# Student Exercises | Statistics for Genomics

*your name here*

## Statistics for Genomics Lab

### *Exercise 3.1: Examining Binomially Distributed Data*

1. Using R, simulate the following experimental data:

- use the `rbinom()` function to create a new data set
- look at the arguments to see how they relate to the trials, probability of each trial, and number of replicates using tab complete or help- then assign each argument based on 20 observations of a fair coin toss
- use the `hist()` function to plot your simulated data
- what do the y- and x-axes represent?
- what is the most common outcome, in terms of the number of "successes" per trial? Does this make sense?

2. Now, vary the `probability` argument to reflect an unfair coin toss
3. Now, change your script to do the following:

- perform 200 trials for each of the 100 replicates
- perform 2000 trials for each of the 100 replicate
- how do the distributions change when you go from 20 to 200 to 2000 trials?
- what happens to the curve as your probability of "success" gets smaller or larger than 0.5?

---

### *Exercise 3.2: Examining Normally Distributed Data*

1. Simulate a population of 10,000 individual values for a variable x with a mean of 50 and a sd of 5.5.

- From your simulated data take 1000 random samples of size 20, take the mean of each sample, and plot the distribution of these 1000 sample means using the for loop below.

```
x_sample_means <- NULL
for(i in 1:1000){
x_samp <- sample(x, 20, replace=FALSE)
x_sample_means[i] <- mean(x_samp)
}
```

2. For one of your samples, transform the observation values to z-scores (hint- you can use another `for` loop for this)

- plot the distribution of the z-scores, and calculate the mean and the standard deviation.

3. Now, create a second population (called x.lognorm) by log-transforming all 10,000 values from population "x".

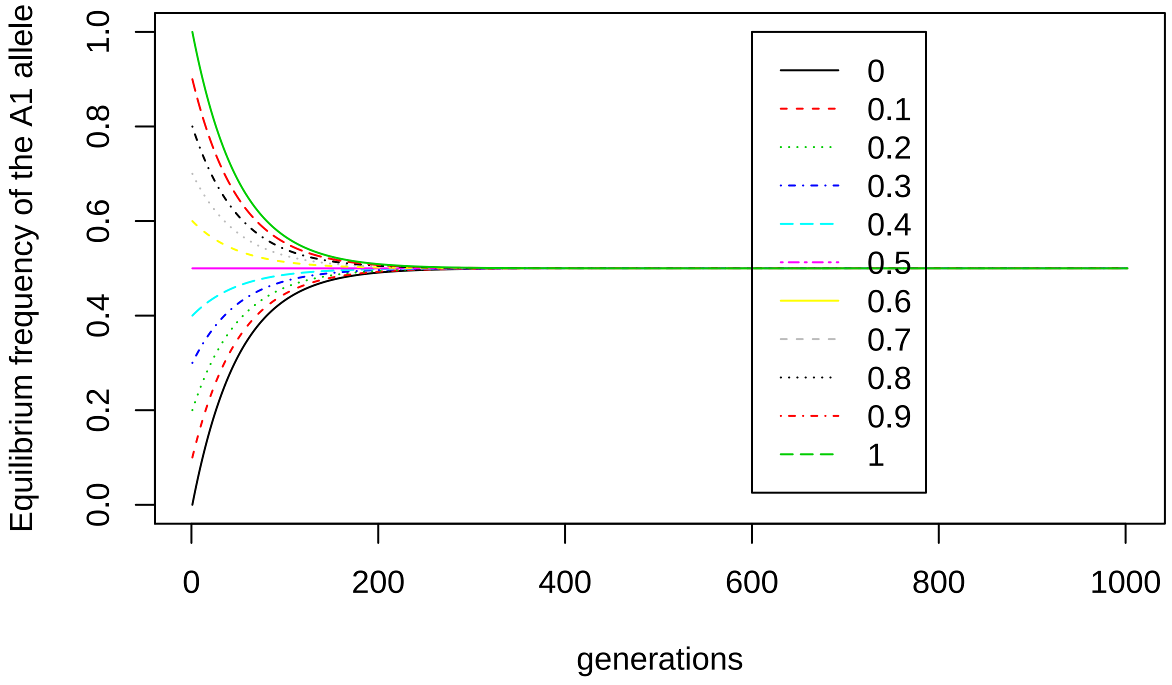
- plot the histogram of these data- how is it different?

---

### *Exercise 3.3: An example of creating your own function to model forward and reverse mutation equilibrium*

Populations of organisms constantly experience mutations, often in opposing directions for the same genes. For example, some mutations may create a new function, whereas others destroy that function. The relative rates of those mutations will determine the equilibrium frequencies of the wild type and mutant alleles in a population. Below is a function, defined using `R`, to calculate that equilibrium frequency over time. The values for the different forward and reverse mutations, starting frequencies, and generations for the simulation are defined in the first script. Change those values in the last line to see how they affect the equilibrium frequency in the population.

```r
forward_reverse_mutation <- function(forward = 0.00, reverse = 0.00, start_freq = 0.5, generations = 10

{
    freq <- start_freq + (1-start_freq)*forward - start_freq*reverse
    freq_array = NULL
    freq_array <- rbind(freq_array, start_freq, freq)

    for (j in 1:generations) {
        freq <- freq + (1-freq)*forward - freq*reverse
        freq_array <- rbind(freq_array, freq)
            }

matplot (freq_array, xlab="generations", ylab="Equilibrium frequency of the A1 allele", ylim=c(0:1.0), 
legend(600, 1, start_freq, col = 1:20, lty = 1:20)

}

forward_reverse_mutation(forward = 0.01, reverse = 0.01, start_freq = seq(0.0, 1.0, by = 0.1), generati
```



1. How does the relative magnitude of forward and reverse mutations affect your final allele frequency once your population reaching equilibrium?

- What if the rates are equal? What if they are largely different?

2. How does the magnitude of your mutation rates affect time to equilibrium?

- What happens if they are very small? Very large?

3. How does the start frequency of A1 allele factor into both time to equilibrium and magnitude of frequency change when your mutation rates are equal? What about when they are not equal?

---

## *Exercise 3.4: Parametric t-test in R*

1. Using R to make a dummy data set that contains one continuous and one categorical value with two levels. To do so draw individuals of the two different factor levels from normal distributions with slightly different means but equal standard deviations. Take 100 observations per factor level.

2. Now, perform a t-test to see if the two populations are statistically different from one another (see below)

```
boxplot(continuous_variable~cat_variable, dataset name)
t.test(continuous_variable~cat_variable, dataset name)
```

3. Repeat steps 1 and 2 above but use different sample sizes (e.g. 10 , then 100, then 1000, then 10,000 obsv. per factor level). How does the sample size affect the statistics?
4. What if you run the test again and make the means of the categorical groups more dissimilar?
5. What happens if you change the standard deviation of your normal distributions to be very small or very large?

---

## *Exercise 3.5: Test for Hardy Weinberg Equilibrium in a two-allele model using a Chi-square statistical distribution*

Another common statistical scenario is to test whether counts in different categories of variables are random or not. A good example is the combination of alleles in to genotypes in a diploid, sexually reproducing population. This so-called Hardy-Weinberg Equilibrium (HWE) expectation is simply a statement that the choice of each allele is an independent random event. Therefore, the null expectation under random mating and lack of strong selection is that the genotypes will be predicted by the binomial sampling of alleles based upon their frequencies. We can test against these expectations using the Chi-square statistic. Below modify the counts of genotypes to see how this affects your conclusion as to whether the population is in HWE.

```
# initial genotype frequencies
AA_counts <- 11
Aa_counts <- 43
aa_counts <- 0
```

```
# total number of alleles, allele frequencies and probabilities
num.alleles <- 2*sum(AA_counts,Aa_counts,aa_counts)
num.a.alleles <- 2*aa_counts + Aa_counts
num.A.alleles <- 2*AA_counts + Aa_counts
p <- num.A.alleles / num.alleles
q <- num.a.alleles / num.alleles
```

```
# Expected genotype probabilities in the F1
expected.AA <- p^2
expected.Aa <- 2 * p * q
expected.aa <- q^2
# Expected genotype frequencies in the F1
E.AA <- expected.AA * num.alleles/2
```

```
E.Aa <- expected.Aa * num.alleles/2
E.aa <- expected.aa * num.alleles/2
```

```
# Calculate the chi squared statistic
chi.sq <- ((AA_counts-E.AA)^2 - E.AA) / E.AA  +
          ((Aa_counts-E.Aa)^2 - E.Aa) / E.Aa  +
          ((aa_counts-E.aa)^2 - E.aa) / E.aa
print(chi.sq)
```

```
## [1] 20.63219
```

- You've calculated the observed Chi-square value above
- To calculate the p-value associated for this statistic with 2 degrees of use the equation below with the observed chi-square
- Significant p-values ($< 0.05$) mean that the population deviates from HWE

```
1- pchisq(chi.sq,df = 2)
```

```
## [1] 3.309611e-05
```

1. Based on these result, do you reject or fail to reject the null hypothesis?

- what does this tell you about the population in question?
- is there a genotype that appears to be under selection?

---

## *Exercise 3.6: Linear Models to examine the relationship of different hormone levels and thyroid morphology in zebrafish*

Write a code chunk to read in the 'perchlorate2.csv' data set in the data folder. This file includes data from a test in which multiple strains of zebrafish were exposed to multiple concentrations of the toxic compound perchlorate, which interferes with thyroid development and function. Several outcomes of fish thyroid morphology and hormone concentration were measured. The data set includes the following variables in this order:

- Strain of zebrafish (categorical)
- Perchlorate_Level (continuous)
- T4_Hormone_Level (continuous)
- Follicle_Area (continuous)
- Number_of_Follicles (count)
- Age_dpf (continuous)
- Age_Category (categorical)
- Testes_Area (continuous)
- Testes_StageNow (categorical)

1. Modify the code below to build a linear model of two **continuous** variables. You can check for variable class using `str`.

- What is the response? What is the predictor?
- Notice how the output of the linear model is specified to a new object.

```
my_lm <- lm(XXX ~ YYY)
```

- Now look at a summary of the linear model

```
summary(my_lm)
print(my_lm)
```

2. Now let's produce a regression plot

```r
plot(XXX~YYY, col = "blue")
abline(my_lm, col = "red")
```

- What does the line represent?

3. You can use the `$` operator to select out parameter values and estimates from your model object. Plot the residuals over the fitted values by extracting them from your model object.

- Are there any obvious patterns in this plot?
- What happens when you `plot` the linear model object itself?
- Using these plots, does it appear that you have violated any assumptions?

---

## *Exercise 3.7: Multiple linear regression*

1. Using the perchlorate dataset, create a multiplicative multiple linear regression model using the `lm` function that examines the affects of perchlorate level and age on one of our phenotypic outcomes.

- What can you conclude form the output?
- Is there an interactive affect of perchlorate and age? What does that tell you biologically?

2. Plot the residuals of this model over the fitted values.

- Any problems? Why?
- Are your predictors truly continuous? So are we using the right test...?

---

## *Exercise 3.8: 2-by-2 fixed effect factorial ANOVA examining the effects of microbiome treatment on RNA expression levels in the guts of threespine stickleback*

Load the RNAseq_microbiome.csv dataset which represents RNAseq data collected for three different genes (FGF3, Sox10 and SLC5) from the guts of threespine stickleback fish. In this experiment two different genotypes of stickleback (ocean and freshwater) were tested by subjecting each to two different microbiome treatments (conventional or mono-associated). Stickleback can be made germ free and then either exposed to the conventional microbiome or just one microbe. After exposure the guts were extracted, RNA isolated, and RNAseq data collected. This data set has just three genes.

- Is `R` correctly interpreting the class of your variables? It is always good to check this before doing anything.

1. Make boxplots to graphically illustrate differences in gene expression distributions among genotype and microbiota treatment for each gene.

- Hint: You may need to create a new variable for this.
- Any obvious patterns?
- Would you guess there is a difference among means?
- Does it look like there may be an interaction?

2. Fit the factorial linear model - two different ways to do the same thing- then use the summary function to look at your results.

```r
rna_aov <- aov(gene ~ microbiota + genotype + microbiota:genotype)
rna_aov <- aov(gene ~ microbiota*genotype)
```

3. Is there a significant interaction between genotype and treatment?

- Fit the reduced, additive model
- Bonus: You can use the `anova` function to compare the fit of your models, to test if the interaction should be included or if the reduced model is sufficient. A significant *P-value* for this analysis of deviance test implies that the model fit is significantly different among models. Order your model objects by increasing complexity, that way you can test if the addition of the interaction term significantly improves the fit of your model. Based on this test, does the interaction significantly improve the model fit?

---

## *Exercise 3.9: NMDS for the analysis of multivariate data*

Nonmetric Multidimensional Scaling (NMDS) is a form of nonparametric clustering. As such, some of the distributional assumptions of multivariate analyses are relaxed. The package VEGAN was originally derived for ecological data across plots or sites. For example, enumeration of species across different locations. However, the approach is very applicable to genetic data across geographic locals or phenotypes, or gene expression data across different samples.

Now examine the 'yeast.tsv' data set, which is a measurement of RNAseq levels of numerous genes from six different yeast samples from different vineyards.

```
library (vegan)
```

```
## Loading required package: permute
```

```
## Loading required package: lattice
```

```
## This is vegan 2.5-3
```

```
yeast_data <- read.table('data/yeast.tsv', row.names = 1, header = T, sep = '\t')
```

1. We'll first turn the raw data matrix into a dissimilarity matrix for all samples. The `decostand` function is a form of normalization.

```
vare.dis <- vegdist(decostand(yeast_data, "hell"), "euclidean")
#print (vare.dis)
```

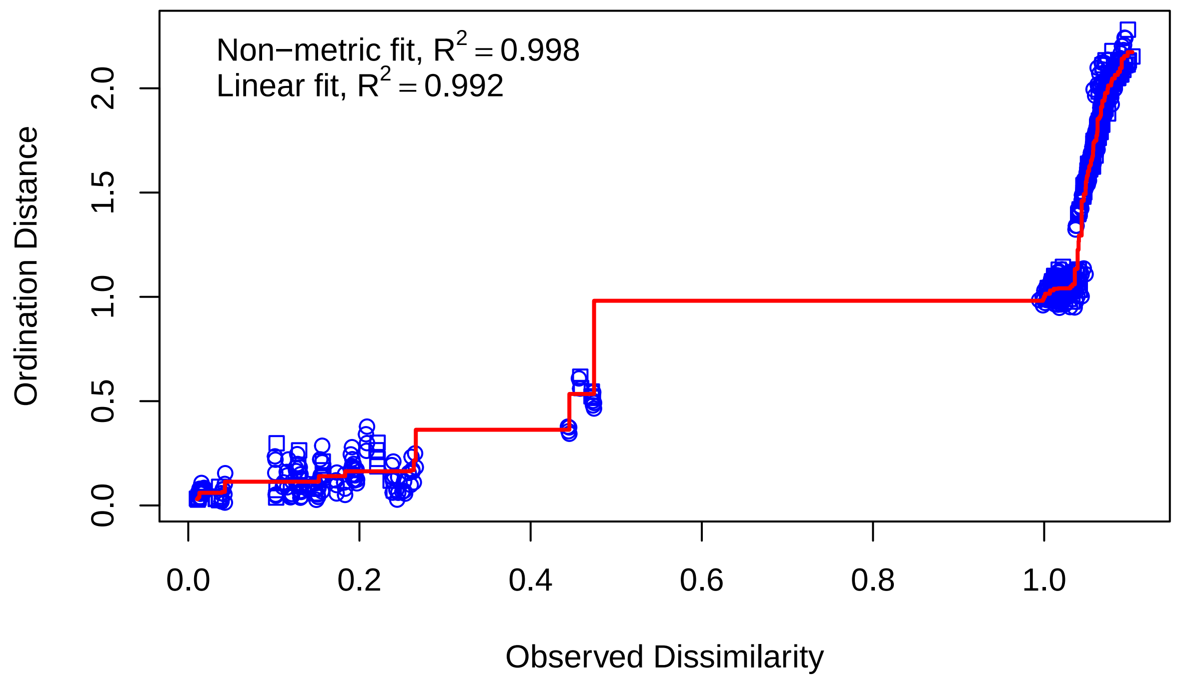
2. Now we'll perform the clustering of the samples using multidimensional scaling. The goal of this is to represent complex data in lower dimensions without losing too much information. Take a look at the 'stress' values of moving from a higher to lower dimensionality of the data. Usually a value of 0.15 or lower is considered acceptable and indicates a good model fit.

```
vare.mds0 <- monoMDS(vare.dis)
print (vare.mds0)
```

```
##
## Call:
## monoMDS(dist = vare.dis)
##
## Non-metric Multidimensional Scaling
##
## 39 points, dissimilarity 'euclidean', call 'vegdist(x = decostand(yeast_data, "hell"), method = "eucl
##
## Dimensions: 2
## Stress:     0.04261236
## Stress type 1, weak ties
## Scores scaled to unit root mean square, rotated to principal components
## Stopped after 95 iterations: Stress nearly unchanged (ratio > sratmax)
```

3. Let's take a look at how the dissimilarities among samples maps onto the ordination distance. Notice that there is a fit with the data, but we're no longer assumed consistent linearity over the entire data set.

```r
stressplot(vare.mds0, vare.dis)
```



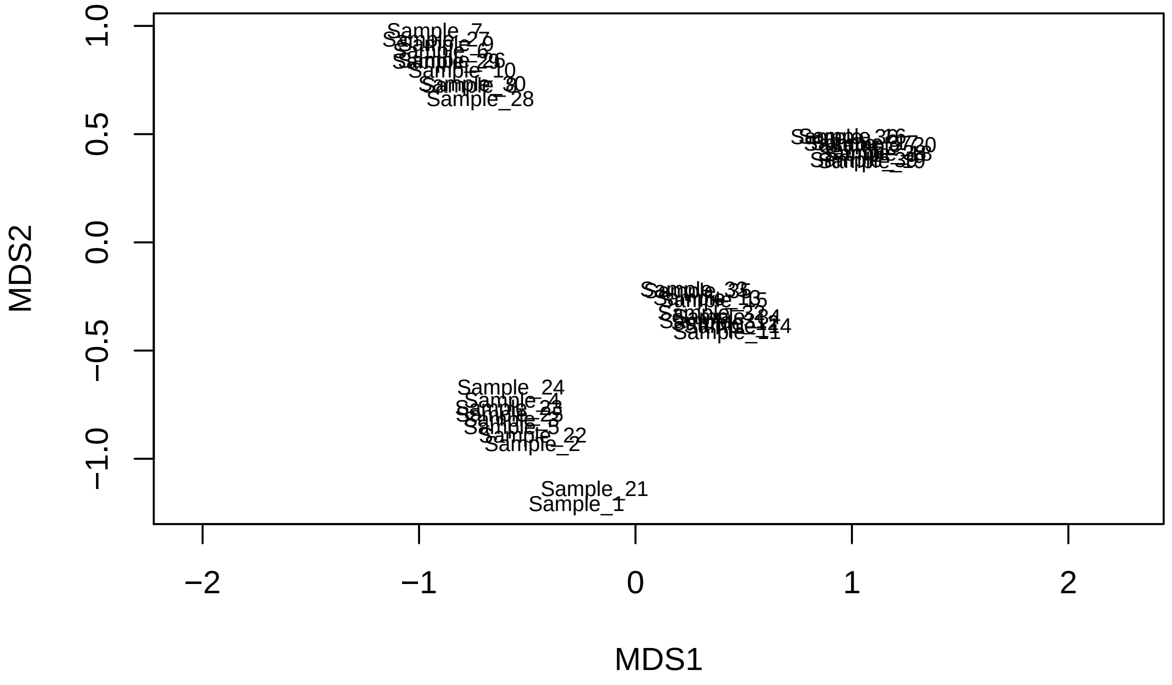
-

What does the R^2 value tell you? Is the model accurately predicting the observed dissimilarity?

