Practical course using the R software

Multivariate analysis of genetic data: exploring group diversity

Thibaut Jombart

Abstract

This practical course tackles the question of group diversity in genetic data analysis using R [8]. It consists of two main parts: first, how to infer groups when these are unknown, and second, how to use group information when describing the genetic diversity. The practical uses mostly the packages *adegenet* [5], *ape* [7] and *ade4* [1, 3, 2], but others like *genetics* [9] and *hierfstat* [4] are also required.

Contents

```
> library(adegenet)
```

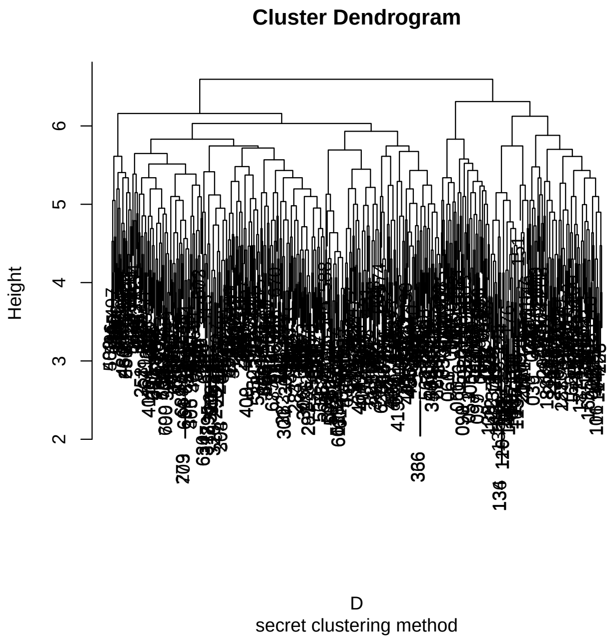
1 Defining genetic clusters

Group information is not always known when analysing genetic data. Even when some prior clustering can be defined, it is not always obvious that these are the best genetic clusters that can be defined. In this section, we illustrate two simple approaches for defining genetic clusters.

1.1 Hierarchical clustering

Hierarchical clustering can be used to represent genetic distances as trees, and indirectly to define genetic clusters. This is achieved by cutting the tree at a certain distance, and pooling the tips descending from the few retained branches into the same clusters (cutree). Load the data microbov, replace the missing data, and compute the Euclidean distances between individuals (functions na.replace and dist). Then, use hclust to obtain a hierarchical clustering of the individual which forms strong groups (choose the right method(s)).

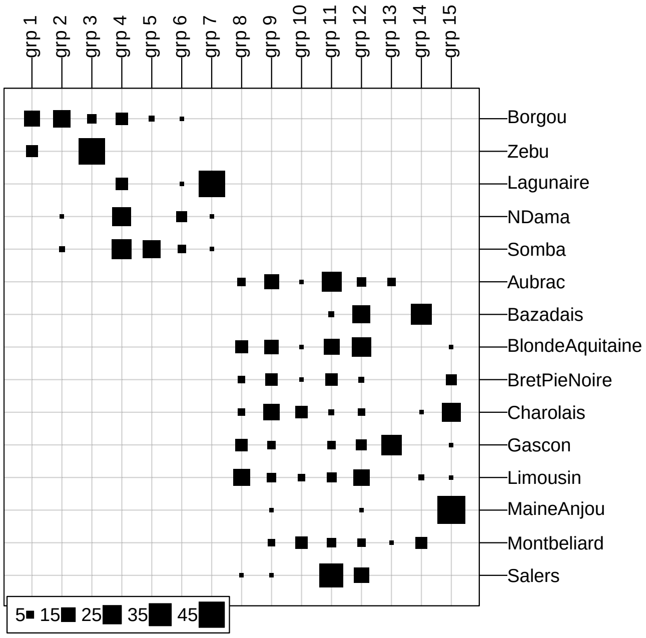
```
> data(microbov)
> X <- na.replace(microbov, method = "mean")$tab

 Replaced 6325 missing values


> D <- dist(X)
> h1 <- hclust(D, method = "complete")


> plot(h1, sub = "secret clustering method")
```



Cut the tree into two groups. Do these groups match any prior clustering of the data (see content of `microbov$other`)? Remember that `table` can be used to build contingency tables. Accordingly, what is the main component of the genetic variability in these cattle breeds?

Repeat this analysis by cutting the tree into as many clusters as there are breeds in the dataset (this can be extracted by `pop`), and name the result `grp`. Build a contingency table to see the match between inferred groups and breeds. Use `table.value` to represent the result, then interprete it:

```
> table.value(table(pop(microbov), grp), col.lab = paste("grp",
+     1:15))
```



Can some groups be identified to species? to breeds? Are some species more admixed than others?

