Ordination and Clustering

Applied Statistics – A Practical Course

Thomas Petzoldt

2024-10-22

0.1 Introduction

The following demo demonstrates the reduction of a three dimensional data set into two dimensions by principal component analysis (PCA) and nonmetric multidimensional scaling.

The demo uses artificial data and a 3D plotting package, so the code may look somewhat technical. But don't worry. This is for demonstration of the main principles. Practical demonstrations with real data will follow.

0.2 Packages and data set

First we load the required packages and a test data set `multivar.csv`. It consists of 3 clusters of correlated multivariate normally distributed data points that were generated with the `rmvnorm` function. Each data subset has a separate mean value and a common variance-covariance matrix σ that is created at random. A separate script is used to generate the data.

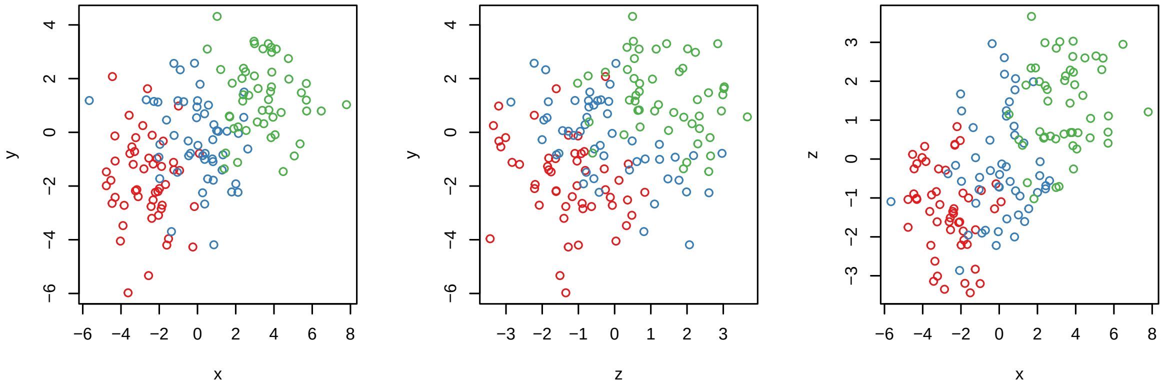
```r
library("rgl")
library("vegan")
library("vegan3d")

A <- read.csv("../data/multivar.csv")
```

0.3 Plotting

The data set has three columns, so we can, in principle, plot all column variables versus each other, or use a 3D plot.

