CCA. Una es la sintaxis matricial simple (por defecto)

$$cca(X, Y, Z)$$

donde X = matriz de datos de la comunidad o marco de datos, se debe dar; Y = matriz o marco de datos de restricción, normalmente de variables ambientales (puede omitirse); si la matriz Y se suministra, se utiliza para llevar a cabo el CCA, si no se suministra, se calcula el AC. Z = matriz condicionante o marco de datos

(también puede omitirse). Si se suministra la matriz Z , el efecto se partirá de la matriz de la comunidad de la comunidad.

Otra es la sintaxis de la fórmula:

$$cca(formula, data, na.action = na.fail, subset = NULL)$$

donde, la fórmula es la fórmula típica del modelo: el lado izquierdo de la fórmula debe ser la matriz de datos de la comunidad (X); el lado derecho define el modelo de restricción: las variables de restricción pueden contener factores ordenados o desordenados, interacciones entre las variables y funciones de las variables; y las variables condicionantes pueden estar dadas dentro de una condición de función especial para las variables condicionantes (covariables) parcializadas antes del análisis. Así, los siguientes comandos son equivalentes: $cca(X, Y, Z)$, $cca(X \sim Y + condition(Z))$, donde Y y Z se refieren a las restricciones y las condiciones respectivamente. Los datos son un marco de datos que contiene las variables del lado derecho de la fórmula del modelo. La función `na.action()` se utiliza para tratar los valores perdidos en las restricciones o condiciones. El valor por defecto (`na.fail`) es dejar de tener valores perdidos; `na.omit` es eliminar todas las filas con valores perdidos; `na.exclude` es mantener todas las observaciones pero dar NA para los resultados que no se pueden calcular. Sin embargo, los valores perdidos nunca se permiten en los datos de las comunidades dependientes. El subconjunto se utiliza para subconjuntos de filas de datos. En esta sección, utilizaremos los datos de los fumadores para realizar el CCA mediante la función `cca()` del paquete `vegan`. Hemos mencionado anteriormente que para realizar el CCA, la matriz de variables ambientales debe ser suministrada, y de lo contrario la función `cca()` calcula CA. Como recordamos, los datos de los fumadores tienen dos conjuntos de datos: `throat.otu.tab` (marco de datos de abundancia de la comunidad) y `throat.meta` (metadatos que incluyen dos variables binarias `SmokingStatus` y `Sex`, y dos variables continuas `Age` y `PackYears`). Para ejecutar un CCA, cargamos el paquete `vegan` y no transformamos los datos de abundancia de la comunidad por el método Hellinger como hicimos en RDA. De lo contrario, la distancia χ^2 no puede ser calculada y los resultados no pueden ser interpretados.

En el siguiente CCA, los datos de abundancia de fumadores están restringidos por cuatro variables ambientales ambientales en los datos de `throat.meta`, incluyendo `SmokingStatus`, `Age`, `Sex`, y `PackYears`.

```
smoker_cca <- cca(throat.otu.tab ~ ., throat.meta)
smoker_cca

## Call: cca(formula = throat.otu.tab ~ SmokingStatus + Age + Sex +
## PackYears, data = throat.meta)
##
##              Inertia Proportion Rank
## Total          4.9330      1.0000
## Constrained    0.3782      0.0767    4
## Unconstrained  4.5548      0.9233   55
## Inertia is scaled Chi-square
##
## Eigenvalues for constrained axes:
##   CCA1   CCA2   CCA3   CCA4
## 0.1517 0.0912 0.0781 0.0572
##
## Eigenvalues for unconstrained axes:
##   CA1   CA2   CA3   CA4   CA5   CA6   CA7   CA8
## 0.457 0.358 0.325 0.299 0.237 0.187 0.165 0.160
## (Showing 8 of 55 unconstrained eigenvalues)
```

La primera sección después de la “Call:” da los *coeficientes de contingencia media al cuadrado* del análisis. En el CCA desempeñan el mismo papel que la varianza total en la RDA. Como la matriz de comunidad se

convierte en distancia χ^2 , las entradas son coeficientes de contingencia. La variación total antes de someter la matriz a regresión ponderada es de 4.933; ésta es la variación que podría explicarse. La variación en la matriz de la comunidad que se explica después de la regresión ponderada es 0.3782; esta es la variación que será explicada por los ejes en el CCA. La varianza de los residuos de la regresión es de 4.5548; se trata de la variación que no será explicada por los ejes en el CCA, que puede ser sometida a CA.

