5. Worksheet: Alpha Diversity

Trang Nguyen; Z620: Quantitative Biodiversity, Indiana University

29 janvier, 2025

OVERVIEW

In this exercise, we will explore aspects of local or site-specific diversity, also known as alpha (α) diversity. First we will quantify two of the fundamental components of (α) diversity: **richness** and **evenness**. From there, we will then discuss ways to integrate richness and evenness, which will include univariate metrics of diversity along with an investigation of the **species abundance distribution (SAD)**.

Directions:

1. In the Markdown version of this document in your cloned repo, change "Student Name" on line 3 (above) to your name.
2. Complete as much of the worksheet as possible during class.
3. Use the handout as a guide; it contains a more complete description of data sets along with the proper scripting needed to carry out the exercise.
4. Answer questions in the worksheet. Space for your answer is provided in this document and indicated by the ">" character. If you need a second paragraph be sure to start the first line with ">". You should notice that the answer is highlighted in green by RStudio (color may vary if you changed the editor theme).
5. Before you leave the classroom, **push** this file to your GitHub repo.
6. For the assignment portion of the worksheet, follow the directions at the bottom of this file.
7. When you are done, **Knit** the text and code into a PDF file.
8. After Knitting, submit the completed exercise by creating a **pull request** via GitHub. Your pull request should include this file `AlphaDiversity_Worskheet.Rmd` and the PDF output of `Knitr` (`AlphaDiversity_Worskheet.pdf`).

1) R SETUP

In the R code chunk below, please provide the code to: 1) Clear your R environment, 2) Print your current working directory, 3) Set your working directory to your `Week-2/` folder folder, and 4) Load the `vegan` R package (be sure to install first if you have not already).

```
rm(list = ls())
print(getwd())
```

```
## [1] "C:/Users/ttran/OneDrive - Indiana University/SP25 - Quantitative Biodiversity/QB2025_Nguyen/Weel
```

```
setwd(getwd())
```

2) LOADING DATA

In the R code chunk below, do the following: 1) Load the BCI dataset, and 2) Display the structure of the dataset (if the structure is long, use the `max.level = 0` argument to show the basic information).

```
# install.packages("vegan")
require(vegan)
```

```
## Le chargement a nécessité le package : vegan
```

```
## Warning: le package 'vegan' a été compilé avec la version R 4.4.2
```

```
## Le chargement a nécessité le package : permute
```

```
## Warning: le package 'permute' a été compilé avec la version R 4.4.2
```

```
## Le chargement a nécessité le package : lattice
```

```
## This is vegan 2.6-8
```

```
data("BCI")
dim(BCI)
```

```
## [1]  50 225
```

```
# Structure of dataset
str(BCI, max.level = 0)
```

```
## 'data.frame':    50 obs. of  225 variables:
##  - attr(*, "original.names")= chr [1:225] "Abarema.macradenium" "Acacia.melanoceras" "Acalypha.divers
```

3) SPECIES RICHNESS

Species richness (S) refers to the number of species in a system or the number of species observed in a sample.

Observed richness

In the R code chunk below, do the following:

1. Write a function called `S.obs` to calculate observed richness
2. Use your function to determine the number of species in `site1` of the BCI data set, and
3. Compare the output of your function to the output of the `specnumber()` function in `vegan`.

```r
# function S observed
S.obs = function(x = "" ){
  rowSums(x > 0) * 1
  }

# Check the S_obs for site1 in BCI
S.obs(BCI[1,])
```

```
##  1
## 93
```

```r
# Compare output
specnumber(BCI[1,])
```

```
##  1
## 93
```

```r
# Comparison
specnumber(BCI[1,]) == S.obs(BCI[1,])
```

```
##    1
## TRUE
```

```r
# Question 1 answer
specnumber(BCI[1:4,])
```

```
##  1  2  3  4
## 93 84 90 94
```

Question 1: Does `specnumber()` from `vegan` return the same value for observed richness in `site1` as our function `S.obs`? What is the species richness of the first four sites (i.e., rows) of the BCI matrix?

> ***Answer 1***: Yes it returned the same value for the observed richness in site1 as our function. The species richness of the first four sites are 93, 84, 90, 94

Coverage: How well did you sample your site?

In the R code chunk below, do the following:

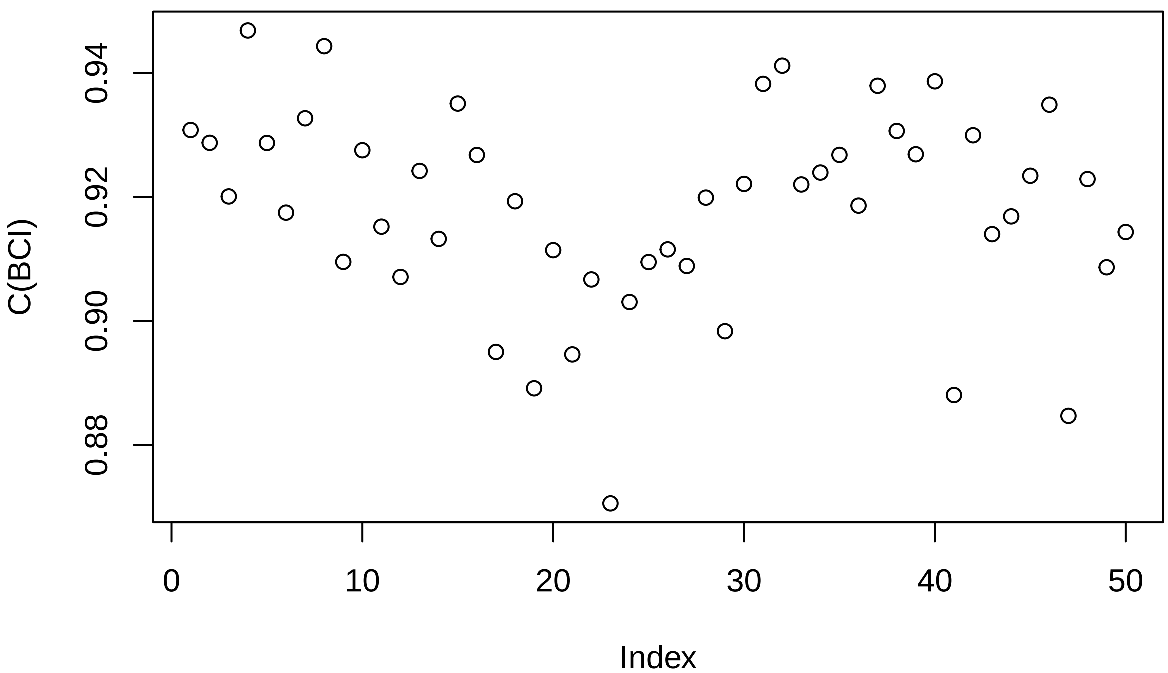
1. Write a function to calculate Good's Coverage, and
2. Use that function to calculate coverage for all sites in the BCI matrix.

```r
# Good's Coverage
C = function(x = ""){
    1 - (rowSums(x == 1) / rowSums(x))
}

# Calculate coverage for all sites
C(BCI)
```

```
##         1         2         3         4         5         6         7         8
## 0.9308036 0.9287356 0.9200864 0.9468504 0.9287129 0.9174757 0.9326923 0.9443155
##         9        10        11        12        13        14        15        16
## 0.9095355 0.9275362 0.9152120 0.9071038 0.9242054 0.9132420 0.9350649 0.9267735
##        17        18        19        20        21        22        23        24
## 0.8950131 0.9193084 0.8891455 0.9114219 0.8946078 0.9066986 0.8705882 0.9030612
##        25        26        27        28        29        30        31        32
## 0.9095023 0.9115479 0.9088729 0.9198966 0.8983516 0.9221053 0.9382423 0.9411765
##        33        34        35        36        37        38        39        40
## 0.9220183 0.9239374 0.9267887 0.9186047 0.9379310 0.9306488 0.9268868 0.9386503
##        41        42        43        44        45        46        47        48
## 0.8880597 0.9299517 0.9140049 0.9168704 0.9234234 0.9348837 0.8847059 0.9228916
##        49        50
## 0.9086651 0.9143519
```

```
plot(C(BCI))
```



Question 2: Answer the following questions about coverage:

a. What is the range of values that can be generated by Good's Coverage?
b. What would we conclude from Good's Coverage if n_i equaled N?
c. What portion of taxa in `site1` was represented by singletons?
d. Make some observations about coverage at the BCI plots.

