Bioestadística en R - parte práctica clase 1, Módulo 1 : Análisis de la diversidad.

Ph D. Stephanie Hereira Pacheco

Contents

Parte 1 **1**
`hillR` . . . 1
Cálculo de la diversidad taxonómica, funcional y filogenética de cada sitio o muestra (alfa diversidad). . . . 3
Cálculo de la diversidad taxonómica, funcional y filogenética de a través de diferentes sitios o muestras. . . . 5
Cálculo de la diversidad pareada taxonómica, funcional y filogenética . . . 5
Ejemplo aplicado . . . 7
`betapart` . . . 7
Ejemplo aplicado . . . 10
`vegan` . . . 11

Parte 2 **12**
iNEXT . . . 12
Ejemplo aplicado . . . 19
Curvas en vegan . . . 20

REFERENCIAS **22**

Parte 1

El propósito de este cuadernillo es aplicar el cálculo u obtención de algunos conocimientos teóricos impartidos en la clase 1 en el módulo I del curso de Bioestadística en R. Hay varios paquetes y funciones en R que nos permitirán explorar nuestros datos ecológicos. Inicialmente veremos cómo calcular los índices de diversidad en el marco de los números de Hill utilizando el paquete `hillR` , luego exploraremos el paquete `betapart` que nos permitirá obtener la partición de la diversidad beta. De igual forma trataremos revisaremos un poco del paquete `vegan` que nos permite calcular otros índices de diversidad y otras matrices de distancia de la diversidad beta.

`hillR`[1]

El paquete `hillR` contiene funciones para calcular la diversidad taxonómica, funcional y filogenética en el marco de los números de Hill. Los métodos están basados en la referencia (Chao, Chiu, and Jost 2014) que ya hemos tratado en la parte teórica.

Para instalar este paquete usamos el comando:

[1] https://github.com/daijiang/hillR

```
install.packages("hillR")
# o instala la versión en desarrollo del github
devtools::install_github("daijiang/hillR")
```

También instalaremos el paquete `FD` de dónde usaremos la data ejemplo:

```
install.packages("FD")
```

Usaremos la data **dummy** de ejemplo que viene en el paquete `FD` para utilizar las funciones:

```
set.seed(123)
library(FD)
dummy_data <- dummy
comunidades<-  dummy_data$abun
funciones <- dummy_data$trait
arbol <- ape::rtree(n = ncol(comunidades), tip.label = paste0("sp", 1:ncol(comunidades)))
```

En este caso, Lo primero es usar la función **'set.seed'** o sembrar semilla, para indicarle a R que nos guarde los cálculos en esta semilla y cada vez que lo corramos den el mismo valor (eso es cuando son muestreos aleatorios como es el caso de la función *rtree* del paquete 'ape'. Exploremos los tipos de datos que tenemos:

```
class(comunidades)
```

```
## [1] "matrix" "array"
```

```
class(funciones)
```

```
## [1] "data.frame"
```

```
class(arbol)
```

```
## [1] "phylo"
```

Definimos nuestras datas a usar:

- comunidades: tabla con counts o recuentos de especies (columnas) por sitio o muestra (filas).
- funciones : tabla con funciones y resultados (numéricos o categóricos) por especie descrita en nuestra tabla de comunidades. Especies como filas y funciones como columnas.
- arbol: objeto tipo "phylo" tipo lista con vértices y nodos describiendo relaciones filogenéticas.

Veamos:

```
head(comunidades)
```

```
##      sp1 sp2 sp3 sp4 sp5 sp6 sp7 sp8
## com1   1   1   0   0   4   2   0   0
## com2   0   0   0   2   1   0   0   5
## com3   2   0   0   0   0   1   0   3
## com4   1   0   7   0   0   0   0   0
## com5   0   0   2   3   3   0   0   0
## com6   0   3   0   0   5   6   1   6
```

```
head(funciones)
```

```
##     num1 num2 fac1 fac2 ord1 ord2 bin1 bin2
## sp1  9.0  4.5    A    X    3    2    0    1
## sp2  8.1  6.0    A    Z <NA>    1    0    1
```

```
## sp3   NA  2.3    C    Y    5    3    1    1
## sp4  3.2  5.4    B    Z    1    7    0    0
## sp5  5.8  1.2    C    X    2    6   NA    0
## sp6  3.4  8.5    C    Y    2    1    1    1
```

```
head(arbol)
```

```
## $edge
##       [,1] [,2]
##  [1,]    9   10
##  [2,]   10   11
##  [3,]   11   12
##  [4,]   12    1
##  [5,]   12    2
##  [6,]   11    3
##  [7,]   10   13
##  [8,]   13   14
##  [9,]   14   15
## [10,]   15    4
## [11,]   15    5
## [12,]   14    6
## [13,]   13    7
## [14,]    9    8
##
## $tip.label
## [1] "sp2" "sp6" "sp3" "sp5" "sp4" "sp8" "sp7" "sp1"
##
## $Nnode
## [1] 7
##
## $edge.length
##  [1] 0.10292468 0.89982497 0.24608773 0.04205953 0.32792072 0.95450365
##  [7] 0.88953932 0.69280341 0.64050681 0.99426978 0.65570580 0.70853047
## [13] 0.54406602 0.59414202
```

Cálculo de la diversidad taxonómica, funcional y filogenética de cada sitio o muestra (alfa diversidad).

Primero cargamos la librería `hillR` previamente instalada:

```
library(hillR)
```

Luego calculamos la diversidad taxonómica al orden q=0 (podemos escoger qué número de orden, recuerden q=0 es igual a riqueza), de la siguiente manera:

```
hill_taxa(comunidades, q = 0)
```

```
##  com1  com2  com3  com4  com5  com6  com7  com8  com9 com10
##     4     3     3     2     3     5     3     4     5     4
```

De igual forma la diversidad funcional al mismo orden:

```
hill_func(comunidades, funciones, q = 0)
```

```
##           com1      com2      com3      com4      com5      com6      com7
## Q    0.4016663 0.1922618 0.2780442 0.1146261 0.3816159  0.404177 0.2934143
## FDis 0.3481687 0.1670560 0.2375808 0.1146261 0.3211366  0.330233 0.2532751
## D_q  4.0974923 3.6518111 3.2454591 3.0237158 3.0655375  5.233241 3.1470056
## MD_q 1.6458245 0.7021037 0.9023810 0.3465969 1.1698580  2.115156 0.9233765
## FD_q 6.7437533 2.5639502 2.9286406 1.0480104 3.5862436 11.069121 2.9058708
##           com8       com9     com10
## Q    0.3343662  0.4156546 0.3844765
## FDis 0.2877931  0.3421687 0.3503927
## D_q  4.3998540  5.2114653 4.1694097
## MD_q 1.4711625  2.1661695 1.6030400
## FD_q 6.4729004 11.2889174 6.6837303
```

Y por último la diversidad filogenética al mismo orden:

```
hill_phylo(comunidades, arbol, q = 0)
```

```
##     com1     com2     com3     com4     com5     com6     com7     com8
## 5.430079 4.684280 4.461773 2.551395 5.830078 6.088533 4.763594 6.046474
##     com9    com10
## 6.262164 5.080340
```

Los resultados que nos da cada función son:

1. **hill_taxa()** nos da un vector con el va valor de diversidad alfa para cada sitio o muestra, $q = 0$ (por defecto) para obtener la riqueza de especies , $q = 1$ para obtener la entropía de shannon (especies frecuentes) y $q = 2$ nos dará el inverso de simpson (especies dominantes).