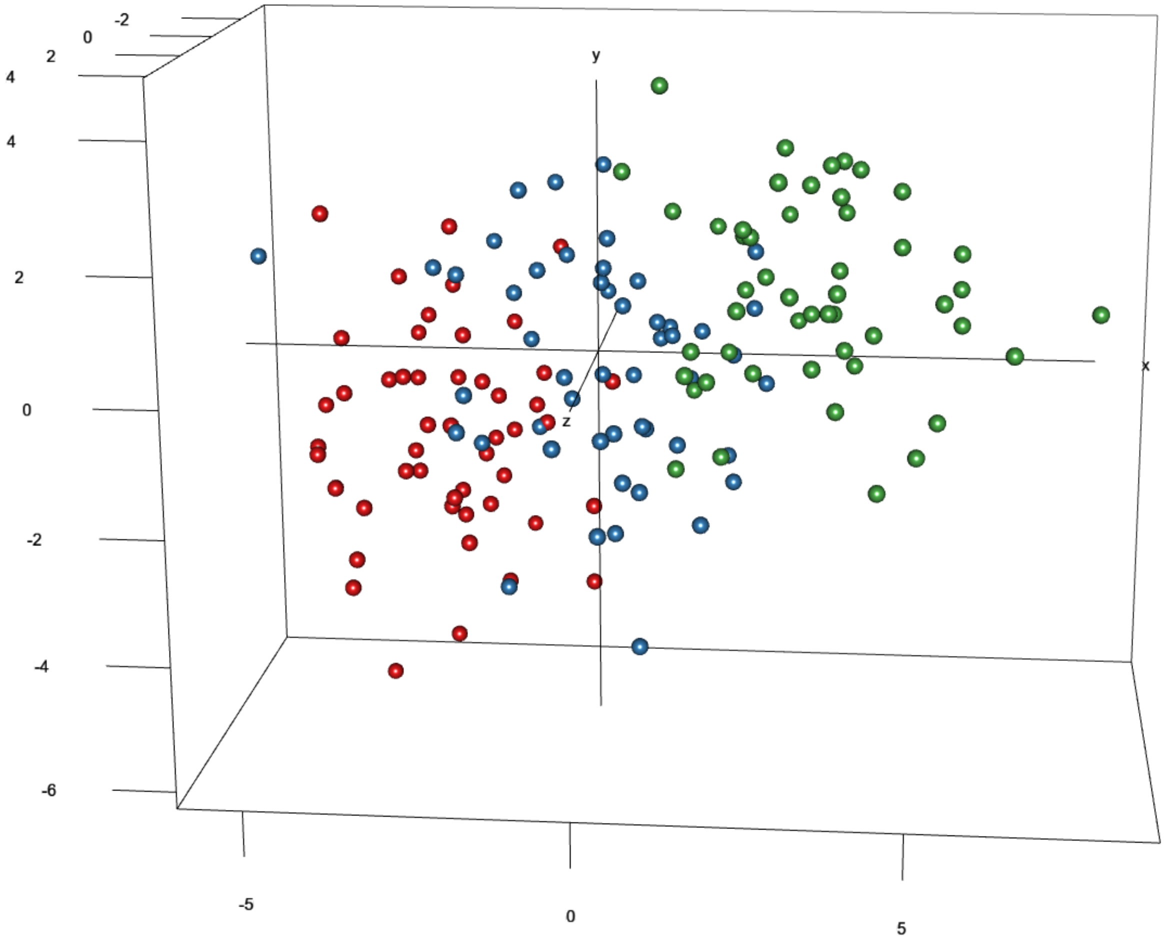
Note: The figure below can be rotated and zoomed with the mouse. Here we employ R's 3D graphics interface package **rgl** and the multivariate 3D visualisation package **vegan3d**.

```
nsamp    <- nrow(A) / 3  # number of points in each of the 3 samples
mycolors <- rep(c("#e41a1c", "#377eb8", "#4daf4a"), each = nsamp)

par(mfrow=c(1,3))
plot(y ~ x, data=A, col=mycolors)
plot(y ~ z, data=A, col=mycolors)
plot(z ~ x, data=A, col=mycolors)
```



We see that the colors overlap more or less in the 3 figures, so let's try to improve this in the 3D view. The goal is to rotate the axes such, so that the colored dots form well separated clusters.

```
ordirgl(A, type="p", ax.col = "black", col=mycolors, box=FALSE)
view3d(theta = 5, phi = 15, fov=30, zoom=0.7)
axes3d(labels=FALSE)
```



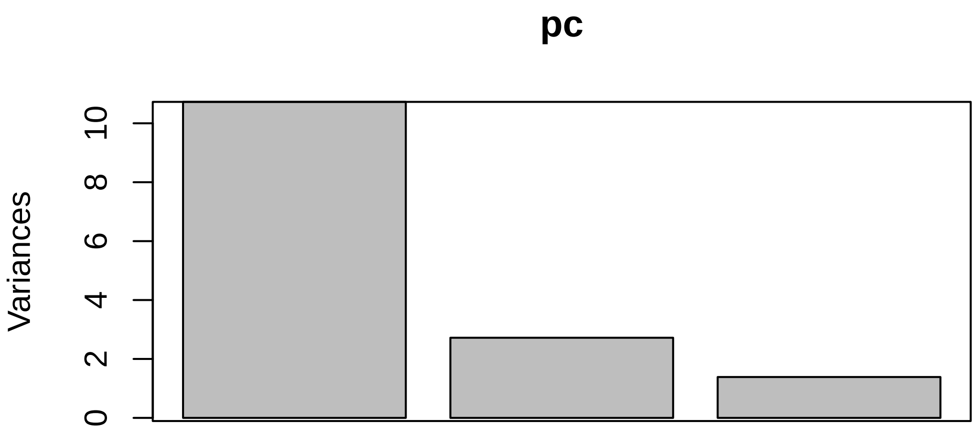
We see that the coordinate axes can be rotated in the direction of maximum variance of the data, so that overlap between data points is minimized. We can try to do this by hand, or let the computer do this. This is then calles a “principal components analysis” (PCA).

