Multivariate analysis in **R**

Oliver Purschke
(with contributions from Helge Bruelheide, Henrik von Wehrden and Heike Zimmermann)

January 27, 2017

In this practical you will perform cluster and ordination analyses, using **R** and the add-on packages **vegan**: **gclus**, and **clustSim**. You will use three data sets (available in **vegan**), (i) the dune meadow vegetation (*dune*) and environmental data (*dune.env*), (ii) a reduced version of the *dune* meadow vegetation data (five of the most common species in five sites) and (iii) the *varespec* and *varechem* data describing vegetation and soil characteristics in a pine forest.

1 Multivariate data

First, lets load the required **R** packages and data.

```
library(vegan)
library(knitr)
library(clusterSim)
library(gclus)

data(dune)
data(dune.env)
```

1.1 Exploring the *dune* vegetation and environmental data

```
str(dune)
```

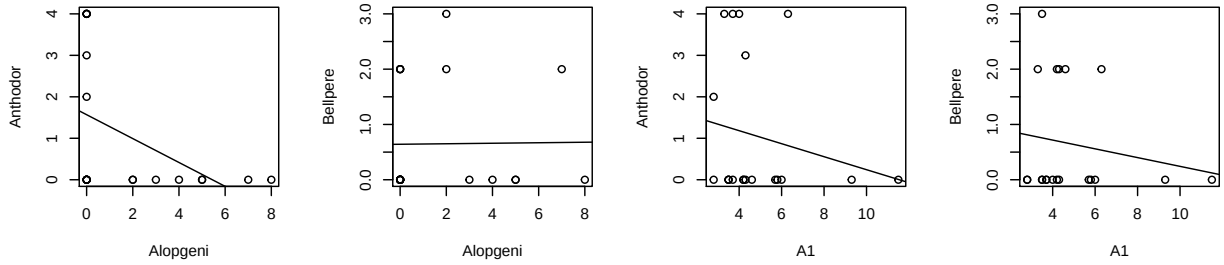
```
## 'data.frame': 20 obs. of  30 variables:
##  $ Achimill: num  1 3 0 0 2 2 2 0 0 4 ...
##  $ Agrostol: num  0 0 4 8 0 0 0 4 3 0 ...
##  $ Airaprae: num  0 0 0 0 0 0 0 0 0 0 ...
##  $ Alopgeni: num  0 2 7 2 0 0 0 5 3 0 ...
##  $ Anthodor: num  0 0 0 0 4 3 2 0 0 4 ...
##  $ Bellpere: num  0 3 2 2 2 0 0 0 0 2 ...
##  $ Bromhord: num  0 4 0 3 2 0 2 0 0 4 ...
##  $ Chenalbu: num  0 0 0 0 0 0 0 0 0 0 ...
##  $ Cirsarve: num  0 0 0 2 0 0 0 0 0 0 ...
##  $ Comapalu: num  0 0 0 0 0 0 0 0 0 0 ...
##  $ Eleopalu: num  0 0 0 0 0 0 0 4 0 0 ...
##  $ Elymrepe: num  4 4 4 4 4 0 0 0 6 0 ...
##  $ Empenigr: num  0 0 0 0 0 0 0 0 0 0 ...
##  $ Hyporadi: num  0 0 0 0 0 0 0 0 0 0 ...
##  $ Juncarti: num  0 0 0 0 0 0 0 4 4 0 ...
##  $ Juncbufo: num  0 0 0 0 0 0 2 0 4 0 ...
##  $ Lolipere: num  7 5 6 5 2 6 6 4 2 6 ...
```

```
##  $ Planlanc: num  0 0 0 0 5 5 5 0 0 3 ...
##  $ Poaprat : num  4 4 5 4 2 3 4 4 4 4 ...
##  $ Poatriv : num  2 7 6 5 6 4 5 4 5 4 ...
##  $ Ranuflam: num  0 0 0 0 0 0 0 2 0 0 ...
##  $ Rumeacet: num  0 0 0 0 5 6 3 0 2 0 ...
##  $ Sagiproc: num  0 0 0 5 0 0 0 2 2 0 ...
##  $ Salirepe: num  0 0 0 0 0 0 0 0 0 0 ...
##  $ Scorautu: num  0 5 2 2 3 3 3 3 2 3 ...
##  $ Trifprat: num  0 0 0 0 2 5 2 0 0 0 ...
##  $ Trifrepe: num  0 5 2 1 2 5 2 2 3 6 ...
##  $ Vicilath: num  0 0 0 0 0 0 0 0 0 1 ...
##  $ Bracruta: num  0 0 2 2 2 6 2 2 2 2 ...
##  $ Callcusp: num  0 0 0 0 0 0 0 0 0 0 ...

str(dune.env)

## 'data.frame': 20 obs. of  5 variables:
##  $ A1        : num  2.8 3.5 4.3 4.2 6.3 4.3 2.8 4.2 3.7 3.3 ...
##  $ Moisture  : Ord.factor w/ 4 levels "1"<"2"<"4"<"5": 1 1 2 2 1 1 1 4 3 2 ...
##  $ Management: Factor w/ 4 levels "BF","HF","NM",..: 4 1 4 4 2 2 2 2 2 1 ...
##  $ Use       : Ord.factor w/ 3 levels "Hayfield"<"Haypastu"<..: 2 2 2 2 1 2 3 3 1 1 ...
##  $ Manure    : Ord.factor w/ 5 levels "0"<"1"<"2"<"3"<..: 5 3 5 5 3 3 4 4 2 2 ...
```

Question: How many species were sampled in how many sites? What character do the environmental variables have?

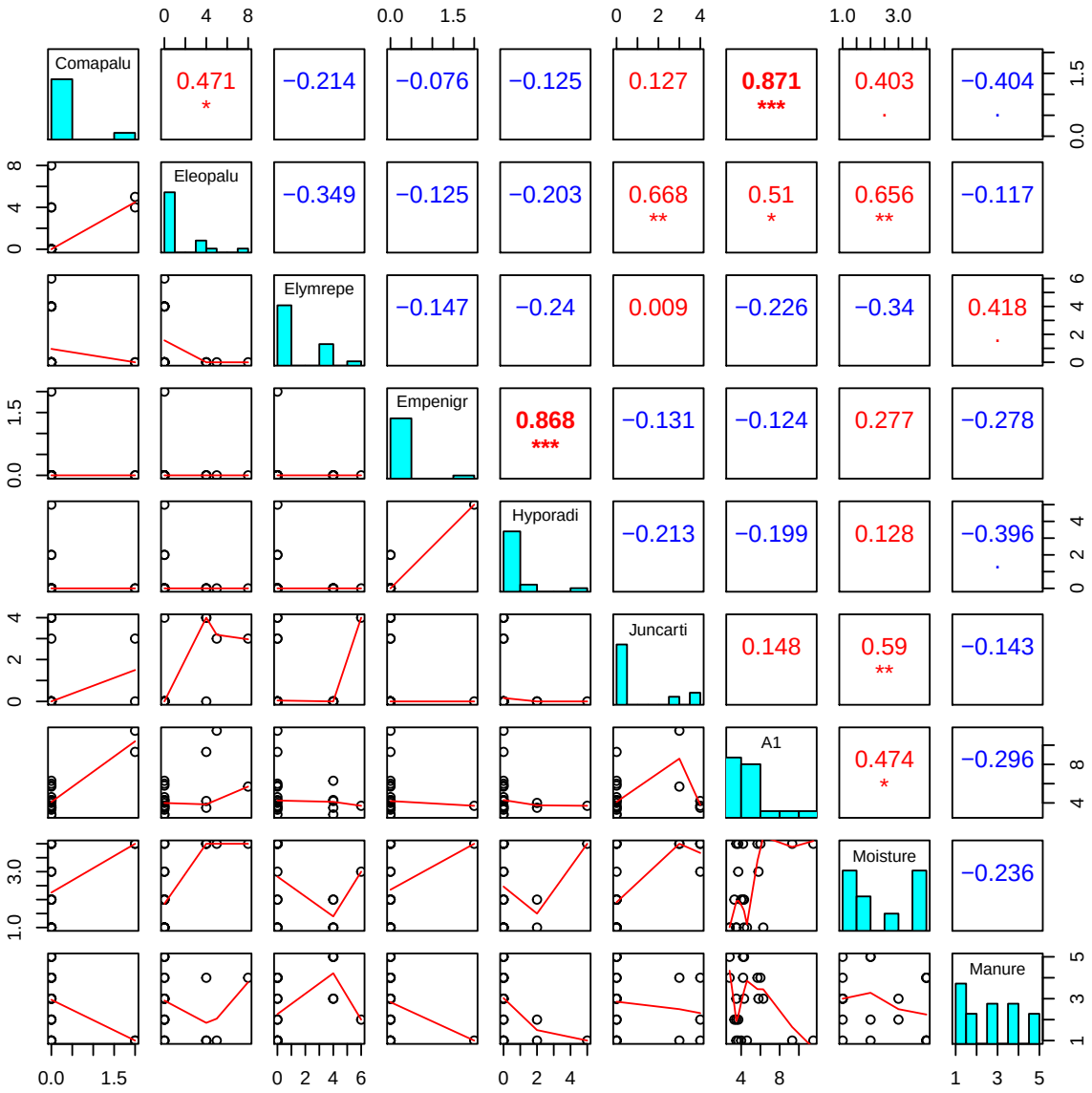
1.2 Visualizing relationships among species and environmental variables

```
par(mfrow=c(1,4))
plot (dune[,5]~dune[,4],ylab=names(dune[5]), xlab=names(dune[4]))
abline(lm(dune[,5]~dune[,4]))
plot (dune[,6]~dune[,4],ylab=names(dune[6]), xlab=names(dune[4]))
abline(lm(dune[,6]~dune[,4]))
plot (dune[,5]~dune.env[,1],ylab=names(dune[5]), xlab=names(dune.env[1]))
abline(lm(dune[,5]~dune.env[,1]))
plot (dune[,6]~dune.env[,1],ylab=names(dune[6]), xlab=names(dune.env[1]))
abline(lm(dune[,6]~dune.env[,1]))
```



To explore many pairwise relationships between species and between species and environmental variables at once, run the following function:

```r
# Author: Francois Gillet, February 2007
panel.cor <- function(x, y, method = "pearson", digits = 3, cex.cor = 1.2) {
    usr <- par("usr")
    on.exit(par(usr))
    par(usr = c(0, 1, 0, 1))
    r <- cor(x, y, method = method)
    ra <- cor.test(x, y, method = method)$p.value
    txt <- round(r, digits)
    sig <- 1
    prefix <- ""
    if (ra <= 0.1)
        prefix <- "."
    if (ra <= 0.05)
        prefix <- "*"
    if (ra <= 0.01)
        prefix <- "**"
    if (ra <= 0.001)
        prefix <- "***"
    if (ra <= 0.001)
        sig <- 2
    color <- 2
    if (r < 0)
        color <- 4
    # color <- 'gray10' if(r < 0) color <- 'gray50'
    txt <- paste(txt, prefix, sep = "\n")
    text(0.5, 0.5, txt, cex = cex.cor, font = sig, col = color)
}
## Put histograms on the diagonal
panel.hist <- function(x, ...) {
    usr <- par("usr")
    on.exit(par(usr))
    par(usr = c(usr[1:2], 0, 1.5))
    h <- hist(x, plot = FALSE)
    breaks <- h$breaks
    nB <- length(breaks)
    y <- h$counts
    y <- y/max(y)
    rect(breaks[-nB], 0, breaks[-1], y, col = "cyan", ...)
    # rect(breaks[-nB], 0, breaks[-1], y, col='gray', ...)
}

pairs(cbind(dune[, 10:15], dune.env[, c(1, 2, 5)]), lower.panel = panel.smooth,
    upper.panel = panel.cor, diag.panel = panel.hist)
```



To quantify all possible inter-relations between the 30 species and the five environenmental descriptors, we would need to carry out 435 $(n * (n-1)/2)$ separate analyses. This would be very tidious and still we wouldn't see the overall picture.