Aquí, observamos que el CCA no tiene mucho éxito porque sólo 0.0767 o 0,0767 (véase la columna Proporción y la fila Restringsida) de la variación total de datos fue capturada en el CCA por las cuatro variables. La varianza explicada por ejes particulares puede encontrarse mediante la función `summary()`:

```
head(summary(smoker_cca))
```

```
##
## Call:
## cca(formula = throat.otu.tab ~ SmokingStatus + Age + Sex + PackYears,      data = throat_meta)
##
## Partitioning of scaled Chi-square:
##              Inertia Proportion
## Total          4.933      1.0000
## Constrained     0.378      0.0767
## Unconstrained   4.555      0.9233
##
## Eigenvalues, and their contribution to the scaled Chi-square
##
## Importance of components:
##              CCA1  CCA2  CCA3  CCA4  CA1  CA2  CA3  CA4
## Eigenvalue      0.1517 0.0912 0.0781 0.0572 0.4567 0.3581 0.3247 0.2989
## Proportion Explained 0.0308 0.0185 0.0158 0.0116 0.0926 0.0726 0.0658 0.0606
## Cumulative Proportion 0.0308 0.0492 0.0651 0.0767 0.1692 0.2418 0.3076 0.3682
##              CA5  CA6  CA7  CA8  CA9  CA10  CA11  CA12
## Eigenvalue      0.2373 0.1865 0.1653 0.1604 0.1280 0.1207 0.1189 0.1101
## Proportion Explained 0.0481 0.0378 0.0335 0.0325 0.0259 0.0245 0.0241 0.0223
## Cumulative Proportion 0.4163 0.4542 0.4877 0.5202 0.5461 0.5706 0.5947 0.6170
##              CA13  CA14  CA15  CA16  CA17  CA18  CA19  CA20
## Eigenvalue      0.1097 0.0999 0.0904 0.0859 0.0831 0.0783 0.0726 0.0665
## Proportion Explained 0.0222 0.0203 0.0183 0.0174 0.0168 0.0159 0.0147 0.0135
## Cumulative Proportion 0.6392 0.6595 0.6778 0.6952 0.7120 0.7279 0.7426 0.7561
##              CA21  CA22  CA23  CA24  CA25  CA26  CA27
## Eigenvalue      0.0637 0.0612 0.0591 0.0555 0.0531 0.0499 0.04870
## Proportion Explained 0.0129 0.0124 0.0120 0.0113 0.0108 0.0101 0.00987
## Cumulative Proportion 0.7690 0.7814 0.7934 0.8047 0.8154 0.8256 0.83543
##              CA28  CA29  CA30  CA31  CA32  CA33  CA34
## Eigenvalue      0.04757 0.04608 0.04326 0.04260 0.04030 0.03908 0.03808
## Proportion Explained 0.00964 0.00934 0.00877 0.00864 0.00817 0.00792 0.00772
## Cumulative Proportion 0.84507 0.85441 0.86318 0.87182 0.87999 0.88791 0.89563
##              CA35  CA36  CA37  CA38  CA39  CA40  CA41
## Eigenvalue      0.03580 0.03498 0.0340 0.0331 0.03105 0.03089 0.02951
## Proportion Explained 0.00726 0.00709 0.0069 0.0067 0.00629 0.00626 0.00598
## Cumulative Proportion 0.90289 0.90998 0.9169 0.9236 0.92987 0.93613 0.94211
##              CA42  CA43  CA44  CA45  CA46  CA47  CA48
## Eigenvalue      0.02793 0.02646 0.02574 0.02359 0.02256 0.02217 0.02092
## Proportion Explained 0.00566 0.00536 0.00522 0.00478 0.00457 0.00449 0.00424
## Cumulative Proportion 0.94778 0.95314 0.95836 0.96314 0.96771 0.97221 0.97645
##              CA49  CA50  CA51  CA52  CA53  CA54  CA55
```

```

## Eigenvalue          0.02013 0.0192 0.01798 0.01757 0.01556 0.01342 0.01228
## Proportion Explained 0.00408 0.0039 0.00365 0.00356 0.00316 0.00272 0.00249
## Cumulative Proportion 0.98053 0.9844 0.98807 0.99164 0.99479 0.99751 1.00000
##
## Accumulated constrained eigenvalues
## Importance of components:
##           CCA1   CCA2   CCA3   CCA4
## Eigenvalue    0.152 0.0912 0.0781 0.0572
## Proportion Explained 0.401 0.2411 0.2065 0.1512
## Cumulative Proportion 0.401 0.6422 0.8488 1.0000
##
## 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   CA1   CA2
## 4695  0.0116  1.177  0.475  0.1308  0.0491  0.0182
## 2983  0.5220  0.510  1.025  1.2222  0.1317  0.6923
## 2554  0.0277  1.202  0.666  0.0181 -0.1559  1.0291
## 3315 -0.0679  0.382  0.474 -0.1129 -0.4215  3.1688
## 879   -1.1588  0.745 -0.160 -0.8216 -0.6967 -0.5789
## 1313 -0.0124 -1.048 -0.370  0.0333 -0.2758  3.2476
## ....
##
##
## Site scores (weighted averages of species scores)
##
##           CCA1   CCA2   CCA3   CCA4   CA1   CA2
## ESC_1.1_OPL -2.510  2.1425  0.415 -2.318203  2.6670 -0.6690
## ESC_1.3_OPL  1.309 -0.0894 -0.376 -0.176963  0.0602 -0.3029
## ESC_1.4_OPL  1.445 -0.0145 -0.364  0.078627  0.2178 -0.6817
## ESC_1.5_OPL  1.538  0.4089 -0.589  0.896660 -0.4024 -1.2717
## ESC_1.6_OPL  2.579  0.8192 -2.732  0.254050  0.2062 -0.5353
## ESC_1.10_OPL 0.385 -0.9651 -0.891  0.000538 -0.1476 -0.0726
## ....
##
##
## Site constraints (linear combinations of constraining variables)
##
##           CCA1   CCA2   CCA3   CCA4   CA1   CA2
## ESC_1.1_OPL -1.125 0.7768 -0.132 -0.961  2.6670 -0.6690
## ESC_1.3_OPL  1.123 0.0630  0.341  0.414  0.0602 -0.3029
## ESC_1.4_OPL  1.067 0.0273  0.578  0.567  0.2178 -0.6817
## ESC_1.5_OPL -0.029 1.8898  0.889  0.728 -0.4024 -1.2717
## ESC_1.6_OPL  1.261 0.1173 -0.834  0.195  0.2062 -0.5353
## ESC_1.10_OPL 1.598 0.2708 -3.324 -0.438 -0.1476 -0.0726
## ....
##
##
## Biplot scores for constraining variables
##

