La tabla de taxonomía relaciona el nombre de las taxa con el linaje taxonómico de éstas, i.e., vincula una variante de secuencia, o ASV, con los rangos taxonómicos desde Reino hasta Género o Especie dependiendo del nivel de resolución del análisis.

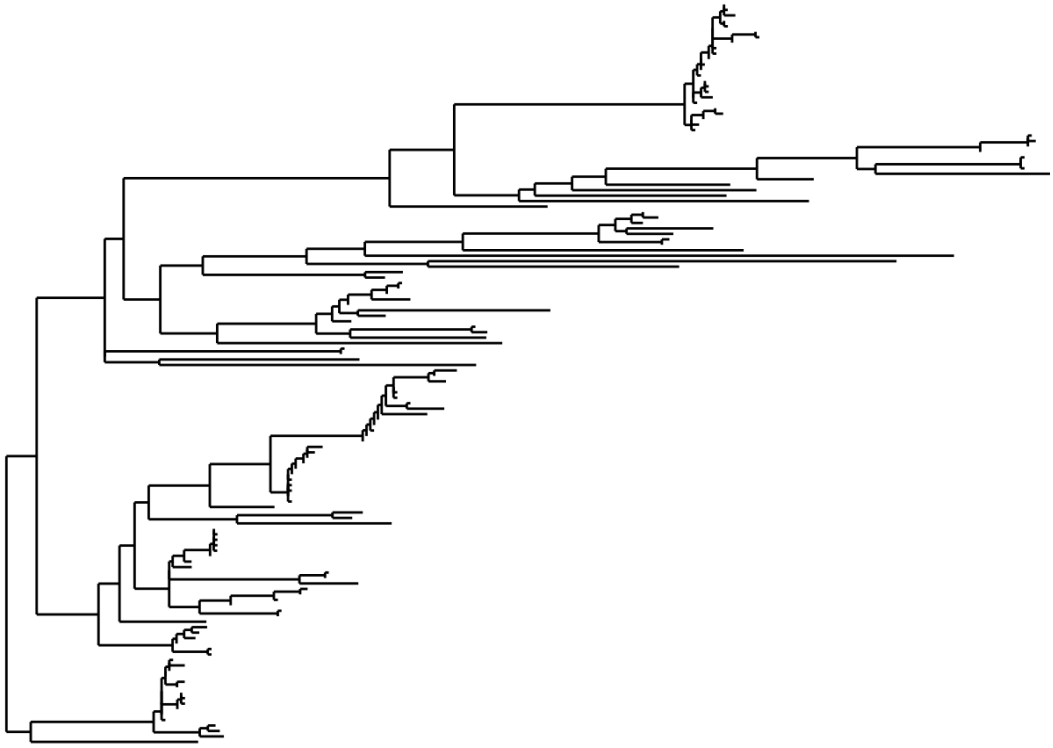
Veamos ahora el otro componente esencial que es la metadata.

SAMPLEid 123

Finalmente, tenemos el árbol filogenético, que es opcional en phyloseq, que nos muestra las relaciones evolutivas entre las taxa de todas las muestras. Es opcional porque normalmente cuando hacemos shotgun metagenomics no contamos con un marcador universal y por lo tanto no hay filogenia.

Podemos graficar simplemente la filogenia con la función plot_tree.

```
# Esta es la filogenia asociada a las taxa en nuestro objeto phyloseq

plot_tree(psd5, method = "treeonly", ladderize = "left")
```



Ahora, el objeto `phyloseq` se ha vuelto una suerte de estándar en la industria ya que otros paquetes ahora usan esta estructura de datos para sus propias funciones. Uno de esos paquetes es `microbiome` y `ampvis`. Podemos

fácilmente obtener un resumen global de nuestro objeto `phyloseq` usando la función `summarize_phyloseq`.

- Primero cargamos el paquete con `library(microbiome)`.

```
summarize_phyloseq(psd5)
## Compositional = NO
## 1] Min. number of reads = 1123
## 2] Max. number of reads = 103541
## 3] Total number of reads = 2606004
## 4] Average number of reads = 29954.0689655172
## 5] Median number of reads = 23576
## 7] Sparsity = 0.679428668018932
## 6] Any OTU sum to 1 or less? NO
## 8] Number of singletons = 0
## 9] Percent of OTUs that are singletons 0
## 10] Number of sample variables are: 13
## sample_ID
## bioproject_accession
## study
## biosample_accession
## experiment
## run
## SRA_Sample
## geo_loc_name
## collection_date
## sample_type
## species
## common_name
## AvgSpotLen
```

Este comando nos muestra el mínimo y máximo de reads, número total y promedio de reads, etc. También muestra los encabezados de las columans en la tabla de metadata.

- Veamos ahora una tabla que mezcle metadata, taxonomíaa y abundancia del taxon más abundante de cada muestra.

```
df <- psmelt(psd5)
```

out sampple

4144	ASV19	SRR6442763	37149	F3	PRJNA428495	SRP128093	SAMN08292252	
	SRX3533919	SRR6442763	SRS2809194	Chile: Chiloe	2015	skin		
	Balaenoptera musculus	blue whale	500	Bacteria	Proteobacteria			
	Gammaproteobacteria	Cardiobacteriales	Cardiobacteriaceae	NA				

También es importante tener una visión de cómo se distribuyen las muestras de acuerdo a la metadata. En este ejemplo, graficamos la frecuencia de muestras de acuerdo a la ubicación geográfica (geo_loc_name) y a la especie de ballena de donde la muestra fue obtenida (species).

res <- plot_frequencies(sample_data(psd5), "geo_loc_name", "species")

print(res$plot)

figbarras

tablaR

Ahora veamos cómo podemos filtrar y hacer *subsetting* de un objeto phyloseq. Esto lo hacemos con tres grupos de funciones, i.e., `filter`, `subset`, y `prune`. Filtrar se refiere a filtrar según alguna regla lógica. Ya lo hicimos en la parte de control de calidad cuando llamamos la función `filter_taxa(psd1, function(x) mean(x) > 1e-5, TRUE)`. Acá le pedíamos a la función `filter_taxa` que sobre el objeto `psd5`, calculara la media de *read counts* para cada taxa y si este resultado era menor que 1e-5, lo eliminara. Veamos un ejemplo diferente y filtremos según abundancia.

- Primero transformamos en abundancia relativa y luego filtramos.

```
# Transformamos las cuentas en porcentaje
psd5r  = transform_sample_counts(psd5, function(x) x / sum(x) )

# Filtramos las taxa con una abundancia inferior al 1%
(psd5r.filtrado = filter_taxa(psd5r, function(x) sum(x) > 1, TRUE))
## phyloseq-class experiment-level object
## otu_table()   OTU Table:         [ 24 taxa and 87 samples ]
## sample_data() Sample Data:       [ 87 samples by 13 sample variables ]
## tax_table()   Taxonomy Table:    [ 24 taxa by 6 taxonomic ranks ]
## phy_tree()    Phylogenetic Tree: [ 24 tips and 23 internal nodes ]
```

¿Cuántas taxa permanecen en nuestro objeto phyloseq? Con una operación tan simple como la que acabamos de aplicar, nos damos cuenta que la mayoría de las taxa presentes en nuestras muestras están en muy baja abundancia.

- Ahora imaginemos la situación donde queremos filtrar nuestro objeto pero en función de un taxon en específico.

	Genus
ASV1	NA
ASV2	NA
ASV3	NA
ASV4	NA
ASV5	Stenotrophomonas
ASV6	Moraxella
ASV7	Tenacibaculum
ASV8	Klebsiella
ASV9	Tenacibaculum
ASV10	NA
ASV11	NA
ASV12	Pseudomonas
ASV13	NA
ASV15	NA
ASV16	Moraxella
ASV17	Moraxella
ASV18	NA
ASV19	NA

	Genus
ASV20	NA
ASV22	Pseudomonas
ASV23	Tenacibaculum
ASV24	Achromobacter
ASV25	Catenococcus
ASV28	Escherichia/Shigell

```
# Ahora filtramos de acuerdo a _Moraxella_
(subset_taxa(psd5r.filtrado, Genus=="Moraxella") -> psd5r.filtrado.moraxella)
## phyloseq-class experiment-level object
## otu_table()   OTU Table:         [ 3 taxa and 87 samples ]
## sample_data() Sample Data:       [ 87 samples by 13 sample variables ]
## tax_table()   Taxonomy Table:    [ 3 taxa by 6 taxonomic ranks ]
## phy_tree()    Phylogenetic Tree: [ 3 tips and 2 internal nodes ]
# También podríamos todo lo que NO es _Moraxella_
(subset_taxa(psd5r.filtrado, Genus!="Moraxella") ->
psd5r.filtrado.NoMoraxella)
## phyloseq-class experiment-level object
## otu_table()   OTU Table:         [ 10 taxa and 87 samples ]
## sample_data() Sample Data:       [ 87 samples by 13 sample variables ]
## tax_table()   Taxonomy Table:    [ 10 taxa by 6 taxonomic ranks ]
## phy_tree()    Phylogenetic Tree: [ 10 tips and 9 internal nodes ]
```

- Otra manera de filtrar un objeto `phyloseq` es en base a algún atributo presente en `sample_data`. Por ejemplo, con estos datos uno podría querer estudiar el microbioma de las ballenas por separado. Para esto crearíamos tres objetos `phyloseq` a partir de `psd5`.

```
(psd5.blue = subset_samples(psd5, species == "Balaenoptera musculus"))
## phyloseq-class experiment-level object
## otu_table()   OTU Table:         [ 136 taxa and 26 samples ]
## sample_data() Sample Data:       [ 26 samples by 13 sample variables ]
## tax_table()   Taxonomy Table:    [ 136 taxa by 6 taxonomic ranks ]
## phy_tree()    Phylogenetic Tree: [ 136 tips and 135 internal nodes ]
(psd5.fin = subset_samples(psd5, species == "Balaenoptera physalus"))
## phyloseq-class experiment-level object
## otu_table()   OTU Table:         [ 136 taxa and 6 samples ]
## sample_data() Sample Data:       [ 6 samples by 13 sample variables ]
## tax_table()   Taxonomy Table:    [ 136 taxa by 6 taxonomic ranks ]
```

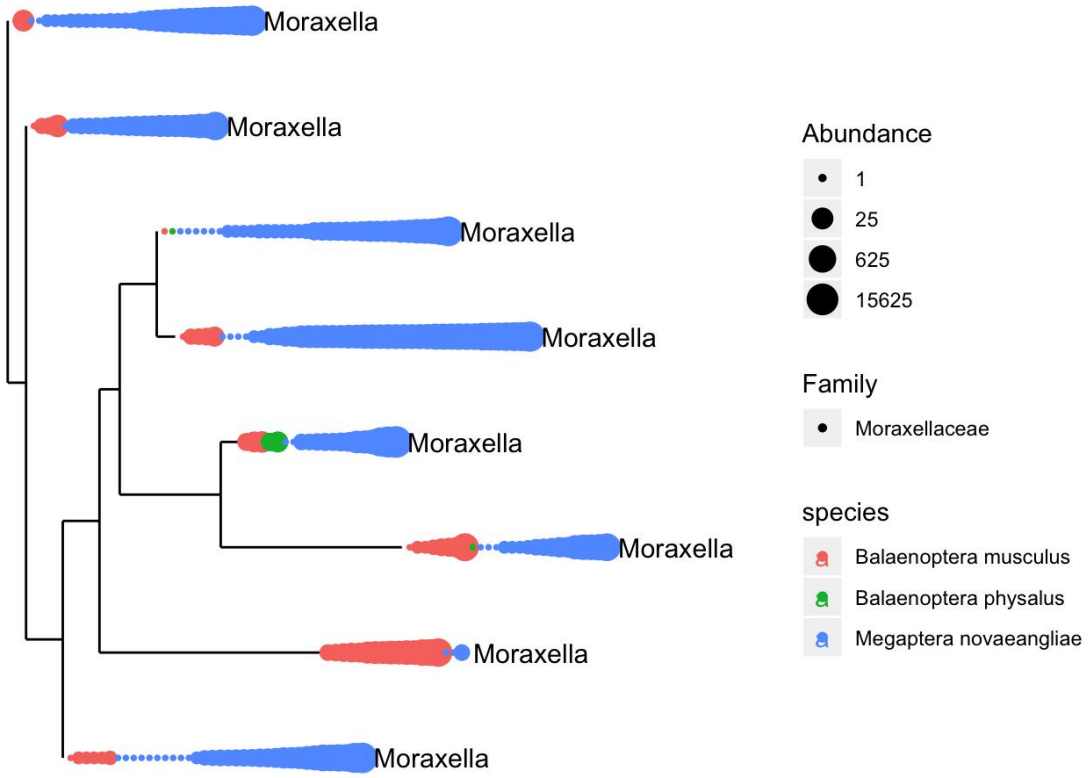
```
## phy_tree()    Phylogenetic Tree: [ 136 tips and 135 internal nodes ]
(psd5.joro = subset_samples(psd5, species == "Megaptera novaeangliae"))
## phyloseq-class experiment-level object
## otu_table()   OTU Table:         [ 136 taxa and 53 samples ]
## sample_data() Sample Data:       [ 53 samples by 13 sample variables ]
## tax_table()   Taxonomy Table:    [ 136 taxa by 6 taxonomic ranks ]
## phy_tree()    Phylogenetic Tree: [ 136 tips and 135 internal nodes ]
```

- Alternativamente, podríamos decicir estudiar solo tres de las cuatro especies de ballenas que tenemos representadas en `psd5`.

```
psd5 = subset_samples(subset_samples(psd5, species != "Eubalaena australis"))
```

- El comando `prune_samples()` también es muy usado ya que nos permite usar un vector con las muestras que queremos mantener (similar a `subset_samples`) o un vector lógico donde las muestras que queremos mantener son verdaderas.

```
# Primero seleccionamos solo el género _Moraxella_
subset_taxa(psd5, Genus=="Moraxella") -> psd5.moraxella

# Luego nos quedamos con las muestras que solo cumplen con la condición, i,e,
que poseen una abundancia de _Moraxella_ de más de 5 reads
prune_samples(sample_sums(psd5.moraxella)>=5, psd5.moraxella) ->
psd5.moraxella

# Y finalmente visualizamos los resultados mapeados en el árbol filogenético
plot_tree(psd5.moraxella, color="species", shape="Family", label.tips="Genus",
size="abundance")
```



Inmediatamente podemos apreciar que la distribución de *Moraxella* es mayor en ballena jorobada que en las otras dos especies, azul y fin.

