Chapter 6 - Multidimensional Qualitative Data

Contingency tables

The comparison of qualitative descriptors is based on contingency tables.

In multiway tables, the hypothesses testes are often quite complex because they take into account interactions among descriptors.

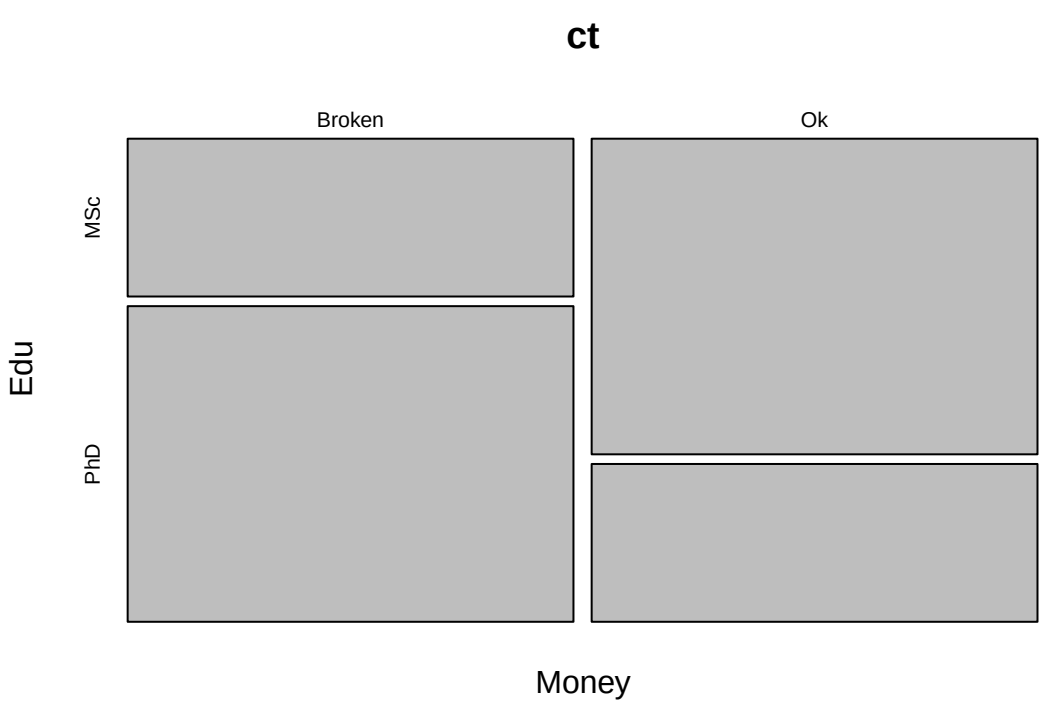
```
Money <- factor(c("Broken","Broken","Ok","Broken","Ok","Ok"))
Edu <- factor(c("PhD","PhD","PhD","MSc","MSc","MSc"))
ct <- table(Money,Edu)
ct
```

```
##         Edu
## Money    MSc PhD
##   Broken   1   2
##   Ok       2   1
```

```
t (ct) #Transpose
```

```
##      Money
## Edu   Broken Ok
##   MSc      1  2
##   PhD      2  1
```

```
mosaicplot(ct) # Mosaicplot
```



```r
### Silly data
x <- c(rep("Ok",0.3*500),rep("Broken",0.7*500))
y <- c(rep("PhD",0.4*500),rep("MSc",0.6*500))
Money <- sample(x)
Edu <- sample(y)
mytable <- table(Money,Edu)
mytable
```

```
##         Edu
## Money    MSc PhD
##   Broken 209 141
##   Ok      91  59
```

```r
margin.table(mytable, 1) # x frequencies (summed over y)
```

```
## Money
## Broken     Ok
##    350    150
```

```r
margin.table(mytable, 2) # y frequencies (summed over x)
```

```
## Edu
## MSc PhD
## 300 200
```

```r
prop.table(mytable, 1) # row percentages (Probability of Broken or ok)
```

```
##         Edu
## Money          MSc       PhD
##   Broken 0.5971429 0.4028571
##   Ok     0.6066667 0.3933333
```

```r
prop.table(mytable, 2) # column percentages (Probability of PhD or MSc)
```

```
##         Edu
## Money          MSc       PhD
##   Broken 0.6966667 0.7050000
##   Ok     0.3033333 0.2950000
```

```r
prop.table(mytable) # cell percentages (Probability of broken/ok being PhD or MSc))
```

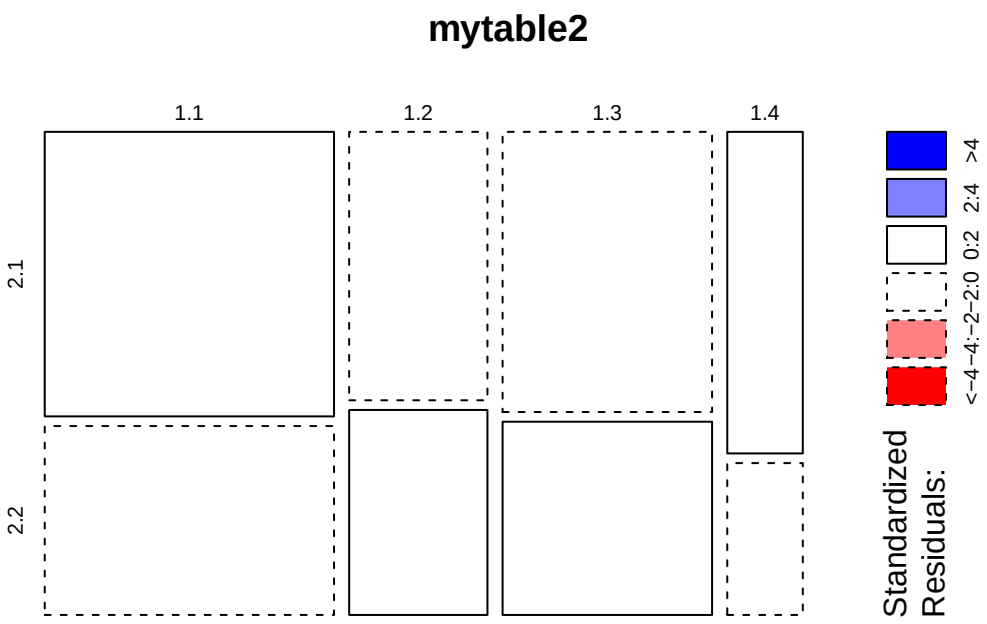
```
##         Edu
## Money      MSc   PhD
##   Broken 0.418 0.282
##   Ok     0.182 0.118
```

```r
### 3-Way Frequency Table
z <- c(rep("happy",0.6*500),rep("unhappy",0.4*500))
Happi <- sample(z)

mytable2 <- ftable(Happi,Money,Edu) #3 way table
mytable2
```