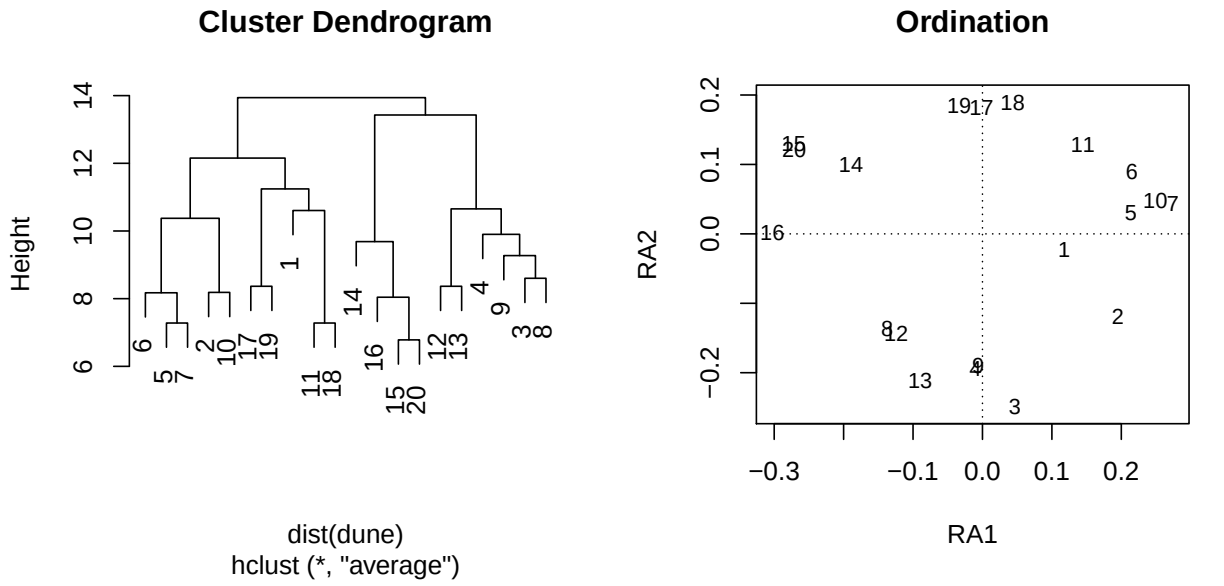
2 Multivatiate techniques

Multivariate methods help to explore simultaneously the relationships between all the objects (e.g. sites) with respect to their characteristics (species composition).

2.1 A quick appetizer

```
dune.clust <- hclust(dist(dune), "ave")
dune.ca <- decorana(dist(dune), ira = 1)
```

```
par(mfrow = c(1,2))
plot(dune.clust)
plot(dune.ca, type = "t", display = "sites", main = "Ordination")
```



3 Cluster analysis (step by step)

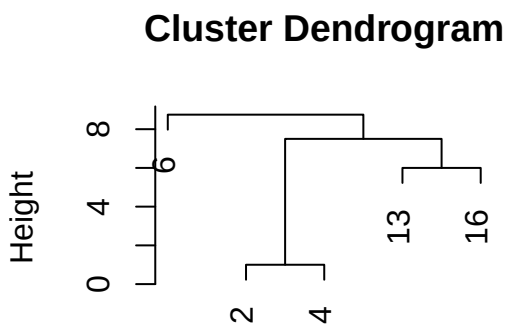
Here, you will perform hierarchical agglomerative clustering, which fuses objects (e.g. sites) together into single clusters that form a dendrogram.

3.1 Example using the reduced dune data set

Run cluster analysis on the small subset of the data (five sites x five species). Here, you use exactly the same settings (Manhattan-distance, average-linkage) that you used for generating the dendrogram in the exercise this morning.

```
?hclust
dune5 <- dune[c(2,13,4,16,6), c(6,16,15,18,21)]

dune5.man <- dist(dune5, "manhattan")
dune5.man

##     2 13  4 16
## 13  8
## 4   1  7
## 16  8  6  7
## 6   8 10  7 10

dune5.man.ave.clust <- hclust(dune5.man, method = "average")
```

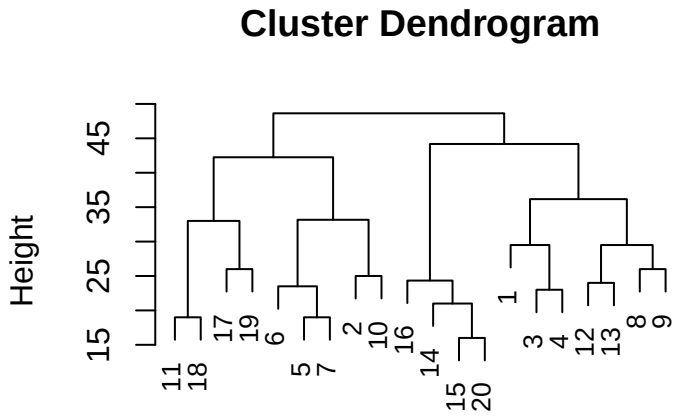
```
plot(dune5.man.ave.clust, xlab="", sub="")
```



Q: Which two sites or groups of sites are most similar or most dissimilar?
Compare with the dendrogram you generated by hand.
Repeat the analysis with different types of linkage techniques; will those give you similar groupings of sites?

3.2 Using the full dune data set

```
dune.man <- dist(dune, "manhattan")
dune.man.ave.clust <- hclust(dune.man, method = "average")
plot(dune.man.ave.clust, xlab="", sub="", cex=.8)
```



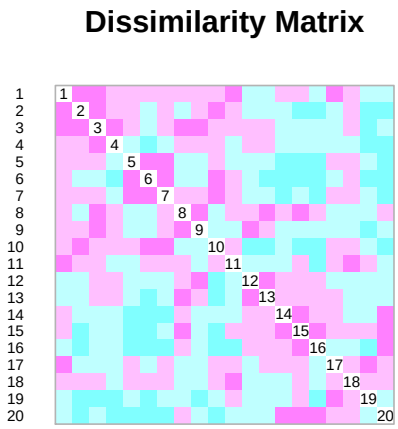
3.2.1 Calculating and visualizing distance measures

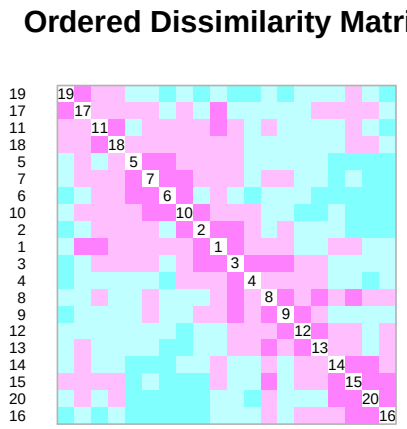
So far, you have only used the Manhattan- (City-Block) distance. Try alternative distances as well, e.g. Euclidean distance. The **gclus** package and the coldiss - function (below) are very handy to visualize distance matrices (or square matrices in general).

```
# Author: Francois Gillet, August 2009

coldiss <- function(D, nc = 4, byrank = TRUE, diag = FALSE)
{
require(gclus)
if (max(D)>1) D <- D/max(D)
if (byrank) {
spe.color = dmat.color(1-D, cm.colors(nc))
}
else {
spe.color = dmat.color(1-D, byrank=FALSE, cm.colors(nc))
}
spe.o = order.single(1-D)
speo.color = spe.color[spe.o,spe.o]
op = par(mfrow=c(1,2), pty="s")
if (diag) {
plotcolors(spe.color, rlabels=attributes(D)$Labels,
main="Dissimilarity Matrix",
dlabels=attributes(D)$Labels)
plotcolors(speo.color, rlabels=attributes(D)$Labels[spe.o],
main="Ordered Dissimilarity Matrix",
dlabels=attributes(D)$Labels[spe.o])
}
else {
plotcolors(spe.color, rlabels=attributes(D)$Labels,
main="Dissimilarity Matrix")
plotcolors(speo.color, rlabels=attributes(D)$Labels[spe.o],
main="Ordered Dissimilarity Matrix")
}
par(op)
}
```

Visualize Manhattan distances (magenta (reddish) - maximally similar; cyan (blueish) - maximally dissimilar):

```
coldiss(dune.man, byrank = FALSE, diag = TRUE)
```

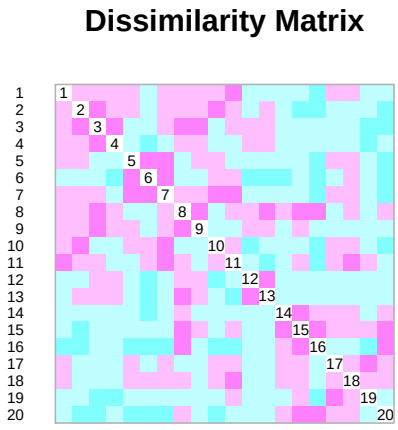


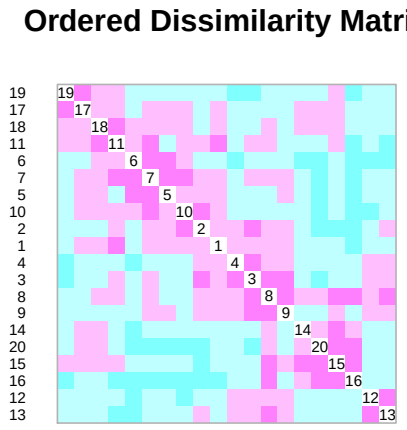


Q: Compare with the Manhattan-distance matrix you calculated by hand?

Now, visualize euclidean distances:

```
dune.euclid <- dist(dune, "euclid")
coldiss(dune.euclid, byrank = FALSE, diag = TRUE)
```





Q: Are the most similar pairs of sites according to Euclidian distances similar to those obtained using Manhattan distance?
FYI: Even more distance metrics can be calculated using the "vegdist"-function in *vegan*.

3.2.2 Comparing distances - the Mantel test

Tests correlations between two distance matrices, the extent to which one matrix resembles another one. This is somehow similar to the Pearson's correlation coeffient (r), but significance testing is done differently

(using permutations).

```
mantel(dune.man, dune.euclid)

##
## Mantel statistic based on Pearson's product-moment correlation
##
## Call:
## mantel(xdis = dune.man, ydis = dune.euclid)
##
## Mantel statistic r: 0.9514
##       Significance: 0.001
##
## Upper quantiles of permutations (null model):
##   90%   95% 97.5%   99%
## 0.105 0.144 0.174 0.214
## Permutation: free
## Number of permutations: 999
```

Q: Are the two distance matrices significantly similar to each other?

3.2.3 Finding the optimal clustering procedure

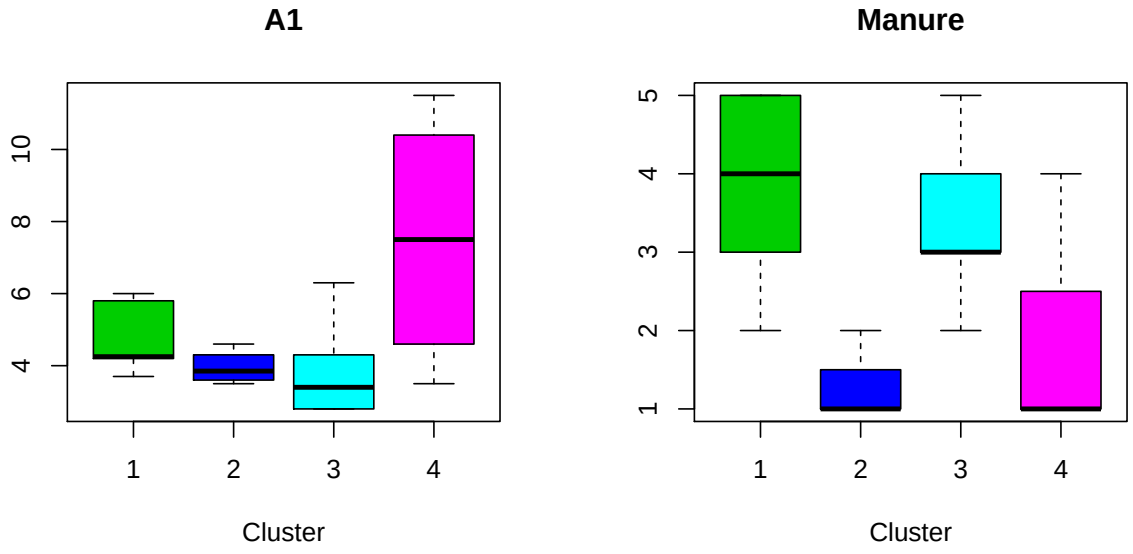
You can use the following function to test among all possible combinations from a range of distance measures, linkage methods and number of clusters to find the optimal one.

```
library(clusterSim)
?cluster.Sim
clustsim <- cluster.Sim(dune, 3, 2, 4)
# 7- mixed data (e.g. intervals, such as the cover classes in the dune dataset)
# 2- minimun number of clusters,
# 4 - maximum number of clusters
clustsim[4:6]

## $distance
##
## "GDM2"
##
## $method
##
## "average"
##
## $classes
## [1] "4"
```

3.3 Linking cluster analysis to environmental data

Another clustering method (K-means clustering) can be used to generate a classification based on a prefined number of clusters (groups). Such groups can then be linked to measured environmental variables in an ANOVA-type of analysis.

```
kclust <- kmeans(dune, centers = 4)

par(mfrow = c(1, 2))
for (i in c(1, 5)) {
    boxplot(dune.env[, i] ~ kclust$cluster, col = 3:6, main = colnames(dune.env)[i],
        xlab = "Cluster")
}
```



