Genetic clustering and hybrid detection using *adegenet*

Thibaut Jombart [*]

Imperial College London

MRC Centre for Outbreak Analysis and Modelling

November 7, 2016

Abstract

This tutorial presents an overview of likelihood-based genetic clustering in *adegenet*, as implemented by the function `genclust.em`. After a brief presentation of the rationale of the method, we illustrate its use in two situations: for identifying genetic clusters, and then for detecting hybrids.

[*] `thibautjombart@gmail.com`

Contents

1 The method, in a nutshell

1.1 Model formulation

1.1.1 General likelihood

Likelihood-based genetic clustering in *adegenet* is implemented by the function `genclust.em`. This approach uses Hardy-Weinberg's equilibrium to define the probabilities of observing given genotypes under known allele frequencies. Let $x_{i,j}$ be the distribution of allele counts at a given locus j for individual i. For now, we assume the group to which i belongs is known (noted g_i), and has known allele frequencies $f_{j,g}$. For any level of ploidy π and any codominant marker, the likelihood of $x_{i,j}$ is defined as:

$$p(x_{i,j}|f_{j,g_i}, \pi) = \mathcal{M}(x_{i,j}, f_{j,g_i}, \pi) \tag{1}$$

where $\mathcal{M}$ is refers to the multinomial probability mass function.

Assuming independence between loci ($j = 1, \dots, J$), the likelihood of individual i across all loci is:

$$p(x_i|f_{1,g_i}, \dots, f_{J,g_i}, \pi) = \prod_j p(x_{i,j}|f_{j,g_i}, \pi) \tag{2}$$

Assuming further independence between individuals, conditional on their group membership $g = g_1, \dots, g_I$ and all allele frequencies in all groups f the total likelihood is defined as:

$$p(x|f, g, \pi) = \prod_i p(x_i|f_{1,g_i}, \dots, f_{J,g_i}, \pi) \tag{3}$$

1.1.2 Useful particular cases

As particular case, the likelihood of all possible diploid genotypes with alleles A and B is defined, noting f_{g_i} the vector of allele frequencies in a given group g_i as:

$$p(AA|f_{g_i}, 2) = f_A^2 \tag{4}$$

$$p(B|f_{g_i}, 2) = f_B^2 \tag{5}$$

$$p(AB|f_{g_i}, 2) = 2f_A f_B \tag{6}$$

Note that for haploid data, this is even simpler:

$$p(A|f_{g_i}, 1) = f_A \tag{7}$$

$$p(B|f_{g_i}, 1) = f_B \tag{8}$$

1.1.3 Group membership probabilities

Assuming any individual i comes from one of the sampled groups $1, \dots, G$, the probability that i belongs to group g is defined as the ratio of the likelihood:

$$p(g_i = g) = \frac{p(x_i|g_i = g, f_g, \pi)}{\prod_g p(x_i|g_i = g, f_g, \pi)} \tag{9}$$

1.2 Estimation using the EM algorithm

The model formulation above supposes that both the groups g, and the allele frequencies in these groups f_g, are known. In practice, though, these need to be estimated. This is achieved using the expectation-maximization algorithm, which we apply as follow:

1. define initial groups for invididuals, g
2. (expectation) compute allele frequencies f_g and then $p(g_i = g)$ for all i and g
3. (maximization) assign individuals to their most likely group
4. return to 2) until convergence

Here, we consider that the algorithm converged if the change in the global log-likelihood is less than 1e-14. The advantage of this algorithm is that it converges very fast, typically in less than 10 iterations. The first step can be done at random, in which case several runs of the EM algorithm can be useful to ensure that the best solution (with highest log-likelihood) is attained. Alternatively, clusters can be defined by another fast clustering method, such as the k-means implemented in `find.clusters`.

1.3 Identifying hybrids

The allele frequency f_h in a hybrid population h_w is modelled as weighted averages of the allele frequencies in the parental populations g_1 and g_2:

$$f_{h_w} = w f_{g_1} + (1 - w) f_{g_2} \tag{10}$$

where w has value between 0 and 1. Typical values of w are 0.5 for F1 hybrids, 0.75 for backcrosses F1/1, 0.25 for backcrosses F1/2, etc.