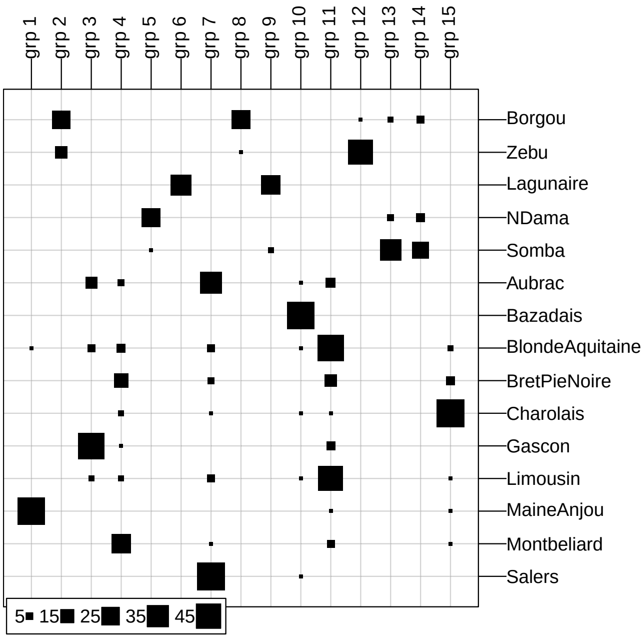
1.2 K-means

K-means is another, non-hierarchical approach for defining genetic clusters. It relies on the usual ANOVA model for multivariate data $\mathbf{X} \in \mathbb{R}^{n \times p}$:

$$\mathrm{VAR}(\mathbf{X}) = \mathrm{B}(\mathbf{X}) + \mathrm{W}(\mathbf{X})$$

where VAR, B and W correspond to, respectively, the total variance, the variance between groups, and the variance within groups. K-means then consists in finding groups that will minimize $\mathrm{W}(\mathbf{X})$. Using the function kmeans, find 15 genetic clusters in the microbov data; how do the inferred clusters compare to the breeds? Reproduce the same figure as before for these results; the figure should ressemble:

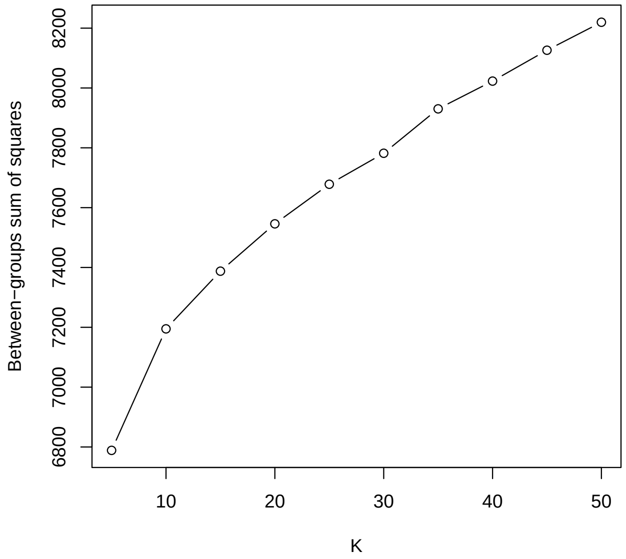
```
> grp <- kmeans(X, 15)
```



What differences can you see? What are the species which are easily genetically identified using K-means?

Look at the output of K-means clustering, and try to identify the sums of squares corresponding to the variance partition model. How do these behave when increasing the number of clusters? Using sapply, retrieve the results of several K-means where K is increased gradually. Plot the results; one example of obtained result would be:

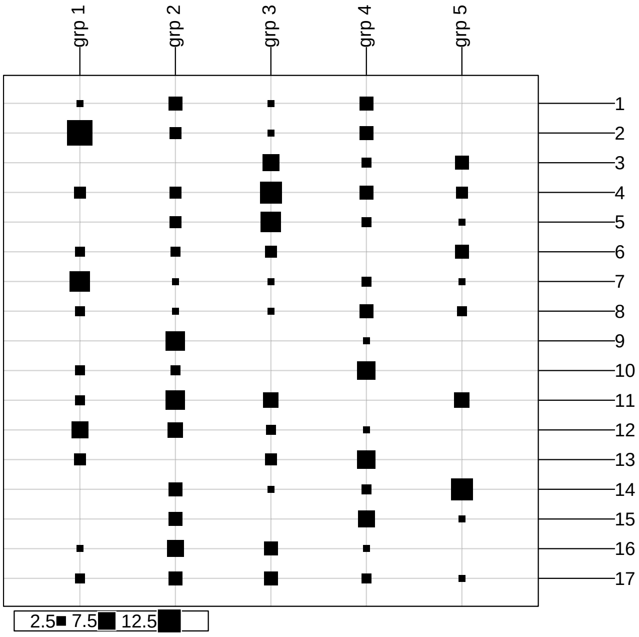
```
> totalSS <- sum(X * X)
> temp <- sapply(seq(5, 50, by = 5), function(k) totalSS - sum(kmeans(X,
+     k)$withinss))

> plot(seq(5, 50, by = 5), temp, ylab = "Between-groups sum of squares",
+     type = "b", xlab = "K")
```



How can you interprete this observation? Can B($\mathbf{X}$) or W($\mathbf{X}$) be used for selecting the best K?