- Otra situación muy común ocurre cuando queremos remover contaminantes u otras taxa no deseadas. Esto se puede hacer fácilmente con el comando `prune_taxa`.

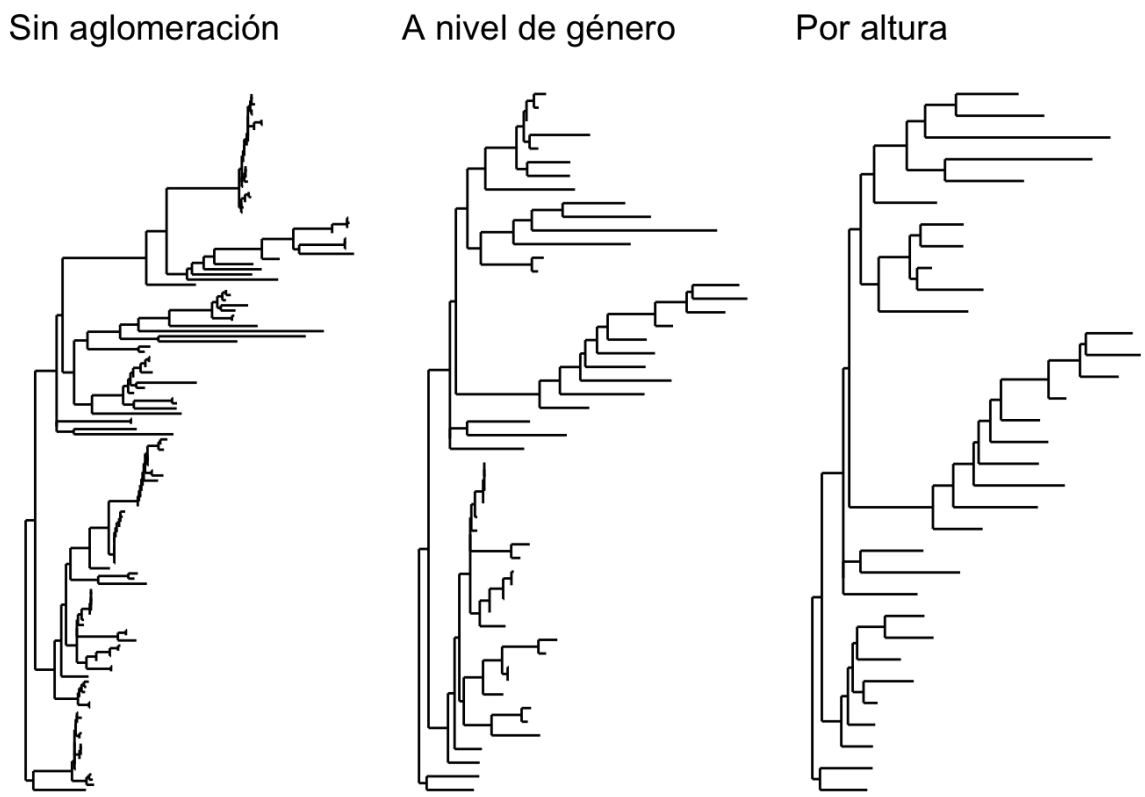
```
# Primero definimos las taxa que no queremos
badTaxa = c("ASV134", "ASV104", "ASV68")

# Creamos una lista con todos los nombres de los taxa presentes en el objeto
`psd5`
allTaxa = taxa_names(psd5)

# Nos quedamos con la diferencia entre badTaxa y allTaxa
keepTaxa <- allTaxa[!(allTaxa %in% badTaxa)]

#Ejecutamos `prune_taxa` sobre psd5
(psd5.prune = prune_taxa(keepTaxa, psd5))
## phyloseq-class experiment-level object
## otu_table()   OTU Table:         [ 133 taxa and 85 samples ]
## sample_data() Sample Data:       [ 85 samples by 13 sample variables ]
## tax_table()   Taxonomy Table:    [ 133 taxa by 6 taxonomic ranks ]
## phy_tree()    Phylogenetic Tree: [ 133 tips and 132 internal nodes ]
```

Para finalizar esta sección, un par de funciones muy útiles en `phyloseq` son `tax_glom()` y `tip_glom()`. Ambas funciones tratan de agrupar o

aglomerar un objeto de acuerdo a alguna propiedad, de esta manera simplificándolo. Por ejemplo, es muy probable que uno tenga varias ASVs del mismo género ya que si bien a nivel de secuencia son diferentes, estas corresponden al mismo género. En cierta forma ya lo vimos cuando seleccionamos el género *Moraxella*. El objeto resultante tenía ocho taxa, todas ellas *Moraxella*.

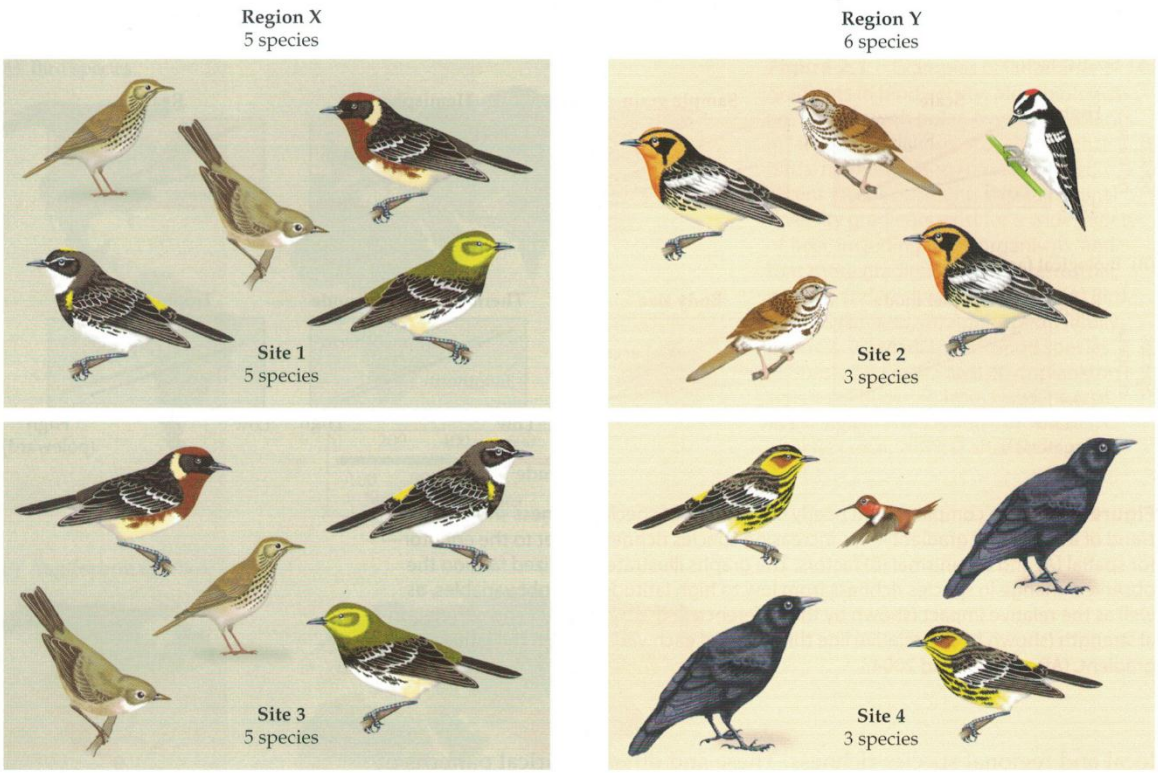
- Para hacer visualizaciones y otros análisis puede ser conveniente colapsar o aglomerar estas secuencias del mismo género u otro rango taxonómico. Al mismo tiempo, `tip_glom` realiza una función similar pero basándose en una "altura" arbitraria en el árbol filogenético.

```
# Primero aglomeramos por género
psd5.genus = tax_glom(psd5, "Genus", NArm = FALSE)
# Luego por altura en el árbol filogenético
h1 = 0.4
psd5.tip = tip_glom(psd5, h = h1)
# Grafiquemos una comparación para visualizar las diferencias
multiPlotTitleTextSize = 15
p2tree = plot_tree(psd5, method = "treeonly",
                   ladderize = "left",
                   title = "Sin aglomeración") +
  theme(plot.title = element_text(size = multiPlotTitleTextSize))
p3tree = plot_tree(psd5.genus, method = "treeonly",
                   ladderize = "left", title = "A nivel de género") +
  theme(plot.title = element_text(size = multiPlotTitleTextSize))
p4tree = plot_tree(psd5.tip, method = "treeonly",
                   ladderize = "left", title = "Por altura") +
  theme(plot.title = element_text(size = multiPlotTitleTextSize))

# Graficamos los árboles juntos
grid.arrange(nrow = 1, p2tree, p3tree, p4tree)
```



3 Introducción al análisis de diversidad

¿Qué entendemos por diversidad? Al menos podemos conceptualizar diversidad a dos niveles: diversidad genética o morfológica, y biodiversidad. En el estudio de comunidades, tomamos prestado el concepto de biodiversidad de ecología de comunidades donde estamos interesados en la riqueza de especies (número de especies diferentes en una comunidad o diversidad alfa), en las diferencias y similitudes entre comunidades (diversidad beta), y en algunos casos en la diversidad total de un región o paisaje ecológico (*landscape*; diversidad gama).



En la figura observamos la diversidad de aves en dos regiones, X e Y, y cuatro sitios, 1-4 (figura tomada de Community Ecology de Mittelbach). La diversidad alfa es mayor en los sitios 1 y 3 con 5 especies cada uno. La diversidad beta mide la cantidad de cambio o *turnover* de especies entre sitios. En la figura, la región Y tiene una diversidad beta mayor que la región X porque el cambio o *turnover* de especies entre el sitio 2 y el 4 es mayor que entre el sitio 1 y el 3 (que tienen las mismas 5 especies). La diversidad gama mide la diversidad total dentro de una región, por lo tanto en nuestro ejemplo la diversidad gama es mayor en la región Y porque contiene 6 especies en total versus la región X que tiene 5 especies.

3.1 Medidas de riqueza, uniformidad, dominancia, diversidad filogenética (diversidad alfa)

En el contexto metagenómico, medimos diversidad alfa usando una serie de medidas prestadas de ecología que nos permiten caracterizar una comunidad microbiana. `phyloseq` tiene una función muy útil que nos permite calcular y graficar hasta siete medidas, i .e., Observed (simplemente el número de taxa o riqueza), Chao1 (la riqueza ajustada por probabilidad de no observar especies), ACE (riqueza que toma en cuenta la abundancia relativa), Shannon (abundancia relativa de taxa), Simpson (1 - la probabilidad de que observemos

aleatoriamente dos bacterias en una comunidad y que pertenezcan a diferentes especies), Inverse Simpson (1 / Simpson), y Fisher (riqueza tomando en cuenta abundancia).

- En `phyloseq` simplemente llamamos la función `plot_richness` y podemos visualizar las medidas de diversidad.

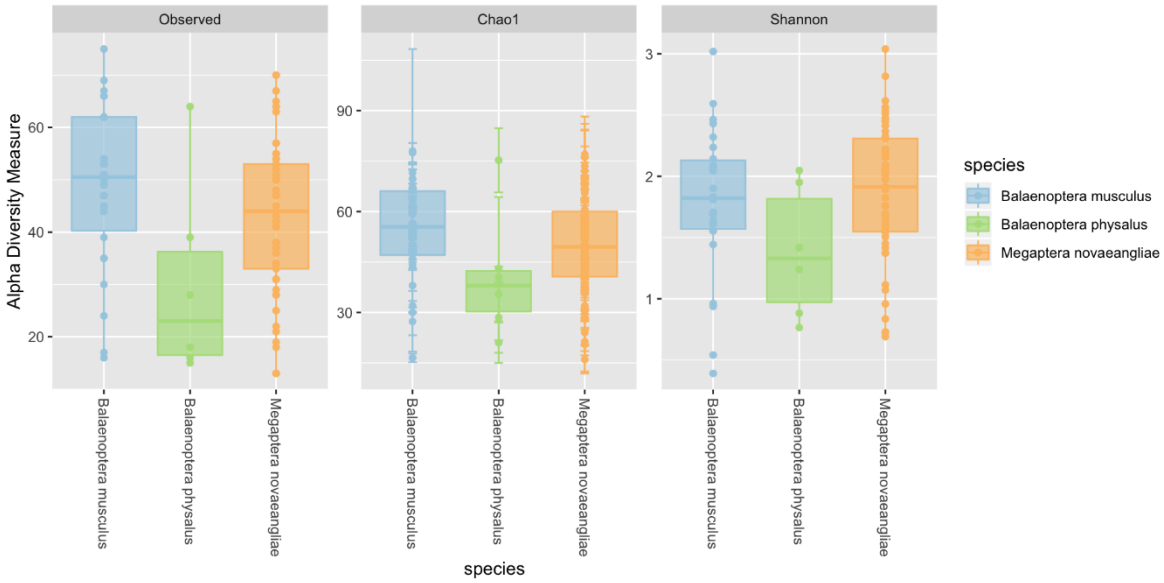
```
plot_richness(psd5, color = "species", x = "species", measures = c("Observed",
"Chao1", "Shannon")) +
```

3.1 Medidas de riqueza, uniformidad, dominancia, diversidad filogenética (diversidad alfa)

En el contexto metagenómico, medimos diversidad alfa usando una serie de medidas prestadas de ecología que nos permiten caracterizar una comunidad microbiana. `phyloseq` tiene una función muy útil que nos permite calcular y graficar hasta siete medidas, i .e., Observed (simplemente el número de taxa o riqueza), Chao1 (la riqueza ajustada por probabilidad de no observar especies), ACE (riqueza que toma en cuenta la abundancia relativa), Shannon (abundancia relativa de taxa), Simpson (1 - la probabilidad de que observemos aleatoriamente dos bacterias en una comunidad y que pertenezcan a diferentes especies), Inverse Simpson (1 / Simpson), y Fisher (riqueza tomando en cuenta abundancia).

- En `phyloseq` simplemente llamamos la función `plot_richness` y podemos visualizar las medidas de diversidad.

```
plot_richness(psd5, color = "species", x = "species", measures = c("Observed",
"Chao1", "Shannon"))
```



En el ejemplo, solo graficamos Observed, Chao1 y Shannon usando el argumento `measures = c("Observed", "Chao1", "Shannon")`. Si quisieramos obtener todas las medidas simplemente eliminamos este argumento y por defecto `phyloseq` graficará todo.