4. Now let's look at the grouping of the samples in this lower dimensional space.

```r
ordiplot (vare.mds0, type = "t")
```

```
## species scores not available
```

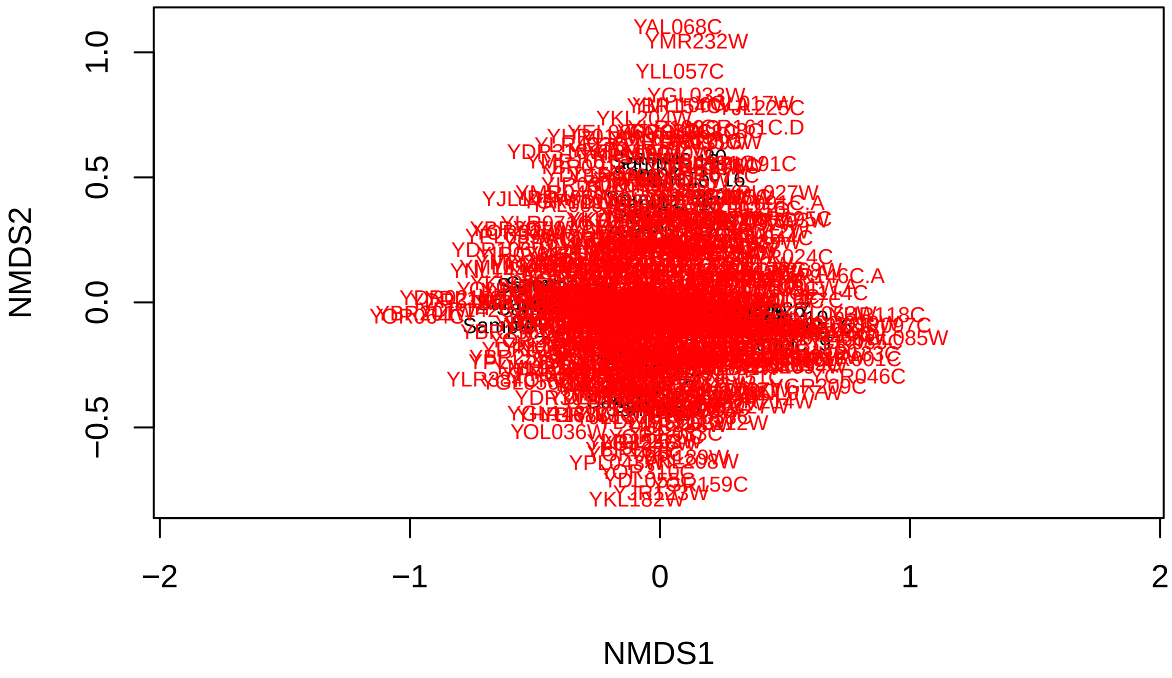


-Any

clustering?

5. Now we can rerun the ordination and add all of the data (genes) as well.

```r
vare.mds <- metaMDS(yeast_data, trace = F)
plot (vare.mds, type = "t")
```



- How does this plot compare to your first plot?

6. We can run a PCA on our data as well.

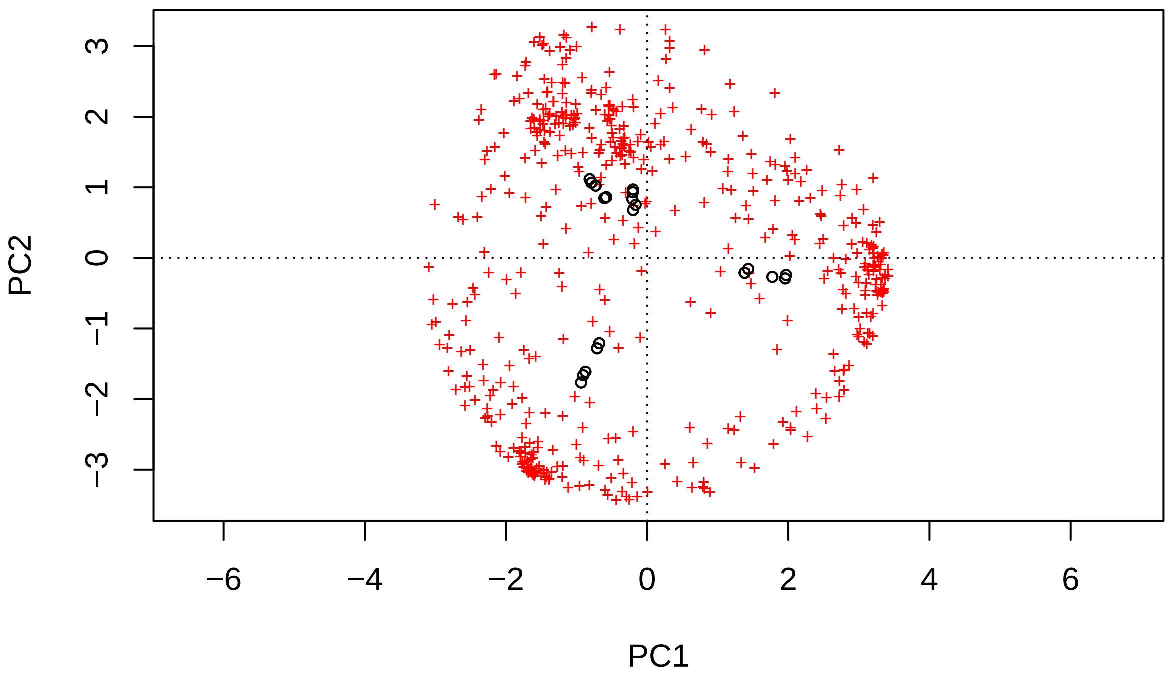
```r
vare.pca <- rda(yeast_data, scale = TRUE)
print (vare.pca)
```

```
## Call: rda(X = yeast_data, scale = TRUE)
##
##                Inertia Rank
## Total              499
## Unconstrained      499   20
## Inertia is correlations
##
## Eigenvalues for unconstrained axes:
##    PC1    PC2    PC3    PC4    PC5    PC6    PC7    PC8
## 151.62 144.97 140.56  16.77   7.82   6.80   6.66   5.29
## (Showing 8 of 20 unconstrained eigenvalues)
```

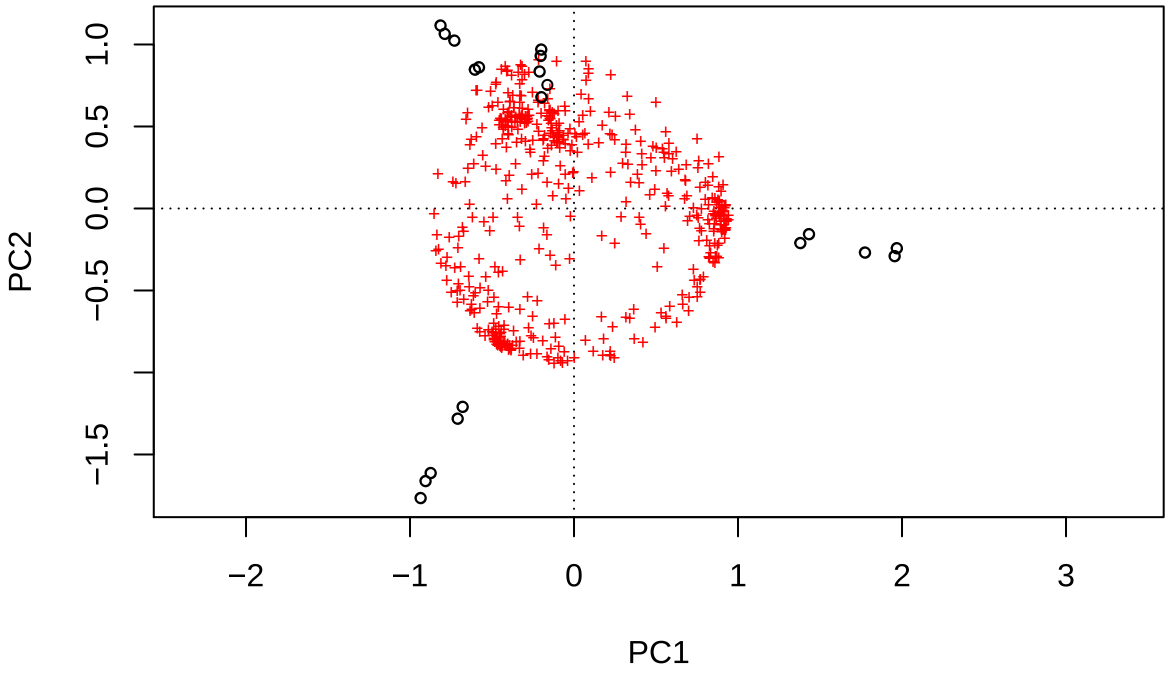
- What do you notice about the eignevalues of the PCs?

Showing both the locations of the samples. Try both plots.

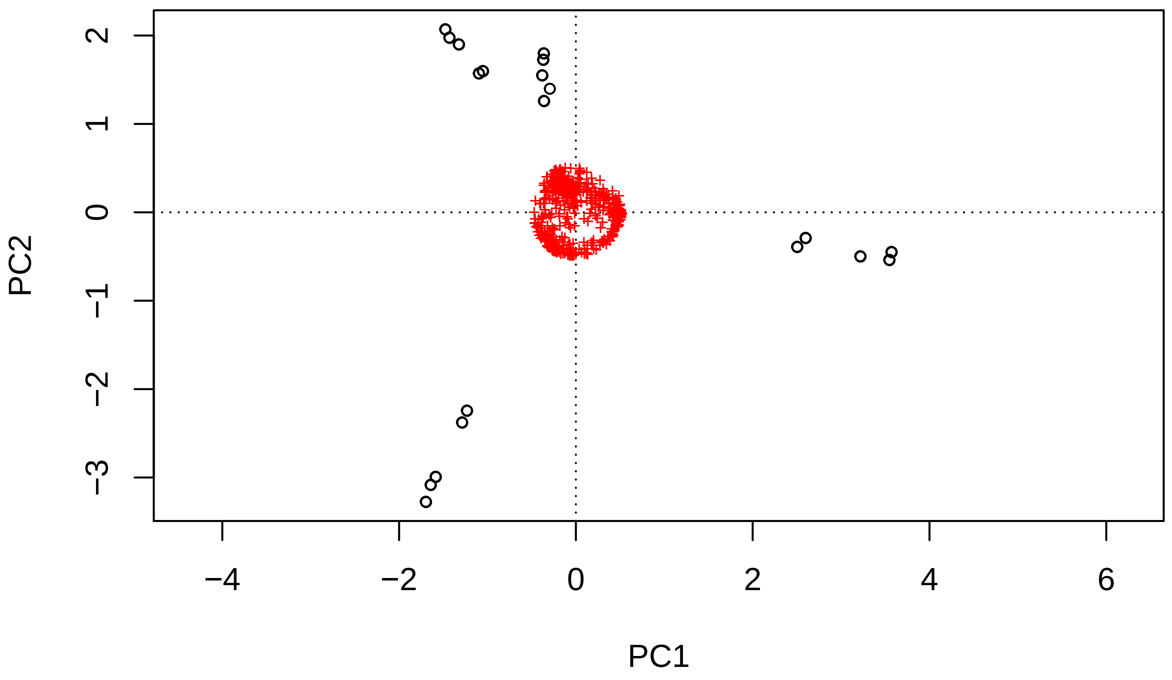
```r
plot (vare.pca, scaling = -1)
```



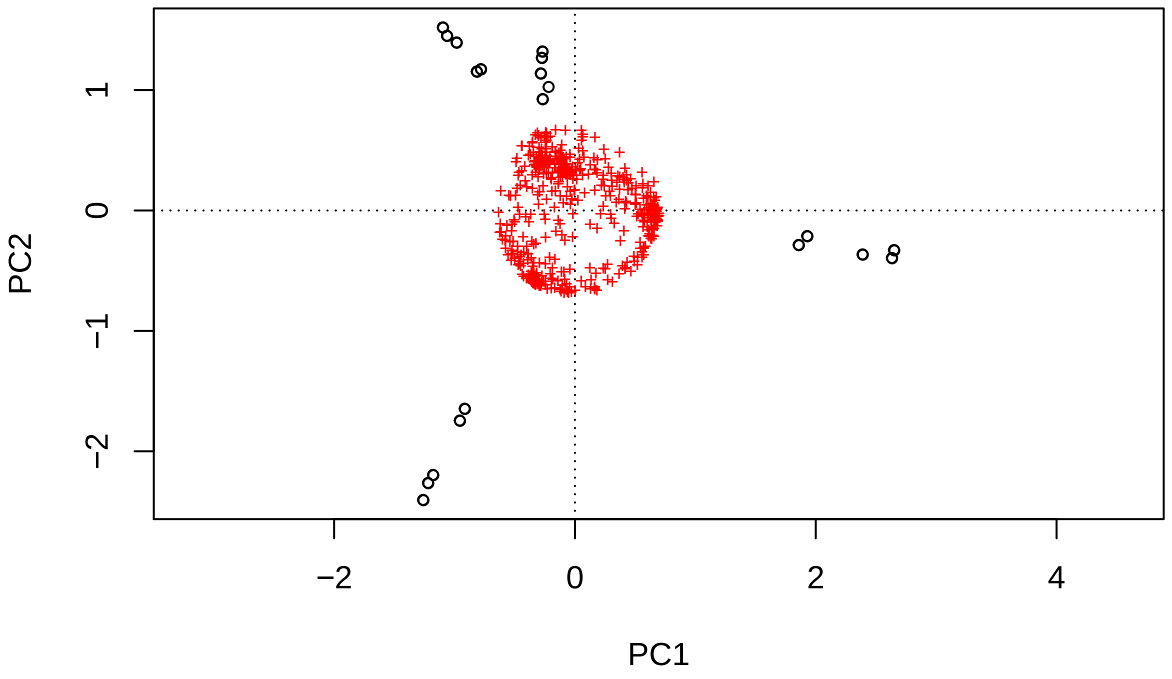
```
plot (vare.pca, scaling = 1)
```



```
plot (vare.pca, scaling = 2)
```