2. **hill_func()** nos dará una matrix con la información de:

- Q : Q de Rao,
- D_q : el numero efectivo de epecies distintas igualmente abundantes y funcionales
- MD_q : diversidad funcional media por especie, la suma efectiva de las distancias por pares entre una especie fija y todas las demás especies
- FD_q : diversidad funcional total, la distancia funcional total efectiva entre especies del conjunto

  Más detalle de estos índices de diversidad funcional consultar la referencia "Chiu, Chun-Huo, and Anne Chao. Distance-Based Functional Diversity Measures and Their Decomposition: A Framework Based on Hill Numbers, 2014". (Chiu and Chao 2014)

3. **hill_phylo** nos dará un vector de diversidad filogenética basada en el número de Hill ('PD(T)', longitud total efectiva de la rama) para todos los sitios.

Si queremos calcular a otra orden solo ponemos la que queremos, por ejemplo:

```
hill_taxa(comunidades, q = 1)
```

```
##     com1     com2     com3     com4     com5     com6     com7     com8
## 3.363586 2.460233 2.749459 1.457569 2.951152 4.395212 2.906907 3.217222
##     com9    com10
## 4.341153 3.086164
```

```
hill_taxa(comunidades, q = 2)
```

```
##     com1     com2     com3     com4     com5     com6     com7     com8
## 2.909091 2.133333 2.571429 1.280000 2.909091 4.121495 2.813953 2.688889
##     com9    com10
## 3.903226 2.666667
```

Cálculo de la diversidad taxonómica, funcional y filogenética de a través de diferentes sitios o muestras.

Este script calcula la diversidad gamma, alfa, y beta a traves de todas las comunidades o muestras asi como su similitud.

Si comm>2 la gamma diversidad es la diversidad juntada (pooled) del ensamble, la alfa es el promedio de la diversidad a través de todos los sitios y la beta es a través de todas las comunidades.

También nos da la medida de homogeneidad de MacArthur, la similtud local (traslape de especies / overlap similar al de Sorensen) y la similitud regional (traslape de especies / overlap similar al de Jaccard).

Esto para las comunidades:

```
hill_taxa_parti(comunidades, q = 0)
```

```
##   q TD_gamma TD_alpha  TD_beta M_homog local_similarity region_similarity
## 1 0        8      3.6 2.222222    0.45        0.8641975         0.3888889
```

Funciones:

```
hill_func_parti(comunidades, funciones, q = 0)
```

```
##   q raoQ_gamma FD_gamma FD_alpha FD_beta local_similarity region_similarity
## 1 0  0.4529152 29.66099 14.15941 2.09479        0.9889415         0.4720957
```

Y filogenias:

```
hill_phylo_parti(comunidades, arbol, q = 0)
```

```
##   q PD_gamma PD_alpha  PD_beta local_similarity region_similarity
## 1 0 8.292885 5.119871 1.619745        0.9311395          0.574868
```

Cálculo de la diversidad pareada taxonómica, funcional y filogenética

Calcula la diversidad pareada gamma, alfa y beta para las comunidades así como la similitud. La diversidad pareada hace el cáclulo entre pares de sitios o muestreasw generando ya sea un data.frame o una matriz de distancia que puede ser usada para futuros análisis multivareado. En este ejemplo y por defecto obtendremos el data.frame para la diversidad taxonómica:

```
hill_taxa_parti_pairwise(comunidades, q = 0, .progress = FALSE)
```

```
## # A tibble: 45 x 8
##       q site1 site2 TD_gamma TD_alpha TD_beta local_similarity region_similar~1
```

```
##    <dbl> <chr> <chr>    <dbl>    <dbl>   <dbl>            <dbl>            <dbl>
##  1     0 com1  com2         6      3.5    1.71            0.286            0.167
##  2     0 com1  com3         5      3.5    1.43            0.571            0.4
##  3     0 com2  com3         5      3      1.67            0.333            0.2
##  4     0 com1  com4         5      3      1.67            0.333            0.2
##  5     0 com2  com4         5      2.5    2               0                0
##  6     0 com3  com4         4      2.5    1.6             0.4              0.25
##  7     0 com1  com5         6      3.5    1.71            0.286            0.167
##  8     0 com2  com5         4      3      1.33            0.667            0.5
##  9     0 com3  com5         6      3      2               0                0
## 10     0 com4  com5         4      2.5    1.6             0.4              0.25
## # ... with 35 more rows, and abbreviated variable name 1: region_similarity
```

Ahora para la diversidad funcional:

```
hill_func_parti_pairwise(comunidades, funciones, q = 0, show_warning = FALSE, .progress =FALSE)
```

```
## # A tibble: 45 x 8
##        q site1 site2 FD_gamma FD_alpha FD_beta local_similarity region_similar~1
##    <dbl> <chr> <chr>    <dbl>    <dbl>   <dbl>            <dbl>            <dbl>
##  1     0 com1  com2      15.9     10.3    1.54            0.821            0.534
##  2     0 com1  com3      10.7      7.96   1.35            0.883            0.655
##  3     0 com2  com3      11.1      7.03   1.58            0.807            0.511
##  4     0 com1  com4      11.6      7.90   1.47            0.843            0.573
##  5     0 com2  com4      11.7      6.85   1.70            0.765            0.449
##  6     0 com3  com4       6.60     4.45   1.48            0.839            0.566
##  7     0 com1  com5      17.3     11.3    1.54            0.821            0.534
##  8     0 com2  com5       7.86     5.92   1.33            0.891            0.671
##  9     0 com3  com5      16.2      9.72   1.66            0.780            0.469
## 10     0 com4  com5       8.00     5.32   1.50            0.832            0.554
## # ... with 35 more rows, and abbreviated variable name 1: region_similarity
```

Y también para la diversidad filogenética:

```
hill_phylo_parti_pairwise(comunidades, arbol, q = 0, show_warning = FALSE, .progress = FALSE)
```

```
## # A tibble: 45 x 8
##        q site1 site2 PD_gamma PD_alpha PD_beta local_similarity region_similar~1
##    <dbl> <chr> <chr>    <dbl>    <dbl>   <dbl>            <dbl>            <dbl>
##  1     0 com1  com2      6.79     5.06    1.34            0.657            0.489
##  2     0 com1  com3      6.14     4.95    1.24            0.759            0.611
##  3     0 com2  com3      6.75     4.57    1.48            0.523            0.355
##  4     0 com1  com4      6.38     3.99    1.60            0.400            0.250
##  5     0 com2  com4      7.13     3.62    1.97            0.0284           0.0144
##  6     0 com3  com4      5.42     3.51    1.54            0.455            0.295
##  7     0 com1  com5      7.04     5.63    1.25            0.750            0.599
##  8     0 com2  com5      6.54     5.26    1.24            0.756            0.608
##  9     0 com3  com5      7.71     5.15    1.50            0.502            0.335
## 10     0 com4  com5      6.42     4.19    1.53            0.467            0.305
## # ... with 35 more rows, and abbreviated variable name 1: region_similarity
```