```

```
##          CCA1  CCA2  CCA3  CCA4 CA1 CA2
## SmokingStatusSmoker 0.831  0.452  0.0592  0.3183  0  0
## Age                 0.202 -0.178 -0.7739  0.5730  0  0
## SexMale             0.682 -0.730  0.0444  0.0112  0  0
## PackYears          0.615  0.163 -0.7705  0.0307  0  0
##
##
## Centroids for factor constraints
##
##          CCA1  CCA2  CCA3  CCA4 CA1 CA2
## SmokingStatusNonSmoker -0.746 -0.406 -0.0531 -0.28569  0  0
## SmokingStatusSmoker    0.926  0.503  0.0660  0.35459  0  0
## SexFemale              -0.922  0.985 -0.0599 -0.01516  0  0
## SexMale                 0.505 -0.540  0.0328  0.00831  0  0
```

La varianza explicada por los cuatro primeros ejes restringidos (CCA1, CCA2, CCA3, CCA4): 3,07, 1,85, 1,58 y 1,16 %, respectivamente. La varianza explicada por el primer eje no restringido (CCA1) es del 9,26 %. La sección de “Valores propios restringidos acumulados” proporciona los valores propios asociados a la proyección. Como tenemos cuatro variables en nuestro marco de datos ambiental, hay cuatro valores propios limitados. Como se muestra aquí, el primer eje representa aproximadamente el 40 % de la variación restringida, con el segundo eje el segundo eje con un 24 %, el tercero con un 21 % y el cuarto con un 15 %.

A continuación, trazamos los resultados del CCA para el escalado de tipo 1 utilizando la función `plot()`. En el gráfico, queremos mostrar las restricciones lineales o “puntuaciones LC” (`display = "lc"`) y los centroides de los niveles de las variables factoriales (`display = "cn"`).

```
plot(smoker_cca, scaling=1, display = c("lc","cn"), main="Biplot CCA - scaling 1")
```