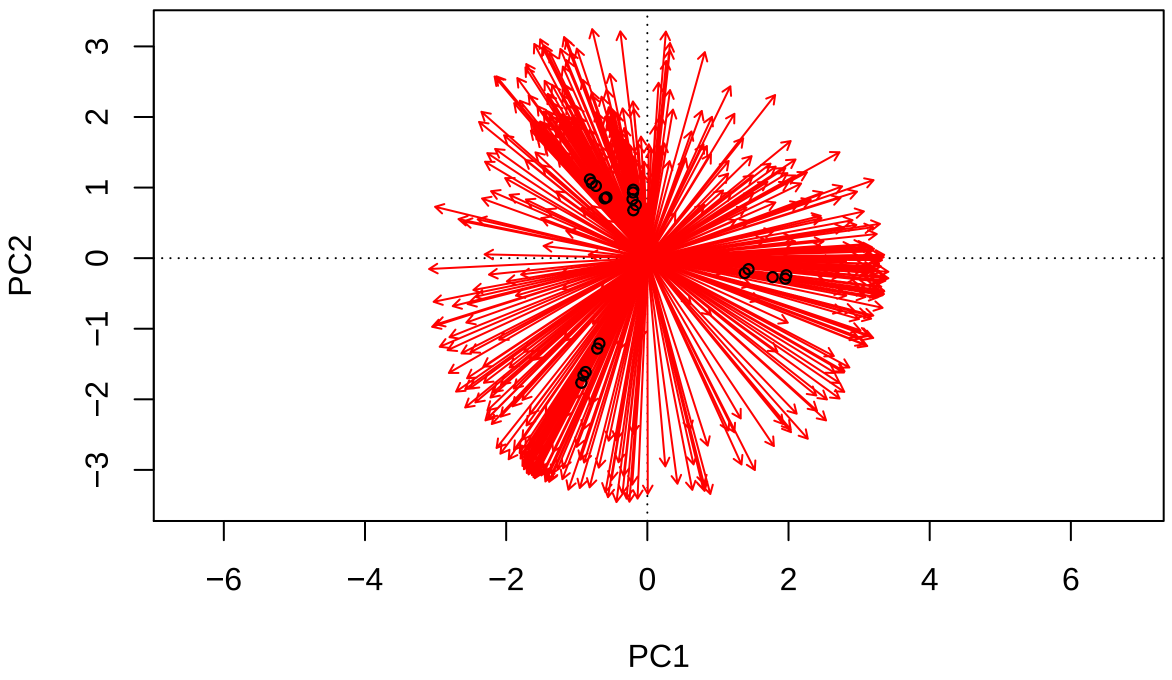
```r
plot (vare.pca, scaling = 3)
```



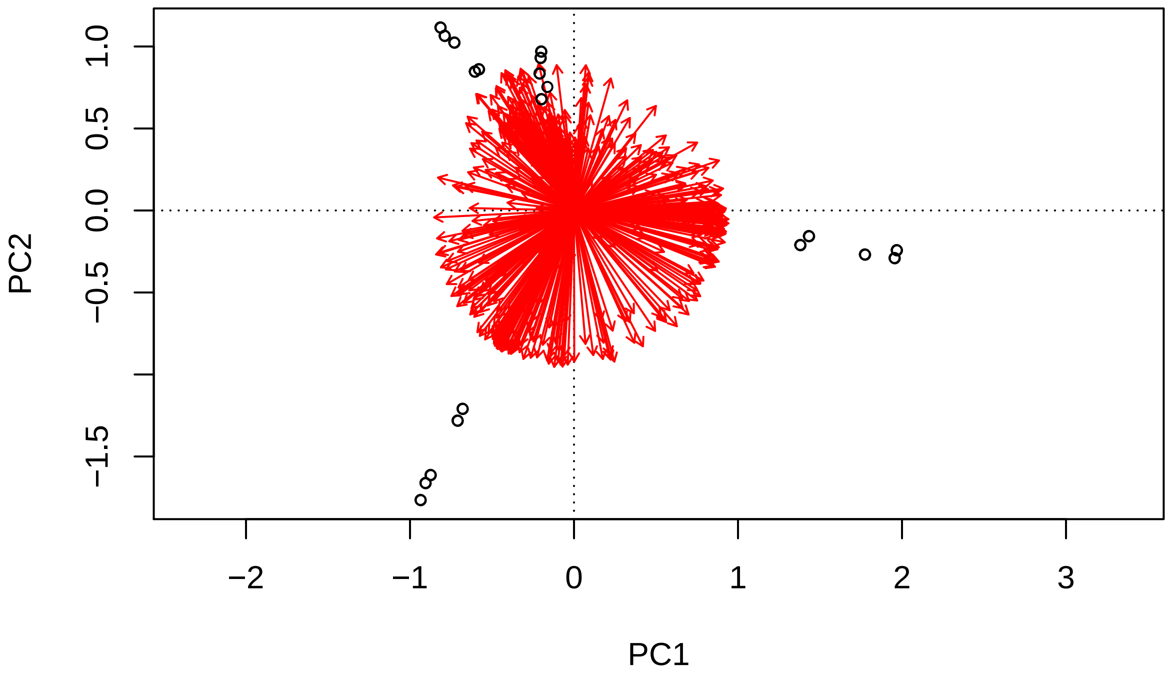
- What are these plots showing? What is in red? What is in black?

And the weighting of the original variables (expression level) on the PCs

```r
biplot (vare.pca, scaling = -1)
```



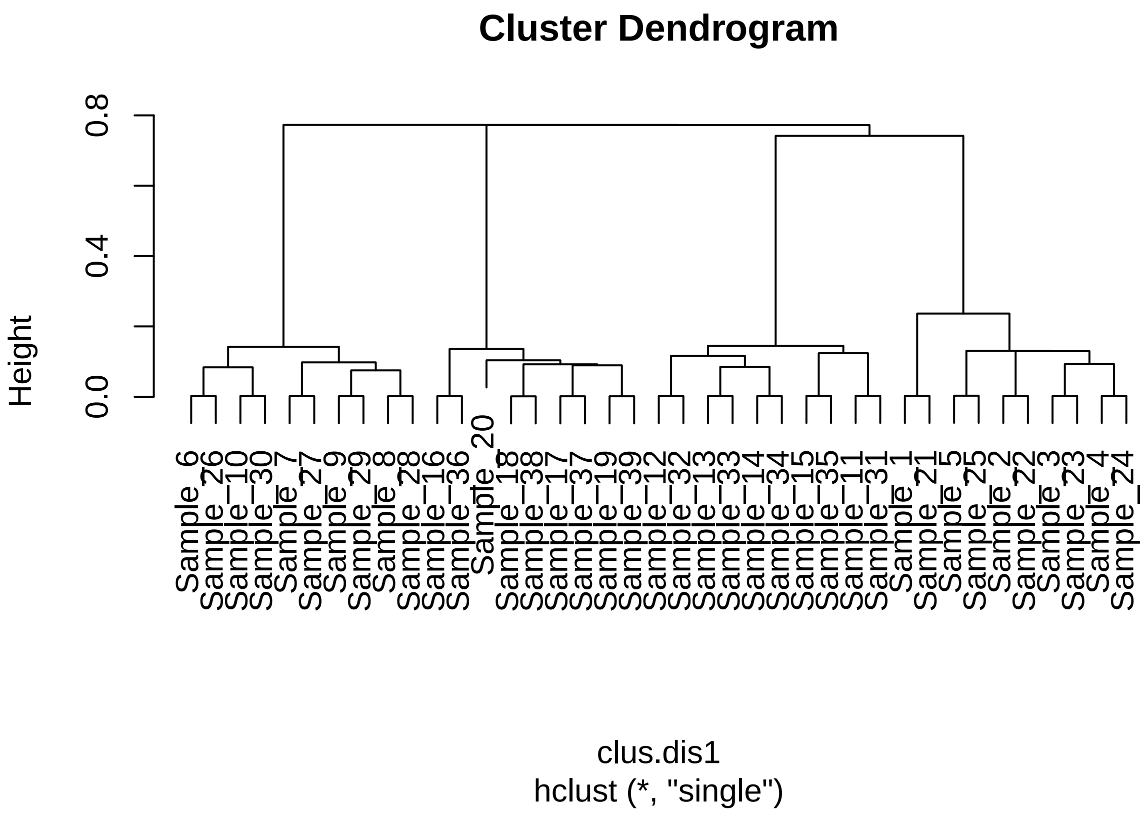
```
biplot (vare.pca, scaling = 1)
```



7. Lastly, we can use the dissimilarity matrices to perform hierarchical clustering. Try both the non-normalized (clus.dis1) and normalized (clus.dis2) distances.

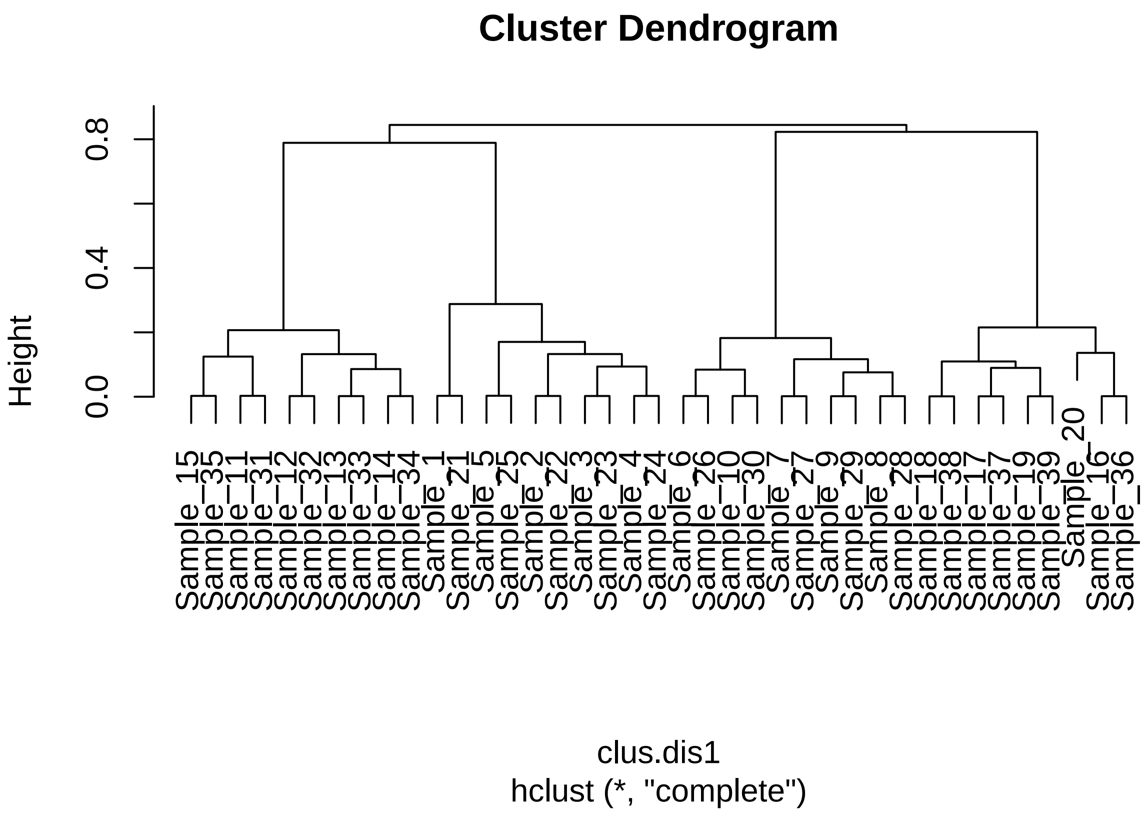
```
clus.dis1 <- vegdist(yeast_data)
clus.dis2 <- vegdist(decostand(yeast_data, "hell"), "euclidean")
```

```
cluster1 <- hclust(clus.dis1, "single")
plot(cluster1)
```

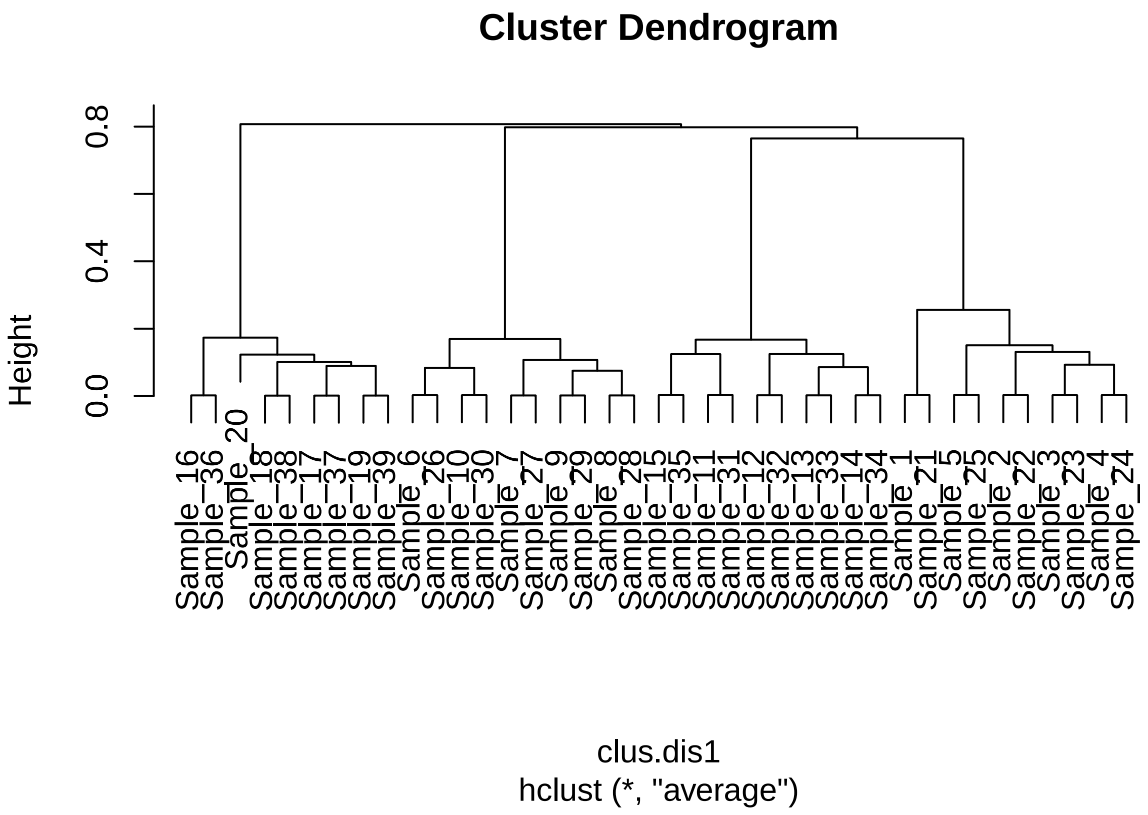


Now, try these different versions of clustering. What is different about them?

```
cluster_complete <- hclust(clus.dis1, "complete")
plot(cluster_complete)
```



```r
cluster_average <- hclust(clus.dis1, "average")
plot(cluster_average)
```



7. Lastly, let's ask R to cut the tree into several clusters for us. I've written it as three. Try it with different numbers of clusters and the different types of clustering from above.

```
grp <- cutree(cluster1, 4)
print (grp)
```

```
##  Sample_1  Sample_2  Sample_3  Sample_4  Sample_5  Sample_6  Sample_7
##         1         1         1         1         1         2         2
##  Sample_8  Sample_9 Sample_10 Sample_11 Sample_12 Sample_13 Sample_14
##         2         2         2         3         3         3         3
## Sample_15 Sample_16 Sample_17 Sample_18 Sample_19 Sample_20 Sample_21
##         3         4         4         4         4         4         1
## Sample_22 Sample_23 Sample_24 Sample_25 Sample_26 Sample_27 Sample_28
##         1         1         1         1         2         2         2
## Sample_29 Sample_30 Sample_31 Sample_32 Sample_33 Sample_34 Sample_35
##         2         2         3         3         3         3         3
## Sample_36 Sample_37 Sample_38 Sample_39
##         4         4         4         4
```

---

### *Exercise 3.10: Biodiversity analysis in R - pulling it all together*

This last exercise comes to us thanks to the hard work of Steven Kembel, who is a faculty member at the University of Montreal in Canada. Steve was a postdoctoral scholar at the University of Oregon and put together this very nice exercise to demonstrate how to pull numerous different R packages and scripts together into an integrated data analysis pipeline. The purpose of this particular script is to examine the biodiversity of grassland plants in ecological communities in Canada using several common ecological statistics.

### General background

In this workshop we are going to analyze a data set on the biodiversity of grassland plants in Alberta. This data set consists of data on the occurrence of grassland plants at several different sites in Alberta, along with information on their functional traits and phylogenetic relationships.

Dr. Kembel described this data set in more detail in the following paper: S.W. Kembel and J.F. Cahill, Jr. 2011. Independent evolution of leaf and root traits within and among temperate grassland plant communities. PLoS ONE 6(6): e19992. (doi:10.1371/journal.pone.0019992).

## Getting everything set

To begin, make sure you have loaded the picante package and the workspace image that contains all of the data files by running the following commands.

```
load("R_biodiversity_workspace.RData")
```

## Getting biodiversity data into R

The first thing we need to do is import all the data we need into R.

We will want to make sure the different packages we are going to use are loaded. We will be using functions from the **ape**, **picante**, and **vegan** packages today. Since **picante** depends on the other two packages, loading it will load the other two as well.

```
## Loading required package: ape
```

```
## Loading required package: nlme
```

## Community data

Ecological community data consist of observations of the (relative) abundance of species in different samples. In our case, the abundance measure is percent cover of different plant species in 20x20m quadrats in grasslands in different habitat types.

