Master Sciences Pour l'Environnement

Parcours Gestion de l'Environnement et Écologie Littorale

Analyse de données / Data analysis
Partie 5 / Part 5

B. Simon-Bouhet

La Rochelle Université

First semester

What will we talk about?...

13. Principal Component Analysis

Multivariate exploratory analysis

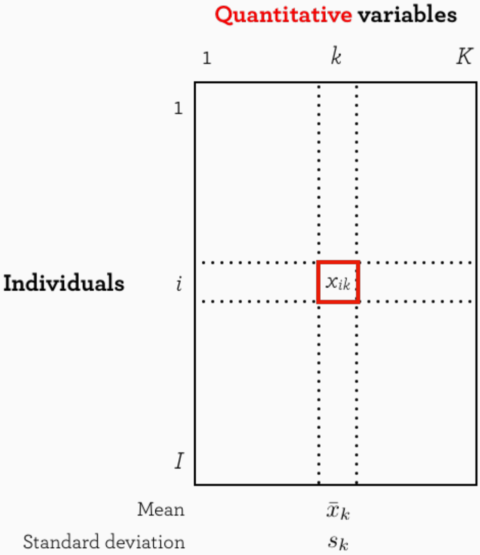
A lot of methods

English	French
Correspondance analysis	Analyse factorielle des correspondances
Reciprocal averaging	Analyse factorielle des correspondances
Multidimensional scaling	Positionnement multidimensionnel
Discriminant analysis	Analyse discriminante
Redundancy analysis	Analyse de redondance
Canonical correspondance analysis	Analyse canonique des correspondances

The mother of all methods: Principal Component Analysis.

Outline

Data, notations



Examples of data

Individuals	Variables	x_{ik}
Animal	Biometric measures (size, mass, femur length…)	Measure k for animal i
Station	Chemical measurements (nitrates, nitrites, phosphates)	Measure k for station i
Country	Economic indicators (GDP, unemployment rate, …)	Value of indicator k for country i
Student	Exam topics (physiology, statistics, …)	Grade obtained by student i in topic k
People surveyed	Quantitative questions (age, salary, number of kids, …)	Answer of individual i to question k

Outline

The rows of the PCA table

A PCA table can be examined either :

- by rows
- by columns

When examining the rows of the PCA table, we focus on:

- the similarities between individuals
- the response profiles
- the variability of profiles between individuals

We want to get a summary of all similarities/dissimilarities between individuals

The columns of the PCA table

A PCA table can be examined either :

- by rows
- by columns

When examining by columns of the PCA table, we focus on:

- the links between variables
- the relationship between variables
- the correlation coefficient

We want to get a summary of all relationships between variables

Reminder...

The correlation coefficient

$$\text{Sample mean: } \bar{x} = \frac{\sum_{i=1}^{n} x_i}{n}$$

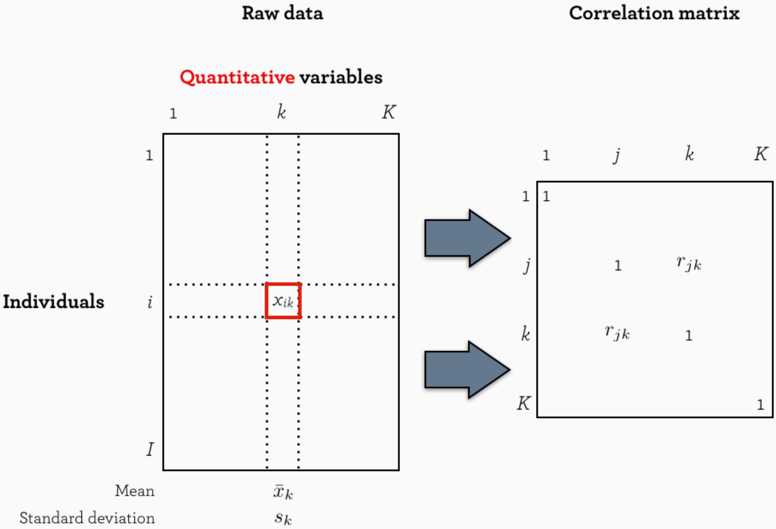
$$\text{Sample variance: } s_x^2 = \frac{\sum_{i=1}^{n} (x_i - \bar{x})^2}{n-1}$$

$$\text{Covariance: } cov(x,y) = \frac{\sum_{i=1}^{n} (x_i - \bar{x})(y_i - \bar{y})}{n-1}$$

$$\text{Correlation: } cor(x,y) = r_{xy} = \frac{cov(x,y)}{s_x s_y}$$

Reminder...