Deciding of the number of optimal clusters (K) to be retained is often tricky. This can be achieved by computing K-means solution for a range of K, and then selecting the K giving the best Bayesian Information Criterion (BIC). This is achieved by the function `find.clusters`. In addition, this function orthogonalizes and reduces the data using PCA as a prior step to K-means. Use this function to search for the optimal number of clusters in `microbov`. How many clusters would you retain? Compare them to the breed information: when 8 clusters are retained, what are the most admixed breeds?

Repeat the same analyses for the `nancycats` data. What can you say about the likely profile of admixture between these cat colonies?



2 Describing genetic clusters

There are different ways of using genetic cluster information in the multivariate analysis of genetic data. The methods vary according to the purpose of the study: describing the differences between groups, describing how individuals are structured in the different clusters, etc.

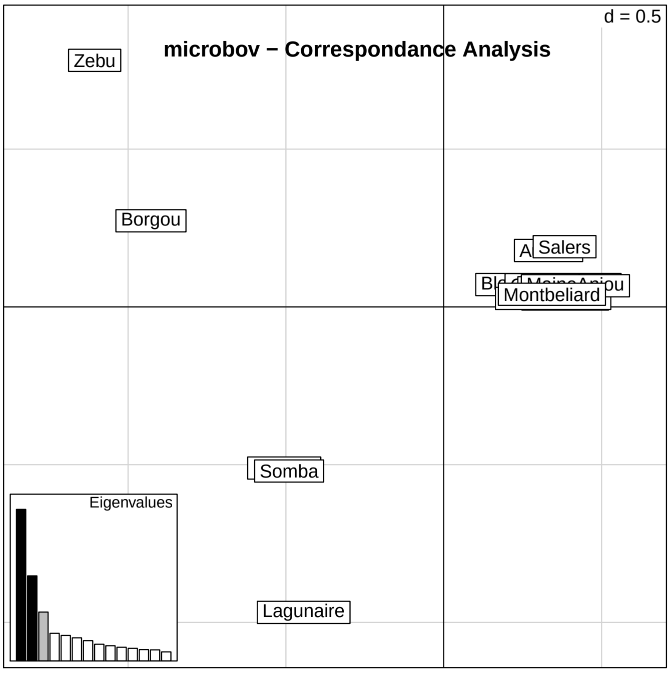
2.1 Analysis of group data

Sometimes we are just interested in the diversity between groups, and not interested by variations occuring within clusters. In such cases, we can analyse data directly at a population level. The first step doing so is computing allele counts by populations (using `genind2genpop`). These data can then be:

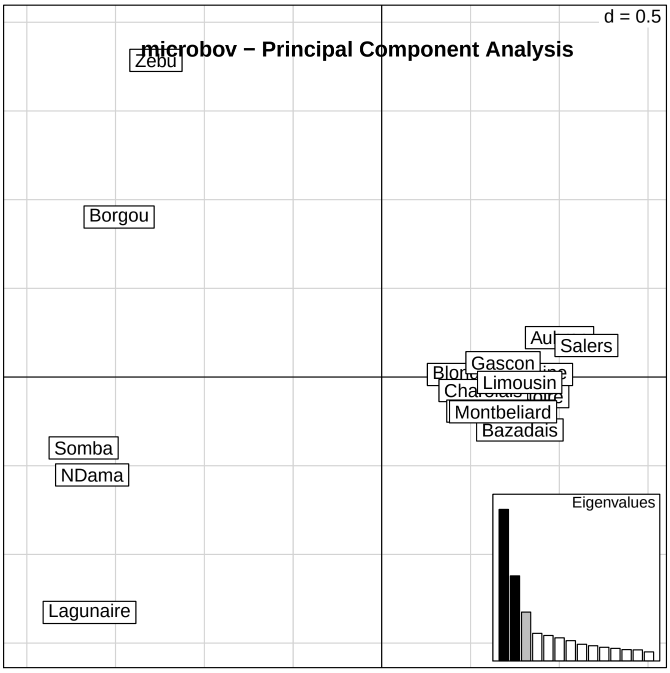
- ⋆ directly analysed using Correspondance Analysis (CA, `dudi.coa`), which is appropriate for contingency tables
- ⋆ translated into scaled allele frequencies (`makefreq` or `scaleGen`), and analysed by PCA (PCA, `dudi.pca`)
- ⋆ used to compute genetic distances between populations (`dist.genpop`) which are in turn summarised by Principal Coordinates Analysis (PCoA, `dudi.pco`)

Try applying these approaches to the dataset `microbov`, and compare the obtained results.