> ***Answer 2a***: The range of values that can be generated by Good's Coverage is from 0 to 1. 1 means that there are many species that are not singletons, meaning good coverage, while 0 means poor coverage.

Answer 2b: We would have C = 0

Answer 2c: In site1, we got 7% of the total number of individuals in the sample that is represented by singletons.

Answer 2d: I think most sites have quite good coverage (close to 90%). The lowest coverage is 0.87, which is still quite good.

Estimated richness

In the R code chunk below, do the following:

1. Load the microbial dataset (located in the `Week-2/data` folder),
2. Transform and transpose the data as needed (see handout),
3. Create a new vector (`soilbac1`) by indexing the bacterial OTU abundances of any site in the dataset,
4. Calculate the observed richness at that particular site, and
5. Calculate coverage of that site

```
# Week2-Alpha\5.AlphaDiversity_Worksheet.Rmd
# 1. Load the microbial dataset (located in the `Week-2/data` folder),

soilbac = read.table("./data/soilbac.txt", sep="\t", header=TRUE, row.names=1)

# 2. Transform and transpose the data as needed (see handout),
soilbac.t = as.data.frame(t(soilbac))

# 3. Create a new vector (`soilbac1`) by indexing the bacterial OTU abundances of any site in the datas
soilbac1 = soilbac.t[1,]

# 4. Calculate the observed richness at that particular site, and
S_obs_soilbac1 = S.obs(soilbac1)
S_obs_soilbac1
```

```
## T1_1
## 1074
```

```
# 5. Calculate coverage of that site
C_soilbac1 = C(soilbac1) # nolint: object_name_linter.
C_soilbac1
```

```
##      T1_1
## 0.6479471
```

Question 3: Answer the following questions about the soil bacterial dataset.

a. How many sequences did we recover from the sample `soilbac1`, i.e. *N*?
b. What is the observed richness of `soilbac1`?
c. How does coverage compare between the BCI sample (`site1`) and the KBS sample (`soilbac1`)?

```r
# a.  How many sequences did we recover from the sample `soilbac1`, i.e. *N*?
sum(soilbac1)
```

```
## [1] 2119
```

Answer 3a: We recovered 2119 sequences for sample soilbac1

Answer 3b: The observed richness for the sample soilbac1 is 1074

Answer 3c: The coverage of soilbac1 is 0.648, much lower than the coverage of site1 in BIC.

Richness estimators

In the R code chunk below, do the following:

1. Write a function to calculate **Chao1**,
2. Write a function to calculate **Chao2**,
3. Write a function to calculate **ACE**, and
4. Use these functions to estimate richness at `site1` and `soilbac1`.

```r
# 1. Write a function to calculate **Chao1**,
S.chao1 = function(x = ""){
    S.obs(x) + (sum(x == 1)^ 2) / (2 * sum(x == 2))
}

# 2. Write a function to calculate **Chao2**,
S.chao2 = function(site="", SbyS=""){
    SbyS = as.data.frame(SbyS)
    x = SbyS[site,]
    SbyS.pa = (SbyS > 0) * 1 # convert SbyS matrix in to presence/absence matrix
    Q1 = sum(colSums(SbyS.pa) == 1) # species observed once
    Q2 = sum(colSums(SbyS.pa) == 2) # species observed twice
    S.Chao2 = S.obs(x) + (Q1^2) / (2 * Q2)
    return(S.Chao2)

}
# 3. Write a function to calculate **ACE**, and
S.ace = function(x = "", thresh = 10){
    x = x[x>0]                           # exclude taxas that have 0  presence
    S.abund = length(which(x > thresh)) # richness of abundant species
    S.rare = length(which(x <= thresh)) # richness of rare species
    singletons = length(which(x == 1))  # number of singletons
    N.rare = sum(x[x <= thresh])        # total number of rare species ( abundance )
    C.ace = 1 - singletons / N.rare     # coverage of species that occur more than once
    i = c(1:length(x))

    # function that calculates the number of species that occur at i times
    count = function(i, y){
        length(y[y == i])
```

```r
    }
    a.1 = sapply(i, count, x)           # number of individuals in richness i richness classes
    f.1 = (i * ( i - 1)) * a.1          # k(k-1) * f(k) sensu Gotelli
    G.ace = (S.rare / C.ace ) * (sum(f.1) / (N.rare * (N.rare - 1))) #
    S.ace_res = S.abund + (S.rare / C.ace) + (singletons / C.ace) * max(G.ace, 0)
    return(S.ace_res)

}
# 4. Use these functions to estimate richness at `site1` and `soilbac1`.
S.chao1(BCI[1,])
```

```
##        1
## 119.6944
```

```r
S.chao2(1, BCI)
```

```
##        1
## 104.6053
```

```r
S.ace(BCI[1,])
```

```
## [1] 475.2733
```

```r
S.chao1(soilbac1)
```

```
##     T1_1
## 2628.514
```

```r
S.chao2(1, soilbac.t)
```

```
##     T1_1
## 21055.39
```

```r
S.ace(soilbac1)
```

```
## [1] 22079.39
```

```r
# dim(soilbac.t)
```

Question 4: What is the difference between ACE and the Chao estimators? Do the estimators give consistent results? Which one would you choose to use and why?

> ***Answer 4***: **Do the estimators give consistent results?** The estimators give consistent results for the soilbac dataset, but not for the BIC dataset.
>
> **Which one would you choose to use and why?** Before answering, let's take several points into consideration: The BIC dataset consists of 50 sites and 225 species, while the dataset of soilbac.t has 11 sites and 13310 species. The Chao1 estimator uses abundance data, focusing on singletons and doubletons within the site to estimate the richness. The Chao2 estimator

uses incidence data, focusing on singletons and doubletons accross all the sites to estimate the richness. In a site where there are many more singletons compared to doubletons within the site, than accross the sites, the Chao1 estimator will yield higher results than Chao2 estimator.

The ACE estimator uses abundance data, focusing on the number of rare species and the number of singletons to estimate the richness. I think ACE is better suited when there is a large number of species and the dataset includes many rare species with varying abundances rather than singletons and doubletons.