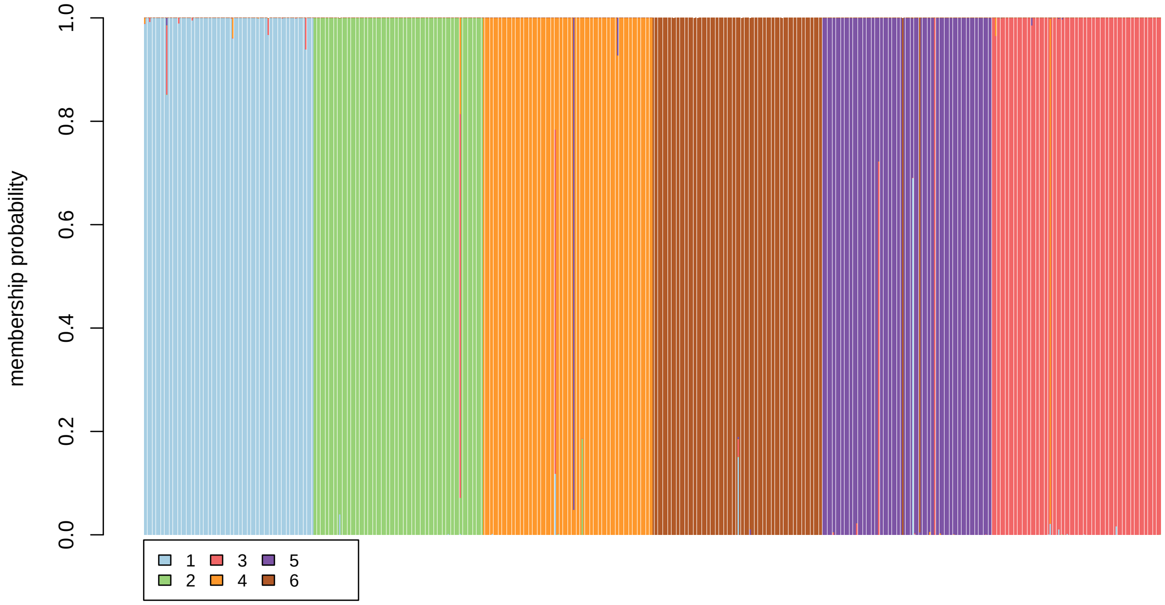
The EM algorithm described above can be extended to account for various hybrids using:

1. define initial groups for invididuals, g
2. (expectation) compute allele frequencies for parental populations f_{g_1} and f_{g_2} and subsequently for all hybrid populations h_w, and then $p(g_i = g)$ and $p(g_i = h_w)$ for all i, g and h_w
3. (maximization) assign individuals to their most likely group
4. return to 2) until convergence

2 Example using simulated data

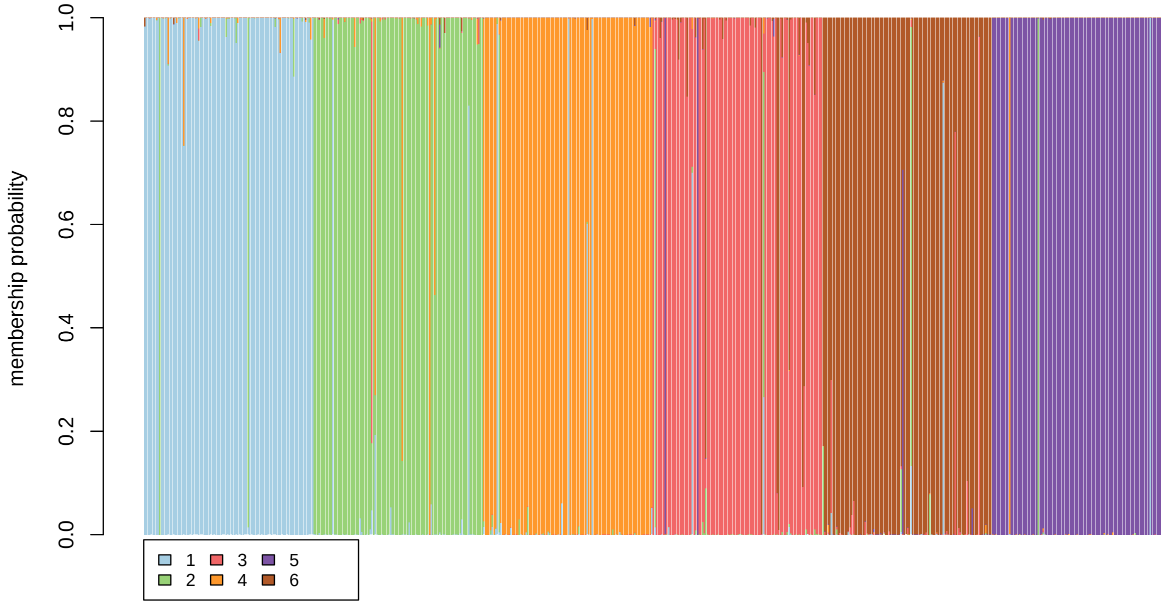
2.1 In the absence of hybrids: DAPC data revisited

```
library(adegenet)
data(dapcIllus)
sapply(dapcIllus, nPop)

##  a  b  c  d
##  6  6 12 24

a.clust <- genclust.em(dapcIllus$a, k = 6)
table(pop(dapcIllus$a), a.clust$group)

##
##         1   2   3   4   5   6
##    P1 100   0   0   0   0   0
##    P2   0  99   1   0   0   0
##    P3   0   0   1  98   1   0
##    P4   0   0   0   0   0 100
##    P5   1   0   2   0  95   2
##    P6   0   0  99   1   0   0

compoplot(a.clust)
```