¿Hay un efecto significativo de la diversidad alfa según especie de ballena? Eso lo podríamos probar rápidamente con un análisis de varianza (ANOVA). Para este ejemplo utilicemos otra medida de diversidad, una que `phyloseq` no incorpora. **Faith's Phylogenetic Diversity** es un índice introducido por Daniel Faith en 1992 que no solo considera número de especies sino que también considera qué tanto se parecen estas especies filogenéticamente. Esto es muy relevante porque nos entrega una medida rápida para evaluar prioridades de conservación de ecosistemas, o si se trata de comunidades microbianas, donde tenemos mayor probabilidad de encontrar funciones génicas novedosas.

```
# Guardamos un dataframe con las medidas de diversidad alfa
alpha_pd <- estimate_pd(psd5)
# Combinamos la metadata con alpha.diversity
data <- cbind(sample_data(psd5), alpha_pd)
# Y calculamos un ANOVA
psd5.anova <- aov(PD ~ species, data)
# install.packages("xtable")
library(xtable)
psd5.anova.table <- xtable(psd5.anova)
```

	Df	Sum Sq	Mean Sq	F value
species	2	40.46002	20.23001	5.21047
Residuals	82	318.37078	3.88257	NA

- El paquete `microbiome` ofrece otras herramientas para evaluar diversidad que son accesibles fácilmente a través de su función `global`.

```
tab <- global(psd5, index = "all")
head(tab)
##            richness_0 richness_20 richness_50 richness_80 observed
## SRR6442697         31          31          31          16       31
## SRR6442698         38          38          38          17       38
## SRR6442699         25          25          25          12       25
## SRR6442700         53          53          53          36       53
## SRR6442701         48          48          48          32       48
## SRR6442702         42          42          42          22       42
##            diversities_inverse_simpson diversities_gini_simpson
## SRR6442697                    6.948971                0.8560938
## SRR6442698                    3.673899                0.7278096
## SRR6442699                    6.461545                0.8452383
## SRR6442700                    7.707637                0.8702586
## SRR6442701                    4.520999                0.7788099
## SRR6442702                    4.873575                0.7948118
##            diversities_shannon diversities_fisher diversities_coverage
## SRR6442697            2.181249           4.100949                    3
## SRR6442698            1.913672           5.781418                    2
## SRR6442699            2.043740           3.117388                    3
## SRR6442700            2.412447           6.104153                    3
## SRR6442701            2.038536           5.949163                    2
## SRR6442702            1.759212           4.346443                    3
##            evenness_camargo evenness_pielou evenness_simpson evenness_evar
## SRR6442697       0.04991266       0.6351942       0.05109538    0.09528851
## SRR6442698       0.04179604       0.5260830       0.02701396    0.13983565
## SRR6442699       0.04419760       0.6349235       0.04751136    0.07442449
## SRR6442700       0.06210963       0.6076245       0.05667380    0.08087544
## SRR6442701       0.04676204       0.5265900       0.03324264    0.10368220
## SRR6442702       0.03423354       0.4706709       0.03583511    0.07058635
##            evenness_bulla dominance_dbp dominance_dmn dominance_absolute
## SRR6442697     0.09213899     0.2669465     0.3980669                2099
## SRR6442698     0.11343557     0.4878935     0.6179177                2015
## SRR6442699     0.07294103     0.2597910     0.4137021                2461
## SRR6442700     0.14903074     0.2305235     0.4228024                8300
## SRR6442701     0.13473186     0.4078386     0.5421693                7742
## SRR6442702     0.06516802     0.2799936     0.4927869               19137
##            dominance_relative dominance_simpson dominance_core_abundance
## SRR6442697          0.2669465         0.1439062                0.2618593
## SRR6442698          0.4878935         0.2721904                0.6714286
## SRR6442699          0.2597910         0.1547617                0.4718674
## SRR6442700          0.2305235         0.1297414                0.6071379
## SRR6442701          0.4078386         0.2211901                0.2298899
## SRR6442702          0.2799936         0.2051882                0.4588576
##            dominance_gini rarity_log_modulo_skewness rarity_low_abundance
## SRR6442697      0.9500873                   2.059940          0.009538344
## SRR6442698      0.9582040                   2.060444          0.015012107
## SRR6442699      0.9558024                   2.038428          0.004011401
## SRR6442700      0.9378904                   2.051077          0.009359811
## SRR6442701      0.9532380                   2.054745          0.006900911
## SRR6442702      0.9657665                   2.053781          0.009861298
```

```
##            rarity_noncore_abundance rarity_rare_abundance
## SRR6442697              0.131756327           0.131756327
## SRR6442698              0.095883777           0.095883777
## SRR6442699              0.137337697           0.137337697
## SRR6442700              0.012887099           0.012887099
## SRR6442701              0.031607228           0.031607228
## SRR6442702              0.003131035           0.003131035
```

richness_0richness_20richness_50richness_80observeddiversities_inverse_simpsondiversities_gini_simpsondiversities_shannondiversities_fisherdiversities_coverageevenness_camargoevenness_pielouevenness_simpsonevenness_evarevenness_bulladominance_dbpdominance_dmndominance_absolutedominance_relativedominance_simpsondominance_core_abundancedominance_ginirarity_log_modulo_skewnessrarity_low_abundancerarity_noncore_abundancerarity_rare_abundanceSRR6442697313131163 16.950.862.184.1030.050.640.050.100.090.270.4020990.270.140.260.952.060.010.130.13SRR6442698383831738 3.670.731.915.7820.040.530.030.140.110.490.6220150.490.270.670.962.060.020.100.10SRR6442699252525122 56.460.852.043.1230.040.630.050.070.070.260.4124610.260.150.470.962.040.000.140.14SRR6442700535353365 37.710.872.416.1030.060.610.060.080.150.230.4283000.230.130.610.942.050.010.010.01SRR6442701484848324 84.520.782.045.9520.050.530.030.100.130.410.5477420.410.220.230.952.050.010.030.03SRR6442702424242224 24.870.791.764.3530.030.470.040.070.070.280.49191370.280.210.460.972.050.010.000.00

La función `global` nos da 26 medidas de diversidad que nos ayudan a entender la estructura de las comunidades microbianas. En general, estas medidas se dividen en riqueza, diversidad, dominancia, rareza, cobertura y uniformidad.

- El paquete `microbiome` ofrece funciones para calcular cada uno de estos aspectos de las comunidades microbianas.

```
# Riqueza
tab <- richness(psd5)
# Dominancia
tab <- dominance(psd5, index = "all")
# Rareza
tab <- rarity(psd5, index = "all")
# Cobertura
tab <- coverage(psd5, threshold = 0.5)
# Desigualdad
tab <- inequality(psd5)
# Uniformidad
tab <- evenness(psd5, "all")
```

- Veamos un ejemplo concreto estimando diversidad, graficando los resultados y calculando significancia estadística. Para esto usamos el paquete `ggpubr` que genera “publication-ready plots”, algo que siempre es deseable (ejecuta `library(ggpubr)`).

```r
# Generamos un objeto `phyloseq` sin taxa que sume 0 reads
psd5.2 <- prune_taxa(taxa_sums(psd5) > 0, psd5)
# Calculamos los índices de diversidad
tab <- diversities(psd5.2, index = "all")
# Y finalmente visualizamos la tabla de resultados
head(tab)
```

```
##             inverse_simpson gini_simpson  shannon   fisher coverage
## SRR6442697        6.948971    0.8560938 2.181249 4.100949        3
## SRR6442698        3.673899    0.7278096 1.913672 5.781418        2
## SRR6442699        6.461545    0.8452383 2.043740 3.117388        3
## SRR6442700        7.707637    0.8702586 2.412447 6.104153        3
## SRR6442701        4.520999    0.7788099 2.038536 5.949163        2
## SRR6442702        4.873575    0.7948118 1.759212 4.346443        3
```

	inverse_simpson	gini_simpson	shannon	fisher	coverage
SRR6442697	6.95	0.86	2.18	4.10	3
SRR6442698	3.67	0.73	1.91	5.78	2
SRR6442699	6.46	0.85	2.04	3.12	3
SRR6442700	7.71	0.87	2.41	6.10	3
SRR6442701	4.52	0.78	2.04	5.95	2
SRR6442702	4.87	0.79	1.76	4.35	3

- Ahora necesitamos extraer la metadata de nuestro objeto `phyloseq`.

```r
psd5.2.meta <- meta(psd5.2)
head(psd5.2.meta)
```

```
##             sample_ID bioproject_accession     study biosample_accession
## SRR6442697      EMA4          PRJNA428495 SRP128093        SAMN08292292
## SRR6442698      EMA3          PRJNA428495 SRP128093        SAMN08292291
## SRR6442699      EMA2          PRJNA428495 SRP128093        SAMN08292284
## SRR6442700     EMA19          PRJNA428495 SRP128093        SAMN08292283
## SRR6442701     EMA21          PRJNA428495 SRP128093        SAMN08292286
## SRR6442702     EMA20          PRJNA428495 SRP128093        SAMN08292285
##             experiment        run SRA_Sample               geo_loc_name
## SRR6442697 SRX3533985 SRR6442697 SRS2809259 Chile: Estrecho_Magallanes
## SRR6442698 SRX3533984 SRR6442698 SRS2809258 Chile: Estrecho_Magallanes
## SRR6442699 SRX3533983 SRR6442699 SRS2809257 Chile: Estrecho_Magallanes
## SRR6442700 SRX3533982 SRR6442700 SRS2809256 Chile: Estrecho_Magallanes
## SRR6442701 SRX3533981 SRR6442701 SRS2809255 Chile: Estrecho_Magallanes
## SRR6442702 SRX3533980 SRR6442702 SRS2809254 Chile: Estrecho_Magallanes
##             collection_date sample_type                species
## SRR6442697            2017        skin Megaptera novaeangliae
## SRR6442698            2017        skin Megaptera novaeangliae
## SRR6442699            2017        skin Megaptera novaeangliae
## SRR6442700            2017        skin Megaptera novaeangliae
## SRR6442701            2017        skin Megaptera novaeangliae
## SRR6442702            2017        skin Megaptera novaeangliae
```

```
##                common_name AvgSpotLen
## SRR6442697 humpback whale        501
## SRR6442698 humpback whale        500
## SRR6442699 humpback whale        501
## SRR6442700 humpback whale        500
## SRR6442701 humpback whale        499
## SRR6442702 humpback whale        500
```

	sample_ID	bioproject_accession	study	biosample_accession
SRR6442697	EMA4	PRJNA428495	SRP128093	SAMN08292292
SRR6442698	EMA3	PRJNA428495	SRP128093	SAMN08292291
SRR6442699	EMA2	PRJNA428495	SRP128093	SAMN08292284
SRR6442700	EMA19	PRJNA428495	SRP128093	SAMN08292283
SRR6442701	EMA21	PRJNA428495	SRP128093	SAMN08292286
SRR6442702	EMA20	PRJNA428495	SRP128093	SAMN08292285

experiment	run	SRA_Sample	geo_loc_name	collection_date
SRX3533985	SRR6442697	SRS2809259	Chile: Estrecho_Magallanes	2017
SRX3533984	SRR6442698	SRS2809258	Chile: Estrecho_Magallanes	2017
SRX3533983	SRR6442699	SRS2809257	Chile: Estrecho_Magallanes	2017
SRX3533982	SRR6442700	SRS2809256	Chile: Estrecho_Magallanes	2017
SRX3533981	SRR6442701	SRS2809255	Chile: Estrecho_Magallanes	2017
SRX3533980	SRR6442702	SRS2809254	Chile: Estrecho_Magallanes	2017

collection_date	sample_type	species	common_name	AvgSpotLen
2017	skin	Megaptera novaeangliae	humpback whale	501
2017	skin	Megaptera novaeangliae	humpback whale	500
2017	skin	Megaptera novaeangliae	humpback whale	501
2017	skin	Megaptera novaeangliae	humpback whale	500
2017	skin	Megaptera novaeangliae	humpback whale	499
2017	skin	Megaptera novaeangliae	humpback whale	500

- Luego agregamos la tabla de diversidad a la metadata.

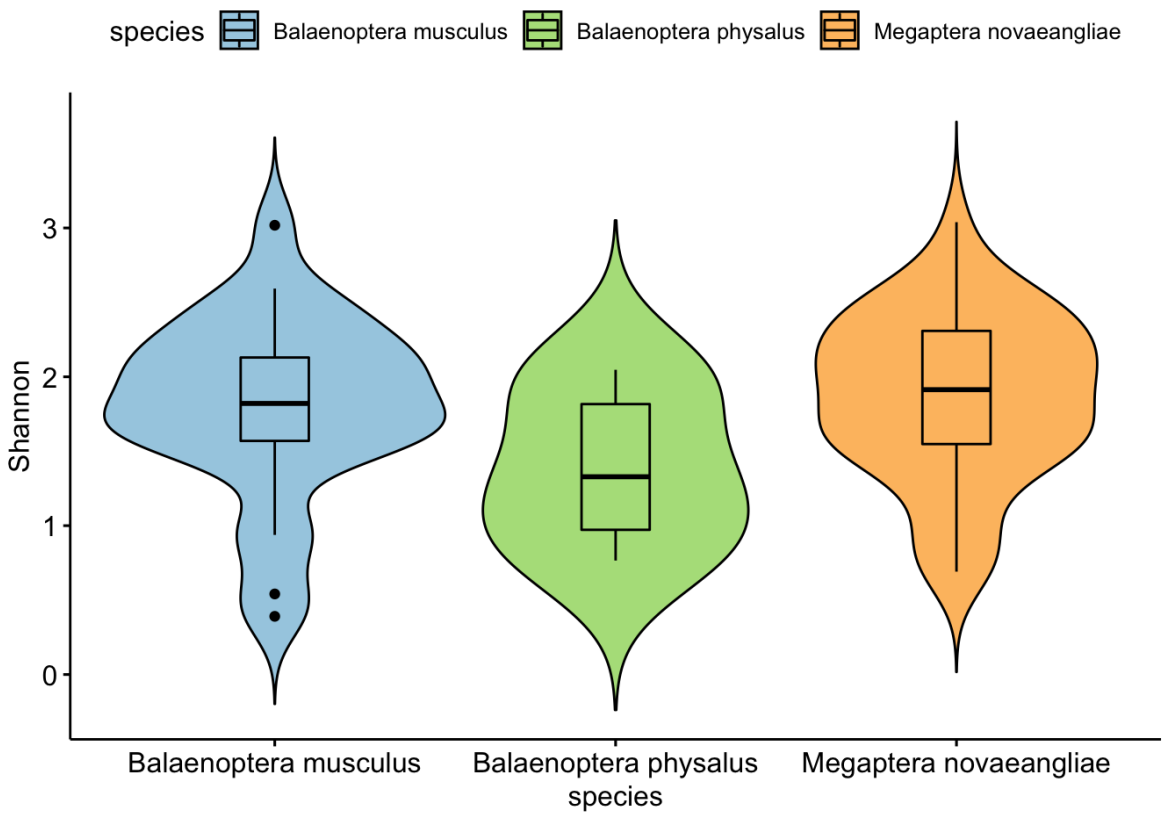
```
psd5.2.meta$Shannon <- tab$shannon
psd5.2.meta$InverseSimpson <- tab$inverse_simpson
```