The format for community data is a data.frame with samples in the rows and species in the columns. Our data are already in this format so we can load them using the following command. Note that since we've set our working directory to the folder containing all the data files, we just have to type the filename.

```
# read community data
# use plot IDs as rownames (first column of data)
# use species names as colnames (default read.csv is header=TRUE)
# replace filename with file.choose() to open interactive window
comm <- read.csv("data/grassland.community.csv", header=TRUE, row.names=1)
```

By reading the data in this way, we have set the species names as the column names, and the sample names as the row names. This is important to note - we didn't load these labels in as data - they are the *names* of the rows/columns. Later this will make it easier for us to link different data sets. Let's check to make sure our rows and columns have reasonable-looking names.

```
class(comm)
```

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

```
# get the dimension of the community object (rows x columns)
dim(comm)
```

```
## [1] 27 76
```

```
rownames(comm)
```

```
##  [1] "mix-O-1"  "mix-O-2"  "mix-O-3"  "mix-O-4"  "mix-O-5"  "mix-O-6"
##  [7] "mix-O-7"  "fes-K-8"  "fes-K-9"  "fes-K-10" "fes-K-11" "fes-K-12"
## [13] "fes-K-13" "fes-K-14" "fes-K-15" "fes-K-16" "fes-K-17" "mix-H-18"
## [19] "mix-H-19" "mix-H-20" "mix-H-21" "mix-H-22" "mix-H-23" "mix-H-24"
## [25] "mix-H-25" "mix-H-26" "mix-H-27"
```

```
head(colnames(comm))
```

```
## [1] "Antennaria_parvifolia"
## [2] "Artemisia_cana"
## [3] "Artemisia_frigida"
## [4] "Symphyotrichum_ericoides_var._ericoides"
## [5] "Bouteloua_gracilis"
## [6] "Carex_filifolia"
```

```
# take a peek at the data (just the first five rows/columns)
comm[1:5, 1:5]
```

```
##         Antennaria_parvifolia Artemisia_cana Artemisia_frigida
## mix-O-1                    10             10                50
## mix-O-2                     0             10                50
## mix-O-3                    20             20                30
## mix-O-4                     0              0                 0
## mix-O-5                     0             10                 0
##         Symphyotrichum_ericoides_var._ericoides Bouteloua_gracilis
## mix-O-1                                      10                 70
## mix-O-2                                      10                 90
## mix-O-3                                      10                 60
## mix-O-4                                       0                  90
## mix-O-5                                       0                 100
```

Each cell contains the percent cover of a species in a sample (there is a file in the 'data' folder that lists all of the names of the different plant species). Many multivariate methods are sensitive to the total abundance in a sample, so we should probably convert these absolute abundance estimates to a relative abundance estimate. We can do this with a function from the **vegan** package.

```
# check total abundance in each sample
apply(comm, 1, sum)
```

```
##  mix-O-1  mix-O-2  mix-O-3  mix-O-4  mix-O-5  mix-O-6  mix-O-7  fes-K-8
##      640      630      710      350      400      650      560      960
##  fes-K-9 fes-K-10 fes-K-11 fes-K-12 fes-K-13 fes-K-14 fes-K-15 fes-K-16
##      960      980      830      980      980      830      640     1080
## fes-K-17 mix-H-18 mix-H-19 mix-H-20 mix-H-21 mix-H-22 mix-H-23 mix-H-24
##      710      440      590      540      340      420      400      600
## mix-H-25 mix-H-26 mix-H-27
##      540      590      420
```

```
# Turn percent cover to relative abundance by dividing each value by sample total abundance
comm <- decostand(comm, method="total")
# check total abundance in each sample
apply(comm, 1, sum)
```

```
##  mix-O-1  mix-O-2  mix-O-3  mix-O-4  mix-O-5  mix-O-6  mix-O-7  fes-K-8
##        1        1        1        1        1        1        1        1
##  fes-K-9 fes-K-10 fes-K-11 fes-K-12 fes-K-13 fes-K-14 fes-K-15 fes-K-16
##        1        1        1        1        1        1        1        1
## fes-K-17 mix-H-18 mix-H-19 mix-H-20 mix-H-21 mix-H-22 mix-H-23 mix-H-24
##        1        1        1        1        1        1        1        1
## mix-H-25 mix-H-26 mix-H-27
##        1        1        1
```

```
# look at the transformed data
comm[1:5, 1:5]
```

```
##         Antennaria_parvifolia Artemisia_cana Artemisia_frigida
## mix-O-1            0.01562500     0.01562500        0.07812500
## mix-O-2            0.00000000     0.01587302        0.07936508
## mix-O-3            0.02816901     0.02816901        0.04225352
## mix-O-4            0.00000000     0.00000000        0.00000000
## mix-O-5            0.00000000     0.02500000        0.00000000
##         Symphyotrichum_ericoides_var._ericoides Bouteloua_gracilis
## mix-O-1                              0.01562500         0.10937500
## mix-O-2                              0.01587302         0.14285714
## mix-O-3                              0.01408451         0.08450704
## mix-O-4                              0.00000000         0.25714286
## mix-O-5                              0.00000000         0.25000000
```

## Trait data

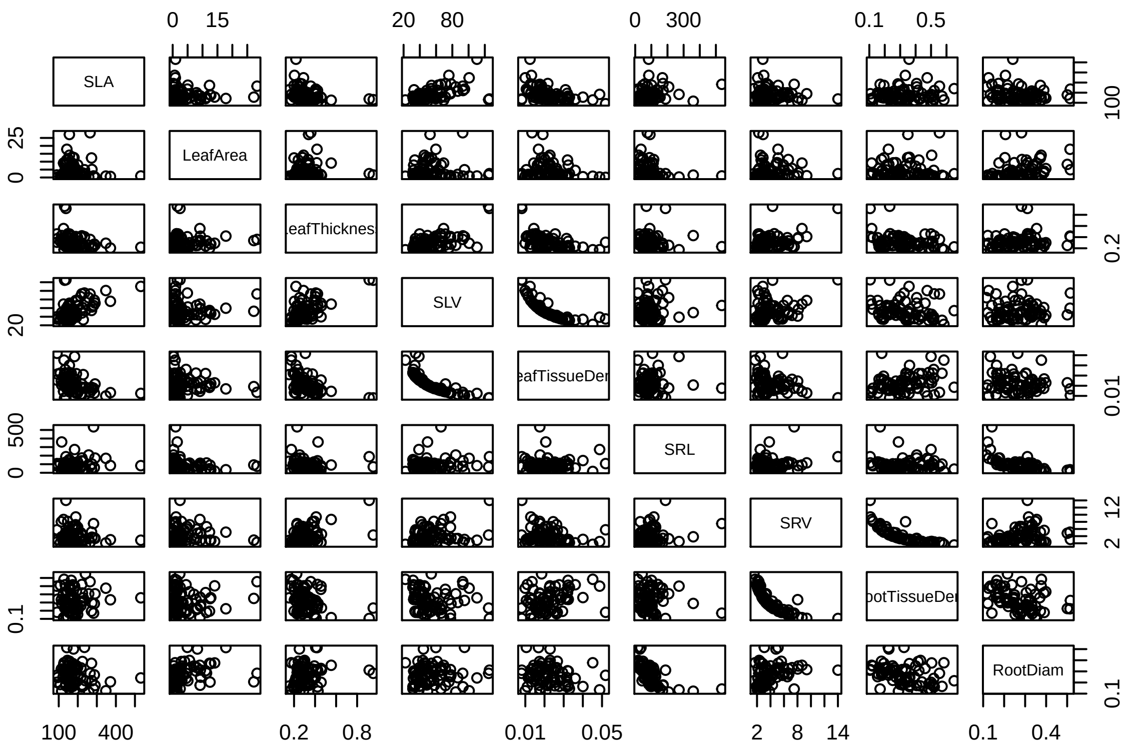
We also have information on the leaf and root traits of each species. We can load these data in the same way as the community data, but now we will have species in the rows and traits in the columns.

```
# replace filename with file.choose() to open interactive window
traits <- read.csv("data/species.traits.csv", header=TRUE, row.names=1)
```

```
# take a peek at the data
head(traits)
```

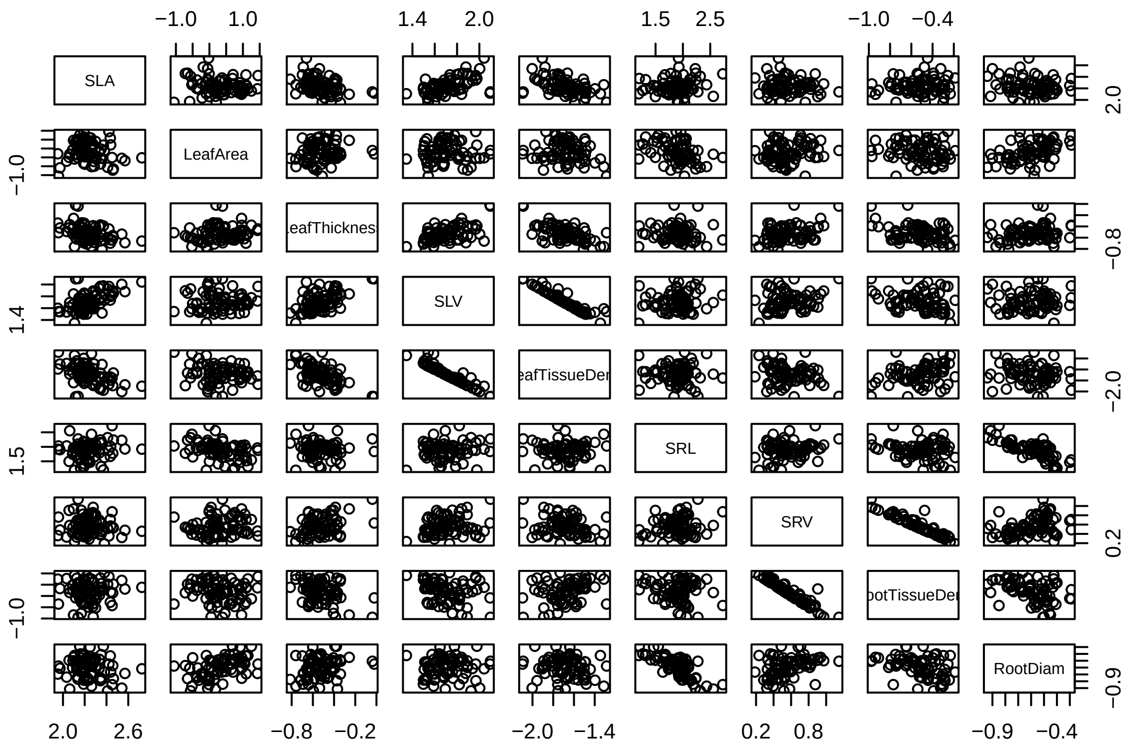
```
##                              SLA  LeafArea LeafThickness       SLV
## Achillea_millefolium    140.2663  9.275390     0.4163333  59.56525
## Allium_textile          137.7006  2.445361     0.9147222 125.69496
## Amelanchier_alnifolia   156.1014 14.064856     0.2900000  45.45227
## Androsace_occidentalis 257.2050  0.274745     0.2535000  84.22189
## Antennaria_neglecta     171.0442  1.731990     0.2810000  48.14442
## Antennaria_parvifolia   193.8718  0.317200     0.2466667  47.64151
##                        LeafTissueDens       SRL       SRV RootTissueDens
## Achillea_millefolium      0.018085428  74.14570  5.038776      0.2553510
## Allium_textile            0.008137136 187.85485 14.013757      0.1049986
## Amelanchier_alnifolia     0.022841232  20.87560  2.518939      0.5039683
## Androsace_occidentalis    0.017706402 207.45582  3.291592      0.4071078
## Antennaria_neglecta       0.020920233 124.73397  6.710526      0.1593750
## Antennaria_parvifolia     0.021048340  44.93859  4.003997      0.2504167
##                         RootDiam
## Achillea_millefolium   0.3123600
## Allium_textile         0.3107833
## Amelanchier_alnifolia  0.3760667
## Androsace_occidentalis 0.1148714
## Antennaria_neglecta    0.2749500
## Antennaria_parvifolia  0.3495500
```

```
# plot the data
pairs(traits)
```



```
# some variables look skewed - log transform all variables
traits <- log10(traits)
# plot the transformed data
```

```r
pairs(traits)
```



## Metadata

We have some information about the samples, including the habitat and site they were collected from, and a few basic environmental variables such as slope and moisture regime.

```r
# replace filename with file.choose() to open interactive window
metadata <- read.csv("data/plot.metadata.csv", header=TRUE, row.names=1)
# take a peek at the data
head(metadata)
```

```
##             habitat    site slope aspect slope.position rel.moisture
## mix-O-1 Mixedgrass Onefour     0    270            3.0            1
## mix-O-2 Mixedgrass Onefour    20    130            1.5            2
## mix-O-3 Mixedgrass Onefour     5     90            1.0            2
## mix-O-4 Mixedgrass Onefour     5     40            2.0            1
## mix-O-5 Mixedgrass Onefour     5    130            2.0            1
## mix-O-6 Mixedgrass Onefour     1     90            3.0            1
```

## Phylogeny

If you have a phylogeny in the commonly used Newick or Nexus format, it can be imported into R with the read.tree or read.nexus functions.

```r
# replace filename with file.choose() to open interactive window
phy <- read.tree("data/grassland.phylogeny.newick")
class(phy)
```

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

```
phy
```

```
##
## Phylogenetic tree with 76 tips and 68 internal nodes.
##
## Tip labels:
##  Antennaria_neglecta, Antennaria_parvifolia, Erigeron_glabellus, Erigeron_pumilus, Heterotheca_villos
## Node labels:
##  , , , , , , ...
##
## Rooted; includes branch lengths.
```

Our phylogeny is a special object of type `phylo`. The `phylo` format itself is documented at the **ape** homepage (http://ape.mpl.ird.fr/). A `phylo` object is a special type of `list` object - it has different elements such as tip labels and edge lengths, and R knows how to summarize and plot a `phylo` object due to the way it is defined by the **ape** package.

```
# list the elements of our phylogeny
names(phy)
```

```
## [1] "edge"        "edge.length" "Nnode"       "node.label"  "tip.label"
## [6] "root.edge"
```

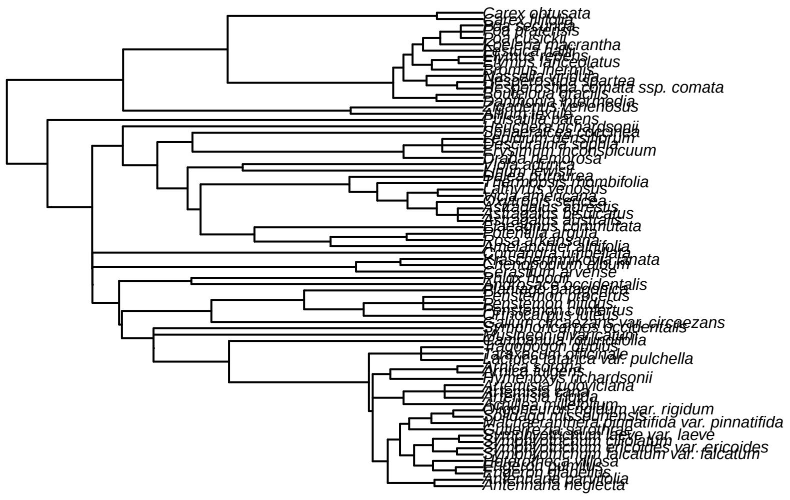
```
# what are the first few tip labels?
phy$tip.label[1:5]
```

```
## [1] "Antennaria_neglecta"   "Antennaria_parvifolia" "Erigeron_glabellus"
## [4] "Erigeron_pumilus"      "Heterotheca_villosa"
```

```
# how many tips does our phylogeny have?
Ntip(phy)
```

```
## [1] 76
```

```
# plot our phylogeny (the cex argument makes the labels small enough to read)
plot(phy, cex=0.5)
```



## Cleaning and matching data sets

Our workspace contains the community, trait, phylogeny, and metadata that we will need for our analyses.

```r
ls()
```

```
##  [1] "aa_counts"                 "Aa_counts"
##  [3] "AA_counts"                 "chi.sq"
##  [5] "clus.dis1"                 "clus.dis2"
##  [7] "cluster_average"           "cluster_complete"
##  [9] "cluster1"                  "comm"
## [11] "E.aa"                      "E.Aa"
## [13] "E.AA"                      "expected.aa"
## [15] "expected.Aa"               "expected.AA"
## [17] "forward_reverse_mutation" "grp"
## [19] "metadata"                  "num.a.alleles"
## [21] "num.A.alleles"             "num.alleles"
## [23] "p"                         "phy"
## [25] "q"                         "traits"
## [27] "vare.dis"                  "vare.mds"
## [29] "vare.mds0"                 "vare.pca"
## [31] "yeast_data"
```

The data sets we are using today have already been cleaned up so that they contain the same species and the same samples, but often when we are working with our own data, there will be mismatches among different types of data. For example, our community data might only contain a subset of the species in our phylogeny, or there might be some species for which we have trait information but no phylogenetic information. For some analyses, R will assume that species are in the same order in both the community data set and the phylogeny. Sometimes there might be a typo in the labels for a dataset, and we will want to catch those before proceeding.