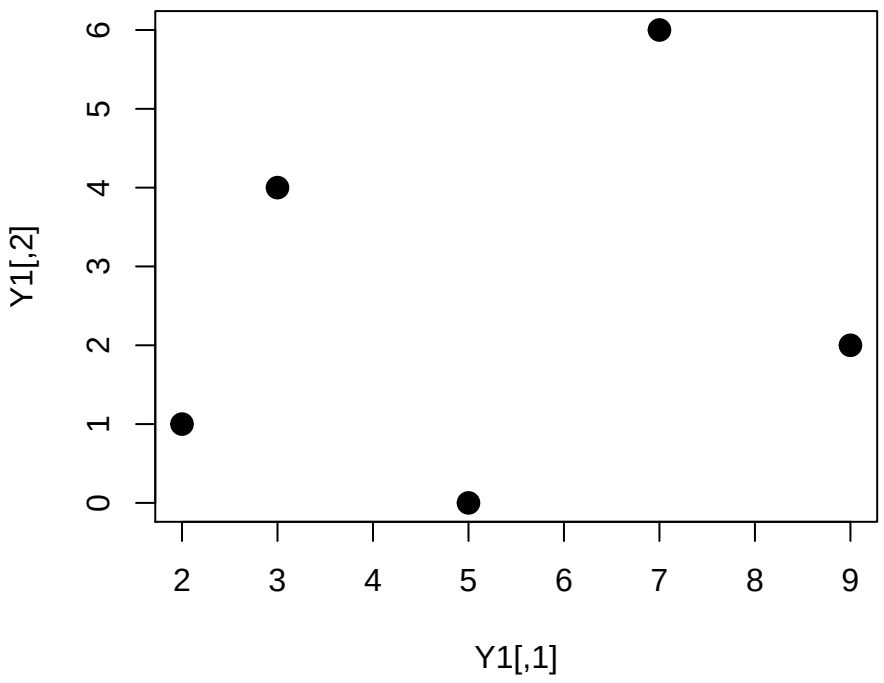
3.4 Advanced: Perform a cluster analysis on the *varespec* data and link it to a continuous descriptor from the varechem data

4 Ordination - Basic steps

4.1 Two species example

Example from Legendre & Legendre (1998), p. 392:

```
Y1 <- matrix(c(2,3,5,7,9,1,4,0,6,2),nrow=5, ncol=2)
Y1

##      [,1] [,2]
## [1,]    2    1
## [2,]    3    4
## [3,]    5    0
## [4,]    7    6
## [5,]    9    2

plot(Y1, pch=21, col="black", bg="black", cex=1.5)
```



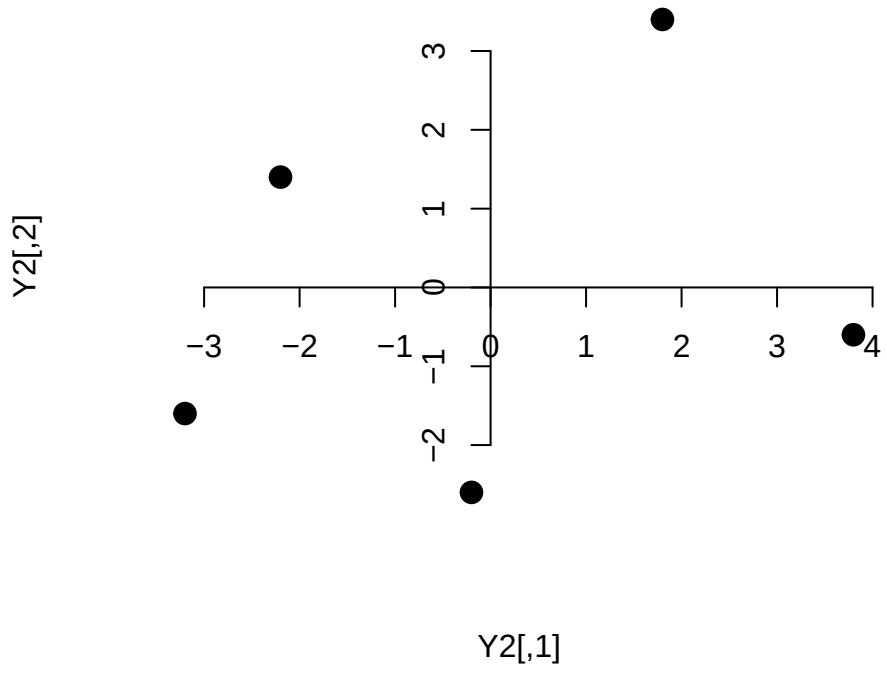
4.2 Centering the attributes in a data matrix

Shift of the origin of the coordinate system into the centre.

```
Y2 <- matrix(0,nrow=5, ncol=2)
Y2[,1] <- Y1[,1]-mean(Y1[,1])
Y2[,2] <- Y1[,2]-mean(Y1[,2])
Y2
```

```
##      [,1] [,2]
## [1,] -3.2 -1.6
## [2,] -2.2  1.4
## [3,] -0.2 -2.6
## [4,]  1.8  3.4
## [5,]  3.8 -0.6
```

```
plot(Y2, pch=21, col="black", bg="black", cex=1.5, bty="n",  xaxt = "n", yaxt = "n")
axis(1, pos=0)
axis(2, pos=0)
```



4.3 Rotation

Rotate the point cloud such that the maximum variance is found on the first axis.

```
dispersion.matrix <- t(Y2) %*% Y2

# divide by n-1
```

```r
dispersion.matrix/4
```

```
##      [,1] [,2]
## [1,]  8.2  1.6
## [2,]  1.6  5.8
```

```r
# or more simple
S <- var(Y2)
S
```

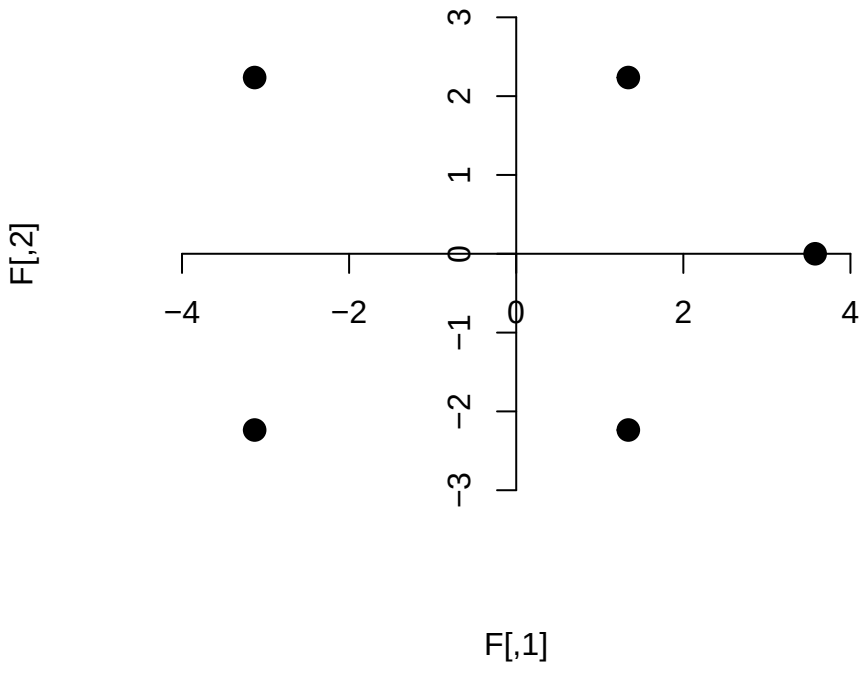
```
##      [,1] [,2]
## [1,]  8.2  1.6
## [2,]  1.6  5.8
```

```r
# get the Eigenvectors
U <- solve(eigen(S)$vectors)

# matrix of eigenvectors U
U
```

```
##            [,1]       [,2]
## [1,] -0.8944272 -0.4472136
## [2,]  0.4472136 -0.8944272
```

```r
F <- Y2 %*% t(U)
```

```r
plot(F, pch = 21, col = "black", bg = "black", cex = 1.5, bty = "n", xaxt = "n",
    yaxt = "n", xlim = c(-4, 4), ylim = c(-3, 3))
axis(1, pos = 0)
axis(2, pos = 0)
```



5 Unconstrained ordination (indirect gradient analysis)

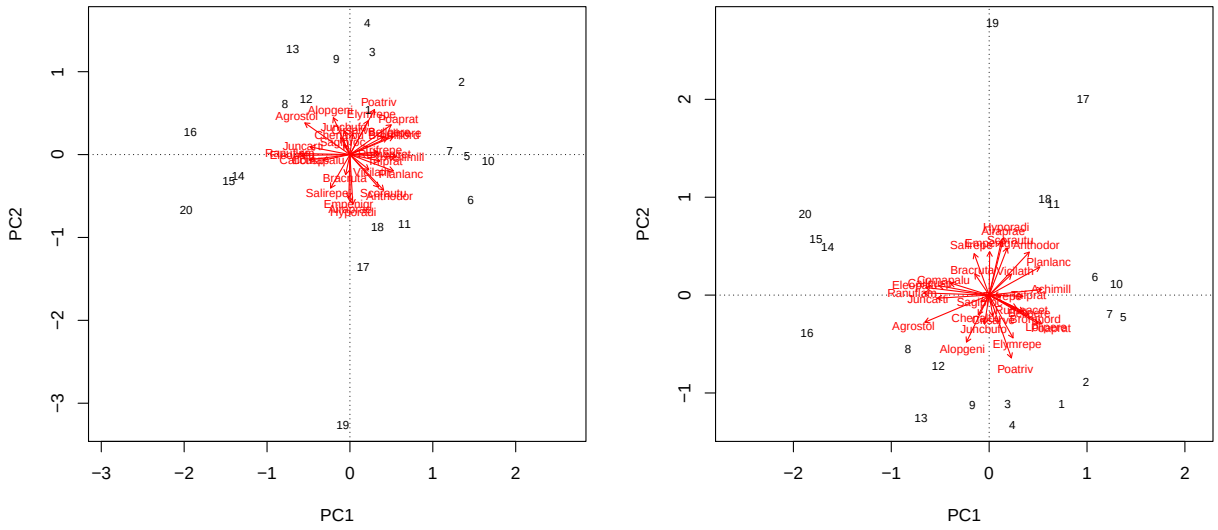
5.1 Principal component analysis - PCA

```
data(dune)
model1<-rda(dune,center=T,scale=T)
```

Remember, because it is based on Euclidean distances but species similarities among sites based on species information are not strikly Euclidean, PCA is not really suitable for species cover values. However, appropiate transformations (such as Hellinger-transformation) can circumvent this problem.

```
model2 <- rda(decostand(dune, "hell"),center=T,scale=T)
```

```
par(mfrow=c(1,2))
biplot(model1)
biplot(model2)
```



Q: Does the ordination based on untransformed species data yield different configurations of sites than the ordination based on Hellinger-transformed species data?

You can also print the summary of the eigenvalues which gives you the eigenvalues, proportion explained and cumulative proportion explained by the ordination axes (principal components):

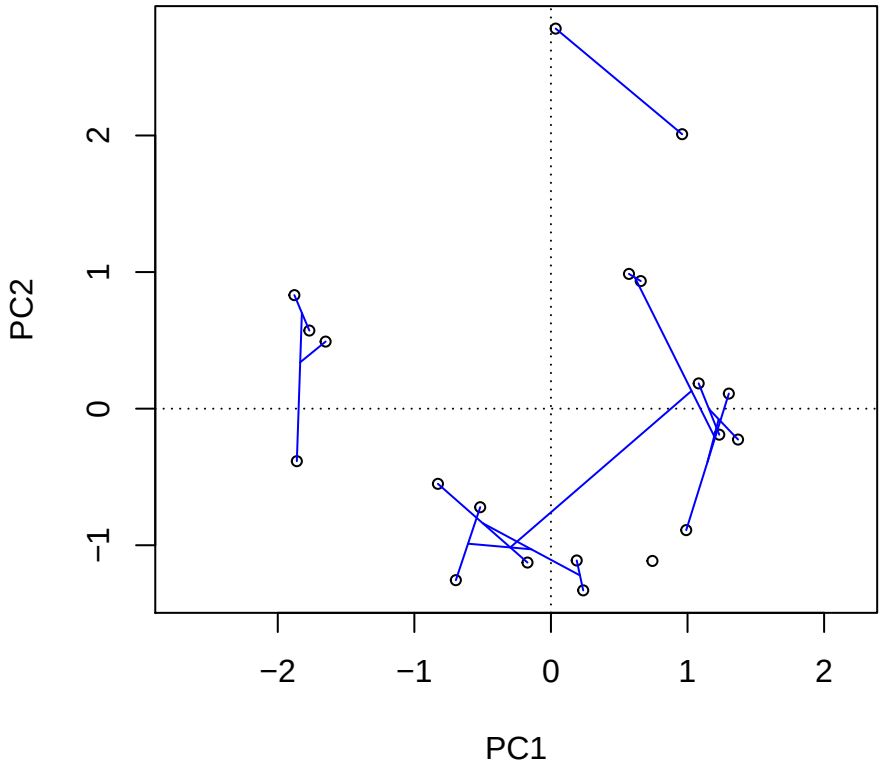
```
summary(eigenvals(model2))

## Importance of components:
##                           PC1    PC2    PC3     PC4     PC5     PC6     PC7
## Eigenvalue            7.141 5.1999 3.2739 2.72312 2.42247 1.86689 1.62155
## Proportion Explained  0.238 0.1733 0.1091 0.09077 0.08075 0.06223 0.05405
## Cumulative Proportion 0.238 0.4114 0.5205 0.61127 0.69202 0.75425 0.80830
##                           PC8     PC9   PC10    PC11    PC12    PC13
## Eigenvalue            1.17624 1.03552 0.8011 0.65005 0.59471 0.53224
## Proportion Explained  0.03921 0.03452 0.0267 0.02167 0.01982 0.01774
## Cumulative Proportion 0.84751 0.88203 0.9087 0.93040 0.95022 0.96796
##                          PC14    PC15    PC16    PC17    PC18    PC19
## Eigenvalue            0.35672 0.23330 0.15034 0.11330 0.05853 0.04893
## Proportion Explained  0.01189 0.00778 0.00501 0.00378 0.00195 0.00163
## Cumulative Proportion 0.97985 0.98763 0.99264 0.99642 0.99837 1.00000
```

Q: Interpret the summary of the eigenvectors of a PCA on the Hellinger-transformed species data.
What is the percentage of total variation explained by each of first three ordination axes separately?
What is the percentage of total variation explained by the first three ordination axes in conjunction?