- En este ejercicio nos interesa comparar la diversidad entre especies de ballenas. Recordemos que tenemos datos para tres especies de ballenas: azul, fin y jorobada. Necesitamos crear una lista de comparasiones de a pares para poder visualizar y calcular significancia estadística de manera simultánea.

```
# Obtenemos las variables desde nuestro objeto `phyloseq`
spps <- levels(psd5.2.meta$species)
# Creamos una lista de lo que queremos comparar
pares.spps <- combn(seq_along(spps), 2, simplify = FALSE, FUN =
function(i)spps[i])
# Imprimimos en pantalla el resultado
print(pares.spps)
## [[1]]
## [1] "Balaenoptera musculus" "Balaenoptera physalus"
##
## [[2]]
## [1] "Balaenoptera musculus"  "Megaptera novaeangliae"
##
## [[3]]
## [1] "Balaenoptera physalus"  "Megaptera novaeangliae"
```

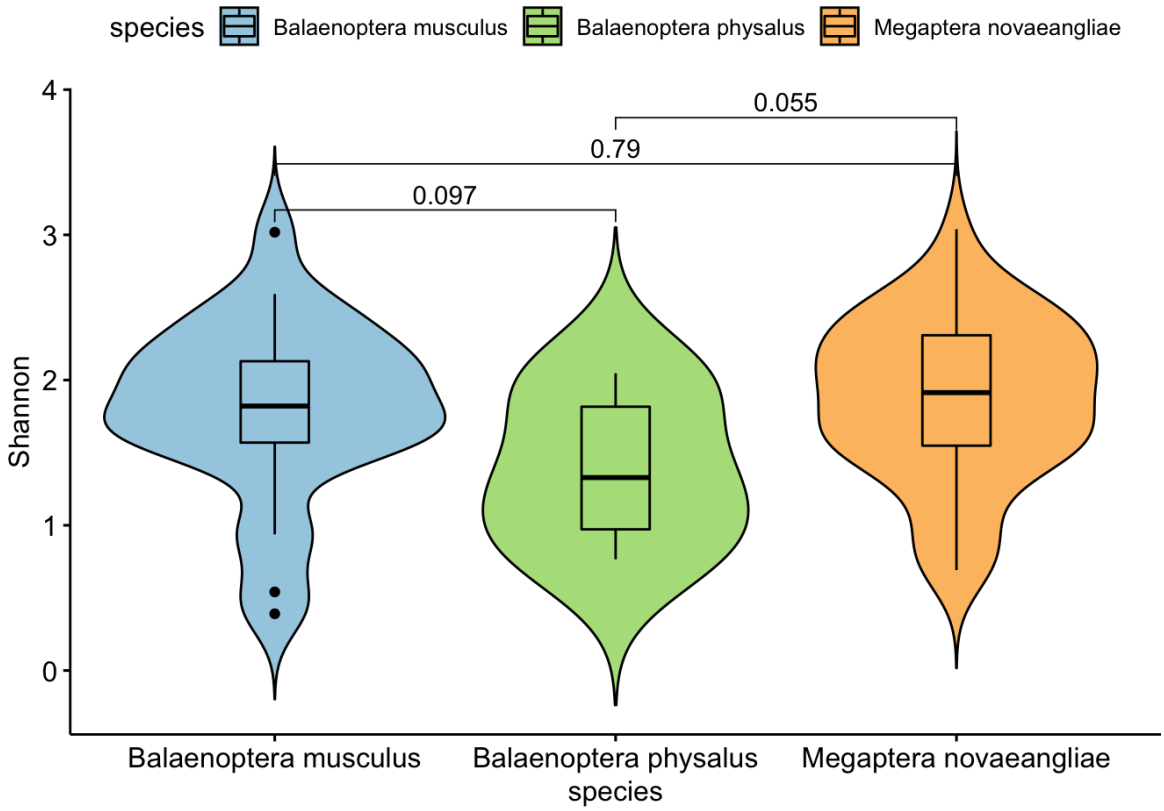
- Con la función `ggviolin` podemos generar un gráfico de violín en un solo paso de la siguiente forma.

```
p1 <- ggviolin(psd5.2.meta, x = "species", y = "Shannon",
 add = "boxplot", fill = "species", palette = c("#a6cee3", "#b2df8a",
"#fdbf6f"))
print(p1)
```



- Ahora necesitamos evaluar la significancia estadística entre los estimados de diversidad de las muestras de ballenas. De nuevo, en una línea, tenemos nuestra figura lista para el artículo.

```
p1 <- p1 + stat_compare_means(comparisons = pares.spps)
print(p1)
```



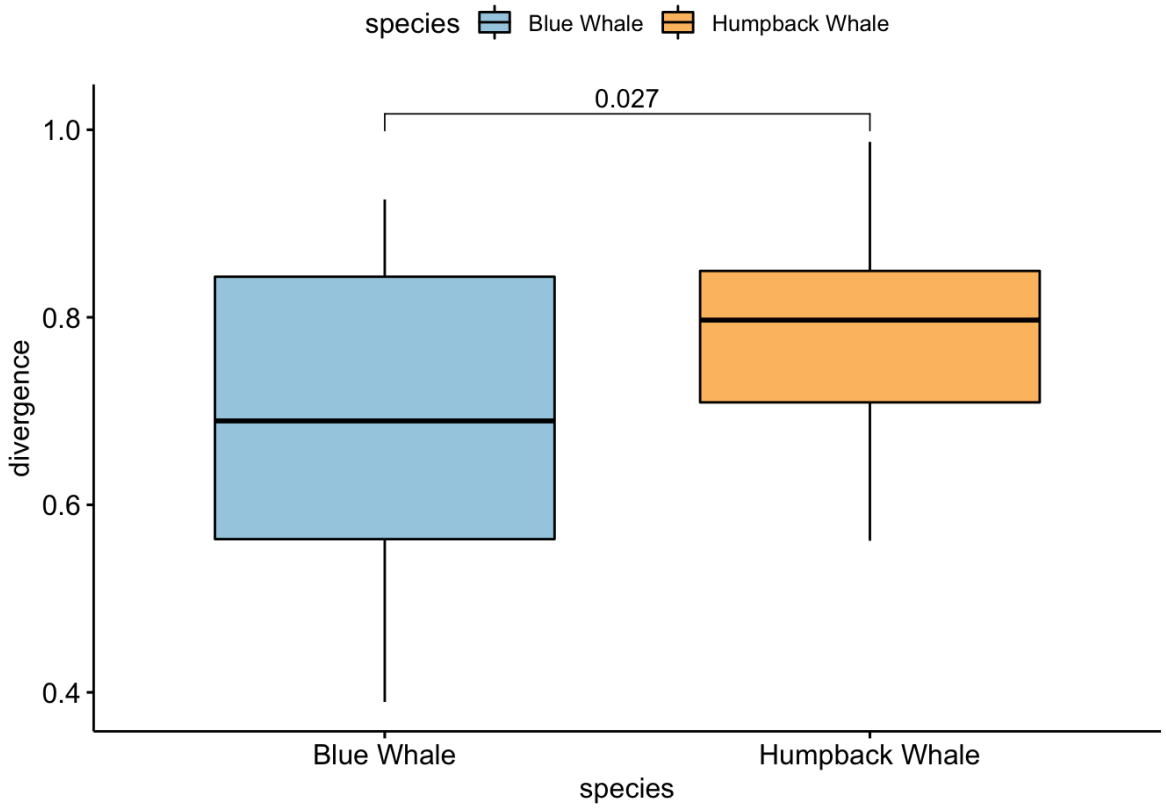
3.2 Diversidad beta y escalamiento multidimensional (Bray-Curtis, UniFrac, t-SNE)

En cuanto a diversidad beta podemos calcular similitud global a través de todas las muestras de interés o también podemos cuantificar la divergencia de un grupo y compararla con la divergencia de otro.

- Veamos este último caso primero.

```
# Calculamos las divergencias para ballena azul y jorobada
div.azul <- divergence(subset_samples(psd5, species == "Balaenoptera
musculus"))
div.joro <- divergence(subset_samples(psd5, species == "Megaptera
novaeangliae"))
# transformamos el resultado anterior en _dataframes_
data.frame(div.azul) -> df.div.azul
data.frame(div.joro) -> df.div.joro
# Agregamos columnas a nuestros _dataframes_
```

```r
mutate(df.div.azul, species = "Blue Whale") -> df.div.azul
mutate(df.div.joro, species = "Humpback Whale") -> df.div.joro
# Cambiamos los nombres de las columans de manera que sean iguales an ambos
_dataframes_
colnames(df.div.azul) <- c("divergence", "species")
colnames(df.div.joro) <- c("divergence", "species")
# Los combinamos en un _dataframe_
rbind(df.div.azul, df.div.joro) -> div.boxplot

# Y finalmente graficamos y realizamos una comparación estadística
p2 <- ggboxplot(data = div.boxplot, x = "species", y = "divergence", fill =
"species", palette = c("#a6cee3", "#fdbf6f"))
p2 + stat_compare_means(comparisons = list(c("Blue Whale", "Humpback Whale")))
```

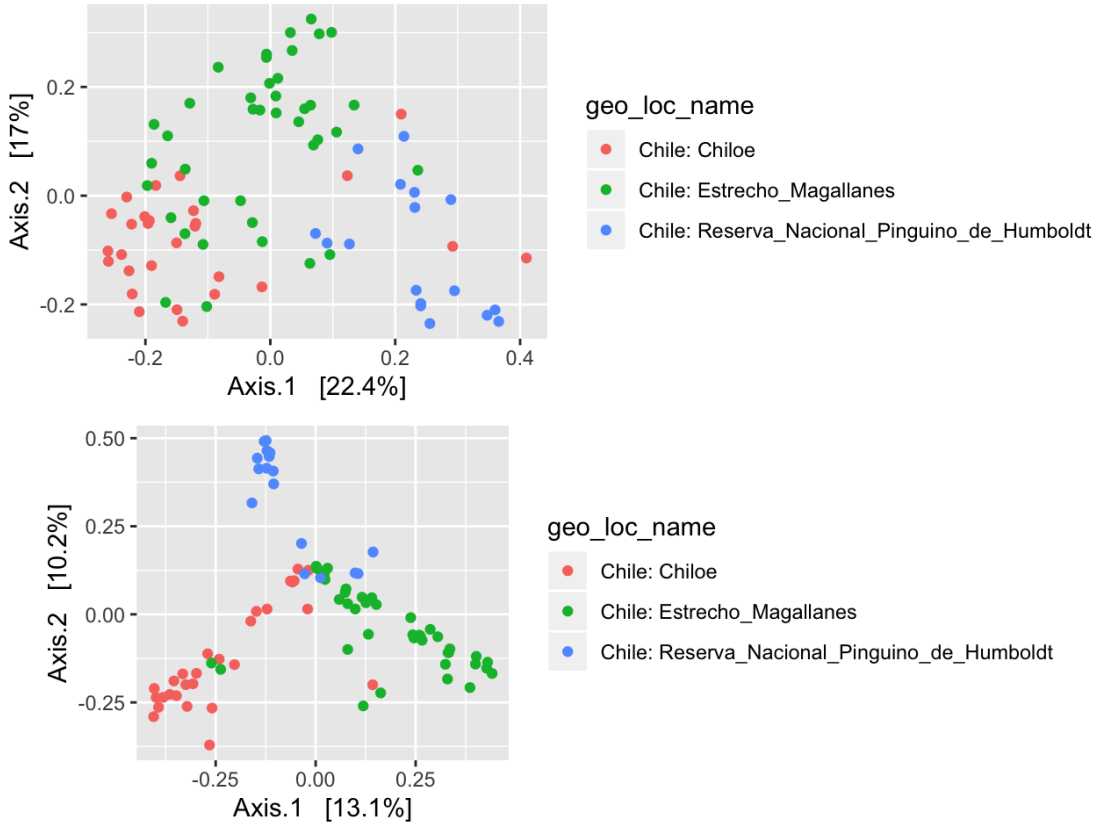


Existen diferentes medidas de similitud (o disimilitud, i.e., 1 - similitud) disponibles que nos permiten entender las relaciones entre nuestras muestras. En general todas producen matrices de distancia comparables. El paquete `phyloseq` ofrece un gran número de medidas de distancia. Las más populares son UniFrac y Weighted UniFrac (medidas que consideran filogenia) y otras independientes de filogenia como: Jaccard, Manhattan, Euclidian, Bray-Curtis, Canberra, etc. Por otra parte, la matriz de distancia resultante no se usa en aislación sino que en conjunto con algún método de ordinación o escalamiento multidimensional (*ordination*). De nuevo, `phyloseq` ofrece un gran

número de métodos entre los cuales se encuentran: detrended y canonical correspondence analysis, Double Principal Coordinate Analysis, Non-metric MultiDimenstional Scaling, y MDS/PCoA.