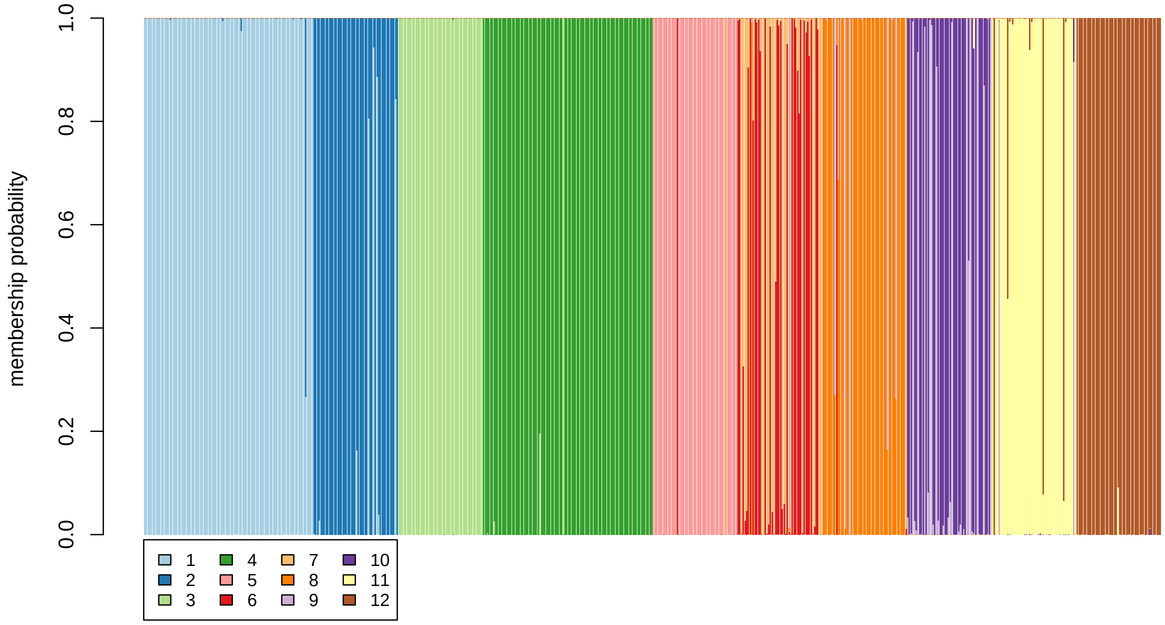


```
b.clust <- genclust.em(dapcIllus$b, k = 6)
table(pop(dapcIllus$b), b.clust$group)

##
```