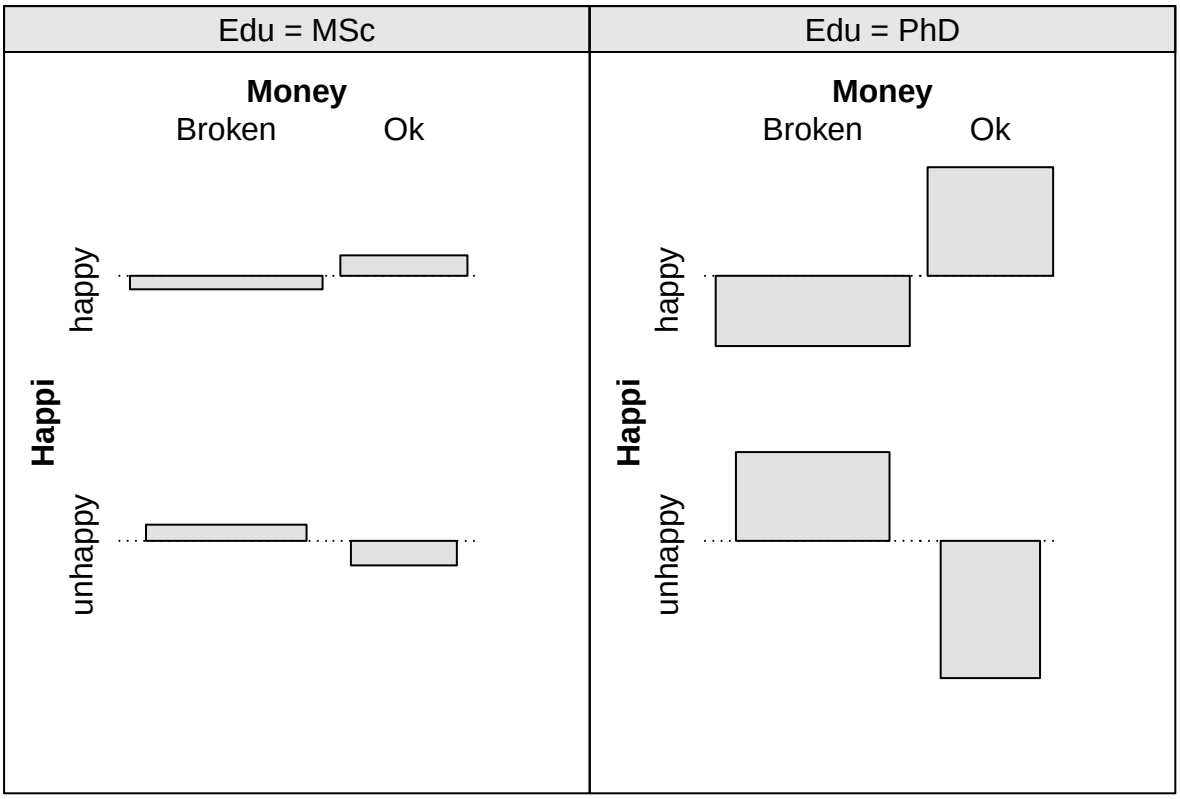
```
##                 Edu MSc PhD
## Happi   Money
## happy   Broken     122  81
##         Ok          55  42
## unhappy Broken      87  60
##         Ok          36  17
```

```r
mosaicplot(mytable2,shade=T) #Standarized resioduals

library (vcd)
```

```
## Loading required package: grid
```




```r
cotabplot(mytable2, panel = cotab_coindep, shade = TRUE, legend = FALSE, type = "assoc")
```



Entropy

One main problem is measuring the amount of information contained in each descriptor, and the amount of information that the two descriptors have in common. When the descriptors are qualitative, the order of the information is not important. In information systems, entropy and information are synonymous. Entropy then is the average number of binary questoins that are required in assigning each object to its correct state. Therfore, how much information is gained by asking binary questions and answering them after observing the objects is equal to the degree of undertainty.

```
# From http://stackoverflow.com/questions/27254550/calculating-entropy
info <- function(CLASS.FREQ){
  freq.class <- CLASS.FREQ
  info <- 0
  for(i in 1:length(freq.class)){
    if(freq.class[[i]] != 0){ # zero check in class
      entropy <- -sum(freq.class[[i]] * log2(freq.class[[i]]))  #I calculate the entropy for each class
    }else{
      entropy <- 0
    }
    info <- info + entropy # sum up entropy from all classes
  }
  return(info)
}

freqs <- table(Edu)/length(Edu)
freqs2 <- table(Money)/length(Money)
```

```r
freqs3 <- table(Happi)/length(Happi)

#Calculate entropy:
info(freqs)  # (Bits)
```

```
## [1] 0.9709506
```

```r
info(freqs2)
```

```
## [1] 0.8812909
```

```r
-sum(freqs * log2(freqs))
```

```
## [1] 0.9709506
```

```r
library (entropy)
entropy.empirical(freqs, unit="log2")
```

```
## [1] 0.9709506
```

```r
#With package entropy
freqs.empirical(freqs) #Edu
```

```
## Edu
## MSc PhD
## 0.6 0.4
```

```r
entropy(freqs, method="ML") # Also "MM", "Jeffreys", "Laplace", "SG", "minimax", "CS", "NSB", "shrink"
```

```
## [1] 0.6730117
```

```r
entropy.empirical(freqs, unit=c("log"))  #Nats
```

```
## [1] 0.6730117
```

```r
entropy.empirical(freqs, unit=c("log2"))  #Bits
```

```
## [1] 0.9709506
```

```r
entropy.empirical(freqs, unit=c("log10")) #Logits
```

```
## [1] 0.2922853
```

```r
# Kullback-Leiber (KL) divergence
KL.plugin (freqs, freqs2)  #from Happi to Money.
```

```
## [1] 0.02258242
```

```r
KL.plugin (freqs2, freqs3)  #from Money to Edu
```

```
## [1] 0.02160085
```

```r
KL.plugin (freqs, freqs3)  #from Happi to Edu
```

```
## [1] 0
```

Also, diversity indexes such as the Shannon's index are a measure of entropy of the system.

