The RZooRoH package

Tom Druet, Amandine Bertrand, Naveen Kadri & Mathieu Gautier

2018-11-19

The `RZooRoH` package offers functions to identify Homozygous-by-Descent (HBD) segments and to estimate individual autozygosity (or inbreeding coefficients). HBD segments are created when an individual inherits two copies of the same chromosome segment from an ancestor. The two copies are inherited through different pathways, one copy was inherited from the mother, the second from the father. This happens in presence of inbreeding, when parents are related (they share a common ancestor). In absence of mutations, this creates long runs of homozygous genotypes (RoH). The length of the segments depends on the number of generations from the individual to the common ancestor (the generations from the two distinct paths must be summed) or the size of the inbreeding loop. Most often, multiple ancestors contribute to the autozygosity of an individual. These ancestors trace back to different generations in the past and are hence associated with inbreeding loops of variable size. As a result, the length of the HBD segments is expected to vary. Therefore, the model used in the package relies on multiple HBD classes related to the age of the segments (longer segments and smaller rates for recent autozygosity / recent common ancestor).

The `RZooRoH` package is available for most platforms (Linux, MS Windows and MacOSX) from the CRAN repository (https://CRAN.R-project.org/package=RZooRoH).

1 Installation

To install the package, open R and run:

```
install.packages("RZooRoH")
```

2 The multiple HBD classes model

2.1 The hidden Markov model with two classes (HBD *vs* non-HBD)

The model is an hidden Markov model (**HMM**) describing the genome of an individual as a succession of HBD and non-HBD segments. The unobserved HBD status is evaluated at each marker position. The two HBD status (the position is HBD or is not HBD) are not observed and referred to as hidden states. To compute the probability of a specific succession of HBD and non-HBD segments, the model requires the probability to continue or end the current segment (related to so-called transition probabilities) and the probability to observe the genotypes or the reads (with sequencing data) conditionally on the HBD status (so-called emission probabilities). In the case the current segment stops, we need also to define the probability to start a new HBD or non-HBD segment:

- The length of HBD segments is exponentially distributed (Thompson, 2013). We can use this to model the probability to continue or stop a segment. The probability to end a segment between two markers separated by d Morgans is e^{-Rd}, where R is the rate of the exponential distribution. The expected length of HBD segments is then 1/R Morgans (high rates corresponding to shorter segments).
- The probability to observe a genotype depends on the HBD status, the allele frequencies, the genotyping error rate and the mutation rate. In HBD segments, genotypes are expected to be mostly homozygous with higher probabilities to observe frequent alleles. In non-HBD segments, genotypes are expected to follow Hardy-Weinberg proportions. When genotypes are not know with high confidence (e.g., with low fold-sequencing data), the HMM framework allows to use the genotype probabilities and to integrate over the three possible genotypes.

- The probability to start a new HBD or non-HBD segment will be equal to the mixing proportion. An HMM is indeed a mixture distribution and the mixing proportions defines how frequent the different hidden states are.

More details on one HBD class HMM can be found in Leutenegger (2003), Vieira et al. (2016), Narasimhan et al. (2016) and Druet and Gautier (2017).

2.2 Extending to multiple HBD classes

Assuming a single HBD class amounts to consider that all HBD segments have the same expected length. This might be interpreted biologically as considering that all the autozygosity traces back to a single ancestor or several ancestors living in the same generation. A multiple HBD classes model assumes instead that each HBD class has its own expected length and frequency allowing to fit more realistic situations where ancestors contributing to autozygosity trace back to different generations in the past. This is the case in most populations (see Druet and Gautier (2017) or Sole et al. (2017) for examples).

The multiple HBD classes HMM still describes the genome of an individual as a succession of HBD and non-HBD segments. However, the HBD segments are categorized in different classes (e.g., very long, long, normal, short or very short HBD segments). Each of the classes defines an hidden state. The principle is the same as for the first HMM except that several classes of HBD segments are defined (based on their length). To compute the probability of a specific succession of HBD and non-HBD segments, the model requires the same probabilities as before:

- The length of HBD segments is exponentially distributed. Now, each class has its own rate R_k, where k is the class number. The probability to end a segment from class k between two markers separated by d Morgans is $e^{-R_k d}$. The expected length of HBD segments from class k is then $1/R_k$ Morgans (high rates corresponding to shorter segments).
- The probability to observe a genotype depends on the HBD status, the allele frequencies, the genotyping error rate and the mutation rate. The same emission probabilities are used for all HBD classes (see above).
- The probability to start a new HBD or non-HBD segment will be equal to the mixing proportion. Each class k has its own mixing coefficient M_k. This coefficient defines the frequency (the abundance) of the segments from each class.

A full description of the model is described in Druet and Gautier (2017).

2.3 Identifying HBD segments and estimating HBD probabilities with an HMM

We presented above some elements to compute the probability of a single succession of HBD and non-HBD segments but this is not our main interest. Indeed, we want to compute the probability of a possible sequence (of states) in order to find the best sequence or to estimate probabilities by integration over all sequences. The number of possible sequences of states increases rapidly with the number of markers $nsnp$ and is equal to K^{nsnp}, where K is the number of states (classes).

Fortunately, in the HMM framework we can efficiently compute the likelihood of the data and the probability to belong to each class at each marker position with the forward-backward algorithm (Rabiner, 1989). With the forward-backward algorithm, the probabilities are obtained by integration over all possible sequences of states. As a result we obtain at a marker position, the probability that the genome of the studied individual belongs to a HBD segment from class k. The probabilities are obtained by integrating over all possible length of HBD segments (over all possible window sizes).

Alternatively, it is also possible to use the Viterbi algorithm (Rabiner, 1989) to identify the most likely sequence of states or succession of HBD and non-HBD segments. In that case, every marker position is assigned to one of the K classes and stretches of markers assigned to the same class form HBD segments. We do not longer obtain probabilities.

2.4 Model and results interpretation

The different HBD classes are defined by their specific rates R_k. The length of HBD segments from class k is exponentially distributed with rate R_k and mean $1/R_k$. Classes with lower rates correspond to longer HBD segments from more recent common ancestors. Therefore, different HBD classes can be interpreted as HBD segments of different groups of ancestors tracing back to different generations in the past. The rate of the class is approximately equal to the size of the inbreeding loop in generations (Druet and Gautier, 2017). So, the rate of the class is approximately equal to twice the number of generations to the common ancestor. For instance, a class with a rate R_k equal to 10 would correspond approximately to ancestors five generations ago whereas a class with a rate R_k equal to 100 would correspond approximately to ancestors fifty generations ago. These are off course no precise estimations of age of HBD segments but rather qualitative measures.

The HMM relies on two sets of parameters, the rates R_k and the mixing coefficients M_k. The interpretation of the rates R_k has been explained in the previous paragraph. With single HBD class models, the mixing coefficient can be interpreted as the inbreeding coefficient (see (Leutenegger et al., 2003)). However, in multiple HBD classes models, the mixing coefficients have no straightforward biological interpretation. They represent the frequency of the segments from different classes but not their contribution to total autozygosity. Indeed, one long segment can contribution more than many small segments. To estimate the contribution of each class to the genome we will rather use the estimated probabilities (see next).

With the forward-backward algorithm, we obtain at each marker position the probability to belong to each state. We call the probability estimated at one marker position a local probability whereas global probabilities are obtained by averaging local probabilities over the genome, they are genome-wide estimates. The HMM provides local and global probabilities to belong to each class. Global probabilities provide the contribution from each class to the genome and for HBD classes their provide an estimate of autozygosity associated with one class. We call the estimated contribution of one HBD class as the realized autozygosity.

The sum of probabilities to belong to each HBD class provide an estimate of the total HBD probability. The sum can be local and provides hence estimate of local HBD probability that can be used for homozygosity/autozygosity mapping experiments (Wang et al., 2009; Leutenegger et al., 2006). If the sum is genome-wide, then its provides an estimate of total autozygosity.

Inbreeding coefficients are estimated with respect to a base population that needs to be defined by the user. The different HBD classes and their rates R_k allow to define base populations since rates are related to the "age of the ancestors" (see first paragraph of this section). By summing the global probabilities from all HBD classes with a rate smaller or equal than a threshold T ($R_k \leq T$), we obtain an estimate of the inbreeding coefficient F when the base population is set approximately 0.5^*T generations in the past (a few more generations).

Finally, when running the Viterbi algorithm, we obtain segments associated with different HBD classes (stretches of markers assigned to the same HBD class). We can define HBD segments as segments associated with any of the HBD classes or restrict the definition to HBD classes with a rate smaller than a threshold (eventually defining the base population).

