

# **‘From Statistics to Data Mining’ Computer Lab Session n° 6: Principal Component Analysis**


---

**Master 1 COSI / CPS<sup>2</sup>  
Saint-Étienne, France**

Fabrice Muhlenbach

Laboratoire Hubert Curien, UMR CNRS 5516  
Université Jean Monnet de Saint-Étienne  
18 rue du Professeur Benoît Lauras  
42000 SAINT-ÉTIENNE, FRANCE  
<https://perso.univ-st-etienne.fr/muhlfabr/>

## **Outcome**

The objective of this lab is to become familiar with  functions for working with PCA.

## **1 Introduction: « Mon Violon Tombe Mais Je Sauve Une Note »**

### **How can we represent high-dimensional data?**

Three important features of the planets in our solar system are:

- a) the distance to the Sun
- b) the equatorial diameter
- c) the density.

	Distance	Diameter	Density
Mercury	0.387	4878	5.42
Venus	0.723	12104	5.25
Earth	1.000	12756	5.52
Mars	1.524	6787	3.94
Jupiter	5.203	142800	1.31
Saturn	9.539	120660	0.69
Uranus	19.180	51118	1.29
Neptune	30.060	49528	1.64
Pluto	39.530	2300	2.03

We will load the planet data stored in an Excel file, and we will use the names in the first column for naming each row (after that, we will remove the first column).

```
library(XLConnect)
planets <- readWorksheet(loadWorkbook("planets.xlsx"), sheet=1)
```


Instead of using an Excel sheet, we can load the planet data stored in a CSV file and do the same thing:

```
library(readr)
planets <- read_csv("~/R/data/planets.csv",
                    header = TRUE, sep = ",",
                    quote = "\"'\" , _dec_=\".\")
```

We print the data, we use the names in the first column for naming each row and we will remove the first column:

```
planets
row.names(planets) <- planets[,1]
planets[,1] <- NULL
planets
```

## Pairwise Representation

The dataset has now 3 numerical variables. We can represent the data 2 by 2 by using the pairwise representation with the  function `pairs()`.

```
pairs(planets)
```

It is difficult to make a relevant conclusion just by seeing this plot. With the diameter and distance variables, some points are concentrated on an unique set (a cluster).

This is due to the astronomical scale. We can fix this problem by taking into account the log transform of each variable.

```
planets.log <- log(planets)
colnames(planets.log) <- paste("log(", colnames(planets), ")", sep="")
pairs(planets.log)
```

This new representation is better than the previous one, but we have many plots to study (if we have  $d$  variables, we can have  $d \times (d - 1)$  or at least  $\frac{d \times (d - 1)}{2}$  plots to see), and we don't have the information of the name of the planets.

## 2D Representation and Plot Size

In this context, we have one dimension representing the diameter of the planets (the second variable). We can use this information for changing the size of the plot and represent the data with the two other variables for the X- and Y-axes.

We will plot our graph in a new window, e.g. with the size  $800 \times 600$  pixels. The result can be seen on the Figure 1.