0.4 Principal components

R contains several functions for principal components analysis, for example `princomp` and `prcomp` in the **stats** package or function `rda` in the **vegan** package. The result is, that the coordinate system is rotated in the direction om maximum variance, using Eigen value calculations. The resulting “synthetic” new dimensions are then the principal components.

Each principal component represents then a specific fraction of the total variance of the data in ascending order. This can be plotted as bar chart or printed with `summary`.

```
pc <- prcomp(A)
plot(pc)
box()
```

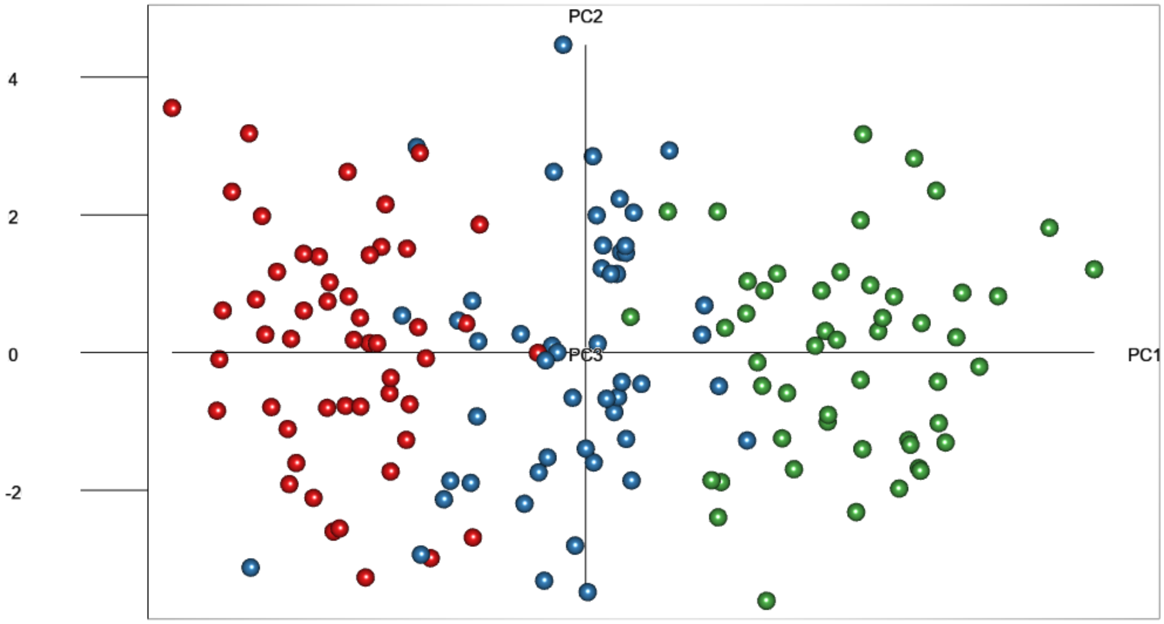


```
summary(pc)
```

```
Importance of components:
                          PC1    PC2    PC3
Standard deviation     3.2753 1.6495 1.1778
Proportion of Variance 0.7231 0.1834 0.0935
Cumulative Proportion  0.7231 0.9065 1.0000
```

The objects in the rotated coordinate axes can then be visualized with function `biplot` in 2 dimensions (shown later with real data) or with `ordirgl` in 3D. Function `view3d` rotates the plot in the direction of PC1 and PC2, but you can use the mouse to see that there is still a 3rd dimension.

```
pc <- prcomp(A)
ordirgl(pc, type="p", display="sites",
        ax.col = "black", col=mycolors)
view3d(theta = 0, phi = 0, fov=0, zoom=0.7)
axes3d()
```



0.5 Nonmetric multi-dimensional scaling (NMDS)

The PCA is a very useful technique as it reduces dimensions without bias, but as we have seen, large proportions of information may still be found at higher dimensions.