5.1.1 Combining clustering and ordination results

You can also project a dendrogram from the cluster analysis onto an ordination, to see whether sites that form a cluster in the dendrogram (i.e. are connected by blue lines) also form clusters in the ordination. And vice versa, whether sites that cluster in the ordination form a cluster in the dendrogram (i.e. are connected by blue lines).

```
plot(model2, type = "p", display="sites")
ordicluster(model2, hclust(vegdist(decostand(dune, "hell"))), prune=3, col = "blue")
```



Q: Are the ordination and clustering results consistent? If not, what could be the reason?

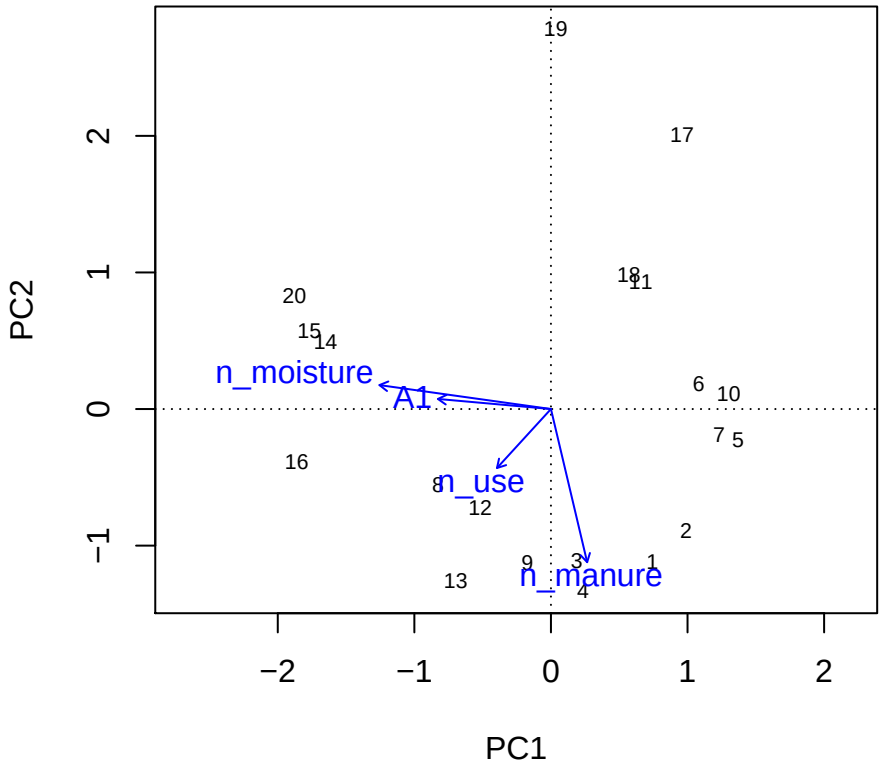
5.1.2 Fitting environmental vectors onto an ordination

The interpretability of ordination plots can be enhanced by adding environmental information to an existing ordination plot.

```
data(dune.env)

dune.env$n_moisture <- as.numeric(dune.env$Moisture)
dune.env$n_use <- as.numeric(dune.env$Use)
dune.env$n_manure <- as.numeric(dune.env$Manure)

fit <- envfit(model2, dune.env[,c(1,6,7,8)], perm = 1000)
fit

##
## ***VECTORS
##
##                 PC1      PC2     r2   Pr(>r)
```

```
## A1          -0.99604  0.08892 0.3377 0.028971 *
## n_moisture -0.99039  0.13828 0.7860 0.000999 ***
## n_use       -0.67507 -0.73776 0.1675 0.218781
## n_manure     0.23020 -0.97314 0.6493 0.000999 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
## Permutation: free
## Number of permutations: 1000
```

```
plot(model2,display=c("sites"))
plot(fit)
```



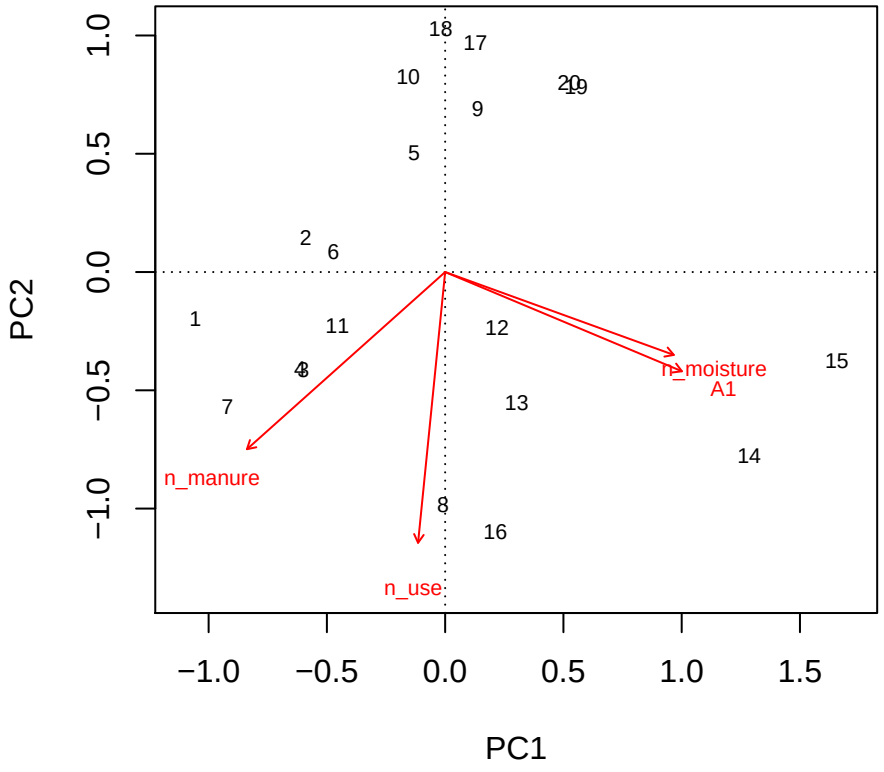
Q: Are the two ordination axes correlated to particular environmental gradients? Identify sets of sites that have high or low values for a particular environmental variable.

5.1.3 PCA on environmental data

Here, you use PCA on environmental variables to (i) reduce the multiple environmental variables into only few uncorrelated axes (dimensions) and (ii) explore correlation between environmental descriptors. Because variables are measured in different units, you need to scale the environmental variables prior to analysis.

```
n_env <- dune.env[,c(1,6,7,8)]
environ <- rda(n_env,center=T,scale=T)
```

```r
biplot(environ)
```




```r
summary(eigenvals(environ))

## Importance of components:
##                           PC1    PC2    PC3     PC4
## Eigenvalue            1.684 1.3781 0.5579 0.38005
## Proportion Explained  0.421 0.3445 0.1395 0.09501
## Cumulative Proportion 0.421 0.7655 0.9050 1.00000

scores(environ, display = "sp")

##                   PC1        PC2
## A1          1.1776484 -0.4935640
## n_moisture  1.1372860 -0.4122517
## n_use      -0.1347561 -1.3466378
## n_manure   -0.9857486 -0.8811834
## attr(,"const")
## [1] 2.952592
```

Q: In the ordination biplot, which variables are highly correlated or uncorrelated with each other?
Look at the summary output. How much of the total variance in the environmental data is explained by the first two axes individually and cummulatively?

Which environmental gradients are represented by the first and second ordination axis, i.e. which variables have very high or very low scores on the respective axis)?

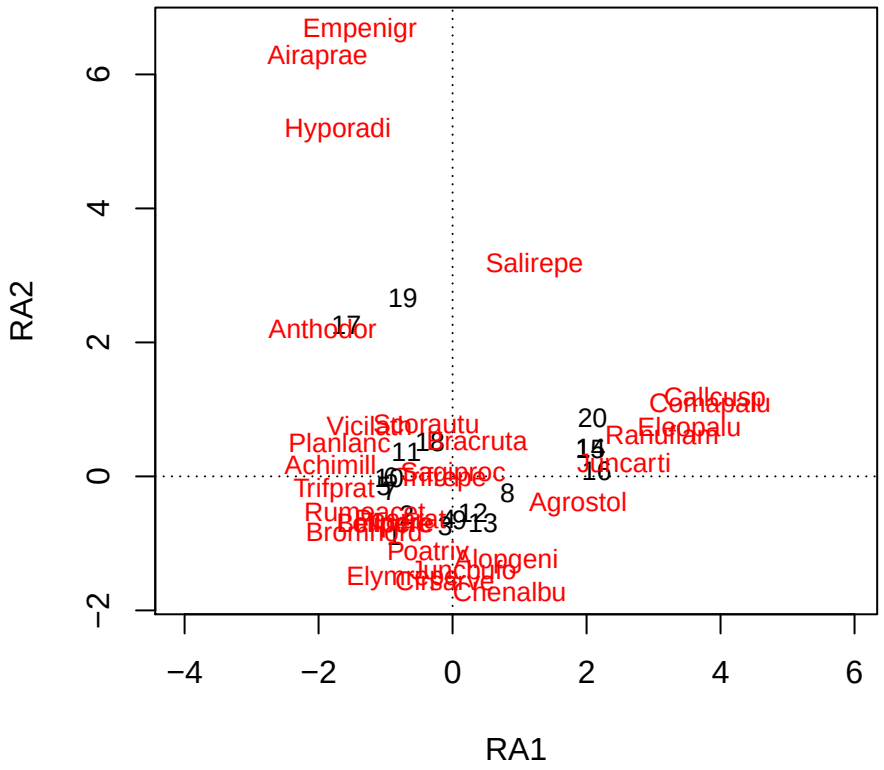
5.2 Correspondence analysis - CA

Correspondence analysis has been shown be a more robust method where species show unimodal, rather than linear, responses to an underlying environmental gradient.

```
CA1 <- decorana(dune,ira=1)
```

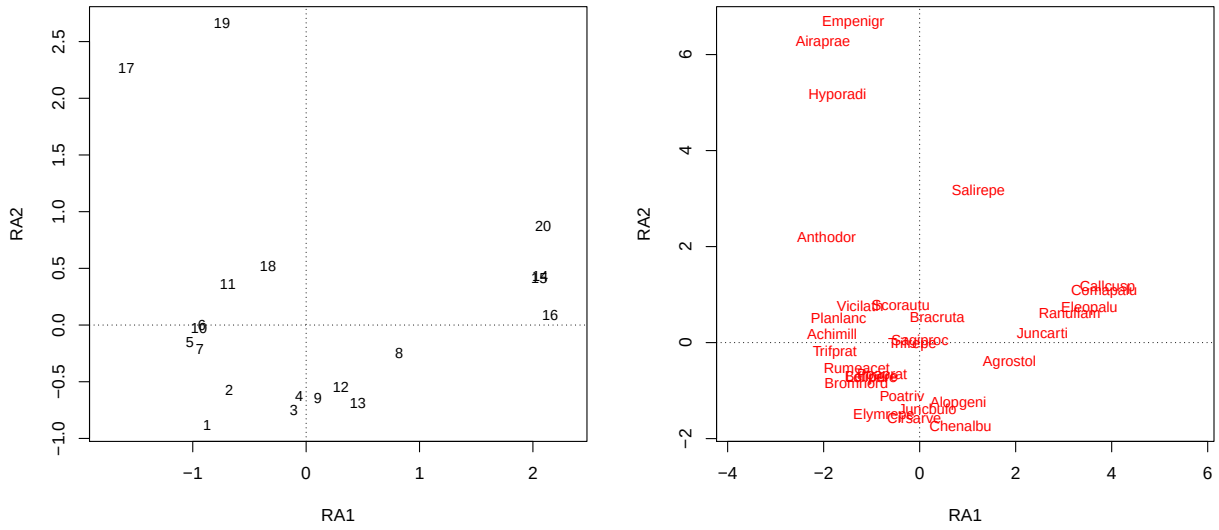
The "ira" argument defines whether standard or detrended correspondence analysis is carried out. ira=1 basic reciprocal averaging = correspondence analysis= CA ira=0, default, detrended correspondence analysis = DCA

```
plot(CA1)
```