Veamos el ejemplo de cuando queremos obtener las matrices de distancia al orden 0:

```
beta_q0_mat<-hill_taxa_parti_pairwise(
  comunidades, q = 0, .progress = FALSE,
```

```
  output = "matrix", pairs = "full")
class(beta_q0_mat)
```

```
## [1] "list"
```

Esta vez en vez de un data.frame nos dará una lista que contendrá 6 elementos, definidos así:

q = el orden al cual fue realizado el cálculo

TD_gamma= gamma pareada

TD_alpha=alpha pareada

Td_beta: dependiendo del orden q (q=0 Jaccard/Sorensen, q=1 Horn y q=2 Morisita-Horn).

local_similarity : traslape de especies similar a Sorensen (q=0)

region_similarity: traslape de especies similar a Jaccard (q=0)

Ejemplo aplicado

Tomemos la data 'spiders' del paquete iNEXT que veremos más adelante. Para más información de estos puedes consultar la referencia (Sackett et al. 2011).

```
library(iNEXT)
data("spider")
```

¿Cuál de los dos tratamientos da valores más altos de riqueza?

betapart

Revisaremos el paquete `betapart`(Baselga and Orme 2012) del mismo autor que introduce los conceptos de partición de la diversidad beta en los componentes de recambio (turnover) y anidamiento (nestedness).

En este paquete genera las funciones para calcular diversidad beta teniendo en cuenta esta partición que vimos previamente en la parte teórica. Para instalarlo ponemos la siguiente línea de código:

```
install.packages("betapart")
```

Primero llamamos al paquete previamente instalado. Luego definiremos el conjunto de datos de ejemplo, y diremos que las tres primeras columnas pertenecen a un bosque no perturbado y las tres siguientes a un bosque no perturbado:

```
library(betapart)
comm<- data.frame(comm =1:6,
                          sp1=c(2,2,3,0,0,1),
                          sp2=c(2,2,0,1,1,2),
                          sp3=c(1,0,1,2,3,2),
                          sp4=c(1,0,1,0,2,0),
                          sp5=c(1,2,0,0,0,1),
                          sp6=c(2,2,1,0,0,0),
                          sp7=c(0,0,0,1,0,1),
                          sp8=c(1,0,1,0,1,0), row.names = 1)


groups <- factor(c(rep(1,3), rep(2,3)),
                 labels = c("noperturbado","perturbado"))
head(comm);groups
```

```
##   sp1 sp2 sp3 sp4 sp5 sp6 sp7 sp8
```

```
## 1   2   2   1   1   1   2   0   1
## 2   2   2   0   0   2   2   0   0
## 3   3   0   1   1   0   1   0   1
## 4   0   1   2   0   0   0   1   0
## 5   0   1   3   2   0   0   0   1
## 6   1   2   2   0   1   0   1   0
```

```
## [1] noperturbado noperturbado noperturbado perturbado   perturbado
## [6] perturbado
## Levels: noperturbado perturbado
```

Los índices Jaccard y Sorensen en los que está basado este paquete no consideran la abundancia de especies (q=0), así que convertiremos nuestra data a tipo incidencia (presencia/ausencia):

```
presabs<-ifelse(comm>0,1,0)
```

Ahora, calcularemos el índice de Sorensen y sus particiones de recambio y anidamiento. En su lugar, podemos calcular el índice de Jaccard usando el argumento index.family="jaccard".

```
dist_comp<-beta.pair(presabs, index.family="sorensen")
```

Vemos que el resultado es una lista de tres matrices:

```
head(dist_comp)
```

```
## $beta.sim
##           1         2         3         4         5
## 2 0.0000000
## 3 0.0000000 0.5000000
## 4 0.3333333 0.6666667 0.6666667
## 5 0.0000000 0.7500000 0.2500000 0.3333333
## 6 0.2000000 0.2500000 0.6000000 0.0000000 0.5000000
##
## $beta.sne
##            1          2          3          4          5
## 2 0.27272727
## 3 0.16666667 0.05555556
## 4 0.26666667 0.04761905 0.08333333
## 5 0.27272727 0.00000000 0.08333333 0.09523810
## 6 0.13333333 0.08333333 0.00000000 0.25000000 0.05555556
##
## $beta.sor
##           1         2         3         4         5
## 2 0.2727273
## 3 0.1666667 0.5555556
## 4 0.6000000 0.7142857 0.7500000
## 5 0.2727273 0.7500000 0.3333333 0.4285714
## 6 0.3333333 0.3333333 0.6000000 0.2500000 0.5555556
```

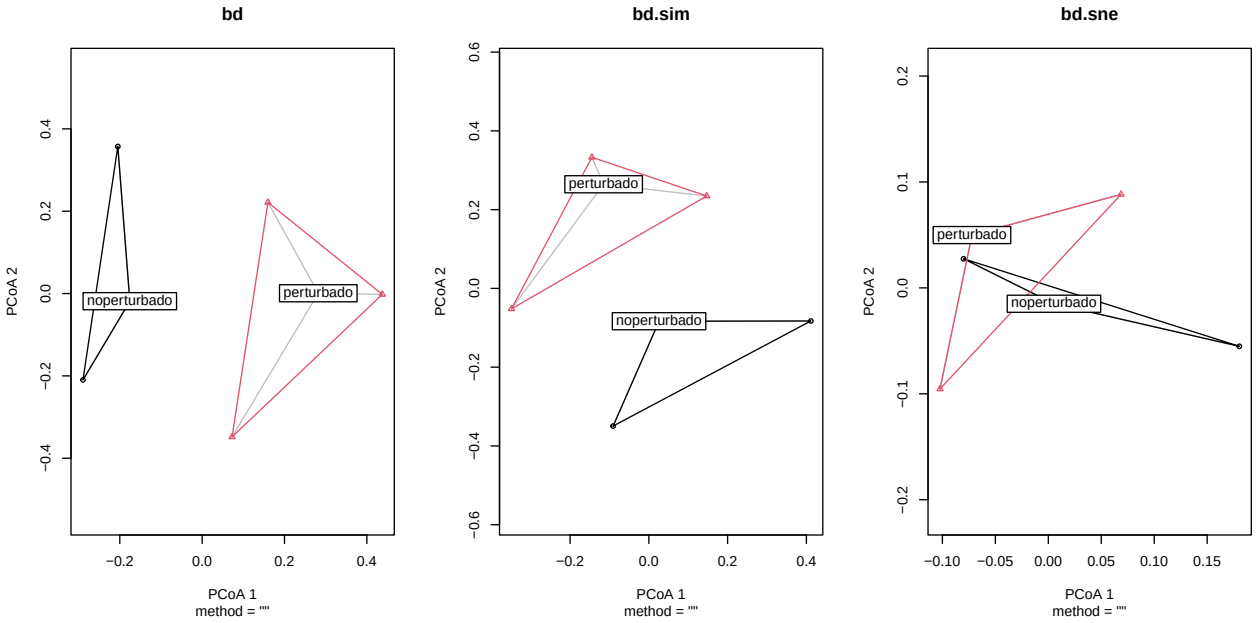
Para obtener la partición del índice de recambio de Sorensen por pares entre comunidades, escriba: dist_comp$beta.sim. Para obtener la partición de anidamiento, escriba: dist_comp$beta.sne Para obtener toda la diversidad beta: dist_comp$beta.sor.