- Probemos entonces hacer un análisis tipo PCoA con una matriz de distancia que considera las relaciones filogenéticas y otra que no.

```
psd5.mds.unifrac <- ordinate(psd5, method = "MDS", distance = "unifrac")
evals <- psd5.mds.unifrac$values$Eigenvalues
pord1 <- plot_ordination(psd5, psd5.mds.unifrac, color = "geo_loc_name") +
  labs(col = "geo_loc_name") +
  coord_fixed(sqrt(evals[2] / evals[1]))

psd5.mds.bray <- ordinate(psd5, method = "MDS", distance = "bray")
evals <- psd5.mds.bray$values$Eigenvalues
pord2 <- plot_ordination(psd5, psd5.mds.bray, color = "geo_loc_name") +
  labs(col = "geo_loc_name") +
  coord_fixed(sqrt(evals[2] / evals[1]))

grid.arrange(pord1, pord2)
```



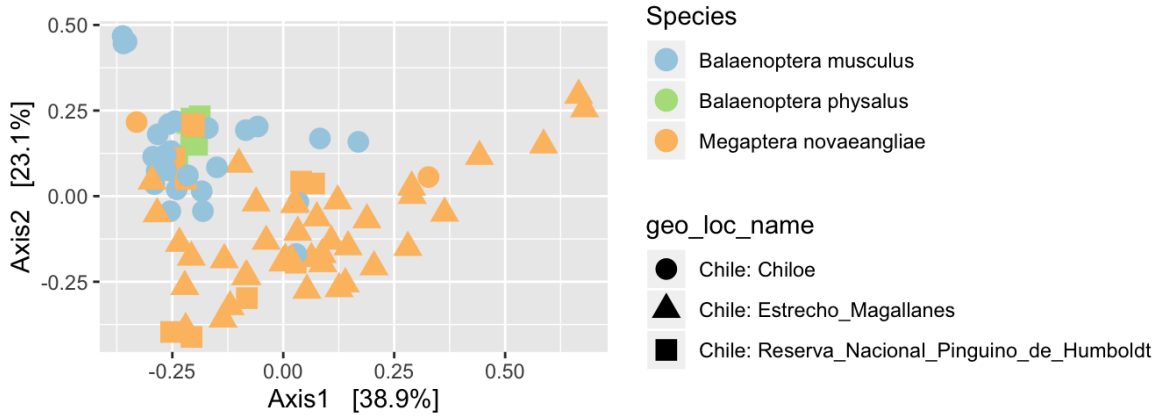
Nota que los gráficos de dispersión donde se visualiza este tipo de análisis están escalados según la cantidad de variación que los ejes explican. En

general lo que estos métodos pretenden hacer es tratar de encontrar el menor número de vectores matemáticos que maximicen la separación entre las muestras (puntos en el gráfico). Esto nace de la imposibilidad de graficar eficientemente datos multidimencionales. Los datos que estamos analizando ciertamente son multidimensionales en el sentido que tenemos más de 100 taxa que varían simultáneamente en cada una de las 90+ muestras que tenemos. Volviendo a los ejes, estos no suman 100% porque hay otros ejes que no estamos usando para graficar y que contribuyen con el resto de la variación. Al graficarlos de manera simétrica distorsionaríamos las relaciones entre los puntos, especialmente si estamos comparando dos o más gráficos.

En específico para comunidades microbianas, el método de doble análisis de coordenadas principales o (DPCoA) es muy apropiado porque analiza conjuntamente dos tipos de datos: una tabla de disimilitud que representa diferencias entre especies y una matriz de abundancia que representa la distribución de especies entre las comunidades. El resultado final es un ensamble del espacio multidimensional que correlacionan las especies con las comunidades. El método fue publicado en 2004.

- Veamos un ejemplo con nuestros datos.

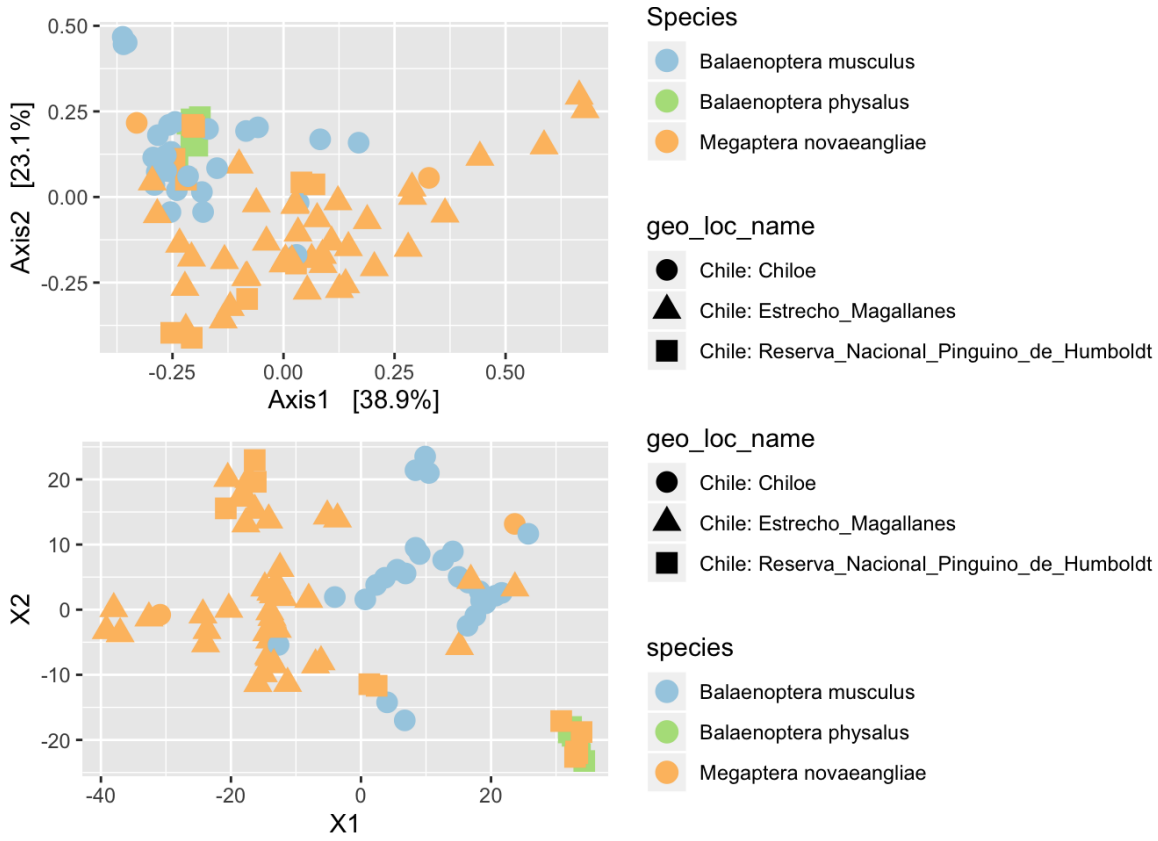
```
psd5.dpcoa.unifrac <- ordinate(psd5, method = "DPCoA", distance = "dpcoa")
evals <- psd5.dpcoa.unifrac$eig
pord3 <- plot_ordination(psd5, psd5.dpcoa.unifrac, color = "species", shape =
"geo_loc_name") +
  labs(col = "Species") +
  coord_fixed(sqrt(evals[2] / evals[1])) +
  scale_color_manual(values = c("#a6cee3", "#b2df8a", "#fdbf6f")) +
  scale_fill_manual(values = c("#a6cee3", "#b2df8a", "#fdbf6f")) +
  geom_point(size=4)
pord3
```



- Ahora exploremos escalamiento multidimensional usando un método reciente conocido como t-SNE o t-Distributed Stochastic Neighbor Embedding. *t-SNE* difiere de otros métodos en que hace énfasis en las distancias locales en

vez de las distancias globales, de esta forma generando una mayor resolución o separación entre los puntos o muestras.

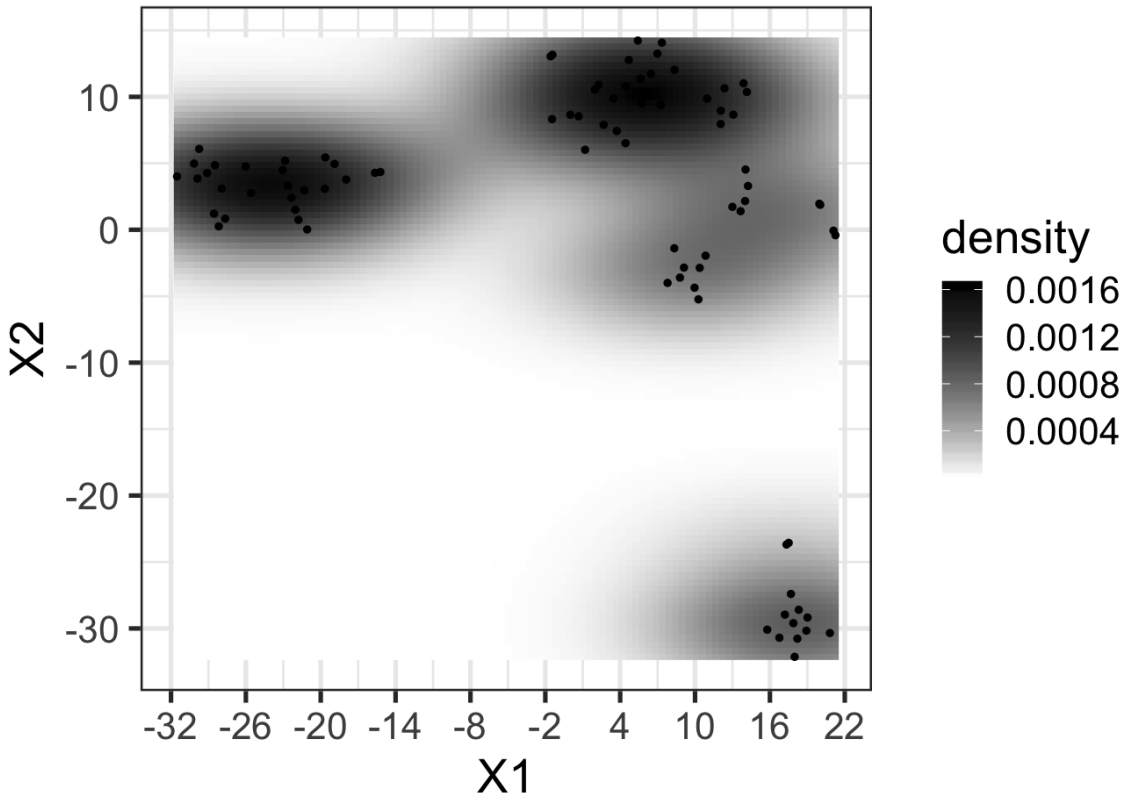
```
library(tsnemicrobiota)

tsne_res <- tsne_phyloseq(psd5, distance= "dpcoa", perplexity = 8, verbose=0,
rng_seed = 3901)

# Graficamos
pord4 <- plot_tsne_phyloseq(psd5, tsne_res, color = "species", shape =
"geo_loc_name") +
  geom_point(size=4) +
  scale_color_manual(values = c("#a6cee3", "#b2df8a", "#fdbf6f")) +
  scale_fill_manual(values = c("#a6cee3", "#b2df8a", "#fdbf6f"))

grid.arrange(pord3, pord4)
```



Ambos gráficos usan distintos métodos pero la misma medida de distancia. Los resultados son similares aunque las agrupaciones de puntos o muestras ocupan distinto espacio en el gráfico.

- Otro uso de estas medidas es a través de la visualización de la densidad de las muestras en el espacio.

```r
method <- "tsne"
trans <- "hellinger"
distance <- "euclidean"

# Matriz de distancia
psd5 <- microbiome::transform(psd5, trans)

# Calculamos similitud entre muestras
dm <- vegdist(otu_table(psd5), distance)

# Corremos t-SNE
tsne_out <- Rtsne(dm, dims = 2, perplexity = 8)
proj <- tsne_out$Y
rownames(proj) <- rownames(otu_table(psd5))
data.frame(proj) -> proj
proj$species <- sample_data(psd5)[,11]

pland <- plot_landscape(proj[,1:2], legend = T, size = 4)
print(pland)
```



3.3 Análisis de abundancias y visualizaciones

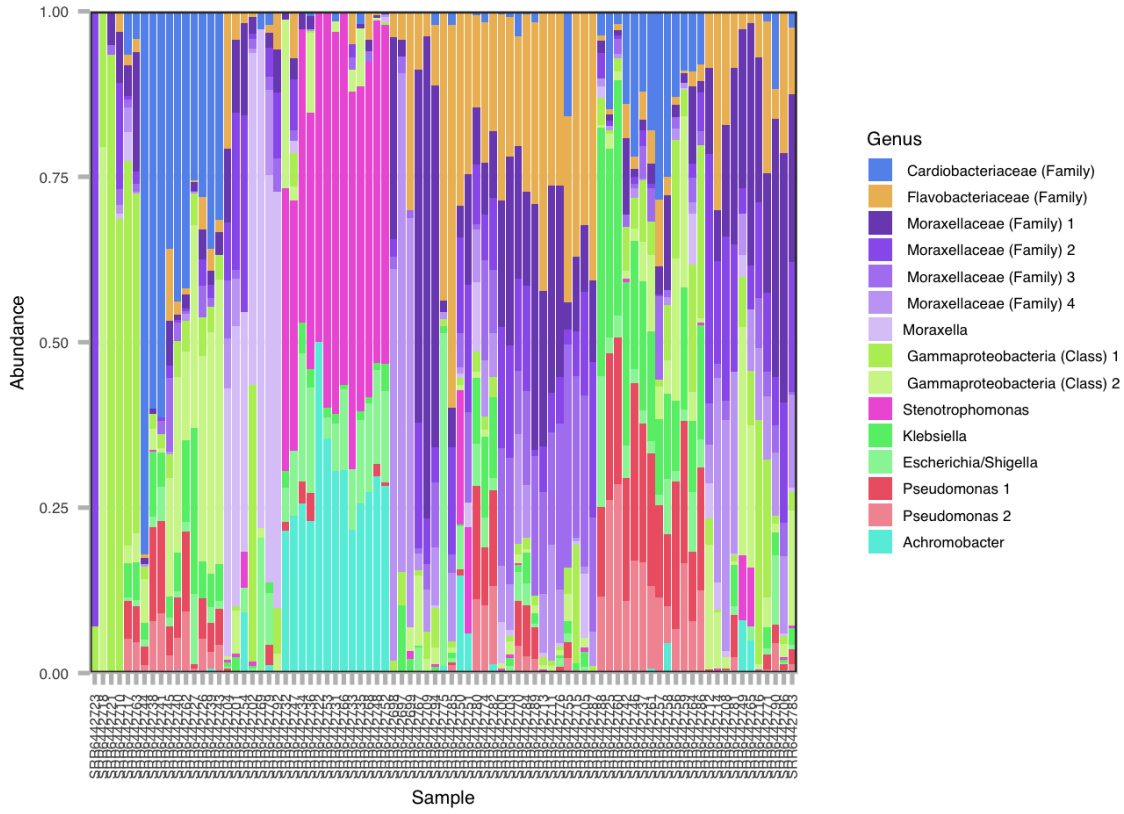
3.3.1 Gráfico de barras apiladas

Una manera muy eficiente de obtener una visión general de la composición de las muestras es a través de un gráfico de barras apiladas. Existe una variadad de funciones que pueden hacer esto, sin embargo vamos a usar el paquete creado por un ex-miembro del laboratorio ya que tiene la ventaja de poder agrupar por *hierarchical clustering* las muestras entre otras ventajas.

```
# Necesitamos obtener las taxa más abundantes, en este caso el top 15
top15 <- get_top_taxa(physeq_obj = psd5, n = 15, relative = T,
                      discard_other = T, other_label = "Other")
# Ya que no todas las taxa fueron clasificadas a nivel de especie, generamos
etiquetas compuestas de distintos rangos taxonómicos para el gráfico
top15 <- name_taxa(top15, label = "", species = F, other_label = "Other")
# Finalmente graficamos
fantaxtic_bar(top15, color_by = "Family", label_by = "Genus", facet_by = NULL,
grid_by = NULL, other_color = "Grey") -> ptop15
```

```
##                          Level N.color.shades Central.color
## 1           Cardiobacteriaceae               1       #6495ED
## 2            Flavobacteriaceae               1       #EDBC64
## 3                Moraxellaceae               5       #9A64ED
## 4  Gammaproteobacteria (Class)               2       #B7ED64
## 5            Xanthomonadaceae               1       #ED64DA
## 6           Enterobacteriaceae               2       #64ED77
## 7            Pseudomonadaceae               2       #ED6473
## 8            Burkholderiaceae               1       #64EDDE
```

```
ptop15
```



