

A compositional data analysis package for R providing multiple approaches

K.G. van den Boogaart¹, R. Tolosana-Delgado²

¹Ernst-Moritz-Arndt-Universität, Greifswald, Germany; boogaart@uni-greifswald.de

²Universitat de Girona, Spain

Abstract

”compositions” is a new R-package for the analysis of compositional and positive data. It contains four classes corresponding to the four different types of compositional and positive geometry (including the Aitchison geometry). It provides means for computation, plotting and high-level multivariate statistical analysis in all four geometries. These geometries are treated in a fully analogous way, based on the principle of working in coordinates, and the object-oriented programming paradigm of R. In this way, called functions automatically select the most appropriate type of analysis as a function of the geometry. The graphical capabilities include ternary diagrams and tetrahedrons, various compositional plots (boxplots, barplots, piecharts) and extensive graphical tools for principal components. Afterwards, orthon and proportion lines, straight lines and ellipses in all geometries can be added to plots. The package is accompanied by a hands-on-introduction, documentation for every function, demos of the graphical capabilities and plenty of usage examples. It allows direct and parallel computation in all four vector spaces and provides the beginner with a copy-and-paste style of data analysis, while letting advanced users keep the functionality and customizability they demand of R, as well as all necessary tools to add own analysis routines. A complete example is included in the appendix.

Key words: CODA, R, compositional data analysis .

1 Introduction

Compositions is a new R-package freely available from <http://www.cran.r-project.org> or from <http://www.stat.boogaart.de/compositions>. To run it you need a current version of the statistics program **R** (R Development Core Team 2004), which is available for free under the GNU-public License and for all platforms and can be downloaded from <http://www.cran.r-project.org>. Installation procedures are described on the web-pages. A hands-on-getting-started-introduction is provided with the package in the `.../inst/doc` subfolder and also available from its own web-site <http://www.stat.boogaart.de/compositions>. The aim of this paper is to give an overview of the functionality of the package and to discuss its basic working principles. With the package we tried to follow the principles in R as close as possible, even taking them sometimes a step further. Thus, some of the principles discussed here will sound familiar to those knowing R or S in depth. This paper is not a step-by-step introduction. Such an introduction is delivered with the package. A complete example is included in the appendix, to offer an illustration of the concepts treated in this contribution.

Learning curve and overview relevant of sources of help

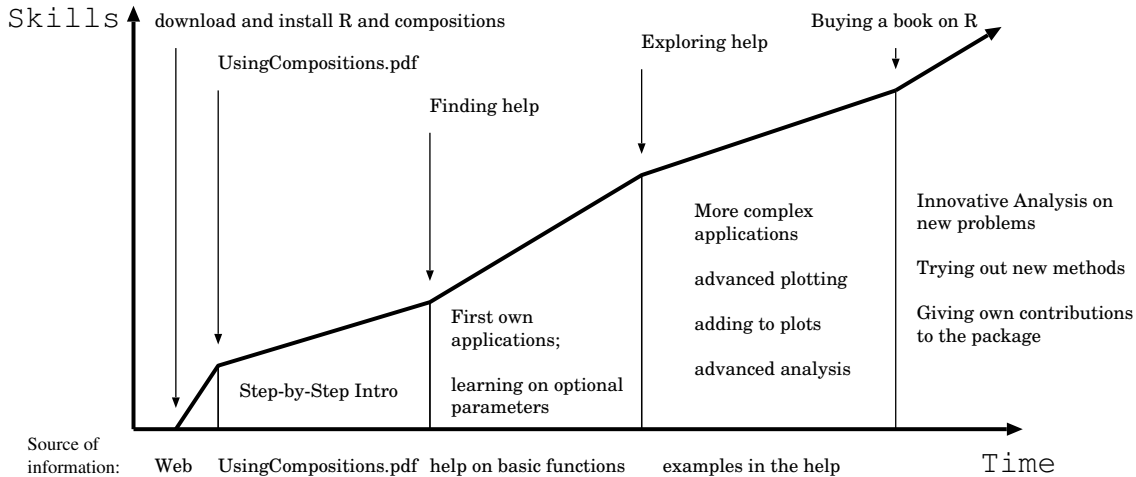


Figure 1: Steps in the learning user paradigm

2 The mathematical and user side concepts of “compositions”

2.1 Documentation for every step on a smooth learning curve

Command-line-based software such as R is felt as a big obstacle to the “see-and-click” user. However a command based interface can handle much more complex communication with the computer. However it is necessary to train that language from the simple first steps on.

The first paradigm of compositions is the paradigm of the learning user. We do expect that the user is willing to read, to explore and to learn. There is a considerable amount of learning needed before one can perform his first own useful analysis. However we provide a shallow learning curve for this steps and on that learned basics one is able to accommodate every growing demand. After downloading and installing R, and downloading the package and UsingCompositions.pdf from the web, one can follow the step-by-step introduction given therein. It guides through the installation process and gives a first tour through the most important features and cases of the package. Afterwards the user will be able to apply what was learned to his own data sets and find possibilities to customize the plots to his own needs, by consulting the help. A detailed help is given for each function including directly working examples and descriptions for every possible parameter. In many cases details, references and hyperlinks to related functions are provided. This concept is illustrated in Figure 1. Clearly this approach is more demanding than a simple menu-and-dialog-based interface. The practical difference between them is best explained with toys: a transformer can do many things and is most desired by children; however, lego building blocks educate, allow the phantasy to dwell, and are good for castles and spaceships, for bridges and horses, for figures and houses, and fascinate children and parents.

2.2 The multiple geometry approach

How statistics should be applied to *amounts of parts in a whole* (positive and compositional data) has been a strongly discussed issue in the last 20 years, since Aitchison(1982)) put forward his log-ratio transformation strategy. This transformation strategy has evolved during the last years into the idea that a geometry underlies any statistical analysis. Various geometries have been used

to analyze positive or compositional data, and there has been argues on which is the right one. The most recent arguments can be found in Rehder&Zier 2002), Shurtz 2003), and Otero et al.2005). In our opinion, choosing the underlying geometry is a step so important, that it shouldn't be left to the software default.

Thus the “compositions” package provides four different geometries (or scales) for such data, which are represented by four different classes called ”rplus”, ”aplus”, ”rcomp”, and ”acomp”. A fifth class ”rmult” representing general multivariate data sets is added for convenience. All four geometries are treated in their own right and without any preference.

rplus The simplest approach is to analyze a multivariate vector of amounts just as a multivariate data set in real geometry. The sample space $(\mathbf{R}_0^+)^D$ is in this cases seen as a convex cone in the (Euclidean) vector space $(R^D, +, -)$. However the only operation defined within this set is the sum. This space is closed under addition, convex combinations (e.g. mean) and multiplication with a positive constant. All other induced operations map amounts into the full \mathbf{R}^D and thus result in an ”rmult”-object.

aplus Often amounts (positive data) are skewed and better analyzed by log-normal models, or on a log scale. A (Euclidean) vector space structure $((\mathbf{R}^+), \oplus, \odot, (\cdot, \cdot))$ is induced on the positive space $(\mathbf{R}_0^+)^D$ by the log transform as a bijective mapping to \mathbf{R}^D :

$$(x \oplus y)_i = x_i y_i, \quad x, y \in \mathbf{R}^D$$

$$(\alpha \odot x)_i = x_i^\alpha, \quad \alpha \in \mathbf{R}, \quad x \in \mathbf{R}^D$$

rcomp If the total sum is constant or irrelevant, a set of amounts can be seen as an element of the real simplex $S_D = \{x \in \mathbf{R}^D : x_i \geq 0, \sum_{i=1}^D x_i = \kappa\} \subset \mathbf{R}^D$, with all its known problems such as spurious correlation (Chayes 1960). However this approach is still the most used one and still has its justification in some cases, e.g. when mass conservation is ensured or additivity of mass is the most important property. The sample space is here a convex subset of \mathbf{R}^D with relative dimension $D - 1$. Only taking convex combinations (such as a mean) map into the simplex again. All other induced operations potentially leave the sampling space and map into the general \mathbf{R}^D .

acomp However, when the total sum is constant or irrelevant, and not the additivity of mass, but e.g. laws of action of masses (chemical equilibrium) are the important feature, or conservation of mass cannot be ensured, the data should be seen as relative compositions, and can be modeled as elements in the Aitchison simplex geometry $A_D = (S_D, \oplus, \odot) \sim (R^{D-1}, +, \cdot)$ induced by the *itr*-Transformation (Egozcue et al. 2003), where \oplus is the perturbation (Aitchison(1986))

$$(x \oplus y)_i = \frac{x_i y_i}{\sum_j x_j y_j}$$

and \odot the power transform

$$(\alpha \odot x)_i = \frac{x_i^\alpha}{\sum_j y_j^\alpha}, \quad \alpha \in \mathbf{R}, \quad x \in S_D$$

Summarizing, the user should first decide whether the data should be analyzed consistently with additivity of mass or consistently with action of masses, and on whether or not the total sum is informative. Then, these answers point to which one of the four geometries should be used, thus setting the class of the data:

```
> mydataset = read.table("MyData.txt",header=TRUE) # Read data from file
> X = acomp(mydataset) # Select class
```

Afterwards every step of the analysis is performed in the geometry linked to the selected class.

2.3 The principle of analogous analysis

It is a policy of the package to treat all four geometries as similar as possible, to allow a direct comparison of results. The same commands can be used for every geometry. The user selects, what he wants to do by choosing a generic command from those existing in R (e.g. `princomp` for principal component analysis or `plot` for plotting), and applies it to the data:

```
> plot(X)                # Plots e.g. Ternary diagrams
> princomp(X)           # Computes principal component analysis
...