LogLinear Hierarchical

```r
library(MASS)
mytable <- xtabs(~ Happi + Money + Edu)  #3-way contingency table

# Mutual Independence:
loglm(~ Happi + Money + Edu, mytable) #Ho: Pairwise independent.
```

```
## Call:
## loglm(formula = ~Happi + Money + Edu, data = mytable)
##
## Statistics:
##                       X^2 df  P(> X^2)
## Likelihood Ratio 3.864311  4 0.4246807
## Pearson          3.687169  4 0.4499948
```

```r
# Partial Independence:
loglm(~ Happi + Money + Edu + Money * Edu, mytable)
```

```
## Call:
## loglm(formula = ~Happi + Money + Edu + Money * Edu, data = mytable)
##
## Statistics:
##                       X^2 df  P(> X^2)
## Likelihood Ratio 3.824586  3 0.2810393
## Pearson          3.697038  3 0.2960916
```

```r
# Ho: Happiness is partially independent of Money and Education
# (i.e., Happi is independent of the composite variable MoneyEdu).

#Conditional Independence:
loglm(~Happi+Money+Edu+ Happi*Edu + Money*Edu, mytable)
```

```
## Call:
## loglm(formula = ~Happi + Money + Edu + Happi * Edu + Money *
##     Edu, data = mytable)
##
## Statistics:
##                       X^2 df  P(> X^2)
## Likelihood Ratio 3.511607  2 0.1727684
## Pearson          3.428215  2 0.1801244
```

```r
#Ho: Happiness is independent of Money, given Edu.

# Ho: No Three-Way Interaction
loglm(~Happi+Money+Edu+Happi*Money+Happi*Edu+Money*Edu, mytable)
```

```
## Call:
## loglm(formula = ~Happi + Money + Edu + Happi * Money + Happi *
##     Edu + Money * Edu, data = mytable)
##
## Statistics:
##                       X^2 df  P(> X^2)
## Likelihood Ratio 1.534708  1 0.2154072
## Pearson          1.522365  1 0.2172620
```

Species Diversity

install.packages('vegan')

```r
library(vegan)
```

```
## Loading required package: permute
## Loading required package: lattice
## This is vegan 2.3-2
```

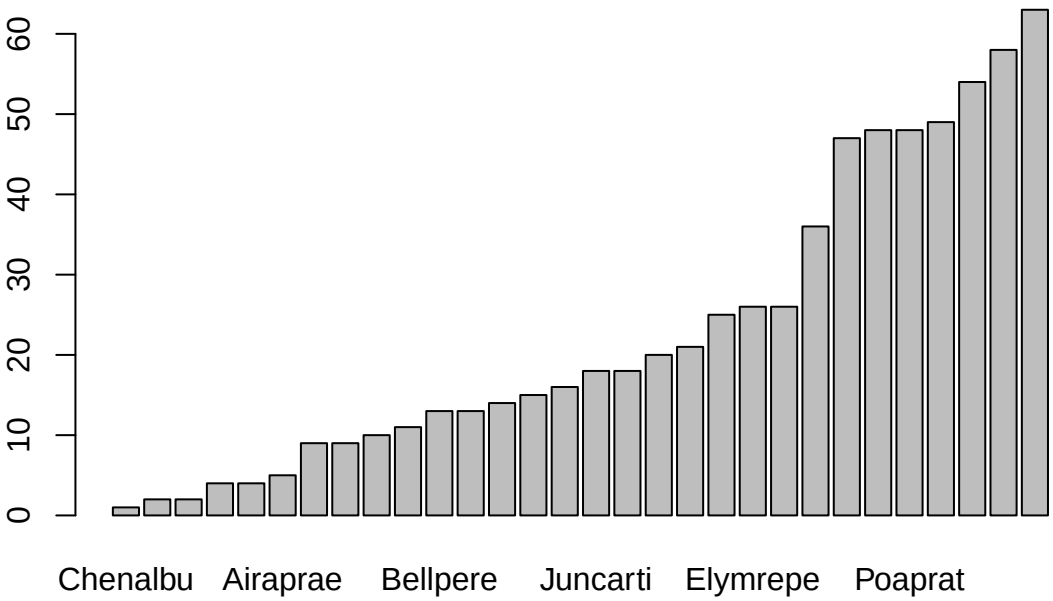
```r
#dune data sets
data(dune) #Species data
data(dune.env) #Environmental data
?dune

#Explore
View(dune)
colSums(dune) #Species total abundances
```

```
## Achimill Agrostol Airaprae Alopgeni Anthodor Bellpere Bromhord Chenalbu
##       16       48        5       36       21       13       15        1
## Cirsarve Comapalu Eleopalu Elymrepe Empenigr Hyporadi Juncarti Juncbufo
##        2        4       25       26        2        9       18       13
## Lolipere Planlanc  Poaprat  Poatriv Ranuflam Rumeacet Sagiproc Salirepe
##       58       26       48       63       14       18       20       11
## Scorautu Trifprat Trifrepe Vicilath Bracruta Callcusp
##       54        9       47        4       49       10
```