3 Differences with other approaches

3.1 Some differences with window-based approaches identifying ROH

Several approaches to identify ROH (later interpreted as HBD segments) are based on gathering information in sliding windows. With rule-based methods (McQuillan et al., 2008), stretches of genotypes fulfilling certain criteria (e.g., number of SNPs, number of heterozygous and missing genotypes, marker density, window length, marker spacing) are interpreted as HBD segments. Ideally, the criteria should be optimized for every data set (population, genotyping technology, etc.). Such an approach is implemented in PLINK (Purcell et al., 2007). In likelihood-based approaches (Broman and Weber, 1999; Pemberton et al., 2012), LOD-scores are computed to classify windows as HBD or non-HBD. The marker allele frequencies and genotyping errors are

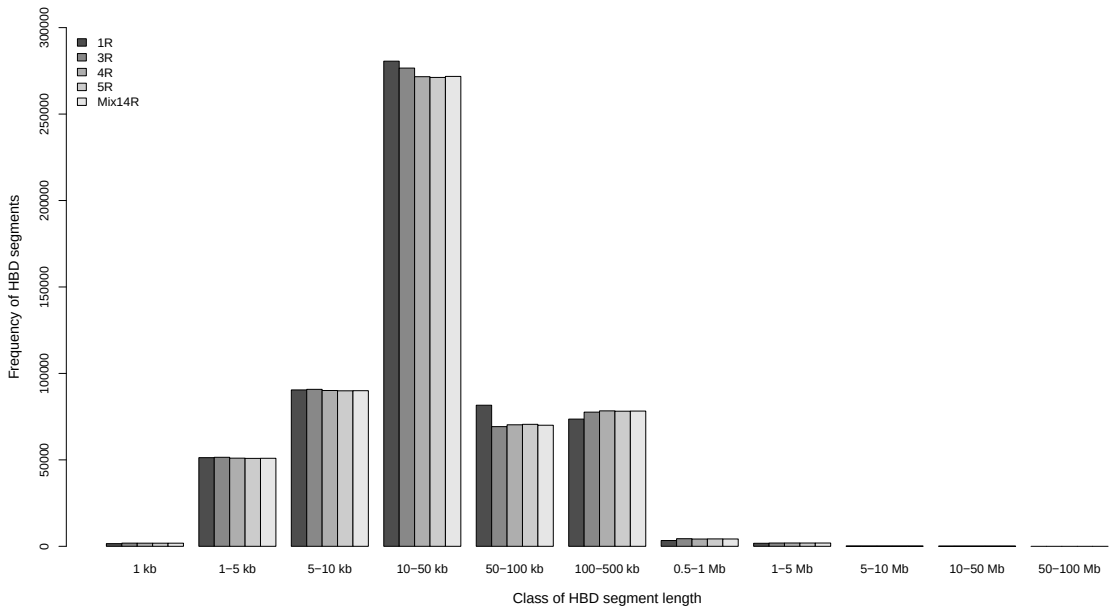


Figure 1: Distribution of length of HBD segments identified in Belgian Blue beef cattle with different models (results from Sole et al. (2017)). We observe that a model with 1 HBD class (1R) has more segments of intermediary size (10-100 kb). The model with 14 classes identifies more long segments (> 500 kb).

used to compute the likelihood, making the approach less sensitive to marker recruitment bias. The optimal window size is determined by selecting the smallest value resulting in a clear bi-modal distribution of LOD scores. Likelihood-based approaches should be preferred to rule-based approaches since they better account for allele frequencies and genotyping errors.

The emission probabilities of the HMM use the allele frequencies and the genotyping error rates similarly to likelihood-based ROH. Therefore, they are less sensitive to marker recruitment bias or filtering criteria (minimum MAF). Sometimes rule-based ROH are run inappropriately with monomorphic markers. In HMM, these markers are automatically non-informative. In addition, HMM use information from the genetic map (distance between markers), taking automatically into account marker density or marker spacing and being more robust to variable recombination rates along the genome. HMM also automatically explore all possible lengths of HBD segments (they do not require the definition of an optimal window size) and provide HBD probabilities. Another benefit from the HMM is that they can work with genotype probabilities or likelihoods and with irregular marker spacing (exome, GBS). Therefore, they can also efficiently handle exome or whole-genome (including low-fold) sequence data (Magi et al., 2014; Narasimhan et al., 2016; Vieira et al., 2016).

ROH-based approaches and HMM have been compared in a few simulations studies. As often, the simulated scenarios can influence the results (the selected marker density, a uniform marker spacing, the presence of genotyping errors or not, etc). In addition, the parameters of the ROH can be adapted to better fit the simulations. Therefore, comparisons should be interpreted with caution. Recently, Narasimhan et al. (2016) found that HMM had lower false positive rates (FPR) and false negative rates (FNR) compared to ROH estimated with PLINK. Druet and Gautier (2017) concluded that the differences between the three approaches was small when informativity was high (many SNPs per HBD segment) whereas the HMM performed better for shorter segments or at lower marker density in terms of the estimated realized inbreeding coefficient or the local autozygosity estimation (e.g. at a locus). By changing rules to call ROH (e.g., window size or number of heterozygous SNPs in ROH), it was possible to optimize the behavior of the window-based approaches. For instance, the FNR can be increased at the expense of a

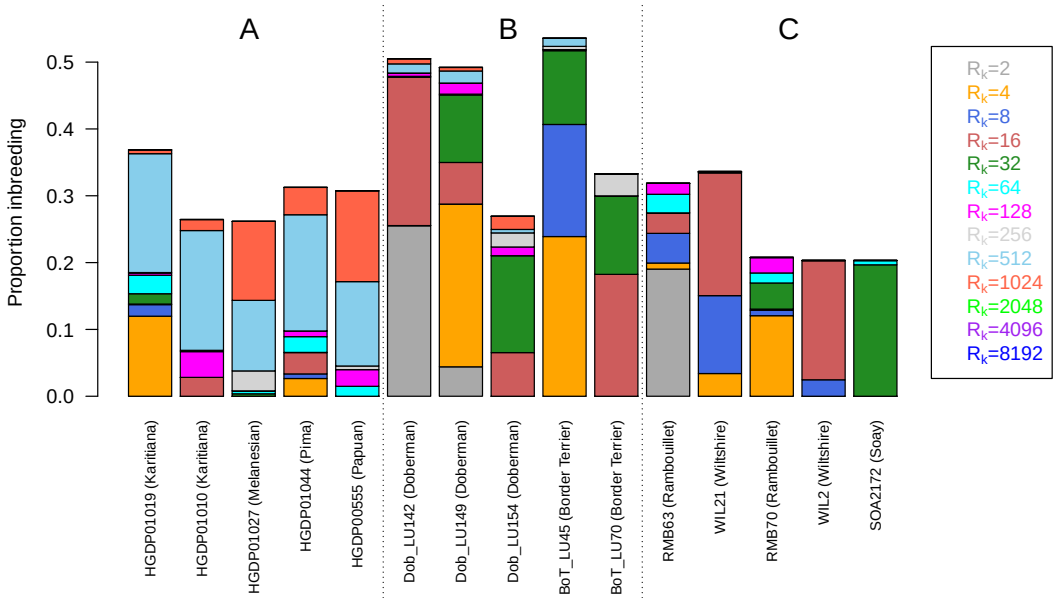


Figure 2: Estimated level of autozygosity per HBD class in five humans (A), five dogs (B) and five sheeps (C). Each colour is associated with a distinct class (defined by its rate). The heights of each colour bar represent the estimated level of autozygosity associated with the class, and the total height represents the total estimated autozygosity (results from Druet and Gautier (2017)). Three dogs have total values close to 0.5 and the method indicates that 25 percent of the genome is associated with HBD classes with rates equal to 2 or 4. These values are compatible with parent-offspring matings combined with additional autozygosity from more distance ancestors. The most inbred sheep has autozygosity levels higher than 0.30 but mainly associated with more distant ancestors (approximately 8 generations in the past). These examples illustrate that the partitioning in different HBD class might help to understand the relationship between the parents or past demographic events at the individual level.

higher FPR by using more aggressive parameters (e.g., shorter windows).

The use of HMM is particularly valuable when information is sparser (Druet and Gautier, 2017) and HBD versus non-HBD classification is more uncertain (e.g., low fold-sequencing experiments, lower marker density, older and shorter HBD segments, biased genotyping arrays, etc.). It is also very useful when the information is more variable (variable allele frequencies, distances between markers, recombination rates, coverages, genotyping errors). For instance, Magi et al. (2014) showed that an HMM based approach outperforms PLINK when applied to whole-exome sequences data by considering the distances between consecutive SNPs. Vieira et al. (2016) demonstrated the importance to use genotype likelihoods (as integrated in some HMM approaches) instead of genotypes (as used in window-based approaches) when dealing with low-fold sequencing data. HMM estimating HBD probabilities provide information on the uncertainty associated with the inference when the information is degraded. Overall, the HMM approach is particularly beneficial in situations encountered in wild organisms including lower marker densities, low-fold sequencing data, marker recruitment bias, uneven marker spacing, variable genotyping error rates, variable recombination rates, etc.