```

The computations and plots are then done in the most appropriate way for the selected geometry. This can result in very small differences (e.g. ternary diagrams for "rcomp" and "acom" are almost equal), or in very dramatic ones (e.g. when plotting confidence ellipses). However some operations are not meaningful in one or the other geometry: for instance, centered plotting in ternary diagrams is well-defined (von Eynatten et al. 2002, e.g.) for "acom", while it is meaningless for "rcomp". In such cases a warning or an explicit error is given. The degree of analogy are limited by the different level of structure in the sample spaces: a convex set (for rcomp), a convex cone (for rplus), a vector space with a simple mapping (for aplus), and a vector space with a complex mapping (for acomp).

Technically, the principle of analogous analysis is implemented based on the virtual function mechanism in R. For each of the four classes and each of the basic analysis functions such as `var`, `cov`, `variation`, `summary`, `plot`, `boxplot`, `barplot`, `princomp`, `biplot`, `dist`, `cdt`, `idt`, `ellipses`, `lines`, ..., there exist an overloaded routine named *functionname.classname* (e.g. `var.acomp`) which is called, when the user gives a command of the form *functionname(AnObjectOfClass)* (e.g. `var(X)`). Correspondingly the help on this functions can be found by a command like `? var.acomp` rather than `?var` which gives the help of the standard routine, i.e. for real data.

2.4 Default transforms and the principle of working in coordinates

To achieve maximum analogy between the different approaches we use the principle of working in coordinates (Pawlowsky-Glahn 2003) for all four classes. Before the analysis, the data is mapped by an isometric transformation, like *ilr* into \mathbf{R}^D or a subset of \mathbf{R}^D . These transforms are in general defined as mappings from one of the spaces to the real space represented by the "rmult" class. Depending on the purpose, we can choose between different mappings. For "acom" the transforms are the well known additive logratio- $alr(x) = (\ln x_i - \ln x_D)_{i=1, \dots, D-1}$, centered logratio- $clr(x) = \left(\ln x_i - \frac{1}{D} \sum_{i=1}^D \ln x_i \right)$ and the isometric logratio- transform $ilr(x) = B_D clr(x)$, for some special $B_D \in \mathbf{R}^{D-1 \times D}$. *clr* is used whenever the relation to the original parts must be preserved and *ilr*, when it is important to have a surjective mapping giving a non-singular variance matrix. The *alr*-transform is never used automatically, since it is not isometric.

For the "rcomp" simplex we defined analogous linear transformations:

additive planar transform: $apt(x) := (x_i)_{i=1, \dots, D-1}$, which just deletes the last part,

centered planar transform: $cpt(x) := (x_i - \frac{1}{D})$, mapping a composition into a vector of components summing up to zero,

isometric planar transform: $ipt(x) := B_D cpt(x)$, .

The computations are performed on the image space and then either back-transformed or somehow represented in a way meaningful in the original scale. To achieve that we have defined a centered default transform `cdt` and an isometric default transform `idt`. The centered default transform is an isometric injective mapping into \mathbf{R}^D preserving the meaning of the parts, however not necessarily

surjective. The isometric default transform is a isometric mapping to \mathbf{R}^K , with some $K \leq D$ and thus not preserving dimension. However, its image has full relative dimension in \mathbf{R}^K .

Since both properties can be met in one single mapping for "rplus" and "aplus" it is only necessary to define a single mapping for them. This is the isometric identity transform for "rplus" (iit, simple inclusion mapping to \mathbf{R}^D) and the isometric log transform (ilt, the componentwise log). The cdt and the idt are then defined as given in Table 1.

Table 1: Definition of the default transforms for the different classes.

	cdt	idt
acomp	clr	ilr
rcomp	cpt	ipt
aplus	ilt	
rplus	iit	

2.5 The principle of computation in the vector space

All four geometries form vector spaces or subsets of vector spaces. The package philosophy is to allow computation in these vector spaces by the standard operators +, -, *, /, %*%.

2.5.1 Sum and difference

+ and - are addition and subtraction of two vectors. When the result is meaningful in the original scale it is reported in that scale. When it is not meaningful in the scale but as a vector it is reported as a "rmult" object. When the result is not meaningful at all, an error is given. That means that + applied to two "acomp" compositions results in the perturbation, and that + applied to two "rcomp" compositions is meaningless and will result in an error. In contrast, the difference of two "rcomp" compositions is an increment vector of class "rmult", which can be added to an "rcomp" vector to result in an "rcomp" vector.

2.5.2 Scalar multiplication

* and / are multiplication with a scalar or its inverse, thus one of the arguments must be a scalar, while the other might be a vector. Again, when the result is meaningful in the original scale it is reported in that scale, when it is not meaningful in the scale but as a vector it is reported as a "rmult" object, and an error is given when the result is not meaningful at all. That means that * applied to an "acomp" composition results in the power transform, and that * applied to two "rcomp" compositions is meaningless and will issue an error.

2.5.3 Moments

With commands like `mean(X)` or `var(X)`, `cov(X,Y)`, `variation(X)`, `mvar(X)`, `msd(X)` it is possible to compute moments of the data sets. This is meaningful for all scales, since all scales are at least convex subsets of vector spaces. The results are computed in the coordinates. If the result can be interpreted as a vector of the original set, it is reported as such. If the result is a tensor in the space (e.g. for `var`) it is reported as a matrix in the coordinates implied by the `cdt`-transform. These matrices can be converted to the matrices in `idt` basis, with `clrvar2ilr`, when both differ. If the result is a scalar (e.g. for metric variance `mvar`) it is reported as a single number. The variation matrix is reported as it is, as a matrix.

2.5.4 Scalar product and matrix multiplication

The operator for scalar product and matrix multiplication is `%*%`. If you multiply two amount objects with that operator it gives the scalar product in the specified geometry. For instance, the following two commands give the same result: `norm(X)^2` and `X %*% X`.

If you multiply one of the amount objects with a matrix like in `powerofpsdmatrix(var(X),-1/2) %*% (X-mean(X))` the linear mapping encoded in the matrix is applied to the vectors. The matrix can either be given in `cdt` or `idt` coordinates at your option. In this case we would get a fully normalized data set with variance 1 in all directions. As we can check with:

```
> var(powerofpsdmatrix(var(X),-0.5) %*% (X-mean(X)))
var(powerofpsdmatrix(var(X),-0.5) %*% (X-mean(X)))
  [,1] [,2] [,3] [,4] [,5]
[1,]  0.8 -0.2 -0.2 -0.2 -0.2
[2,] -0.2  0.8 -0.2 -0.2 -0.2
[3,] -0.2 -0.2  0.8 -0.2 -0.2
[4,] -0.2 -0.2 -0.2  0.8 -0.2
[5,] -0.2 -0.2 -0.2 -0.2  0.8
```

3 R principles applied to compositions

"compositions" tries to follow the paradigms of R as close as possible for the following reasons:

- those who know R can learn it very fast
- those who do not know R, can learn at the same time R and "compositions"
- R provides a large portion of flexibility for free (e.g. graphical parameters)
- the R principles are already proven to be useful and likely to prevail

The most important paradigms are discussed here.

3.1 Data is stored in objects

The compositional data is stored in a classed object, which is derived from a standard R matrix with rows and columns. It is marked with a class attribute, which tells you the type of data each time the data is printed.

```
> X
      Cu      Zn      Pb      Cd      Co
[1,] 0.424629320 0.419450330 0.15016391 3.024494e-03 2.731947e-03
[2,] 0.106808985 0.121425471 0.77174518 1.015393e-05 1.020898e-05
[3,] 0.014877911 0.028036695 0.95693185 7.140262e-05 8.213902e-05
[4,] 0.036551756 0.058622001 0.89799359 2.079226e-03 4.753427e-03
...
[59,] 0.332754787 0.377426426 0.28233184 4.174645e-03 3.312304e-03
[60,] 0.091526365 0.606590444 0.28135556 1.335939e-02 7.168243e-03
attr("class")
[1] "acomp"
```

3.2 Data-driven analysis and generic functions

Based on the class of the data a different routine is called that performs the analysis within the framework of the selected scale. This is sometimes done by generic functions and sometimes automatically by calling further functions, which behave generically. A generic function is a wrapper which recognizes the class of its arguments and calls a specific function adapted to that class. An example of a generic function is for instance the following set of functions

```
cdt          <- function(x) UseMethod("cdt",x)
cdt.default <- function(x) x
cdt.acomp   <- clr
cdt.rcomp   <- cpt
cdt.aplus   <- ilt
cdt.rplus   <- iit
```

Note the conciseness of these definitions. A simpler example might be the cov routine:

```
function (x, y = NULL, ...) {
  cov(cdt(x), cdt(y), ...)
}
```

The covariance is just defined as the covariance of the centered default transforms. Note that these two data sets do not need to be of the same type. This is a clear example, that the principle of working in coordinates is not only a principle of *thinking*, but also a principle of *programming*.

Since compositional data analysis is multivariate by nature, the routines need to be adapted to the dimension of the data set. It is e.g. not sufficient to draw a single ternary diagram, when plotting five-part data:

```
> plot(X)          # plots a newly defined ternary diagram matrix, Fig. 2
> boxplot(X)       # plots boxplots of all pairwise ratios in log-scale, Fig. 3
```

Both these plots are newly defined in the package and consistent with the assumed geometry (see Figures 2 and 3 for results).