Note that site and species scores are at the same scale.

```
par(mfrow=c(1,2))
plot(CA1,display=c("sites"))
plot(CA1,display=c("species"))
```

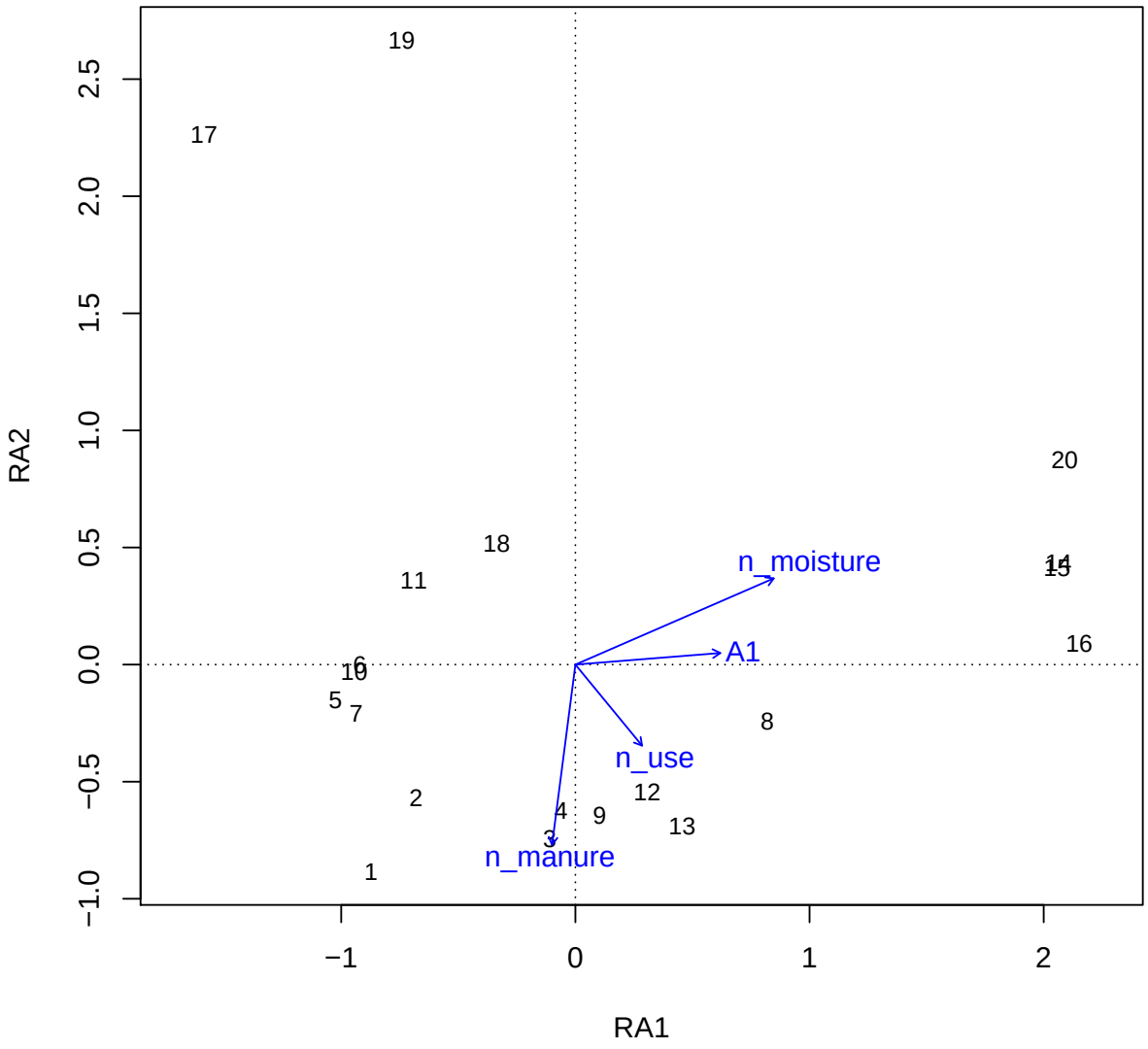



```
print(CA1)

##
## Call:
## decorana(veg = dune, ira = 1)
##
## Orthogonal correspondence analysis.
##
##                RA1    RA2    RA3   RA4
## Eigenvalues 0.536 0.4001 0.2598 0.176
```

Q: What are the eigenvalue for axes 1 and 2?

```
fit <- envfit(CA1, n_env, perm = 1000)
fit

##
## ***VECTORS
##
##                 RA1      RA2     r2   Pr(>r)
## A1          0.99682  0.07966 0.3104 0.035964 *
## n_moisture  0.91712  0.39860 0.6868 0.000999 ***
## n_use       0.63528 -0.77228 0.1616 0.179820
## n_manure   -0.12828 -0.99174 0.4848 0.000999 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
## Permutation: free
## Number of permutations: 1000

scores(fit, "vectors")

##                    RA1         RA2
## A1          0.55535366  0.04437778
## n_moisture  0.76007557  0.33034557
```

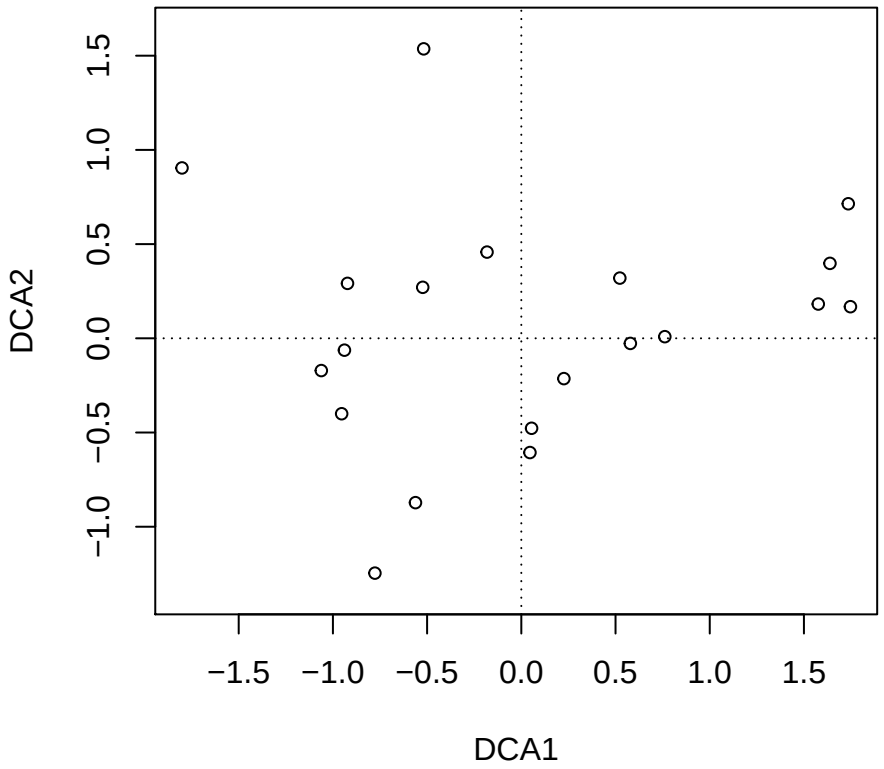
```
## n_use       0.25540159 -0.31047919
## n_manure   -0.08931823 -0.69053694

plot(CA1,display=c("sites"))
plot(fit, col = "blue")
```



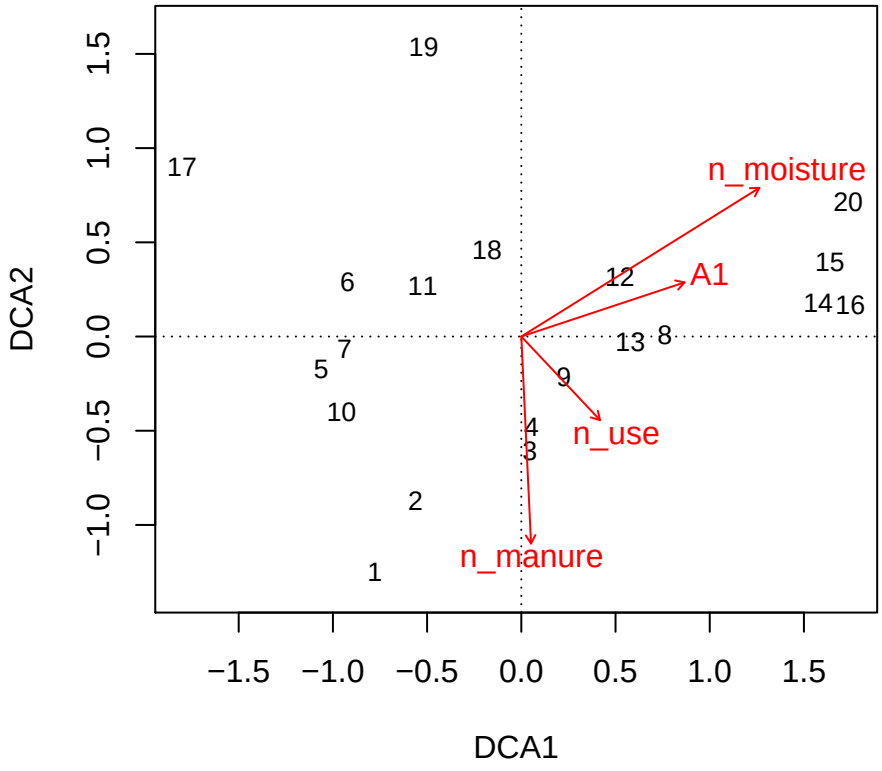
5.3 Detrended correspondence analysis - DCA

Correspondence analysis (CA) along very long environmental gradients is prone to the "arch-effect", where objects in ordination space show curved patterns. Detrended correspondence analysis (DCA) can deal with this issue.

```
DCA1<-decorana(dune, ira=0, iweigh=1)
print(DCA1)

##
## Call:
## decorana(veg = dune, iweigh = 1, ira = 0)
##
## Detrended correspondence analysis with 26 segments.
## Rescaling of axes with 4 iterations.
## Downweighting of rare species from fraction 1/5.
##
##                   DCA1   DCA2    DCA3    DCA4
## Eigenvalues     0.5033 0.2632 0.12978 0.10773
## Decorana values 0.5296 0.2243 0.04982 0.03053
## Axis lengths    3.5478 2.7827 1.48812 1.24633

plot(DCA1,display=c("sites"),type="points")
```

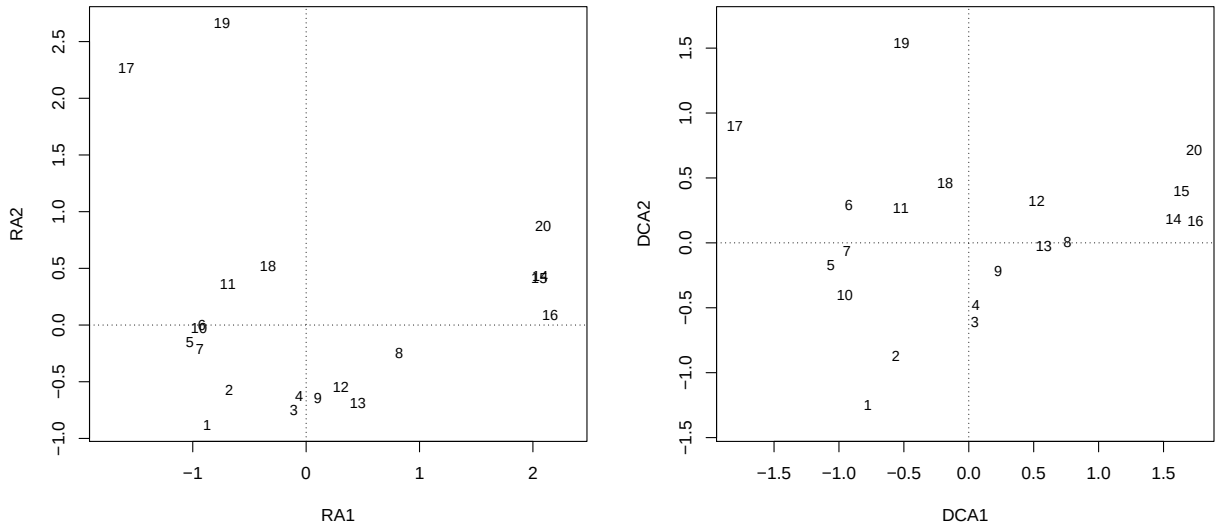


```
fit3<- envfit(DCA1, n_env, perm = 1000)
plot(DCA1,display=c("sites"))
plot(fit3, col = "red")
```