Finally, HMM present also conceptual differences with ROH-based methods since, by providing probabilities, they do not rely on the selection of various thresholds such as marker spacing, window size, minimum number of SNPs, etc. that should be re-defined for each data set.

3.2 Benefits of using multiple HBD classes

The multiple HBD classes model assumes that each HBD class has its own expected length and frequency (Druet and Gautier, 2017) allowing to fit realistic situations where ancestors contributing to autozygosity trace back to different generations in the past. These classes are related to the age of common ancestors associated with the HBD classes (see section 2.4): classes with longer segments (lower rates) correspond to more recent common ancestors and vice versa.

Here are some benefits of using multiple HBD classes:

- It results in a better fit of individual genetic data and more accurate estimations of autozygosity levels both locally and globally, particularly in complex populations. We showed through simulations that singles HBD class models might underestimate the autozygosity when multiple generations contribute to it (Druet and Gautier, 2017). Similarly, we illustrated with real cattle data that the use of single HBD class models results in lower estimates of autozygosity.
- With a single HBD class the distribution of length of identified HBD segments is more concentrated at intermediate values (Sole et al. (2017); see Figure 1). Indeed, the smallest segments are not captured with a single class whereas long segments are fragmented in multiple smaller segments. Obtaining the correct length of HBD segments is essential to interpret the results, in particular to estimate the age of the ancestor.
- These extended models offer the possibility to reveal the recent demographic history by partitioning HBD segments in different age-related classes and to estimate the contribution of different generations in the past to the current autozygosity, at both an individual and a population scales (Druet and Gautier, 2017).
  - At individual scale, presence of long HBD segments in HBD classes with rates R_k of two to four indicates that the parents were highly related (parent-offspring matings or brother-sister matings). Age of the HBD classes and their importance help to determine the relationship between parents and the mating habits (see Figure 2 for an example). This might be particularly interesting in studies on wild populations where relationships between parents are often unknown.
  - At the population level, large contributions of a class to autozygosity indicates a reduced effective population size (Ne) at that time period possibly associated with a bottleneck or a founder effect. Similarly, low contribution to autozygosity suggests large Ne in a past period.
- The use of multiple HBD classes makes LD pruning less important because recent and ancient HBD segments are separated in distinct classes (Druet and Gautier, 2017). Indeed, we compared the results obtained with a single HBD class HMM applied with a pruning strategy (the data set with $>$ 600,000 SNPs was divided in 100 data sets with $\sim$ 6500 SNPs, 1% of the data) (Leutenegger et al., 2011) and those obtained with our model (using 14 HBD classes and without pruning). The results from both studies were consistent; for instance the autozygosity estimated by Leutenegger et al. (2011) were highly correlated to the recent autozygosity associated with the first four HBD classes (see Figure 3). More details are described in (Druet and Gautier, 2017).

4 Input data

4.1 Data format and conversion from PED or VCF files

Several data formats are accepted, including the Oxford GEN format (Marchini et al., 2007). SNPs are always organized by row and individuals per column. The first columns should include marker information and subsequent column genotype / sequence information per individual. The package runs on autosomes (e.g., no sexual chromosome) and for bi-allelic markers.

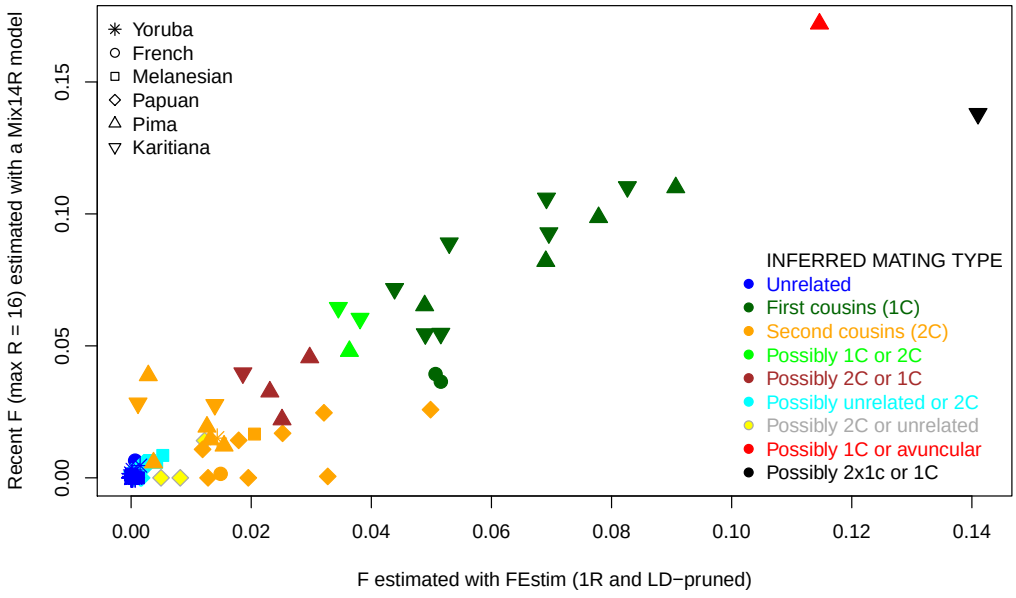


Figure 3: Comparison of individual autozygosity estimated with FEstim (1 HBD class) and a pruning strategy with recent autzoygosity classes estimated with a multiple HBD classes model (results from Druet and Gautier (2017)).

4.1.1 Fields for marker information

The number of columns used to described the markers is not fixed, by default we expect five columns with marker information (chromosome, marker name, position, first allele and second allele). Two columns are mandatory: the chromosome field and the position field. The other fields are not used. More or less columns can be provided and the order of the columns can be changed as the read function (`zoodata`) allows to specify the information.

The position field should ideally be in genetic distances. The package works with distances in cM multiplied by 1,000,000. When genetic distances are not available, physical distances in base pairs (bp) can be used, assuming 1 Mb = 1 cM as observed in some species. Internally, distances are converted to Morgans (division by 100,000,000) and rates can be interpreted as a function of number of generations to the ancestor. If positions are expressed on a different scale, then the model will adjust by rescaling the rates by the same magnitude (in case the rates are estimated).

SNP with a null position will be discarded. Ideally, this should done prior to load data in `RZooRoH`.

4.1.2 Fields for genotype / sequence information

After the columns for the marker information come the columns with the genotype / sequence information. One, two or three columns are expected per individual according to the format:

- GT format: 1 column per individual containing the genotypes expressed as allele dosages, 0 for AA, 1 for AB and 2 for BB. Missing genotypes are indicated by a value of 9.
- GL format: three columns per individual containing genotype likelihoods in phred scores for the three possible genotypes REF/REF, REF/ALT, ALT/ALT. This field is available in some VCF files (obtained

after variant calling with sequencing data).

- GP format: three columns per individual containing genotypes probabilities for the three possible genotypes AA, AB and BB. This format corresponds to the Oxford GEN format.
- AD format: two columns per individual containing the number of reads for the first allele and for the second allele. This field is also available is some VCF files.