This is, where the nonmetric dimensional scaling comes into play. Here, we can request a number of dimensions k, typically $k = 2$ or $k = 3$ and then the computer tries hard to squeeze the information as much as possible into these dimensions. The aim is, that the distances between in all dimensions are represented in two or three dimensions only. Though it is done in an iterative way, it is clear, that this is not perfectly possible and will introduce considerable distortion, called stress.

```
mds <- metaMDS(A, distance="euclid", scale=TRUE, autotransform = FALSE, k=2)
```

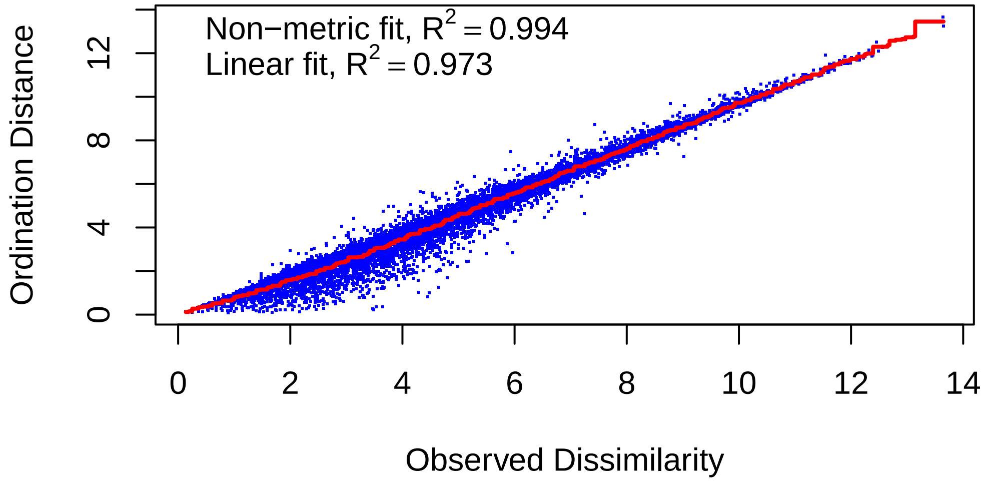
```
'comm' has negative data: 'autotransform', 'noshare' and 'wascores' set to FALSE
```

```
Run 0 stress 0.07782409
Run 1 stress 0.09722861
Run 2 stress 0.07783594
... Procrustes: rmse 0.0003824761  max resid 0.003870836
... Similar to previous best
Run 3 stress 0.09722861
Run 4 stress 0.07782409
... Procrustes: rmse 9.278635e-06  max resid 6.751774e-05
... Similar to previous best
Run 5 stress 0.07783595
```

```
... Procrustes: rmse 0.0003828048  max resid 0.003878591
... Similar to previous best
Run 6 stress 0.07783593
... Procrustes: rmse 0.0003820739  max resid 0.003871127
... Similar to previous best
Run 7 stress 0.07783593
... Procrustes: rmse 0.0003803489  max resid 0.003856084
... Similar to previous best
Run 8 stress 0.07782409
... Procrustes: rmse 2.935802e-06  max resid 1.952866e-05
... Similar to previous best
Run 9 stress 0.09722861
Run 10 stress 0.0972286
Run 11 stress 0.07782409
... New best solution
... Procrustes: rmse 6.256783e-06  max resid 6.575919e-05
... Similar to previous best
Run 12 stress 0.07783593
... Procrustes: rmse 0.0003797485  max resid 0.003853115
... Similar to previous best
Run 13 stress 0.07782409
... New best solution
... Procrustes: rmse 3.302781e-06  max resid 1.993058e-05
... Similar to previous best
Run 14 stress 0.0972286
Run 15 stress 0.07782409
... Procrustes: rmse 3.643497e-06  max resid 2.764178e-05
... Similar to previous best
Run 16 stress 0.07782409
... Procrustes: rmse 3.994882e-06  max resid 3.687607e-05
... Similar to previous best
Run 17 stress 0.07782409
... Procrustes: rmse 3.452381e-06  max resid 3.243292e-05
... Similar to previous best
Run 18 stress 0.09722861
Run 19 stress 0.07782409
... Procrustes: rmse 4.386977e-06  max resid 3.485243e-05
... Similar to previous best
Run 20 stress 0.09722253
*** Best solution repeated 5 times
```