Rarefaction

In the R code chunk below, please do the following:

1. Calculate observed richness for all samples in `soilbac`,
2. Determine the size of the smallest sample,
3. Use the `rarefy()` function to rarefy each sample to this level,
4. Plot the rarefaction results, and
5. Add the 1:1 line and label.

```r
# 1. Calculate observed richness for all samples in `soilbac`,
soilbac.S = S.obs(soilbac.t)
soilbac.S
```

```
## T1_1 T1_2 T1_3 T7_1 T7_2 T7_3 DF_1 DF_2 CF_1 CF_2 CF_3
## 1074 1302 1174 1416 1406 1143 1806 1151  924 1122  851
```

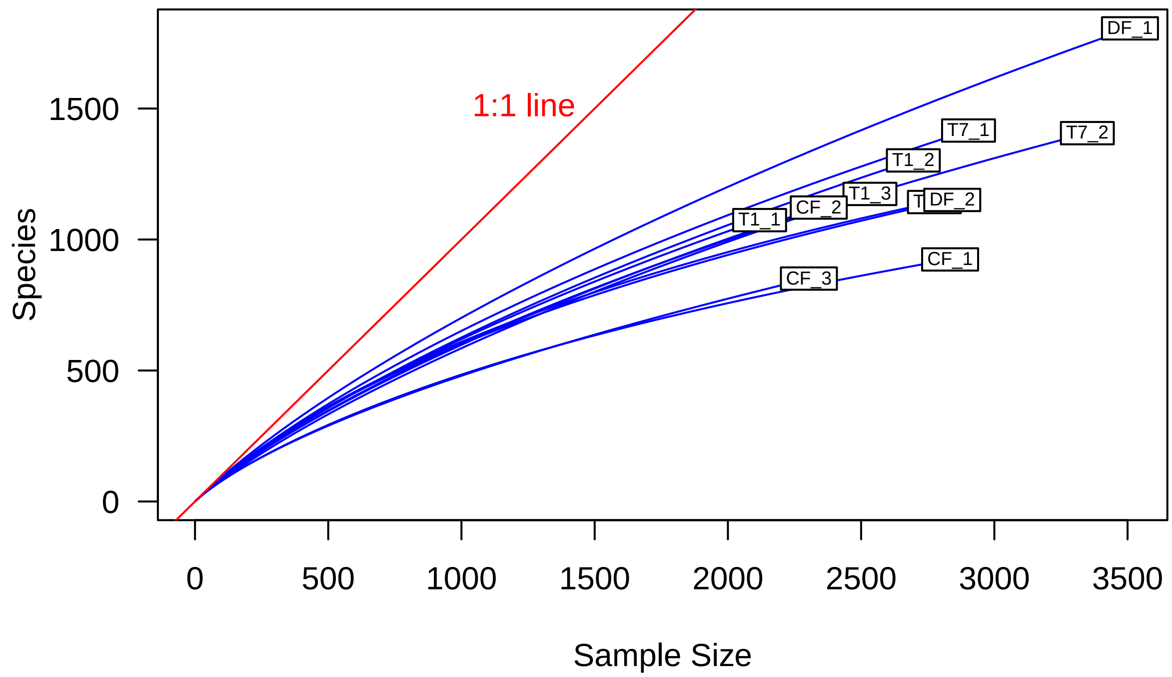
```r
# 2. Determine the size of the smallest sample,
min.N = min(rowSums(soilbac.t))
min.N
```

```
## [1] 2119
```

```r
# 3. Use the `rarefy()` function to rarefy each sample to this level,
S.rarefy = rarefy(x = soilbac.t, sample = min.N, se = TRUE)
S.rarefy
```

```
##    T1_1       T1_2        T1_3       T7_1       T7_2       T7_3       DF_1
## S  1074 1099.69226 1033.618984 1138.85781 1039.66098 984.552923 1254.09586
## se    0    9.92876    8.668344   11.10399   12.38929   9.376365   13.53094
##          DF_2       CF_1        CF_2       CF_3
## S  973.839396 783.458905 1045.431707 804.746297
## se   9.782744   9.075373    6.673692   5.623012
## attr(,"Subsample")
## [1] 2119
```

```r
# 4. Plot the rarefaction results, and
rarecurve(x=soilbac.t, step=20, col="blue", cex=0.6, las=1)
# 5. Add the 1:1 line and label.
abline(0, 1, col="red")
text(1500, 1500, "1:1 line", col="red", pos=2)
```



4) SPECIES EVNENNESS

Here, we consider how abundance varies among species, that is, **species evenness**.

Visualizing evenness: the rank abundance curve (RAC)

One of the most common ways to visualize evenness is in a **rank-abundance curve** (sometime referred to as a rank-abundance distribution or Whittaker plot). An RAC can be constructed by ranking species from the most abundant to the least abundant without respect to species labels (and hence no worries about 'ties' in abundance).

In the R code chunk below, do the following:

1. Write a function to construct a RAC,
2. Be sure your function removes species that have zero abundances,
3. Order the vector (RAC) from greatest (most abundant) to least (least abundant), and
4. Return the ranked vector

```
# 1. Write a function to construct a RAC,
RAC = function(x = ""){
    x.abs = x[x>0]
    x.abs.ranked = x.abs[order(-x.abs)]
```

```
    as.data.frame(lapply(x.abs.ranked, unlist))
    return(x.abs.ranked)

}
```

Now, let us examine the RAC for `site1` of the BCI data set.

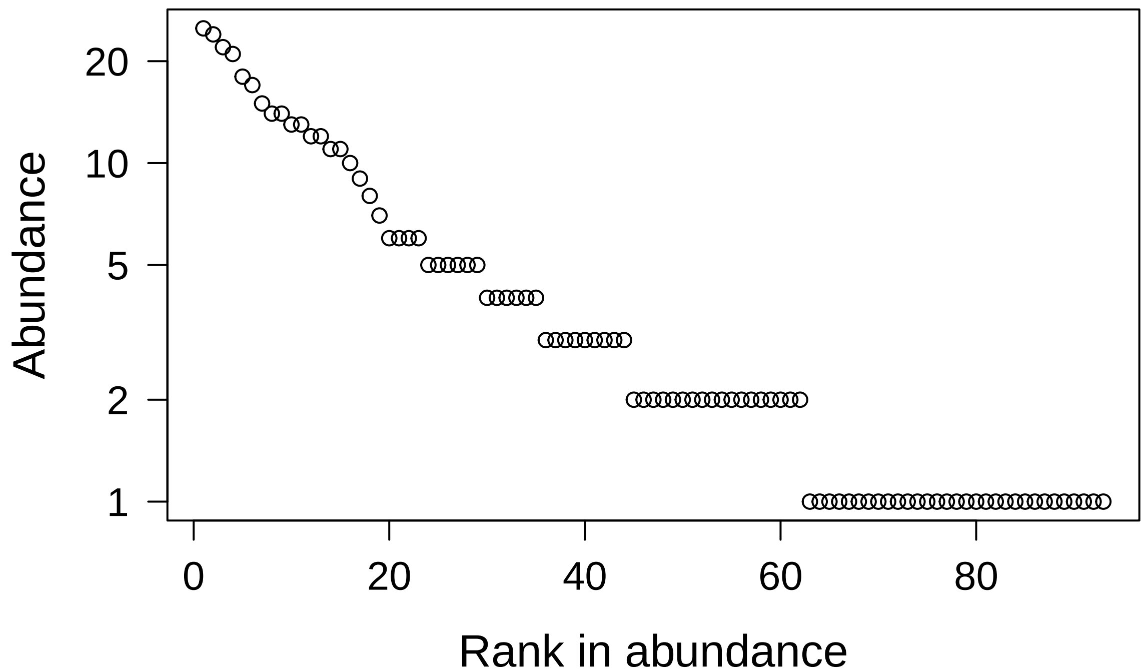
In the R code chunk below, do the following:

1. Create a sequence of ranks and plot the RAC with natural-log-transformed abundances,
2. Label the x-axis "Rank in abundance" and the y-axis "log(abundance)"

```
# 1. Create a sequence of ranks and plot the RAC with natural-log-transformed abundances,
# 2. Label the x-axis "Rank in abundance" and the y-axis "log(abundance)"
plot.new()
site1 = BCI[1,]

rac = RAC(x=site1)
ranks = as.vector(seq(1, length(rac)))

opar = par(no.readonly = TRUE)              # save default parameters
par(mar = c(5.1, 5.1, 4.1, 2.1))             # set margins
plot(ranks, log(rac),
    xlab = "Rank in abundance", ylab = "Abundance",
    type = "p", axes=F, las=1, cex.lab=1.4, cex.axis=1.25)
box()
axis(side=1, labels=T, cex.axis=1.25)
axis(side=2, las=T, cex.axis=1.25, labels = c(1,2,5,10,20), at=log(c(1,2,5,10,20))) # manually set y-ax
```




```
par = opar
```

Question 5: What effect does visualizing species abundance data on a log-scaled axis have on how we interpret evenness in the RAC?

> ***Answer 5***: The log-scaled axis helps to increase the smaller value gaps and decrease the larger value gaps. This allows us to better visualize the evenness of the species in the community.

Now that we have visualized unevennes, it is time to quantify it using Simpson's evenness ($E_{1/D}$) and Smith and Wilson's evenness index (E_{var}).

Simpson's evenness ($E_{1/D}$)

In the R code chunk below, do the following:

1. Write the function to calculate $E_{1/D}$, and
2. Calculate $E_{1/D}$ for `site1`.

```
# 1. Write the function to calculate $E_{1/D}$, and
SimpE = function(x=""){
    S = S.obs(x)
    x = as.data.frame(x)
    D = diversity(x, "inv")
```

```
    E = D/S
    return(E)
}
# 2. Calculate $E_{1/D}$ for `site1`.
SimpE(site1)
```

```
##         1
## 0.4238232
```

```
# ?diversity
```

Smith and Wilson's evenness index (E_{var})

In the R code chunk below, please do the following:

1. Write the function to calculate E_{var},
2. Calculate E_{var} for `site1`, and
3. Compare $E_{1/D}$ and E_{var}.

```
# 1. Write the function to calculate $E_{var}$,
Evar = function(x){
    x = as.vector(x[x>0])
    1 - (2 / pi) * atan(var(log(x)))
}
# 2. Calculate $E_{var}$ for `site1`, and
Evar(site1)
```

```
## [1] 0.5067211
```

```
# 3. Compare $E_{1/D}$ and $E_{var}$.
print(paste("Simpson Evenness index:", SimpE(site1)))
```

```
## [1] "Simpson Evenness index: 0.423823159246214"
```

```
print(paste("Smith and Wilson's Evenness index:", Evar(site1)))
```

```
## [1] "Smith and Wilson's Evenness index: 0.506721104457681"
```

Question 6: Compare estimates of evenness for `site1` of BCI using $E_{1/D}$ and E_{var}. Do they agree? If so, why? If not, why? What can you infer from the results.

> ***Answer 6***: I think the indices do not agree because they measure different concepts of evenness. Simpson's index reflects dominance effects more strongly and S&W's index focuses more on how evenly abundances are distributed across species.In this case, we can see that Simpson's Evenness index is slightly lower than S&W's Evenness index, suggesting that the community has some dominant species that reduce the evenness.

5) INTEGRATING RICHNESS AND EVENNESS: DIVERSITY METRICS

So far, we have introduced two primary aspects of diversity, i.e., richness and evenness. Here, we will use popular indices to estimate diversity, which explicitly incorporate richness and evenness. We will write our own diversity functions and compare them against the functions in `vegan`.

Shannon's diversity (a.k.a., Shannon's entropy)

In the R code chunk below, please do the following:

1. Provide the code for calculating H' (Shannon's diversity),
2. Compare this estimate with the output of `vegan`'s diversity function using method = "shannon".

```
# 1. Provide the code for calculating H' (Shannon's diversity),
H = function(x = ""){
    H = 0
    for (n_i in x){
        if (n_i > 0){
            p_i = n_i / sum(x)
            H = H - p_i * log(p_i)
        }
    }
    return(H)
}

# 2. Compare this estimate with the output of `vegan`'s diversity function using method = "shannon".
print(paste("Our Shannon's diversity:", H(site1)))
```

```
## [1] "Our Shannon's diversity: 4.01841166223236"
```

```
print(paste("Vegan's Shannon's diversity:", diversity(site1, "shannon")))
```

```
## [1] "Vegan's Shannon's diversity: 4.01841166223236"
```

Simpson's diversity (or dominance)

In the R code chunk below, please do the following:

1. Provide the code for calculating D (Simpson's diversity),
2. Calculate both the inverse (1/D) and 1 - D,
3. Compare this estimate with the output of `vegan's` diversity function using method = "simp".

```
# 1. Provide the code for calculating D (Simpson's diversity),
SimpD = function(x=""){
    D = 0
    N = sum(x)
    for (n_i in x){
        D = D + (n_i^2)/(N^2)
```

```
    }
    return(D)
}
# 2. Calculate both the inverse (1/D) and 1 - D,
D.inv = 1/SimpD(site1)
D.sub = 1 - SimpD(site1)
# 2. Compare this estimate with the output of `vegan's` diversity function using method = "simp".
print(paste("Our Simpson's diversity:", SimpD(site1)))
```

```
## [1] "Our Simpson's diversity: 0.0253706951530612"
```

```
print(paste("Vegan's Simpson's diversity:", diversity(site1, "simp")))
```

```
## [1] "Vegan's Simpson's diversity: 0.974629304846939"
```

```
print(paste("Our Inverse Simpson's diversity:", D.inv))
```

```
## [1] "Our Inverse Simpson's diversity: 39.4155538098979"
```

```
print(paste("Vegan's Inverse Simpson's diversity:", diversity(site1, "inv")))
```

```
## [1] "Vegan's Inverse Simpson's diversity: 39.4155538098979"
```

```
print(paste("Our 1 - Simpson's diversity:", D.sub))
```

```
## [1] "Our 1 - Simpson's diversity: 0.974629304846939"
```

```
print(paste("Vegan's 1 - Simpson's diversity:", 1 - diversity(site1, "simp")))
```

```
## [1] "Vegan's 1 - Simpson's diversity: 0.0253706951530612"
```

Fisher's $\boldsymbol{\alpha}$

In the R code chunk below, please do the following:

1. Provide the code for calculating Fisher's $\boldsymbol{\alpha}$,
2. Calculate Fisher's $\boldsymbol{\alpha}$ for `site1` of BCI.

```
# rac = as.vector(site1[site1 > 0])
# invD = diversity(rac, "inv")
# invD

# Fisher = fisher.alpha(rac)
# Fisher

# 1. Provide the code for calculating Fisher's $\boldsymbol\alpha$,
# Fisher = fisher.alpha()


# 2. Calculate Fisher's $\boldsymbol\alpha$ for `site1` of BCI.
print(paste("Fisher's alpha of site1:", fisher.alpha(site1)))
```

```
## [1] "Fisher's alpha of site1: 35.6729742325981"
```

Question 7: How is Fisher's $\boldsymbol{\alpha}$ different from $E_{H'}$ and E_{var}? What does Fisher's $\boldsymbol{\alpha}$ take into account that $E_{H'}$ and E_{var} do not?