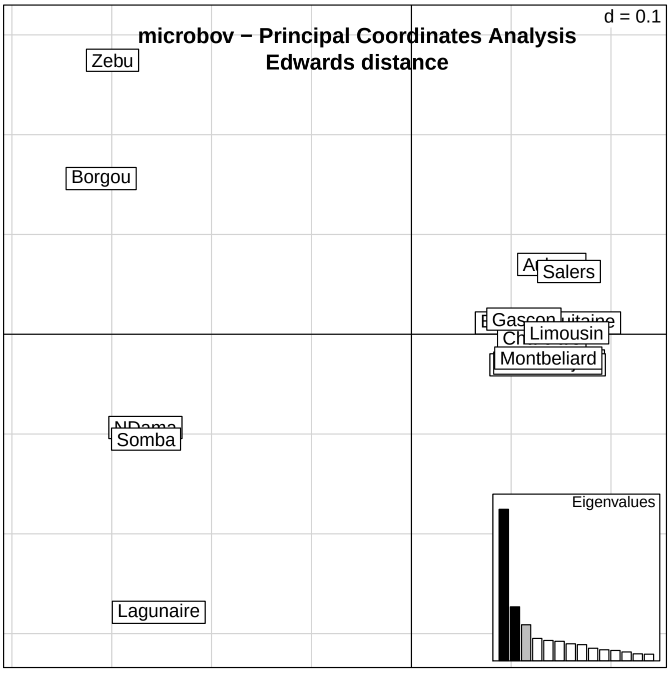
```
> x <- genind2genpop(microbov)

 Converting data from a genind to a genpop object...

...done.

> ca1 <- dudi.coa(truenames(x), scannf = FALSE, nf = 3)
> s.label(ca1$li)
> add.scatter.eig(ca1$eig, 3, 2, 1)
> title("microbov - Correspondance Analysis")
```



```
> pca1 <- dudi.pca(scaleGen(x, scale = FALSE), scale = FALSE, scannf = FALSE,
+     nf = 3)
> s.label(pca1$li)
> add.scatter.eig(ca1$eig, 3, 2, 1, posi = "bottomright")
> title("microbov - Principal Component Analysis")
```

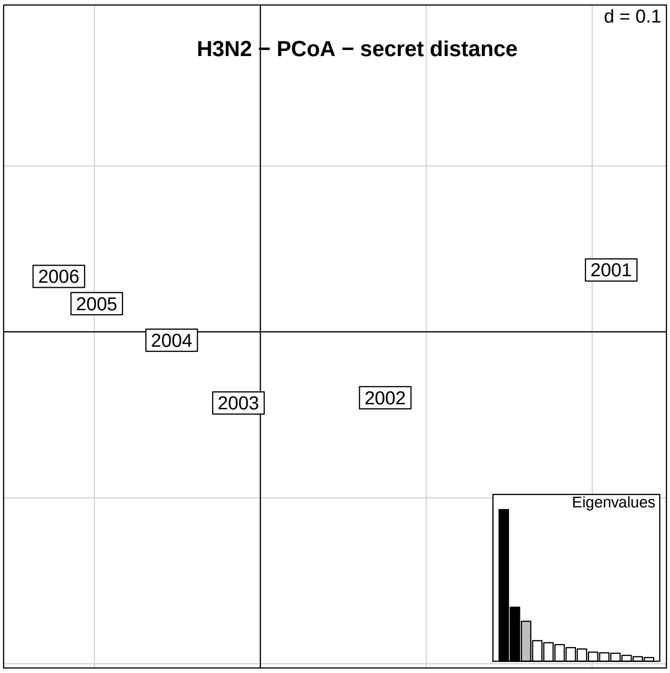


```
> pco1 <- dudi.pco(dist.genpop(x, meth = 2), nf = 3, scannf = FALSE)
> s.label(pco1$li)
> add.scatter.eig(pco1$eig, 3, 2, 1, posi = "bottomright")
> title("microbov - Principal Coordinates Analysis\nEdwards distance")
```