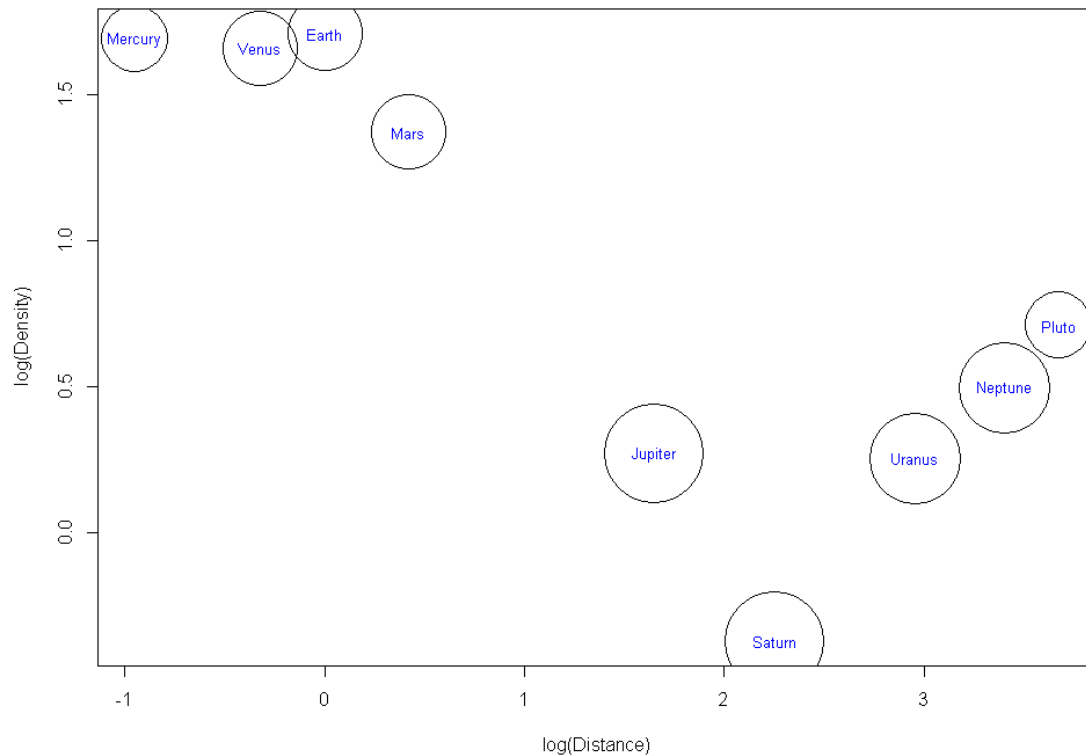


FIG. 1 – 2D Representation of the Planets.

```
# Create the plot without plotting the points
win.graph(800, 600, 10)

plot(x = planets.log[,1], y= planets.log[,3], cex=(round(planets.log[,2])),
      xlab = colnames(planets.log[1]), ylab = colnames(planets.log[3]))

# Write text to the position of each point.
text(x=planets.log[,1], y=planets.log[,3],
      labels=names, cex = 0.8, col = "blue")
```

This plot (Figure 1) is a little misleading: we can see two groups of circle: one on the top-left and the other on the bottom-right. The diameters of the planets, represented by the size of the circles, is not presented in the same way of the two others.

### 3D Representation

In **R**, we can easily represent the data in 3-dimensions with the **rgl** package. If this library is not installed on your computer, use the appropriate function (`install.packages("rgl")`).

```
library(rgl)
plot3d(x=planets$log[,1],
      y=planets$log[,2],
      z=planets$log[,3],
      xlab=colnames(planets$log[1]),
      ylab=colnames(planets$log[2]),
      zlab=colnames(planets$log[3]))
text3d(x=planets$log[,1],
      y=planets$log[,2],
      z=planets$log[,3],
      text=names,
      cex=0.8,
      col="blue")
```

The plot obtained is an interactive one: we can rotate the cube for having another view of the scatterplot.

On the Figure 2, we can see that they are two clusters (one with Mercury, Venus, Earth and Mars, and another cluster with Jupiter, Saturn, Uranus and Neptune) and an unique point.

In statistics, this kind of point, far from the other ones in the representation space, is denoted as “outlier”.

This outlier corresponds to Pluto. Note that in 2006 the International Astronomical Union (IAU) excluded Pluto to the list of the planets and reclassified it as a member of the “dwarf planet” category.

## Representation with Dimensionality Reduction

In conclusion, even if the 3D representation works well for this dataset, we must consider the case where there are more variables than only 3. In this context, we must proceed to a dimensionality reduction: finding a low-dimensional representation (a model) for high-dimensional data.

With the linear transformation, we can obtain a more compact representation: it is the so called “Principal Component Analysis” (PCA). This analysis is widely used for dimensionality reduction (projecting data points of a  $d$ -dimensional space onto a  $M$ -dimensional subspace) and for feature extraction.

In **R**, we can easily compute a PCA with the `princomp` function. It performs a principal components analysis on the given numeric data matrix and returns the results as an object of class `princomp`.

The calculation is done using `eigen` (for computing eigenvalues and eigenvectors of matrices) on the correlation or covariance matrix, as determined by `cor` (for computing the correlation).

The `print` method for these objects prints the results in a nice format and the `plot` method produces a *scree plot*. There is also a `biplot` method.

```
planets.pca <- princomp(planets$log)
planets.pca
win.graph(800, 600, 10)
biplot(planets.pca)
```

The Figure 3 is obtained by using the `princomp` function on the planets data and plotting the results obtained on the new axes. We can easily see the two clusters (corresponding to the terrestrial planets on the left and the gas giants on the right) and the outlier (Pluto, on the top).