> ***Answer 7***: Fisher's alpha measures richness and does not explicitly measure evenness. Unlike other diversity indices, it is designed to account for richness independently of evenness. Fisher's alpha assumes that species abundances follow a log-series distribution, which makes it particularly useful for communities where this distribution is observed. This assumption allows it to estimate richness in datasets with skewed abundance distributions. Fisher's alpha takes into account the number of species and the relative abundance of each species in the community, while $E_{H'}$ and E_{var} do not.

6) HILL NUMBERS

Remember that we have learned about the advantages of Hill Numbers to measure and compare diversity among samples. We also learned to explore the effects of rare species in a community by examining diversity for a series of exponents q.

Question 8: Using `site1` of BCI and `vegan` package, a) calculate Hill numbers for q exponent 0, 1 and 2 (richness, exponential Shannon's entropy, and inverse Simpson's diversity). b) Interpret the effect of rare species in your community based on the response of diversity to increasing exponent q.

```
D_0 = S.obs(site1)
D_1 = exp(diversity(site1, index="shannon"))
D_2 = diversity(site1, index="invsimpson")
tsallis(site1, scales = seq(0,2,1), hill=TRUE)
```

```
##        0        1        2
## 93.00000 55.61270 39.41555
## attr(,"class")
## [1] "tsallis" "renyi"   "numeric"
```

```
print(paste("Hill number for q=0:", D_0))
```

```
## [1] "Hill number for q=0: 93"
```

```
print(paste("Hill number for q=1:", D_1))
```

```
## [1] "Hill number for q=1: 55.612703881371"
```

```
print(paste("Hill number for q=2:", D_2))
```

```
## [1] "Hill number for q=2: 39.4155538098979"
```

> ***Answer 8a***: Please see the results above: ***Answer 8b***: For q=0, we consider all species equally, regardless of their abundances. For q=1, we weight species according to their relative abundances, but all species contribute proportionally. Rare species still influence the diversity estimate, but less than in q=0. For q=2, we give more weight to the most abundant species, and rare species have little influence on the diversity estimate. This means that as q increases, the diversity measure becomes less sensitive to rare species and more sensitive to common species.

##7) MOVING BEYOND UNIVARIATE METRICS OF α DIVERSITY

The diversity metrics that we just learned about attempt to integrate richness and evenness into a single, univariate metric. Although useful, information is invariably lost in this process. If we go back to the rank-abundance curve, we can retrieve additional information – and in some cases – make inferences about the processes influencing the structure of an ecological system.

Species abundance models

The RAC is a simple data structure that is both a vector of abundances. It is also a row in the site-by-species matrix (minus the zeros, i.e., absences).

Predicting the form of the RAC is the first test that any biodiversity theory must pass and there are no less than 20 models that have attempted to explain the uneven form of the RAC across ecological systems.

In the R code chunk below, please do the following:

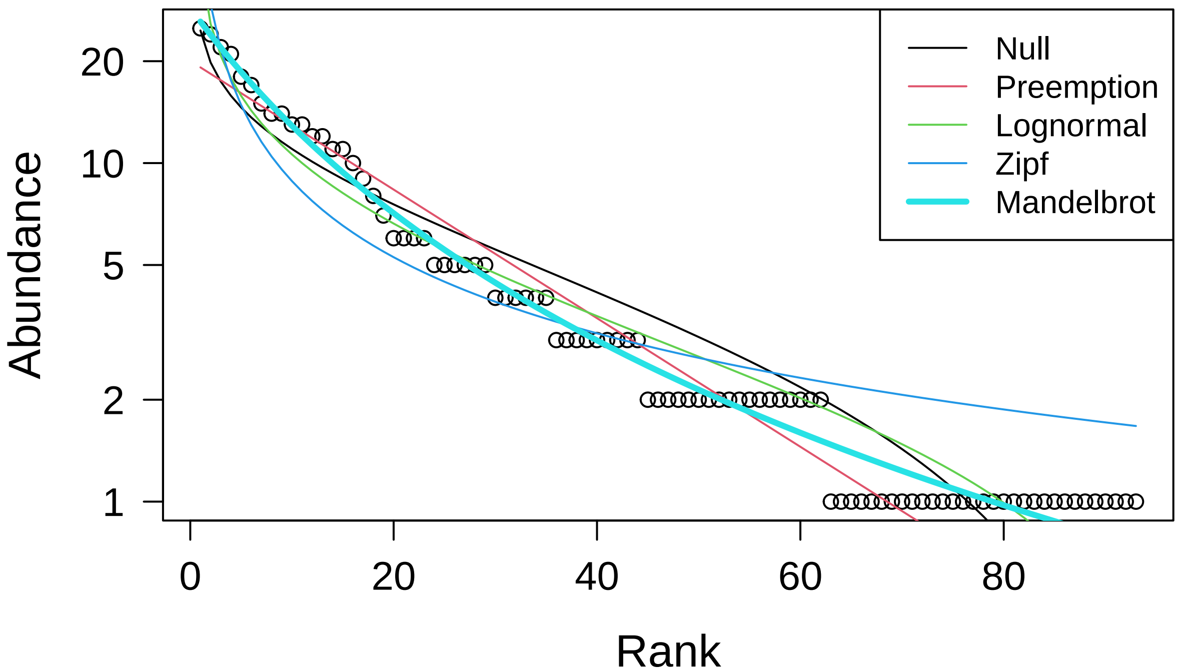
1. Use the `radfit()` function in the `vegan` package to fit the predictions of various species abundance models to the RAC of `site1` in BCI,
2. Display the results of the `radfit()` function, and
3. Plot the results of the `radfit()` function using the code provided in the handout.

```
# 1. Use the `radfit()` function in the `vegan` package to fit the predictions of various species abund
RACresults = radfit(site1)

# 2. Display the results of the `radfit()` function, and
RACresults
```

```
##
## RAD models, family poisson
## No. of species 93, total abundance 448
##
##             par1      par2     par3    Deviance AIC      BIC
## Null                                    39.5261 315.4362 315.4362
## Preemption  0.042797                    21.8939 299.8041 302.3367
## Lognormal   1.0687    1.0186            25.1528 305.0629 310.1281
## Zipf        0.11033  -0.74705           61.0465 340.9567 346.0219
## Mandelbrot  100.52   -2.312    24.084    4.2271 286.1372 293.7350
```

```
# 3. Plot the results of the `radfit()` function using the code provided in the handout.
plot.new()
plot(RACresults, las=1, cex.lab=1.4, cex.axis=1.25)
```



Question 9: Answer the following questions about the rank abundance curves: a) Based on the output of `radfit()` and plotting above, discuss which model best fits our rank-abundance curve for `site1`? b) Can we make any inferences about the forces, processes, and/or mechanisms influencing the structure of our system, e.g., an ecological community?

> ***Answer 9a***: Based on the output of 'radfit()', the model that fits best for site1 should be the one with lowest deviance value, and lowest AIC, BIC scores which is in this case Mandelbrot. Also for the plot, we see that the Mendelbrot model fits the data best. ***Answer 9b***: Based on what I've read about the Mandelbrot model, it's a generalized Zipf law which assumes species abundances follow a power-law distribution, often observed in communities with strong dominance and resource constraints. I think this suggests a highly uneven structure with a few dominant species and many rare species. This pattern can arise from strong competition among species, where dominant species monopolize resources, limiting the abundance of others.