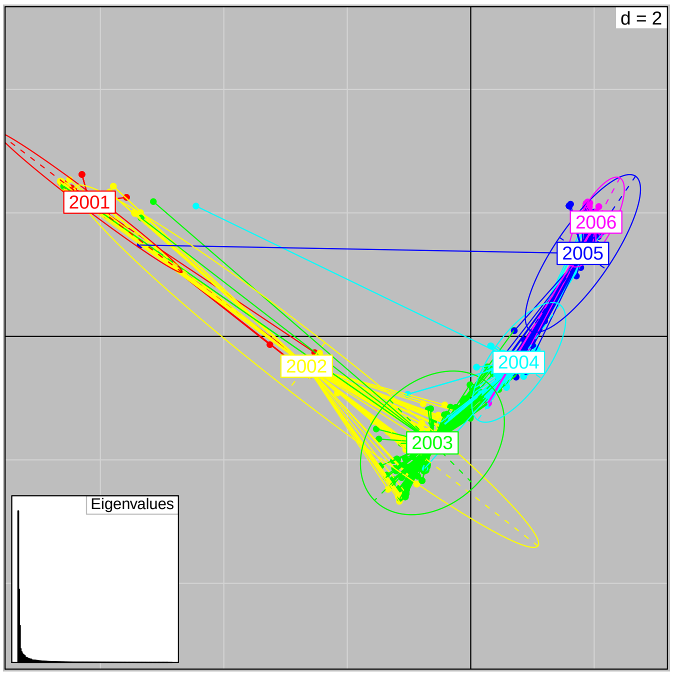
For the PCoA, try different Euclidean distances (see dist.genpop), and compare the results. Does the genetic distance employed seem to matter?

We will leave the cattle at a rest for now, and switch to Human seasonal influenza, H3N2. The dataset H3N2 contains SNPs derived from 1903 hemagglutinin RNA sequences. Obtain allele counts per year of sampling (see content of H3N2$other) using genind2genpop, and compute *i*) Edward's distance *ii*) Reynolds' distance and *iii*) Roger's distance between years. Perform the PCoA of these distances and plot the results; here is a result for one of the requested distances:



What is the meaning of the first principal components? Are the results coherent between the three distances? To have yet another point of view, perform the PCA of the individual data, and represent group information when plotting the principal components using s.class.

```
> pca1 <- dudi.pca(scaleGen(H3N2, miss = "mean", scale = FALSE),
+     scale = FALSE, scannf = FALSE, nf = 2)
> par(bg = "grey")
> s.class(pca1$li, factor(f), col = rainbow(6))
> add.scatter.eig(pca1$eig, 2, 1, 2)
```



What can we say about the genetic evolution of influenza between years 2001-2006? Can we safely discard information about individual isolates, and just work with data pooled by year of epidemic?

2.2 Between-class analyses

In many situations, discarding information about the individual observations leads to a considerable loss of information. However, basic analyses working at an individual level –mainly PCA– are not optimal in terms of group separation. This is due to the fact that PCA focuses on the total variability, while only variability between groups should be optimized.

Let us remember the basic multivariate ANOVA model ($\mathbf{P}$ being the projector onto dummy vectors of group membership $\mathbf{H}$: $\mathbf{P} = \mathbf{H}(\mathbf{H}^T\mathbf{D}\mathbf{H})^{-1}\mathbf{H}^T\mathbf{D}$):

$$\mathbf{X} = \mathbf{P}\mathbf{X} + (\mathbf{I} - \mathbf{P})\mathbf{X}$$

which leads to the variance partition seen before with the K-means:

$$\text{VAR}(\mathbf{X}) = \text{B}(\mathbf{X}) + \text{W}(\mathbf{X})$$

with:

- $\star$ $\text{VAR}(\mathbf{X}) = \text{trace}(\mathbf{X}^T\mathbf{D}\mathbf{X})$
- $\star$ $\text{B}(\mathbf{X}) = \text{trace}(\mathbf{X}^T\mathbf{P}^T\mathbf{D}\mathbf{P}\mathbf{X})$

- ⋆ $\mathrm{W}(\mathbf{X}) = \mathrm{trace}(\mathbf{X}^T(\mathbf{I}-\mathbf{P})^T\mathbf{D}(\mathbf{I}-\mathbf{P})\mathbf{X})$

where $\mathbf{D}$ is a metric (in PCA, it can be replaced by $1/n$) and $\mathbf{I}$ is the identity matrix.

We can verify that ordinary PCA decomposes the total variance:

```
> X <- scaleGen(H3N2, miss = "mean", scale = FALSE)
> pca1 <- dudi.pca(X, scale = FALSE, scannf = FALSE, nf = 2)
> sum(pca1$eig)

[1] 15.05856

> D <- diag(1/nrow(X), nrow(X))
> sum(diag(t(X) %*% D %*% X))

[1] 15.05856
```

Interestingly, it is possible to modify a multivariate analysis so as to decompose $\mathrm{B}(\mathbf{X})$ or $\mathrm{W}(\mathbf{X})$ instead of $\mathrm{VAR}(\mathbf{X})$. Corresponding methods are respectively called *between-class* (or *inter-class*) and *within-class* (or *intra-class*) analyses. If the basic method is a PCA, then we will be performing between-class PCA and within-class PCA. These methods are implemented by the functions `between` and `within`. Here is an example of how to perform the between-class PCA:

```
> f <- H3N2$other$epid
> bpca1 <- between(pca1, fac = factor(f), scannf = FALSE, nf = 3)
```