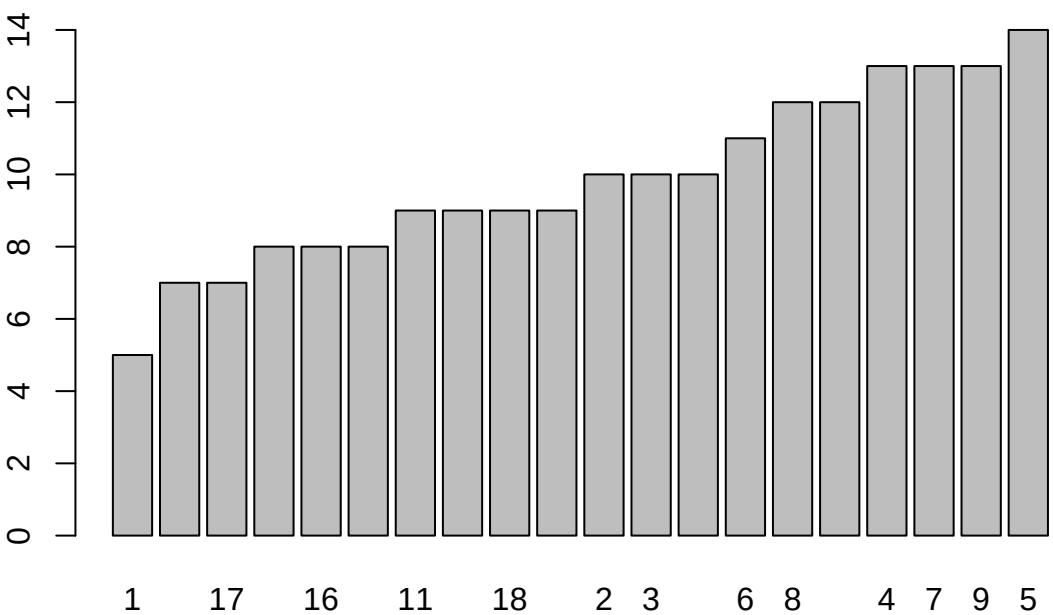
```r
rowSums(dune) #Sites total abundances
```

```
##  1  2  3  4  5  6  7  8  9 10 11 12 13 14 15 16 17 18 19 20
## 18 42 40 45 43 48 40 40 42 43 32 35 33 24 23 33 15 27 31 31
```

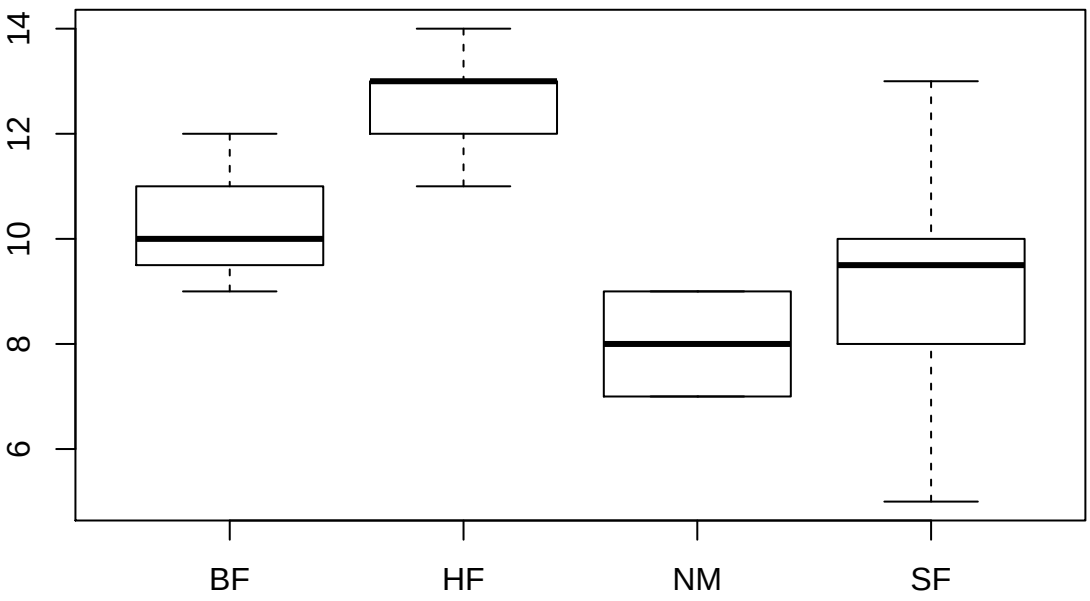
```r
#Distribution of species abundances
barplot(sort(colSums(dune)))
```



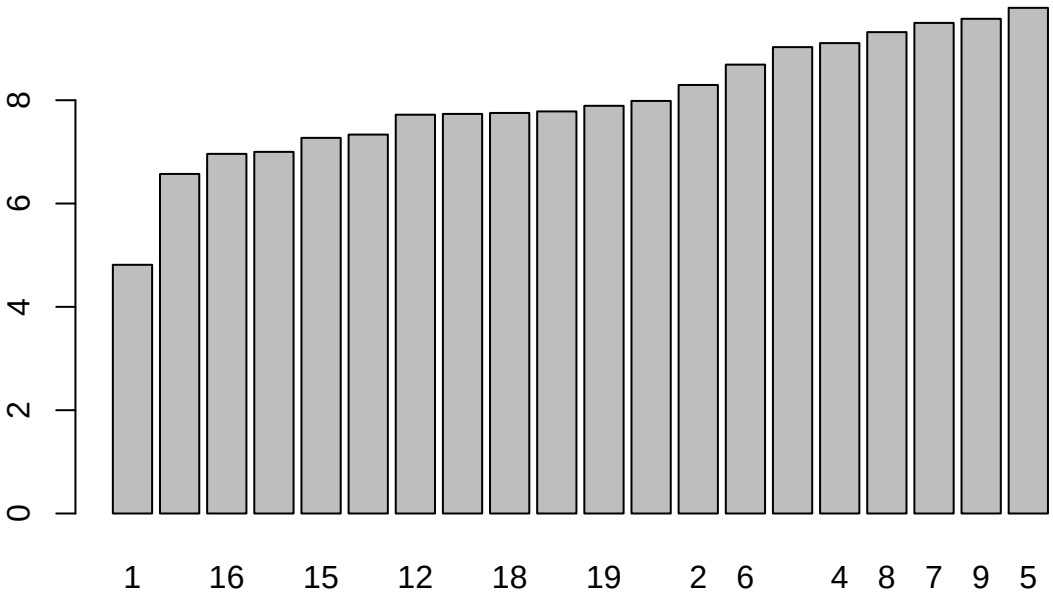

```
### 1. Entropy of order a = 0  ###

# 1.1 simple richness
S <- specnumber(dune)
barplot(sort(S))
```