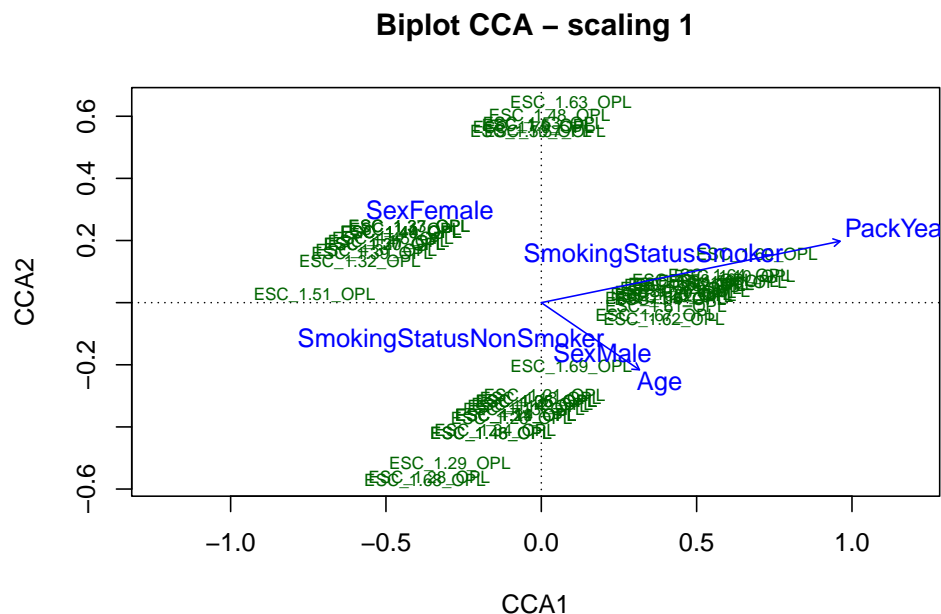


Figura 44: CCA biplot de la abundancia de la base smoker.throat restringido por las variables SmokingStatus, Age, Sex, PackYear con escalado tipo 1

La escala de tipo 1 muestra cuatro grupos de muestras, con el grupo de fumadores vinculado a PackYears. En este análisis, el primer eje se asocia con el aumento de PackYears mientras que el segundo está asociado con la disminución de la Edad.

Al igual que en RDA, realizamos una prueba de permutación para el modelo global, cada eje y cada variable explicativa. Para asegurarse de obtener el mismo resultado de la prueba de permutación cada vez que se ejecute, establezca la misma semilla.

```
set.seed(123)
anova(smoker_cca, step=1000) %>%
  kbl(booktabs = TRUE, align = "c",
      caption = "ANOVA test global: El resultado no es estadísticamente significativo") %>%
  kable_styling(position = "center",
               latex_options = c("hold_position", "striped"),
               font_size = 9)
```

Cuadro 10: ANOVA test global: El resultado no es estadísticamente significativo

	Df	ChiSquare	F	Pr(>F)
Model	4	0.3782	1.142	0.14
Residual	55	4.5548	NA	NA

Como el resultado anterior no es estadísticamente significativo, entonces realizamos una prueba de significancia por cada eje.

```
set.seed(123)
anova(smoker_cca, by = 'axis', step=1000) %>%
  kbl(booktabs = TRUE, align = "c",
      caption = "ANOVA test por ejes: El primer eje es significativo") %>%
  kable_styling(position = "center",
               latex_options = c("hold_position", "striped"),
               font_size = 9)
```

Cuadro 11: ANOVA test por ejes: El primer eje es significativo

	Df	ChiSquare	F	Pr(>F)
CCA1	1	0.1517	1.8318	0.093
CCA2	1	0.0912	1.1009	0.824
CCA3	1	0.0781	0.9432	0.853
CCA4	1	0.0572	0.6906	0.941
Residual	55	4.5548	NA	NA

Ahora probamos la significancia de cada variable explicativa.

```
set.seed(123)
anova(smoker_cca, by = 'terms') %>%
  kbl(booktabs = TRUE, align = "c",
      caption = "ANOVA test por variables explicativas: SmokingStatus es significativo con un p-value de 0.009") %>%
  kable_styling(position = "center",
               latex_options = c("hold_position", "striped"),
               font_size = 9)
```


Cuadro 12: ANOVA test por variables explicativas: SmokingStatus es significativo con un p-value de 0.009

	Df	ChiSquare	F	Pr(>F)
SmokingStatus	1	0.1295	1.5639	0.008
Age	1	0.0737	0.8895	0.676
Sex	1	0.1049	1.2663	0.078
PackYears	1	0.0701	0.8468	0.700
Residual	55	4.5548	NA	NA