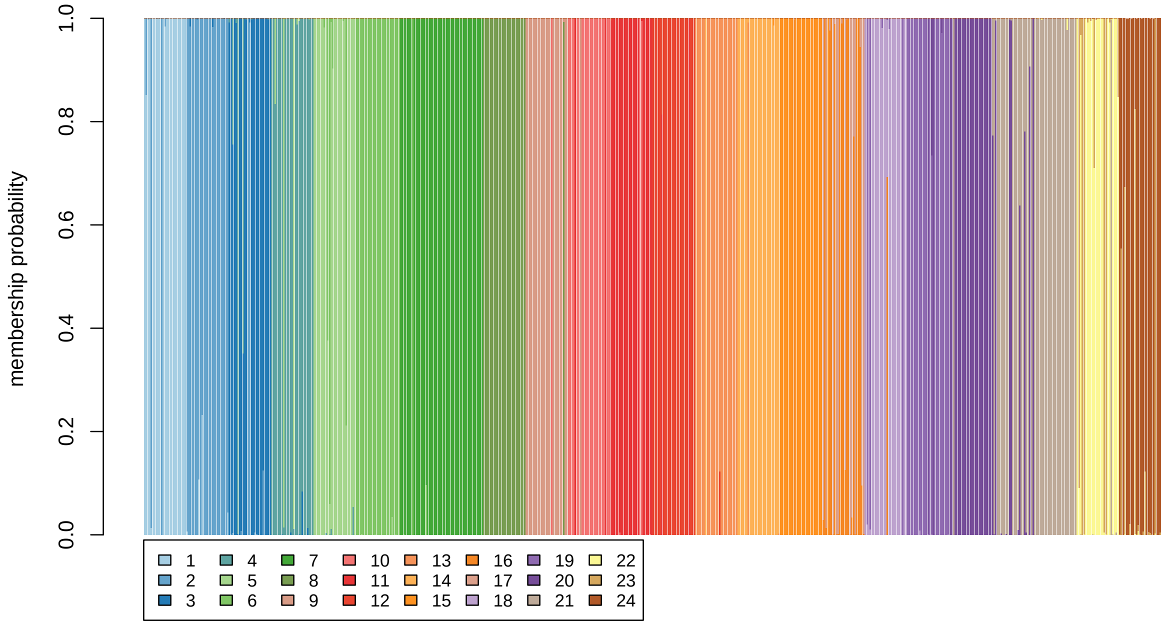
```
##         1  2  3  4  5  6
##   P1 98  2  0  0  0  0
##   P2  2 93  1  4  0  0
##   P3  4  1  0 95  0  0
##   P4  1  2 88  0  2  7
##   P5  1  1  2  0  1 95
##   P6  1  1  0  1 97  0

compoplot(b.clust)
```



```
c.clust <- genclust.em(dapcIllus$c, k = 12)
table(pop(dapcIllus$c), c.clust$group)

##
##          1  2  3  4  5  6  7  8  9 10 11 12
##   P01 50  0  0  0  0  0  0  0  0  0  0  0
##   P02 49  1  0  0  0  0  0  0  0  0  0  0
##   P03  4 46  0  0  0  0  0  0  0  0  0  0
##   P04  0  0 50  0  0  0  0  0  0  0  0  0
##   P05  0  0  1 49  0  0  0  0  0  0  0  0
##   P06  0  0  0 50  0  0  0  0  0  0  0  0
##   P07  0  0  0  0 48  1  1  0  0  0  0  0
##   P08  0  0  0  0  0 28 20  1  1  0  0  0
##   P09  0  0  0  0  0  1  1 43  5  0  0  0
##   P10  0  0  0  0  0  0  0  0 13 37  0  0
##   P11  0  0  0  0  0  0  0  0  2  0 44  4
##   P12  0  0  0  0  0  0  0  0  0  0  0 50

compoplot(c.clust)
```



```
d.clust <- genclust.em(dapcIllus$d, k = 24)
table(pop(dapcIllus$d), d.clust$group)

##
##          1  2  3  4  5  6  7  8  9 10 11 12 13 14 15 16 17 18 19 20 21 22 23
##   P01 23  2  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0
##   P02  0 25  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0
##   P03  0  2 21  2  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0
##   P04  0  0  0 23  2  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0
##   P05  0  0  0  0 22  3  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0
##   P06  0  0  0  0  0 25  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0
##   P07  0  0  0  0  0  2 23  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0
##   P08  0  0  0  0  0  0 25  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0
##   P09  0  0  0  0  0  0  0 25  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0
##   P10  0  0  0  0  0  0  0  1 23  1  0  0  0  0  0  0  0  0  0  0  0  0  0
##   P11  0  0  0  0  0  0  0  0  1 22  2  0  0  0  0  0  0  0  0  0  0  0  0
##   P12  0  0  0  0  0  0  0  0  0  2 23  0  0  0  0  0  0  0  0  0  0  0  0
##   P13  0  0  0  0  0  0  0  0  0  0  2 23  0  0  0  0  0  0  0  0  0  0  0
##   P14  0  0  0  0  0  0  0  0  0  0  0  0 24  1  0  0  0  0  0  0  0  0  0
##   P15  0  0  0  0  0  0  0  0  0  0  0  0  1 24  0  0  0  0  0  0  0  0  0
##   P16  0  0  0  0  0  0  0  0  0  0  0  0  0  0 24  1  0  0  0  0  0  0  0
##   P17  0  0  0  0  0  0  0  0  0  0  0  0  0  0  1 13 11  0  0  0  0  0  0
##   P18  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  1  0 21  3  0  0  0  0
##   P19  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0 23  2  0  0  0
##   P20  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0 24  1  0  0
##   P21  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  8 17  0  0
##   P22  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0 25  0  0
##   P23  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  1 19  5
##   P24  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  4
##
##       24
##   P01  0
##   P02  0
##   P03  0
##   P04  0
##   P05  0
##   P06  0
##   P07  0
##   P08  0
##   P09  0
##   P10  0
##   P11  0
##   P12  0
##   P13  0
##   P14  0
##   P15  0
##   P16  0
##   P17  0
##   P18  0
##   P19  0
##   P20  0
##   P21  0
##   P22  0
##   P23  0
##   P24 21

compoplot(d.clust, n.col=8)
```