There are several functions in **picante** that are designed to make sure different data sets match with one another. We should check that our phylogeny and community contain the same species, and that they are in the same order. The `match.phylo.comm` takes a community object and a phylo object, reports any species that are not present in both data sets, and outputs a version of each object in the same order and containing the same species.

```r
# check for mismatches/missing species
combined <- match.phylo.comm(phy, comm)
# the resulting object is a list with $phy and $comm elements.
# replace our original data with the sorted/matched data
phy <- combined$phy
comm <- combined$comm
```

We should do the same matching for our trait data.

```r
combined <- match.phylo.data(phy, traits)
# the resulting object is a list with $phy and $data elements.
# replace our original data with the sorted/matched data
phy <- combined$phy
traits <- combined$data
```

We should also check whether our community data and metadata are in the same order.

```r
all.equal(rownames(comm), rownames(metadata))
```

```
## [1] TRUE
```

```r
# they all match - if they didn't we could sort them to the same order
# sort metadata rows to be in the same order as community rows
metadata <- metadata[rownames(comm), ]
```

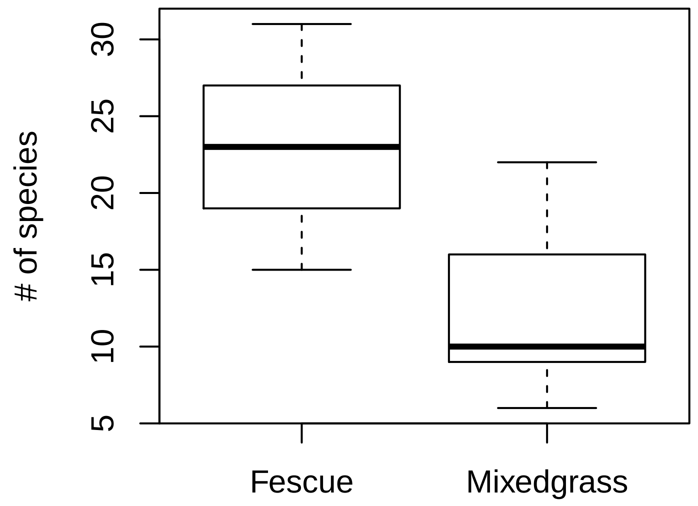
We're done! All of our data are now ready for analysis. In each of the sections below we will explore different ways of analyzing the biodiversity of plants in these grasslands.

## Visualizing and summarizing biodiversity data

### Community richness and diversity

At a most basic level, we can ask about the overall taxonomic diversity of these grasslands. How many plant species are there? Do habitats differ in species richness?

```
# compare species richness between fescue and mixedgrass habitats
boxplot(specnumber(comm) ~ metadata$habitat, ylab="# of species")
```

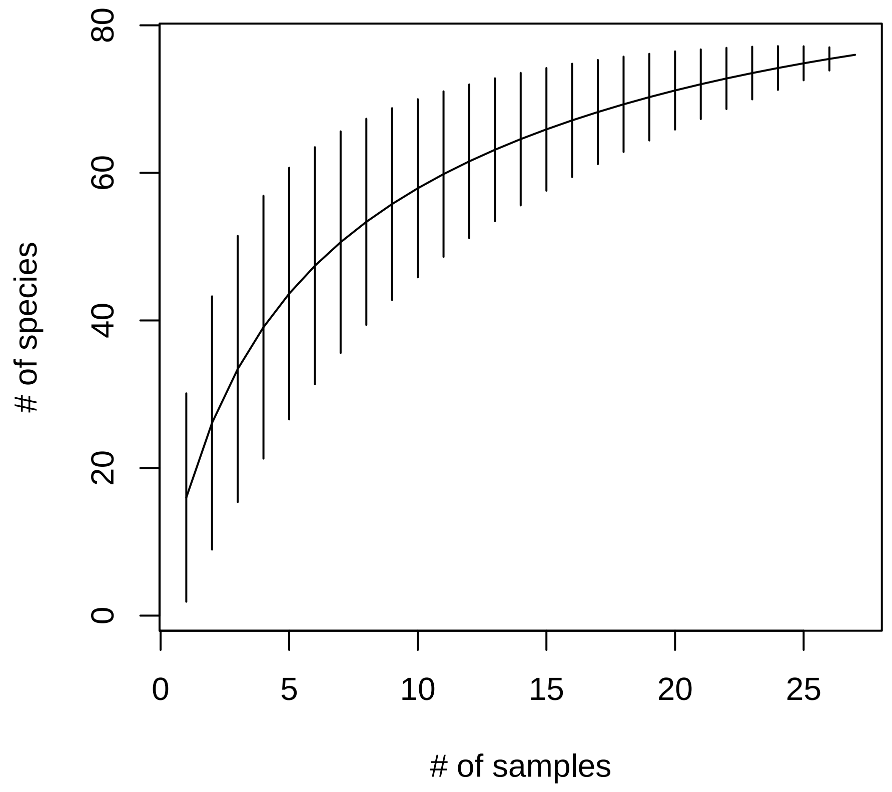


```
# statistical test of difference
t.test(specnumber(comm) ~ metadata$habitat)
```

```
##
##  Welch Two Sample t-test
##
## data:  specnumber(comm) by metadata$habitat
## t = 5.1371, df = 17.179, p-value = 7.972e-05
## alternative hypothesis: true difference in means is not equal to 0
## 95 percent confidence interval:
##   6.274287 15.008066
## sample estimates:
##     mean in group Fescue mean in group Mixedgrass
##                 22.70000                 12.05882
```

Did we do a good job of sampling the diversity that is out there? We can look at a collector's curve to assess this. What do you conclude, based on the graph?

```
# plot species accumulion curve across samples
plot(specaccum(comm), xlab="# of samples", ylab="# of species")
```



## Multivariate community analysis

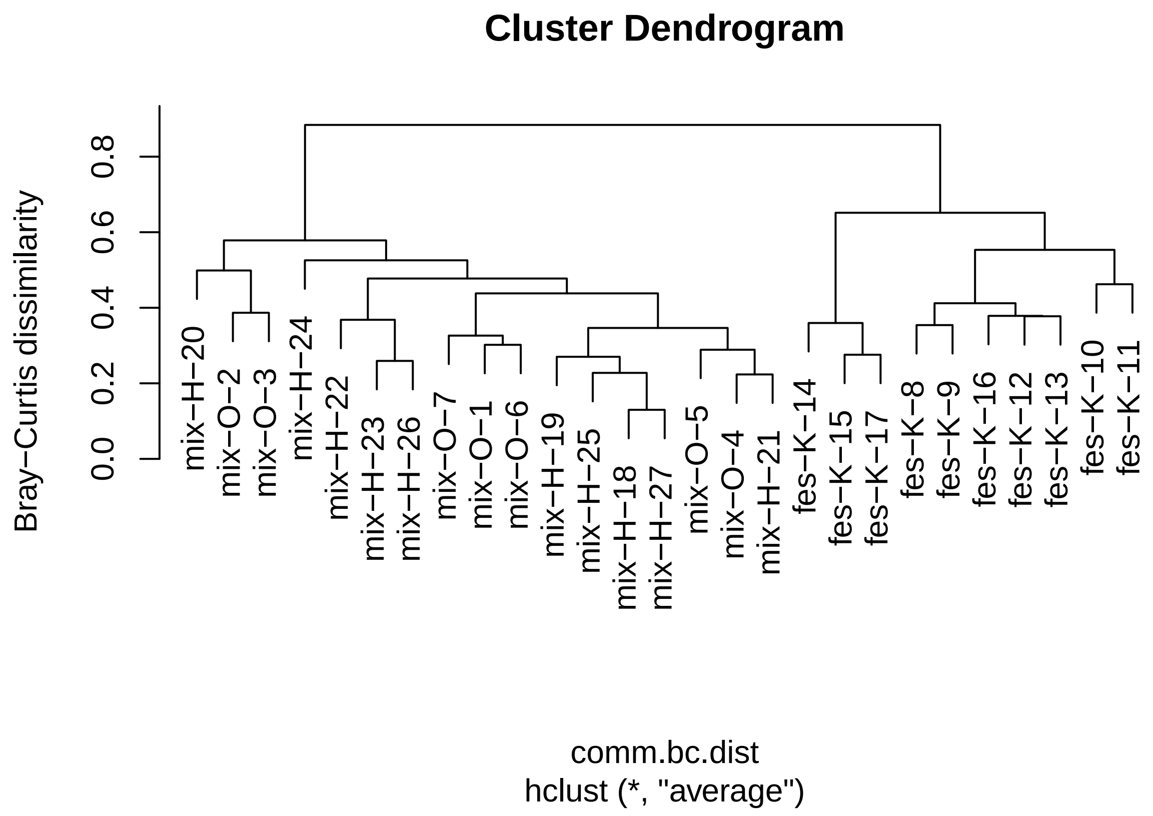
- How does the composition of plant communities vary across different samples?
- How are habitat type and environmental variables related to plant community composition?

We can use multivariate ordination methods to explore community structure in more detail. These methods are available in the **vegan** package, which also includes excellent documentation and tutorials for these methods. The book "Numerical Ecology in R" by Borcard et al. gives a great overview of multivariate analysis methods.

### Hierarchical clustering

We can cluster together plots based on their overall community composition. We will calculate Bray-Curtis dissimilarity among all the samples, an abundance-weighted measure of how dissimilar two communities are in terms of their species composition. We will then cluster together communities that are similar using an agglomerative hierarchical clustering algorithm.

```r
# calculate Bray-Curtis distance among samples
comm.bc.dist <- vegdist(comm, method="bray")
# cluster communities using average-linkage algorithm
comm.bc.clust <- hclust(comm.bc.dist, method="average")
# plot cluster diagram
plot(comm.bc.clust, ylab="Bray-Curtis dissimilarity")
```



What does your plot tell you about whether the two habitats contain different plant communities?

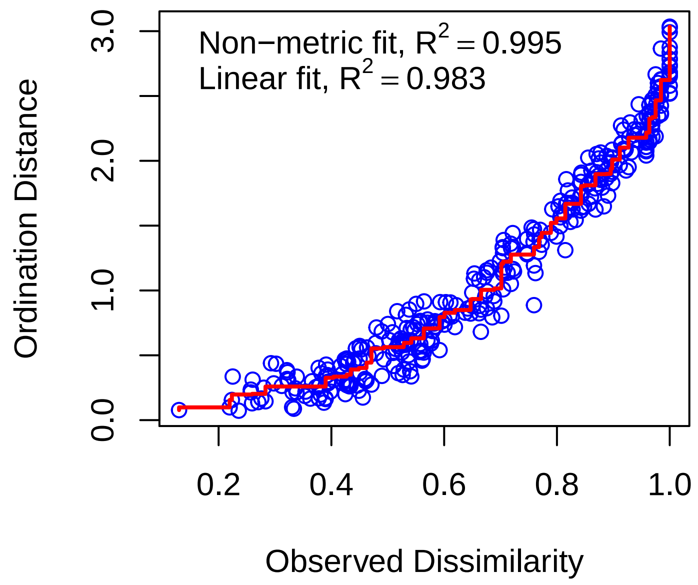
## Ordination

There are numerous ordination methods available in R. For now, let's use non-metric multidimensional scaling (NMDS) again to visualize the multivariate structure of these communities.

```
# The metaMDS function automatically transforms data and checks solution robustness
comm.bc.mds <- metaMDS(comm, dist="bray")
```

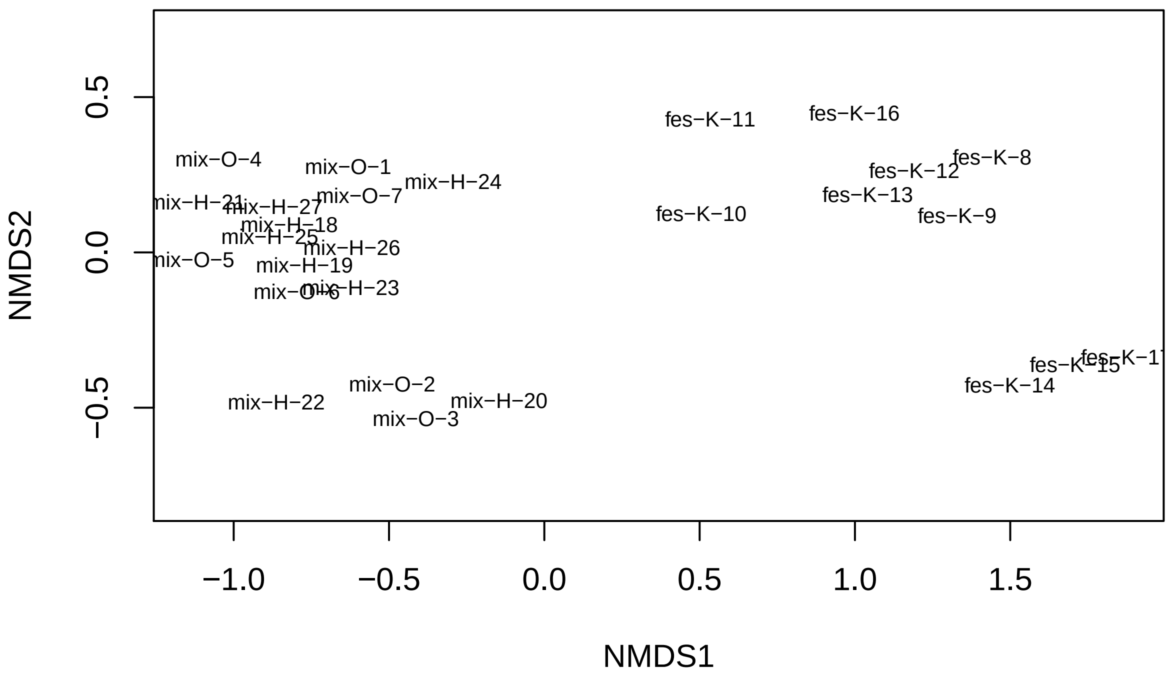
```
## Run 0 stress 0.07174232
## Run 1 stress 0.07957534
## Run 2 stress 0.07173992
## ... New best solution
## ... Procrustes: rmse 0.0008570244  max resid 0.003896048
## ... Similar to previous best
## Run 3 stress 0.0804576
## Run 4 stress 0.0748094
## Run 5 stress 0.08529145
## Run 6 stress 0.07480952
## Run 7 stress 0.08289128
## Run 8 stress 0.0748094
## Run 9 stress 0.07344474
## Run 10 stress 0.08006826
## Run 11 stress 0.07173981
## ... New best solution
## ... Procrustes: rmse 0.0003942072  max resid 0.001778549
## ... Similar to previous best
## Run 12 stress 0.07577702
## Run 13 stress 0.07174027
## ... Procrustes: rmse 0.0005221001  max resid 0.002384208
## ... Similar to previous best
## Run 14 stress 0.07935465
## Run 15 stress 0.07869153
## Run 16 stress 0.07886586
## Run 17 stress 0.08468726
## Run 18 stress 0.07377401
## Run 19 stress 0.07377215
## Run 20 stress 0.08008981
## *** Solution reached
```

```
# Assess goodness of ordination fit (stress plot)
stressplot(comm.bc.mds)
```

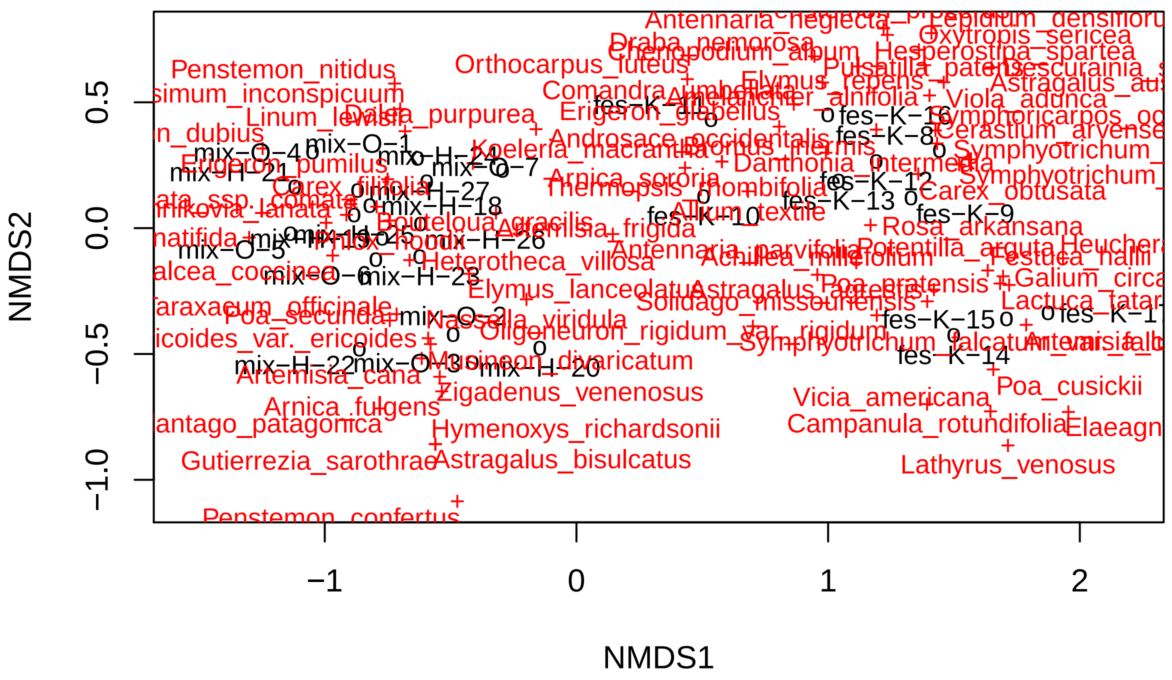


We can plot the ordination results in a variety of different ways.