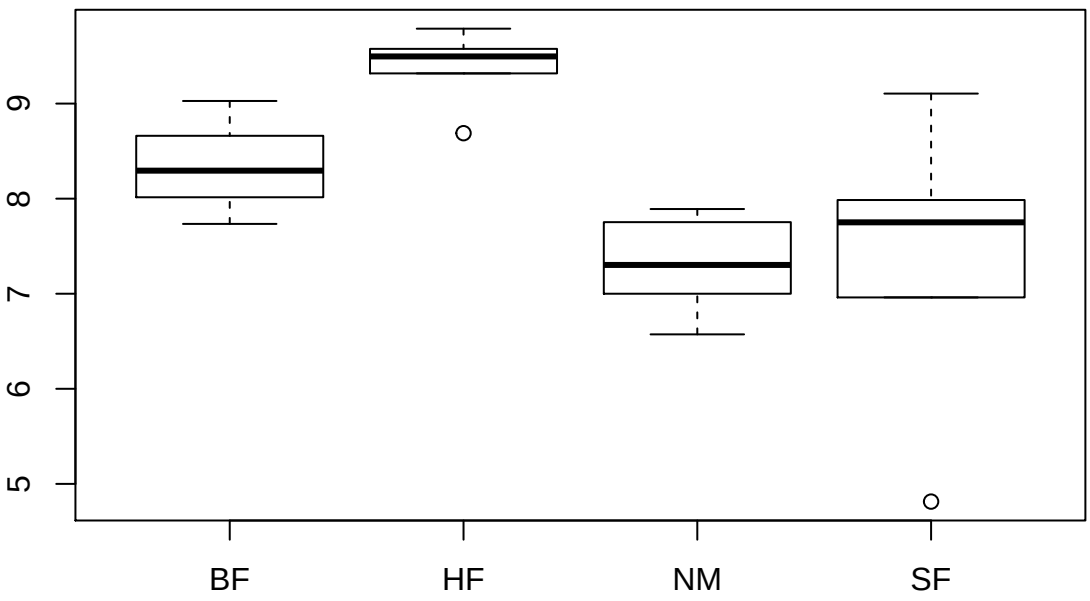

```
plot(x = dune.env$Management, y = S)
```



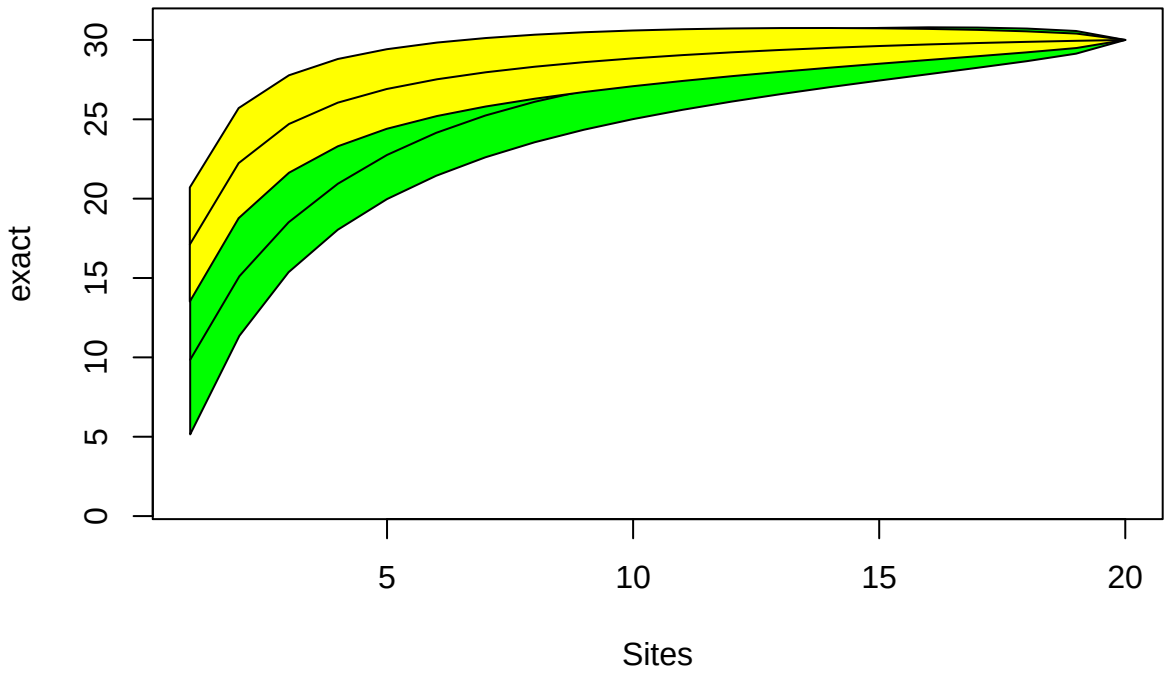

```
#BUT: incorrect to compare the diversities of sampling units having different sizes


# 1.2 Rarefied sp. richness (expected number of species in a standardized sampling size)
rar <- rarefy(dune, sample = min(rowSums(dune)))
barplot(sort(rar))
```




```
plot(x = dune.env$Management, y = rar)
```