2.2 Looking for hybrids

2.2.1 Simulating hybrids using hybridize

Simulate hybrids F1

```
set.seed(1)
data(microbov)

zebu <- microbov[pop="Zebu"]
salers <- microbov[pop="Salers"]
hyb <- hybridize(zebu, salers, n=30)
x <- repool(zebu, salers, hyb)
```

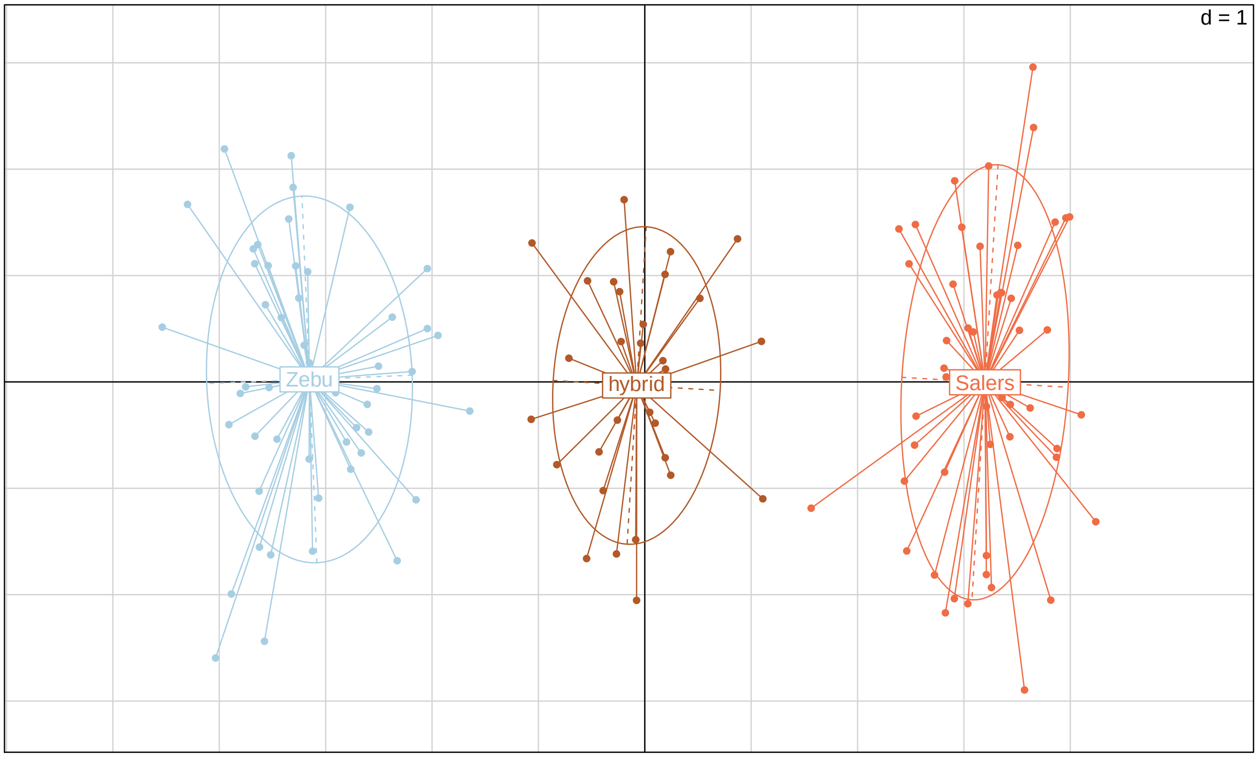
Simulate hybrids backcross (F1 / parental)

```
f1.zebu <- hybridize(hyb, zebu, 20, pop = "f1.zebu")
f1.salers <- hybridize(hyb, salers, 25, pop = "f1.salers")
y <- repool(x, f1.zebu, f1.salers)
```