Question 10: Answer the following questions about the preemption model: a. What does the preemption model assume about the relationship between total abundance (N) and total resources that can be preempted? b. Why does the niche preemption model look like a straight line in the RAD plot?

> ***Answer 10a***: The preemption model assumes a geometric distribution where each species preempts a fixed proportion of the resources. ***Answer 10b***: This is because the plot was in log scale and we have a geometric distribution.

Question 11: Why is it important to account for the number of parameters a model uses when judging how well it explains a given set of data?

Answer 11: Normally, the more complex the model is, the better it will fit the data. However, this is not always the case. A model that is too complex may overfit the data, meaning it will fit the noise in the data rather than the underlying pattern.

SYNTHESIS

1. As stated by Magurran (2004) the $D = \sum p_i^2$ derivation of Simpson's Diversity only applies to communities of infinite size. For anything but an infinitely large community, Simpson's Diversity index is calculated as $D = \sum \frac{n_i(n_i-1)}{N(N-1)}$. Assuming a finite community, calculate Simpson's D, 1 - D, and Simpson's inverse (i.e. 1/D) for `site 1` of the BCI site-by-species matrix.

```
Simp_D2004 = function(x){
    N = sum(x)
    D = 0
    for (n_i in x){
        D = D + (n_i * (n_i - 1)) / (N * (N - 1))
    }
    return(D)
}

print(paste("Simpson's D:", Simp_D2004(site1)))
```

```
## [1] "Simpson's D: 0.0231903163950144"
```

```
print(paste("1 - Simpson's D:", 1 - Simp_D2004(site1)))
```

```
## [1] "1 - Simpson's D: 0.976809683604986"
```

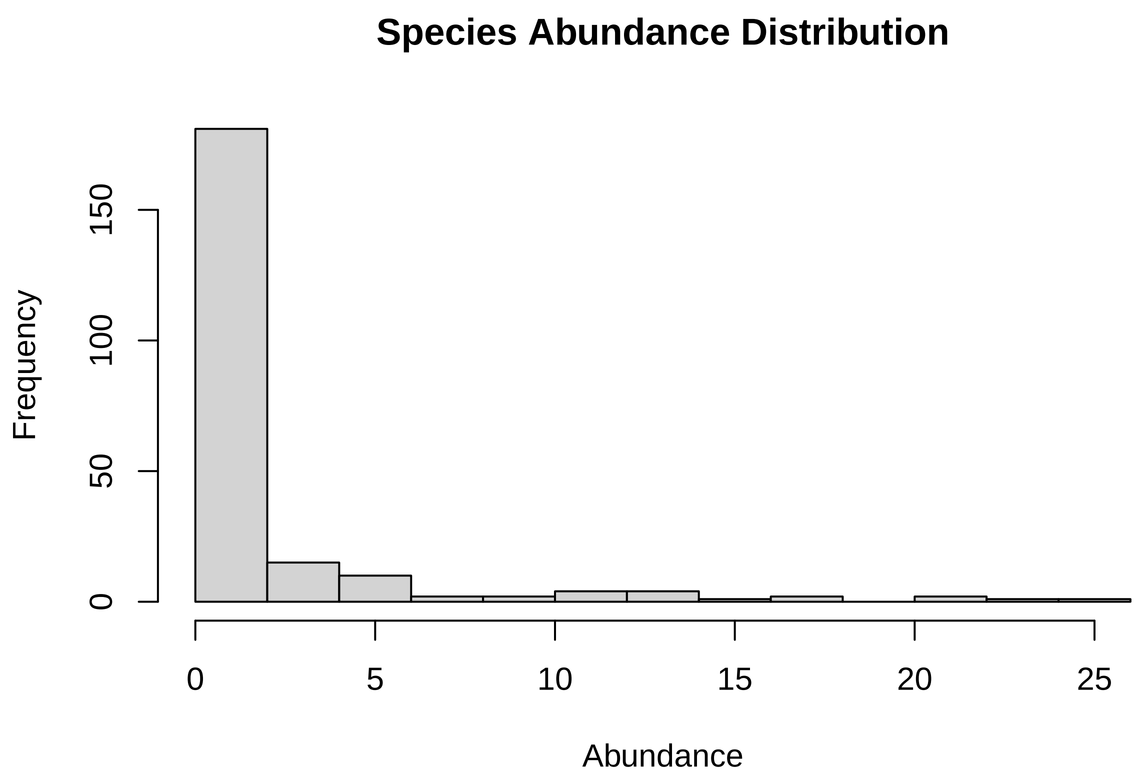
```
print(paste("1/Simpson's D:", 1/Simp_D2004(site1)))
```

```
## [1] "1/Simpson's D: 43.1214470284238"
```

2. Along with the rank-abundance curve (RAC), another way to visualize the distribution of abundance among species is with a histogram (a.k.a., frequency distribution) that shows the frequency of different abundance classes. For example, in a given sample, there may be 10 species represented by a single individual, 8 species with two individuals, 4 species with three individuals, and so on. In fact, the rank-abundance curve and the frequency distribution are the two most common ways to visualize the species-abundance distribution (SAD) and to test species abundance models and biodiversity theories. To address this homework question, use the R function **hist()** to plot the frequency distribution for `site 1` of the BCI site-by-species matrix, and describe the general pattern you see.

```
# convert back to be matrix to extract numerical data
site1_abundances <- as.vector(as.matrix(site1))

plot.new()
hist(site1_abundances,
 main = "Species Abundance Distribution",
 xlab = "Abundance", ylab = "Frequency")
```



3. We asked you to find a biodiversity dataset with your partner. This data could be one of your own or it could be something that you obtained from the literature. Load that dataset.

```
# Load your biodiversity dataset
data = read.table("./data/sps_macrozoobenthos_allyear.csv", sep=",", header=TRUE, row.names=1)
data = as.data.frame(data)
# summary(data)
dim(data)
```

```
## [1]  19 211
```

How many sites are there? How many species are there in the entire site-by-species matrix? There are 19 sites and 211 species in the data.

Any other interesting observations based on what you learned this week? Please see the analysis below. First, our dataset has quite good coverage. Most species are not singletons. Second, chao1 estimators show that accross sites, we have more species than we observed. Third, the chao2 estimator is lower than the chao1 estimator, suggesting that there are less singletons and doubletons in the dataset. For the rarefaction curve, we see that most curves rise sharply at the beginning, which means that the more sampling there are, the more new species there are. But then, many curves' slopes start to stabilize at higher sample sizes, this suggests that suggesting most species have already been observed.