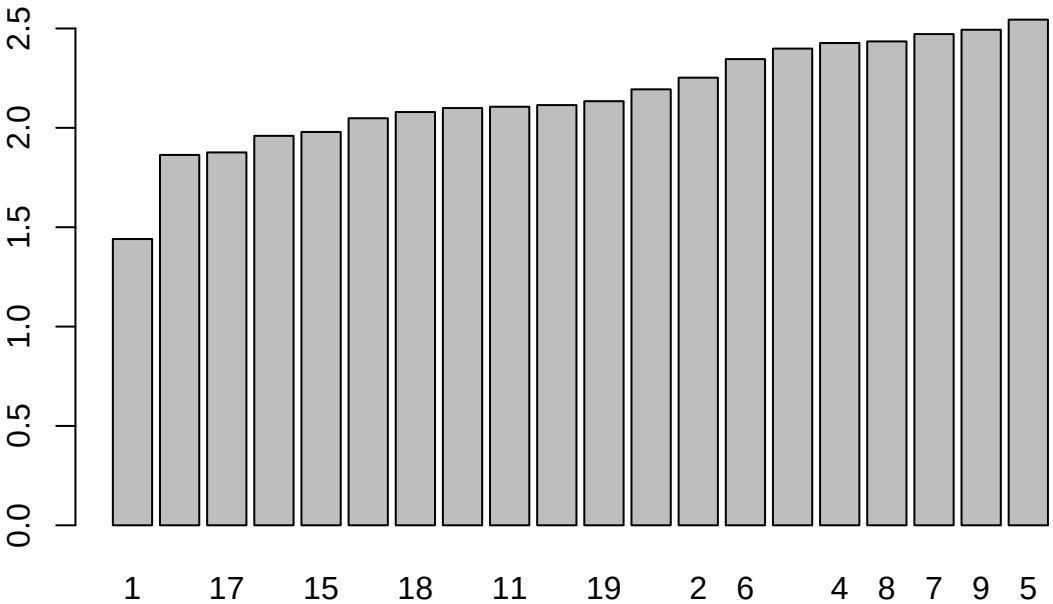



```
# 1.3 Species accumulation curves (to assess sufficient sampling)
spac<-specaccum(dune)
plot(spac, ci.type = "polygon", ci.col = "green")

# 1.4 Rarefied species accum curves
spac.rar <- specaccum(dune, method = "rarefaction")
plot(spac.rar, ci.type = "polygon", ci.col = "yellow", add=TRUE)
```

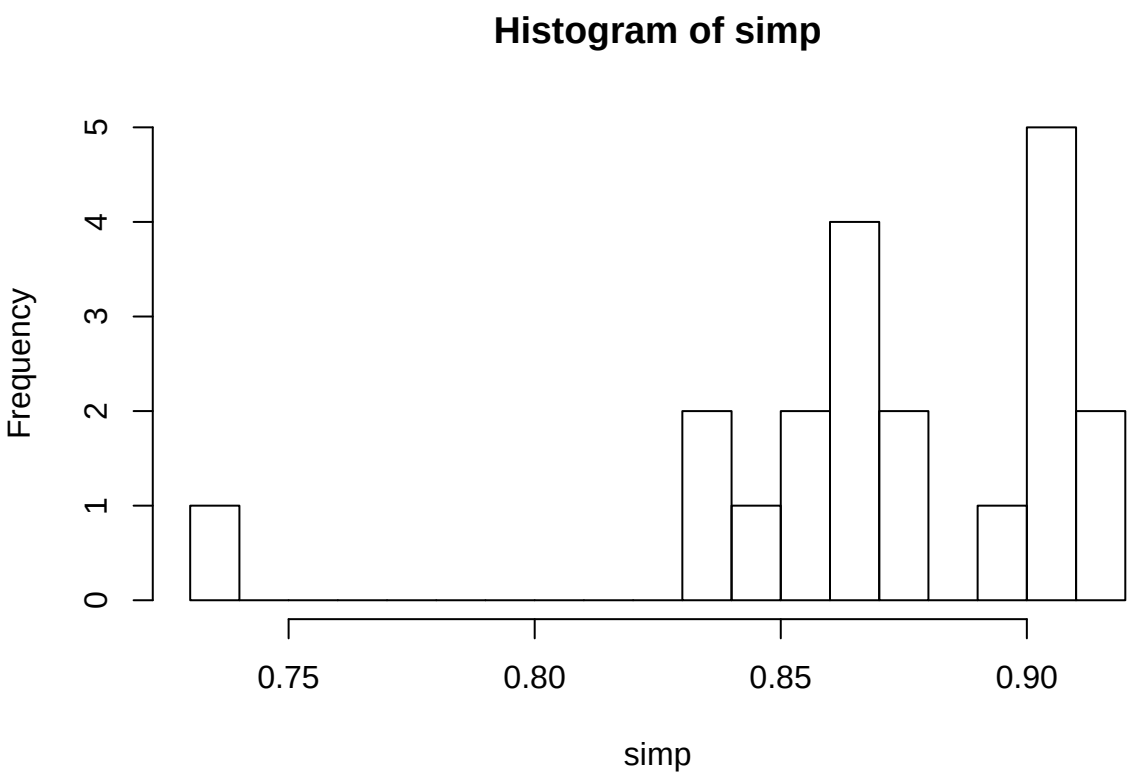



```
#method = "rarefaction" finds the expected species richness and its
#standard deviation by sampling individuals instead of sites.
#It achieves this by applying function rarefy() with number of individuals
#corresponding to average number of individuals per site.
```