The correlation matrix



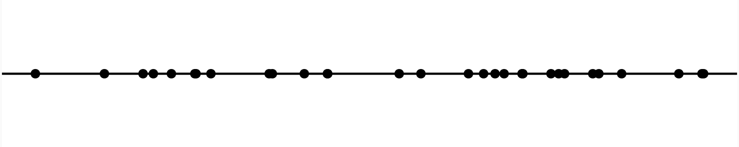
Outline

Visualising information about individuals

One variable

Imagine we only had one variable.

What would be the simplest method to visualize the similarities/dissimilarities between individuals?

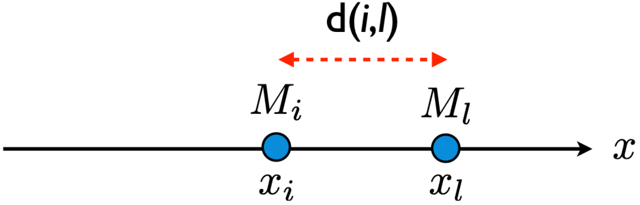


What would be the simplest metric one could use to measure the similarity between individuals?

Visualising information about individuals

One variable

When we have only 1 dimension, the Euclidian distance between two points is computed as :



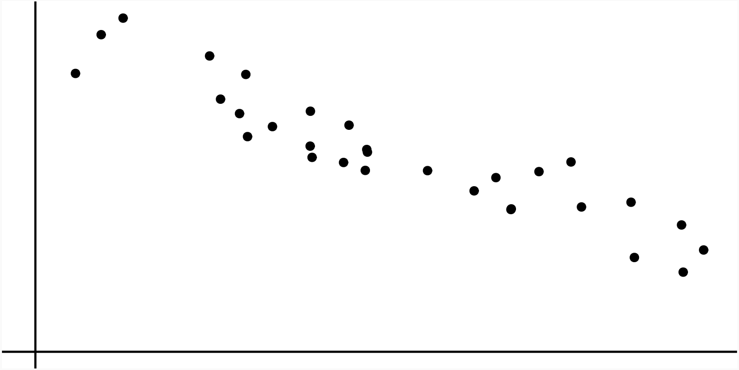
$$d^2(i, l) = (x_i - x_l)^2$$

Visualising information about individuals

Two variables

Imagine we only had two variables now.

What would be the simplest method to visualize the similarities/dissimilarities between individuals?

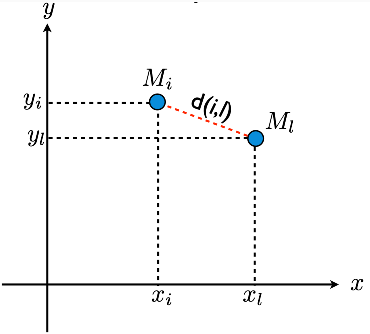


What would be the simplest metric one could use to measure the similarity between individuals?

Visualising information about individuals

Two variables

When we have 2 dimensions, the Euclidian distance between two points is computed as :



$$d^2(i,l) = (x_i - x_l)^2 + (y_i - y_l)^2$$

That's the Pythagorean theorem!

Visualising information about individuals

K variables

What's true for one and two variables is also true for K variables

Indeed, the Pythagorean theorem can be generalized to compute the Euclidian distance between points in K dimensions:

$$d^2(i,l) = \sum_{k=1}^{K} (x_{ik} - x_{lk})^2$$

Important

In a K-dimensional cloud of points, the smaller the Euclidian distance between two points, the greater their similarity.

But how do you visualise K dimensions ? We need projections…

Visualising information about individuals

K variables

So we want to know what the shape of our cloud of data looks like in order to:

- identify individuals that are similar or dissimilar to one another
- visualize portions of the cloud of data where patterns appear (e.g. lots of points clustered together, spread…)