3.3 Results are stored in objects

The last principle goes even further. When we perform statistical computations in R the result is typically an object. This means, that we can apply further functions to the result to plot it in a specific way or to extract further information. For instance,

```
> pr = princomp(X)          # computes PCA in acomp geometry
> plot(pr)                  # plots the screeplot of the PCA
> plot(pr,type="biplot")    # plots the biplot of the PCA
```

Furthermore the results still carry the information on which type of analysis they are based on and can thus be treated differently by subsequent analysis.

```
> class(pr)                # class of the result
[1] "princomp.acomp" "princomp"
> plot(pr,type="relative") # plots compositional loadings
```

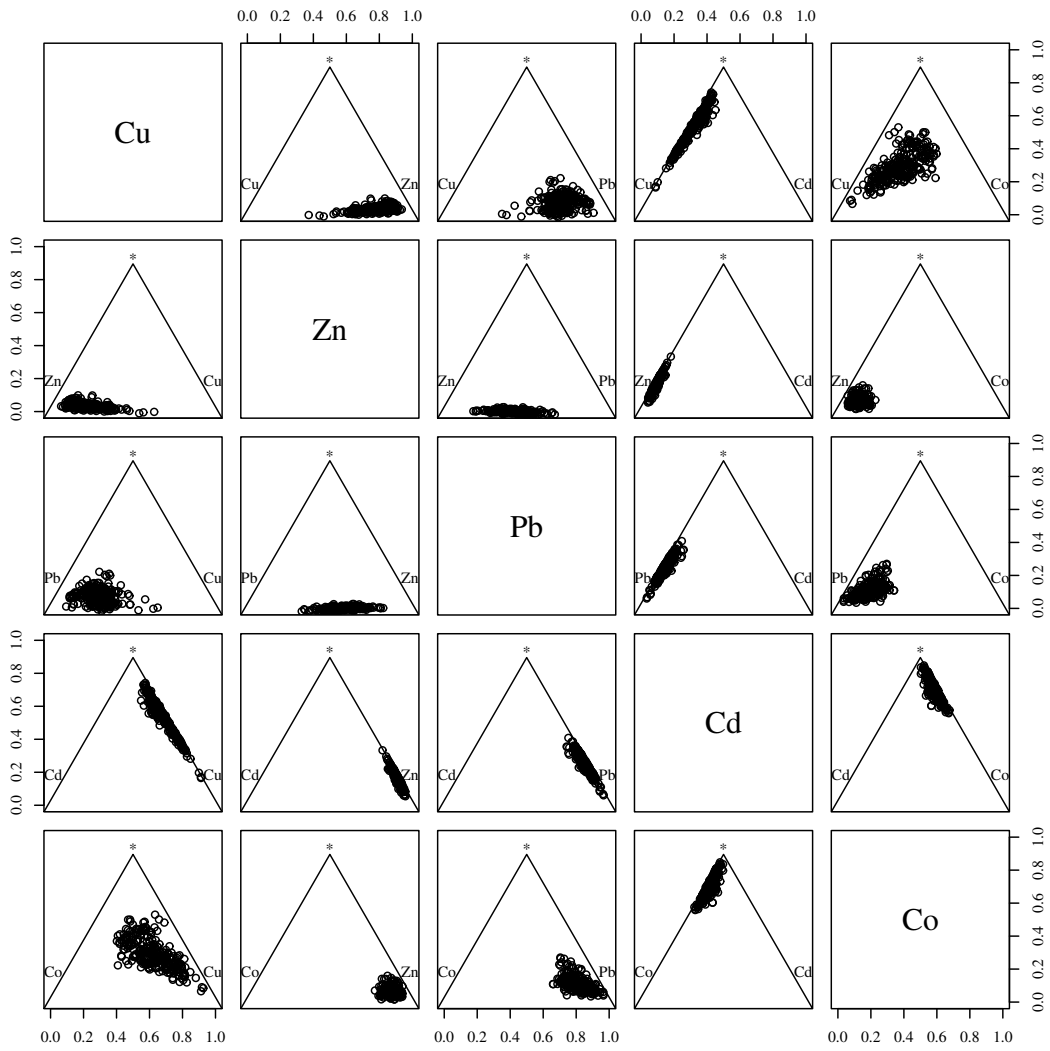


Figure 2: Result of `plot(X)`, with `X` an `acomp` or `rcomp` object (no difference in this case).

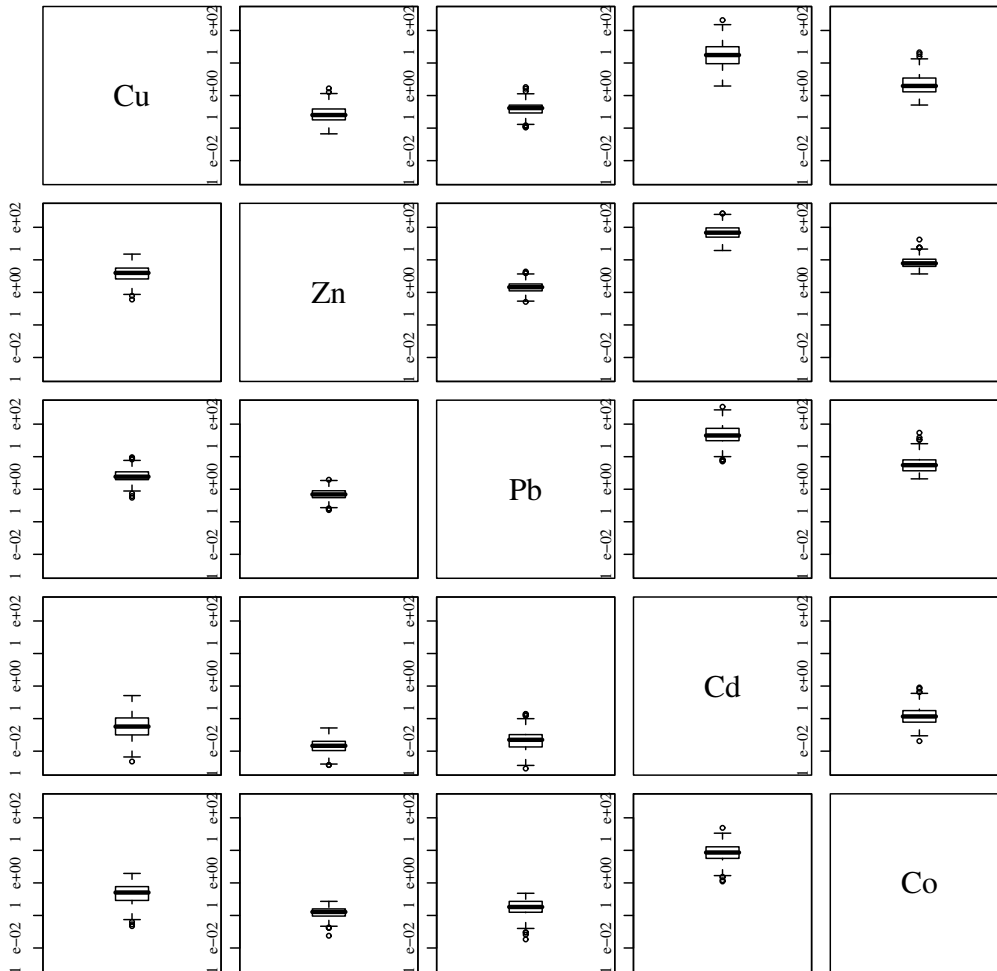


Figure 3: Result of `boxplot(X)`, with `X` an `acomp` object. Note that each cell contains the box-plot of the log-ratio of the parts of the corresponding row and column, thus the matrix of box-plots is "antisymmetric".

As one can see the result has two classes. One is the standard class `princomp` and allows to treat the object with all R-routines that are available for postprocessing principal component analysis. The other is a special class for PCA-results in `acomp` geometry and provides us with additional functionality specially designed for this situation, e.g. the relative plot of loadings.

3.4 Optional Parameters extend functionality

Like this `type="relative"` parameter, which can be present or not, there are many optional parameters to the commands, which modify or extend their functionality. In the first steps, it is not necessary to know about these parameters to use the commands. However, when a more complex analysis is needed one can find the possible optional parameters in the help and gain more flexibility by using them.

3.5 Standard plot parameters are used

A special case of those optional parameters are the standard plotting parameters such as `color` (`col=`), `line width` (`lwd=`), `margins` (`mar=`), `x` and `y-axis limits` (`xlim=`, `ylim=`), and the `main` title (`main=`) of the plot. These parameters are available on most functions, were they make any sense and are not documented with the function, but in the help to the standard R-routine `par`, which stands for plotting-parameters.

This principle holds not only for plots, but also for other standard analysis parameters, like `not available remove` (`na.rm`), in generic analysis routines such as variance or mean. This parameter controls the way missing values are treated.

3.6 Adding to existing plots

The usage of this standard parameters develop their full flexibility together with another central feature of R: adding more data, descriptions and additional features to existing plots. We extended this functionality of R also to multi-panel plots, for which we are particularly fond of. For instance, this can be used to add a confidence region to a ternary diagram matrix.

```
> plot(X) # plot ternary diagram
> ellipses(mean(X), var(X)/nrow(X)*qf(0.975, ncol(X)-1, nrow(X)), col="red")
> # and add the confidence ellipses of the mean in red.
```

The key here is to decide that `qf(0.975, ncol(X)-1, nrow(X))` is the correct quantile, and not the technical problem of drawing the ellipse. A routine doing this automatically should be added soon.

3.7 Parallel computation on data sets

The classical advantage of worksheet programs is that they allow the user to handle much data in at a time. However, R is simpler in this aspect than most worksheets. The whole compositional data set can be handled in simple formulae at once. For instance, a command like

```
> Y = (X-mean(X))/msd(X)
```

results in a centered and standardized version of `X`. Beware of the fact, that the meaning of the `-` here is inverse perturbation since `X` is an `acomp` object. Similarly, `/` denotes the inverse power transform. However the major R advantage is the list-wise computation: a data set of compositions is treated as a list of vectors. When we operate a list of vectors with a single vector or a single scalar by a operation like `+`, `-`, `*`, `/`, `%*%` (the inner product), the same operation is applied to

every entry of the list and a list is returned. When we operate a list with a list, then the first entry of the first list is operated with the first entry on the second list, the second with the second and so on. The same happens with functions. A function applied to a list of vectors results in a list of results. For instance,

```
> norm( X - X )
[1] 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
[39] 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
>
```

gives a list of zeros, which is the evident result of subtracting \ominus from row of X itself, resulting in $(1/5, 1/5, 1/5, 1/5, 1/5)$, and then taking the Aitchison norm of this composition. Other functions like `mean`, `standarddeviation` and `variance`, take a list and result in a single composition or matrix. These concepts makes the computation on lists very easy and results in very simple formulae for complex computations.