Different sites have different asymptotic behavior. Some sites, for ex: "site_54_73_12.79," continue to rise even at large sample sizes, indicating incomplete sampling and the possibility of more undiscovered species. Other sites reach a plateau, which means that nearly all species present have been sampled. Sites with curves that flatten early have lower overall diversity. Sites with long-rising curves suggest greater species richness that requires more sampling effort to capture.

```
# 1. Calculate the observed richness for all sites in the data

data_S = S.obs(data)
data_S
```

```
## site_54.03_11.37 site_54.06_11.06 site_54.11_11.18 site_54.18_11.75
##               40               11               20              130
## site_54.22_12.09 site_54.32_11.55  site_54.47_12.2 site_54.55_11.33
##               79               80                6              130
## site_54.63_14.28 site_54.71_13.95  site_54.72_13.7 site_54.73_12.79
##               49               31               16               98
##  site_54.93_13.5  site_55.0_14.08 site_55.25_14.51 site_55.25_15.98
##               25               43               36               19
##  site_55.55_18.4  site_59.58_23.3  site_59.92_25.6
##                8                4                3
```

```
# 2. Determine the size of the smallest sample
min.N_data = min(rowSums(data))
min.N_data
```

```
## [1] 33
```

```
print(paste("The size of the smallest sample is:", min.N_data))
```

```
## [1] "The size of the smallest sample is: 33"
```

```
# Good's Coverage
C_data = C(data)
C_data
```

```
## site_54.03_11.37 site_54.06_11.06 site_54.11_11.18 site_54.18_11.75
##        0.9999357        0.9989562        1.0000000        0.9997142
## site_54.22_12.09 site_54.32_11.55  site_54.47_12.2 site_54.55_11.33
##        0.9996052        0.9994608        1.0000000        0.9992967
## site_54.63_14.28 site_54.71_13.95  site_54.72_13.7 site_54.73_12.79
##        0.9998598        1.0000000        0.9996003        0.9998784
##  site_54.93_13.5  site_55.0_14.08 site_55.25_14.51 site_55.25_15.98
##        1.0000000        0.9995653        0.0000000        0.9989950
##  site_55.55_18.4  site_59.58_23.3  site_59.92_25.6
##        1.0000000        1.0000000        1.0000000
```

```
# Richness estimators
S.chao1_data = S.chao1(data)
print("Chao1 richness estimator for the data:")
```

```
## [1] "Chao1 richness estimator for the data:"
```

```r
S.chao1_data
```

```
## site_54.03_11.37 site_54.06_11.06 site_54.11_11.18 site_54.18_11.75
##         312.8421         283.8421         292.8421         402.8421
## site_54.22_12.09 site_54.32_11.55  site_54.47_12.2 site_54.55_11.33
##         351.8421         352.8421         278.8421         402.8421
## site_54.63_14.28 site_54.71_13.95  site_54.72_13.7 site_54.73_12.79
##         321.8421         303.8421         288.8421         370.8421
##  site_54.93_13.5  site_55.0_14.08 site_55.25_14.51 site_55.25_15.98
##         297.8421         315.8421         308.8421         291.8421
##  site_55.55_18.4  site_59.58_23.3  site_59.92_25.6
##         280.8421         276.8421         275.8421
```

```r
S.chao2_data = S.chao2(1, data)
print("Chao2 richness estimator for the data at first site:")
```

```
## [1] "Chao2 richness estimator for the data at first site:"
```

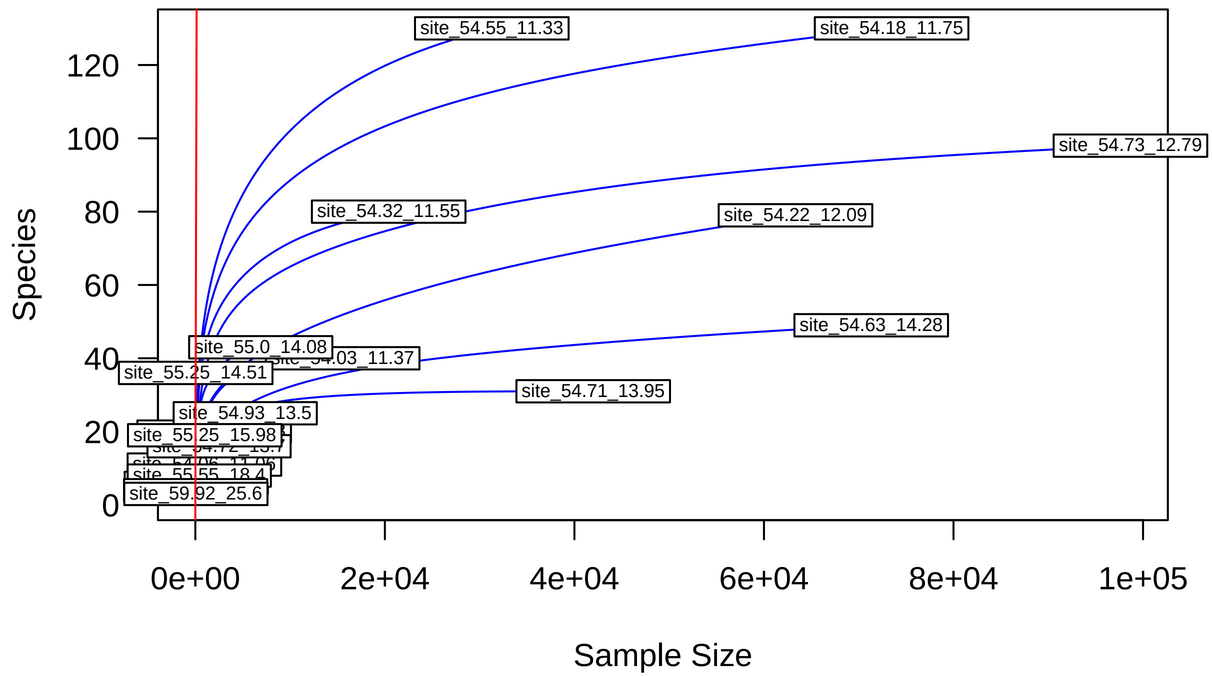
```r
S.chao2_data
```

```
## site_54.03_11.37
##         78.78205
```

```r
# Calculate the rarefaction curve for the data
S.rarefy_data = rarefy(x = data, sample = min.N_data, se = TRUE)
S.rarefy_data
```

```
##    site_54.03_11.37 site_54.06_11.06 site_54.11_11.18 site_54.18_11.75
## S          8.284741         4.469775         6.452522        13.749274
## se         1.450905         1.221011         1.222613         1.919918
##    site_54.22_12.09 site_54.32_11.55 site_54.47_12.2 site_54.55_11.33
## S          7.622998        11.826803       5.9709146        12.043799
## se         1.445091         1.858434       0.1687044         1.949651
##    site_54.63_14.28 site_54.71_13.95 site_54.72_13.7 site_54.73_12.79
## S          5.500678         7.996178        8.994003         9.654866
## se         1.250004         1.209397        1.210248         1.679723
##    site_54.93_13.5 site_55.0_14.08 site_55.25_14.51 site_55.25_15.98
## S         8.507741        9.903226               33         7.611853
## se        1.595841        1.775658                0         1.392250
##    site_55.55_18.4 site_59.58_23.3 site_59.92_25.6
## S         5.256999               4       2.9197735
## se        1.033516               0       0.2716753
## attr(,"Subsample")
## [1] 33
```