Por último, vamos a hacer una selección hacia delante utilizando la función `ordistep()` del paquete `vegan`. Como dijimos en RDA, la función `ordistep()` es aplicable con las funciones `rda()`, `cca()` o `cmdscale()`. La comparación de variables se basa en el criterio AIC y en los valores p de la prueba de permutación de Monte Carlo.

```
ordistep(cca(throat.otu.tab ~ 1, data=throat_meta), scope=formula(smoker_cca),
          direction="forward", pstep=1000)
```

```
##
## Start: throat.otu.tab ~ 1
##
##              Df AIC      F Pr(>F)
## + SmokingStatus 1 539 1.56 0.025 *
## + Sex           1 539 1.44 0.030 *
## + PackYears     1 539 1.28 0.190
## + Age           1 540 0.89 0.585
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Step: throat.otu.tab ~ SmokingStatus
##
##              Df AIC      F Pr(>F)
## + Sex         1 540 1.28 0.06 .
## + PackYears   1 540 0.97 0.51
## + Age         1 540 0.89 0.66
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Call: cca(formula = throat.otu.tab ~ SmokingStatus, data =
## throat_meta)
##
##              Inertia Proportion Rank
## Total         4.9330      1.0000
## Constrained   0.1295      0.0263    1
## Unconstrained 4.8035      0.9737   58
## Inertia is scaled Chi-square
##
## Eigenvalues for constrained axes:
##   CCA1
## 0.1295
##
## Eigenvalues for unconstrained axes:
##   CA1  CA2  CA3  CA4  CA5  CA6  CA7  CA8
```

```
## 0.484 0.383 0.332 0.302 0.249 0.188 0.177 0.161
## (Showing 8 of 58 unconstrained eigenvalues)
```

```
smoker_cca_final <- cca(throat.otu.tab ~ SmokingStatus, data=throat_meta)
anova.cca(smoker_cca_final, step=1000)%>%
  kbl(booktabs = TRUE, align = "c",
      caption = "ANOVA test global")%>%
  kable_styling(position = "center",
      latex_options = c("hold_position", "striped"),
      font_size = 9)
```

Cuadro 13: ANOVA test global

	Df	ChiSquare	F	Pr(>F)
Model	1	0.1295	1.564	0.004
Residual	58	4.8035	NA	NA

```
anova.cca(smoker_cca_final, step=1000, by="terms") %>%
  kbl(booktabs = TRUE, align = "c",
      caption = "ANOVA test por variables explicativass")%>%
  kable_styling(position = "center",
      latex_options = c("hold_position", "striped"),
      font_size = 9)
```

Cuadro 14: ANOVA test por variables explicativass

	Df	ChiSquare	F	Pr(>F)
SmokingStatus	1	0.1295	1.564	0.004
Residual	58	4.8035	NA	NA

La condición de fumador alcanza el mismo nivel de significación con un p -valor de 0,007. Sin embargo, el modelo de parsimonia ha dado sus frutos con un residuo mayor.

4.2.3. Análisis restringido de coordenadas principales (CAP)

CAP (también llamado análisis restringido de proximidades en el paquete **vegan**), es un método de ordenación similar a RDA. Es simplemente un *análisis de redundancia de los resultados del análisis de coordenadas principales (o escalamiento multidimensional métrico)* (Anderson y Willis 2003). CAP permite índices de disimilitud no euclidianos, como la distancia Manhattan o Bray-Curtis. Si se especifica la distancia euclidiana como método de ordenación, los resultados serán idénticos a los de RDA. La función **capscale()** del paquete **vegan** se utiliza para implementar CAP. Necesita una matriz de disimilitud como conjunto de datos de entrada, que puede calcularse mediante las funciones **vegdist()**, **dist()**, o cualquier otro método que produzca matrices de similitud. Se necesitan dos pasos necesarios: primero, utiliza la función **cmdscale()** para ordenar la matriz de disimilitud, luego utiliza RDA para analizar estos resultados. A diferencia de RDA, en el que se pueden utilizar tanto la matriz como la la función **capscale()** sólo se puede llamar con la sintaxis de la fórmula. A continuación se muestra un uso de la función **capscale()**.

```
capscale(formula, data, distance = "bray", dfun = vegdist)
```

