Training in ade4 in R - Module I: Basic methods

Practical 1: understanding multivariate methods in 3D

Stéphane Dray

2023-12-03

Introduction

In this practical we will try to understand the main principles of multivariate methods by interactive representation in 3 dimensions using the `rgl` package

First, load the required packages:

```
library(ade4)
library(rgl)
```

and then the data set `doubs` available in `ade4`

```
data(doubs)
```

Data

The data set contains information on 30 sites sampled on the Doubs river:

- Measurments of 11 environmental variables in `$env`
- Abundances of 27 fish species in `$fish`
- Spatial coordinates of the sites in `$xy`

More details? Try `?doubs`

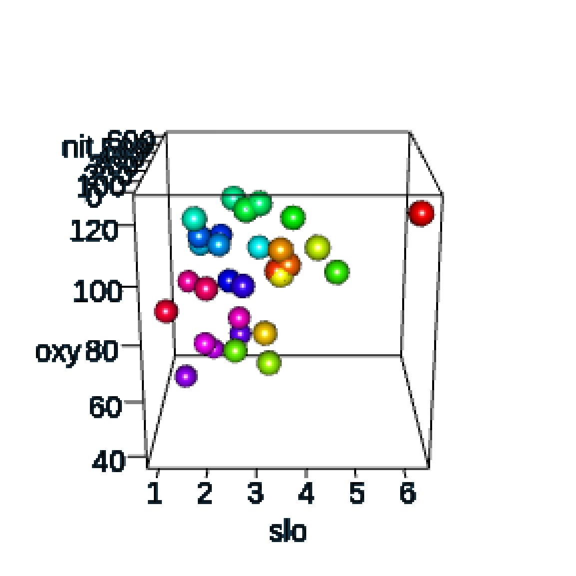
Select 3 environmental variables for this practical and define a vector to color the different sites:

```
tab <- doubs$env[, c(3, 8, 10)]
color <- rainbow(30, alpha = 0.5)
```

Representing data: space of individuals

Load some utilities functions available in the R directory and plot the original data:

```r
source("../../R/3D-utils.R")
plot3d(tab, type = "s", col = color)
```



- Our aim it to rotate the cloud to have the best viewpoint on the data.
- How can you charachterize this viewpoint?
- Look at scales!

This representation does not use the same scaling for variables due to the differences in ranges:

```
summary(tab)
```

The same representation with iso-scaling:

```
lims <- 1.1 * c(min(tab), max(tab))
plot3d(tab, type = "s", col = color, xlim = lims, ylim = lims,
    zlim = lims)
```

- What is the problem?
- How to solve it?

Data are in different units. Main variations are mainly driven by differences in variances. Standardization is required.

Center the data and see which geometric operation is induced:

```
## centring
tab.c <- scale(tab, center = TRUE, scale = FALSE)
myplot3d(tab.c, colpoints = color, colaxes = "black")
summary(tab.c)
```

Scale the data and see which geometric operation is induced:

```
## scaling
tab.sc <- scale(tab, center = TRUE, scale = TRUE)
myplot3d(tab.sc, colpoints = color, colaxes = "black")
```

Rotate and find the best viewpoint.

Principal component analysis

Draw the principal axes

```
pca <- dudi.pca(tab, nf = 3, scannf = FALSE)
addvar3d(t(pca$c1), col = "red")
```

and project the sites on the principal axes

```
## first PCA axis
for (i in 1:nrow(pca$tab)) segments3d(rbind(pca$c1[,
    1] * pca$li[i, 1], tab.sc[i, ]), col = color[i],
    lty = 3)

## second PCA axis
for (i in 1:nrow(pca$tab)) segments3d(rbind(pca$c1[,
    2] * pca$li[i, 2], tab.sc[i, ]), col = color[i])
```

Lastly, we can use the principal axes as a new system of coordinates and represent the data in this new system

```
open3d()
myplot3d(pca$li, colpoints = color, colaxes = "black")
addvar3d((pca$c1), col = "red")
```

Space of variables

The same approach can be used to search for the best representation of variables.

```
tab2 <- doubs$env[c(1, 11, 27), ]
tab2.sc <- t(scale(doubs$env[c(1, 11, 27), ]))
color.var <- rainbow(11)
myplot3d2(tab2.sc, colarrows = color.var, colaxes = "black")

## add PCA axes
pca2 <- dudi.pca(tab2, nf = 3, scannf = FALSE)
addvar3d(t(pca2$l1), col = "red")
```

Go back to course 1

PCA by hand

Use matrix algebra and the functions `cor`, `%*%` and `eigen` to recompute the outputs of PCA by hand

Bonus: Perform the singular value decomposition (function `svd`) of the standardized table (`scalewt(tab)`) and compare to the outputs of PCA

Go back to course 1