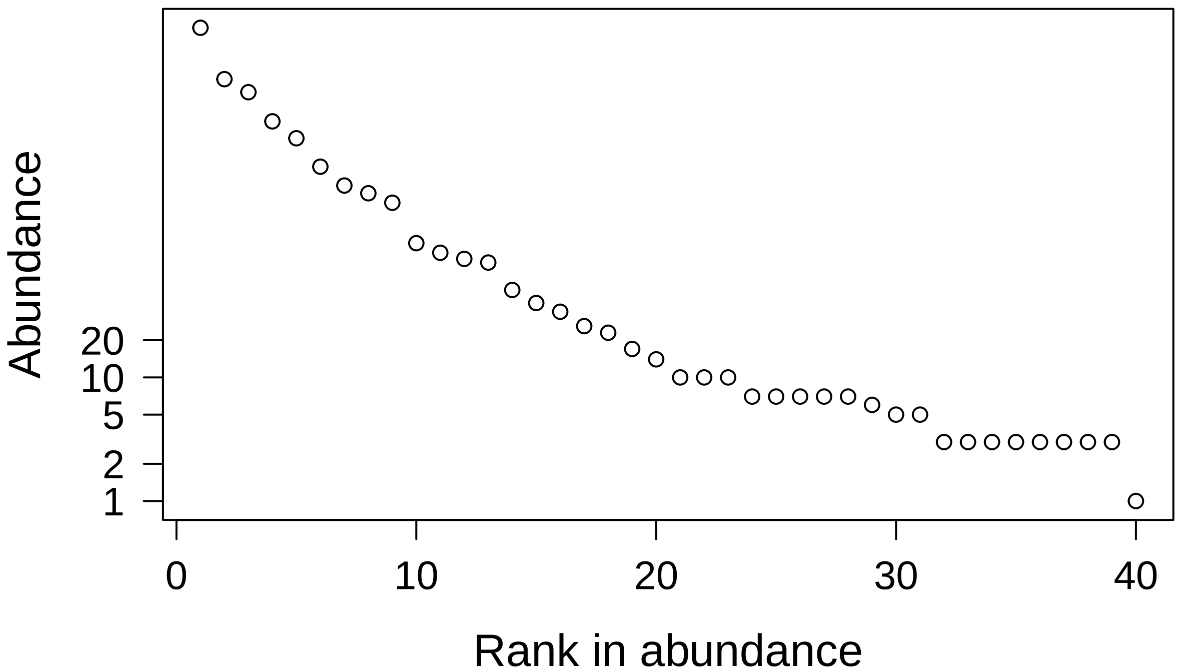
```r
# Plot the rarefaction curve for the data
plot.new()
rarecurve(x=data, step=20, col="blue", cex=0.6, las=1)
# 5. Add the 1:1 line and label.
abline(0, 1, col="red")
text(1500, 1500, "1:1 line", col="red", pos=2)
```




```
# Calculate the Shannon's diversity for all sites in the data
H_data = apply(data, 1, H)
H_data
```

```
## site_54.03_11.37 site_54.06_11.06 site_54.11_11.18 site_54.18_11.75
##        1.8918530        0.8645374        1.6379961        2.8089316
## site_54.22_12.09 site_54.32_11.55  site_54.47_12.2 site_54.55_11.33
##        1.6515312        2.5198529        1.7480673        2.5513930
## site_54.63_14.28 site_54.71_13.95  site_54.72_13.7 site_54.73_12.79
##        1.3133380        1.9350693        2.0843924        2.1721697
##  site_54.93_13.5  site_55.0_14.08 site_55.25_14.51 site_55.25_15.98
##        1.8330485        2.0929787        3.5835189        1.6619967
##  site_55.55_18.4  site_59.58_23.3  site_59.92_25.6
##        1.2627638        1.2899560        0.6194392
```

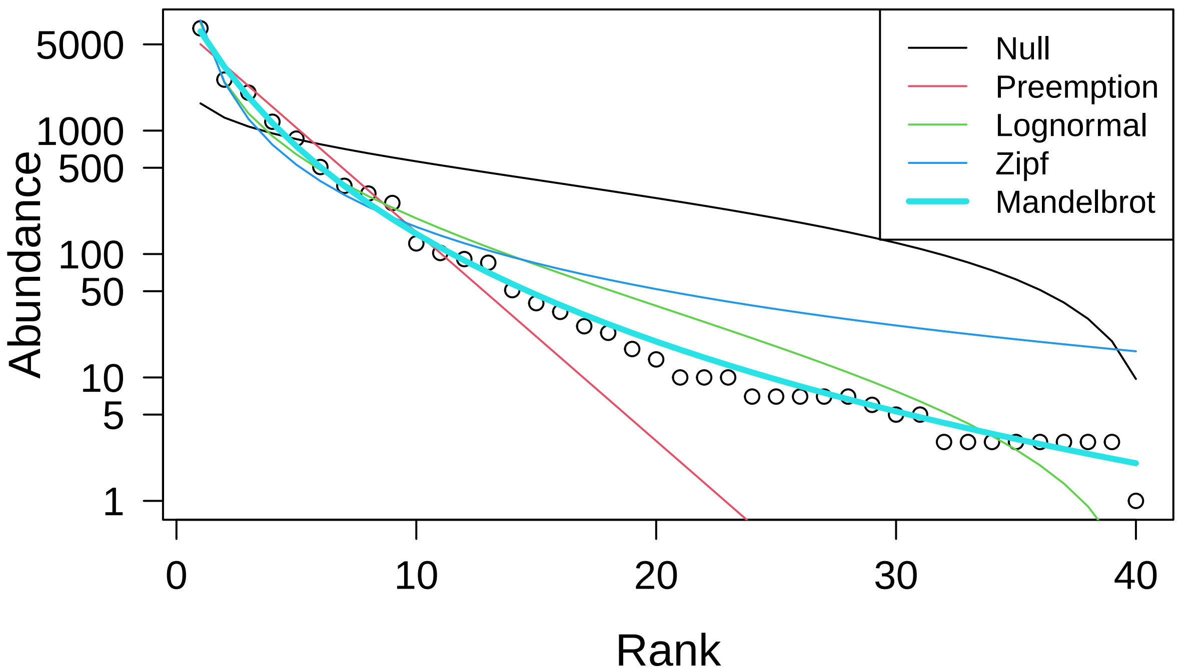
```
# Check the RAC for the data
site1 = data[1,]
rac_data = RAC(x=site1)
ranks_data = as.vector(seq(1, length(rac_data)))
plot.new()
plot(ranks_data, log(rac_data),
    xlab = "Rank in abundance", ylab = "Abundance",
    type = "p", axes=F, las=1, cex.lab=1.4, cex.axis=1.25)
box()
axis(side=1, labels=T, cex.axis=1.25)
axis(side=2, las=T, cex.axis=1.25, labels = c(1,2,5,10,20), at=log(c(1,2,5,10,20))) # manually set y-ax
```




```r
# Fit the species abundance models to the RAC of site1 in the data
RACresults_data = radfit(site1)
RACresults_data
```

```
##
## RAD models, family poisson
## No. of species 40, total abundance 15546
##
##            par1     par2    par3    Deviance AIC      BIC
## Null                                20848.29 21056.08 21056.08
## Preemption  0.32267                  1774.86  1984.65  1986.34
## Lognormal   3.564    2.3904           835.63  1047.42  1050.80
## Zipf        0.50369 -1.6739          1843.67  2055.47  2058.84
## Mandelbrot  210.86   -3.7738  4.2293   269.04   482.83   487.90
```

```r
plot.new()
plot(RACresults_data, las=1, cex.lab=1.4, cex.axis=1.25)
```



Next, we also see that the Mandelbrot model fits the dataset the most. As we have seen previously, this model assumes species abundances follow a power-law distribution, often observed in communities with strong dominance and resource constraints.

SUBMITTING YOUR ASSIGNMENT

Use Knitr to create a PDF of your completed 5.AlphaDiversity_Worksheet.Rmd document, push it to GitHub, and create a pull request. Please make sure your updated repo include both the pdf and RMarkdown files.

Unless otherwise noted, this assignment is du