Ya obtenidas estas matrices podemos aplicar cualquier análisis multivariado. En el presente ejemplo, si queremos comparar las diversidades beta de las comunidades agregadas por los tratamientos de "no perturbado" y "perturbado", podemos utilizar el análisis "betadisper" del paquete 'vegan':

```
library(vegan)
bd<-betadisper(dist_comp$beta.sor,groups)
bd.sim<-betadisper(dist_comp$beta.sim,groups)
bd.sne<-betadisper(dist_comp$beta.sne,groups)



par(mfrow=c(1,3))
plot(bd);plot(bd.sim);plot(bd.sne)
```



El gráfico de dispersión beta indica que hay una diferencia en la composición de especies de los fragmentos de bosque perturbado y no perturbado. Con estas matrices de distancia luego también podemos aplicar las diversidad técnicas multivariadas que vimos en el capítulo pasado como perMANOVA, cluster y los diferentes métodos de ordeniación para corroborar estas diferencias en composición.

Si queremos obtener estos mismos resultados pero para todos los sitios, entonces usamos la función **beta.multi()**:

```
dist.multi<-beta.multi(presabs,index.family ="sorensen" )
head(dist.multi)
```

```
## $beta.SIM
## [1] 0.4871795
##
## $beta.SNE
## [1] 0.1206637
##
## $beta.SOR
## [1] 0.6078431
```

Ejemplo aplicado

Con la misma data trabajada anteriormente vamos a verificar si dan resultados similares utilizando el índice de Jaccard.

vegan

Esta paquetería es una excelente herramienta para analizar datos ecológicos. En el caso de alpha diversidad se pueden calcular los índices tradicionales (que no están en el marco de los números de Hill).

Aquí realizaremos un ejemplo con una data que trae de ejemplo que se llama BCI (Barro Colorado Island Tree Counts).

Primero llamamos la data:

```
data(BCI, BCI.env)
```

Y podemos calcular los índices de diversidad alpha, como Shannon, Simpson, el inverso de Simpson, Fisher, Especies observadas, beta y gamma:

```
shannon <- diversity(BCI)
simpson <- diversity(BCI, "simpson")
inverso_simp <- diversity(BCI, "inv")
fisher <- fisher.alpha(BCI)
especies_num <- specnumber(BCI)

## La diversidad beta definida como gamma/alpha - 1:
## teniendo en cuenta el número total de especies
(alpha <- with(BCI.env, tapply(specnumber(BCI), Habitat, mean)))
```

```
##  OldHigh   OldLow OldSlope    Swamp    Young
## 85.75000 91.76923 91.58333 94.00000 90.00000
```

```
(gamma <- with(BCI.env, specnumber(BCI, Habitat)))
```

```
##  OldHigh   OldLow OldSlope    Swamp    Young
##      158      210      183      128      117
```

```
gamma/alpha - 1
```

```
##   OldHigh    OldLow  OldSlope     Swamp     Young
## 0.8425656 1.2883487 0.9981802 0.3617021 0.3000000
```

```
## de manela similar pero con la diversidad de Shannon
(alpha <- with(BCI.env, tapply(diversity(BCI), Habitat, mean))) # promedio
```

```
##  OldHigh   OldLow OldSlope    Swamp    Young
## 3.638598 3.876413 3.887122 4.003780 3.246729
```

```
(gamma <- with(BCI.env, diversity(BCI, groups=Habitat))) # junta
```

```
##  OldHigh   OldLow OldSlope    Swamp    Young
## 3.873186 4.284972 4.212098 4.164335 3.387536
```

```
## aditiva con la diversidad de Shannon
gamma-alpha
```

```
##   OldHigh    OldLow  OldSlope     Swamp     Young
## 0.2345878 0.4085595 0.3249760 0.1605548 0.1408068
```

Para la beta divesidad podemos calcular nuestras matrices de distancia con la función 'vegdist':

```
library(tidyverse)
distancias<- vegdist(BCI, method = "jaccard")
distancias %>% as.matrix() %>% as.data.frame() %>% dplyr::select(1:4) %>% dplyr::slice(1:4)
```

```
##           1         2         3         4
```

```
## 1 0.0000000 0.4260250 0.5186992 0.5382263
## 2 0.4260250 0.0000000 0.4463668 0.4790323
## 3 0.5186992 0.4463668 0.0000000 0.4898911
## 4 0.5382263 0.4790323 0.4898911 0.0000000
```

Parte 2

iNEXT

iNEXT (iNterpolation and EXTrapolation) [2]

Es un paquete disponible en R para rarefacción y extrapolación de la diversidad de especies en el marco de los números de Hill. Véase (Chao and Jost 2012) y (Chao et al. 2014) para metodologías.También está disponible una versión en línea de iNEXT Online para usuarios sin experiencia en R.

iNEXT se centra en tres medidas de los números de Hill de orden q: riqueza de especies (q=0), diversidad de Shannon (q=1, la exponencial de la entropía de Shannon) y diversidad de Simpson (q=2, la inversa de la concentración de Simpson).

Para cada medida de diversidad, iNEXT utiliza la muestra observada de datos de abundancia o incidencia para calcular las estimaciones de diversidad para muestras enrarecidas y extrapoladas y los intervalos de confianza del 95 % (predeterminados) asociados, además de representar gráficamente los dos tipos siguientes de las curvas de rarefacción y extrapolación (R/E):

- Curvas de muestreo R/E basadas en el tamaño de la muestra: iNEXT calcula estimaciones de diversidad para muestras enrarecidas y extrapoladas hasta el doble del tamaño de la muestra de referencia (por defecto) o un tamaño especificado por el usuario. Este tipo de curva de muestreo traza las estimaciones de diversidad con respecto al tamaño de la muestra. El tamaño de la muestra se refiere al número de individuos en una muestra para datos de abundancia, mientras que se refiere al número de unidades de muestreo para datos de incidencia.
- Curvas de muestreo R/E basadas en la cobertura: iNEXT calcula estimaciones de diversidad para muestras enrarecidas y extrapoladas con integridad de la muestra (medida por la cobertura de la muestra) hasta el valor de cobertura del doble del tamaño de la muestra de referencia (por defecto) o una cobertura especificada por el usuario. Este tipo de curva de muestreo traza las estimaciones de diversidad con respecto a la cobertura de la muestra. Además de los dos tipos anteriores de curvas de muestreo, iNEXT también traza una curva de completitud de la muestra, que describe cómo varía la estimación de la cobertura de la muestra en función del tamaño de la muestra. La curva de integridad de la muestra se puede considerar como un puente que conecta los dos tipos de curvas mencionados anteriormente.