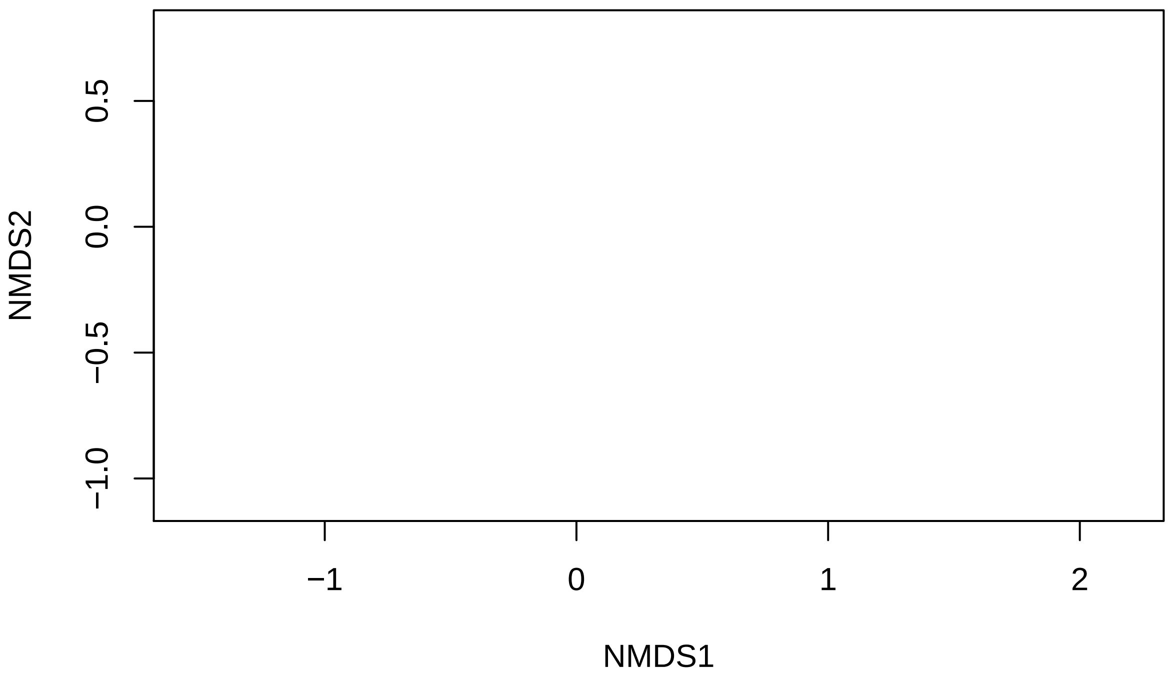
```
# plot site scores as text
ordiplot(comm.bc.mds, display="sites", type="text")
```



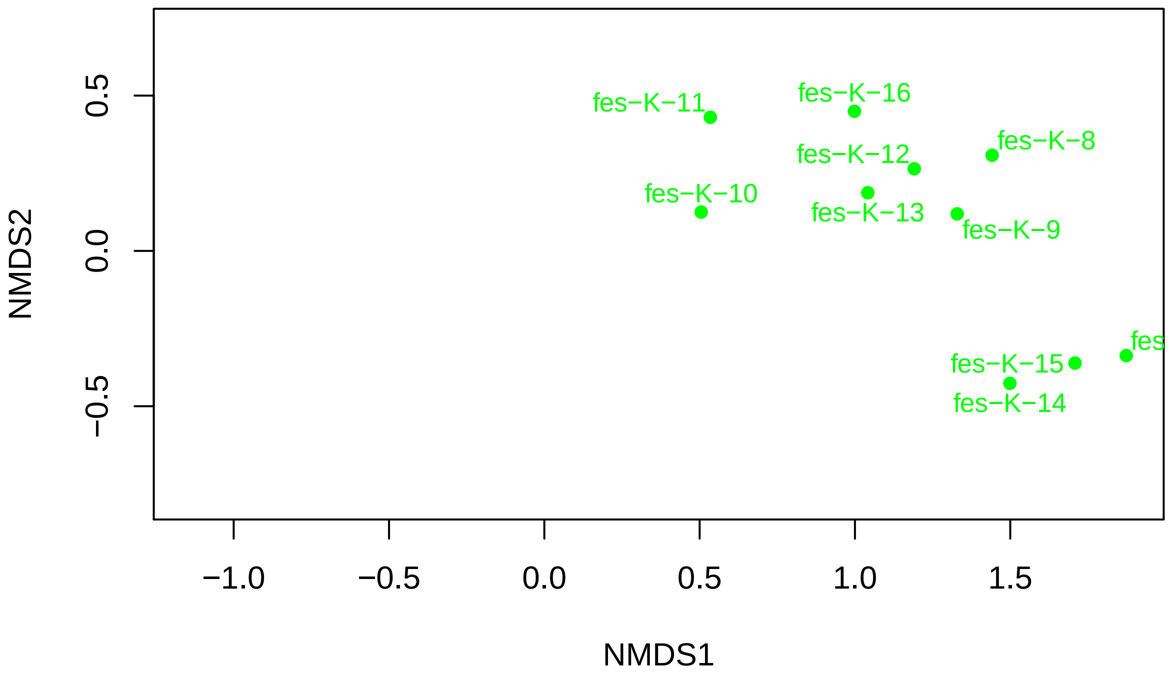
```
# automated plotting of results - tries to eliminate overlapping labels
ordipointlabel(comm.bc.mds)
```



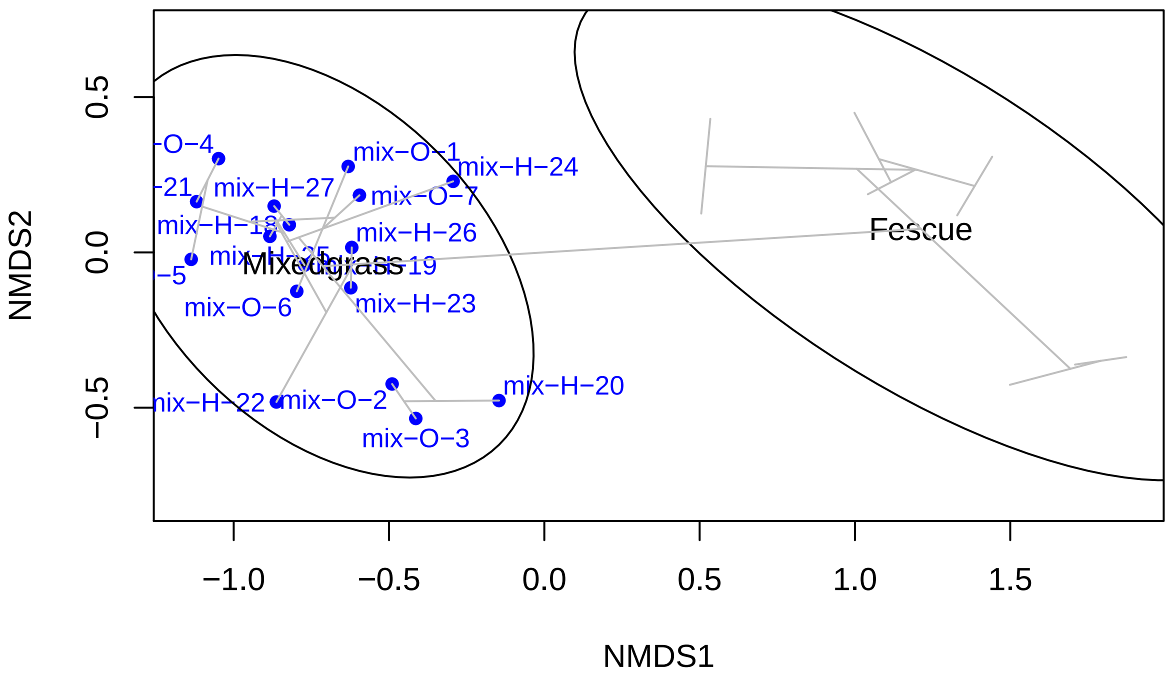
```
# ordination plots are highly customizable
# set up the plotting area but don't plot anything yet
mds.fig <- ordiplot(comm.bc.mds, type="none")
```



```r
# plot just the samples, colour by habitat, pch=19 means plot a circle
ordipointlabel(mds.fig$sites, "sites", pch=19, col="green", select=metadata$habitat=="Fescue")
```

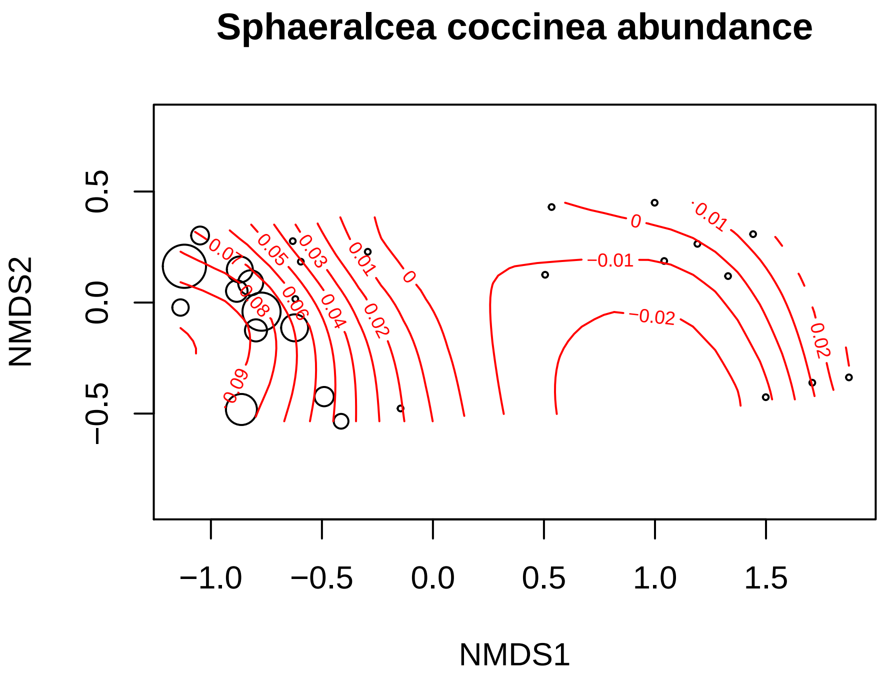


```r
ordipointlabel(mds.fig, "sites", pch=19, col="blue", select=metadata$habitat=="Mixedgrass")
# add confidence ellipses around habitat types
ordiellipse(comm.bc.mds, metadata$habitat, conf=0.95, label=TRUE)
# overlay the cluster results we calculated earlier
ordicluster(comm.bc.mds, comm.bc.clust, col="gray")
```



We can also visualize the abundance of species. The `ordisurf` function fits a smooth surface to estimates of species abundance.

```r
# plot Sphaeralcea abundance. cex increases the size of bubbles.
ordisurf(comm.bc.mds, comm[,"Sphaeralcea_coccinea"], bubble=TRUE, main="Sphaeralcea coccinea abundance"
```



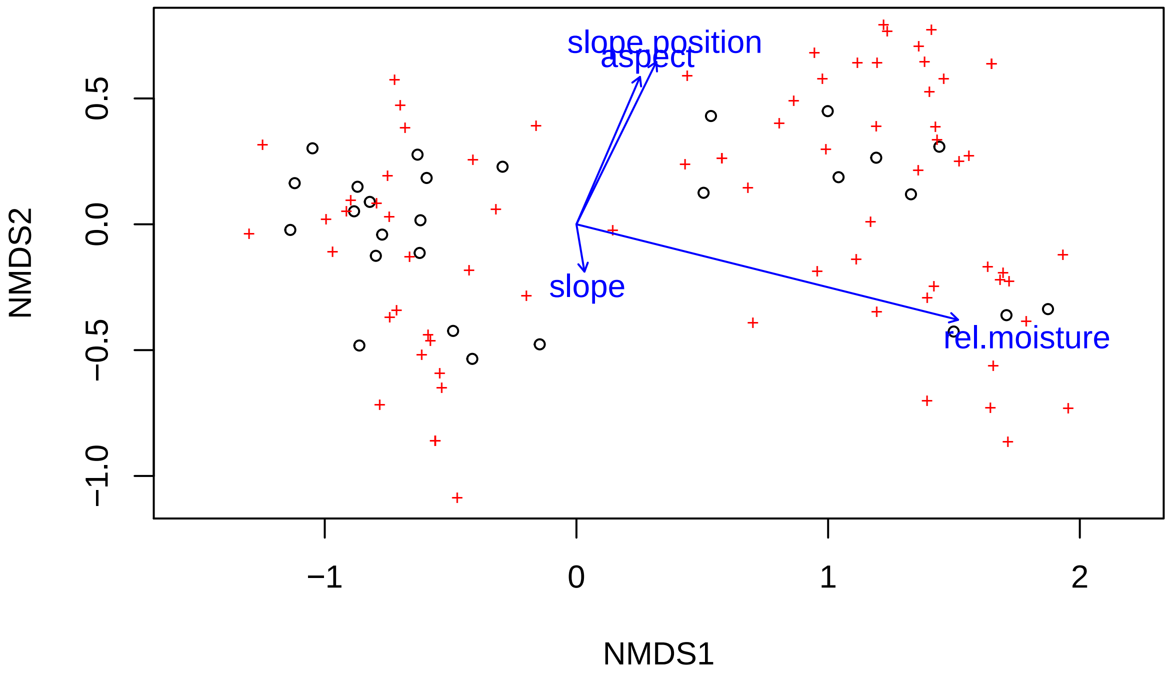
```
##
## Family: gaussian
## Link function: identity
##
## Formula:
## y ~ s(x1, x2, k = 10, bs = "tp", fx = FALSE)
```

```
##
## Estimated degrees of freedom:
## 4.31  total = 5.31
##
## REML score: -43.62134
```

**Adding environmental and trait data to ordinations**

- How are environmental variables correlated with the ordination axes?

```
ordiplot(comm.bc.mds)
# calculate and plot environmental variable correlations with the axes
# use the subset of metadata that are environmental data
plot(envfit(comm.bc.mds, metadata[,3:6]))
```



It is also possible to do a constrained ordination such as constrained correspondence analysis (CCA) or redundancy analysis (RDA), where trait or environmental data are incorporated directly into the ordination. These methods are implemented in the functions `cca` and `rda` in **vegan**.

## Trait evolution

### Phylogenetic signal

The idea of phylogenetic niche conservatism (the ecological similarity of closely related species) has attracted a lot of attention recently, for example in the widely used framework of inferring community assembly processes based on knowledge of community phylogenetic structure plus the phylogenetic conservatism of traits. (Webb et al. 2002).

Phylogenetic signal is a quantitative measure of the degree to which phylogeny predicts the ecological similarity of species. The K statistic is a measure of phylogenetic signal that compares the observed signal in a trait to the signal under a Brownian motion model of trait evolution on a phylogeny (Blomberg et al. 2003). K values of 1 correspond to a Brownian motion process, which implies some degree of phylogenetic signal or conservatism. K values closer to zero correspond to a random or convergent pattern of evolution, while K values greater than 1 indicate strong phylogenetic signal and conservatism of traits. The statistical significance of phylogenetic signal can be evaluated by comparing observed patterns of the variance of independent contrasts of the trait to a null model of shuffling taxa labels across the tips of the phylogeny. These tests are implemented in the `Kcalc`, `phylosignal`, and `multiPhylosignal` functions.

Let's measure phylogenetic signal in these data.

```
# one way to do it - apply the Kcalc function to each column of the data.frame
apply(traits, 2, Kcalc, phy)
```

```
##            SLA       LeafArea  LeafThickness            SLV LeafTissueDens
##      0.2563107      0.4231067      0.2418525      0.3310724      0.3298808
##            SRL            SRV RootTissueDens       RootDiam
##      0.2290093      0.2698531      0.2544904      0.3150881
```

```
# another way to do it with significance testing
# we have to convert the tree to be dichotomous before calculating P-values
multiPhylosignal(traits, multi2di(phy))
```

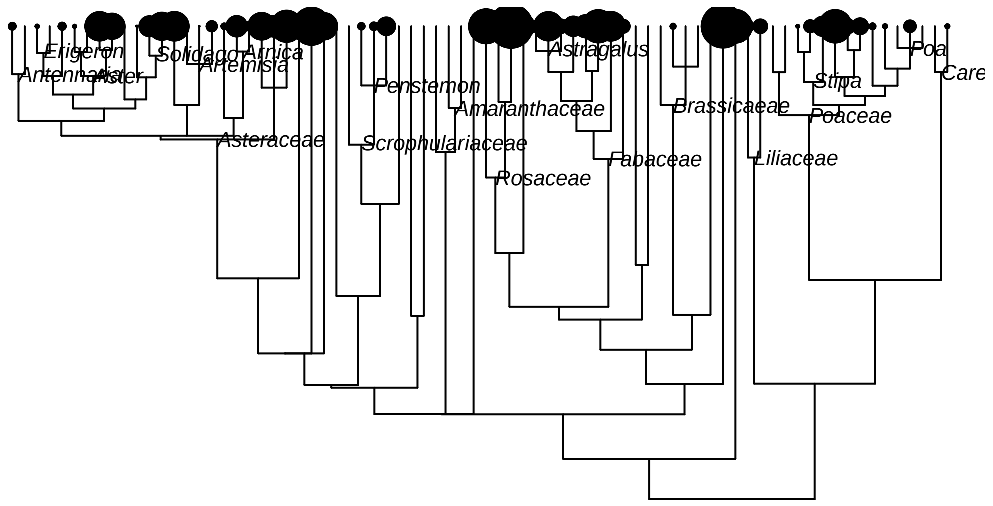
```
##                        K PIC.variance.obs PIC.variance.rnd.mean
## SLA            0.2563107     0.0006068179          0.0008462953
## LeafArea       0.4231067     0.0055507994          0.0132392745
## LeafThickness  0.2418525     0.0006669402          0.0008857578
## SLV            0.3310724     0.0005728618          0.0010767032
## LeafTissueDens 0.3298808     0.0006688721          0.0012380831
## SRL            0.2290093     0.0028408725          0.0037058240
## SRV            0.2698531     0.0011064783          0.0016700141
## RootTissueDens 0.2544904     0.0010609020          0.0015309866
## RootDiam       0.3150881     0.0005510416          0.0009860134
##                PIC.variance.P PIC.variance.Z
## SLA                     0.039      -1.644935
## LeafArea                0.001      -4.020852
## LeafThickness           0.085      -1.355367
## SLV                     0.001      -3.104165
## LeafTissueDens          0.001      -2.980366
## SRL                     0.068      -1.362272
## SRV                     0.004      -2.369931
## RootTissueDens          0.010      -2.114282
## RootDiam                0.001      -3.160181
```

In the output, `K` is the K statistic (magnitude of signal vs. Brownian motion), and `PIC.variance.P` is the P-value of the test for non-random signal. Most variables show more phylogenetic signal than expected by chance.