4 Features of the package

The features of the package are best enumerated in catchwords

4.1 Computation

- Transforms: `alr`, `clr`, `ilr`, `apt`, `cpt`, `ipt`, `ilt`, `iit`, `cdt`, `idt`, `ult`
- Inverse Transforms: `alr.inv`, `clr.inv`, `il.inv`, `apt.inv`, `cpt.inv`, `ipt.inv`, `ilt.inv`, `iit.inv`
- Vectorized computation: `+`, `-`, `*`, `/`, scalar product and linear mapping `%*%`
- Moments: `mean`, `var`, `variation`, `mvar`(metric variance), `mstd`(metric standardeviation), `mcor` (metric correlation)
- Simulation:
 - Simplex Uniform `runif.acomp`
 - Dirichlet: `rDirichlet`
 - Normal distribution in each scale `rnorm`
 - Multivariate Lognormal `rlnorm`

We still look for a contributor for the Aitchison distribution.

- Other:
 - Additive and multiplicative marginal compositions `acompmargin`, `rcompmargin`
 - Subcompositions (`acomp(X,parts=...),...`)
 - `endpointCoordinates`
 - `totals` of rows
 - `summary` of data sets

4.2 Analysis

Regarding complex statistical analysis, a principle of the package is to rely on existing routines in R, when they can be easily combined with basic functions of the “compositions”-package. A few routines specific to compositions have nevertheless been added. In our opinion, too many specific interface reduces the flexibility of R.

- *Principal Component Analysis*

There is a big extension interface for PCA including

- compositional (clr-transformed) biplots, screeplots, compositional loading plots
- expression of loadings as compositions
- correction of the rank of the matrix

- *Cluster Analysis*

Cluster Analysis can be done with the standard R-routine for Cluster Analysis (`hclust`). It has to be nevertheless based on differences valid for the simplex, computed with the `dist` routines available in the package. This follows the simple paradigm of generic functions. The commands to be given are identical to those for a standard cluster analysis in R including all possible options. The adaption to a compositional geometry is handled automatically once the data set is given a compositional class.

```
> cl = hclust(dist(X,method="manhattan"),method="complete")
> plot(cl) # plot the Dendrogram
```

It is worth commenting that it is perfectly valid to use a city block (Manhattan) metric here, which is computed in the `cdt` transform and is fully valid in the geometry of the simplex. Cluster analysis are not always based on Euclidean distances, thus Aitchison distance is not a must in compositional cluster analysis. The results of the `hclust` are not different from the non-compositional case and can be treated similar. For instance, by using the standard routine `cutree(cl,k)` we would get a factor variable with the groups obtained cutting the dendrogram of `cl` at `k` groups.

- *Discrimination Analysis*

Similarly, there is no need of specific compositional routines for discrimination analysis. If the discrimination information is provided by compositions, it is enough to wrap the data set by the `idt`-Transform and to use a list specifying both the discrimination dataset and the grouping factor. For instance, by using any of the functions `lda`, `qda`, `fda`, ..., and `predict` from the MASS package, we could write

```
> lda(groups~Y,data=list(groups=groups,Y=idt(X)))
...
```

No further provisions are necessary and not giving a new interface keeps all the flexibility present in a growing system like R. Note: to load the MASS package, we must type once the command `library("MASS")`.

- *Linear Models*

The same paradigm is valid for linear models such as regression or analysis of variance, which in this way can use compositional data-types as dependent or independent variables. If the response is compositional, the residuals and variances are given in `idt`-coordinates and can be back-transformed with the appropriate back-transform (`iit.inv`, `ilt.inv`, `ipt.inv` or `ilr.inv`).

- *Normal confidence regions*

Confidence regions can be plotted based on the `ellipses` command. There is still a discussion in the team on how should be the user interface for confidence region routines (which parameters, default values, etc.), so they are likely to be added in near future.

4.3 Plotting

- Ternary diagram, scatter plot and log-log-plot and matrices of these plots
 - Euclidean and Aitchison straight lines
 - Euclidean and Aitchison segments
 - Euclidean and Aitchison ellipses
 - isoproportion lines
 - isoportion lines
 - centering and scaling in aplus and acomp geometries
 - grouped plotting, colors, symbols, adding to plots, labels, formulae,...
- compositional and relational boxplots specific to compositions
- several types of barplots specific to compositions
- pie-charts were already present in R
- compositional Normal Quantile-Quantile plots.
- diagnostic plots for principal component analysis

5 Conclusions

The package is fit for a broad range of users from the beginner, over the expert to the developer. It requires nevertheless the will to learn. The flexibility in the package is wide enough to potentially incorporate any multivariate analysis methodology already implemented in R into a compositional context. This is not done by adding ad-hoc interfaces, but by adding simple routines, which fit well into the R context and bridge R-packages to its use with compositions and amounts. Thus, in our opinion, the effort of learning R and "compositions" is by far compensated by the unlimited availability of routines in R specific for *any* analysis.

Acknowledgements

This package was mostly programmed and documented during a research stage of several months in Greifswald (Germany) of the second author. We thank funding from the *Deutsche Akademische Austauschdienst* (German Academic Exchange Service, Ref: 314-A/04/33586), the *Agència de Gestió d'Ajuts Universitaris i de Recerca* (catalan Agency for Management of University and Research Grants, Ref: 2004-BE-00147) and the International Association for Mathematical Geology.

REFERENCES

- Aitchison, J. (1982). The Statistical Analysis of Compositional Data. *Journal of the Royal Statistical Society, Series B (Statistical Methodology)*, 44 , 139-177
- Aitchison, J. (1986). *The Statistical Analysis of Compositional Data*. Monographs on Statistics and Applied Probability. Chapman & Hall Ltd., London (UK), 416 p.
- Barceló-Vidal, C., Martín-Fernández, J. A., Pawlowsky-Glahn, V. (2001). Mathematical foundations of compositional data analysis. In: Ross, G. (Ed.), *Proceedings of IAMG'01 — The sixth annual conference of the International Association for Mathematical Geology*. Vol. CD-. p. 20 p, electronic publication.

- Chayes, F. (1960). On correlation between variables of constant sum. *Journal of Geophysical Research* 65 (12), 4185–4193.
- Egozcue, J. J., V. Pawlowsky-Glahn, G. Mateu-Figueras, and C. Barceló-Vidal (2003). Isometric logratio transformations for compositional data analysis. *Mathematical Geology* 35(3), 279–300.
- Egozcue, J.J. and V. Pawlowsky-Glahn (2005) Groups of Parts and their Balances in Compositional Data Analysis, *Mathematical Geology*, in press
- Otero, N., Tolosana-Delgado, R., Soler, A., Pawlowsky-Glahn, V. and Canals, A. (2005). Relative vs absolute analysis of compositions: a comparative analysis in surface waters of a Mediterranean river. *Water Research* (in press).
- Pawlowsky-Glahn, Vera and Egozcue, Juan José (2001). Geometric approach to statistical analysis on the simplex. *Stochastic Environmental Research and Risk Assessment (SERRA)* 15(5), 384–398.
- Pawlowsky-Glahn, V., Egozcue, J. J., (2002). BLU estimators and compositional data. *Mathematical Geology* 34 (3), 259–274.
- Pawlowsky-Glahn, Vera (2003). Statistical modelling on coordinates. In: Thió-Henestrosa and Martín-Fernández(2003) *Compositional Data Analysis Workshop – CoDaWork’03, Proceedings*. Universitat de Girona, ISBN 84-8458-111-X, <http://ima.udg.es/Activitats/CoDaWork03/>.
- Pawlowsky-Glahn, V., G. Mateu-Figueras (2005). The Statistical Analysis on Coordinates in Constrained Spaces, in International Statistical Institute. Session (55th :, 2005 : Sydney, N.S.W.) (2005) *Abstract book : 55th session of the International Statistical Institute (ISI)*, 5-12 April 2005, Sydney Convention & Exhibition Centre, Sydney, Australia. ISBN 1-877040-28-2
- Pearson, K., (1897). Mathematical contributions to the theory of evolution. on a form of spurious correlation which may arise when indices are used in the measurement of organs. *Proceedings of the Royal Society of London LX*, 489–502.
- R Development Core Team (2004). *R: A language and environment for statistical computing*. Vienna, Austria: R Foundation for Statistical Computing. ISBN 3-900051-00-3.
- Rehder, S. and Zier, U. (2002), Some remarks about transformations. In: Bayer, Burger, Skala (Eds.), *Proceedings of IAMG’02 — The eight annual conference of the International Association for Mathematical Geology*. Vol. I and II. Berlin (D), pp. 423–428.
- Shurtz, Robert F. (2003). Compositional geometry and mass conservation. *Mathematical Geology* 35 (8), 972–937.
- von Eynatten, H., Barceló-Vidal, C., Pawlowsky-Glahn, V., (2003). Modelling compositional change: the example of chemical weathering of granitoid rocks. *Mathematical Geology* 35 (in press).
- von Eynatten, H., Pawlowsky-Glahn, V., Egozcue, J. J. (2002). Understanding perturbation on the simplex: a simple method to better visualise and interpret compositional data in ternary diagrams. *Mathematical Geology* 34 (3), 249–257.