Verify that the sum of eigenvalues of this analysis equates the variance between groups ($\mathrm{B}(\mathbf{X})$); for this, you will need to compute $\mathbf{H}$, and then $\mathbf{P}$. This requires a few matrix operations:

- ⋆ `A %*% B`: multiplies matrix **A** by matrix **B**
- ⋆ `diag`: either creates a diagonal matrix, or extract the diagonal of an existing square matrix
- ⋆ `t`: transposes a matrix
- ⋆ `ginv`: inverses a symmetric matrix

To obtain $\mathbf{H}$, the matrix of dummy vectors of group memberships, do:

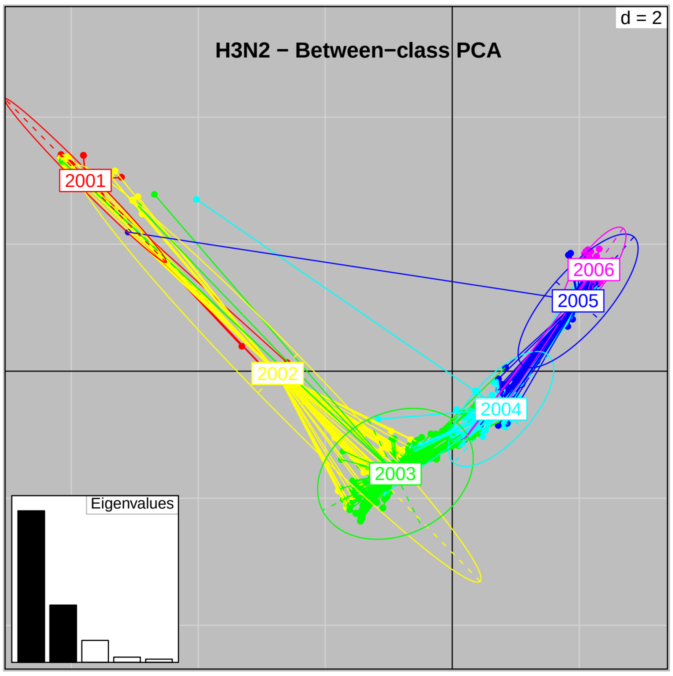
```
> H <- as.matrix(acm.disjonctif(data.frame(f)))
```

You should find that $\mathrm{B}(\mathbf{X})$ is exactly equal to the sum of the eigenvalues of the between-class analysis.

Repeat the same approach for the within-class analysis, and confirm that the variance within groups is fully decomposed by the within-class PCA.

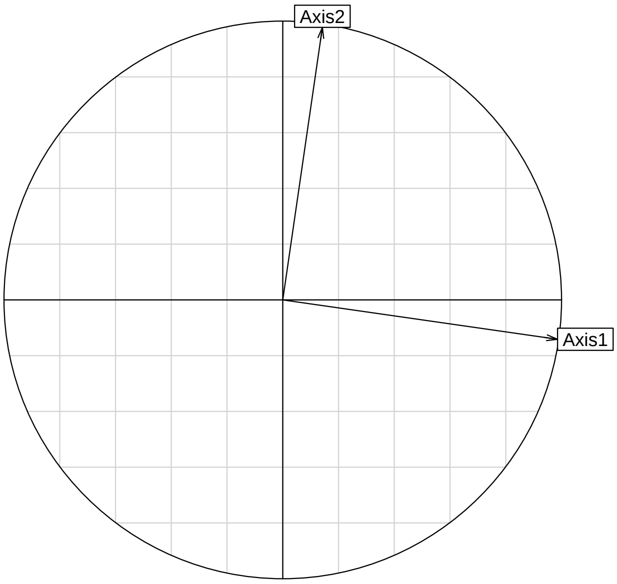
Now that you have checked that we can associate multivariate methods to each component of the variance partitioning into groups, investigate further the results of the between-class PCA.

```
> par(bg = "grey")
> s.class(bpca1$ls, factor(f), col = rainbow(6))
> add.scatter.eig(bpca1$eig, 2, 1, 2)
> title("H3N2 - Between-class PCA")
```



On this scatterplot, the centres of the groups (labels of year) are as scattered as possible. Why is it slightly disappointing? How does it compare to the original PCA? The following figure represents the PCA axes onto the basis of the between-class analysis.

```
> s.corcircle(bpca1$as)
```