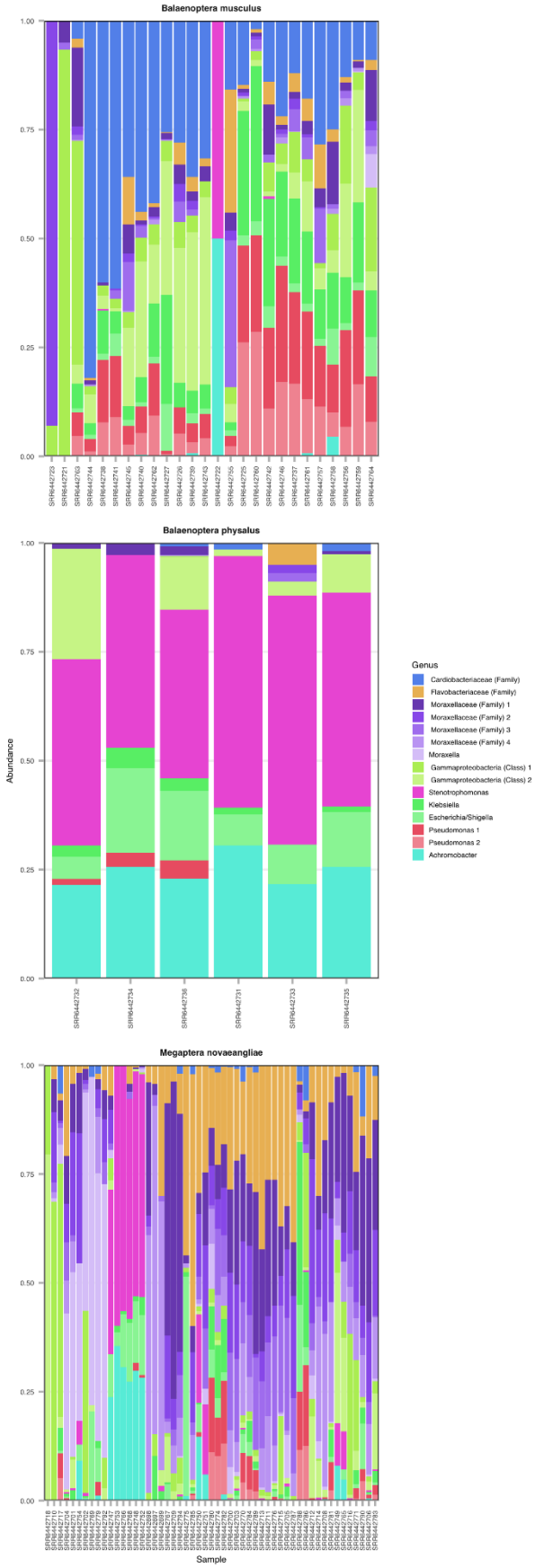
Podemos ver que existen patrones relativamente claros en el gráfico solamente a partir de los colores. Aunque tenemos el nombre de las muestras bajo cada barra, sería mejor poder parcelar este gráfico de manera que quede claro qué muestras corresponden a qué especie de ballena.

- La función `fantaxtic_bar` ofrece estas posibilidades a través de los argumentos `facet_by` y `grid_by`. Grafiquemos de nuevo.

```
fantaxtic_bar(top15, color_by = "Family", label_by = "Genus", facet_by =
"species", grid_by = NULL, other_co
```

```
##                          Level N.color.shades Central.color
## 1          Cardiobacteriaceae                1       #6495ED
## 2           Flavobacteriaceae                1       #EDBC64
## 3               Moraxellaceae                5       #9A64ED
## 4  Gammaproteobacteria (Class)                2       #B7ED64
## 5            Xanthomonadaceae                1       #ED64DA
## 6          Enterobacteriaceae                2       #64ED77
## 7            Pseudomonadaceae                2       #ED6473
## 8            Burkholderiaceae                1       #64EDDE
```

```
ptop15.2
```


Balaenoptera musculus
Balaenoptera physalus
Megaptera novaeangliae
Abundance
Sample
Genus
Cardiobacteriaceae (Family)
Flavobacteriaceae (Family)
Moraxellaceae (Family) 1
Moraxellaceae (Family) 2
Moraxellaceae (Family) 3
Moraxellaceae (Family) 4
Moraxella
Gammaproteobacteria (Class) 1
Gammaproteobacteria (Class) 2
Stenotrophomonas
Klebsiella
Escherichia/Shigella
Pseudomonas 1
Pseudomonas 2
Achromobacter

Ahora queda más claro y se puede observar que las distintas especies de ballena tienen un patrón similar entre sí que es diferente entre las otras, con algunas excepciones. Por ejemplo, para ballena jorobada podemos ver un conjunto de muestras que no se parecen al resto. Usando las herramientas ya aprendidas, ¿A qué corresponden esas muestras?

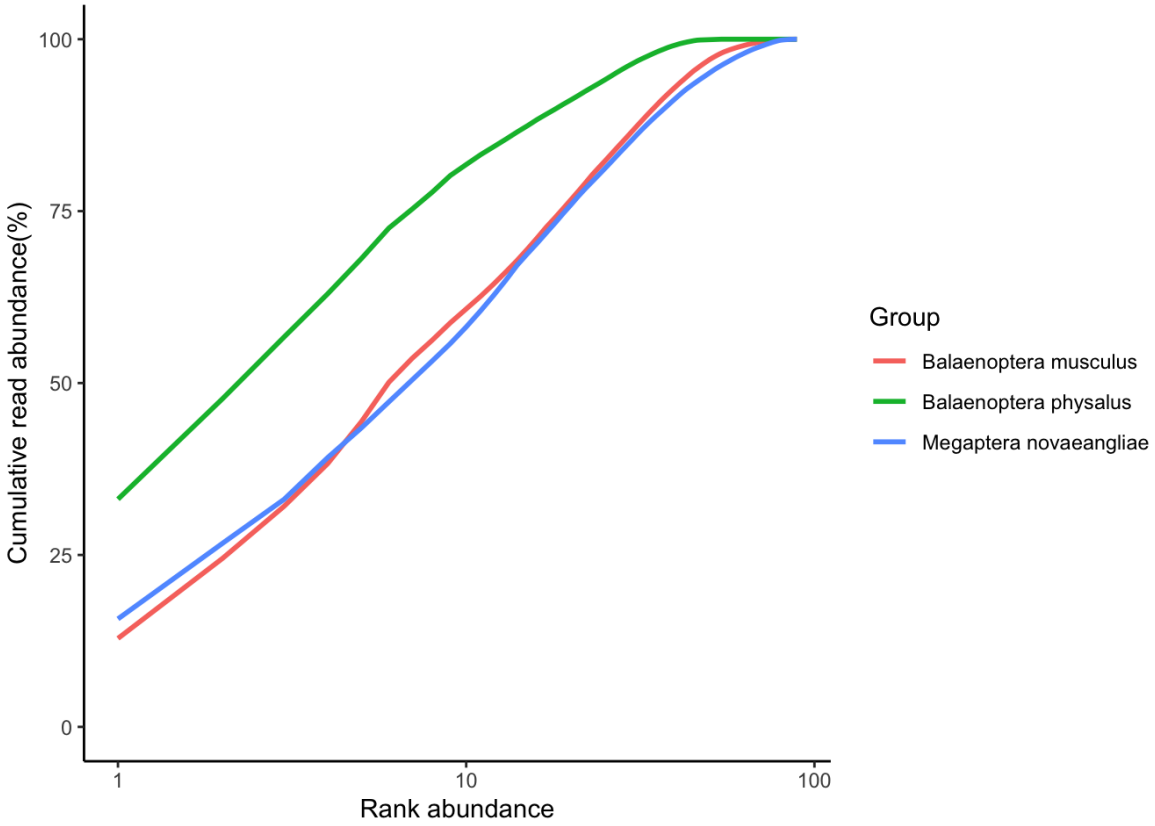
3.3.2 Diferentes visualizaciones de abundancias

Veamos ahora otras herramientas que nos permiten examinar la composición de estas comunidades microbianas. El paquete `ampvis2`, desarrollado por Mads Albertsen y Kasper Skytte Andersen, nos permite hacer precisamente esto. Primero transformemos el objeto `phyloseq` con el cual hemos estado trabajando en un objeto `ampvis2`.

```
library(ampvis2)

# Necesitamos extraer la tabla de read counts y la tabla de taxonomía del
objeto psd5
# Generamos una copia para no sobreescribir psd5
obj <- psd5
# Cambiamos la orientación de la otu_table
t(otu_table(obj)) -> otu_table(obj)
# Extraemos las tablas
otutable <- data.frame(OTU = rownames(phyloseq::otu_table(obj)@.Data),
                       phyloseq::otu_table(obj)@.Data,
                       phyloseq::tax_table(obj)@.Data,
                       check.names = FALSE
)
# Extraemos la metadada
metadata <- data.frame(phyloseq::sample_data(obj),
                       check.names = FALSE
)

# ampvis2 requiere que 1) los rangos taxonómicos sean siete y vayan de Kingdom
a Species y 2) la primera columna de la metadata sea el identificador de cada
muestra
# Entonces duplicamos la columna Género y le cambiamos el nombre a Especie
otutable$Species = otutable$Genus
# Reordenamos la metadata
metadata <-
metadata[,c("run","sample_ID","bioproject_accession","study","biosample_access
ion","experiment","SRA_Sample","geo_loc_name","collection_date","sample_type",
"species","common_name","AvgSpotLen")]

# finalmente generamos el objeto ampvis
av2 <- amp_load(otutable, metadata)
```

- Ahora echemos un vistazo a la estructura de las comunidades utilizando “rank abundance curves”.

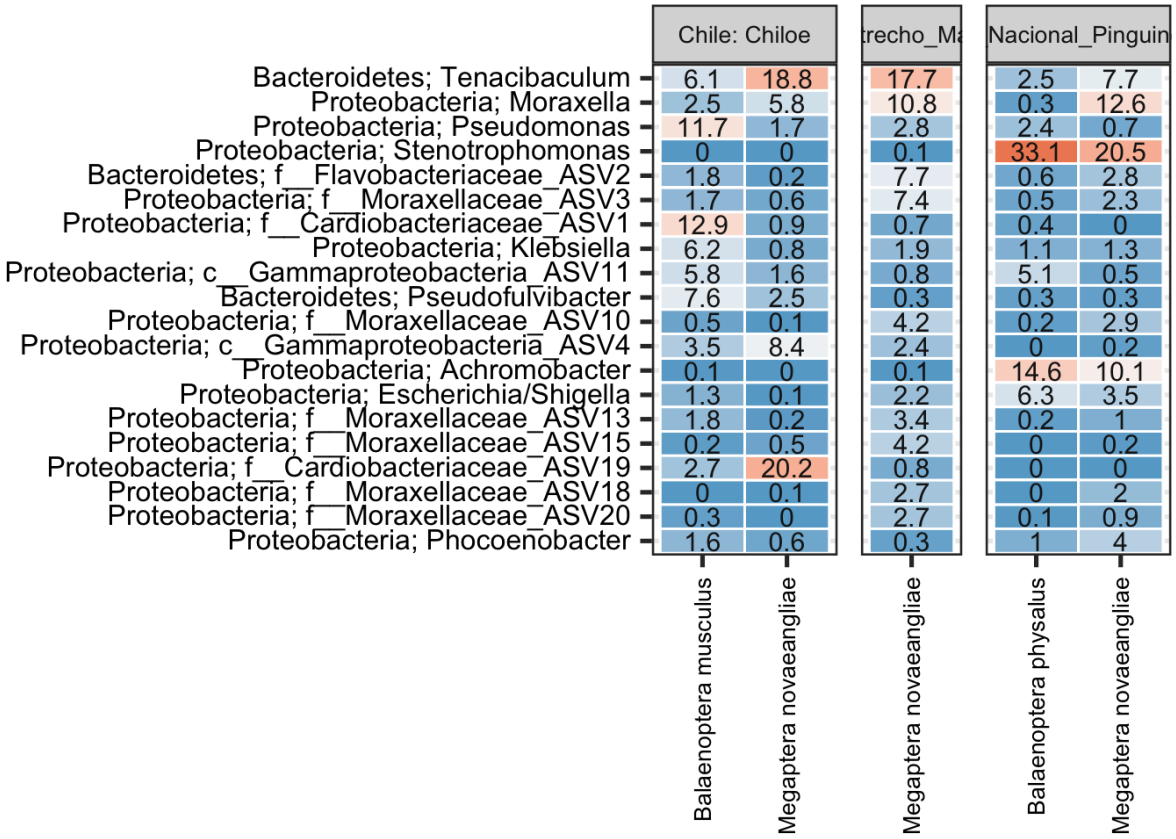
```
amp_rankabundance(av2, plot_log = T, group_by = "species")
```



El gráfico nos muestra que en la medida que vamos sumando las taxa de mayor a menor abundancia (*Rank Abundance*) la abundancia de reads cumulativa va aumentando. Lo importante de observar es la forma de la curva. Una curva que sube rápidamente nos indica que las comunidades están dominadas por unas cuantas taxa. En cambio, lo que observamos en nuestros datos es que las taxa más abundantes solamente dan cuenta de aproximadamente el 25% de la abundancia cumulativa de reads.