2.2.2 Looking for F1 hybrids

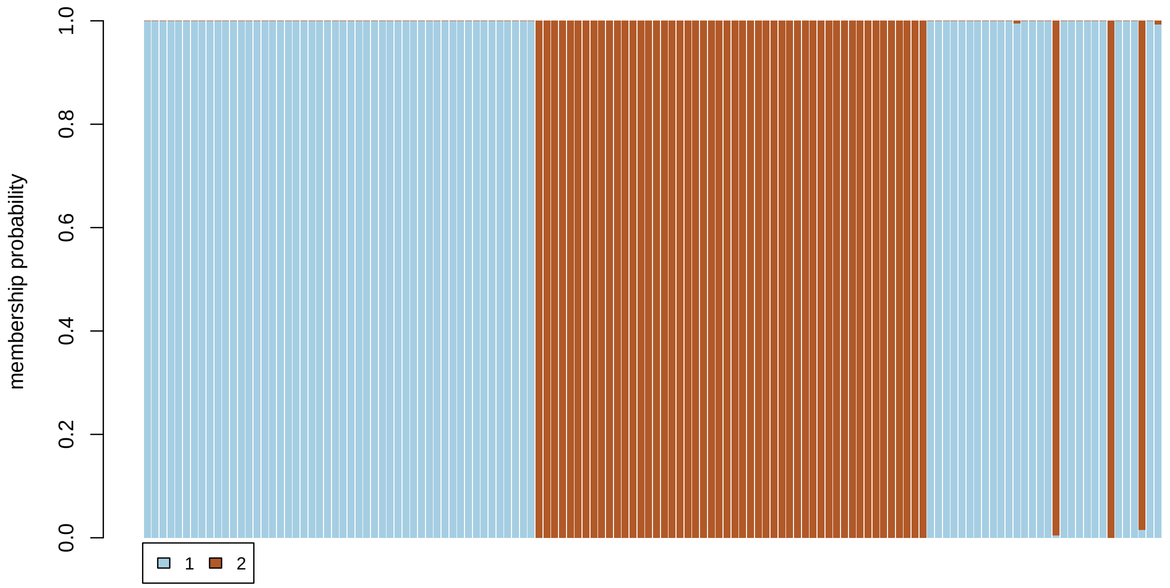
PCA:

```
x.pca <- dudi.pca(tab(x, NA.method="mean"), scannf = FALSE, scale = FALSE)
s.class(x.pca$li, pop(x), col=funky(3))
```



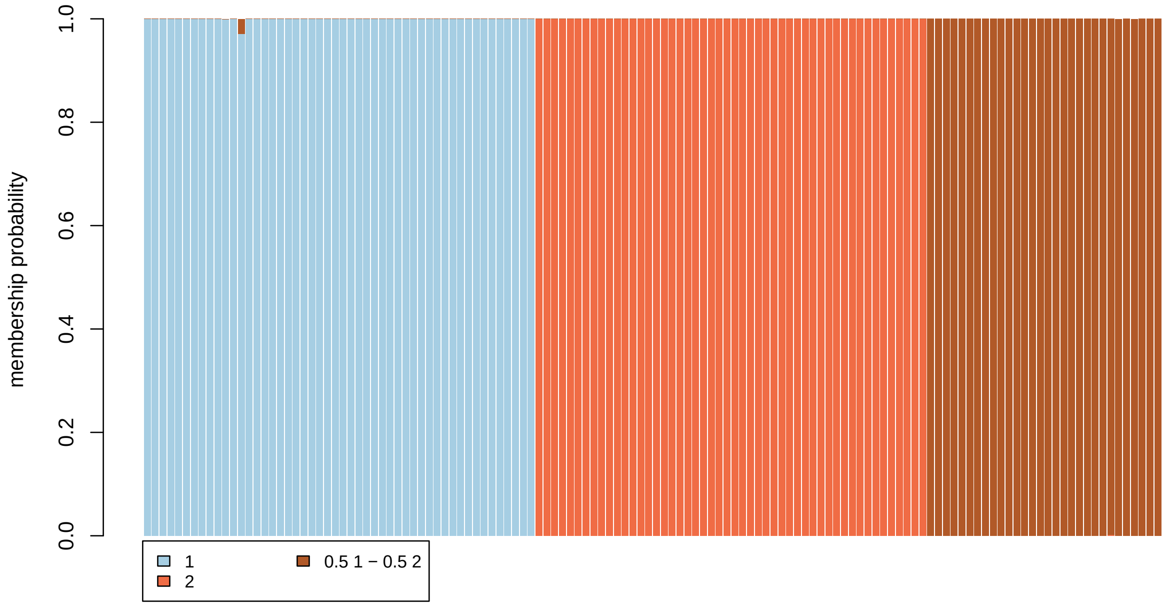
method without hybrids

```
res.no.hyb <- genclust.em(x, k=2, hybrids=FALSE)
compoplot(res.no.hyb, n.col=2)
```