## Visualizing trait evolution

We can visualize trait values on the phylogeny by plotting a different color or size of symbol for each trait value. Let's visualize leaf area, the trait with the strongest phylogenetic signal. The cex argument to the tiplabels function adjusts the size of the trait symbols - some tinkering around with the scaling of the symbol sizes is required depending on the trait.

```
# Plot phylogeny facing upwards. Show node labels but not tip labels. cex shrinks labels.
plot(phy, direction="up", show.tip.label = FALSE, show.node.label=TRUE, cex=0.7)
# Plot leaf area on the phylogeny. cex argument scales symbol size by trait value.
tiplabels(pch = 19, col="black", cex = 3*(traits[,"LeafArea"] / max(traits[,"LeafArea"])))
```



## Phylogenetic analysis of trait relationships

Phylogenetic signal means that closely related species have similar traits. This violates the assumption of independence of data points that is inherent in many methods including correlation and regression (Felsenstein 1985). We can account for non-independence due to phylogenetic signal using methods including phylogenetically independent contrasts and phylogenetic generalized least squares (pGLS).

Generalized least squares methods work just like an ANOVA or linear model - we can test for relationships between categorical or continuous values, optionally taking phylogenetic relatedness into account.

Let's test for a relationship between specific root length (SRL) and root tissue density, taking phylogenetic relationships among species into account.

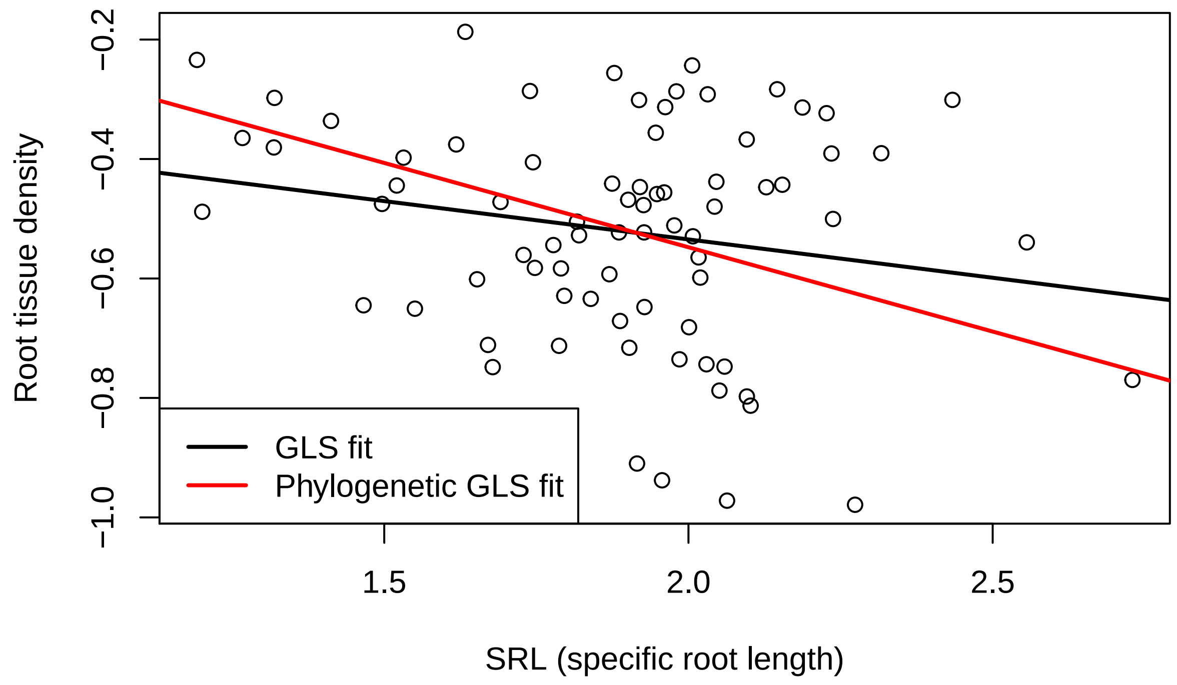
```
# GLS of root tissue density as a function of SRL - non-phylogenetic model
root.gls <- gls(RootTissueDens ~ SRL, data=traits)
anova(root.gls)
```

```
## Denom. DF: 74
##             numDF  F-value p-value
## (Intercept)     1 611.0773  <.0001
## SRL             1   3.0592  0.0844
```

```
# Phylogenetic GLS - adds effect of phylogeny to the model
root.pgls <- gls(RootTissueDens ~ SRL, correlation=corBrownian(value=1,phy), data=traits)
anova(root.pgls)
```

```
## Denom. DF: 74
##             numDF  F-value p-value
## (Intercept)     1 13.42163   5e-04
## SRL             1 20.07297  <.0001
```

```
# plot relationship
plot(RootTissueDens ~ SRL, data=traits, xlab="SRL (specific root length)", ylab="Root tissue density")
# add model fit lines - coef is the model fit coefficients, lwd increases line width
abline(coef(root.gls), lwd=2, col="black")
abline(coef(root.pgls), lwd=2, col="red")
legend("bottomleft", legend=c("GLS fit","Phylogenetic GLS fit"), lwd=2, col=c("black","red"))
```



There is a weak relationship between SRL and root tissue density. The relationship is not significant if we do not take phylogenetic relatedness into account. We see a stronger and significant relationship between SRL and root tissue density after taking phylogenetic relatedness into account.

## Phylogenetic and trait diversity

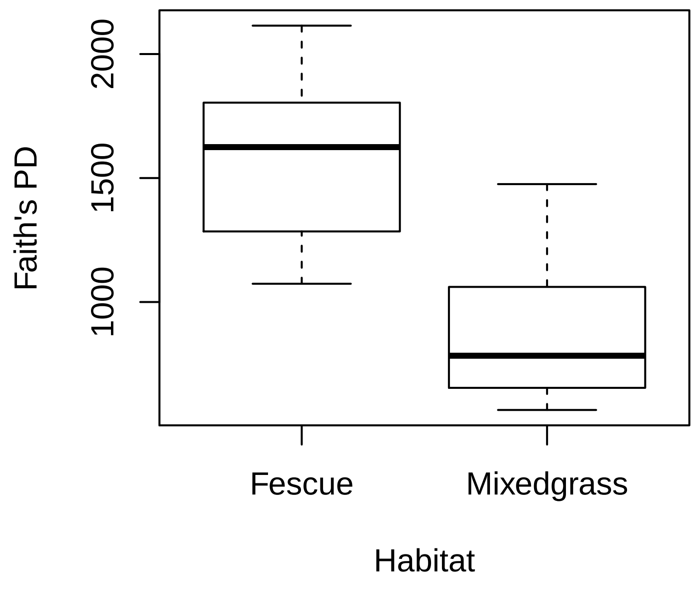
### Phylogenetic diversity

One of the earliest measures of phylogenetic relatedness in ecological communities was the phylogenetic diversity (PD) index proposed by Faith. Faith's PD is defined as the total branch length spanned by the tree including all species in a local community, optionally including the root node of the phylogeny. The `pd` function returns two values for each community, Faith's PD and species richness (SR).

```
# Calculate Faith's PD
comm.pd <- pd(comm, phy)
head(comm.pd)
```

```
##                 PD SR
## mix-O-1 1072.3697 16
## mix-O-2 1475.4767 22
## mix-O-3 1406.1708 21
## mix-O-4  564.5899  6
## mix-O-5  783.4028 10
## mix-O-6 1028.5796 13
```

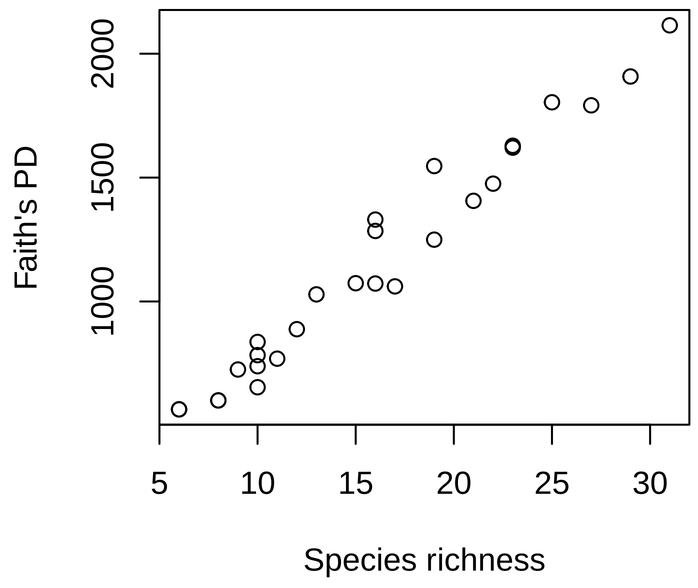
```
# Plot Faith's PD by habitat
boxplot(comm.pd$PD ~ metadata$habitat, xlab="Habitat", ylab="Faith's PD")
```



```
# Test for PD differences among habitats
t.test(comm.pd$PD ~ metadata$habitat)
```

```
##
##  Welch Two Sample t-test
##
## data:  comm.pd$PD by metadata$habitat
## t = 5.7161, df = 17.627, p-value = 2.195e-05
## alternative hypothesis: true difference in means is not equal to 0
## 95 percent confidence interval:
##  451.1886 976.8551
## sample estimates:
##    mean in group Fescue mean in group Mixedgrass
##               1602.2371                 888.2153
```

```
# Compare PD and species richness
plot(comm.pd$PD ~ comm.pd$SR, xlab="Species richness", ylab="Faith's PD")
```



Faith's PD is lower in mixedgrass habitats than in fescue habitats. But Faith's PD is highly correlated with species richness, and we already know that there are fewer species in mixedgrass habitats, so we need some way to compare phylogenetic diversity that takes this fact into account.

### $MPD$, $MNTD$, $SES_{MPD}$ and $SES_{MNTD}$

Another way of thinking about the phylogenetic relatedness of species in a community is to ask 'how closely related are the average pair of species or individuals in a community', and relate the patterns we observe to what we'd expect under various null models of evolution and community assembly. These types of questions are addressed by the measures of community phylogenetic structure such as MPD, MNTD, NRI and NTI described by Webb et al. and implemented in Phylocom.

The function `mpd` will calculate the mean pairwise distance between all species or individuals in each community. Similarly, the `mntd` function calculates the mean nearest taxon distance, the average distance separating each species or individual in the community from its closest heterospecific relative. The `mpd` and `mntd` functions differs slightly from the `pd` function in that they take a distance matrix as input rather than a phylogeny object. A `phylo` object can be converted to a interspecific phylogenetic distance matrix using the `cophenetic` function. Since the mpd and mntd functions can use any distance matrix as input, we can easily calculate trait diversity measures by substituting a trait distance matrix for the phylogenetic distance matrix. We'll return to this idea shortly.

If the community data represent abundance measures, the abundance data can be taken into account. Doing so changes the interpretation of these metrics from the average distance among two randomly chosen species from a community, to the average distance among two randomly chosen individuals in a community.

Measures of 'standardized effect size' of phylogenetic community structure can be calculated for MPD and MNTD by compared observed phylogenetic relatedness to the pattern expected under some null model of phylogeny or community randomization. Standardized effect sizes describe the difference between average phylogenetic distances in the observed communities versus null communities generated with some randomization method, standardized by the standard deviation of phylogenetic distances in the null data:

$SES_{metric} = \frac{Metric_{observed} - mean(Metric_{null})}{sd(Metric_{null})}$

Phylocom users will be familiar with the measures NRI and NTI; $SES_{MPD}$ and $SES_{MNTD}$ are equivalent to -1 times NRI and NTI, respectively. Several different null models can be used to generate the null communities. These include randomizations of the tip labels of the phylogeny, and various community randomizations that can hold community species richness and/or species occurrence frequency constant. These are described in more detail in the help files, as well as in the Phylocom manual. Let's calculate some of these measures of community phylogenetic structure for our example data set. We will ignore abundance information, and use a simple null model of randomly drawing species while keeping sample species richness constant.

```r
# convert phylogenety to a distance matrix
phy.dist <- cophenetic(phy)
# calculate ses.mpd
comm.sesmpd <- ses.mpd(comm, phy.dist, null.model="richness", abundance.weighted=FALSE, runs=999)
head(comm.sesmpd)
```

```
##         ntaxa  mpd.obs mpd.rand.mean mpd.rand.sd mpd.obs.rank   mpd.obs.z
## mix-O-1    16 231.3054      238.0677   11.602747          228 -0.58281933
## mix-O-2    22 239.5479      237.5243    9.754432          509  0.20745436
## mix-O-3    21 236.5260      237.4006    9.831186          406 -0.08895442
## mix-O-4     6 222.5255      239.8566   24.462915          193 -0.70846593
## mix-O-5    10 234.2013      238.2955   16.665847          327 -0.24566418
## mix-O-6    13 239.4120      238.4560   13.385227          461  0.07142243
##         mpd.obs.p runs
## mix-O-1     0.228  999
## mix-O-2     0.509  999
## mix-O-3     0.406  999
## mix-O-4     0.193  999
## mix-O-5     0.327  999
## mix-O-6     0.461  999
```

```r
# calculate ses.mntd
comm.sesmntd <- ses.mntd(comm, phy.dist, null.model="richness", abundance.weighted=FALSE, runs=999)
head(comm.sesmntd)
```

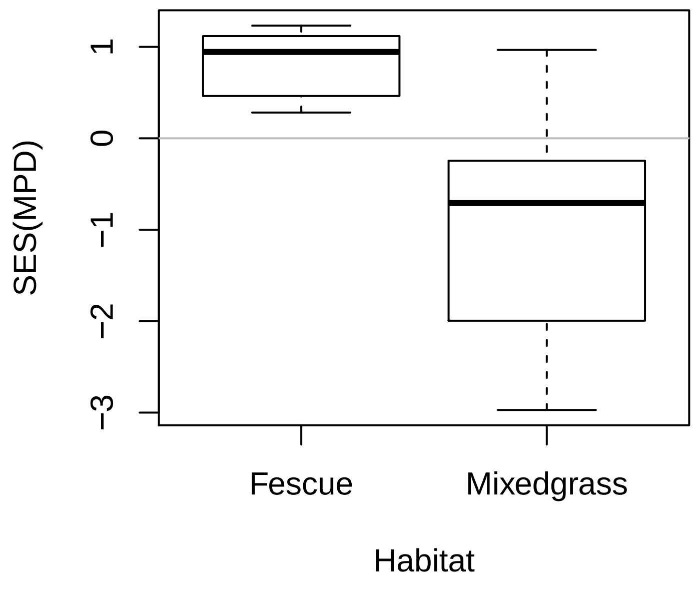
```
##          ntaxa  mntd.obs mntd.rand.mean mntd.rand.sd mntd.obs.rank
## mix-O-1    16  94.98812      103.91932     19.93694           344
## mix-O-2    22  97.41972       93.87682     14.77081           595
## mix-O-3    21  98.71519       94.47906     15.26050           632
## mix-O-4     6 136.86094      157.44900     44.03683           323
## mix-O-5    10 107.36711      123.93514     29.24699           296
## mix-O-6    13 118.94915      112.78640     23.83374           606
##          mntd.obs.z mntd.obs.p runs
## mix-O-1 -0.4479725      0.344  999
## mix-O-2  0.2398585      0.595  999
## mix-O-3  0.2775877      0.632  999
## mix-O-4 -0.4675191      0.323  999
## mix-O-5 -0.5664867      0.296  999
## mix-O-6  0.2585722      0.606  999
```

The output includes the following columns:

- `ntaxa` - Number of taxa in community
- `mpd.obs` - Observed mpd in community
- `mpd.rand.mean` - Mean mpd in null communities
- `mpd.rand.sd` - Standard deviation of mpd in null communities
- `mpd.obs.rank` - Rank of observed mpd vs. null communities
- `mpd.obs.z` - Standardized effect size of mpd vs. null communities (equivalent to -NRI)
- `mpd.obs.p` - P-value (quantile) of observed mpd vs. null communities (= mpd.obs.rank / runs + 1)
- `runs` - Number of randomizations

Positive SES values (mpd.obs.z $> 0$) and high quantiles (mpd.obs.p $> 0.95$) indicate phylogenetic evenness, while negative SES values and low quantiles (mpd.obs.p $< 0.05$) indicate phylogenetic clustering, relative to the null model. MPD is generally thought to be more sensitive to tree-wide patterns of phylogenetic clustering and eveness, while MNTD is more sensitive to patterns of evenness and clustering closer to the tips of the phylogeny.