The resulting stress and the representation of distances in two (or three) dimensions can then be shown in a so-called stressplot. Here, one should not overestimate the hich r^2 values, that are always big, even in bad cases. The shape of the step-line does not matter much either. However, the value of stress is most important and the data points should be close to the red line.

```
stressplot(mds)
```

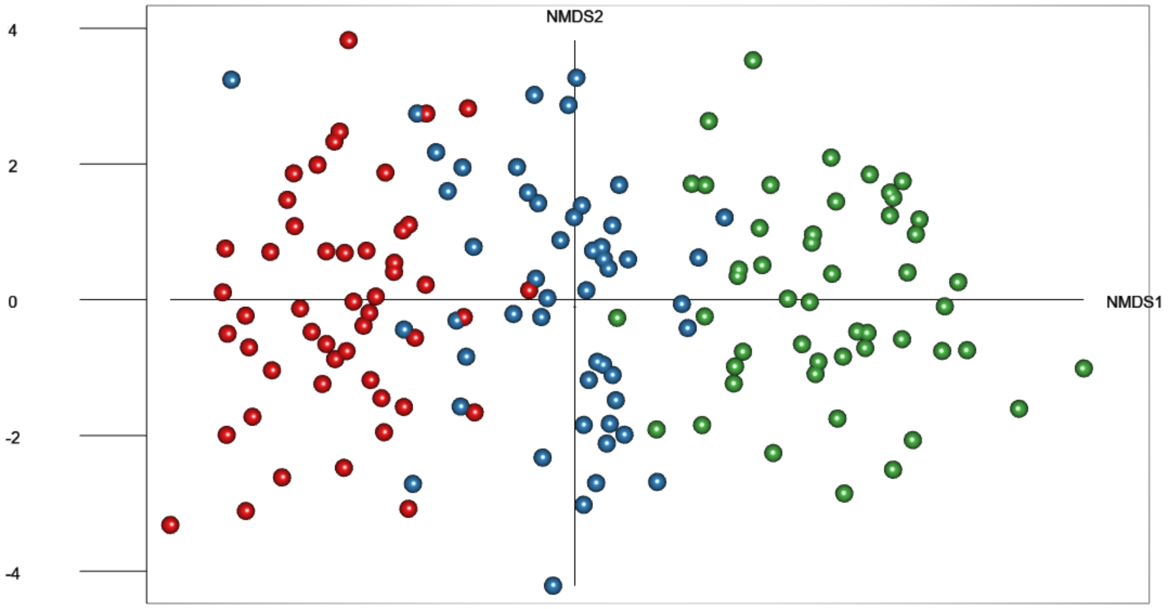


The stress itself shuld be small. As a rule of thumb, a stress value of > 0.2 means that the NMDS was not successful, stress < 0.1 is considered as sufficient, < 0.05 as good, < 0.025 as very good and ≈ 0 as perfect.

```
mds
```

```


Call:
metaMDS(comm = A, distance = "euclid", k = 2, autotransform = FALSE,      scale = TRUE)

global Multidimensional Scaling using monoMDS

Data:     A
Distance: euclidean

Dimensions: 2
Stress:     0.07782409
Stress type 1, weak ties
Best solution was repeated 5 times in 20 tries
The best solution was from try 13 (random start)
Scaling: centring, PC rotation
Species: scores missing
```

```
xyz <- as.data.frame(scores(mds, display="sites"))
xyz$. <- 0
ordirgl(xyz, type="p", ax.col = "black", col=mycolors)
view3d(theta = 0, phi = 0, fov=0, zoom=0.7)
axes3d(labels=FALSE, expand=1.5)
```