Para instalar este paquete:

```
## instalando iNEXT del CRAN
install.packages("iNEXT")

## instalando la versión de desarrollo
install.packages('devtools')
library(devtools)
install_github('AnneChao/iNEXT')
```

```
## cargando el paquete
library(iNEXT)
library(ggplot2)
```

[2] https://github.com/JohnsonHsieh/iNEXT

La función principal es:

iNEXT(x, q=0, datatype="abundance", se=TRUE, conf=0.95, nboot=50)

Donde:

- x : es la data,
- datatype: puede ser "abundance" ó "incidence_raw", ó "incidence_freq",
- se: TRUE o FALSE si se quiere hacer un muestreo tipo 'bootsrap',
- conf: el intervalo de confianza,
- nboot: número de replicaciones de 'bootsrap'

Correremos el ejemplo con la data del paquete, denominada **spider**:

```
data("spider")
str(spider)
```

```
## List of 2
##  $ Girdled: num [1:26] 46 22 17 15 15 9 8 6 6 4 ...
##  $ Logged : num [1:37] 88 22 16 15 13 10 8 8 7 7 ...
```

```
out <- iNEXT(spider, q=c(0, 1, 2), datatype="abundance")
```

Si vemos el output de iNEXT nos da una lista con tres elementos:

- $DataInfo que nos resume la información de la data
- $iNextEst los estimados para las muestras rarificadas y extrapoladas (datos para las curvas)
- $AsyEst muestra la diversidad estimada

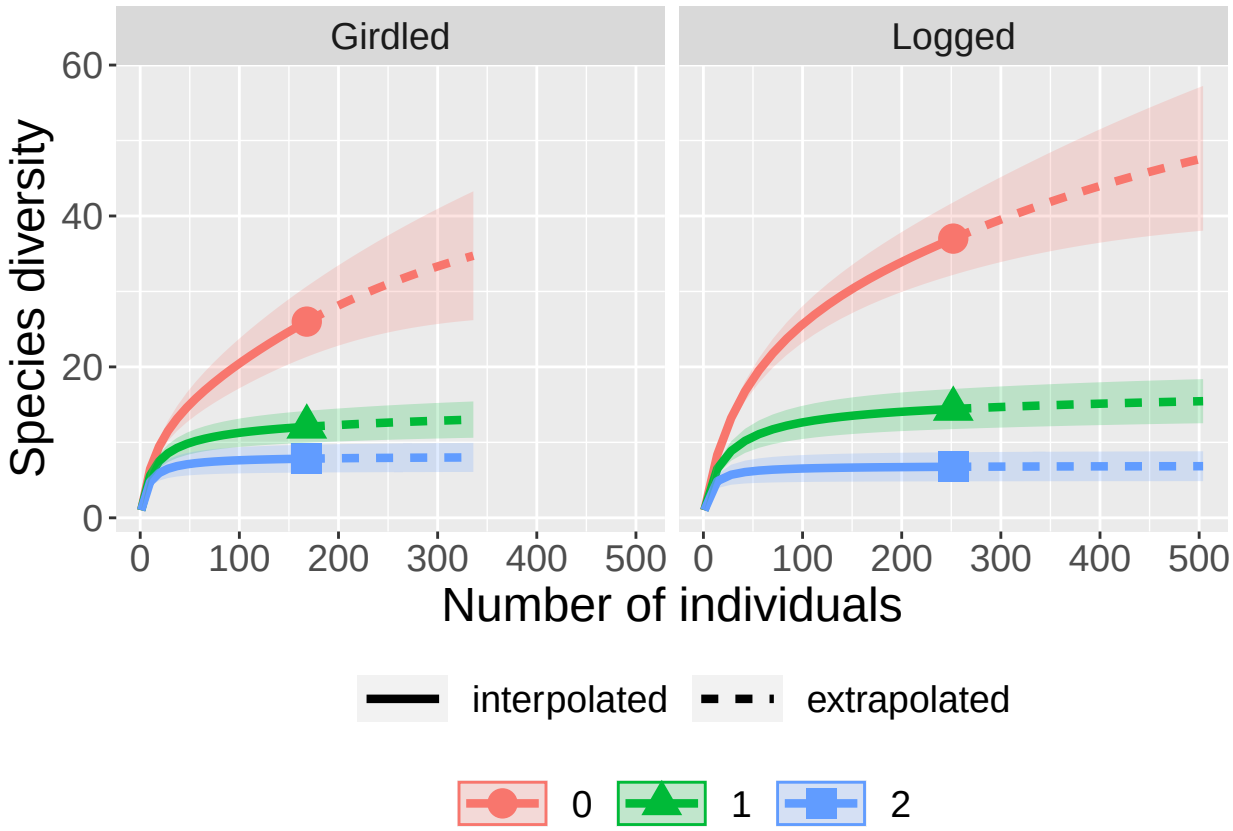
```
out
```

```
## Compare 2 assemblages with Hill number order q = 0, 1, 2.
## $class: iNEXT
##
## $DataInfo: basic data information
##      site   n S.obs     SC f1 f2 f3 f4 f5 f6 f7 f8 f9 f10
## 1 Girdled 168    26 0.9289 12  4  0  1  0  2  0  1  1   0
## 2  Logged 252    37 0.9446 14  4  4  3  1  0  3  2  0   1
##
## $iNextEst: diversity estimates with rarefied and extrapolated samples.
## $Girdled
##       m       method order     qD qD.LCL qD.UCL    SC SC.LCL SC.UCL
## 1     1 interpolated     0  1.000  1.000  1.000 0.122  0.092  0.153
## 10   84 interpolated     0 18.912 15.976 21.848 0.900  0.873  0.926
## 20  168     observed     0 26.000 21.331 30.669 0.929  0.902  0.956
## 30  248 extrapolated     0 30.883 24.501 37.265 0.948  0.917  0.980
## 40  336 extrapolated     0 34.731 26.195 43.267 0.964  0.932  0.995
## 41    1 interpolated     1  1.000  1.000  1.000 0.122  0.092  0.153
## 50   84 interpolated     1 10.964  9.170 12.758 0.900  0.873  0.926
## 60  168     observed     1 12.060  9.947 14.173 0.929  0.899  0.958
## 70  248 extrapolated     1 12.606 10.329 14.884 0.948  0.914  0.982
## 80  336 extrapolated     1 13.014 10.609 15.420 0.964  0.930  0.998
## 81    1 interpolated     2  1.000  1.000  1.000 0.122  0.091  0.154
```

```
## 90   84 interpolated     2  7.532  5.841  9.222 0.900  0.868  0.931
## 100 168     observed     2  7.840  5.997  9.683 0.929  0.897  0.960
## 110 248 extrapolated     2  7.945  6.049  9.841 0.948  0.912  0.985
## 120 336 extrapolated     2  8.004  6.077  9.931 0.964  0.927  1.000
##
## $Logged
##       m       method order     qD qD.LCL qD.UCL    SC SC.LCL SC.UCL
## 1     1 interpolated     0  1.000  1.000  1.000 0.145  0.116  0.173
## 10  126 interpolated     0 28.268 25.439 31.097 0.908  0.888  0.928
## 20  252     observed     0 37.000 32.195 41.805 0.945  0.922  0.967
## 30  371 extrapolated     0 42.786 35.837 49.734 0.958  0.933  0.982
## 40  504 extrapolated     0 47.644 38.051 57.238 0.969  0.945  0.993
## 41    1 interpolated     1  1.000  1.000  1.000 0.145  0.108  0.181
## 50  126 interpolated     1 13.172 10.837 15.507 0.908  0.887  0.930
## 60  252     observed     1 14.421 11.762 17.080 0.945  0.924  0.966
## 70  371 extrapolated     1 15.010 12.201 17.820 0.958  0.933  0.983
## 80  504 extrapolated     1 15.457 12.534 18.381 0.969  0.944  0.994
## 81    1 interpolated     2  1.000  1.000  1.000 0.145  0.101  0.188
## 90  126 interpolated     2  6.610  4.777  8.443 0.908  0.886  0.930
## 100 252     observed     2  6.761  4.834  8.689 0.945  0.921  0.968
## 110 371 extrapolated     2  6.812  4.852  8.771 0.958  0.931  0.985
## 120 504 extrapolated     2  6.840  4.863  8.817 0.969  0.942  0.996
##
##
## $AsyEst: asymptotic diversity estimates along with related statistics.
##      Site         Diversity Observed Estimator   s.e.    LCL     UCL
## 1 Girdled  Species richness   26.000    43.893 14.306 30.511  96.971
## 2 Girdled Shannon diversity   12.060    13.826  1.467 12.060  16.702
## 3 Girdled Simpson diversity    7.840     8.175  0.983  7.840  10.102
## 4  Logged  Species richness   37.000    61.403 18.532 43.502 128.583
## 5  Logged Shannon diversity   14.421    16.337  1.523 14.421  19.322
## 6  Logged Simpson diversity    6.761     6.920  0.921  6.761   8.726
##
## NOTE: Only show five estimates, call iNEXT.object$iNextEst. to show complete output.
```