Appendix: a practical example

Basic descriptive statistics and plots

To show the practical usefulness of these concepts, we have treated a set of four cations (Na, Mg, Ca and NH_4) monthly measured in two rivers during three years, in the Llobregat River Basin (NE, Spain). A full account of the characteristics of the basin and the data set can be found in Otero et al.2005). We will only mention here that both rivers flow from unpolluted areas of the lower Pyrenees through agricultural areas to Manresa, a medium-sized city placed in the middle of the basin, where they join. The landscape is dominated by limestones in the upper part, which progressively pass to tertiary clastic sediment infills of the basin, with an important diapyric halide outcrop. This data set is included in the package, and can be loaded by typing the instruction `> data(Hydrochem)`. The character `>` is the command line prompt of R, and tells us that the program is waiting our instructions. It must not be typed when copying the instructions.

Once loaded the data set, we may select the compositional variables (stored in `Z`) and the river variable (stored in `riv`), by the set of instructions

```
> river = factor(Hydrochem[, "River"])
#preliminarily takes the river variable
> take = as.logical((river=="Cardener")+(river=="UpperLLobregat"))
# select those cases measured in the two interesting rivers
> Z = Hydrochem[take, c("Na", "Mg", "Ca", "NH4")]
# select the interesting subcomposition
> riv = factor(Hydrochem[take, "River"])
# select the interesting rivers
```

Note that lines after the symbol `#` should not be typed, since they are comments ignored by R.

A summary of the main descriptive statistics of this data set, in any of the four geometries can be obtained by simply declaring the data set as an object of the corresponding class, and calling the function `summary` on it. Equivalently, scatter plots and box plots may be obtained in the same way. The next sections show the results of these computations.

rplus geometry

```
> Z=rplus(X)
> summary(Z)
      Na      Mg      Ca      NH4
Min.   2.233  4.545 49.13  0.003075
1st Qu. 16.760 10.050 80.32  0.055090
Median  33.860 19.080 88.50  0.105900
Mean   108.400 26.020 95.76  0.732400
3rd Qu. 187.100 37.670 104.00 0.231900
Max.   958.100 104.500 216.80 11.000000
attr(,"class") [1]
"summary.rplus" "matrix"
> boxplot(X, fak=riv)
# generates figure 4, but not yet implemented in this direct form
> plot(X, pch=18+as.integer(riv), col=c("red","blue")[as.integer(riv)])
# generates figure 5, with symbols and colors
# controlled by "pch" and "col" optional parameters
```

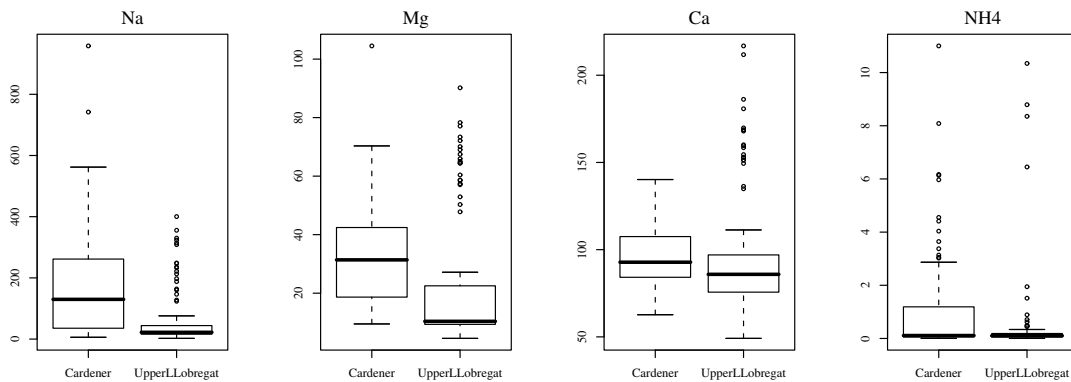


Figure 4: Matrix of box-plots plots in rplus geometry.

Results of the function `summary` are given in the same units than the data set, in this case mg/l. Boxplots (Figure 4) suggest a possible difference between the two rivers, the Cardener and the Upper Llobregat: the first is richer in all four cations but specially in Na and Mg. This is a well-known characteristic, linked to bigger salt tailings (as well as the only natural outcrop in the region) in the basin of the Cardener. All four, but specially NH_4 , show a long upper tail (many samples considered as atypical), which is a classical reason to take a log geometry. Scatter plots (Figure 5) show this long tail too (clearly for NH_4 , subtler in Mg), but also a striking proportional relationship between Na, Mg and Ca, where each river has a different slope. This is a suggestion that the difference between the two rivers will be found as a ratio or a difference.

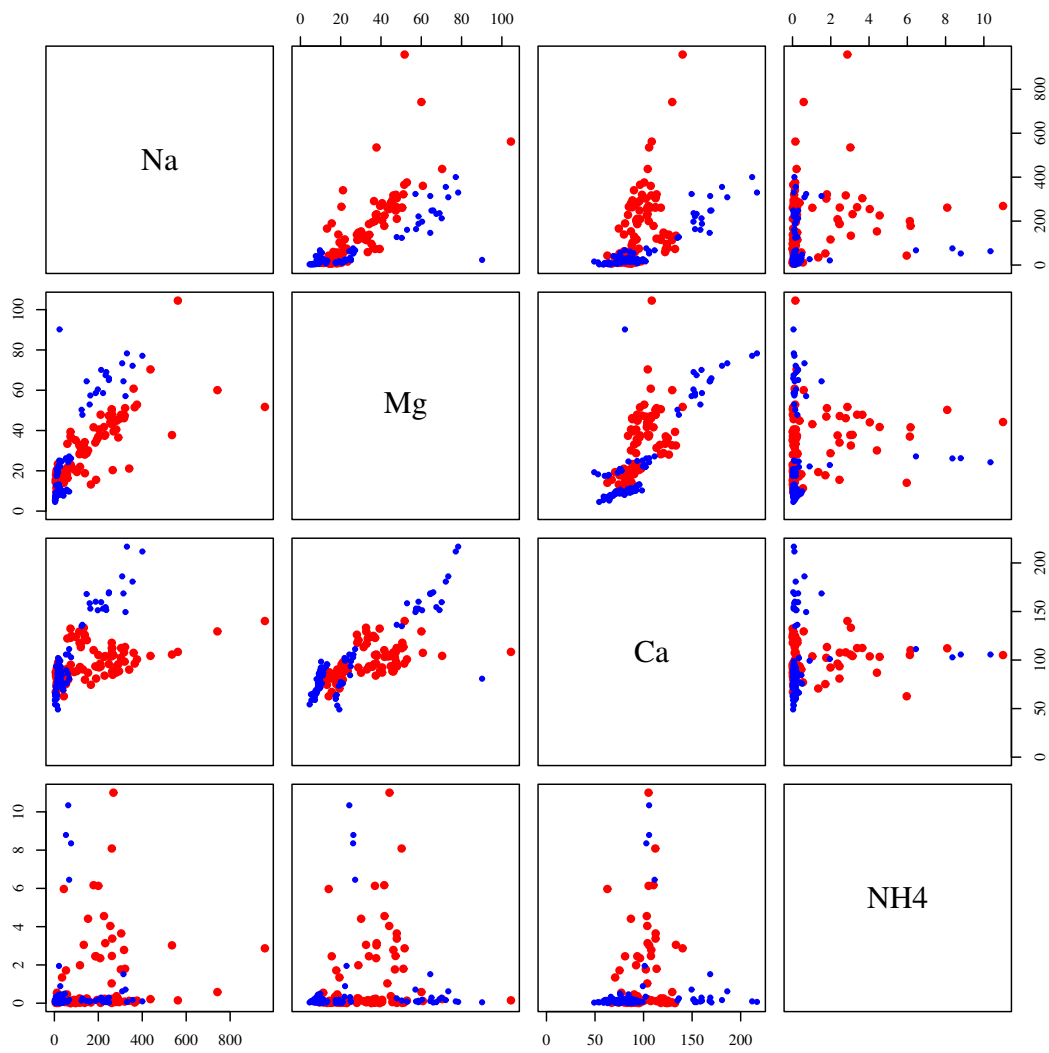


Figure 5: Matrix of scatter plots in rplus geometry.

aplus geometry

```

> Z=aplus(X)
> summary(Z)
      Na      Mg      Ca      NH4
Min.  2.233  4.545  49.13  0.003075
1st Qu. 16.760 10.040 80.32  0.055080
Median 33.860 19.080 88.50  0.105900
Mean  46.590 19.880 92.40  0.144100
3rd Qu.187.000 37.670 103.90 0.231900
Max.   958.100 104.500 216.80 11.000000
attr(,"class")
[1] "summary.aplus" "matrix"
> boxplot(X, fak=riv)
# generates figure 6, but not yet implemented in this direct form
> plot(X, pch=18+as.integer(riv), col=c("red","blue")[as.integer(riv)])
# generates figure 7, with symbols and colors
# controlled by "pch" and "col" optional parameters

```

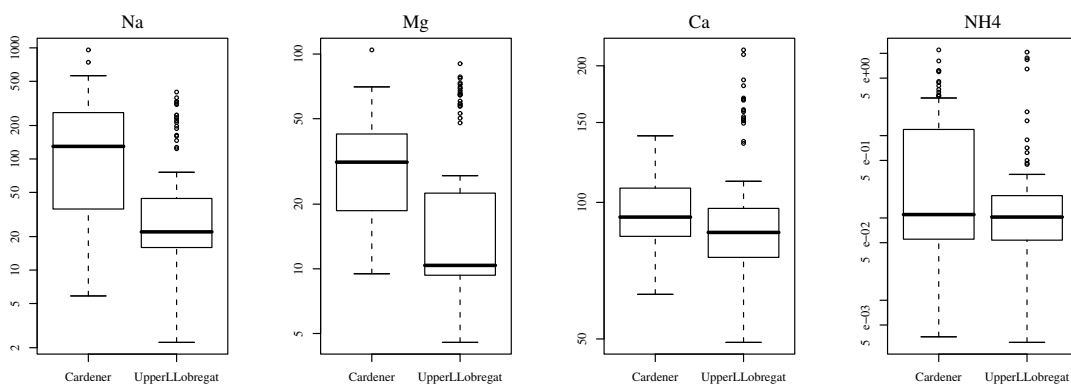


Figure 6: Matrix of box-plots plots in aplus geometry. Note the vertical logarithmic scale.