4.1.3 Examples for the four data formats

Example of a file in **GT format**. There are four columns to describe the markers: chromosome, position (in bp), first allele and second allele. Then we have genotypes for ten individuals (we observe only 0's and 1's, AA and AB).

```
file1 <- system.file("exdata","BBB_PE_gt_subset.txt",package="RZooRoH")
myfile1 <- read.table(file1,header=FALSE)
head(myfile1)
#>     V1    V2 V3 V4 V5 V6 V7 V8 V9 V10 V11 V12 V13 V14
#> 1 chr1  6665  G  A  0  0  0  0  0   0   0   0   0   0
#> 2 chr1  9149  G  A  0  0  0  1  1   0   0   1   0   1
#> 3 chr1 13812  T  C  1  1  0  1  1   0   0   0   1   1
#> 4 chr1 29575  C  G  0  0  0  0  0   0   0   1   0   0
#> 5 chr1 31490  T  C  0  0  0  0  0   0   0   1   0   0
#> 6 chr1 33632  C  T  0  0  0  0  0   0   0   1   0   0
```

Example of a file in **GL format**. There are five columns to describe the markers: chromosome, marker name, position (in bp), first allele and second allele (default format). Then we have genotype likelihoods in phred scores (three values per individual) for ten individuals. We print the first three individuals. The phred scores can be converted in genotype probabilities (see https://samtools.github.io/hts-specs/VCFv4.1.pdf for more information).

```
file2 <- system.file("exdata","BBB_NMP_pl_subset.txt",package="RZooRoH")
myfile2 <- read.table(file2,header=FALSE)
head(myfile2[,1:14])
#>     V1          V2    V3 V4 V5 V6 V7  V8 V9 V10 V11 V12 V13 V14
#> 1 chr1           .  9149  G  A  0 12 213  0   3  41   0   0   0
#> 2 chr1 rs208929886 35740  T  A 37  3   0  0   0   0   0   0   0
#> 3 chr1 rs208268304 35742  C  G 37  3   0  0   0   0   0   0   0
#> 4 chr1 rs384017208 36011  G  A  0  6  78  0   0   0   0   0   0
#> 5 chr1 rs132895573 36337  G  A  0  6  78 32   0  73   0   0   0
#> 6 chr1 rs208669904 36440  A  G  0  3  40 41   3   0   0   0   0
```

Example of a file in **GP format**. There are also five columns to describe the markers as above. Then we have genotype probabilities for AA, AB and BB genotypes (three values per individual) for ten individuals. We print the first three individuals. This corresponds to the oxford GEN format when the marker information columns are chromosome, marker name, position, first allele and second allele.

```
file3 <- system.file("exdata","BBB_NMP_GP_subset.txt",package="RZooRoH")
myfile3 <- read.table(file3,header=FALSE)
head(myfile3[,1:14])
#>     V1            V2   V3 V4 V5       V6       V7       V8       V9
#> 1 chr10  chr10:126_G_A  126  G  A 0.000000 0.000000 0.000000 0.531308
#> 2 chr10  chr10:297_C_T  297  C  T 0.888184 0.111816 0.000000 0.984398
```

```
#> 3 chr10 chr10:1173_C_T 1173  C  T 0.624537 0.313010 0.062454 0.666125
#> 4 chr10 chr10:1184_C_A 1184  C  A 0.799240 0.200760 0.000000 0.000793
#> 5 chr10 chr10:1194_C_A 1194  C  A 0.001670 0.333303 0.665027 0.000000
#> 6 chr10 chr10:1200_T_G 1200  T  G 0.000021 0.333854 0.666125 0.000000
#>        V10      V11 V12 V13 V14
#> 1 0.335233 0.133459   0   0   0
#> 2 0.015602 0.000000   0   0   0
#> 3 0.333854 0.000021   0   0   0
#> 4 0.998891 0.000316   0   0   0
#> 5 0.200760 0.799240   0   0   0
#> 6 0.200760 0.799240   0   0   0
```

For instance, the first individual has missing values for the first marker (the three genotype probabilities are null) and the second individual has genotype probabilities of 0.53, 0.34 and 0.13 for genotypes AA, AB and BB. For the fourth marker, the second individual has a high probability to be heterozygous AB (0.998). The third individual has missing information for the six printed genotypes. Note that these data has been obtained with an average coverage of 1x.

Example of a file in **AD format**. The same five columns are used to describe the markers. Then we have read counts for the two alleles (two values per individual) for ten individuals. We print the first five individuals. For instance, the first individual has four REF and zero ALT alleles for the first markers and one ALT allele for the third marker.

```
file4 <- system.file("exdata","BBB_NMP_ad_subset.txt",package="RZooRoH")
myfile4 <- read.table(file4,header=FALSE)
head(myfile4[,1:15])
#>     V1           V2    V3 V4 V5 V6 V7 V8 V9 V10 V11 V12 V13 V14 V15
#> 1 chr1            .  9149  G  A  4  0  1  0   0   0   2   1   1   0
#> 2 chr1 rs208929886 35740  T  A  0  0  0  0   0   0   0   1   0   0
#> 3 chr1 rs208268304 35742  C  G  0  1  0  0   0   0   0   1   0   0
#> 4 chr1 rs384017208 36011  G  A  2  0  0  0   0   0   1   0   2   0
#> 5 chr1 rs132895573 36337  G  A  2  0  2  1   0   0   1   0   1   1
#> 6 chr1 rs208669904 36440  A  G  1  0  0  1   0   0   1   0   0   1
```

4.1.4 Converting from ped or VCF files

If your file is in ped or VCF format, you can easily convert them to the Oxford GEN format with PLINK (https://www.cog-genomics.org/plink/1.9/) or BCFtools (http://samtools.github.io/bcftools/bcftools.html#convert). `RZooRoH` is then able to read the Oxford GEN format with the GP format.

For ped files, recode them to Oxford GEN format with:

```
plink --file myinput --recode oxford --autosome --out myoutput
```

The `--autosome` option keeps only SNPs on autosomes as required by `RZooRoH`.

For VCF files, BCFtools can be used to recode a VCF to the Oxford GEN format with the convert option:

```
bcftools convert -t ^chrX,chrY,chrM -g outfile --chrom --tag GT myfile.vcf
```

The `--chrom` option is important to obtain chromosome number in the first column. The `--tag` option allows to select which field from the vcf file (GT, PL, GL or GP) is used to generate the genotype probabilities exported in the oxford gen format. The `-t` option allows to exclude chromosomes (this is an example and chromosome names must be adapted if necessary). The needed output data is then outfile.gen.

If some genotype probabilities are missing, with a value of "-nan", you must replace them with "0" (triple 0 is considered as missing). This can be done with this command (bash command):

```
sed -e 's/-nan/0/g' file.gen > newfile.gen
```

4.2 Reading the data

An analysis with `RZooRoH` requires three mains steps: reading the data, defining the model and running the model. Here we describe the first step but first the package must be loaded:

```
library(RZooRoH)
```

The input data must be loaded with the `zoodata` function that creates a `zooin` object containing all the information for further analysis. With the `zoodata` function the user can specify the name of the data file, the genotype / sequence data format, the format of the marker information, minor allele frequencies (**MAF**) filtering rules and eventually the name of files with individuals IDs or with allele frequencies estimated previously (e.g., with a larger data set).

4.2.1 Specifying the data file and genotype / sequence format

The name of the data file is specified with the `genofile=filename` option and the genotype format with the `zformat` option. Four values are possible as described in sections 4.1.1 and 4.1.2: "gt", "gp", "pl" and "ad". The "gt" format is used when the `zformat` is not specified.

For instance, to read the file in the GP format:

```
file3 <- system.file("exdata","BBB_NMP_GP_subset.txt",package="RZooRoH")
BBB_GP <- zoodata(genofile = file3, zformat = "gp")
#> [1] "Number of positions in original file ::"
#> [2] "1000"
#> [1] "Number of positions after MAF filtering  ::"
#> [2] "1000"
```

The function tells that it has red 1000 lines (markers) and none were filtered out. Note that specifying `genofile =` is not required and the same result is obtained with this shorter command:

```
BBB_GP <- zoodata(file3, zformat = "gp")
#> [1] "Number of positions in original file ::"
#> [2] "1000"
#> [1] "Number of positions after MAF filtering  ::"
#> [2] "1000"
```

To read a file in the AD format, the command would be:

```
file2 <- system.file("exdata","BBB_NMP_ad_subset.txt",package="RZooRoH")
BBB_AD <- zoodata(file2, zformat = "ad")
#> [1] "Number of positions in original file ::"
#> [2] "1000"
#> [1] "Number of positions after MAF filtering  ::"
#> [2] "1000"
```

4.2.2 Specifying the format for marker information

As indicated in section 4.1.1, the `zoodata` function requires marker information that comes in the first columns of the file. Two fields are required, a column with the chromosome information and a column with

the position. By default, `zoodata` assumes five columns with marker information and that the chromosome is indicated in the first position whereas the position is in the third column. In the two above examples, the marker information had the default format and there was no need to use options associated with format of marker information fields.

When a different format is used, it can be specified with the `supcol`, `poscol` and `chrcol` options. The `supcol` option indicates the number of columns present before the first genotype / sequence data, the `poscol` option indicates the column with the genetic position information and the `chrcol` indicates the column with the chromosome information.

For instance, the "BBB_PE_GT_subset.txt" file contains genotypes in the GT format, four columns for marker information with the chromosome and genetic positions indicated in first and second columns respectively. To read the data we run the following command (note that the "gt" format does not need to be specified since it is the default format):

```
file1 <- system.file("exdata","BBB_PE_gt_subset.txt",package="RZooRoH")
BBB_GT <- zoodata(file1, supcol = 4, poscol = 2, chrcol = 1)
#> [1] "Number of positions in original file ::"
#> [2] "1000"
#> [1] "Number of positions after MAF filtering  ::"
#> [2] "1000"
```

4.2.3 Minimal allele frequency threshold

With the `min_maf` option, the user can specify a threshold for MAF filtering. For instance, if we want to keep only marker with a MAF higher than 0.05:

```
BBB_GT <- zoodata(file1, supcol = 4, poscol = 2, chrcol = 1, min_maf = 0.05)
#> [1] "Number of positions in original file ::"
#> [2] "1000"
#> [1] "Number of positions after MAF filtering  ::"
#> [2] "929"
```

Now, 929 markers remain after filtering. Note that the allele frequencies were estimated only with ten individuals. Some comments on filtering are providing in section 4.3.