- Veamos ahora qué taxa corresponde a ese 25%. Para esto podemos usar la función `amp_heatmap`.

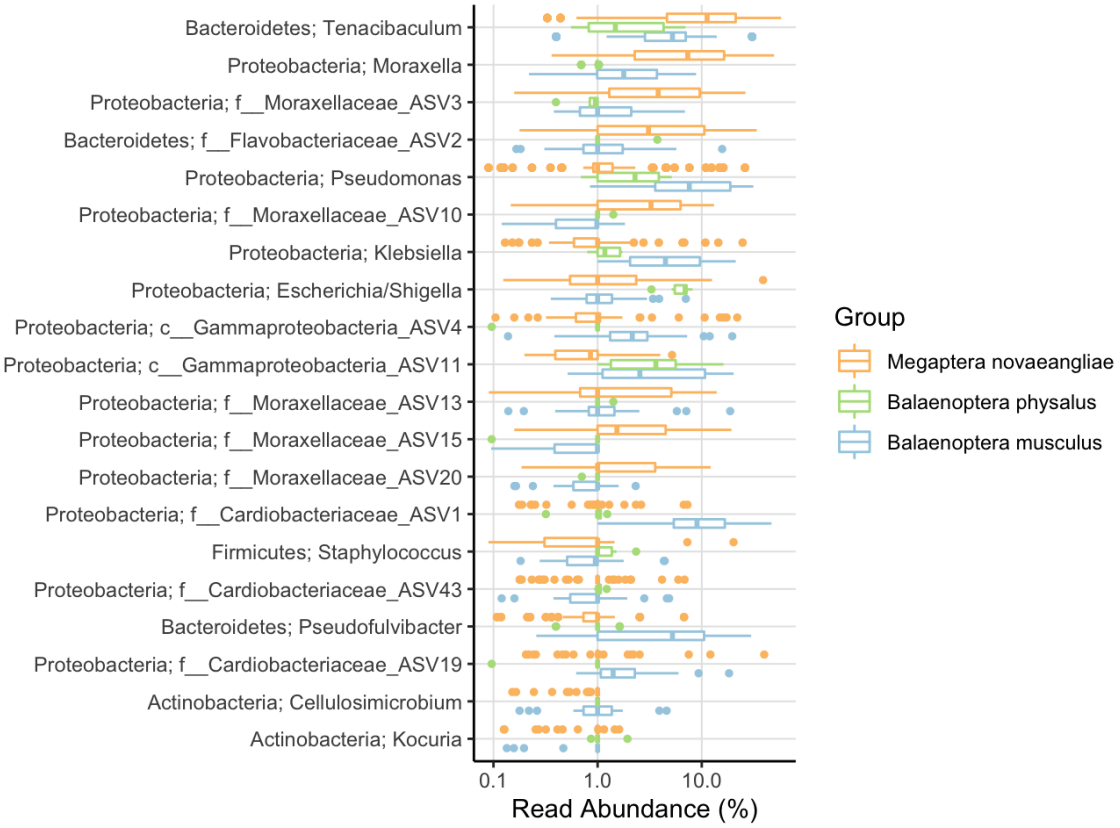
```
amp_heatmap(av2,
            group_by = "species",
            facet_by = "geo_loc_name",
            plot_values = TRUE,
            tax_show = 20,
            tax_aggregate = "Genus",
            tax_add = "Phylum",
            plot_colorscale = "sqrt",
            plot_legendbreaks = c(1, 5, 10)
            )
```



Tenacibaculum parece ser el género más abundante para todas las muestras en todas los sitios de muestreo con excepción de *Balaenoptera physalus* que está dominada por *Stenotrphomonas*. *Moraxella* y distintas variantes de Cardiobacteriaceae de género no conocido. Justamente estos resultados se ajustan a lo conocido para cetáceos y otros mamíferos marinos.

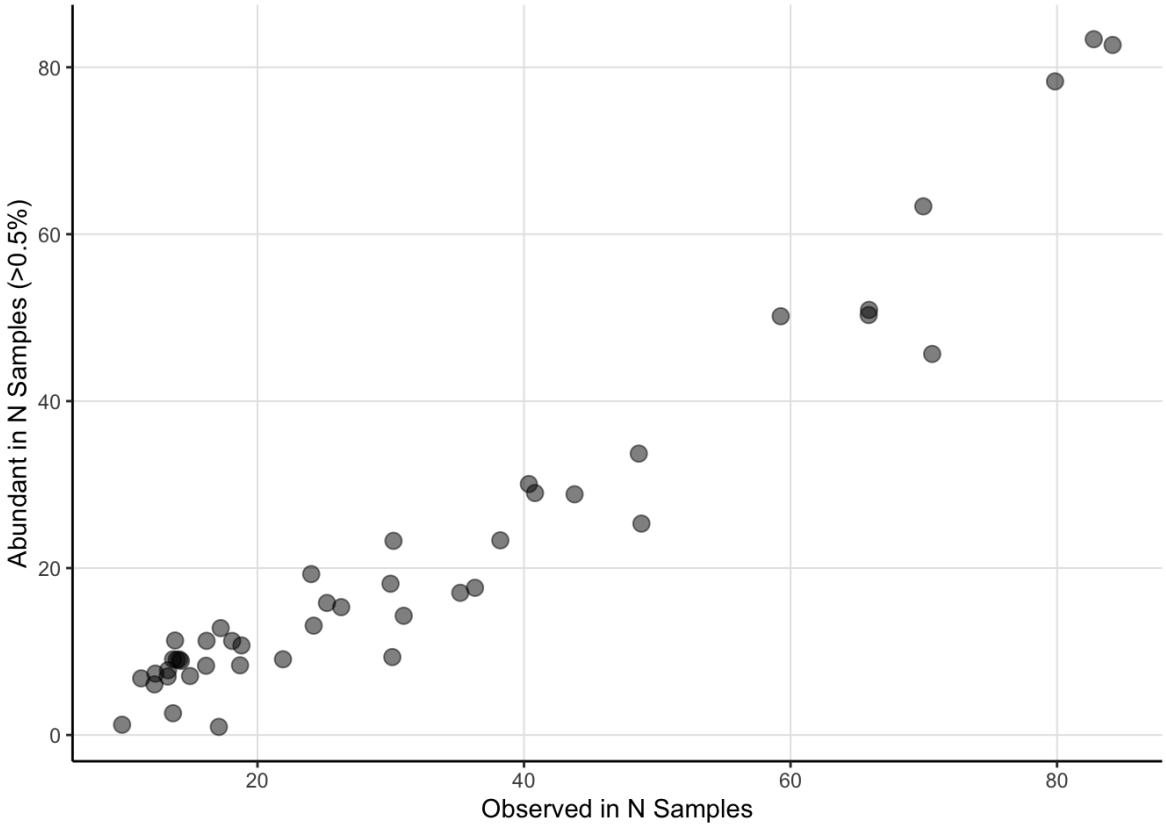
- También podemos realizar una visualización similar pero usando Box Plots.

```
amp_boxplot(av2,
            group_by = "species",
            tax_show = 20,
            tax_aggregate = "Genus",
            tax_add = "Phylum",
            adjust_zero = T,
            plot_log = T) +
  scale_color_manual(values = c("#a6cee3", "#b2df8a", "#fdbf6f")) +
  scale_fill_manual(values = c("#a6cee3", "#b2df8a", "#fdbf6f"))
```



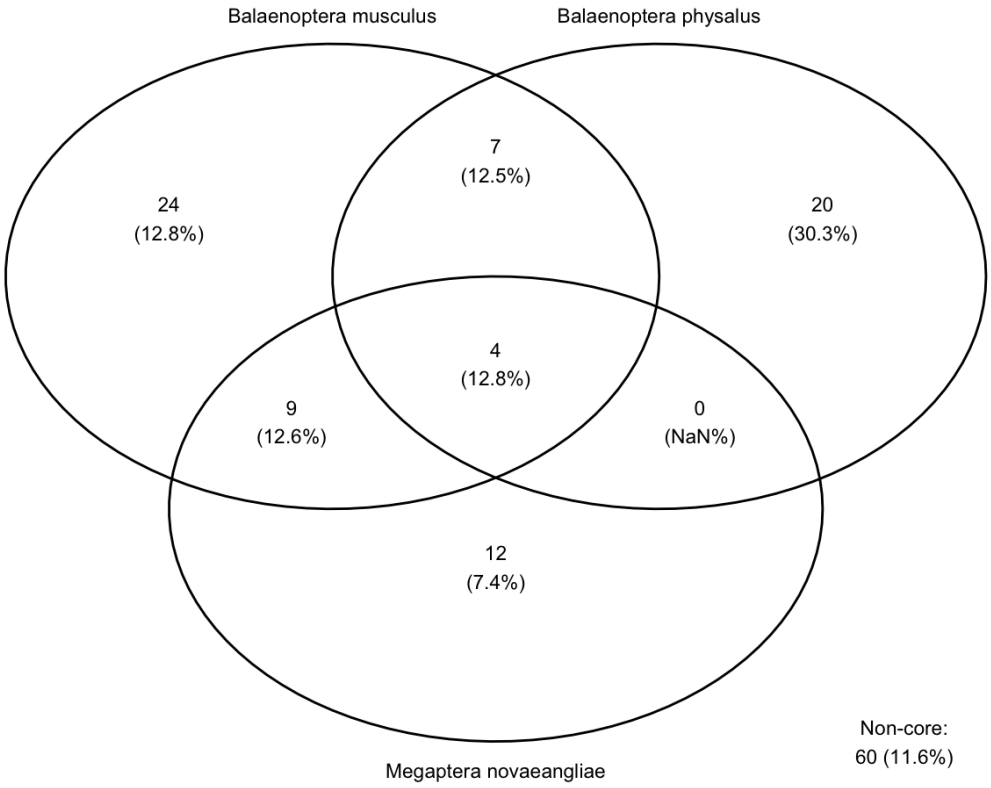
- Veamos ahora si es que algunos de estos microorganismos están compartidos entre todas las muestras. Para esto debemos calcular el *core microbiome* o el conjunto de taxa compartidas entre un cierto umbral porcentual de muestras y de prevalencia intra-muestra.

```
amp_core(av2,
         tax_aggregate = "Family",
         group_by = "Sample",
         abund_thrh = 0.5)
```



- Y visto de otra manera en un diagrama de Venn.

```
amp_venn(av2, group_by = "species", cut_a = 0, cut_f = 50, text_size = 3)
```



3.3.3 Análisis de abundancia diferencial de microorganismos

Hasta ahora hemos visto principalmente análisis exploratorios y algunos test estadísticos para diversidad alfa y beta. Sin embargo, muchas veces queremos determinar exactamente qué taxa está más representada en una condición versus otra y en qué medida. El procedimiento es análogo al análisis de expresión diferencial de genes en transcritómica, e.g., RNA-seq. Es tan así que justamente ocupamos uno de los paquetes de R más populares en transcriptómica, **DESeq2**. Ahora, nuestras muestras se secuenciaron al mismo tiempo y se intentó que se produjera una profundidad uniforme a través de todas las muestras. En la práctica esto no ocurre y al momento de analizar las muestras en el contexto del **análisis diferencial de abundancia** debemos corregir por dos situaciones: el tamaño desigual de las muestras (en número de reads) y la correlación de la variabilidad de los datos con la media de éstos. Para esto último, utilizamos una transformación llamada **Variance Stabilizing Transformation**, cuyo objetivo es encontrar una función f que se aplique a los valores de read counts, de tal forma que en los nuevos valores $y = f(x)$, la variabilidad de y no se relacione con sus valores medios.

- Comencemos por aplicar la transformación en los datos.

```
# Creamos un objeto DESeq2 con la función `phyloseq_to_deseq2`
diagdds = phyloseq_to_deseq2(psd5, ~species)
# Calculamos los factores de tamaño como parte de la normalización de las
muestras
# calculate geometric means prior to estimate size factors
gm_mean = function(x, na.rm=TRUE){
    exp(sum(log(x[x > 0]), na.rm=na.rm) / length(x))
}
geoMeans = apply(counts(diagdds), 1, gm_mean)
diagdds = estimateSizeFactors(diagdds, geoMeans = geoMeans)

# Normalizamos y realizamos el test paramétrico de Wald para determinar taxa
diferencialmente abundante.
diagdds = DESeq(diagdds, test="Wald", fitType="local")
```

Hasta ahora hemos transformado nuestro objeto `phyloseq` en un objeto `DESeq2` de nombre `diagdds`, y hemos normalizado las cuentas y realizado un test paramétrico (Wald Test).