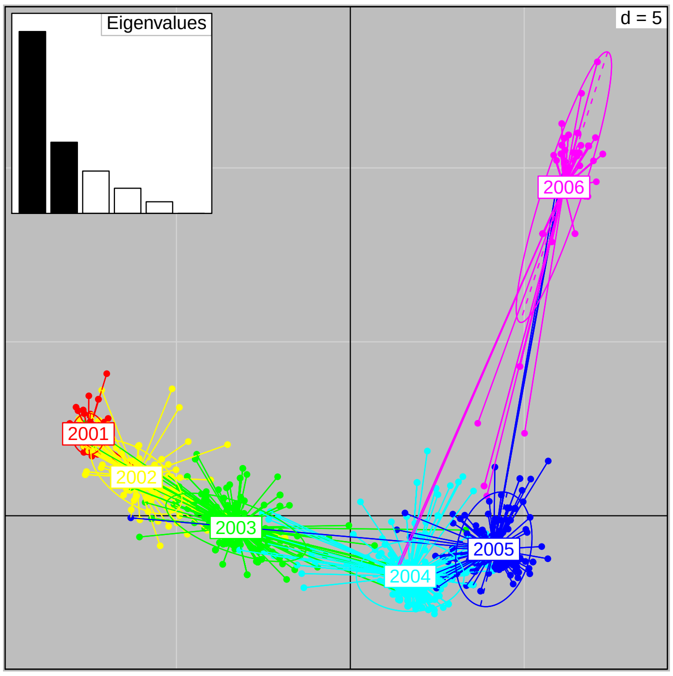
What can we say about the PCA and between-class PCA? One obvious problem of the previous scatterplot is the dispersion of the isolates within 2001 and 2002. To assess best the separation of the isolates by year, we need to maximize dispersion between groups, and to minimize dispersion within groups. This is precisely what Discriminant Analysis does.

2.3 Discriminant Analysis of Principal Components

Discriminant Analysis aims at displaying the best discrimination of individuals into pre-defined groups. However, some technical requirements make it often impractical for genetic data. Discriminant Analysis of Principal Components (DAPC, [6]) has been developped to circumvent this issue. Apply DAPC (function `dapc`) to the H3N2 data, retaining 50 PCs in the preliminary PCA step, and plot the results using `scatter`. The result should ressemble this:

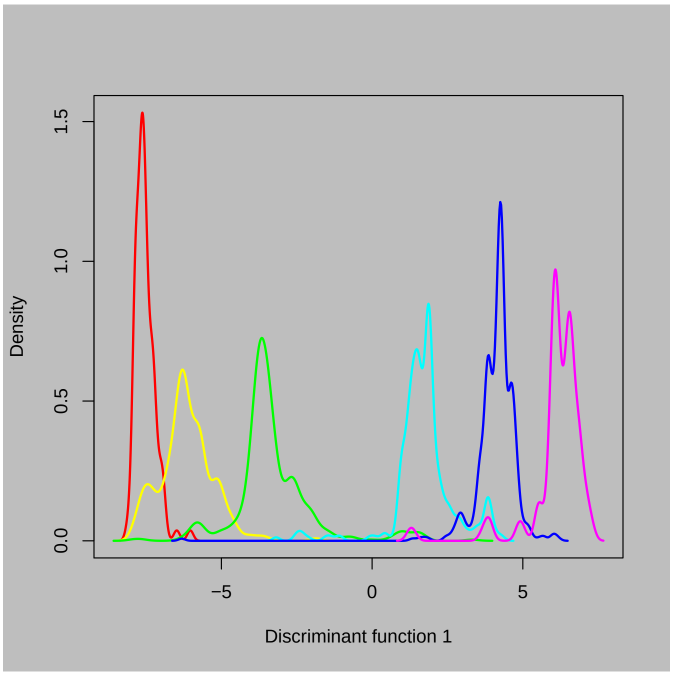
```
> dapc1 <- dapc(H3N2, pop = f, n.pca = 50, n.da = 2, all.contr = TRUE)
```

```
> scatter(dapc1, posi = "top")
```



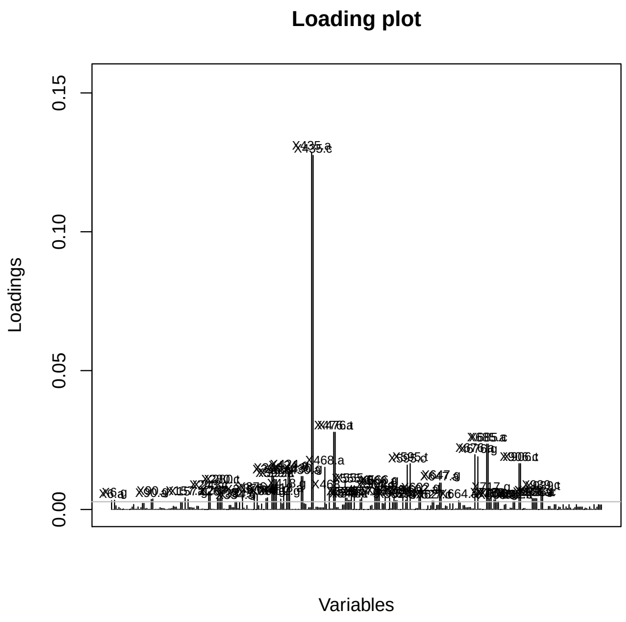
How does this compare to the previous results? What new feature appeared, that was not visible before? To have a better assessment of the structure on each axis, you can plot one dimension at a time, specifying the same axis for `xax` and `yax` in `scatter.dapc`; for instance:

```
> scatter(dapc1, xax = 1, yax = 1)
```