Q: Plot CA and DCA beside each other and check whether the arch(horse-shoe)-effect disappears in DCA.

```
par(mfrow=c(1,2))
plot(CA1,display=c("sites"))
plot(DCA1,,display=c("sites"))
```

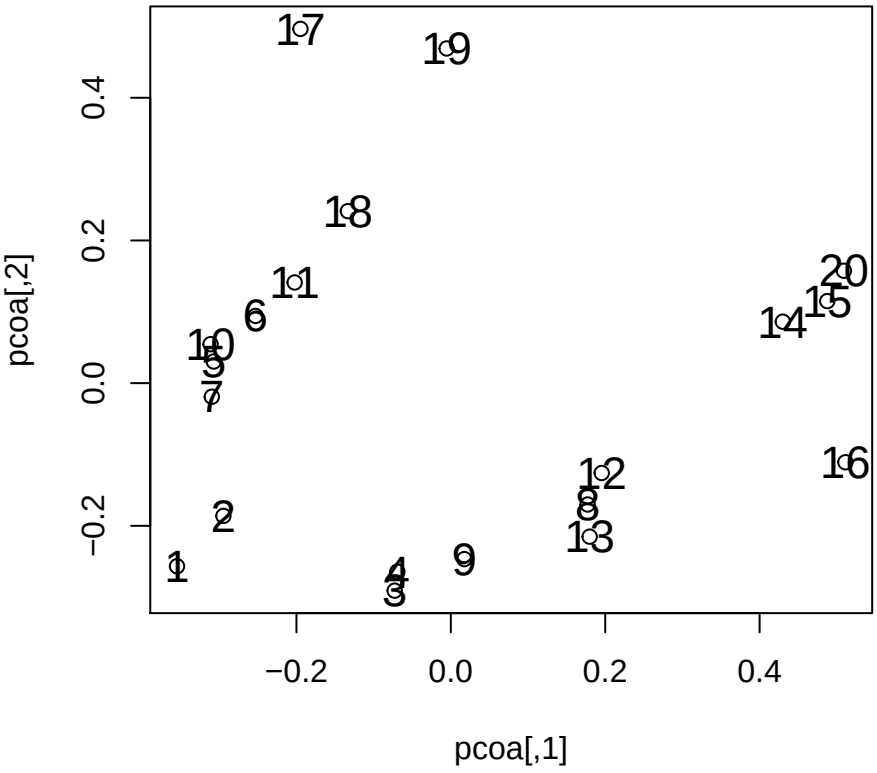


5.4 Principal coordinates analysis - PCoA

While PCA assumes Euclidean distances and CA (as well as DCA) Chi-Squares distances between objects, principal coordinates analysis (PCoA) is more flexible in this respect as it allows for non-Euclidean distances.

```
veg.dist <- vegdist(dune, method="bray")
pcoa <- cmdscale(veg.dist)
```

```
plot(pcoa)
text(pcoa[,1], pcoa[,2], rownames(pcoa), cex = 1.4)
```



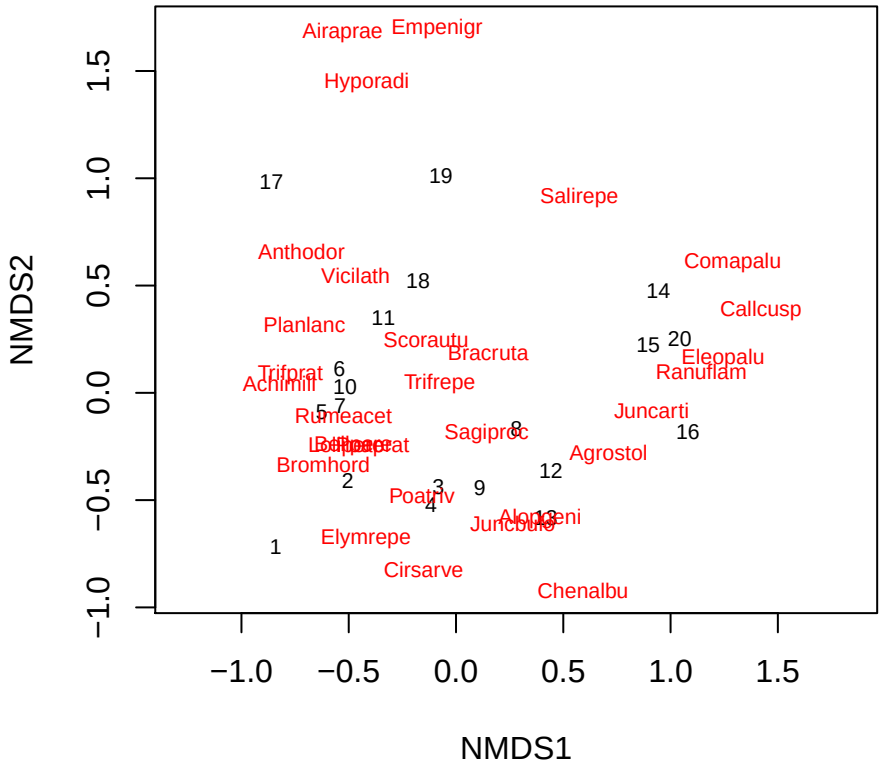
5.5 Nonmetric multidimensional scaling - NMDS

Nonmetric multidimensional scaling (NMDS) is even more flexible than PCoA, as it can find non-linear relationships (unlike other methods), while being able to handle all kinds of distances. However, you may have to run it several times to reach a good "solution". NMDS runs for a VERY long time on huge data sets.

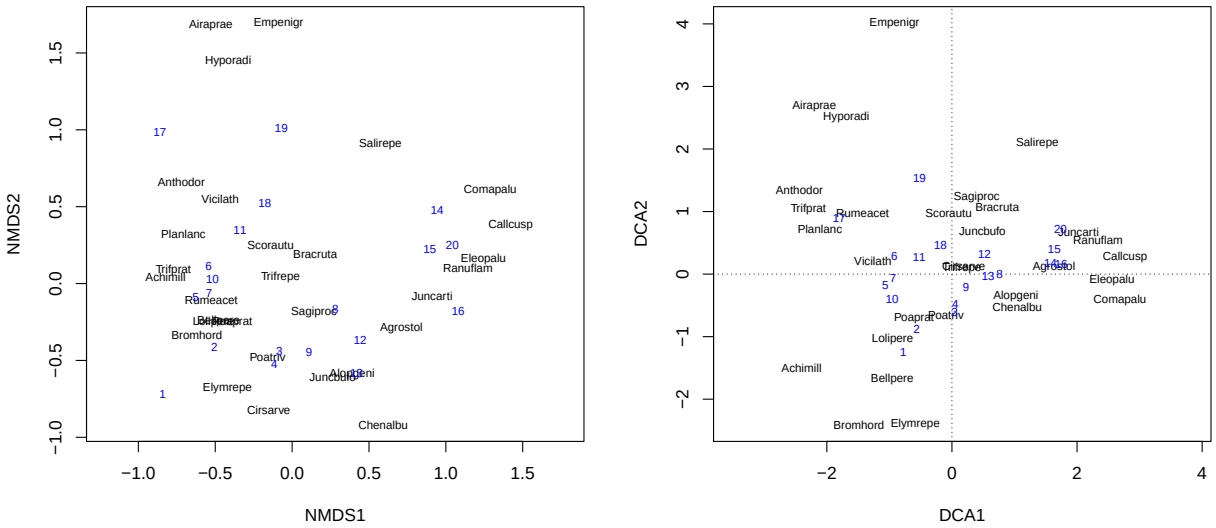
```
nmds <- metaMDS(dune)

## Run 0 stress 0.1192678
## Run 1 stress 0.1812938
## Run 2 stress 0.2075713
## Run 3 stress 0.1183186
## ... New best solution
## ... Procrustes: rmse 0.02027194  max resid 0.06496373
## Run 4 stress 0.1192678
## Run 5 stress 0.1192679
## Run 6 stress 0.1183186
## ... New best solution
## ... Procrustes: rmse 2.00406e-05  max resid 5.942974e-05
## ... Similar to previous best
## Run 7 stress 0.2003482
## Run 8 stress 0.1183186
## ... Procrustes: rmse 7.455498e-05  max resid 0.0001759786
```

```
## ... Similar to previous best
## Run 9 stress 0.1183186
## ... Procrustes: rmse 1.553252e-05  max resid 5.593033e-05
## ... Similar to previous best
## Run 10 stress 0.1192678
## Run 11 stress 0.1183186
## ... Procrustes: rmse 1.626864e-05  max resid 5.213352e-05
## ... Similar to previous best
## Run 12 stress 0.1192686
## Run 13 stress 0.1809578
## Run 14 stress 0.1192684
## Run 15 stress 0.1886532
## Run 16 stress 0.1192679
## Run 17 stress 0.119268
## Run 18 stress 0.1183186
## ... Procrustes: rmse 9.306454e-05  max resid 0.0002789961
## ... Similar to previous best
## Run 19 stress 0.1812942
## Run 20 stress 0.1183186
## ... Procrustes: rmse 8.355486e-05  max resid 0.0002532058
## ... Similar to previous best
## *** Solution reached

nmds

##
## Call:
## metaMDS(comm = dune)
##
## global Multidimensional Scaling using monoMDS
##
## Data:     dune
## Distance: bray
##
## Dimensions: 2
## Stress:     0.1183186
## Stress type 1, weak ties
## Two convergent solutions found after 20 tries
## Scaling: centring, PC rotation, halfchange scaling
## Species: expanded scores based on 'dune'
```

```
plot(nmds, type="t")
```




```
par(mfrow=c(1,2))
plot(nmds, type="n")
text(nmds, dis="species",cex=0.7)
#points(solnmds, pch=21, col="red", bg="yellow", cex=1.2)
text(nmds, "sites", col="blue", cex=0.7)
plot(DCA1,type="n")
text(DCA1, dis="species",cex=0.7)
text(DCA1, dis="sites", col="blue",cex=0.7)
```