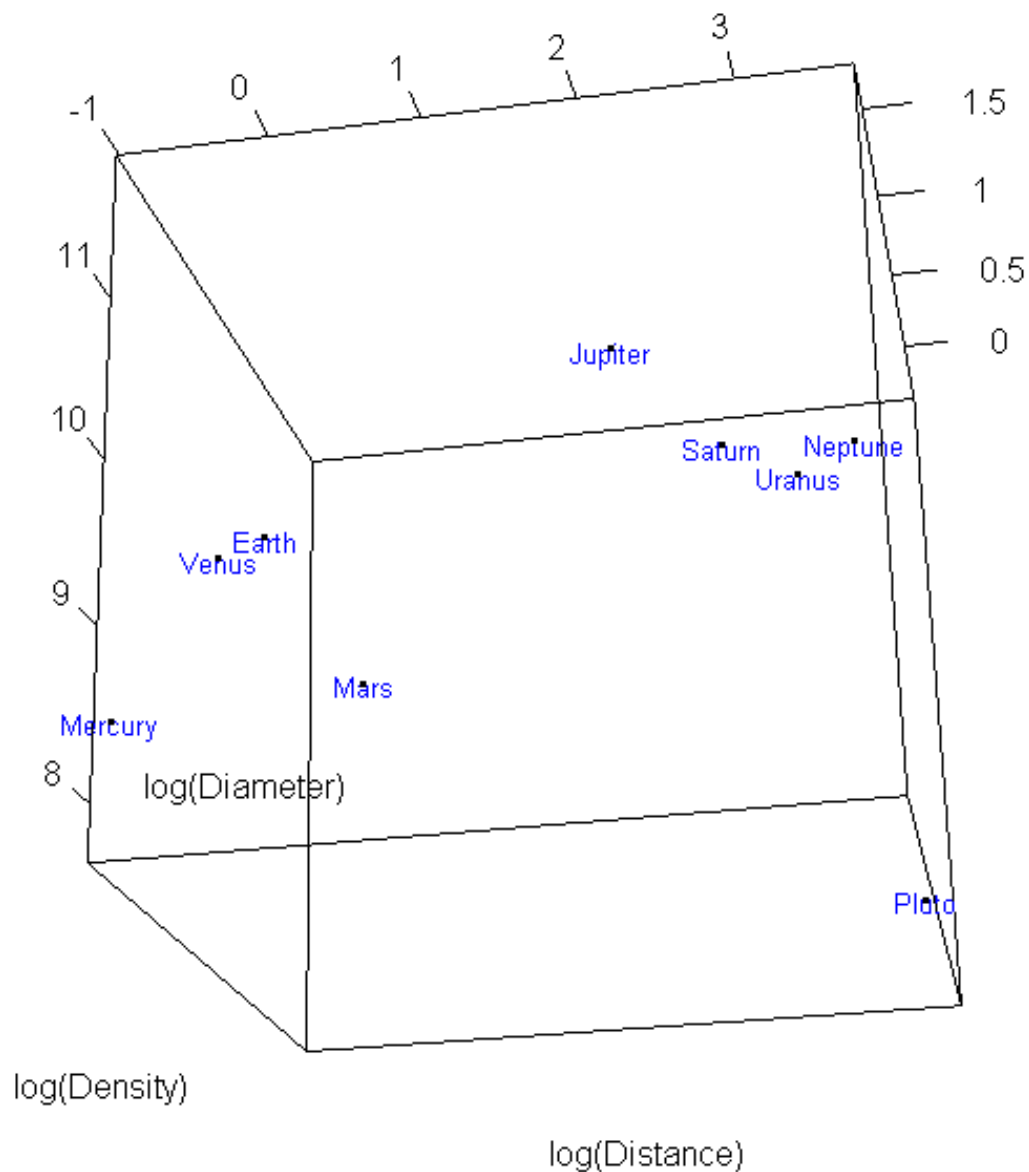


FIG. 2 – 3D Representation of the Planets.