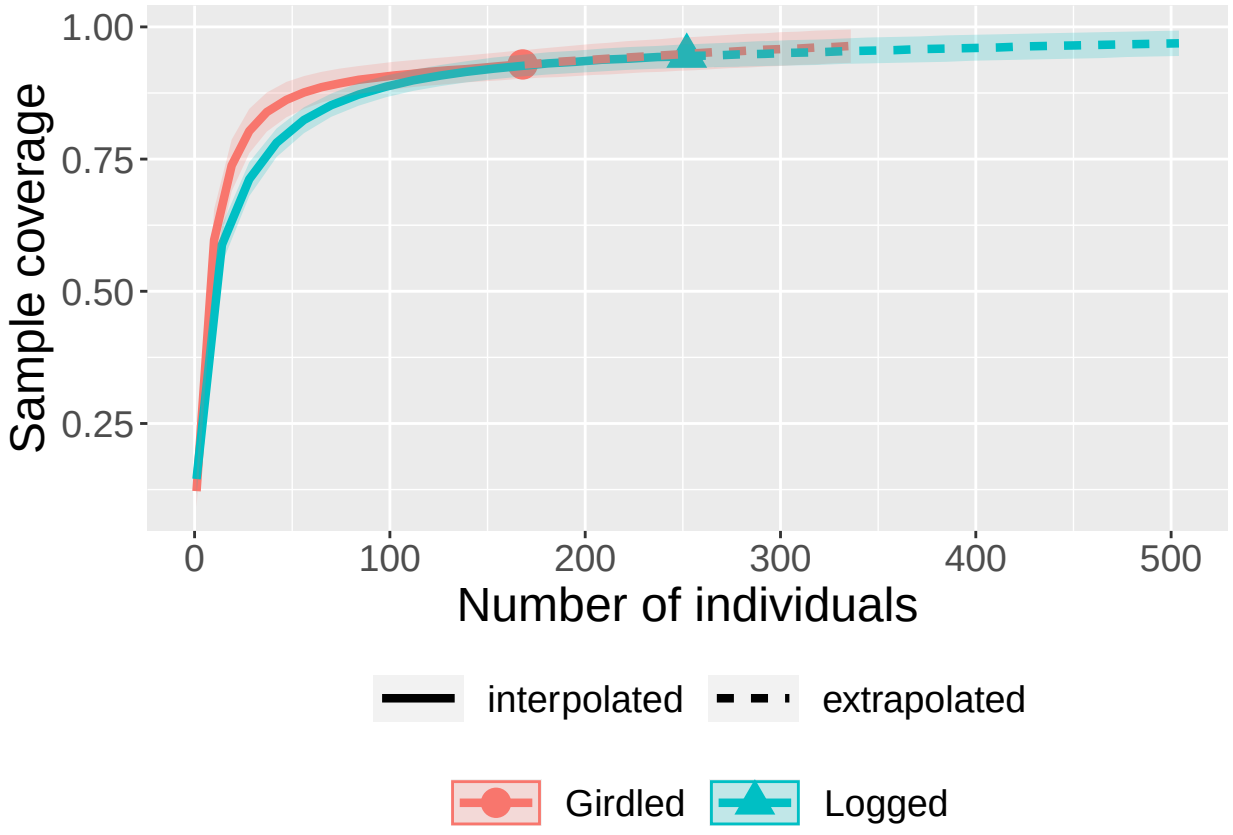
- Podemos obtener ahora nuestras curvas por sitio o en este caso por tratamiento a los 3 órdenes:

```
ggiNEXT(out, type=1, facet.var="site")
```


Girdled
Logged
Species diversity
60
40
20
0
0
100
200
300
400
500
Number of individuals
interpolated
extrapolated
0
1
2

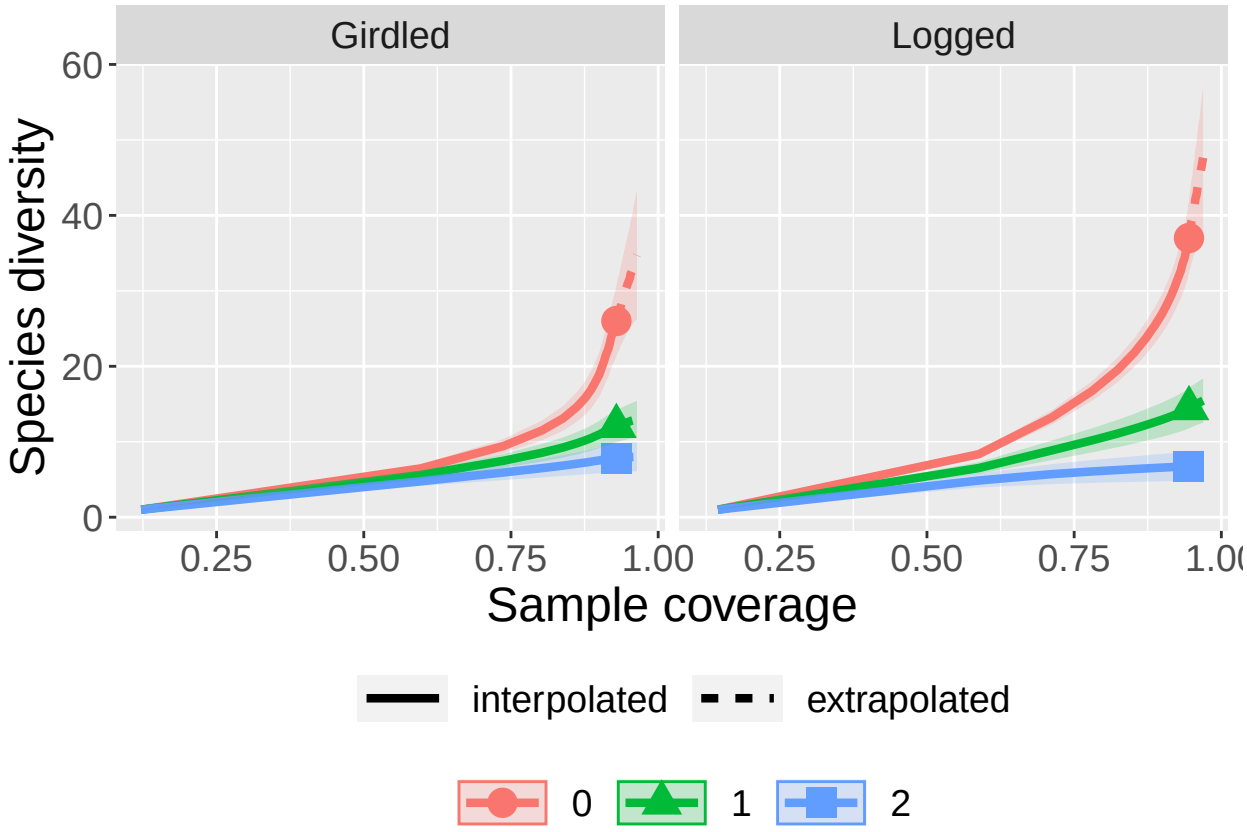
- También podemos obtener ahora nuestras curvas de cobertura:

```
ggiNEXT(out, type=2)
```