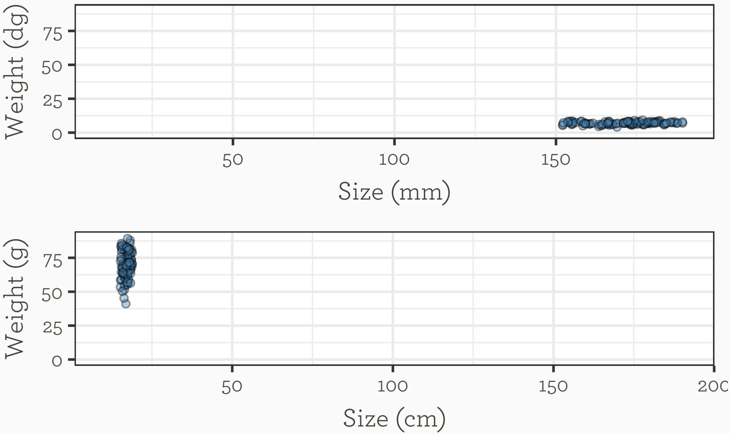
Problem: with more than 3 dimensions, we can't produce any satisfactory visualization

The PCA provides an numerical solution to visualise the best approximation possible of our data on a reduced number of axes.

Visualising information about individuals

On the importance of units...

If we want to study the shape of a cloud of points, we have to get rid of units.



Visualising information about individuals

On the importance of units...

Hence, we need to standardize all variables :

$$x'_{ik} = \frac{x_{ik} - \bar{x}_k}{s_k}$$

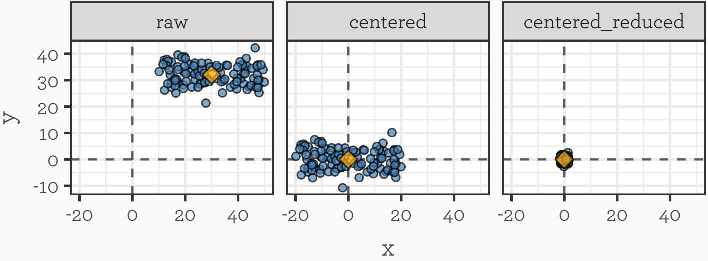
After this step, all variables have:

- a mean of zero
- a standard deviation (and a variance) of one

Thus, all variables will have the same weight in the PCA and the shape of the data cloud will only reflect the internal structure of the data and not the units.

Visualising information about individuals

On the importance of units…



Outline

The recipe for succes

1. Look at the raw dataset
2. Examine the correlation matrix
3. Preform the PCA
4. Look at the eigen values to select only the most usefull factorial axes
5. Compute the threshold value $\left(\frac{100}{K}\right)$
6. Identify all variables that should be interpreted on each of the factorial axes you selected on step 4
7. Examine the correlation circle(s) to orient the factorial axes
8. Try to find the meaning of each factorial axis
9. Visualize the data cloud projected on the factorial axes you selected earlier
10. Interpret

The recipe for succes

1. Raw dataset

```r
# Usefull packages
library(tidyverse)
library(ggrepel)
library(ade4)

# Load data from ade4 package
data(olympic)

# Transform and re-organize data
decath <- as_tibble(olympic$tab) %>%
  mutate(athlete = factor(1:33)) %>%
  relocate(athlete)    # Move "athlete" in front of other columns
```

The recipe for succes

1. Raw dataset

```
decath

# A tibble: 33 x 11
   athlete `100`  long  poid  haut `400` `110`  disq  perc  jave `1500`
   <fct>   <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl>  <dbl>
 1 1        11.2  7.43  15.5  2.27  48.9  15.1  49.3   4.7  61.3   269.
 2 2        10.9  7.45  15.0  1.97  47.7  14.5  44.4   5.1  61.8   273.
 3 3        11.2  7.44  14.2  1.97  48.3  14.8  43.7   5.2  64.2   263.
 4 4        10.6  7.38  15.0  2.03  49.1  14.7  44.8   4.9  64.0   285.
 5 5        11.0  7.43  12.9  1.97  47.4  14.4  41.2   5.2  57.5   257.
 6 6        10.8  7.72  13.6  2.12  48.3  14.2  43.1   4.9  52.2   274.
 7 7        11.2  7.05  14.1  2.06  49.3  14.4  41.7   5.7  61.6   291.
 8 8        11.0  6.95  15.3  2     48.2  14.4  41.3   4.8  63     266.
 9 9        11.2  7.12  14.5  2.03  49.2  14.7  42.4   4.9  66.5   270.
10 10       11.2  7.28  15.2  1.97  48.6  14.8  48.0   5.2  59.5   292.
# ... with 23 more rows
```