### Why « Mon Violon Tombe Mais Je Sauve Une Note »?

« Mon Violon Tombe Mais Je Sauve Une Note » (“my violin is falling down but I save a musical note”) is a French planetary mnemonic (as the English phrase “My Very Easy Method Just Speeds Up Naming Planets”) used to remember the planets and dwarf planets of the Solar System, with the order of words corresponding to increasing sidereal periods of the bodies.

Instead of a phrase, it is possible to use a hand mnemonic as shown on the Figure 4, where the fingers of the left hand represent the terrestrial planets, the fingers of the right hand (palm upward) represent the gas

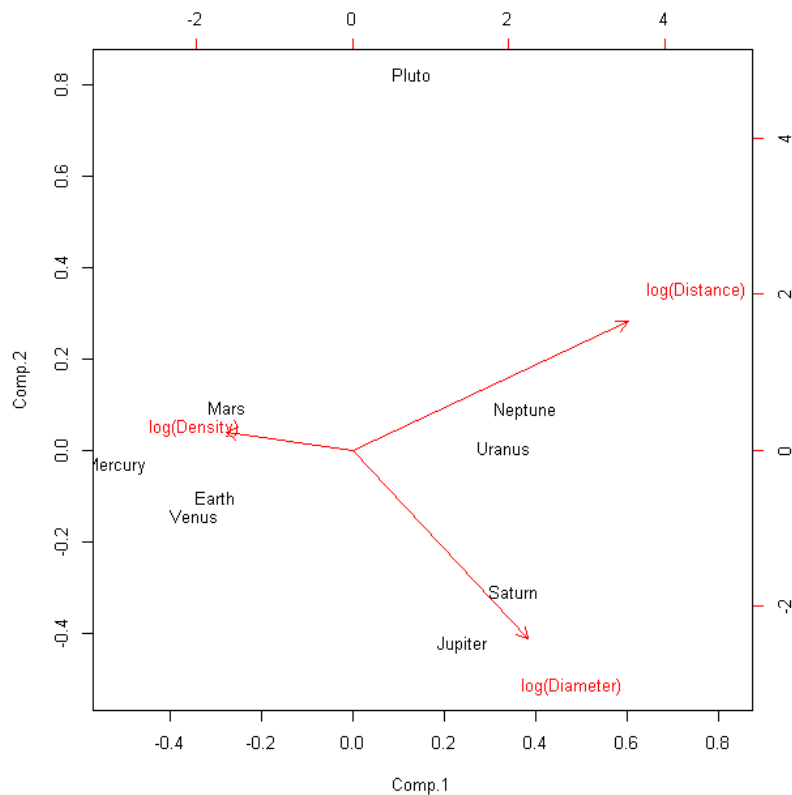


FIG. 3 – Representation of the Planets with the 2 first components.

giants, with Saturn as the ring finger.

## 2 PCA on a 2-Dimensional Dataset, Step by Step

Compute the PCA for  $X = \{(1, 2), (3, 3), (3, 5), (5, 4), (5, 6), (6, 5), (8, 7), (9, 8)\}$ .  
To achieve this task, you have to:

- Plot the data in the original 2D-space.

```
x <- c(1, 3, 3, 5, 5, 6, 8, 9)
y <- c(2, 3, 5, 4, 6, 5, 7, 8)
mydataset <- cbind(x, y)
mydataset
plot(mydataset[, "x"], mydataset[, "y"])
```

Note that for the plot function the option `asp` gives the aspect ratio  $y/x$ . If `asp` is a finite positive value then the window is set up so that one data unit in the  $x$  direction is equal in length to `asp`  $\times$  one data unit in the  $y$  direction. The special case `asp==1` produces plots where distances between points are represented accurately on screen.

```
plot(mydataset[, "x"], mydataset[, "y"], asp=1)
```

- Compute the covariance matrix  $\Sigma$  from the zero mean values  $(x - \bar{x})$ .