A log geometry could be suggested either by the long upper tails observed in the last analysis, or by theoretical arguments (well-known to the compositional data analysis community). The main differences between the results obtained with aplus and rplus geometries lay in the means and in the scatter plots. Comparing both summaries, the aplus mean (geometric mean) is clearly nearer to the median than the rplus mean (arithmetic mean), which is a well-known result. The scatter plots (Figure 7) suggest that, fixed a value of Mg or Ca, Cardener samples have higher Na values than Llobregat ones. Again, a suggestion to take log-ratios is implicit here.

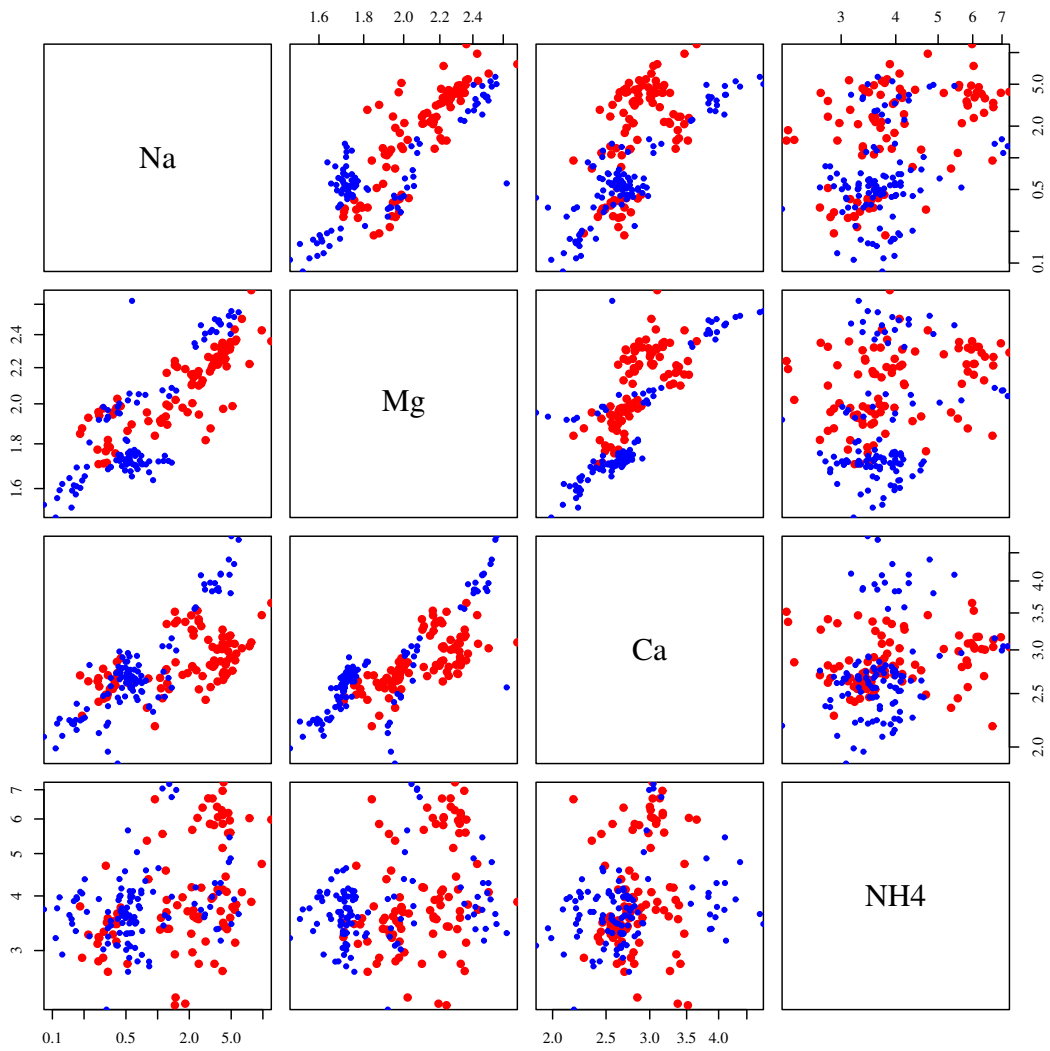


Figure 7: Matrix of scatter plots in aplus geometry. Note the log-log scale.

rcomp geometry

```
> Z=rcomp(X)
> summary(Z)
      Na      Mg      Ca      NH4
Min. 0.03369 0.04485 0.1216 1.469e-05
1st Qu. 0.14260 0.08716 0.3504 3.029e-04
Median 0.25950 0.10620 0.6042 6.443e-04
Mean 0.32900 0.11390 0.5541 2.953e-03
3rd Qu. 0.51280 0.12970 0.7388 1.780e-03
Max. 0.83110 0.46400 0.8829 5.089e-02
attr("class")
[1] "summary.rcomp" "matrix"
> boxplot(X, fak=riv)
# generates figure 8
> plot(X, pch=18+as.integer(riv), col=c("red","blue")[as.integer(riv)])
# generates figure 9, with symbols and colors
# controlled by "pch" and "col" optional parameters
```

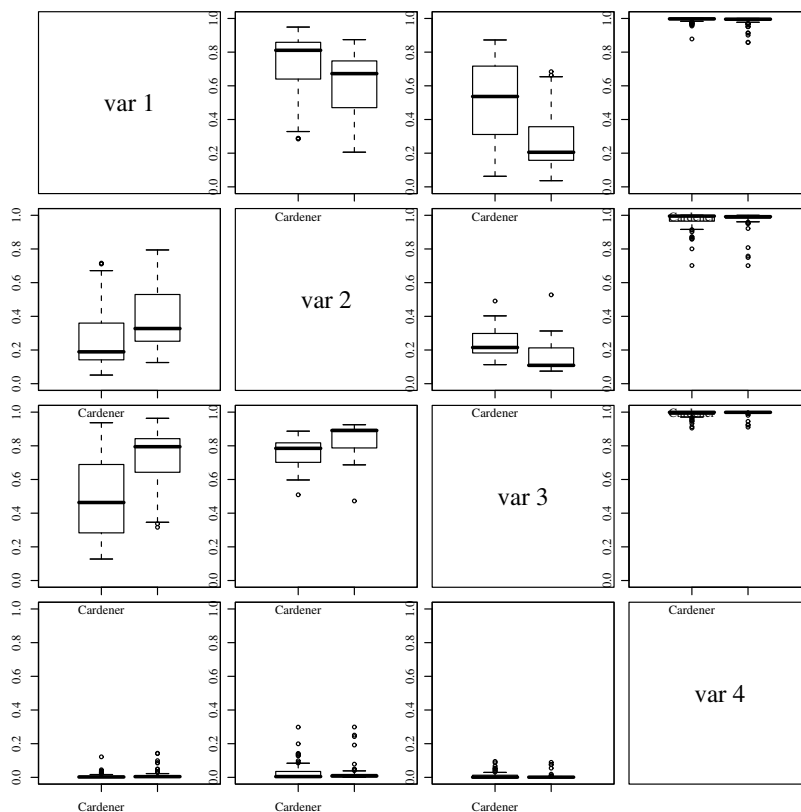


Figure 8: Matrix of box-plots plots in rcomp geometry: Na, Mg, Ca and NH₄ from left to right. This matrix is the array of boxplots of subcompositions formed by row and column parts, and afterwards closed.

When X is declared an rcomp object, it is automatically closed (in this case, to sum up to one, but this can be changed by changing the value of the optional parameter `total`). From the summary, we highlight that only the mean is a composition, and the rest of the statistics are obtained without

attending to the multivariate nature of the set. Regarding Figures 8 and 9, we will only comment cell (1,3). The box-plot shows again that Cardener river will have higher Na values than Llobregat river, fixed a Ca amount. The scatter plot is almost the classic Piper plot (Otero et al.2005), and shows a one-dimensional trend from Na (Cardener samples) to Ca (Llobregat samples).