4.2.4 Additional external files (sample names and allele frequencies)

The `zoodata` function can extract information from two additional files with the `samplefile` and the `allelefreq` options.

The `samplefile` option is used to indicate a file containing the names of the sample in the genotype / sequence data file. These names are not required and will be added to the results (for instance for plotting). The file should contain only one column with the names of the samples.

For instance, to add the name of the samples to the last data set:

```
mysamples <- system.file("exdata","BBB_samples.txt",package="RZooRoH")
BBB_GT <- zoodata(file1, supcol = 4, poscol = 2, chrcol = 1, samplefile = mysamples)
#> [1] "Number of positions in original file ::"
#> [2] "1000"
#> [1] "Number of positions after MAF filtering  ::"
#> [2] "1000"
```

The `allelefreq` option is used to indicate a file containing the allele frequencies of the first allele. The file should contain only one column with these allele frequencies. An example of the use of this option is given later (in section 6.2.3). The use of external frequencies can be useful when these were obtained from a

larger and more informative data set. For instance, when you want to run `RZooRoH` for a few individuals but estimates of allele frequencies are available from a large sample. Similarly, if you have only a few individuals sequenced but you have also frequencies obtained from a pool of individuals. The option allows also to use distinct frequencies such as estimates of ancestral allele frequencies. In addition, it avoids to re-estimate the frequencies for every run. Finally, skipping estimation of allele frequencies can save memory.

4.2.4 Structure of the zooin object

The `zooin` object is intended for internal use. It contains the nine slots necessary for further analysis: genos, bp, chrbound, nind, nsnps, nchr, zformat, sample_ids. These can be accessed by using the "@" symbol. For instance, to obtain the genotype table from the `zooin` object called BBB_GT you need to type BBB_GT@genos.

The zooin@genos is a matrix containing genotypes, genotype probabilities, read counts (the format is stored in the zooin@zformat variable). For the previous examples:

```
head(BBB_GT@genos)
#>      V5 V6 V7 V8 V9 V10 V11 V12 V13 V14
#> [1,]  0  0  0  0  0   0   0   0   0   0
#> [2,]  0  0  0  1  1   0   0   1   0   1
#> [3,]  1  1  0  1  1   0   0   0   1   1
#> [4,]  0  0  0  0  0   0   0   1   0   0
#> [5,]  0  0  0  0  0   0   0   1   0   0
#> [6,]  0  0  0  0  0   0   0   1   0   0

head(BBB_GP@genos[,1:6])
#>             V6       V7       V8       V9      V10      V11
#> [1,] 0.000000 0.000000 0.000000 0.531308 0.335233 0.133459
#> [2,] 0.888184 0.111816 0.000000 0.984398 0.015602 0.000000
#> [3,] 0.624537 0.313010 0.062454 0.666125 0.333854 0.000021
#> [4,] 0.799240 0.200760 0.000000 0.000793 0.998891 0.000316
#> [5,] 0.001670 0.333303 0.665027 0.000000 0.200760 0.799240
#> [6,] 0.000021 0.333854 0.666125 0.000000 0.200760 0.799240
```

The zooin@nind, zooin@nsnps and zooin@nchr represent the number individuals, the number of SNPs (after filtering) and the number of chromosomes. In the previous example, the data contains 10 individuals, 1000 SNPs and only one chromosome:

```
c(BBB_GT@nind, BBB_GT@nsnps, BBB_GT@nchr)
#> [1]   10 1000    1
```

The zooin@chrbound is a matrix with one row per chromosome and two values per row: the number of the first marker in that chromosome and the number of the last marker in that chromosome. The chromosome identifier (name or number) is stored in zooin@chrnames. The zooin@bp is an array with the marker genetic positions. The zooin@freqs contains the estimated or loaded allele frequencies for the first allele.

```
head(cbind(BBB_GT@bp, BBB_GT@freqs))
#>       [,1] [,2]
#> [1,]  6665 0.00
#> [2,]  9149 0.20
#> [3,] 13812 0.30
#> [4,] 29575 0.05
#> [5,] 31490 0.05
#> [6,] 33632 0.05
```

Finally, the zooin@sample_ids contains the sample names:

```
BBB_GT@sample_ids
#>  [1] "Maxx7"    "Kopi"     "Ever5337" "Skippy"   "Unkown"   "GIGA"
#>  [7] "Arlequin" "Black"    "Kenza"    "Minus"
```

4.3. Comment on data filtering

One important property of models using the allele frequencies (likelihood-based ROH or HMM) is that they account for marker allele frequencies and are therefore less sensitive to filtering rules. For rule-based ROH it would be important to remove monomorphic markers and eventually marker with low MAF. In the HMM, monomorphic markers are automatically ignored (because the emission probabilities are identical for HBD and non-HBD states) and when homozygous genotypes are observed, the support for HBD is weighted by the allele frequency. For instance, the probability to observe an homozygous AA is ${f_A}^2$ in non-HBD segments and approximately f_A for HBD segments. So, the emission probability in non-HBD state is approximately equal to the emission probability in HBD state multiplied by f_A. If f_A is high, the probabilities are close with little support in favor of HBD whereas for low f_A, the support for HBD is higher.

Overall, the HMM approach is not to sensitive to filtering on MAF. We compared the filtering rules on the real data sets used in Druet and Gautier (2017) and found little differences in setting `min_maf` to 0, 0.01 or 0.05.

Similarly, our multiple HBD class model is quite robust to LD structure (or absence of LD filtering). With one HBD class HMM, it is sometimes necessary to perform LD pruning to capture only the long HBD fragments and not the small HBD fragments associated to background LD. For instance, Leutenegger et al. (2011) selected subsets of 1% of their data. With our model, the long HBD segments and the small HBD segments go into distinct classes. Applying our model without data filtering, we obtained estimates similar to Leutenegger et al. (2011) in the recent HBD classes whereas additional autozygosity was associated to more ancient HBD classes. See also section 3.2 (and Figure 2) and Druet and Gautier (2017) for more details.

Our models achieved also good performances without LD pruning on data simulated with population genetics models, see Druet and Gautier (2017) for more details.

5 Defining the model

With `RZooRoH` you can define different models. The most important parameters are the number of classes K and the rates R_k for each class. These rates can either be fixed by the user ("pre-defined") or estimated by `RZooRoH` for every individual. The optimal model to select depends on the objectives of the analysis and on the data set.

5.1 The number of non-HBD classes is limited to one

In the `RZooRoH` package we use only one non-HBD class although a model with several non-HBD classes would be possible. There are three main reasons for this choice:

- The non-HBD class can be viewed as segments with very old common ancestors, old enough to present variation. Therefore, their length is expected to be shorter than the smallest HBD segments. It make then sense to define one unique class with non-HBD segment (a class with the eldest common ancestors).
- Our main interest is to identify HBD segment and their length. We want to know whether a position is HBD and how long is the surrounding segment. We don't want long HBD segments to be cut into a succession of small HBD segments. Therefore, we need to model several HBD classes. For non-HBD segments, we are less interest to partition them in classes of different lengths (non-HBD segments can be modeled has a succession of many small HBD segments). Therefore, we don't need to define multiple non-HBD classes.

- Using multiple non-HBD classes would increase the computational burden (without benefit).

Although the original FORTRAN implementation of `ZooRoH` included the possibility to model multiple non-HBD classes, we decided to model only one non-HBD class in the `RZooRoH` package to keep things simple.

5.2 Models with one HBD class and one non-HBD class: `1R` model

The one HBD class HMM are a special case of the models for which the rates R_k are estimated. In that case, the same rate is used for HBD and non-HBD classes as in Leutenegger et al. (2003), Narasimhan et al. (2016) or Vieira et al. (2016). We recommend to us such a model only when all the autozygosity or HBD tracks trace back to one ancestor or several ancestors living in the same generation. That model might also require some LD pruning prior to analysis.