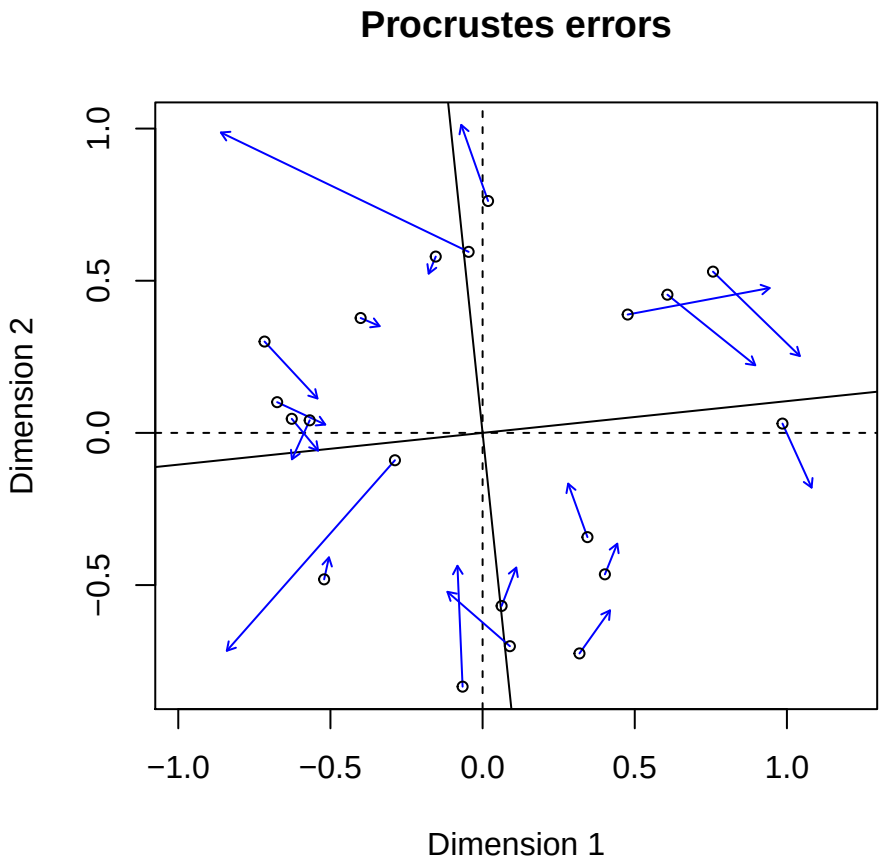


5.6 Comparing ordinations using Procrustes rotation

You can check whether two ordination are really different (i.e. have different configurations) from each other or whether differences between ordinations are just a reflection of different scalings and rotations. Here we use the Procrustes-Rotation-Test to compare the NMDS ordination with PCA (on the Hellinger-transformed species data), and test whether the two ordinations are significantly similar.

```
nmds2 <- metaMDS(dune)

## Run 0 stress 0.1192678
## Run 1 stress 0.1192678
## ... New best solution
## ... Procrustes: rmse 3.697152e-05  max resid 0.0001106741
## ... Similar to previous best
## Run 2 stress 0.1183186
## ... New best solution
## ... Procrustes: rmse 0.0202829  max resid 0.06505109
## Run 3 stress 0.1183186
## ... Procrustes: rmse 3.742428e-05  max resid 0.0001073725
## ... Similar to previous best
## Run 4 stress 0.1183186
## ... Procrustes: rmse 4.915908e-06  max resid 1.398718e-05
## ... Similar to previous best
## Run 5 stress 0.1183186
## ... New best solution
## ... Procrustes: rmse 1.981371e-05  max resid 5.996904e-05
## ... Similar to previous best
## Run 6 stress 0.1886536
## Run 7 stress 0.1183186
## ... Procrustes: rmse 3.51303e-05  max resid 0.0001146371
## ... Similar to previous best
## Run 8 stress 0.1192681
## Run 9 stress 0.1192679
## Run 10 stress 0.1192679
```

```
## Run 11 stress 0.1922241
## Run 12 stress 0.1183186
## ... Procrustes: rmse 4.974337e-05  max resid 0.0001490783
## ... Similar to previous best
## Run 13 stress 0.1183186
## ... New best solution
## ... Procrustes: rmse 2.01796e-05  max resid 6.444703e-05
## ... Similar to previous best
## Run 14 stress 0.1192678
## Run 15 stress 0.192224
## Run 16 stress 0.1808911
## Run 17 stress 0.1192684
## Run 18 stress 0.1183186
## ... New best solution
## ... Procrustes: rmse 2.446088e-05  max resid 8.046263e-05
## ... Similar to previous best
## Run 19 stress 0.1192688
## Run 20 stress 0.1886532
## *** Solution reached

pca2 <- rda(dune)

protest.nmds.pca <- protest(nmds2, pca2)
protest.nmds.pca

##
## Call:
## protest(X = nmds2, Y = pca2)
##
## Procrustes Sum of Squares (m12 squared):        0.2165
## Correlation in a symmetric Procrustes rotation: 0.8852
## Significance:  0.001
##
## Permutation: free
## Number of permutations: 999

pro.nmds.pca <- procrustes(nmds2, pca2)

plot(pro.nmds.pca)
```



The following plot (Procrustes superimposition plot) shows the differences between the NMDS ordination configuration (cicles) and those that from a PCA (arrow tips).

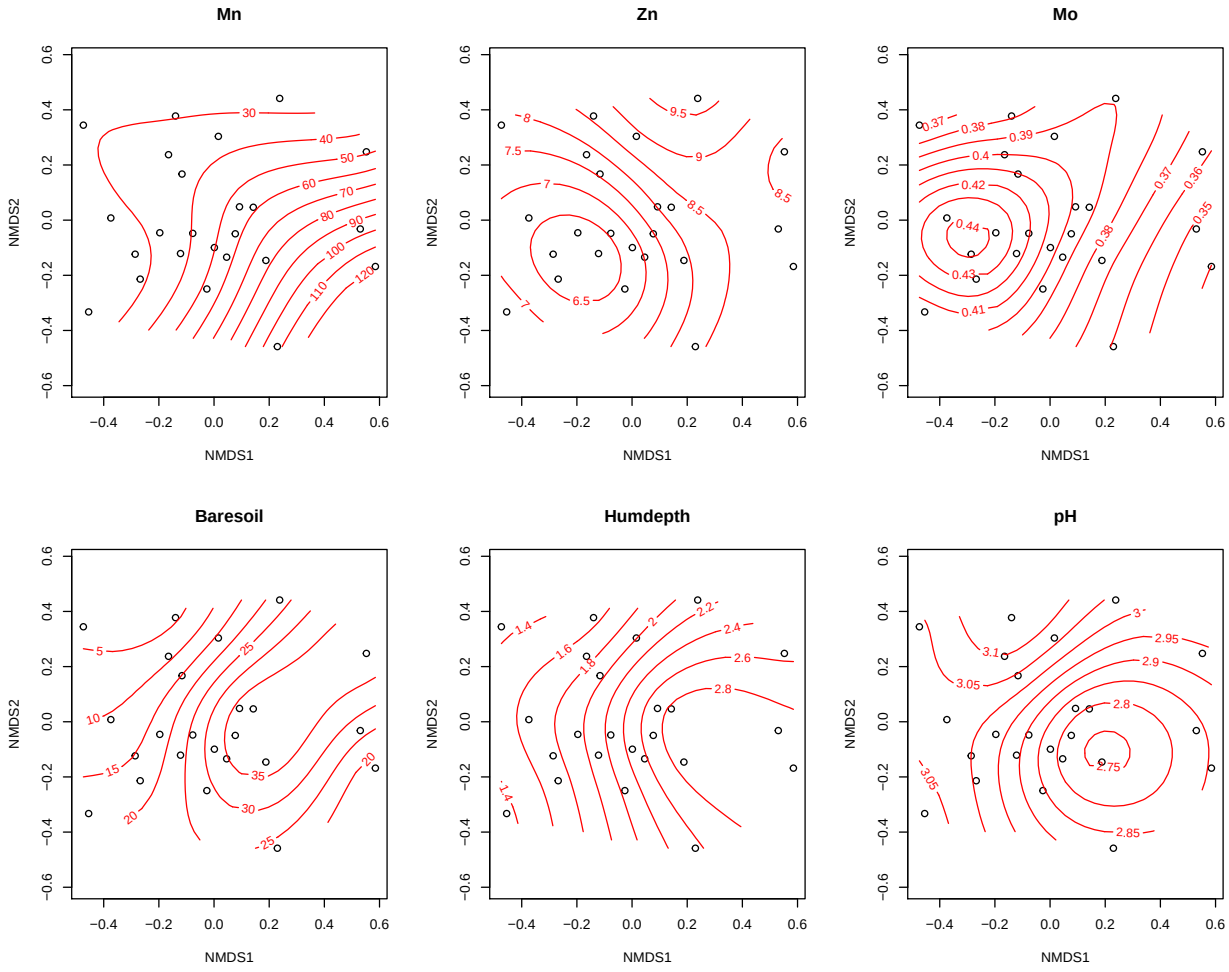
5.7 Surface plotting

So far, we have only visualized linear relationships (expressed as arrows) between the objects in an ordination and environmental variables using the envfit-function, but community responses along environmental gradients are often non-linear.

```
data(varespec)
data(varechem)
nmds <- metaMDS(varespec)

## Square root transformation
## Wisconsin double standardization
## Run 0 stress 0.1843196
## Run 1 stress 0.2290578
## Run 2 stress 0.2114476
## Run 3 stress 0.2048307
## Run 4 stress 0.2218505
## Run 5 stress 0.2251602
## Run 6 stress 0.2414245
## Run 7 stress 0.1825658
## ... New best solution
```

```
## ... Procrustes: rmse 0.04169113  max resid 0.152112
## Run 8 stress 0.2061124
## Run 9 stress 0.195049
## Run 10 stress 0.1845802
## Run 11 stress 0.1967395
## Run 12 stress 0.1969805
## Run 13 stress 0.212657
## Run 14 stress 0.2174195
## Run 15 stress 0.2161251
## Run 16 stress 0.1825664
## ... Procrustes: rmse 0.0001286619  max resid 0.0004064493
## ... Similar to previous best
## Run 17 stress 0.2187619
## Run 18 stress 0.1967393
## Run 19 stress 0.1948413
## Run 20 stress 0.2079056
## *** Solution reached
```

```
par(mfrow=c(2,3))
for (i in 9:14) {
ordisurf(nmds, varechem[,i], main = colnames(varechem)[i])
}
```



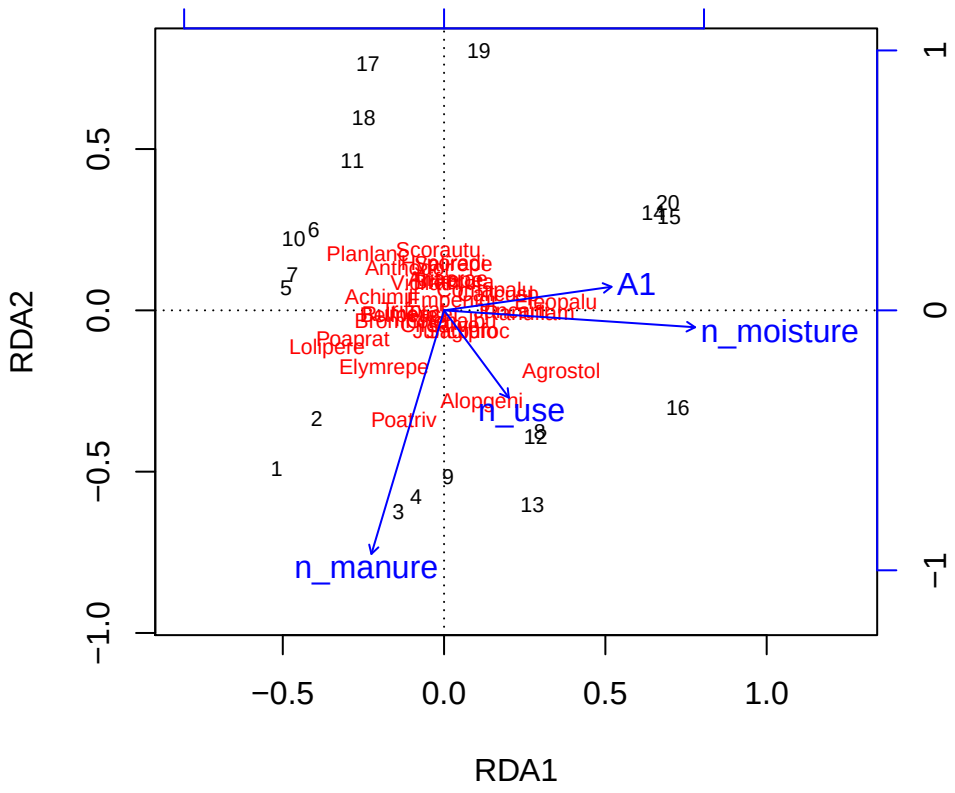
6 Constrained ordination (direct gradient analysis)

Until now, you have performed unconstrained ordination methods, a passive form of gradient analysis, with a-posteriori interpretation of environmental variables. Now, you will run constrained ordination, an approach that directly explores the relationships between community composition and environmental variables.

6.1 Redundancy analysis - RDA

RDA combines regression and PCA. Remember, when running PCA-based methods on species-data, we first have to Hellinger-transform the community data so that it represents Euclidean space.

```
dune.hel <- decostand(dune, "hell")
rda.dune <- rda(dune.hel, dune.env[,c(1,6:8)])
plot(rda.dune)
```




```
print(rda.dune)

## Call: rda(X = dune.hel, Y = dune.env[, c(1, 6:8)])
##
##               Inertia Proportion Rank
## Total          0.5559     1.0000
## Constrained    0.2628     0.4728    4
## Unconstrained  0.2931     0.5272   15
## Inertia is variance
##
## Eigenvalues for constrained axes:
##    RDA1    RDA2    RDA3    RDA4
## 0.14743 0.07987 0.02341 0.01213
##
## Eigenvalues for unconstrained axes:
##     PC1     PC2     PC3     PC4     PC5     PC6     PC7     PC8     PC9
## 0.06120 0.05272 0.04010 0.03481 0.02292 0.02051 0.01655 0.01324 0.00948
##    PC10    PC11    PC12    PC13    PC14    PC15
## 0.00701 0.00602 0.00336 0.00295 0.00142 0.00078
```

Q: Have a look at the summary-output:
What is the total variance (inertia) in the data set?
What is the proportation of variance explained by the environmental variables (constrained proportion)?