method with hybrids

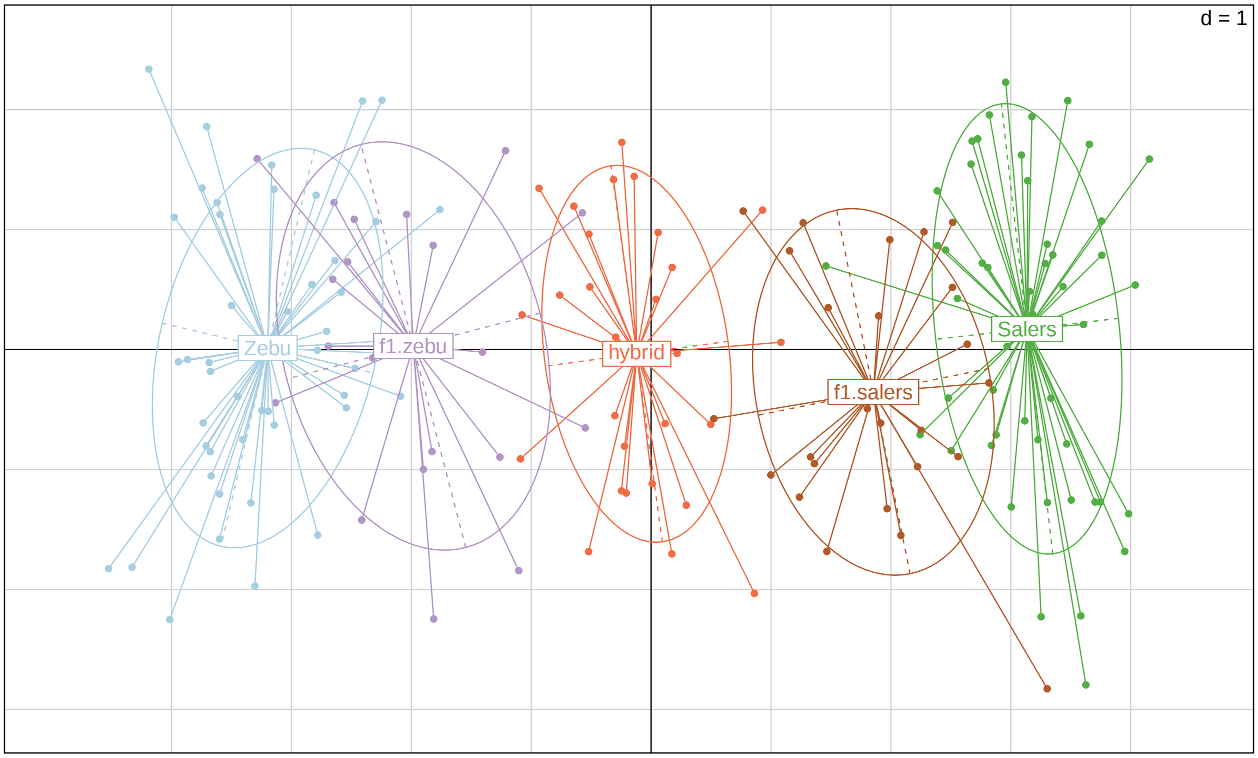
```
res.hyb <- genclust.em(x, k=2, hybrids=TRUE)
compoplot(res.hyb, n.col=2)
```