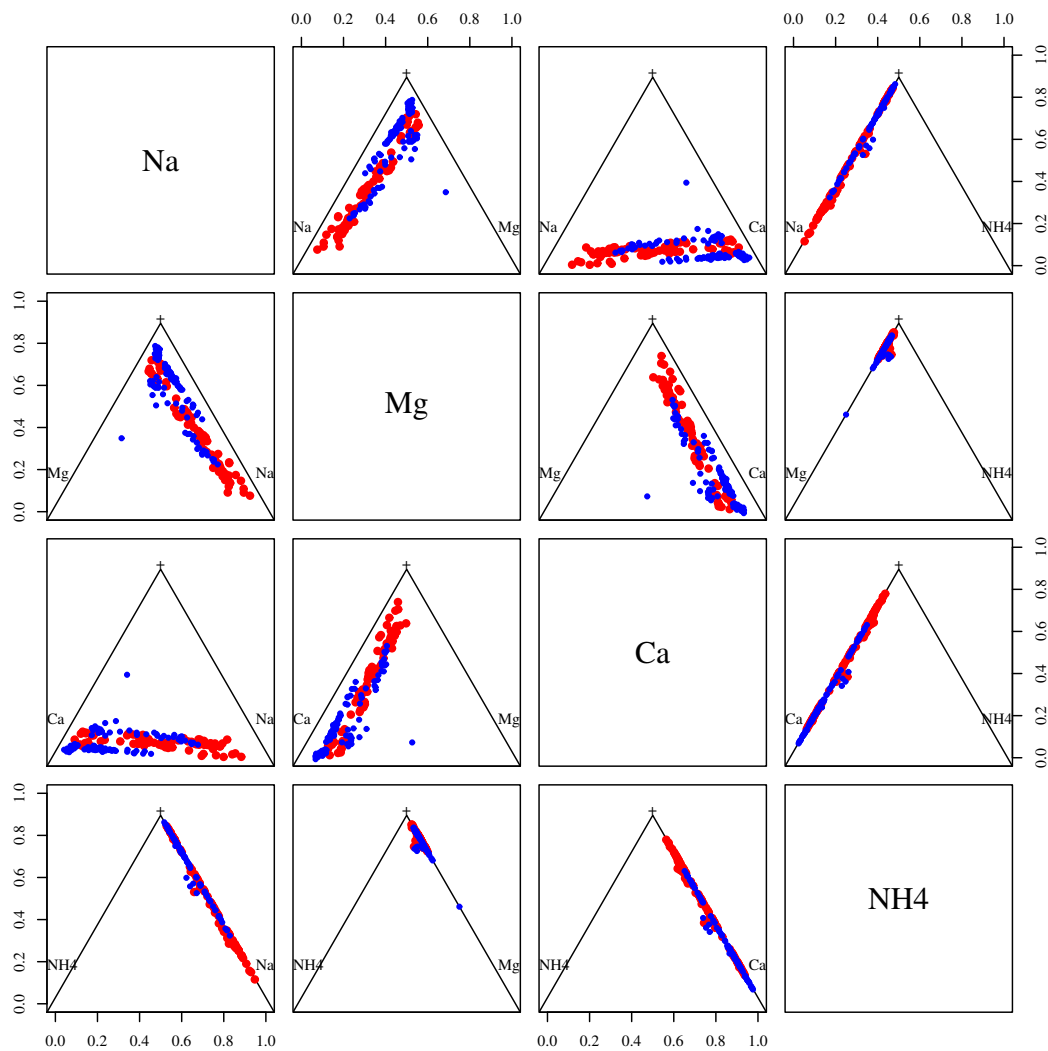


Figure 9: Matrix of scatter plots in rcomp geometry: ternary diagrams defined by row and column parts. Compare with Figure 11. Note that in this case, the third component (marked with the symbol +), is obtained as the amalgamation of the rest of the parts.

acomp geometry

```
> Z=acomp(X)
> summary(Z)
$mean
      Na      Mg      Ca      NH4
0.2929949217 0.1250002452 0.5810983757 0.0009064575
attr(,"class")
[1] "acomp"
```

```
$mean.ratio
      Na      Mg      Ca      NH4
Na  1.000000000 2.343954776 0.504208812 323.2307
Mg  0.426629392 1.000000000 0.215110299 137.8997
Ca  1.983305282 4.648777887 1.000000000 641.0652
NH4 0.003093765 0.007251645 0.001559904  1.0000
```

```
$variation
      Na      Mg      Ca      NH4
Na  0.0000000 0.7525099 1.4774244 2.876780
Mg  0.7525099 0.0000000 0.3135008 2.523983
Ca  1.4774244 0.3135008 0.0000000 2.581879
NH4 2.8767795 2.5239831 2.5818793 0.000000
```

```
$expsd
      Na      Mg      Ca      NH4
Na  1.000000 2.380887 3.371958 5.452680
Mg  2.380887 1.000000 1.750517 4.897402
Ca  3.371958 1.750517 1.000000 4.986941
NH4 5.452680 4.897402 4.986941 1.000000
```

```
$min
      Na      Mg      Ca      NH4
Na  1.000000e+00 2.591311e-01 3.815789e-02 5.978507
Mg  5.397140e-02 1.000000e+00 8.061272e-02 2.344400
Ca  1.463313e-01 8.957044e-01 1.000000e+00 9.552852
NH4 3.942729e-05 9.148764e-05 2.717681e-05 1.000000
```

```
$q1
      Na      Mg      Ca      NH4
Na  1.0000000000 1.293275622 0.192620404 104.2756
Mg  0.2027718675 1.000000000 0.121260922  50.8014
Ca  0.6823203423 2.748430594 1.000000000 363.4892
NH4 0.0009135257 0.002675928 0.000627129  1.0000
```

```
$med
      Na      Mg      Ca      NH4
Na  1.000000000 2.608089261 0.400899834 290.1061
Mg  0.383422460 1.000000000 0.227815385 161.3304
Ca  2.494388659 4.389519179 1.000000000 899.9390
NH4 0.003447015 0.006198459 0.001111186  1.0000
```

```
$q3
      Na      Mg      Ca      NH4
```

```

Na  1.000000000  4.93165171  1.465592213  1094.6599
Mg  0.774761622  1.00000000  0.363844094   373.7026
Ca  5.191561714  8.24668930  1.000000000  1594.5840
NH4 0.009591537  0.01968535  0.002751139    1.0000

```

```

$max
      Na      Mg      Ca      NH4
Na  1.0000000 18.5283311 6.8338088 25363.14
Mg  3.8590501 1.0000000 1.1164397 10930.44
Ca 26.2068966 12.4049904 1.0000000 36796.08
NH4 0.1672658 0.4265485 0.1046808    1.00

```

```

attr("class")
[1] "summary.acomp"
> boxplot(X, fak=riv)
# generates figure 10
> plot(X, pch=18+as.integer(riv), col=c("red","blue")[as.integer(riv)], center=TRUE)
# generates figure 11, with symbols and colors
# controlled by "pch" and "col" optional parameters
# moreover, the optional parameter "center" is
# set to true: compare results with figure 9

```

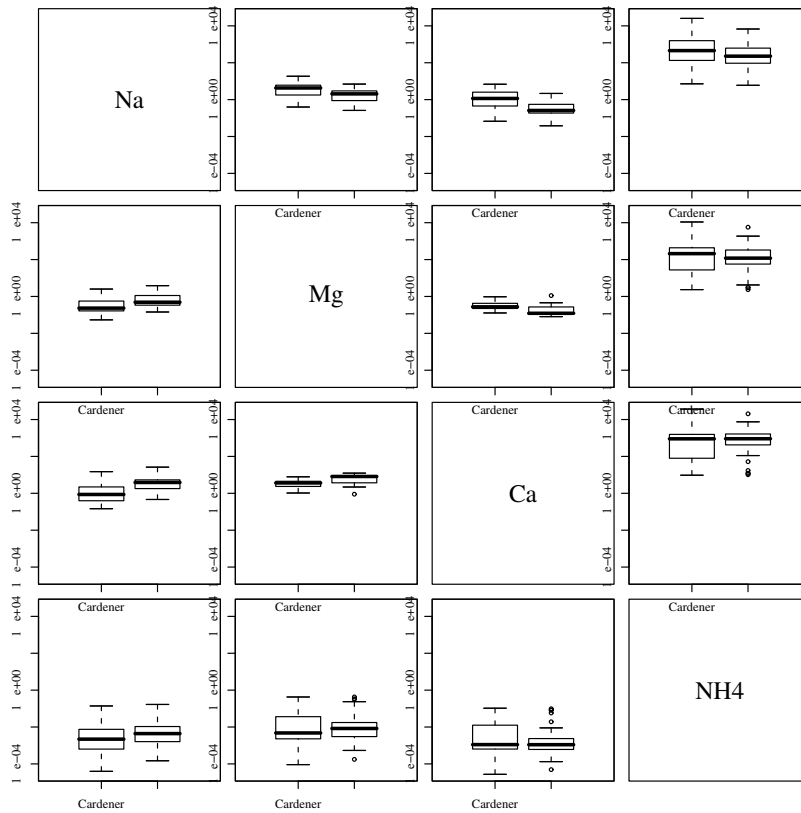


Figure 10: Matrix of box-plots plots in acomp geometry. This matrix is the array of boxplots of log-ratios of row and column parts.

Declaring X an `acomp` object also automatically closes it. From the summary, we highlight that everything is computed on ratios, on all the possible log-ratios among the four parts, Each statistic is then arranged in a matrix of 4×4 elements, and some of them (those defining a composition) are afterwards exponentiated. Figure 9, cell (1,3), deserved a comment: no trend is now visible, due to the centering operation, but it is much clearer that Na-richer samples come from the Cardener river, whereas Ca is higher in Llobregat samples.