Reminder:  $cov(x, y) = \frac{\sum_{i=1}^n (x_i - \bar{x})(y_i - \bar{y})}{n-1}$  or  $cov(x, y) = \frac{\sum_{i=1}^n (x_i - \bar{x})(y_i - \bar{y})}{n}$  without correction.

```
n <- nrow(mydataset)
d <- ncol(mydataset)
xbar = mean(x)
ybar = mean(y)
```

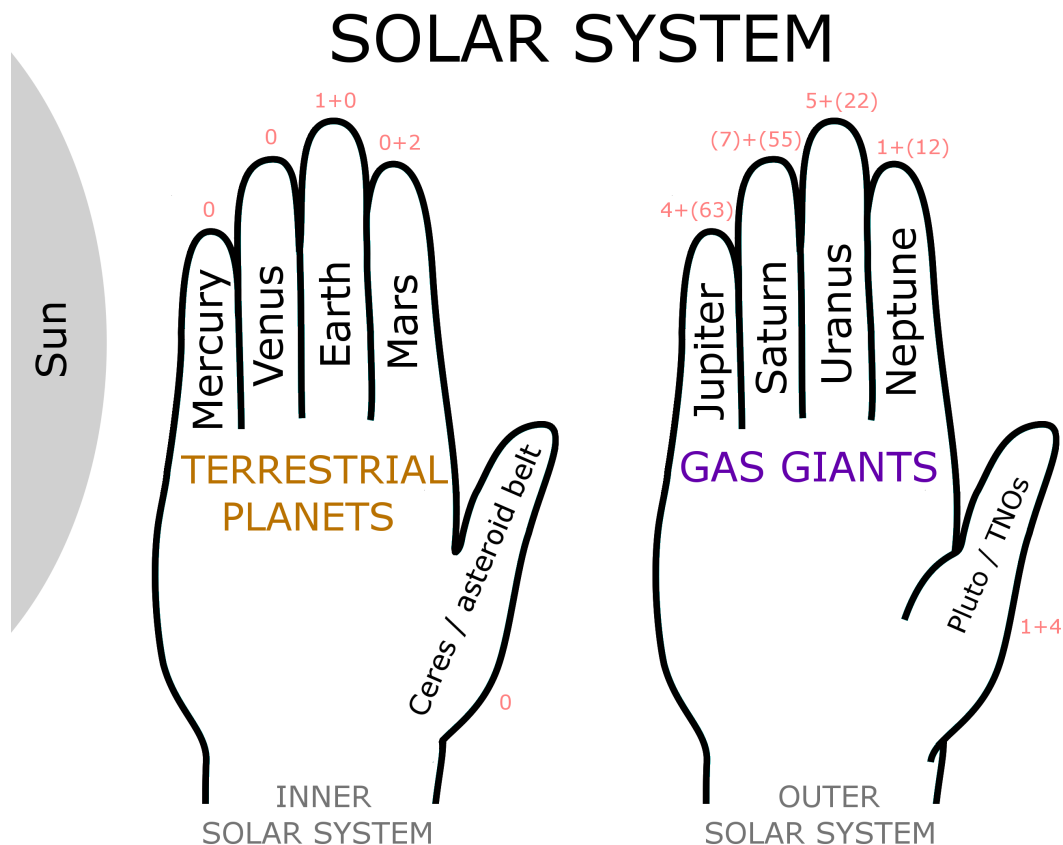


FIG. 4 – A visual mnemonic using the left hand to represent the terrestrial planets with the dwarf planet Ceres (/asteroid belt) and the right hand, palm turned upward, to represent the gas giants with the dwarf planet Pluto. [This file is licensed under the Creative Commons Attribution-Share Alike 3.0 Unported license.]


```

newX = x-xbar
newY = y-ybar

# Covariance without correction
covariance <- 0
for (i in 1:n)
{
  covariance <- covariance + newX[i] * newY[i]
}
covariance <- covariance / (n)
covariance

# Covariance with correction
covariance <- 0
for (i in 1:n)
{
  covariance <- covariance + newX[i] * newY[i]
}
covariance <- covariance / (n - 1)
covariance

```

Or simply by using the  function **COV**:


```

mynewdataset <- data.frame(newX, newY)
covariance <- cov(mynewdataset)
covariance

```

The **COV** function computes and prints

$$C = \begin{pmatrix} cov(x, x) & cov(x, y) \\ cov(y, x) & cov(y, y) \end{pmatrix}$$

- Solve the characteristic equation  $\det(\Sigma - \lambda I) = 0$  to get the eigenvalues and deduce the eigenvectors, then compute the 2 first components. In , we can simply use the **eigen** function.

```

E <- eigen(covariance)
E$vectors
E$values
Component1 <- (E$vectors[1]/E$vectors[2])*newX
Component1
Component2 <- (E$vectors[3]/E$vectors[4])*newY
Component2