donde, formula es una fórmula típica del modelo como se define en `rda()` y `cca()`. El lado izquierdo de la fórmula debe ser una matriz de datos de la comunidad (marco) o una matriz de disimilitud que puede estimarse con la función `vegdist()` o `dist()`. Si el lado izquierdo de la fórmula es una matriz de datos (marco) en lugar de una matriz de disimilitud, entonces se estimará una índice de disimilitud (o distancia) debe proporcionarse como entrada de la distancia. El lado derecho de la fórmula define las restricciones. Las variables de restricción pueden ser variables continuas, factores, términos de interacción o una condición de término especial utilizada como que definen las variables que se van a particionar. Los datos son un marco de datos que contiene las variables del lado derecho de la fórmula del modelo. La *dfun* es la función de distancia o función de disimilitud utilizada. El CAP básico se puede realizar con los siguientes códigos. Las variables de restricción incluyen dos variables binarias (SmokingStatus y Sex), y dos variables continuas (PackYears y Edad). La condición de término especial (Edad) se utiliza para particionar el efecto de la edad.

```
throat_meta <- select(throat.meta, SmokingStatus, Age, Sex, PackYears)
throat_cap <- capscale(throat.otu.tab ~ SmokingStatus + Sex + PackYears + Condition(Age), throat_meta,
                      dist="bray")
throat_cap
```

```
## Call: capscale(formula = throat.otu.tab ~ SmokingStatus + Sex +
## PackYears + Condition(Age), data = throat_meta, distance = "bray")
##
##              Inertia Proportion Rank
## Total          14.0932      1.0000
## Conditional     0.2084      0.0148    1
## Constrained     1.2623      0.0896    3
## Unconstrained  13.0648      0.9270   46
## Imaginary      -0.4425     -0.0314   13
## Inertia is squared Bray distance
## Species scores projected from 'throat.otu.tab'
##
## Eigenvalues for constrained axes:
## CAP1 CAP2 CAP3
## 0.731 0.376 0.155
##
## Eigenvalues for unconstrained axes:
## MDS1 MDS2 MDS3 MDS4 MDS5 MDS6 MDS7 MDS8
## 2.278 1.994 1.144 0.925 0.749 0.615 0.534 0.473
## (Showing 8 of 46 unconstrained eigenvalues)
```

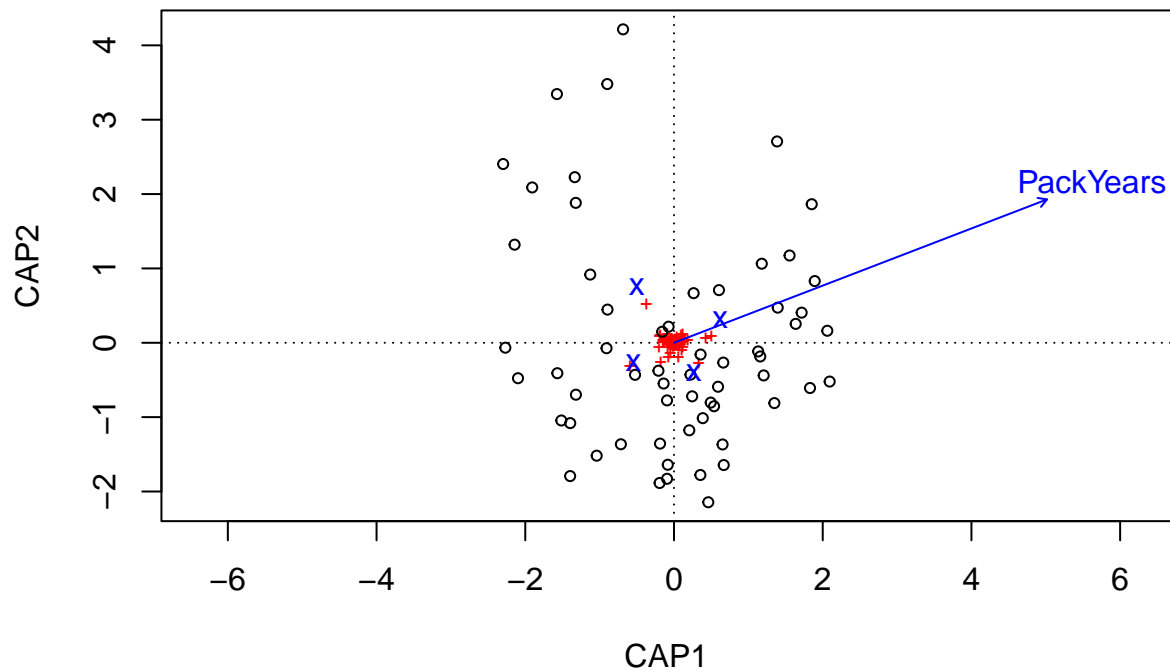
Los tres primeros ejes se denominan “CAP1”, “CAP2” y “CAP3”, y a continuación por el MDS original. Podemos ver que las tres variables restringidas (SmokingStatus, Sexo y PackYears) explican el 8,83 % de la variación total del conjunto de datos.

```
anova(throat_cap)
```

	Df	SumOfSqs	F	Pr(>F)
Model	3	1.262	1.771	0.003
Residual	55	13.065	NA	NA

El modelo es estadísticamente significativo con un valor p de 0.004. La función `plot()`, genera la siguiente figura, pero no es informativa

```
plot(throat_cap)
```



Como estamos interesados en las diferentes disimilitudes entre los fumadores y los no fumadores, extraigamos aquí la información del grupo.