The recipe for succes

2. Correlation matrix

```
decath %>%
  select(-athlete) %>%
  cor() %>%
  round(3)

        100   long   poid   haut    400    110   disq   perc   jave   1500
100   1.000 -0.540 -0.208 -0.146  0.606  0.638 -0.047 -0.389 -0.065  0.261
long -0.540  1.000  0.142  0.273 -0.515 -0.478  0.042  0.350  0.182 -0.396
poid -0.208  0.142  1.000  0.122  0.095 -0.296  0.806  0.480  0.598  0.269
haut -0.146  0.273  0.122  1.000 -0.088 -0.307  0.147  0.213  0.116 -0.114
400   0.606 -0.515  0.095 -0.088  1.000  0.546  0.142 -0.319  0.120  0.587
110   0.638 -0.478 -0.296 -0.307  0.546  1.000 -0.110 -0.522 -0.063  0.143
disq -0.047  0.042  0.806  0.147  0.142 -0.110  1.000  0.344  0.443  0.402
perc -0.389  0.350  0.480  0.213 -0.319 -0.522  0.344  1.000  0.274 -0.031
jave -0.065  0.182  0.598  0.116  0.120 -0.063  0.443  0.274  1.000  0.096
1500  0.261 -0.396  0.269 -0.114  0.587  0.143  0.402 -0.031  0.096  1.000
```

The recipe for succes

3. Perform the PCA

```
# Load the ade4 package
library(ade4)

# Determine the number of variables in the dataset
n_var <- ncol(decath) - 1

# Perform the PCA
results <- decath %>%
  select(-athlete) %>%
  dudi.pca(scannf = FALSE, nf = n_var)

# Compute a few tables that will be useful for steps 4 and 6
contrib <- inertia.dudi(results, col.inertia = TRUE)
inertia <- contrib$tot.inertia
cont_abs <- contrib$col.abs
cont_rel <- contrib$col.rel
```

The recipe for succes

3. Perform the PCA

```
results

Duality diagramm
class: pca dudi
$call: dudi.pca(df = ., scannf = FALSE, nf = n_var)

$nf: 10 axis-components saved
$rank: 10
eigen values: 3.418 2.606 0.9433 0.878 0.5566 ...
  vector length mode    content
1 $cw    10     numeric column weights
2 $lw    33     numeric row weights
3 $eig   10     numeric eigen values

  data.frame nrow ncol content
1 $tab       33   10   modified array
2 $li        33   10   row coordinates
3 $l1        33   10   row normed scores
4 $co        10   10   column coordinates
5 $c1        10   10   column normed scores
other elements: cent norm
```

The recipe for succes

4. Eigen values and the selection of factorial axes

```
inertia

       inertia       cum    cum(%)
Ax1  3.4182381  3.418238  34.18238
Ax2  2.6063931  6.024631  60.24631
Ax3  0.9432964  6.967928  69.67928
Ax4  0.8780212  7.845949  78.45949
Ax5  0.5566267  8.402576  84.02576
Ax6  0.4912275  8.893803  88.93803
Ax7  0.4305952  9.324398  93.24398
Ax8  0.3067981  9.631196  96.31196
Ax9  0.2669494  9.898146  98.98146
Ax10 0.1018542 10.000000 100.00000
```

The recipe for succes

5/6. Threshold and variable selection

```r
# threshold
100 / n_var
```

```
[1] 10
```

```r
# Absolute contributions of original variables to each factorial axis
round(cont_abs, 2)
```

```
     Axis1 Axis2 Axis3 Axis4 Axis5 Axis6 Axis7 Axis8 Axis9 Axis10
100  17.30  2.21  7.15  0.78 19.56  0.09  6.47 44.05  1.17   1.20
long 15.53  2.31  2.85  5.97 13.61  0.88 56.33  2.00  0.21   0.31
poid  7.24 23.38  0.97  1.16  0.01  5.29  1.22  0.53 17.85  42.35
haut  4.51  0.08 73.10 15.05  0.00  0.56  1.83  2.42  1.04   1.43
400  12.66 12.40  3.59  0.65  2.16 10.69  2.00  2.16 42.35  11.34
110  18.79  0.48  1.59 14.61  0.79  4.43  7.43 40.83  4.29   6.75
disq  3.09 25.33  0.21  0.07  0.04 37.81  2.07  0.01  2.80  28.57
perc 14.75  2.24  1.87  2.07 51.37 12.09  7.47  7.67  0.03   0.43
jave  3.24 13.84  3.70 36.06  0.91 19.14 11.69  0.34  9.38   1.71
1500  2.89 17.72  4.95 23.58 11.54  9.02  3.49  0.01 20.87   5.91
```

The recipe for succes

5/6. Threshold and variable selection

```
# threshold
100 / n_var
```

```
[1] 10
```