To interprete results further, you will need the loadings of alleles, which are not provided by default in dapc. Re-run the analysis if needed, and specify you want allele contributions to be returned. Use loadingplot to display allele contributions to the first and to the second structure. This is the result you should obtain for the first axis:

```
> loadingplot(dapc1$var.contr)
```



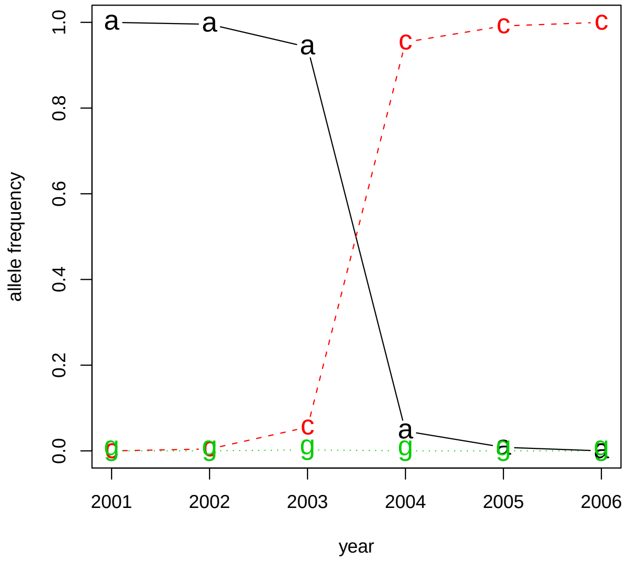
How can you interprete this result? Note that loadingplot returns invisibly some information about the most contributing alleles (thoses indicated on the plot). Save this information, and then examine it. Interprete the following commands and their result:

```
> which(H3N2$loc.names == "435")

L051
  51

> snp435 <- truenames(H3N2[loc = "L051"])
> head(snp435)

         435.a 435.c 435.g
AB434107     1     0     0
AB434108     1     0     0
AB438242     1     0     0
AB438243     1     0     0
AB438244     1     0     0
AB438245     1     0     0
```

```
> temp <- apply(snp435, 2, function(e) tapply(e, f, mean, na.rm = TRUE))
> temp
```

```
           435.a       435.c       435.g
2001 1.000000000 0.000000000 0.000000000
2002 0.995535714 0.004464286 0.000000000
2003 0.942396313 0.055299539 0.002304147
2004 0.046210721 0.953789279 0.000000000
2005 0.008316008 0.991683992 0.000000000
2006 0.000000000 1.000000000 0.000000000
```

```
> matplot(temp, type = "b", pch = c("a", "c", "g"), xlab = "year",
+     ylab = "allele frequency", xaxt = "n", cex = 1.5)
> axis(side = 1, at = 1:6, lab = 2001:2006)
```



Do the same for the second axis. How would you interprete this result? Should we expect a vaccine from 2005 influenza to have worked against the 2006 virus?

References

[1] D. Chessel, A-B. Dufour, and J. Thioulouse. The ade4 package-I- one-table methods. *R News*, 4:5–10, 2004.

[2] S. Dray and A.-B. Dufour. The ade4 package: implementing the duality diagram for ecologists. *Journal of Statistical Software*, 22(4):1–20, 2007.

[3] S. Dray, A.-B. Dufour, and D. Chessel. The ade4 package - II: Two-table and K-table methods. *R News*, 7:47–54, 2007.

[4] J. Goudet. HIERFSTAT, a package for R to compute and test hierarchical F-statistics. *Molecular Ecology Notes*, 5:184–186, 2005.

[5] T. Jombart. adegenet: a R package for the multivariate analysis of genetic markers. *Bioinformatics*, 24:1403–1405, 2008.

[6] T Jombart, S Devillard, and F Balloux. Discriminant analysis of principal components: a new method for the analysis of genetically structured populations. *Genetics*, submitted.

[7] E. Paradis, J. Claude, and K. Strimmer. APE: analyses of phylogenetics and evolution in R language. *Bioinformatics*, 20:289–290, 2004.

[8] R Development Core Team. *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, Vienna, Austria, 2009. ISBN 3-900051-07-0.

[9] G. R. Warnes. The genetics package. *R News*, 3(1):9–13, 2003.