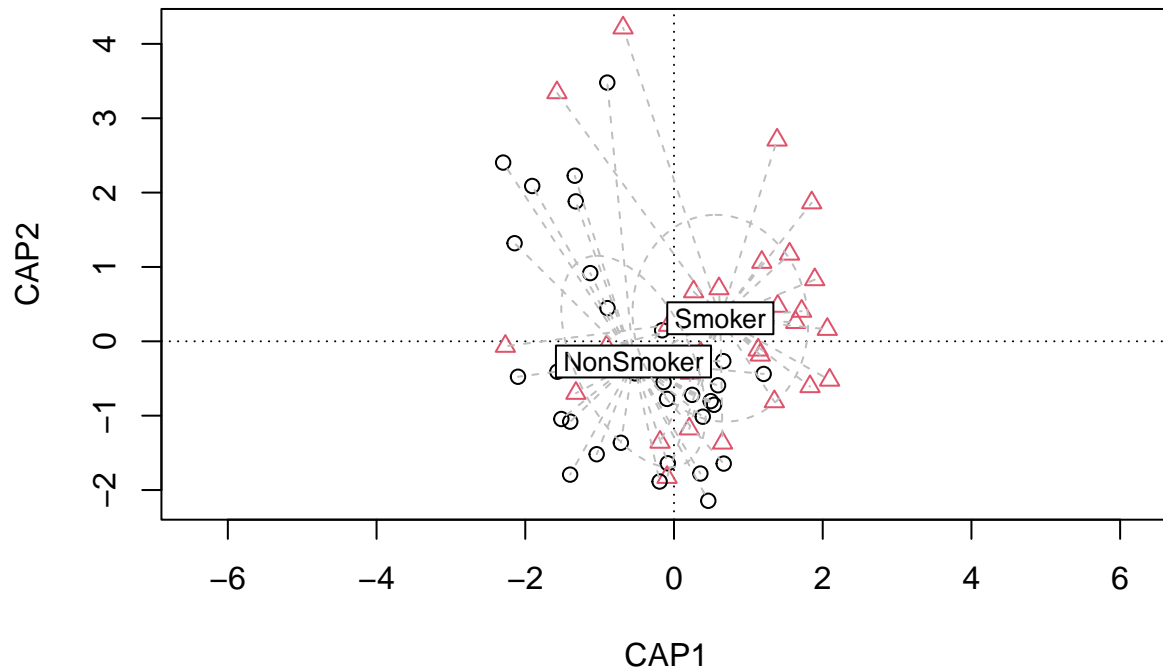
```
groups <- throat.meta$SmokingStatus
groups
```

```
## [1] NonSmoker Smoker Smoker Smoker Smoker Smoker NonSmoker
## [8] NonSmoker NonSmoker NonSmoker Smoker NonSmoker NonSmoker Smoker
## [15] NonSmoker Smoker Smoker NonSmoker NonSmoker NonSmoker NonSmoker
## [22] NonSmoker NonSmoker NonSmoker NonSmoker NonSmoker NonSmoker NonSmoker
## [29] NonSmoker NonSmoker NonSmoker NonSmoker NonSmoker Smoker NonSmoker
## [36] NonSmoker NonSmoker NonSmoker NonSmoker Smoker NonSmoker Smoker
## [43] NonSmoker Smoker Smoker Smoker Smoker Smoker Smoker
## [50] Smoker Smoker Smoker Smoker Smoker Smoker Smoker
## [57] Smoker NonSmoker Smoker Smoker
## Levels: NonSmoker Smoker
```

En el siguiente grupo de códigos R, la función `plot()` genera un diagrama de ordenación CAP vacío; la función `points()` añade puntos al diagrama de ordenación (función de trazado de bajo nivel) creado por la función `plot`. La función `ordispider()` crea spiderplot conectando los miembros individuales del grupo con el centroide del grupo. La función `ordiellipse()` rodea las nubes de puntos dentro del grupo mediante una elipse como las envolventes.

```
# Correr todo junto
plot(throat_cap, type="n")
points(throat_cap, col=as.numeric(as.factor(groups)),
```

```
pch=as.numeric(as.factor(groups)))
ordispider(throat_cap, groups, lty=2, col="grey", label=T)
ordiellipse(throat_cap, groups, lty=2, col="grey", label=F)
```



Al igual que en RDA y CCA, vamos a realizar una prueba de permutación para el modelo global global, cada eje y cada variable explicativa. Para asegurarse de obtener el mismo resultado de la prueba de permutación cada vez que se ejecute, establezca la misma semilla.

```
set.seed(123)
anova(throat_cap, step=1000) # El resultado es estadísticamente significativo
```