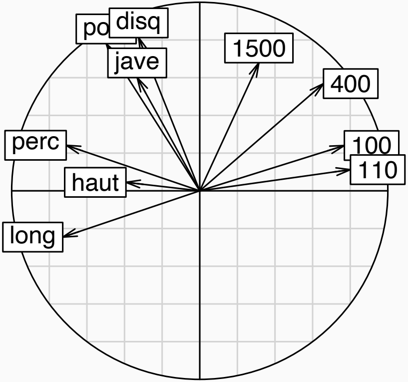
```
# Relative contributions of original variables
round(abs(cont_rel), 2)
```

```
     Axis1 Axis2 Axis3 Axis4 Axis5 Axis6 Axis7 Axis8 Axis9 Axis10
100  59.12  5.77  6.75  0.69 10.89  0.05  2.79 13.51  0.31   0.12
long 53.08  6.03  2.69  5.24  7.58  0.43 24.26  0.61  0.06   0.03
poid 24.75 60.94  0.92  1.02  0.01  2.60  0.53  0.16  4.76   4.31
haut 15.40  0.20 68.96 13.21  0.00  0.27  0.79  0.74  0.28   0.15
400  43.28 32.32  3.39  0.57  1.20  5.25  0.86  0.66 11.31   1.16
110  64.23  1.26  1.50 12.83  0.44  2.18  3.20 12.53  1.15   0.69
disq 10.56 66.03  0.20  0.06  0.02 18.57  0.89  0.00  0.75   2.91
perc 50.43  5.83  1.77  1.82 28.60  5.94  3.22  2.35  0.01   0.04
jave 11.07 36.06  3.49 31.66  0.51  9.40  5.03  0.11  2.50   0.17
1500  9.90 46.19  4.67 20.71  6.43  4.43  1.50  0.00  5.57   0.60
```

The recipe for succes

7/8. Correlation circle, orientation and meaning of the factorial axes

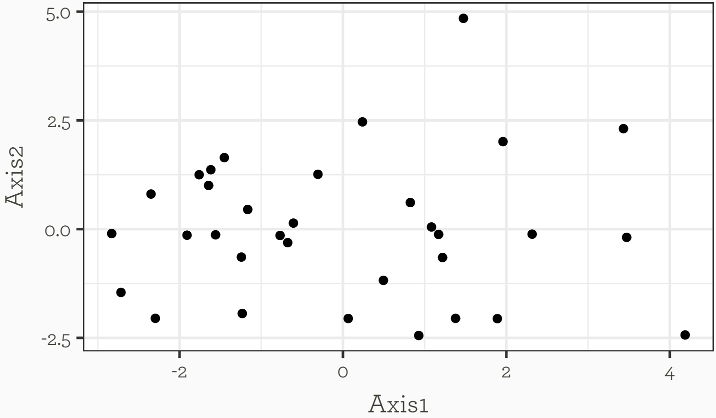
```
s.corcircle(results$co, xax = 1, yax = 2)
```



The recipe for succes

9/10. Visualizing individuals on the factorial axes and interpret

```
results$li %>%
  ggplot(aes(x = Axis1, y = Axis2)) +
  geom_point()
```



The recipe for succes

9/10. Visualizing individuals on the factorial axes and interpret

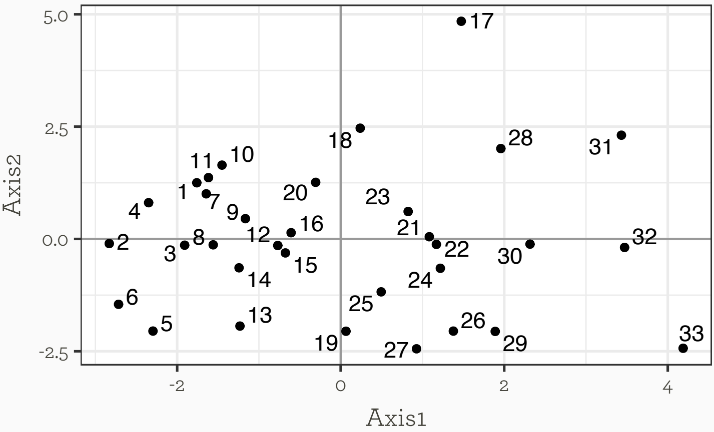
```r
# Load ggrepel
library(ggrepel)

# Individuals plot
pl_indiv <- results$li %>%
  ggplot(aes(x = Axis1, y = Axis2)) +
  geom_hline(yintercept = 0, color = "grey60") +
  geom_vline(xintercept = 0, color = "grey60") +
  geom_point() +
  geom_text_repel(aes(label = decath$athlete))
```