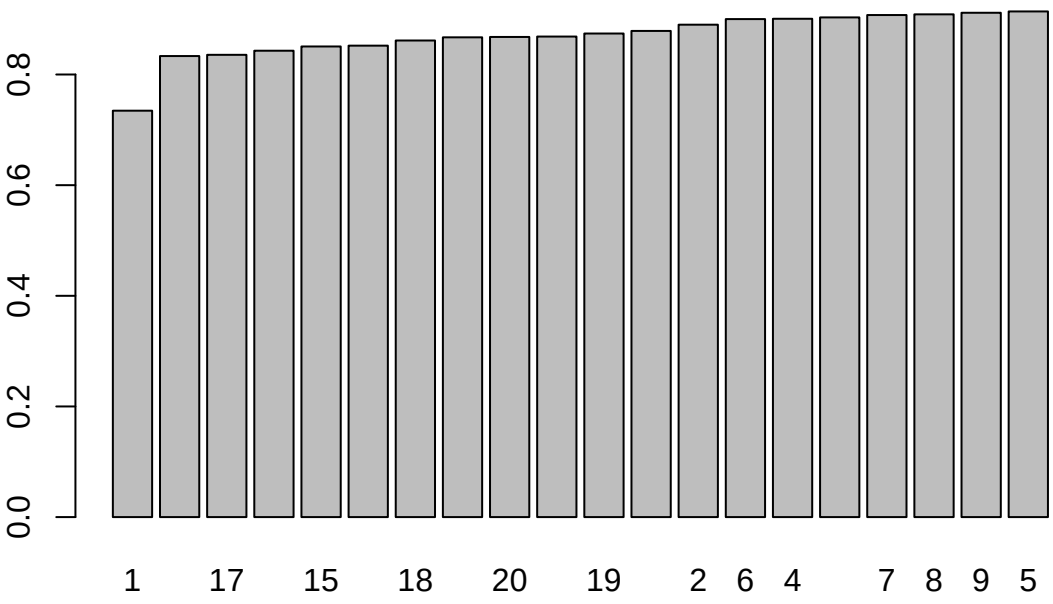
```r
### 2. Entropy of order a = 1  ###

# 2.1 Shannon entropy - considers both species richness and shape of distribution
# UNCERTAINTY about the identity of an organism chosen at random in a sampling unit
shan <- diversity(dune, index = "shannon")
barplot(sort(shan))
```



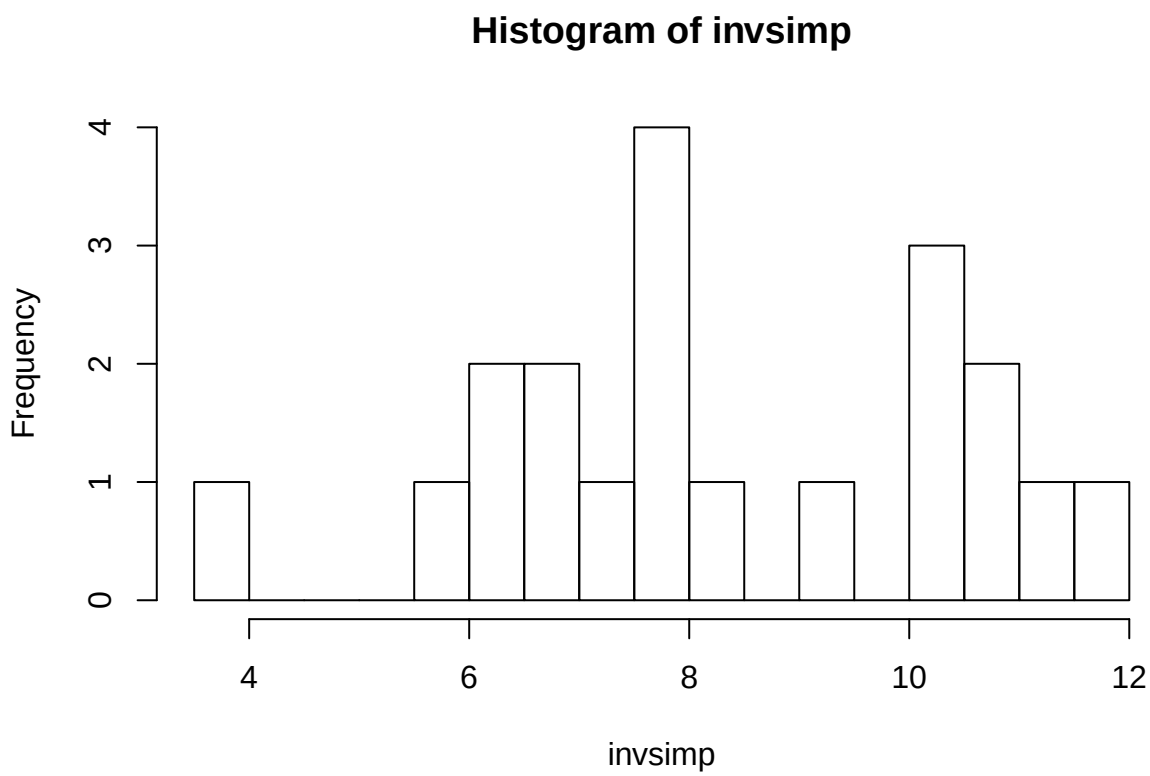

```r
# H = 0 when only one species
# H is low when few dominant species

### 3. Entropy of order a = 2
# equation: 1 - (PROBABILITY that two species belong to the same species)
# LEAST sensitive to rare species

# 3.1 Simpson's

simp <- diversity(dune, index = "simpson")
hist(simp, breaks = 15)
```



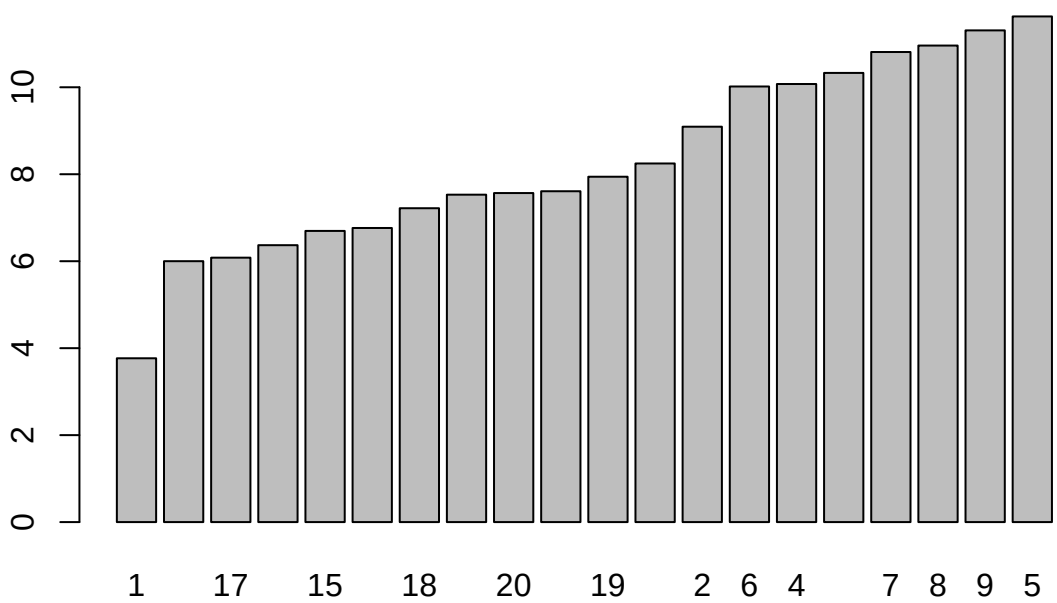

```r
barplot(sort(simp))
```




```r
#Sensitive to abundance of dominant species

# 3.2 Inverse Simpson's
invsimp <- diversity(dune, index = "invsimpson")
hist(invsimp, breaks = 12)
```




```r
barplot(sort(invsimp))
```




```r
#Less sensitive to abundance of dominant species



### 4. Evenness ###

# 4.1 Pielou's evenness J
```