	Df	SumOfSqs	F	Pr(>F)
Model	3	1.262	1.771	0.007
Residual	55	13.065	NA	NA

```
# con un valor p de 0.003
# Entonces probamos significancia con cada eje
anova(throat_cap, by="axis", step=1000) # Hay significancia en el primer eje con
```

	Df	SumOfSqs	F	Pr(>F)
CAP1	1	0.7308	3.0763	0.009
CAP2	1	0.3764	1.5844	0.214
CAP3	1	0.1552	0.6535	0.867
Residual	55	13.0648	NA	NA

```
# un valor p de 0.001 y marginalmente significativo en el segundo eje con un
# valor p de 0.073.
# Ahora probamos con cada variable explicativa
anova(throat_cap, by="terms", step=1000)
```

	Df	SumOfSqs	F	Pr(>F)
SmokingStatus	1	0.6305	2.6544	0.004
Sex	1	0.4438	1.8683	0.045
PackYears	1	0.1880	0.7914	0.703
Residual	55	13.0648	NA	NA

Podemos ver que tanto SmokingStatus como Sex son significativos con valores $p = 0,003$ y $0,035$, respectivamente. Ahora, hagamos la selección hacia delante utilizando la función `ordistep()` del paquete **vegan**. Como dijimos en RDA y CCA, la función compara las variables basándose en el criterio AIC y los valores p de la prueba de permutación de Monte Carlo.

```
step_forward <- ordistep(capscale(throat.otu.tab ~ 1, data=throat_meta),
                        scope=formula(throat_cap), direction="forward", pstep=1000)
```

```
##
## Start: throat.otu.tab ~ 1
##
##              Df AIC      F Pr(>F)
## + SmokingStatus  1 712 2.05  0.015 *
## + Sex            1 712 1.68  0.095 .
## + PackYears      1 712 1.27  0.205
## + Condition(Age) 1 713 0.00
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Step: throat.otu.tab ~ SmokingStatus
##
##              Df AIC      F Pr(>F)
## + Sex            1 712 1.53  0.12
## + PackYears      1 713 0.58  0.74
## + Condition(Age) 1 713 0.00
```

```
# Seleccionamos la variable de Smoking status, el modelo final es ajustado como sigue
cap_final<- capscale(throat.otu.tab ~ SmokingStatus, throat_meta, dist="bray")
anova(cap_final, step=1000)
```

	Df	SumOfSqs	F	Pr(>F)
Model	1	0.6533	2.729	0.003
Residual	58	13.8823	NA	NA

El estatus de fumador alcanza la significación con un valor p de $0,004$, pero el modelo de parsimonia ha dado sus frutos con un residuo mayor.

5. Resumen y discusión

Utilizamos conjuntos de datos de ratones y humanos para ilustrar el análisis exploratorio de los datos del microbioma. En primer lugar, utilizamos el paquete **phyloseq** para ilustrar los cinco gráficos, incluyendo la

riqueza, la barra de abundancia, el mapa de calor, la red y el árbol filogenético. A continuación, introducimos varias familias de *métodos de agrupación* disponibles en los estudios de ecología y microbioma, y nos centramos en ilustrar cuatro métodos de agrupación (*aglomerativo de enlace único*, *aglomerativo de enlace completo*, *aglomerativo de tinta media* y *aglomerativo de varianza mínima de Ward*). También describimos brevemente la relación entre clustering, ordenación y medida de distancia. Por último, ilustramos las *ordenaciones* más comunes *sin restricciones* y *con restricciones*: **PCA**, **PCoA**, **NMDS**, **CA**, **RDA**, **CCA** y **CAP**. Las características de las *ordenaciones no restringidas* se consideran “*exploratorias*”. Sin embargo, las capacidades de las *ordenaciones restringidas* van más allá del mero análisis exploratorio de datos, y se convierten también en pruebas de hipótesis. Hemos introducido las diferentes características de las ordenaciones no restringidas y restringidas. Ilustramos el análisis exploratorio y la prueba de hipótesis (en términos de ordenaciones restringidas) de los datos del microbioma mediante el uso del paquete para analizar los datos del censo del microbioma **phyloseq** y los paquetes que comúnmente se utilizan en ecología, como el paquete **vegan**. Las funciones de extensión, es decir **cleanplot.pca()** y **evplot()** son atractivas y dos criterios para evaluar el rendimiento de las ordenaciones (el criterio de *Kaiser-Guttman* y el modelo *broken-stick*) son útiles para evaluar los análisis de ordenación.