The recipe for succes

9/10. Visualizing individuals on the factorial axes and interpret

```
pl_indiv
```



The recipe for succes

9/10. Visualizing individuals on the factorial axes and interpret

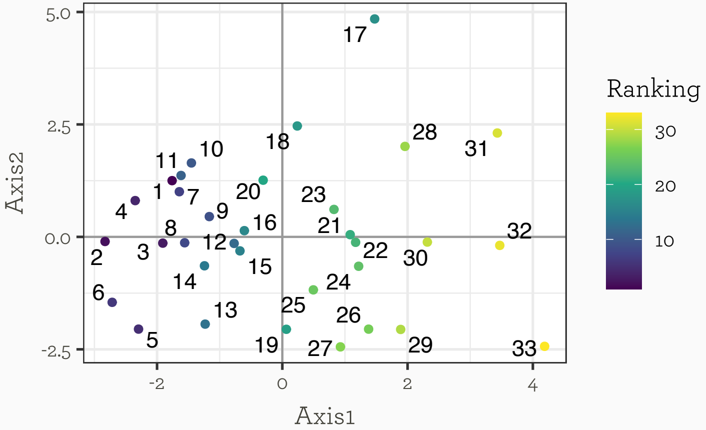
```r
# Load ggrepel
library(ggrepel)

# Individuals plot
pl_indiv2 <- results$li %>%
  ggplot(aes(x = Axis1, y = Axis2)) +
  geom_hline(yintercept = 0, color = "grey60") +
  geom_vline(xintercept = 0, color = "grey60") +
  geom_point(aes(color = as.numeric(decath$athlete))) +
  geom_text_repel(aes(label = decath$athlete)) +
  labs(color = "Ranking") +
  scale_color_viridis_c()
```

The recipe for succes

9/10. Visualizing individuals on the factorial axes and interpret

```
pl_indiv2
```



14. Correspondance Analysis

Outline

Data, example

- Contingency tables : 2 nominal variables M and N with k_M and k_N levels respectively
- Most frequent data type in ecology: stations/species tables
- Either counts (abundances) or presence/absence data

Outline

Objectives of the method

Finding associations between levels

> ***Important***
>
> Are the 2 variables independant, or are they linked in some way?

- Which levels or the first variable are the most identical?
- Which levels or the first variable are the most different?
- Which levels or the second variable are the most identical?
- Which levels or the second variable are the most different?
- Which levels of the first variable are the most strongly associated with which levels of the second variable?

Objectives of the method

Finding associations between levels

Important

Association (or lack of independance) is measured using the χ^2 metric:

$$\chi^2 = \sum_i \frac{(\text{Observed}_i - \text{Expected}_i)^2}{\text{Expected}_i}$$

Contrary to PCA, the correspondance analysis is symmetrical. This means that:

- It doesn't matter which variable is in rows or columns
- We won't need 2 distinct plots to visualize the results: all levels of the 2 variables will appear on a biplot
- No row or column should have a sum of 0

Outline

PCA *vs.* CA

	PCA	CA
Tables	N individuals x K variables	Contingency table: 2 factors
Data	Numerical	Counts, presence/absence
Symmetry	No	Yes
Objectives	Find identical individuals	Find associations between levels:
	Quantify correlations	– within the first variable
	Identify synthetic FA	– within the second variable
	Visualize individuals on FA	– between the 2 variables
Metric	Euclidian distances	χ^2 distances
Matrix	Correlation matrix	Deviation from independance
Method	Eigen values/vectors decomp	Eigen values/vectors decomp
Visualization	Correlation circle	Biplot
	Plot of individuals	
Intrepretation	Loadings of variables on FAs	Distance/Proximity of points
	Synthetic meaning of FAs	
	Position of points along FAs	
Usefull axes	FAs with eigen values > 1	FAs n° 1 and n° 2
Beware of	Discussing axes separaltely	The origin, the Guttman effect
Double zeros	Resemblance	No meaning

Outline

The recipe for succes

1. Look at the raw dataset
2. Preform the CA
3. Look at the eigen values to know how much information is represented on the factorial axes 1 and 2
4. Produce the biplot
5. Interpret the distances as dissimilarities/lack of association