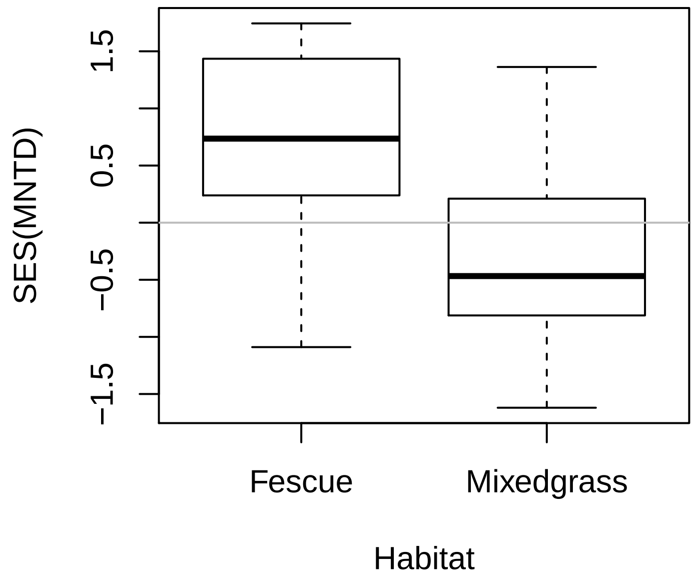
```r
# compare ses.mpd between habitats
plot(comm.sesmpd$mpd.obs.z ~ metadata$habitat, xlab="Habitat", ylab="SES(MPD)")
abline(h=0, col="gray")
```



```r
t.test(comm.sesmpd$mpd.obs.z ~ metadata$habitat)
```

```
##
##  Welch Two Sample t-test
##
## data:  comm.sesmpd$mpd.obs.z by metadata$habitat
## t = 5.9661, df = 20.298, p-value = 7.341e-06
## alternative hypothesis: true difference in means is not equal to 0
## 95 percent confidence interval:
##  1.182872 2.452864
## sample estimates:
##     mean in group Fescue mean in group Mixedgrass
##                0.8314913               -0.9863764
```

```r
# compare ses.mntd between habitats
plot(comm.sesmntd$mntd.obs.z ~ metadata$habitat, xlab="Habitat", ylab="SES(MNTD)")
abline(h=0, col="gray")
```



```
t.test(comm.sesmntd$mntd.obs.z ~ metadata$habitat)
```

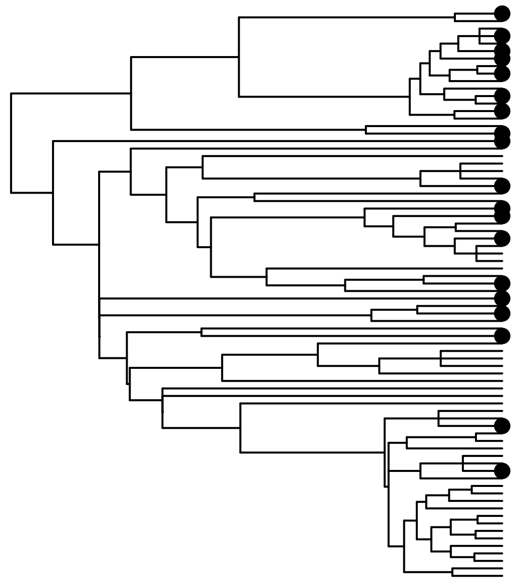
```
##
##  Welch Two Sample t-test
##
## data:  comm.sesmntd$mntd.obs.z by metadata$habitat
## t = 2.7994, df = 14.605, p-value = 0.01375
## alternative hypothesis: true difference in means is not equal to 0
## 95 percent confidence interval:
##  0.228923 1.704418
## sample estimates:
##     mean in group Fescue mean in group Mixedgrass
##                0.5918428               -0.3748276
```

It looks like plant communities from fescue habitats are phylogenetically even more distantly related than expected by chance, ($SES > 0$), and communities from mixedgrass habitats are phylogenetically clustered (more closely related than expected by chance, $SES < 0$).

Let's look at the distribution of species from samples in these different habitats on the phylogeny. Fescue community 'fes-K-11' contains species that are phylogenetically even.

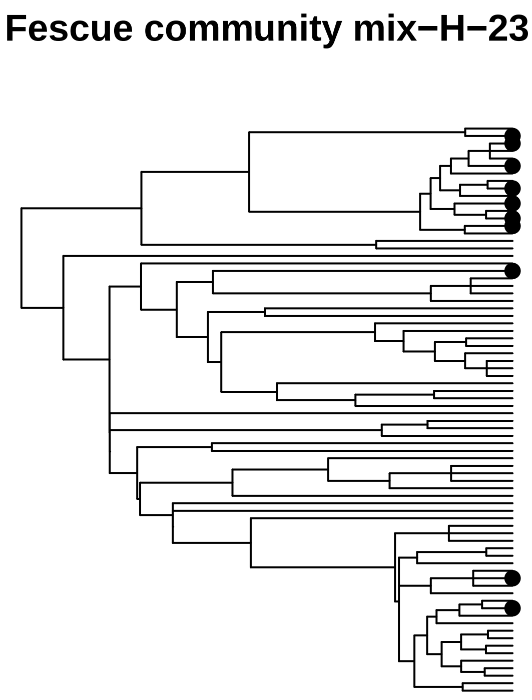
```
# plot species present in a fescue community
plot(phy, show.tip.label=FALSE, main="Fescue community fes-K-11")
tiplabels(tip = which(phy$tip.label %in% colnames(comm)[comm["fes-K-11",]>0]), pch=19)
```

## Fescue community fes−K−11



Mixedgrass community 'mix-H-23' contains species that are phylogenetically clumped.

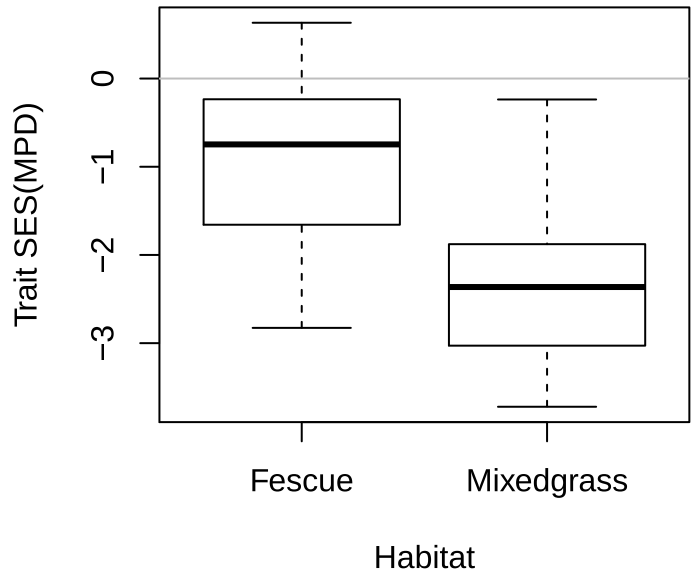
```
# plot species present in a mixedgrass community
plot(phy, show.tip.label=FALSE, main="Fescue community mix-H-23")
tiplabels(tip = which(phy$tip.label %in% colnames(comm)[comm["mix-H-23",]>0]), pch=19)
```



## Trait diversity

We can calculate measures of trait diversity within communities in a manner analogous to the methods we used to calculate phylogenetic diversity. Let's calculate the standardized effect size of functional trait diversity by measuring trait dissimilarity among co-occurring species, and comparing observed trait diversity to a null model.

```r
# calculate trait distance - Euclidean distance among scaled trait values - we want the full distance m
trait.dist <- as.matrix(dist(scale(traits), method="euclidean"))
# calculate trait ses.mpd
comm.sesmpd.traits <- ses.mpd(comm, trait.dist, null.model="richness", abundance.weighted=FALSE, runs=99
# compare trait ses.mpd between habitats
plot(comm.sesmpd.traits$mpd.obs.z ~ metadata$habitat, xlab="Habitat", ylab="Trait SES(MPD)")
abline(h=0, col="gray")
```



In contrast to the pattern we saw for phylogenetic diversity, trait diversity is lower than expected in both habitats ($SES_{MPD} < 0$), indicating that co-occurring plants have similar leaf and root traits, and this pattern of trait clustering is stronger in mixedgrass habitats.

The `treedive` function in **vegan** calculates a measure of functional trait diversity that is similar to Faith's PD.

## Phylogenetic beta-diversity

We can measure patterns of phylogenetic relatedness among communities in a manner similar to the within-community phylogenetic diversity measures described above. The `unifrac` and `phylosor` functions measure the among-community equivalent of Faith's PD, the total unique/shared branch length between communities. The `comdist` and `comdistnt` functions measure the among-community equivalent of MPD and MNTD, the mean pairwise distance or mean nearest taxon distance between pairs of species drawn from two distinct communities.

Let's compare a few different ways of measuring dissimilarity among communities. We've already calculated the Bray-Curtis distance among communities based on shared species (`comm.bc.dist`). Since the Bray-Curtis distance incorporates species abundances, we should use abundance information when calculating phylogenetic and trait diversity as well.

```r
# calculate phylogenetic MNTD beta diversity
comm.mntd.dist <- comdistnt(comm, phy.dist, abundance.weighted=TRUE)
# calculate functional trait MNTD beta diversity
comm.mntd.traits.dist <- comdistnt(comm, trait.dist, abundance.weighted=TRUE)
# calculate Mantel correlation for taxonomic Bray-Curtis vs. phylogenetic MNTD diversity
mantel(comm.bc.dist, comm.mntd.dist)
```

```
##
## Mantel statistic based on Pearson's product-moment correlation
##
## Call:
## mantel(xdis = comm.bc.dist, ydis = comm.mntd.dist)
##
## Mantel statistic r: 0.8597
##       Significance: 0.001
##
## Upper quantiles of permutations (null model):
##    90%    95%  97.5%    99%
## 0.0689 0.1056 0.1357 0.1864
## Permutation: free
## Number of permutations: 999
```

```r
# calculate Mantel correlation for taxonomic Bray-Curtis vs. trait MNTD diversity
mantel(comm.bc.dist, comm.mntd.traits.dist)
```

```
##
## Mantel statistic based on Pearson's product-moment correlation
##
## Call:
## mantel(xdis = comm.bc.dist, ydis = comm.mntd.traits.dist)
##
## Mantel statistic r: 0.9524
##       Significance: 0.001
##
## Upper quantiles of permutations (null model):
##    90%    95%  97.5%    99%
## 0.0846 0.1277 0.1664 0.2253
## Permutation: free
## Number of permutations: 999
```

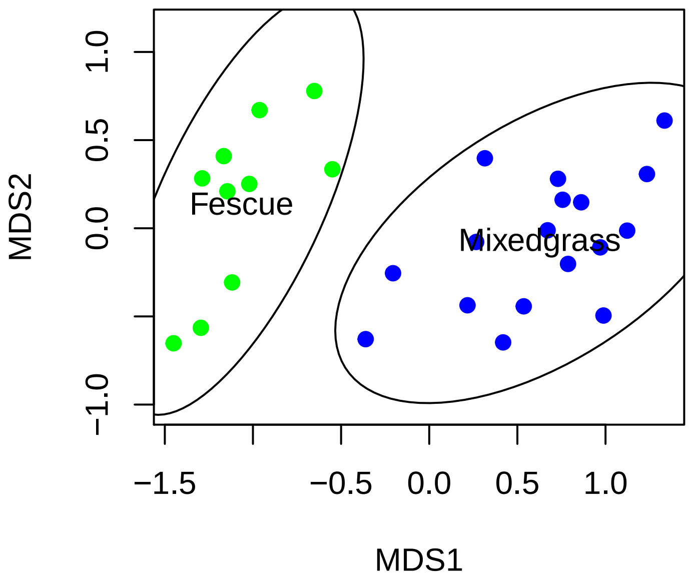
## Phylogeny/trait-based ordinations

Since we can calculate phylogeny- and trait-based measures of dissimilarity among samples, we can also perform an ordination of samples based on these metrics. Let's compare phylogeny- and trait-based ordinations with the species-based ordination we performed earlier.

```
# NMDS ordination of phylogenetic distances - use monoMDS since we only have among-sample distances
comm.mntd.mds <- monoMDS(comm.mntd.dist)
# set up the plotting area but don't plot anything yet
mds.fig <- ordiplot(comm.mntd.mds, type="none")
```

```
## species scores not available
```

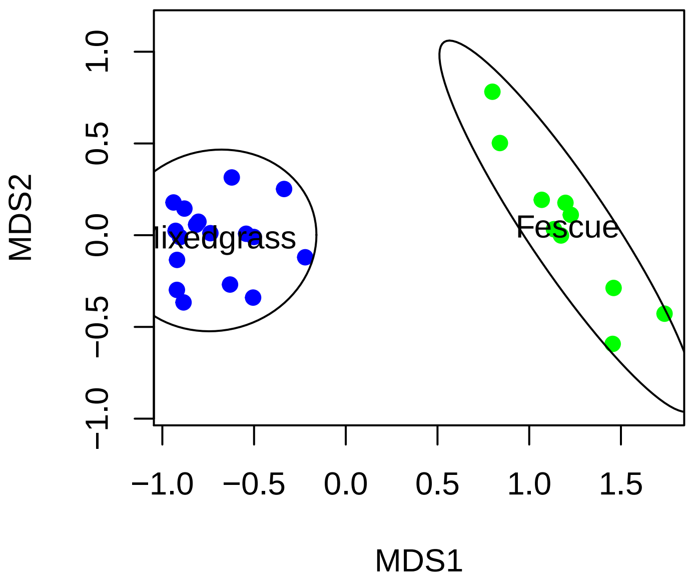
```
# plot just the samples, colour by habitat, pch=19 means plot a circle
points(mds.fig, "sites", pch=19, col="green", select=metadata$habitat=="Fescue")
points(mds.fig, "sites", pch=19, col="blue", select=metadata$habitat=="Mixedgrass")
# add confidence ellipses around habitat types
ordiellipse(comm.mntd.mds, metadata$habitat, conf=0.95, label=TRUE)
```



```
# NMDS ordination of trait distances - use monoMDS since we only have among-sample distances
comm.mntd.traits.mds <- monoMDS(comm.mntd.traits.dist)
# set up the plotting area but don't plot anything yet
mds.fig <- ordiplot(comm.mntd.traits.mds, type="none")
```

```
## species scores not available
```

```
# plot just the samples, colour by habitat, pch=19 means plot a circle
points(mds.fig, "sites", pch=19, col="green", select=metadata$habitat=="Fescue")
points(mds.fig, "sites", pch=19, col="blue", select=metadata$habitat=="Mixedgrass")
# add confidence ellipses around habitat types
ordiellipse(comm.mntd.traits.mds, metadata$habitat, conf=0.95, label=TRUE)
```



It looks like fescue and mixedgrass habitats are quite distinct regardless of how we quantify their biodiversity - they contain different species, phylogenetically distinct taxa, and the traits of species in the two habitats are distinct.

## Testing for multivariate differences among groups

We can quantify the relationship between dissimilarity measures and different explanatory variables using the Permutational MANOVA (a.k.a. AMOVA) framework in the `adonis` function in **vegan**. This method allows ANOVA-like tests of the variance in beta diversity explained by categorical or continuous variables.

Let's quantify the degree to which habitat can explain taxonomic, phylogenetic, and trait dissimilarity among grasslands.

```
# Taxonomic (Bray-Curtis) dissimilarity explained
adonis(comm.bc.dist ~ habitat, data=metadata)
```

```
##
## Call:
## adonis(formula = comm.bc.dist ~ habitat, data = metadata)
##
## Permutation: free
## Number of permutations: 999
##
## Terms added sequentially (first to last)
##
##           Df SumsOfSqs MeanSqs F.Model      R2 Pr(>F)
## habitat    1    3.3656  3.3656  25.124 0.50123  0.001 ***
## Residuals 25    3.3490  0.1340         0.49877
## Total     26    6.7146                 1.00000
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
# Phylogenetic dissimilarity explained
adonis(comm.mntd.dist ~ habitat, data=metadata)
```

```
##
## Call:
## adonis(formula = comm.mntd.dist ~ habitat, data = metadata)
```

```
##
## Permutation: free
## Number of permutations: 999
##
## Terms added sequentially (first to last)
##
##           Df SumsOfSqs MeanSqs F.Model      R2 Pr(>F)
## habitat    1     22814 22813.7  34.904 0.58267  0.001 ***
## Residuals 25     16340   653.6         0.41733
## Total     26     39154                 1.00000
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
# Trait dissimilarity explained
adonis(comm.mntd.traits.dist ~ habitat, data=metadata)
```

```
##
## Call:
## adonis(formula = comm.mntd.traits.dist ~ habitat, data = metadata)
##
## Permutation: free
## Number of permutations: 999
##
## Terms added sequentially (first to last)
##
##           Df SumsOfSqs MeanSqs F.Model      R2 Pr(>F)
## habitat    1   10.9154 10.9154  63.959 0.71897  0.001 ***
## Residuals 25    4.2666  0.1707         0.28103
## Total     26   15.1820                 1.00000
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

These results support the pattern we can see visually in the ordination diagrams. These habitats are distinct in terms of their taxonomic, phylogenetic, and functional trait diversity.