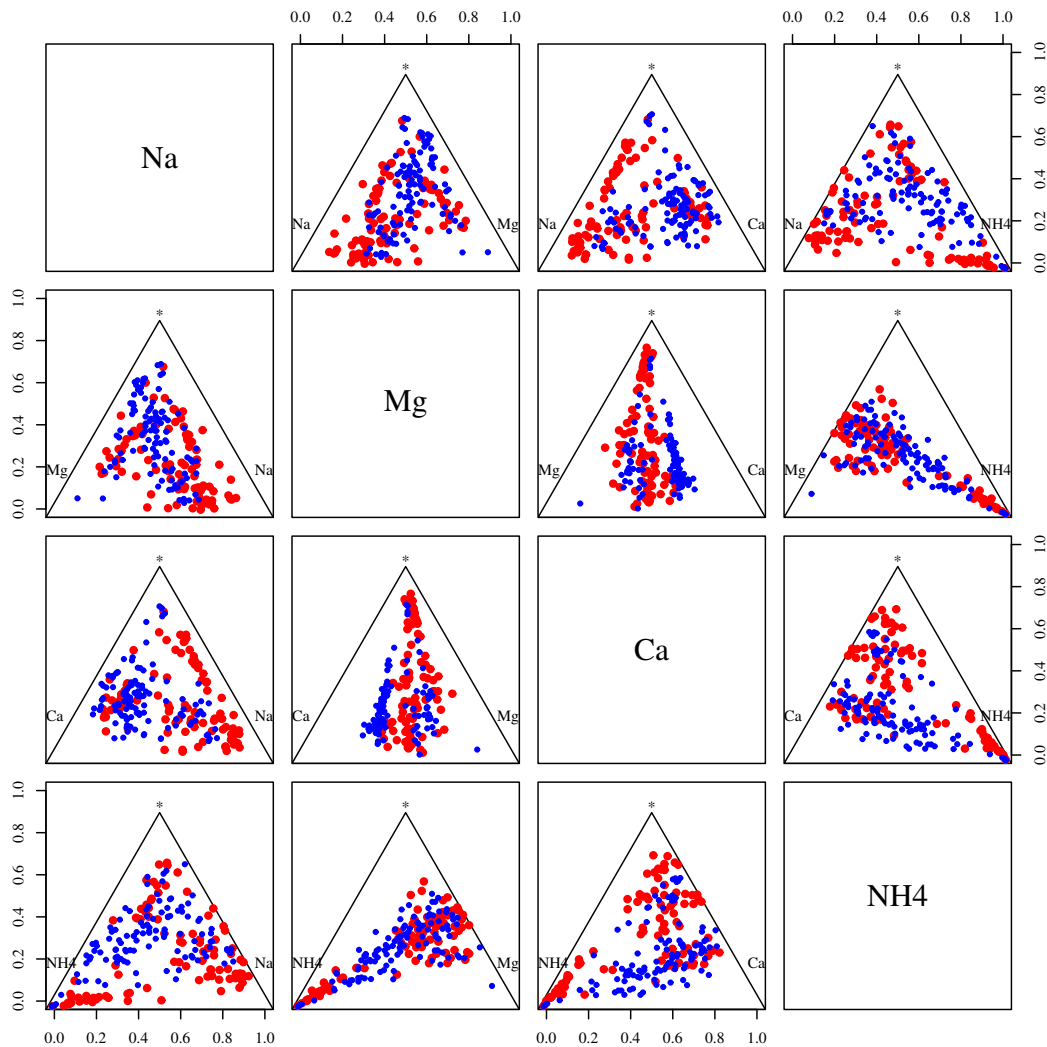


Figure 11: Matrix of scatter plots in `acomp` geometry: centered ternary diagrams defined by row and column parts. Compare with Figure 9. Note that in this case, the third component (marked with the symbol *), is obtained as the geometric mean of the rest of the parts.

Discrimination analysis

This section presents how do the results of a discrimination analysis change when the class of the data set is changed. This is not a complete discrimination analysis, but only an example of how "compositions" work.

rplus geometry

```
> X=rplus(X)
> disc=lda(formula=riv~Y,data=list(Y=idt(X),riv=riv))
> disc
Call:
lda(riv ~ Y, data = list(Y = idt(X), riv = riv))
```

```
Prior probabilities of groups:
      Cardener UpperLLobregat
0.4589372      0.5410628
```

```
Group means:
      YNa      YMg      YCa      YNH4
Cardener 166.48540 31.4264 96.49874 1.0462506
UpperLLobregat 59.05736 21.4314 95.14179 0.4661137
```

```
Coefficients of linear discriminants:
      LD1
YNa -0.007142635
YMg -0.028861797
YCa  0.034567086
YNH4 -0.099519853
```

Note that with this geometry, the group means and the coefficients of the discriminant function can be interpreted directly. In particular, the function is

$$d_{rplus} = (-7.1 \cdot Na - 28.8 \cdot Mg + 34.6 \cdot Ca - 99.5 \cdot NH_4) \cdot 10^{-3}.$$

aplus geometry

```
> X=aplus(X)
> disc=lda(formula=riv~Y,data=list(Y=idt(X),riv=riv))
> disc
Call: lda(riv ~ Y, data = list(Y = idt(X), riv = riv))
```

```
Prior probabilities of groups:
      Cardener UpperLLobregat
0.4589372      0.5410628
```

```
Group means:
      YNa      YMg      YCa      YNH4
Cardener  4.487714 3.318761 4.555098 -1.677987
UpperLLobregat 3.293184 2.710325 4.501621 -2.156640
```

```
Coefficients of linear discriminants:
      LD1
```

```

YNa -0.64924278
YMg -1.52246623
YCa 4.84464820
YNH4 0.06084054
> discmean=ilt.inv(disc$means)
> colnames(discmean)=colnames(X)
> discmean
      Na      Mg      Ca      NH4
Cardener 88.91797 27.62611 95.11605 0.1867496
UpperLlobregat 26.92847 15.03416 90.16313 0.1157133
attr(,"class")
[1] "aplus"

```

With an *aplus* geometry, the group means are returned in a log scale, and they had to be back-transformed with the function `ilt.inv` (inverse of the isometric logarithmic transform, i.e. the exponential of each component). This `discmean` should be compared with the **Group means** of the last subsection. The coefficients of the discriminant function must be taken as the coefficients of a log-linear function

$$d_{aplus} = -0.65 \cdot \ln Na - 1.5 \cdot \ln Mg + 4.8 \cdot \ln Ca - 0.06 \cdot \ln NH_4 = 0.06 \cdot \ln \frac{Ca^{80}}{Na^{11} \cdot Mg^{25} \cdot NH_4}$$

rcomp geometry

```

> X=rcomp(X)
> disc=lda(formula=riv~Y,data=list(Y=idt(X),riv=riv))
> disc
Call: lda(riv ~ Y, data = list(Y = idt(X), riv = riv))

```

Prior probabilities of groups:

```

      Cardener UpperLlobregat
0.4589372      0.5410628

```

Group means:

```

      Y1      Y2      Y3
Cardener 0.22443407 -0.08503305 0.3062623
upperLlobregat -0.02185504 -0.17628175 0.4605713

```

Coefficients of linear discriminants:

```

      LD1
Y1 -3.388166
Y2 -6.450755
Y3 -1.597351
> discmean=ipt.inv(disc$means)
> colnames(discmean)=colnames(X)
> discmean
      Na      Mg      Ca      NH4
Cardener 0.4443656 0.1157823 0.4364862 0.003365934
UpperLlobregat 0.2310730 0.1123756 0.6539488 0.002602623
attr(,"class")
[1] "rcomp"
> discload= ilrBase(D=4) %*% disc$scaling
> rownames(discload)=colnames(X)

```

```

> discload
      LD1
Na  -2.934238
Mg  -4.288940
Ca   2.482092
NH4  4.741087

```

With an rcomp geometry, the group means are returned in the rotated coordinate system of the `ipt` (isometric planar transform). Thus they have to be back-transformed with the function `ilr.inv`. This `discmean` cannot be directly compared with the preceding means, because this is closed. The same happens with the coefficients of the discriminant function: they are given in `ipt` coordinates. This coordinate system has as advantage that the covariance matrix (an information used by the discrimination analysis) is not singular, but as hindrance that it has no one-to-one relation with the original parts. In contrast, `cpt` values keep this one-to-one relation. The transformation between the two systems is obtained with the line `ilrBase(D=4) %*% disc$scaling`. Once transformed the `ipt` discriminant vector to `cpt`, we can build the discrimination function as

$$d_{rcomp} = -2.9 \cdot Na - 4.3 \cdot Mg + 2.5 \cdot Ca + 4.7 \cdot NH_4.$$

acomp geometry

```

> X=acomp(X)
> disc=lda(formula=riv~Y,data=list(Y=idt(X),riv=riv))
> disc
Call: lda(riv ~ Y, data = list(Y = idt(X), riv = riv))

```

Prior probabilities of groups:

Cardener	UpperLlobregat
0.4589372	0.5410628

Group means:

	Y1	Y2	Y3
Cardener	2.097880	1.535182	4.407456
UpperLlobregat	1.392640	1.255636	4.708101

Coefficients of linear discriminants:

```

      LD1
Y1 -0.4699841
Y2 -1.7903313
Y3  1.2217548
> discmean=ilr.inv(disc$means)
> colnames(discmean)=colnames(X)
> discmean
      Na      Mg      Ca      NH4
Cardener  0.4197275 0.1304060 0.4489849 0.0008815311
UpperLlobregat 0.2036311 0.1136871 0.6818068 0.0008750152
attr(,"class")
[1] "acomp"
> discload= ilrBase(D=4) %*% disc$scaling
> rownames(discload)=colnames(X)
> discload
      LD1
Na  -0.407018148
Mg  -1.326126632

```

Ca 1.730483475
NH4 0.002661305

With an `acomp` geometry, the group means are returned as isometric log-ratio coordinates, and the mean compositions can be recovered by the back-transformation (`ilr.inv`). Compare the `discmean` obtained with both geometries `acomp` and `rcomp`. Regarding the discriminant function, the same transformation from `ilr` to `clr` coordinate system should be done, so that we obtain a coefficient linked with each part. These coefficients are again involved in a log-linear discriminant function

$$d_{acomp} = -0.4 \cdot \ln Na - 1.3 \cdot \ln Mg + 1.7 \cdot \ln Ca + 2.6 \cdot 10^{-3} \cdot \ln NH_4 \simeq 0.4 \cdot \ln \frac{Ca^4}{Na \cdot Mg^3}.$$

In the `clr` coefficients, the NH_4 part was seen to be irrelevant, which allowed to get a simplified log-ratio expression. Note that these `clr` coefficients can be back-transformed to a composition, representing a perturbation between the two groups.