The recipe for succes

1. Raw dataset

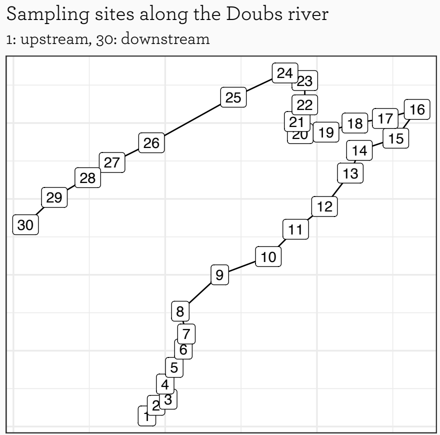
```
# Usefull packages
library(tidyverse)
library(ggrepel)
library(ade4)

# Load data from ade4 package
data(doubs)

# Transform the data
fish <- as_tibble(doubs$fish)

# Dimensions
dim(fish)

[1] 30 27
```



The recipe for succes

1. Raw dataset

```
fish

# A tibble: 30 x 27
    Cogo  Satr  Phph  Neba  Thth  Teso  Chna  Chto  Lele  Lece  Baba
   <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl>
 1     0     3     0     0     0     0     0     0     0     0     0
 2     0     5     4     3     0     0     0     0     0     0     0
 3     0     5     5     5     0     0     0     0     0     0     0
 4     0     4     5     5     0     0     0     0     0     1     0
 5     0     2     3     2     0     0     0     0     5     2     0
 6     0     3     4     5     0     0     0     0     1     2     0
 7     0     5     4     5     0     0     0     0     1     1     0
 8     0     0     0     0     0     0     0     0     0     0     0
 9     0     0     1     3     0     0     0     0     0     5     0
10     0     1     4     4     0     0     0     0     2     2     0
# ... with 20 more rows, and 16 more variables: Spbi <dbl>,
#   Gogo <dbl>, Eslu <dbl>, Pefl <dbl>, Rham <dbl>, Legi <dbl>,
#   Scer <dbl>, Cyca <dbl>, Titi <dbl>, Abbr <dbl>, Icme <dbl>,
#   Acce <dbl>, Ruru <dbl>, Blbj <dbl>, Alal <dbl>, Anan <dbl>
```

The recipe for succes

2. Perform the CA

```r
# Load the ade4 package
library(ade4)

# Determine the number of variables in the dataset
n_var <- min(dim(fish) - 1)

# Perform the CA
resultsCA <- dudi.coa(fish, scannf = FALSE, nf = n_var)

# Compute a table that will be useful for steps 3
contribCA <- inertia.dudi(resultsCA)
inertiaCA <- as_tibble(contribCA$tot.inertia)
```

The recipe for succes

2. Perform the CA

```
resultsCA

Duality diagramm
class: coa dudi
$call: dudi.coa(df = fish, scannf = FALSE, nf = n_var)

$nf: 26 axis-components saved
$rank: 26
eigen values: 0.601 0.1444 0.1073 0.08337 0.05158 ...
  vector length mode    content
1 $cw    27     numeric column weights
2 $lw    30     numeric row weights
3 $eig   26     numeric eigen values

  data.frame nrow ncol content
1 $tab       30   27   modified array
2 $li        30   26   row coordinates
3 $l1        30   26   row normed scores
4 $co        27   26   column coordinates
5 $c1        27   26   column normed scores
other elements: N
```

The recipe for succes

3. Look at the eigenvalues

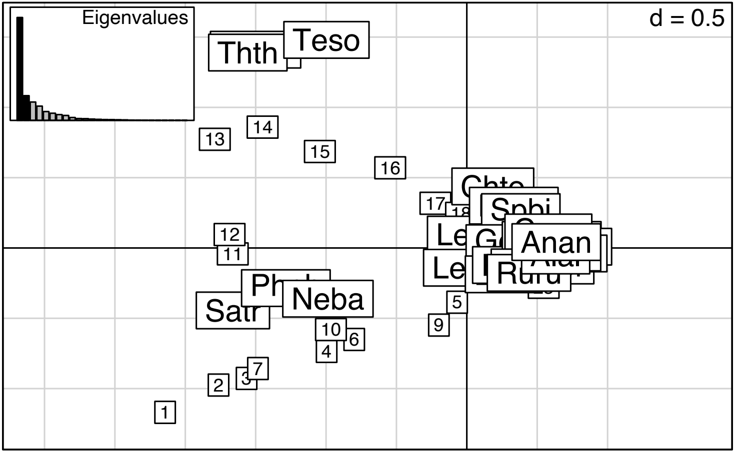
```
# Factorial axes 1 and 2 explain 64% of all variability in the data
inertiaCA

# A tibble: 26 x 3
   inertia   cum `cum(%)`
     <dbl> <dbl>    <dbl>
 1  0.601  0.601     51.5
 2  0.144  0.745     63.9
 3  0.107  0.853     73.1
 4  0.0834 0.936     80.2
 5  0.0516 0.988     84.6
 6  0.0418 1.03      88.2
 7  0.0339 1.06      91.1
 8  0.0288 1.09      93.6
 9  0.0168 1.11      95.0
10  0.0108 1.12      96.0
# ... with 16 more rows
```