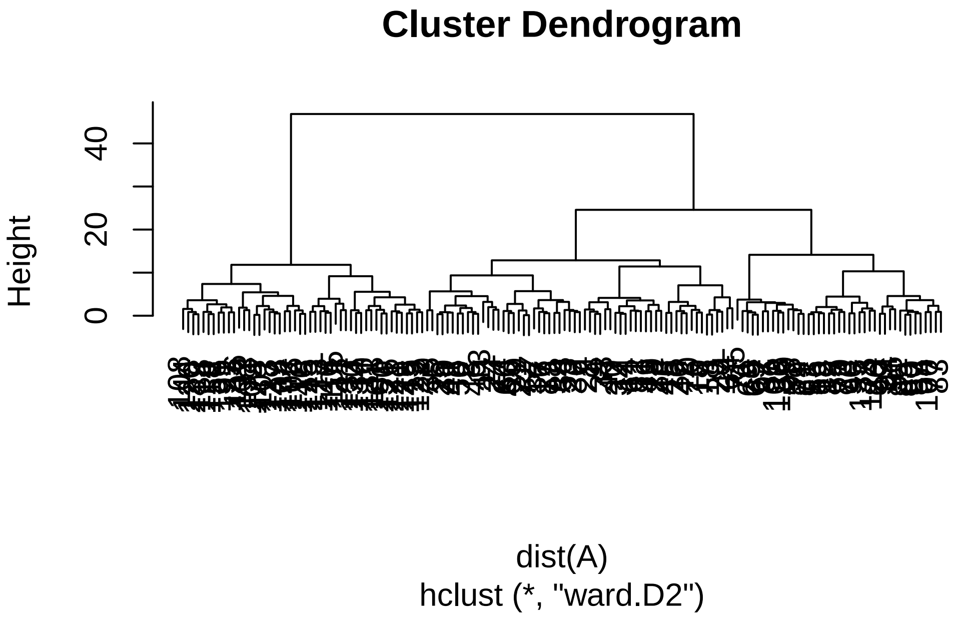
Here, stress is < 0.1. This is o.k., so that we now can have a look at the 2D representation. The figure is again technically 3D, so use the mouse to see that the data are now at a plane.

0.6 Cluster analysis

The methods discussd so far try to map high dimensional structures to lower dimensions as good as possible, but there is still variation left that is not shown, either because it is in a higher dimension as in PCA or because the mapping is "under stress" (NMDS). This means that points shown closely together may not as similar as they appear in the projection plane.