```

- Plot the scatterplot of the dataset, then the 2 first components. They will intersect perpendicularly (on an orthogonal graph).



```

win.graph(800,600,10)
plot(mynewdataset$newX, mynewdataset$newY,
      col="black", xlim=c(-6,6),ylim=c(-6,6), asp=1)
par(new=TRUE)
plot(mynewdataset$newX, Component1, xlab="", ylab="",
      col="blue", type="l", xlim=c(-6,6),ylim=c(-6,6), asp=1)
par(new=TRUE)
plot(mynewdataset$newY, Component2, xlab="", ylab="",
      col="red", type="l", xlim=c(-6,6),ylim=c(-6,6), asp=1)

```

- Plot the biplot of the PCA obtained with the function `princomp`.

```

mynewdataset.pca <- princomp(mynewdataset)
win.graph(800,600,10)
biplot(mynewdataset.pca)


```

### 3 PCA on an High-Dimensional Dataset

#### 3.1 Dataset

In a previous lab session, we have collected some information about the students:

- Name
- Size (height in cm)
- LatBP\_NS (latitude decimal coordinate of the birth place = North-South axis)
- Long\_BP\_EW (longitude decimal coordinate of the birth place = East-West axis)
- NbDays (number of day from the birthday to the present day)
- Age (number of years)

Load the Excel file (or CSV file) with this dataset in , label each row with the column of the name, then remove this unnecessary column.

```

rm(list=ls()) # for clearing all the objects from the workspace
library(XLConnect)
students <- readWorksheet(loadWorkbook("student_data.xlsx"), sheet=1)
students
names <- students[,1]
rownames(students) <- students[,1]
students[,1] <- NULL
students


```

### 3.2 Descriptive Statistics

We can print some useful information about the dataset.

```
# Descriptive Statistics
# Summary
summary(students)
# Number of Observations
n <- nrow(students)
print(n)
# Object Properties
attributes(students)
```

### 3.3 Pairwise Representation, 2D-Study

By plotting the scatterplot matrices (with the  function `pairs()`), we can study the relation between the variables two by two. Are some variables more linked than others? Print the correlation coefficient for answering.

```
pairs(students)
cor(students)
```

The relation between the variables *Age* and *NbDays* is very strong (coefficient correlation = 0.99375884). These two variables are computed by taking into account the birthdate and the present day. The relation between them is given by  $Age = \lfloor (NbDays/365.25) \rfloor$ . We can plot it in a new window.

```
win.graph(800,600,10)
plot(students$NbDays, students$Age,
      xlim=c(7700,12000), ylim=c(20,32))
par(new=TRUE)
x <- seq(7700, 12000, length=2000)
y <- floor(x/365.25)
plot(x, y, col="red", type="l", xlab="", ylab="",
      xlim=c(7700,12000), ylim=c(20,32))
```

### 3.4 PCA with princomp Function

We will compute the PCA with the parameters `cor = TRUE` for indicating that the calculation should use the correlation and `scores = TRUE` for indicating that the score on each principal component need to be calculated.

```
pca.students <- princomp(students, cor = TRUE, scores = TRUE)
pca.students
summary(pca.students)
attributes(pca.students)
```

When printing the summary we can see that the information provided by the fifth and last component is very poor. How can you explain that?

### 3.5 Eigenvalues

We can now calculate and print the variances associated with the axes (i.e., the eigenvalues) and plotting the scree plot.

```
eigenvalues <- pca.students$sdev^2
eigenvalues
plot(1:5, eigenvalues, type="b", ylab="Eigenvalues",
     xlab="Components", main="Scree_Plot")
```

We will print the table of the eigenvalues for each component with a confidence interval  $\alpha = 95\%$ .

```
# Confidence interval of the eigenvalues at 95%
# Hint:
# pnorm(1.96, mean=0, sd=1) - pnorm(-1.96, mean=0, sd=1) = 0.95
val.low <- eigenvalues * exp(-1.96 * sqrt(2.0/(n-1)))
val.high <- eigenvalues * exp(+1.96 * sqrt(2.0/(n-1)))
# Table
table <- cbind(val.low, eigenvalues, val.high)
colnames(table) <- c("Lower_Bound", "Eigenvalues", "Higher_Bound")
print(table, digits=3)
```