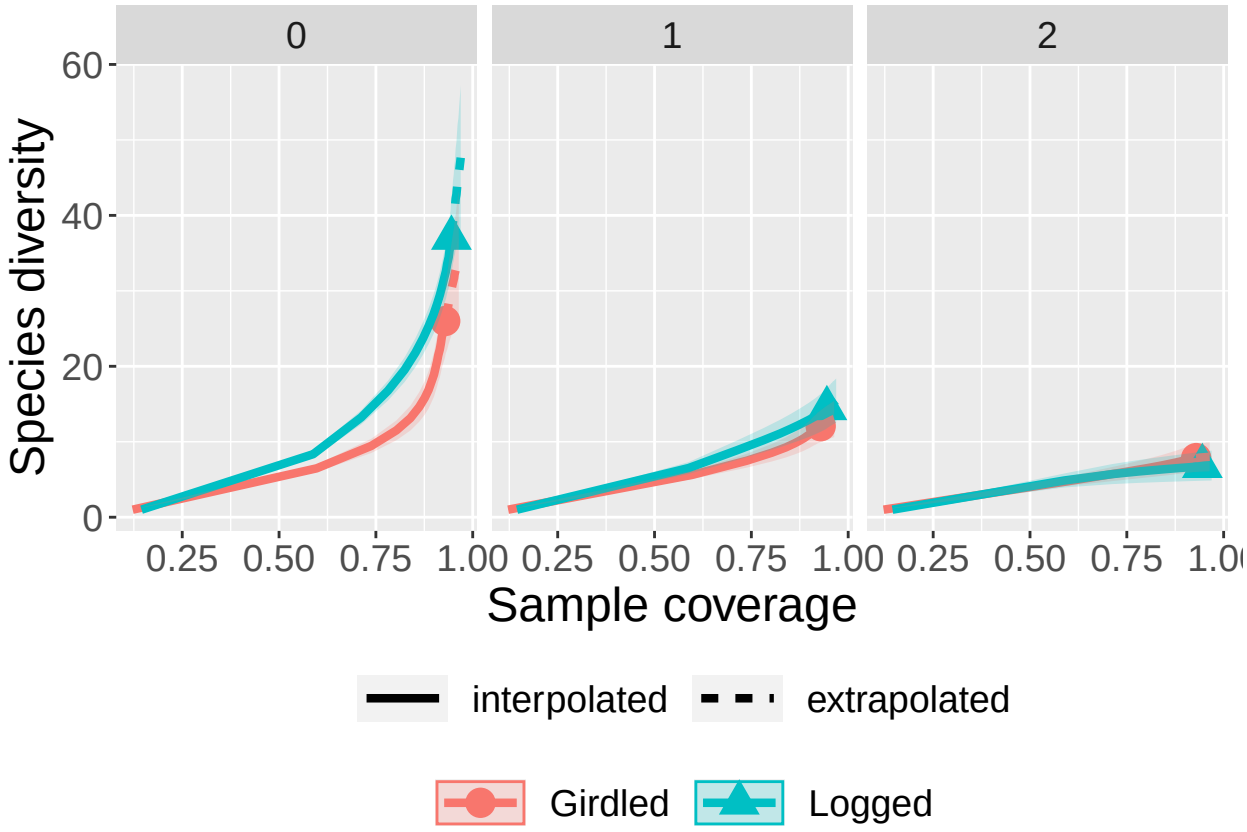
- Por último una gráfica de diversidad de especies vs cobertura para los tratamientos:

```
ggiNEXT(out, type=3, facet.var="site")
```



- Y a los diferentes órdenes:

```
ggiNEXT(out, type=3, facet.var="order")
```



Ahora si se quiere sólo los índices basandonos ya sea en abundancia o incidencia y en tamaño o cobertura, usamos la función **estimateD**, así:

```
estimateD(spider$Logged, datatype="abundance", base="size",  conf=0.95)
```

```
##     m   method order    SC     qD qD.LCL qD.UCL
## 1 252 observed     0 0.945 37.000 31.322 42.678
## 2 252 observed     1 0.945 14.421 11.757 17.085
## 3 252 observed     2 0.945  6.761  5.041  8.482
```

```
estimateD(spider$Girdled, datatype="abundance", base="size",  conf=0.95)
```

```
##     m   method order    SC    qD qD.LCL qD.UCL
## 1 168 observed     0 0.929 26.00 20.338 31.662
## 2 168 observed     1 0.929 12.06  9.927 14.192
## 3 168 observed     2 0.929  7.84  6.110  9.570
```

Ejemplo aplicado

Apliquemos lo visto usando como ejemplo pequeño con datos de incidencia (presencia/ausencia):

```
data(ciliates)
#str(ciliates)
```

Este paquete también cuenta con otras funciones de interés, tales como:

- ChaoEntropy() : Estimación de la entropía/diversidad de Shannon
- ChaoRichness(): Estimación de la riqueza de especies
- ChaoShannon(): Estimación de la entropía/diversidad de Shannon
- ChaoSimpson(): Estimación del índice de Gini-Simpson o diversidad de Simpson
- ChaoSpecies(): Estimación de la riqueza de especies
- EstSimpson: Estimación del índice de Gini-Simpson o diversidad de Simpson

Un ejemplo:

```
ChaoRichness(spider)
```

```
##         Observed Estimator Est_s.e. 95% Lower 95% Upper
## Girdled       26    43.893   14.306    30.511    96.971
## Logged        37    61.403   18.532    43.502   128.583
```

```
ChaoShannon(spider)
```