5.3 Models without pre-defined rates: `KR` models (`RZooRoh` will estimate the rates R_k)

If the goal is to identify all HBD segments that can be captured with the marker density and to estimate the total autozygosity, then a model with a few HBD classes would be indicated. In that case, the optimal number of classes is a function of the marker density. For instance, Solé et al. (2017) showed with cattle data that for low-density arrays only one or two HBD classes are needed whereas for sequencing data, three or four HBD classes were recommended. The BIC can be used to compare models with different number of classes K and to determine the optimal K for each individual (Druet and Gautier, 2017).

This strategy with an optimized number of classes would also be recommended to perform a length-based classification of HBD segments in main categories (e.g. long segments associated with recent relatedness, intermediate segments and short segments associated with LD patterns) as in Pemberton et al. (2012). The main categories make the LD pruning unnecessary.

Optimizing K is also more efficient computationally although convergence is more difficult with R_k rates estimation. The use of the BIC criteria to select K combined with the estimate of the R_k also reduces the risk to fit an inappropriate model and miss some autozygosity (because classes are too distant or do not cover the whole range of segments that can be captured with the current marker density).

However, this strategy is not recommended if the user wants to determine which generations of ancestors contribute to present autozygosity (e.g. to identify past bottlenecks, identify offspring from closely related individuals). In addition, such a strategy with estimation of R_k rates makes comparisons across individuals uneasy because different individuals will have different R_k. Their autozygosity will therefore be partitioned in different HBD classes and eventually estimated with respect to different base populations.

5.4 Models with pre-defined rates: `MixKR` models

In a so-called `MixKR` model, the R_k rates are pre-defined by the user and no longer estimated. Then, `RZooRoH` will only estimate the mixing coefficients that will influence the number of segments in each class.

5.4.1 Selecting the number of classes and their rates

The user should select a set of classes that cover the whole range of HBD segments that can be captured with the genotyping array:

- The smallest rate (most recent class) should not be smaller than 1. A value of two corresponds approximately to a common ancestor present one generation ago as with selfing. So, values lower than two should ideally not be used.
- The highest rate depends on the marker density. For instance, there is not enough data to capture HBD segments smaller than the average marker spacing. With 10 SNPs per cM, the average spacing would be 0.1 cM corresponding to the average segment length with a R_k rate of 1,000. Therefore, there is no

need to put classes with rates higher than 1,000. Previous analyses with simulated and real data sets (Druet and Gautier, 2017; Solé et al., 2017) suggest that setting a few classes with too high R_k doesn't affect the result because these classes are not used (they stay empty) although computational costs are increased. On the other hand, setting to few classes (to small R_k for the last class) is not optimal because the last class will capture its HBD segments and those from the unmodeled older classes with a higher rate.

- Finally, the number of classes K should allow to cover the range from the first to the last rate. With too many classes, the computational burden will dramatically increase and there is probably not enough information to distinguish segments for classes with very similar rates. We like to use series of rates with constant ratio between successive rates (expected lengths are divided by a constant value from one class to the next). We recommended to use a ratio of two or higher to limit the overlap between exponential distributions. For instance the power of two [2, 4, 8, 16, 32, 64, ..., 512, 1024, 2048, 4096] allows to have more classes for long HBD segments that for shorter segments. There is more information to distinguish two exponential distributions with rates 2 and 4 (expected length 25 and 50 cM) than with rates 998 and 1000 (expected length 0.1002 and 0.1000 cM).
- Other series such as the power of five [5, 25, 125, 525, 2625, ...] or ten [10, 100, 1000, 10000] can cover the same range with less classes (reducing the computational cost). However, some resolution is lost because classes regroup more generations of ancestors. For instance, rates of 4, 8, 16 would result in categories for grand-parents, ancestors present 4 or 8 generations ago whereas rates of 5, 25 and 125 would result in categories for ancestors around 2.5 generations ago, 12.5 generations ago and more than 60 generations. With [10, 100, 1000, ...], the resolution would be even poorer with one group for very recent ancestors (centered around 5 generations ago) and one group for more ancients ancestors (centered around 50 generations ago).

The rules and models proposed above should limit the risk to miss autozygosity because the first rate is to high, the last rate is too low or two rates are too distant from each other.

A more complex strategy to select the pre-defined HBD classes would be to first run `KR` models and select the optimal number of classes K with the BIC. Then, we could select as rates R_k for the `MixKR` model the median values of R_k estimated with the optimal `KR` model (the median of R_1's for R_1, the median of R_2's for R_2, etc.). With that strategy, we would obtain a parsimonious `MixKR` model but the model selection step would be time consuming. It would also limit the risk to miss some autozygosity because the model is inappropriate.

5.4.2 Benefits of using pre-defined rates

Compared to `KR` models (with estimation of the rates), `MixKR` models:

- Allow an easier comparison across individuals because they all have the same R_k and the same HBD classes. It seems to be better to use the same set of classes for all individuals.
- Offer more resolution to determine which ancestors contribute to the autozygosity of individuals because most often models with pre-defined rates have more classes. With more classes, we can better identify offspring from closely related parents (with a common ancestor in very recent past) and obtain more precise estimates of demographic events such as bottleneck or founder effects.
- Allow to estimate inbreeding coefficients with respect to the same base population for all individuals (since the HBD classes are the same).
- Fit as well the data as models selected with BIC (Druet and Gautier, 2017).
- Converge often better.
- Have higher computational costs (large models than run longer and require more memory).

Although computationally more intensive, we recommend the use of pre-defined models for these reasons and set "pre-defined" as default value for the `zoomodel` function.

5.5 Using `zoomodel` to define your model

5.5.1 Options

The `zoomodel` function creates a `zmodel` object needed to run our model. The function allows to define whether the R_k rates are pre-defined (default) or not with the `predefined` option, the number of class K (option $K = \dots$). The values of the R_k can either be defined with the `base_rate` option (default = 2). Then, the successive rates will be equal to b^k where b is the selected base rate and k the class number. Alternatively, the `krates` option can be used to specify the K rates. The `mix_coef` determines the starting values for the mixing coefficients. In addition to these parameters, the user can provide the genotyping ϵ with option `err` (the probability to observe a heterozygous genotype in an HBD class) or sequencing error rates (the probability to have a sequencing error in one read, for the "ad" format) with option `seqerr`.

5.5.2 Output: the `zmodel` object

The returned `zmodel` object contains five slots: the zmodel@typeModel can be "mixkr" (for predefined = TRUE) or "kr" (for predefined = FALSE), the zmodel@mix_coef contains the starting values for the mixing coefficients of each class, the zmodel@krates contains the (starting) values for the R_k rates, the zmodel@err contains the genotyping error rate and the zmodel@seqerr contains the sequencing error rate.

5.5.3 Some examples of model definition with `zoomodel`

To define a default model, with 10 classes (9 HBD and 1 non-HBD class) with a ratio between successive rates of two:

```
mix10R <- zoomodel()
mix10R
#> An object of class "zmodel"
#> Slot "typeModel":
#> [1] "mixkr"
#>
#> Slot "mix_coef":
#>  [1] 0.01 0.01 0.01 0.01 0.01 0.01 0.01 0.01 0.01 0.91
#>
#> Slot "krates":
#>  [1]   2   4   8  16  32  64 128 256 512 512
#>
#> Slot "err":
#> [1] 0.001
#>
#> Slot "seqerr":
#> [1] 0.001
```

To access a specific slot use the @ symbol:

```
mix10R@krates
#>  [1]   2   4   8  16  32  64 128 256 512 512
```

To define a model with pre-defined rates for 5 classes (4 HBD and 1 non-HBD class) with a ratio between successive rates of ten:

```
mix5R <- zoomodel(K=5,base=10)
mix5R
```

```
#> An object of class "zmodel"
#> Slot "typeModel":
#> [1] "mixkr"
#>
#> Slot "mix_coef":
#> [1] 0.01 0.01 0.01 0.01 0.96
#>
#> Slot "krates":
#> [1]    10   100  1000 10000 10000
#>
#> Slot "err":
#> [1] 0.001
#>
#> Slot "seqerr":
#> [1] 0.001
```

To define the same model but setting genotyping error rates to 0.01 and sequencing error rates to 0.005:

```
mix5R <- zoomodel(K=5,base=10,err=0.01,seqerr=0.005)
mix5R@err
#> [1] 0.01
mix5R@seqerr
#> [1] 0.005
```

To define a model with four classes, with estimation of the rates and specifying the initial rates to 16, 64, 256 and 256:

```
my.mod4R <- zoomodel(predefined=FALSE,K=4,krates=c(16,64,256,256))
my.mod4R
#> An object of class "zmodel"
#> Slot "typeModel":
#> [1] "kr"
#>
#> Slot "mix_coef":
#> [1] 0.01 0.01 0.01 0.97
#>
#> Slot "krates":
#> [1]  16  64 256 256
#>
#> Slot "err":
#> [1] 0.001
#>
#> Slot "seqerr":
#> [1] 0.001
```

To change the starting values of the mixing coefficients:

```
my.mod4R <- zoomodel(predefined=FALSE,K=4,krates=c(16,64,256,256),
                     mix_coef=c(0.03,0.04,0.13,0.80))
my.mod4R@mix_coef
#> [1] 0.03 0.04 0.13 0.80
```

Finally, to define a model with one HBD-class and common rate for the HBD and non-HBD classes (similar

to the original model from Leutenegger et al. (2003)):

```
my.mod1R <- zoomodel(predefined=FALSE,K=2,krates=c(10,10))
my.mod1R
#> An object of class "zmodel"
#> Slot "typeModel":
#> [1] "kr"
#>
#> Slot "mix_coef":
#> [1] 0.01 0.99
#>
#> Slot "krates":
#> [1] 10 10
#>
#> Slot "err":
#> [1] 0.001
#>
#> Slot "seqerr":
#> [1] 0.001
```

6 Running `zoorun`

The `zoorun` function allows to estimate the parameters of the model, to estimate the global and locus specific realized autozygosity, to partition it in the different HBD classes and to identify the HBD segments.