The recipe for succes

4. Produce the biplot

```
scatter(resultsCA)
```



The recipe for succes

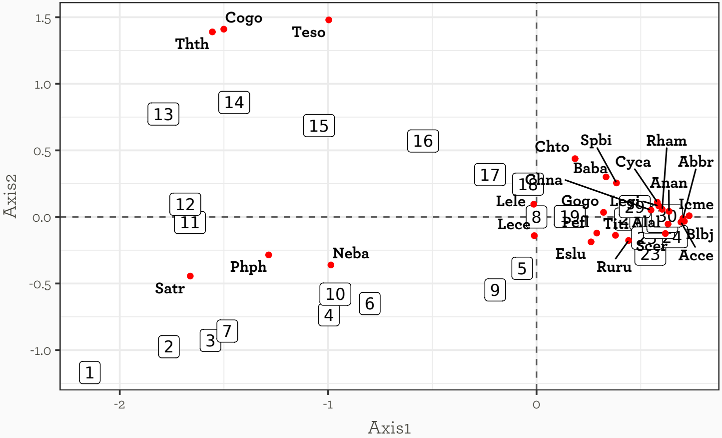
4. Produce the biplot

```
library(ggrepel)
biplot <- ggplot() +
  geom_vline(xintercept = 0, color = "grey35", linetype = 2) +
  geom_hline(yintercept = 0, color = "grey35", linetype = 2) +
  geom_label(data = resultsCA$li,
             aes(x = Axis1, y = Axis2,
                 label = rownames(resultsCA$li)),
             family = "Futura LT Book") +
  geom_point(data = resultsCA$co,
             aes(x = Comp1, y = Comp2), color = "red") +
  geom_text_repel(data = resultsCA$co,
                  aes(x = Comp1, y = Comp2,
                      label = rownames(resultsCA$co)),
                  max.overlaps = 30,
                  family = "ArcherPro-Book")
```

The recipe for succes

4. Produce the biplot

```
biplot
```



The recipe for succes

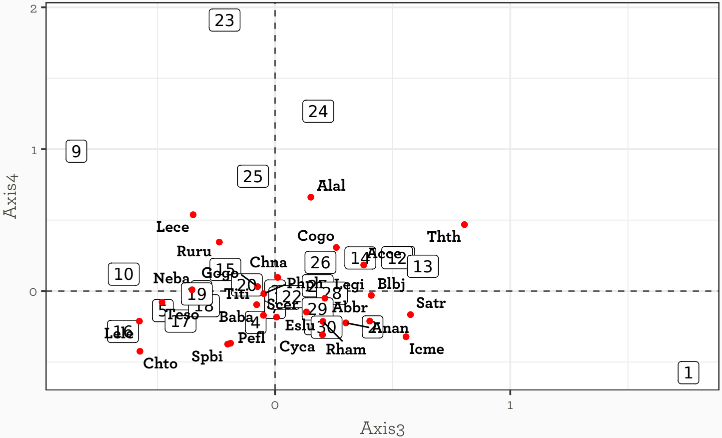
4. Produce the biplot

```
library(ggrepel)
biplot2 <- ggplot() +
  geom_vline(xintercept = 0, color = "grey35", linetype = 2) +
  geom_hline(yintercept = 0, color = "grey35", linetype = 2) +
  geom_label(data = resultsCA$li,
             aes(x = Axis3, y = Axis4,
                 label = rownames(resultsCA$li)),
             family = "Futura LT Book") +
  geom_point(data = resultsCA$co,
             aes(x = Comp3, y = Comp4), color = "red") +
  geom_text_repel(data = resultsCA$co,
                  aes(x = Comp3, y = Comp4,
                      label = rownames(resultsCA$co)),
                  family = "ArcherPro-Book")
```

The recipe for succes

4. Produce the biplot

```
biplot2
```



The Guttman Effect