### 3.6 Graphs

#### 3.6.1 Coordinates of the Variables on the Factorial Axes

First, we have to compute some interesting values (correlations between variables and the components, square value of the correlations and cumulative square of the correlations).

```
# Variables-Axes Correlation
c1 <- pca.students$loadings[,1]*pca.students$sdev[1]
c2 <- pca.students$loadings[,2]*pca.students$sdev[2]

# Correlation
correlation <- cbind(c1,c2)
print(correlation, digits=2)

# Square value of the correlation
print(correlation^2, digits=2)

# Cumulative square of the correlation
print(t(apply(correlation^2, 1, cumsum)), digits=2)
```

### 3.6.2 Correlation Circle Plot

We can now plot the correlation circle.

```
win.graph(800, 600, 10)
plot(c1, c2, xlim=c(-1,+1), ylim=c(-1,+1), type="n")
abline(h=0, v=0)
text(c1, c2, labels=colnames(students), cex=1, col="red")
symbols(0,0, circles=1, inches=FALSE, add=TRUE)
```

### 3.6.3 Representation of the Students in the First Factorial Plane

We can use the two first components for plotting the students.

```
win.graph(800, 600, 10)
plot(pca.students$scores[,1],
     pca.students$scores[,2],
     type="n", xlab="Comp.1", ylab="Comp.2")
abline(h=0, v=0)
text(pca.students$scores[,1],
     pca.students$scores[,2],
     labels=rownames(students), cex=0.75)
```

For a simultaneous representation of the variables and observations, we can use the `biplot` function:

```
biplot(pca.students, cex=0.75)
```

### 3.6.4 Representation of the Students in the First Factorial Cube

With the 2 first components, we represent only 70% of the inertia of the students dataset. With the 3 first components, what is the cumulative proportion represented? (Use `summary(pca.students)`.)

```
library(rgl)
plot3d(x=pca.students$scores[,1],
      y=pca.students$scores[,2],
      z=pca.students$scores[,3],
      xlab="Comp.1",
      ylab="Comp.2",
      zlab="Comp.3")
text3d(x=pca.students$scores[,1],
      y=pca.students$scores[,2],
      z=pca.students$scores[,3],
      xlab="",
      ylab="",
      zlab="",
      text=names,
```

```
cex=0.8 ,
col="blue" )
```

In conclusion, is it relevant to represent the data in a 3D-space?

## 4 Recommended Readings

The main literature for this section is:

- Adler (2010), “R in a Nutshell – a Desktop Quick Reference,” Chapter 16 “Analyzing Data.”
- Everitt and Hothorn (2010), “A Handbook of Statistical Analyses Using R,” Chapter 16 “Principal Component Analysis.”
- Everitt and Hothorn (2011), “An Introduction to Applied Multivariate Analysis with R,” Chapter 3 “Principal Components Analysis.”
- Husson et al. (2010), “Exploratory Multivariate Analysis by Example Using R,” Chapter 1 “Principal Component Analysis (PCA).”
- Kabacoff (2011), “R in Action,” Chapter 14 “Principal components and factor analysis.”
- Tufféry (2010) « Data mining et statistique décisionnelle: L'intelligence des données », Chapitre « L'analyse factorielle » (in French).
- Wehrens (2011), “Chemometrics with R: Multivariate Data Analysis in the Natural Sciences,” Chapter 4 “Principal Component Analysis.”

## References

- Adler, J. (2010). *R in a Nutshell – a Desktop Quick Reference*. O'Reilly.
- Everitt, B. and T. Hothorn (2011). *An Introduction to Applied Multivariate Analysis with R*. Use R! Springer.
- Everitt, B. S. and T. Hothorn (2010). *A Handbook of Statistical Analyses Using R* (2<sup>nd</sup> ed.). Chapman & Hall / CRC.
- Husson, F., S. Lê, and J. Pagès (2010). *Exploratory Multivariate Analysis by Example Using R*. Computer Science & Data Analysis. Chapman & Hall / CRC.
- Kabacoff, R. (2011). *R in Action*. Manning Publications.
- Tufféry, S. (2010). *Data mining et statistique décisionnelle: L'intelligence des données* (3<sup>e</sup> ed.). Paris: Editions Technip.
- Wehrens, R. (2011). *Chemometrics with R: Multivariate Data Analysis in the Natural Sciences*. Use R! Springer.