What is the proportation of variance that remains unexplained (unconstrained proportion)?

```
summary(eigenvals(rda.dune))

## Importance of components:
##                           RDA1    RDA2    RDA3    RDA4    PC1     PC2
## Eigenvalue             0.1474 0.07987 0.02341 0.01213 0.0612 0.05272
## Proportion Explained   0.2652 0.14368 0.04211 0.02181 0.1101 0.09485
## Cumulative Proportion 0.2652 0.40889 0.45099 0.47281 0.5829 0.67774
##                            PC3     PC4     PC5     PC6     PC7     PC8
## Eigenvalue             0.04010 0.03481 0.02292 0.02051 0.01655 0.01324
## Proportion Explained   0.07214 0.06261 0.04123 0.03690 0.02976 0.02381
## Cumulative Proportion 0.74988 0.81249 0.85372 0.89062 0.92039 0.94420
##                             PC9     PC10     PC11     PC12     PC13
## Eigenvalue             0.009482 0.007006 0.006018 0.003362 0.002953
## Proportion Explained   0.017060 0.012600 0.010820 0.006050 0.005310
## Cumulative Proportion 0.961250 0.973860 0.984680 0.990730 0.996040
##                            PC14      PC15
## Eigenvalue             0.001421 0.0007789
## Proportion Explained   0.002560 0.0014000
## Cumulative Proportion 0.998600 1.0000000
```

Q: Have a look at the summary of the eigenvalues:
What is the proportation of variance explained by the first 2 constrained (RDA) axes?
What is the proportation of variance explained by the first 2 unconstrained (PC) axes for the residuals?
In the biplot-scores, which environmental variables are important on RDA-axis 1 and 2?
In the ordination-plot, which species are associated with particular management types, moisture levels etc.?

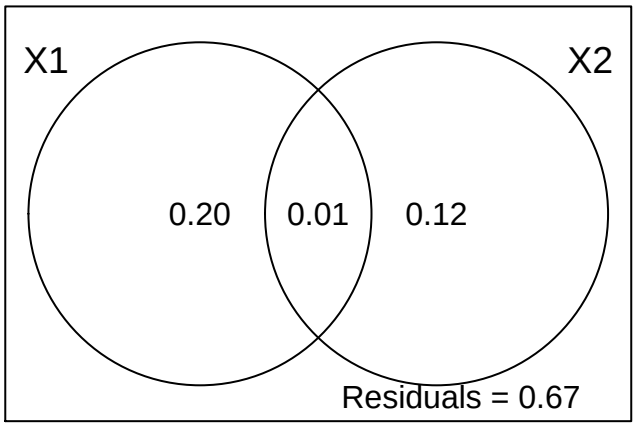
6.1.1 Variation partitioning

You may also be interested in quantifying the extent to which variation in community composition is explained by different sets of variables, e.g. set 1: (soil depth (A1) and moisture (n.moisture) and set 2: land use intensity (n.use) and fertilization (manure)). Variation partitioning analysis allows partialling out the shared and unique contributions of the different sets of predictors in explaining community composition.

```
soil <- dune.env[, c(1, 6)] # set 1
management <- dune.env[, c(7:8)] # set 2

dune.part <- varpart(dune.hel, soil, management)
print(dune.part)
```

```
##
## Partition of variance in RDA
##
## Call: varpart(Y = dune.hel, X = soil, management)
##
## Explanatory tables:
## X1:  soil
## X2:  management
##
## No. of explanatory tables: 2
## Total variation (SS): 10.562
##              Variance: 0.5559
## No. of observations: 20
##
## Partition table:
##                       Df R.squared Adj.R.squared Testable
## [a+b] = X1             2   0.29705       0.21435     TRUE
## [b+c] = X2             2   0.22121       0.12958     TRUE
## [a+b+c] = X1+X2        4   0.47281       0.33222     TRUE
## Individual fractions
## [a] = X1|X2            2                 0.20264     TRUE
## [b]                    0                 0.01171    FALSE
## [c] = X2|X1            2                 0.11787     TRUE
## [d] = Residuals                          0.66778    FALSE
## ---
## Use function 'rda' to test significance of fractions of interest
```

```
plot(dune.part)
```



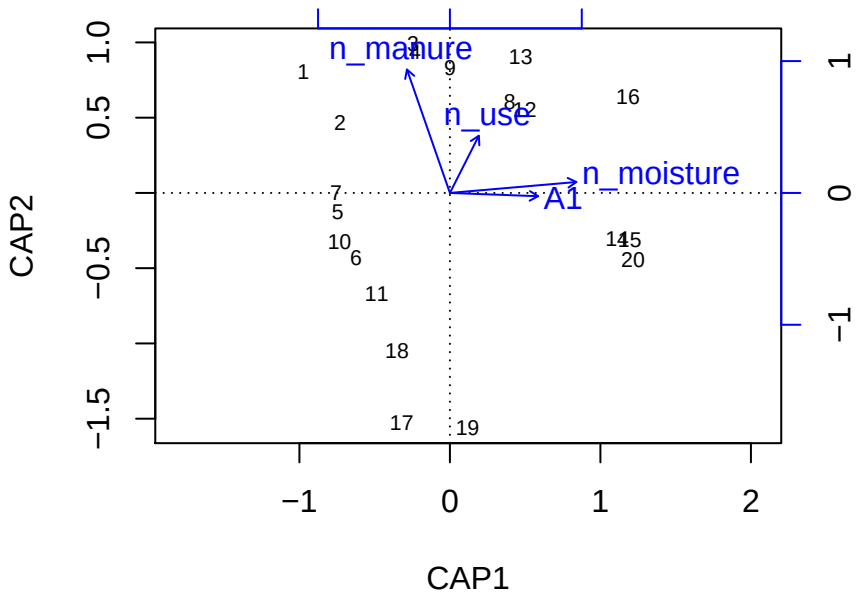
Q: What is the unique contribution (individual fraction) the two variable groups (X1: soil and X2: management) in explaining community composition?
What percentage of the total variation remains unexplained?

6.2 Distance-based redundancy analysis - dbRDA

Unlike RDA, which is based on Euclidean distances, distance-based redundancy analysis (dbRDA) can take all kinds of ecologically meaningful distances (e.g. Bray-Curtis), as it is based on principal coordinates analysis (PCoA).

```
dune.bray <- vegdist(dune, "bray")
dune.db.rda <- capscale(dune.bray ~ A1 + n_moisture + n_use + n_manure,  dune.env)
```

```
plot(dune.db.rda)
```



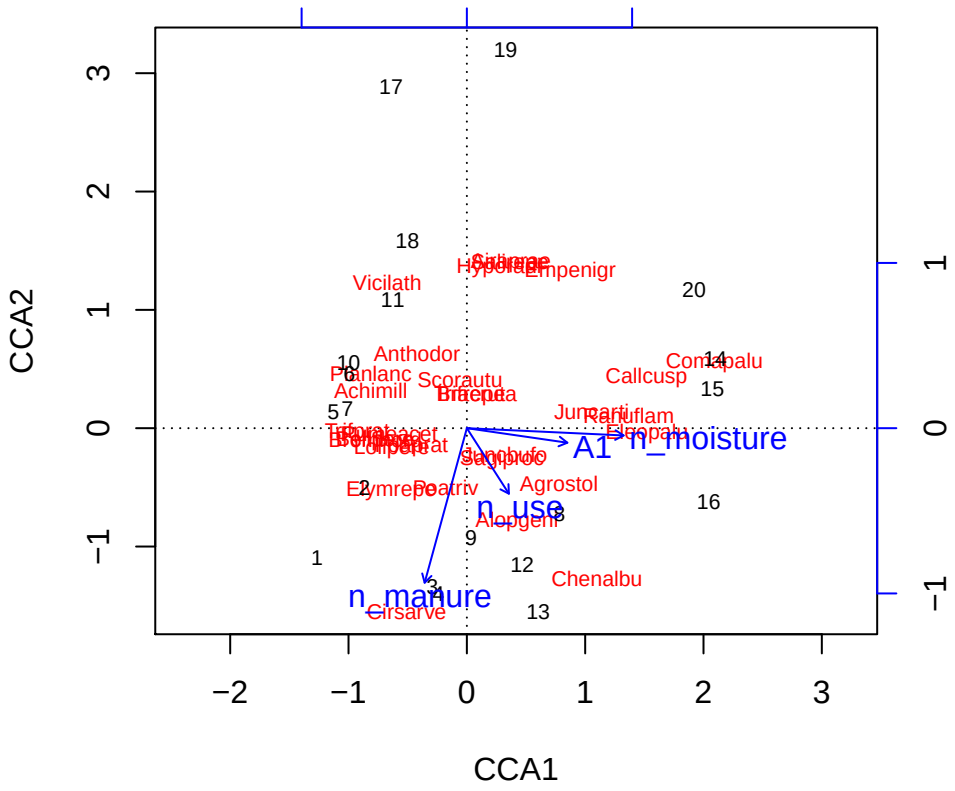
6.3 Canonical correspondence analysis - CCA

Canonical correspondence analysis (CCA) is the constrained version of CA, thus is based on Chi-square distances. Unlike RDA, CCA does not allow for variation partioning analysis and doesn't have the flexibility of incorporating different types of distances (as can be done with db-RDA). CCA-output, however, is interpretated similarily to that of RDA.

```
dune.env$n_moisture <- as.numeric(dune.env$Moisture)
dune.env$n_use <- as.numeric(dune.env$Use)
dune.env$n_manure <- as.numeric(dune.env$Manure)

dune.cca <- vegan::cca(dune, dune.env[,c(1,6:8)])

print(dune.cca)

## Call: cca(X = dune, Y = dune.env[, c(1, 6:8)])
##
##               Inertia Proportion Rank
## Total          2.1153     1.0000
## Constrained    0.8619     0.4075    4
## Unconstrained  1.2534     0.5925   15
## Inertia is mean squared contingency coefficient
##
## Eigenvalues for constrained axes:
##   CCA1   CCA2   CCA3   CCA4
## 0.4411 0.2363 0.1365 0.0480
##
## Eigenvalues for unconstrained axes:
##     CA1     CA2     CA3     CA4     CA5     CA6     CA7     CA8     CA9
```

```
## 0.30011 0.19798 0.17124 0.14013 0.09242 0.07666 0.06410 0.05818 0.04656
##    CA10    CA11    CA12    CA13    CA14    CA15
## 0.03899 0.02094 0.01708 0.01353 0.01086 0.00462

summary(eigenvals(dune.cca))

## Importance of components:
##                          CCA1   CCA2    CCA3    CCA4    CA1     CA2     CA3
## Eigenvalue             0.4411 0.2363 0.13650 0.04802 0.3001 0.19798 0.17124
## Proportion Explained  0.2085 0.1117 0.06453 0.02270 0.1419 0.09359 0.08095
## Cumulative Proportion 0.2085 0.3202 0.38475 0.40746 0.5493 0.64293 0.72388
##                           CA4     CA5     CA6     CA7     CA8     CA9
## Eigenvalue             0.14013 0.09242 0.07666 0.06410 0.05818 0.04656
## Proportion Explained  0.06625 0.04369 0.03624 0.03031 0.02750 0.02201
## Cumulative Proportion 0.79013 0.83382 0.87006 0.90036 0.92787 0.94988
##                          CA10    CA11    CA12    CA13    CA14     CA15
## Eigenvalue             0.03899 0.02094 0.01708 0.01353 0.01086 0.004618
## Proportion Explained  0.01843 0.00990 0.00807 0.00640 0.00514 0.002180
## Cumulative Proportion 0.96831 0.97821 0.98628 0.99268 0.99782 1.000000
```

```
plot(dune.cca)
```



```
sessionInfo()
```

```
## R version 3.3.2 (2016-10-31)
## Platform: x86_64-pc-linux-gnu (64-bit)
## Running under: Ubuntu 14.04.5 LTS
##
## locale:
##  [1] LC_CTYPE=de_DE.UTF-8       LC_NUMERIC=C
##  [3] LC_TIME=de_DE.UTF-8        LC_COLLATE=de_DE.UTF-8
##  [5] LC_MONETARY=de_DE.UTF-8    LC_MESSAGES=de_DE.UTF-8
##  [7] LC_PAPER=de_DE.UTF-8       LC_NAME=C
##  [9] LC_ADDRESS=C               LC_TELEPHONE=C
## [11] LC_MEASUREMENT=de_DE.UTF-8 LC_IDENTIFICATION=C
##
## attached base packages:
## [1] stats     graphics  grDevices utils     datasets  methods   base
##
## other attached packages:
## [1] gclus_1.3.1       clusterSim_0.45-1 MASS_7.3-45       cluster_2.0.5
## [5] vegan_2.4-2       lattice_0.20-34   permute_0.9-4     knitr_1.15.1
##
## loaded via a namespace (and not attached):
##  [1] Rcpp_0.12.9     magrittr_1.5    modeest_2.1     xtable_1.8-2
##  [5] R6_2.2.0        highr_0.6       stringr_1.1.0   tools_3.3.2
##  [9] parallel_3.3.2  grid_3.3.2      nlme_3.1-128    mgcv_1.8-16
## [13] e1071_1.6-7     htmltools_0.3.5 class_7.3-14    ade4_1.7-5
## [17] digest_0.6.11   rgl_0.97.0      Matrix_1.2-8    shiny_1.0.0
## [21] formatR_1.4     htmlwidgets_0.8 mime_0.5        evaluate_0.10
## [25] stringi_1.1.2   R2HTML_2.3.2    jsonlite_1.2    httpuv_1.3.3
```