6.1 General options

The `zoorun` requires two essential elements, the `zmodel` and `zdata` objects that provide respectively the model and the data. These objects are obtained by running prior to the analysis the `zoomodel` and `zoodata` functions as described in sections 4 and 5. By default, the analysis will be performed on all individuals but the user can provide a list of individuals with the optional `ids` option (see examples below). The `ids` are the position of the individuals in the data file (columns, 1 for the first individual, 2 for the second, etc).

In addition, the analysis can be run in parallel by specifying the number of threads with the `nT` option (equal to 1 by default).

6.2 Parameter estimation

6.2.1 Methods for parameter estimation

The parameter estimation is required prior to estimate the realized autozygosity or to identify the HBD segments. Therefore, parameter estimation is set to true by default. When the parameters have been estimated in a previous run, their value can be indicated in the `zmodel` object but this has to be done for each individual separately. This is one of the rare situations where the parameter estimation can be skipped.

Originally, the parameter estimation, for the mixing coefficients M_k and eventually the rates R_k, was performed with the EM algorithm as described in Druet and Gautier (2017). In `RZooRoH`, the estimation of parameters is performed with optimization procedures implemented in the `optim` R package, with the L-BFGS-B method. This procedure can converge is less iterations (with models with few parameters) and at a lower computational cost (one iteration to obtain the likelihood required by `optim` needs to compute only the forward variables whereas the EM algorithm requires both forward and backward variables).

In order to work with unconstrained parameters and to obtain ordered HBD classes (with increasing rates of exponential distributions), we defined new parameters (Zucchini and MacDonald, 2009):

$$\eta_k = \begin{cases} log(R_k - R_{k-1}) & \text{if } 1 < \text{k} < \text{K} \\ log(R_k) & \text{if k} = 1 \text{ or k} = \text{K} \end{cases} \quad (1)$$

$$\tau_k = log(\frac{M_k}{M_K}) \qquad \text{if k} < \text{K} \quad (2)$$

Using the back-transformation, it can be verified that the rates R_k from HBD classes are always positive and ordered, and that the mixing coefficient M_k sum to 1 and are constrained between 0 and 1.

6.2.2 Options for parameter estimation

The method for parameter estimation is selected with the `method` option that can be equal to "opti" (for the new optimization procedure), "estem" for the EM-algorithm or "no" for skipping parameter estimation. By default the value is set to "opti".

It is possible to impose some constraints on the R_k rate parameters (the M_k parameters don't need constraints). With the new optimization procedure, the optional `minr` and `maxr` represent the minimum and the maximum differences between successive rates. We don't recommend to set constraints when method is "opti". With the EM algorithm, the options have different interpretation and represent the minimum and maximum rates. We recommend to set `minr` above 1 (and we force it to 1 when `minr` is smaller than 1).

Two other optional arguments are related with the EM algorithm: the maximum number of iterations (`maxiter` set to 1000 by default) and the convergence criteria (`convem` set to 1e-10 by default).

When `optim` fails to converge, `zoorun` estimates the parameters automatically with the EM algorithm for the concerned individual.

6.2.3 Examples

We start by loading a data set with six individuals from a cattle population genotyped for a low-density array. An additional file with the allele frequencies estimated on a larger sample is also available (this provides an illustration on how to use the `allelefreq` option).

```
freqfile <- (system.file("exdata","typsfrq.txt",package="RZooRoH"))
typfile <- (system.file("exdata","typs.txt",package="RZooRoH"))
frq <- read.table(freqfile,header=FALSE)
bbb <- zoodata(typfile,supcol=4,chrcol=1,poscol=2,allelefreq=frq$V1)
#> [1] "Number of positions in original file ::"
#> [2] "6370"
#> [1] "Number of positions after MAF filtering  ::"
#> [2] "6370"
```

We will then estimate parameters with the my.mod1R model defined in section 5.5.3. We start with default options for the first individual.

```
bbb_results <- zoorun(my.mod1R, bbb, ids = c(1))
#> final  value 6127.228984
#> converged
```

We repeat parameter estimation with the EM algorithm setting the minimum rate to 2 and the maximum rate to 100. We change the maximum number of iterations to 100 and set the convergence criteria to 1e-12.

```
bbb_results2 <- zoorun(my.mod1R, bbb, method="estem",ids = c(1,2), convem=1e-12,
                       maxiter=100, minr=2, maxr=100)
#> [1] "EM algorithm, iterations and loglik :: 101 -6128.59636506626"
#> [1] "EM algorithm, iterations and loglik :: 101 -5767.65950055702"
```

6.3 Estimating realized autozygosity (with partitioning in different HBD classes)

To estimate the realized autozygosity in each HBD class (the percentage of the genome from an individual associated with a specific HBD class), one has simply to set the option `fb` (for Forward-Backward algorithm) to true (the default value). These values are indeed estimated with the Forward-Backward algorithm (see section 2.3) that allows to compute at each marker position the probability to belong to each of the defined classes in the model. For instance, the realized autozygosity has been automatically estimated in the two previous examples (in the 6.2.3 section).

The realized autozygosity in each HBD class is equal to the genome-wide average of the probabilities to belong to each HBD class computed at each marker position. These locus specific HBD probabilities can be obtained by setting the `localhbdp` option to TRUE. By default this value is set to FALSE because it generates large outputs (number of SNPs x number of classes x number of individuals). These values would be interesting for specific applications such as autozygosity mapping experiments.

6.4 Identifying HBD segments

To identify HBD segments we rely on the Viterbi algorithm (see section 2.3). The Viterbi algorithm identifies the most likely sequence of HBD and non-HBD classes along the genome. At each marker, the genome is assigned to one class and HBD segments can be identified as stretches of consecutive markers assigned to the same HBD class. To use the Viterbi algorithm, one has simply to set the `vit` option to TRUE (the default value). As before, the HBD segments were automatically identified in the two previous examples (in the 6.2.3 section).

Note that we prefer to work with HBD probabilities and the forward-backward algorithm than with the HBD segments and the Viterbi algorithm. We do not recommend to estimate the realized autozygosity as the sum of the length of HBD segments divided by the length of the genome (these estimators were found to be less accurate than those obtained with the Forward-Backward algorithm). Similarly, for autozygosity mapping experiments, we recommend to work with local HBD probabilities to take uncertainty into account. The HBD segments should be used only in applications requiring HBD segments or for visualization purposes.

6.5 More examples

To obtain the locus specific HBD probabilities for all individuals with the previous example.

```
bbb_results3 <- zoorun(my.mod1R, bbb, localhbd = TRUE)
#> final  value 6127.228984
#> converged
#> final  value 5767.659061
#> converged
#> final  value 5763.247991
#> converged
#> final  value 5809.962884
```

```
#> converged
#> final  value 5758.887693
#> converged
#> final  value 5383.740551
#> converged
```

To run a model with two HBD classes with pre-defined rates equal to 10 and 100 on the same data set.

```
Mod3R <- zoomodel(K=3,base_rate=10)
bbb_mod3r <- zoorun(Mod3R, bbb, localhbd = TRUE)
#> final  value 6128.132841
#> converged
#> final  value 5768.035614
#> converged
#> final  value 5762.587394
#> converged
#> final  value 5811.283672
#> converged
#> final  value 5760.370243
#> converged
#> final  value 5387.766858
#> converged
```

7 Description of results

Results are grouped in a zres object with 12 slots. These slots can be accessed with the @ symbol. Some slots describe the samples in the analysis:

- ..@nind is simply the number of analyzed individuals in the run;
- ..@ids is an array with the numbers of the analyzed individuals;
- ..@sampleids is an array with the name of the samples (if they were provided) or with their number;
- ..@optimerr is an array indicating whether `optim` ran without error (for each individual).