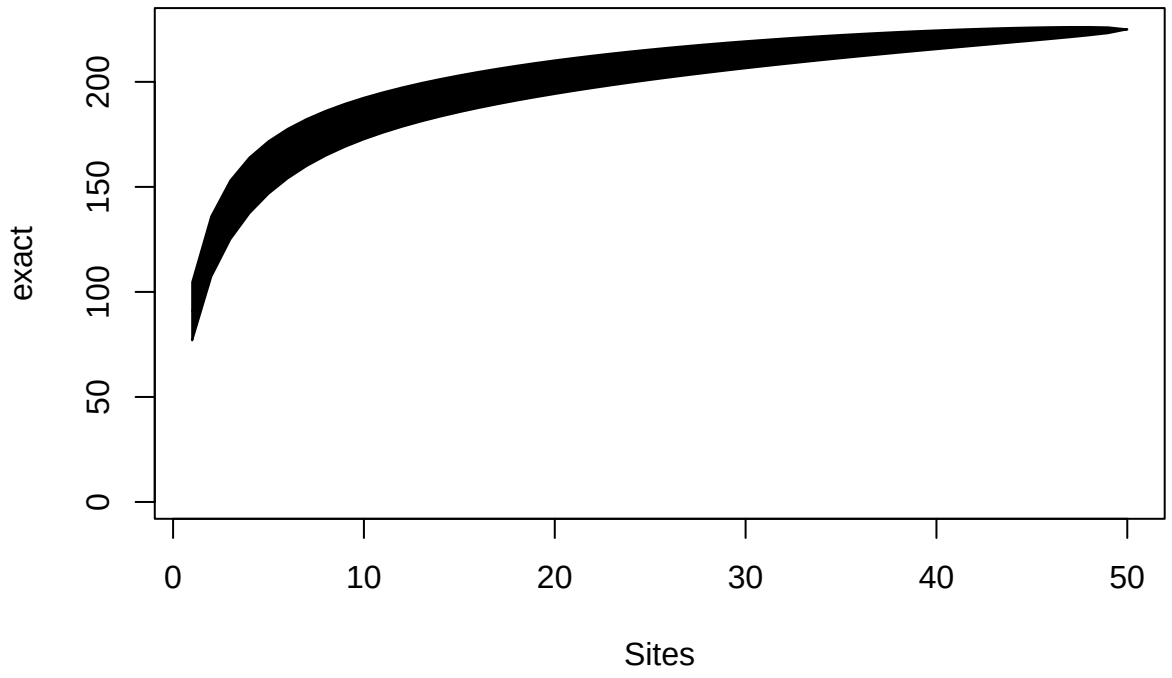
```
##         Observed Estimator Est_s.e 95% Lower 95% Upper
## Girdled    2.490     2.627   0.110     2.490     2.842
## Logged     2.669     2.793   0.099     2.669     2.988
```

Curvas en vegan

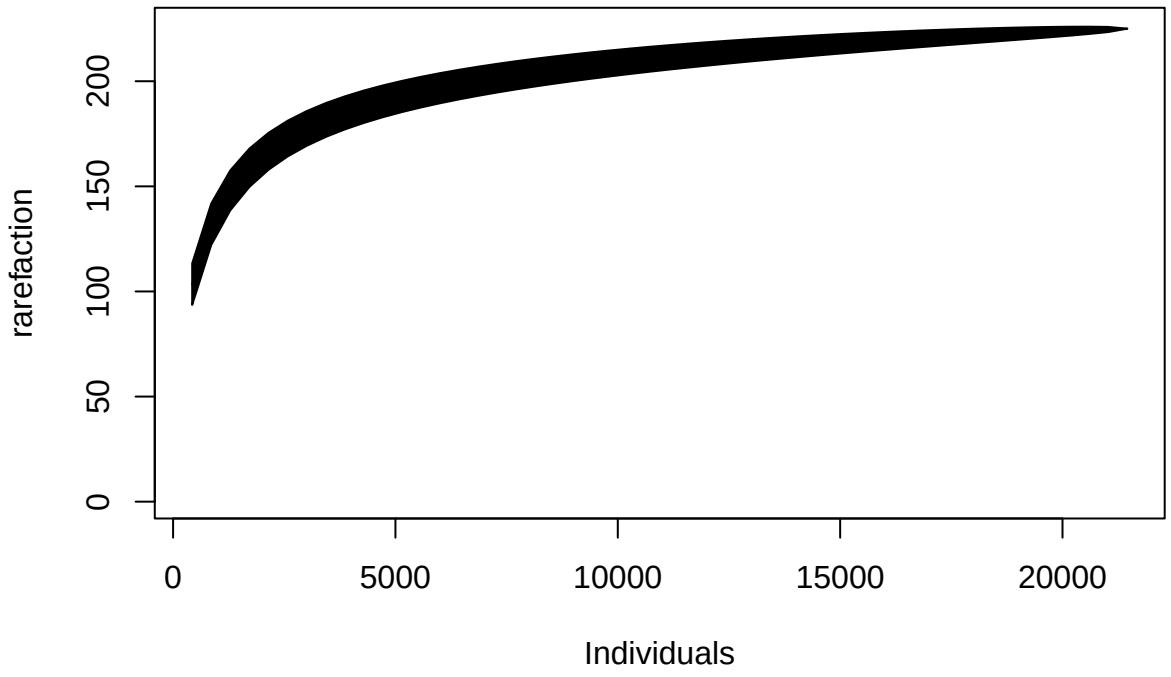
```
library(vegan)
data("BCI")
head(BCI)[1:4,1:4]
```

```
##   Abarema.macradenia Vachellia.melanoceras Acalypha.diversifolia
## 1                  0                     0                     0
## 2                  0                     0                     0
## 3                  0                     0                     0
## 4                  0                     0                     0
##   Acalypha.macrostachya
## 1                     0
## 2                     0
## 3                     0
## 4                     0
```

```
#por sitios
sac <- specaccum(BCI)
plot(sac, ci.type="polygon") #ver vegan para opciones
```




```
#por individuos
sac <- specaccum(BCI, method = "rarefaction")
plot(sac, xvar = "individual", ci.type="polygon")
```


rarefaction
0
50
100
150
200
0
5000
10000
15000
20000
Individuals

REFERENCIAS

Baselga, Andrés, and C. David L. Orme. 2012. "Betapart : An R Package for the Study of Beta Diversity." *Methods in Ecology and Evolution* 3 (5): 808–12. https://doi.org/10.1111/j.2041-210x.2012.00224.x.

Chao, Anne, Chun-Huo Chiu, and Lou Jost. 2014. "Unifying Species Diversity, Phylogenetic Diversity, Functional Diversity, and Related Similarity and Differentiation Measures Through Hill Numbers." *Annual Review of Ecology, Evolution, and Systematics* 45 (1): 297–324. https://doi.org/10.1146/annurev-ecolsys-120213-091540.

Chao, Anne, Nicholas J. Gotelli, T. C. Hsieh, Elizabeth L. Sander, K. H. Ma, Robert K. Colwell, and Aaron M. Ellison. 2014. "Rarefaction and Extrapolation with Hill Numbers: A Framework for Sampling and Estimation in Species Diversity Studies." *Ecological Monographs* 84 (1): 45–67. https://doi.org/10.1890/13-0133.1.

Chao, Anne, and Lou Jost. 2012. "Coverage-Based Rarefaction and Extrapolation: Standardizing Samples by Completeness Rather Than Size." *Ecology* 93 (12): 2533–47. https://doi.org/10.1890/11-1952.1.

Chiu, Chun-Huo, and Anne Chao. 2014. "Distance-Based Functional Diversity Measures and Their Decomposition: A Framework Based on Hill Numbers." Edited by Francesco de Bello. *PLoS ONE* 9 (7): e100014. https://doi.org/10.1371/journal.pone.0100014.

Sackett, Tara E., Sydne Record, Sharon Bewick, Benjamin Baiser, Nathan J. Sanders, and Aaron M. Ellison. 2011. "Response of Macroarthropod Assemblages to the Loss of Hemlock (*Tsuga Canadensis*), a Foundation Species." *Ecosphere* 2 (7): art74. https://doi.org/10.1890/es11-00155.1.