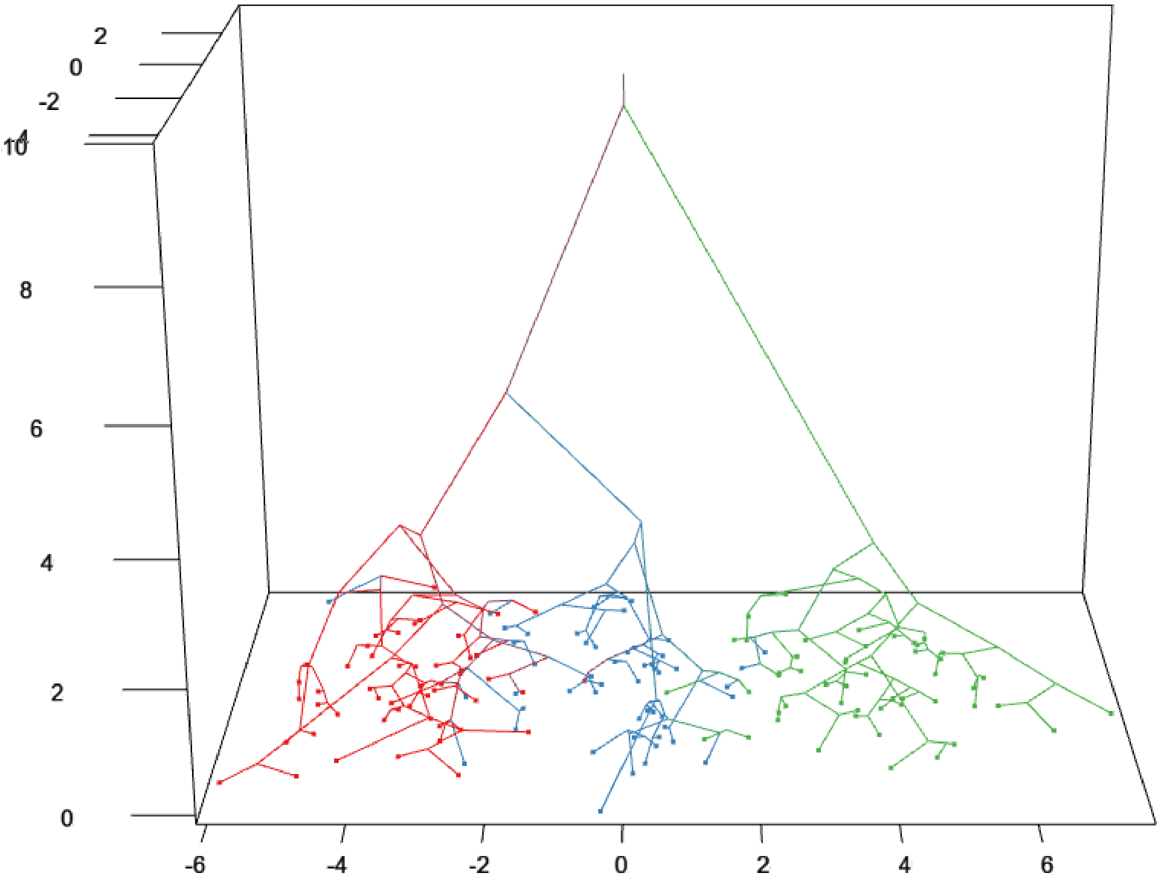
Cluster analysis takes another route. It shows the distance, not the location. Many different cluster analysis methods exist, here we show just an example of hierarchical clustering with "ward.D2" as agglomeration scheme. More about this will be discussed later.

```
hc <- hclust(dist(A), method="ward.D2")
plot(hc)
```



The result can also be combined with PCA or NMDS, here again a 3D visualization.

```
hc <- hclust(dist(A), method="ward.D2")
ordirgltree(ord=mds, cluster=hc, col=mycolors)
axes3d(expand=1.5)
```



The plot can again be rotated and zoomed in. The x- and y axes show the NMDS coordinates and the y axis the euclidean distance. Two-dimensional plots are of course also possible, practical examples will be given later.

1 Glossary

1.1 Distance and dissimilarity measures

1.1.1 Quantitative form

with x_{ij}, x_{ik} abundance of species i at sites (j, k).

Euclidean distance:

$$d_{jk} = \sqrt{\sum (x_{ij} - x_{ik})^2}$$

Manhattan distance:

$$d_{jk} = \sum |x_{ij} - x_{ik}|$$

Gower distance:

$$d_{jk} = \frac{1}{M} \sum \frac{|x_{ij} - x_{ik}|}{\max(x_i) - \min(x_i)}$$

Bray-Curtis dissimilarity:

$$d_{jk} = \frac{\sum |x_{ij} - x_{ik}|}{\sum (x_{ij} + x_{ik})}$$

1.1.2 Binary form

The binary form is applicable to binary and factor variables, where:

- $A, B =$ numbers of species on compared sites
- $J =$ (joint) is the number of species that occur on both compared sites
- $M =$ number of columns (excluding missing values)

Euclidean: $\sqrt{A + B - 2J}$

Manhattan: $A + B - 2J$

Gower: $\frac{A+B-2J}{M}$

Bray-Curtis: $\frac{A+B-2J}{A+B}$

Jaccard: $\frac{2b}{1+b}$ with $b =$ Bray-Curtis dissimilarity

1.1.3 Applications

Additional distance measures and application suggestions are found in the vegdist help page.