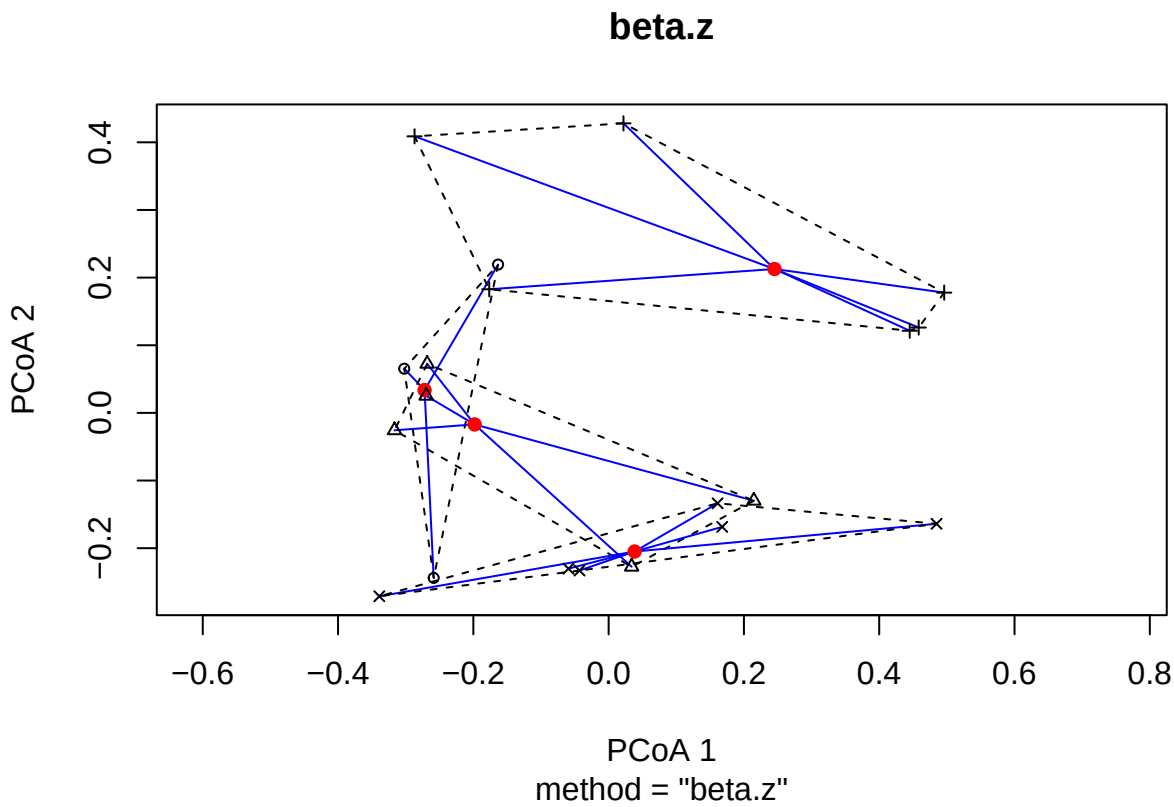
```r
J <- shan/log(specnumber(dune))



### 5. Beta diversity (alpha = diversity within sites, gamma = overall diversity)

# 5.1 Simplest Beta diversity
beta1 <- ncol(dune)/mean(specnumber(dune)) - 1
#problematic because ncol increases with the number of sites even when sites
#are all subsets of the same community.

# 5.2 Pairwise beta diversity
beta2 <- vegdist(dune, binary=TRUE)
mean(beta2)
```

```
## [1] 0.5519907
```

```r
# from pairwise comparison of sites.

# 5.3 Alpha, beta, gamma diversity all together using any diversity index
adipart(dune, index = "simpson")
```

```
## adipart object
##
## Call: adipart(y = dune, index = "simpson")
##
## nullmodel method 'r2dtable' with 99 simulations
## options:  index simpson, weights unif
## alternative hypothesis: statistic is less or greater than simulated values
##
##         statistic     SES     mean     2.5%      50%  97.5% Pr(sim.)
## alpha.1  0.870155 -21.389 0.916733 0.911932 0.916978 0.9209     0.01 **
## gamma    0.945244   0.000 0.945244 0.945244 0.945244 0.9452     1.00
## beta.1   0.075089  21.389 0.028511 0.024329 0.028266 0.0333     0.01 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```r
# MORE COMPLEX: Dispersion-based beta diversity (cluster and many others)
z <- betadiver(dune, "z")
beta.z <- betadisper(z, group = dune.env$Management)

plot(beta.z)
```




```r
TukeyHSD(beta.z) # test differences in dispersion (diversity)
```

```
##   Tukey multiple comparisons of means
##     95% family-wise confidence level
##
## Fit: aov(formula = distances ~ group, data = df)
##
## $group
##               diff         lwr       upr     p adj
## HF-BF -0.05682197 -0.40451674 0.2908728 0.9651033
## NM-BF  0.13255956 -0.20409445 0.4692136 0.6790704
## SF-BF  0.05546893 -0.28118508 0.3921229 0.9642929
## NM-HF  0.18938153 -0.09891174 0.4776748 0.2751871
## SF-HF  0.11229091 -0.17600236 0.4005842 0.6862054
## SF-NM -0.07709062 -0.35196747 0.1977862 0.8523161
```