- Nos queda entonces revisar los resultados usando la función `results`.

```
# Guardamos los resultados en el objeto res
res = results(diagdds, cooksCutoff = FALSE)
# hacemos un poco de aseo y ordenamos la tabla de resultados según p-value, y
dejamos los valores NA al final
res = res[order(res$padj, na.last=NA), ]
```

- Ahora nosotros queremos averiguar sobre ciertos contrastes específicos entre condiciones, e.g., ballena jorobada versus ballena azul. En el contraste, pasamos un vector con el nombre de la columna en la metadata (“species”) e indicamos el numerador de la comparación (“Megaptera novaeangliae”) y el denominador (“Balaenoptera musculus”). Por lo tanto el `log2FoldChange` positivo indicará que ese microorganismo es más abundante en ballena jorobada que en ballena azul, y viceversa.

```
res.joro.azul <- results(diagdds, contrast=c("species","Megaptera
novaeangliae","Balaenoptera musculus"))
```

- Descarga el archivo `res.joro.azul.RDS` **AQUÍ**
- Veamos qué hay en `res.joro.azul`

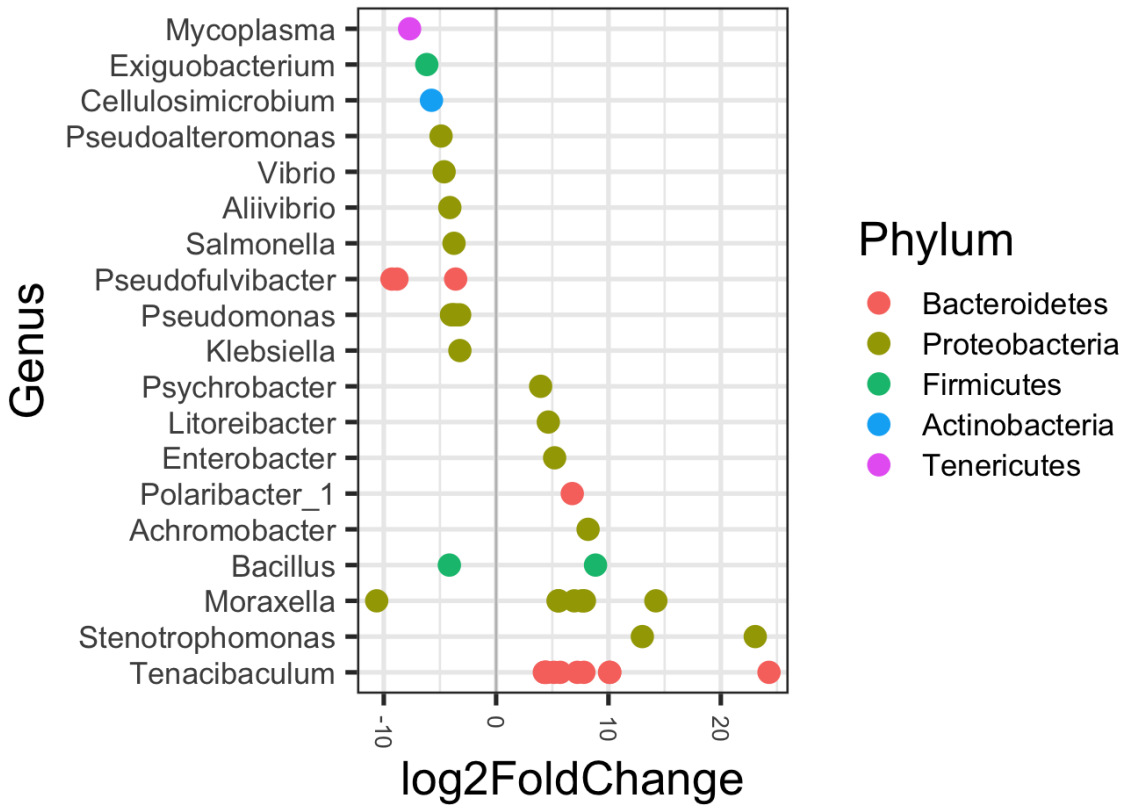
http://bioconductor.org/packages/devel/bioc/vignettes/DESeq2/inst/doc/DESeq2.html#more-information-on-results-columns

tabla 3.1

¿Qué significa cada columna? Revisa la viñeta de `DESeq2` aquí.

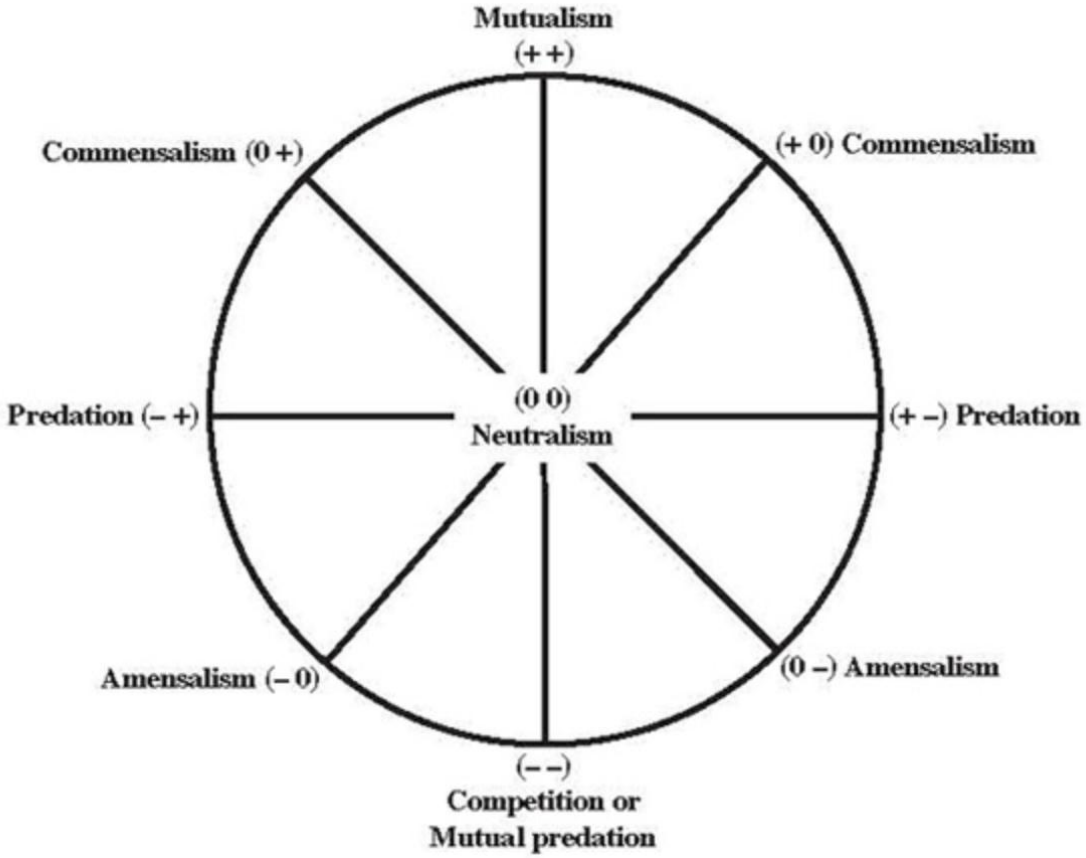
- Ahora establezcamos un umbral de significancia estadística para los valores de *p-value* ajustado o `padj`. Cualquier resultado bajo este umbral será considerado no significativo y viceversa.

```
# Este es nuestro umbral
alpha = 0.01
# Ordenamos la tabla de resultados
res.joro.azul = res.joro.azul[order(res.joro.azul$padj, na.last=NA), ]
# Filtramos según nuestro umbral alpha
sigtab = res.joro.azul[(res.joro.azul$padj < alpha), ]
# Le agregamos la taxonomía a la tabla
sigtab = cbind(as(sigtab, "data.frame"), as(tax_table(psd5)[rownames(sigtab),
], "matrix"))

# Manipulaciones varias para finalmente graficar los resultados
sigtabgen = subset(sigtab, !is.na(Genus))
# Phylum order
x = tapply(sigtabgen$log2FoldChange, sigtabgen$Phylum, function(x) max(x))
x = sort(x, TRUE)
sigtabgen$Phylum = factor(as.character(sigtabgen$Phylum), levels=names(x))
# Genus order
x = tapply(sigtabgen$log2FoldChange, sigtabgen$Genus, function(x) max(x))
x = sort(x, TRUE)
sigtabgen$Genus = factor(as.character(sigtabgen$Genus), levels=names(x))
ggplot(sigtabgen, aes(y=Genus, x=log2FoldChange, color=Phylum)) +
    geom_vline(xintercept = 0.0, color = "gray", size = 0.5) +
    geom_point(size=4) +
    theme(axis.text.x = element_text(angle = -90, hjust = 0, vjust=0.5, size =
10), axis.text.y = element_text(size = 13), legend.text = element_text(size =
13) )
```



3.3.4 Redes de co-ocurrencia

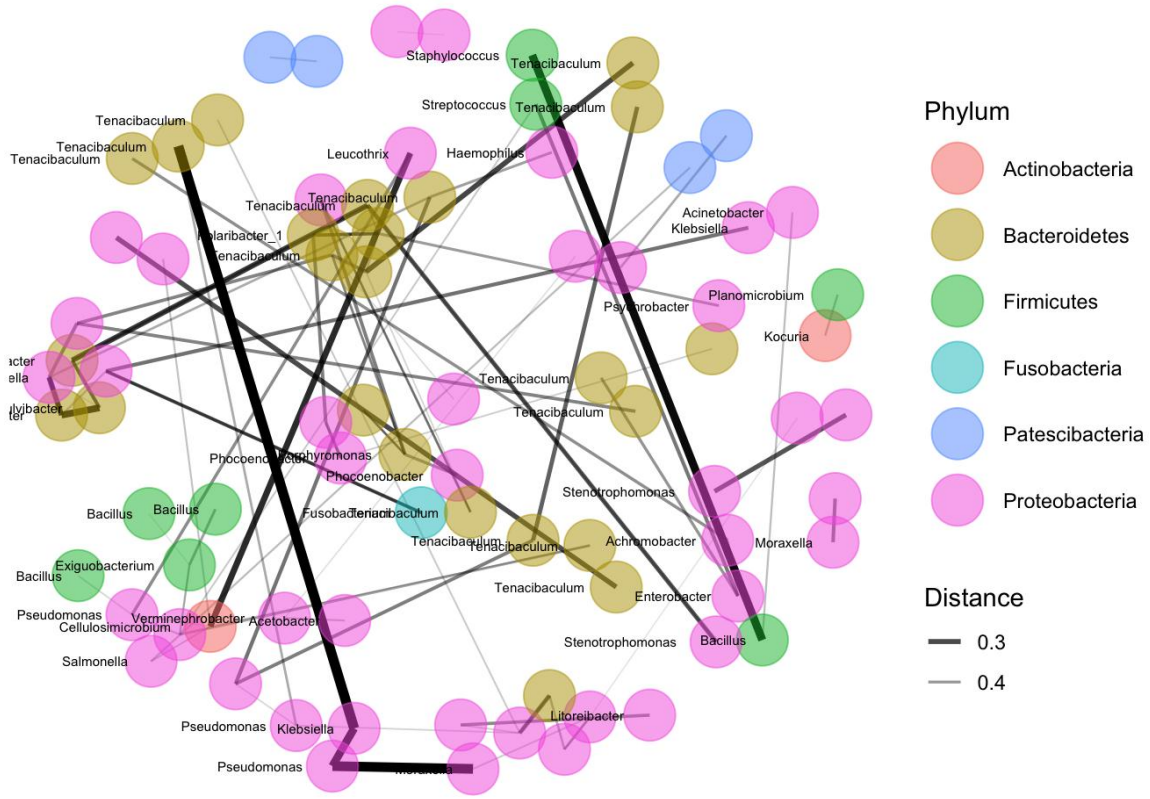
Para finalizar, vamos a echar un vistazo a las capacidades de `phyloseq` para generar redes de co-occurrencia. Las redes de co-ocurrencia nos dan pistas sobre potenciales interacciones ecológicas entre organismos. Estas interacciones pueden ser directas o indirectas (no lo podemos determinar a partir de una red) y nos permiten comenzar a descifrar mecanismos ecológicos detrás de la composición de una comunidad microbiana. En general en ecología tenemos distintos tipos de interacciones:



Donde destacan depredación, competición o depredación mutua, y mutualismo. Cada una de estas relaciones podría ser detectada en una red de co-ocurrencia según patrones de correlación positivos o negativos.

- Veamos como generaríamos una red en `phyloseq`.

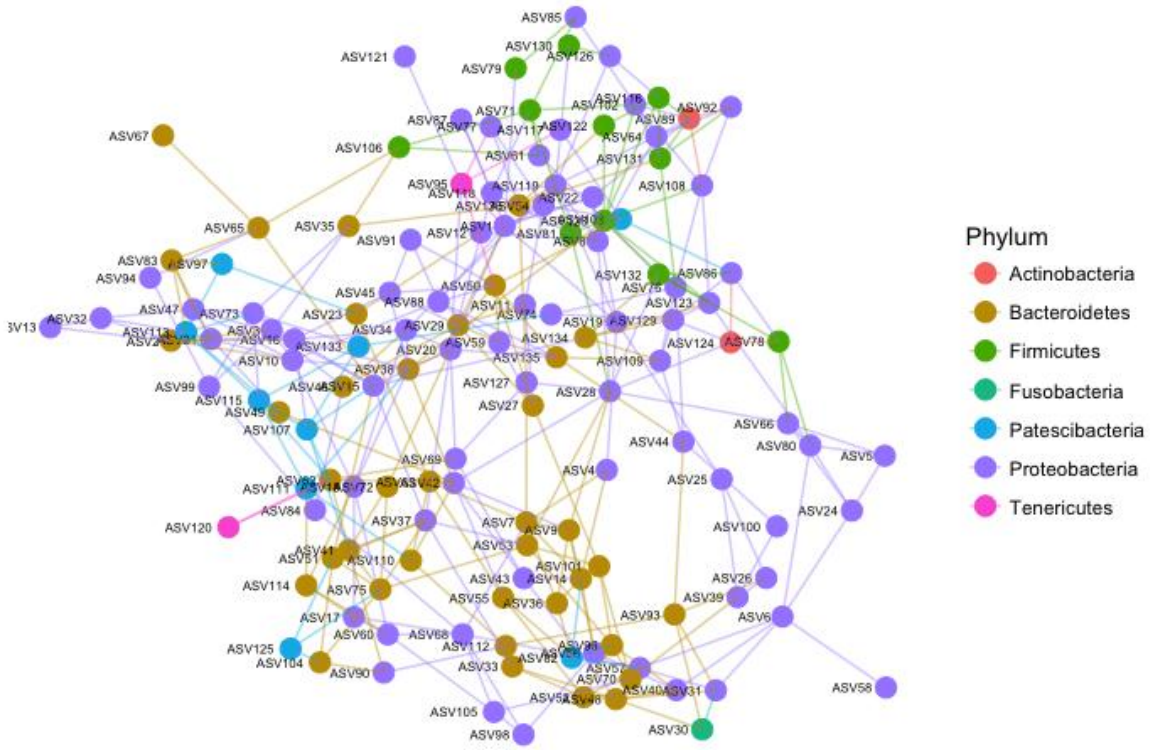
```
plot_net(psd5, type = "taxa", point_label = "Genus", point_size = 10,
point_alpha = 0.5, maxdist = 0.5, color = "Phylum", distance = "bray", laymeth
= "auto")
```



La red generada con `phyloseq` no es una red de co-ocurrencia propiamente tal. Es más bien una red que representa relaciones de distancia entre taxa o muestras. En nuestro ejemplo usamos muestras. Para una red de co-ocurrencia propiamente tal necesitamos usar las funciones del paquete `SpiecEasi`.

- Veamos un ejemplo de cómo hacerlo.

```
se.mb.psd5 <- spiec.easi(psd5, method='mb', lambda.min.ratio=1e-2,
                         nlambda=20, icov.select.params=list(rep.num=50))
ig2.mb <- adj2igraph(se.mb.psd5$refit,
vertex.attr=list(name=taxa_names(psd5)))
plot_network(ig2.mb, psd5, type='taxa', color="Phylum")
```


Phylum
Actinobacteria
Bacteroidetes
Firmicutes
Fusobacteria
Patescibacteria
Proteobacteria
Tenericutes