2.2.3 Looking for F1 and back-crosses

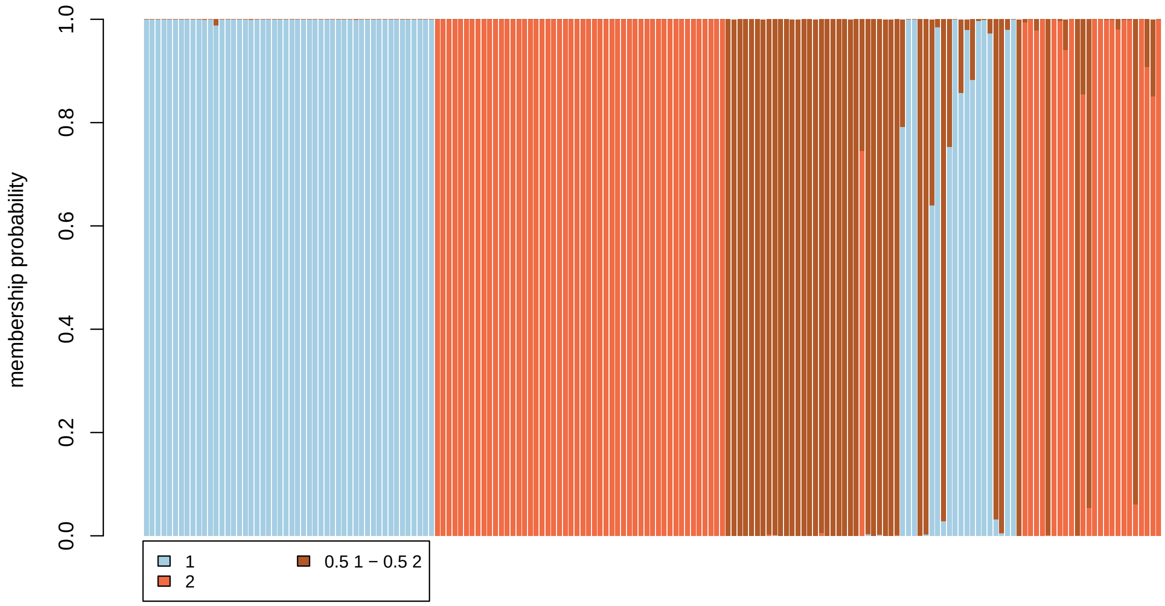
PCA:

```
y.pca <- dudi.pca(tab(y, NA.method = "mean"), scannf = FALSE, scale = FALSE)
s.class(y.pca$li, pop(y), col=funky(5))
```



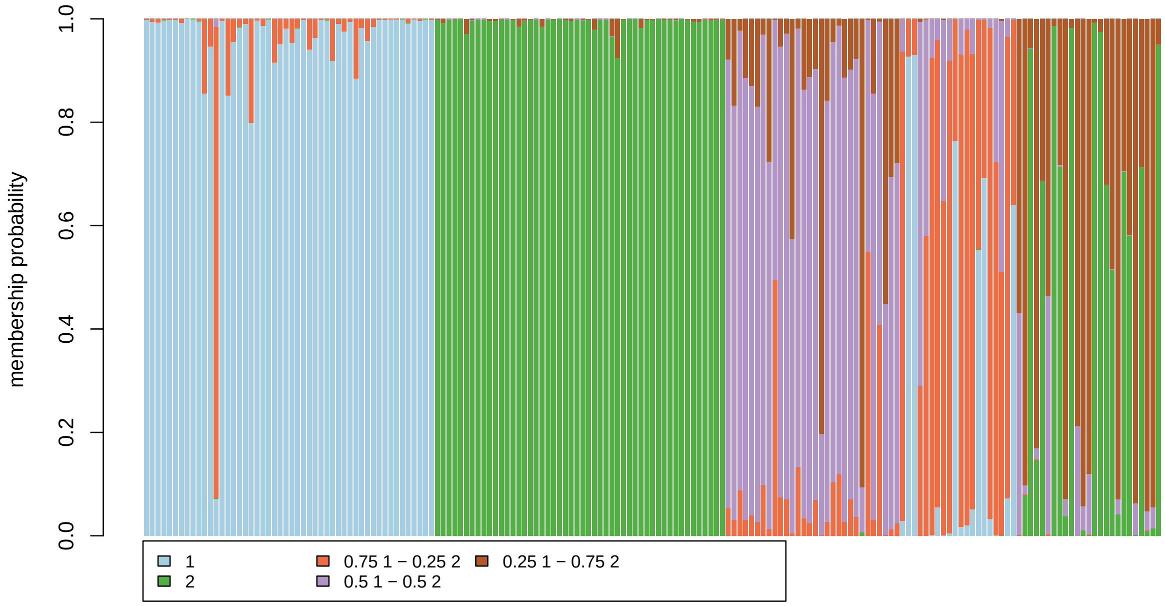
method with hybrids F1 only

```
res2.hyb <- genclust.em(y, k = 2, hybrids = TRUE)
compoplot(res2.hyb, n.col=2)
```



method with back-cross

```
res2.back <- genclust.em(y, k=2, hybrids = TRUE, hybrid.coef = c(.25,.5))
compoplot(res2.back)
```


membership probability