For instance, for the analysis with two individuals:

```
bbb_results2@nind
#> [1] 2
bbb_results2@ids
#> [1] 1 2
```

7.1 Parameters (likelihood and convergence)

Other slots of the zres object contain information on the parameter estimation: the estimated parameters, the log(likelihood) of the model, the BIC at convergence and the number of iterations:

- ..@mixc is a matrix with *nind* columns (the number of individuals) and K rows containing the estimated mixing coefficients M_k for each individual (in rows) and each class (in columns);
- ..@krates is a matrix with *nind* columns (the number of individuals) and K rows containing the (estimated) R_k for each individual (in rows) and each class (in columns);
- ..@modlik is an array with the estimated log(likelihood) of the model at convergence for each individual;
- ..@modbic is an array with the estimated BIC of the model at convergence for each individual;
- ..@niter is an array with the number of iterations for parameter estimation for each individual.

Interpretation of the parameters is described in section 2.4.

To obtain the mixing coefficients M_k with the model with three classes and pre-defined rates:

```
bbb_mod3r@mixc
#>              [,1]         [,2]      [,3]
#> [1,] 6.157226e-05 5.962948e-02 0.9403089
#> [2,] 5.067649e-03 4.718697e-03 0.9902137
#> [3,] 4.950441e-03 2.498444e-02 0.9700651
#> [4,] 4.392999e-03 9.436872e-03 0.9861701
#> [5,] 2.924898e-03 2.580824e-02 0.9712669
#> [6,] 1.427542e-02 5.240913e-07 0.9857241
```

Since rates R_k were pre-defined, they are all the same and correspond to the values specified by the model:

```
bbb_mod3r@krates
#>      [,1] [,2] [,3]
#> [1,]   10  100  100
#> [2,]   10  100  100
#> [3,]   10  100  100
#> [4,]   10  100  100
#> [5,]   10  100  100
#> [6,]   10  100  100
```

With the first model, the R_k vary across individuals (note that with one HBD class the same rate is used for the HBD and the non-HBD class as in Leutenegger et al. (2003)):

```
bbb_results3@krates
#>        [,1]   [,2]
#> [1,] 73.280 73.280
#> [2,]  6.218  6.218
#> [3,] 24.439 24.439
#> [4,] 24.021 24.021
#> [5,] 35.496 35.496
#> [6,] 10.420 10.420
```

We can extract the log(likelihood) or the BIC as follows:

```
cbind(bbb_results3@modlik,bbb_results3@modbic)
#>           [,1]     [,2]
#> [1,] -6127.229 12271.98
#> [2,] -5767.659 11552.84
#> [3,] -5763.248 11544.01
#> [4,] -5809.963 11637.44
#> [5,] -5758.888 11535.29
#> [6,] -5383.741 10785.00
```

These likelihoods or BIC are most useful to compare models or parameter estimation procedures.

7.2 Realized autozygosity per class

The realized autozygosity per HBD class, or the partitioning of the genome is different classes is one of the main outputs of the model. It is returned in the ..@realized slot as a matrix with K columns (the number of

classes) and *nani* lines (one per individual in the analysis). The values [i,j] are the genome-wide contributions (averaged over all positions) of class j to the genome of individual i.

```
bbb_mod3r@realized
#>              [,1]         [,2]      [,3]
#> [1,] 0.0002451684 6.283497e-02 0.9369199
#> [2,] 0.0429870128 4.589439e-03 0.9524235
#> [3,] 0.0276528577 2.482290e-02 0.9475242
#> [4,] 0.0228609411 1.137513e-02 0.9657639
#> [5,] 0.0133014561 2.845040e-02 0.9582481
#> [6,] 0.1003679577 5.268664e-07 0.8996315
```

The first individual has 0.0245% of its genome in the first HBD class ($R_k = 10$), 6.2835% in the second HBD class ($R_k = 100$) and 93.6920% in the non HBD-class ($R_k = 100$). The second individual has more autozygosity in the first class (0.042987 or 4.2987%) and the sixth individual has the highest level in the first class (10.0368%).

With the one-HBD class model, there are only tow columns representing autozygosity versus allozygosity.

```
bbb_results3@realized
#>            [,1]      [,2]
#> [1,] 0.05852398 0.9414760
#> [2,] 0.04253227 0.9574677
#> [3,] 0.04469971 0.9553003
#> [4,] 0.03333425 0.9666657
#> [5,] 0.03563496 0.9643650
#> [6,] 0.10070006 0.8992999
```

In the case of a one-HBD class model, the mixing coefficients M_k and the realized autozygosity are more related but not identical. The first representing the proportion of segments that are HBD and the second the proportion of loci that are HBD.

```
cbind(bbb_results3@realized,bbb_results3@mixc)
#>            [,1]      [,2]       [,3]      [,4]
#> [1,] 0.05852398 0.9414760 0.05798497 0.9420150
#> [2,] 0.04253227 0.9574677 0.04705895 0.9529411
#> [3,] 0.04469971 0.9553003 0.04473379 0.9552662
#> [4,] 0.03333425 0.9666657 0.03031456 0.9696854
#> [5,] 0.03563496 0.9643650 0.03517359 0.9648264
#> [6,] 0.10070006 0.8992999 0.09474106 0.9052589
```

A zres object obtained by running a model with more classes on 110 individuals from the Soay population genotyped for 47,365 SNPs can be loaded with the package. The default model with 10 classes was used to perform the analysis:

```
mix10r <- zoomodel()
mix10r
#> An object of class "zmodel"
#> Slot "typeModel":
#> [1] "mixkr"
#>
#> Slot "mix_coef":
#>  [1] 0.01 0.01 0.01 0.01 0.01 0.01 0.01 0.01 0.01 0.91
```

```
#>
#> Slot "krates":
#>  [1]   2   4   8  16  32  64 128 256 512 512
#>
#> Slot "err":
#> [1] 0.001
#>
#> Slot "seqerr":
#> [1] 0.001
```

The results are stored in the soay_mix10r zres object. To obtain the realized autozygosity for the 20 first individuals:

```
round(soay_mix10r@realized[1:20,],3)
#>       [,1]  [,2]  [,3]  [,4]  [,5]  [,6] [,7] [,8] [,9] [,10]
#>  [1,]    0 0.059 0.000 0.000 0.071 0.117    0    0    0 0.753
#>  [2,]    0 0.000 0.000 0.000 0.083 0.099    0    0    0 0.817
#>  [3,]    0 0.000 0.000 0.000 0.128 0.106    0    0    0 0.766
#>  [4,]    0 0.000 0.000 0.020 0.096 0.101    0    0    0 0.783
#>  [5,]    0 0.000 0.000 0.017 0.107 0.090    0    0    0 0.786
#>  [6,]    0 0.000 0.000 0.000 0.165 0.072    0    0    0 0.763
#>  [7,]    0 0.000 0.000 0.000 0.135 0.092    0    0    0 0.773
#>  [8,]    0 0.000 0.000 0.025 0.093 0.098    0    0    0 0.784
#>  [9,]    0 0.000 0.001 0.059 0.051 0.124    0    0    0 0.766
#> [10,]    0 0.000 0.000 0.057 0.066 0.111    0    0    0 0.767
#> [11,]    0 0.000 0.000 0.044 0.000 0.160    0    0    0 0.796
#> [12,]    0 0.000 0.000 0.000 0.116 0.100    0    0    0 0.783
#> [13,]    0 0.000 0.000 0.000 0.168 0.070    0    0    0 0.762
#> [14,]    0 0.000 0.000 0.000 0.162 0.073    0    0    0 0.766
#> [15,]    0 0.000 0.000 0.001 0.135 0.098    0    0    0 0.767
#> [16,]    0 0.000 0.000 0.022 0.173 0.050    0    0    0 0.754
#> [17,]    0 0.000 0.000 0.000 0.163 0.082    0    0    0 0.756
#> [18,]    0 0.000 0.000 0.000 0.113 0.116    0    0    0 0.771
#> [19,]    0 0.000 0.000 0.000 0.083 0.102    0    0    0 0.815
#> [20,]    0 0.000 0.000 0.001 0.107 0.099    0    0    0 0.794
```

The columns represent the probability to belong to each class (averaged genome-wide). The columns represents the HBD classes with rates R_k equal to 2, 4, 8, 16, 32, 64, 128, 256, 512 and the non-HBD class (last column). The autozygosity is higher than 0.20 for most individuals (the non-HBD class accounts for less than 0.80) and is concentrated in HBD classes 5 and 6 (with rates 32 and 64).

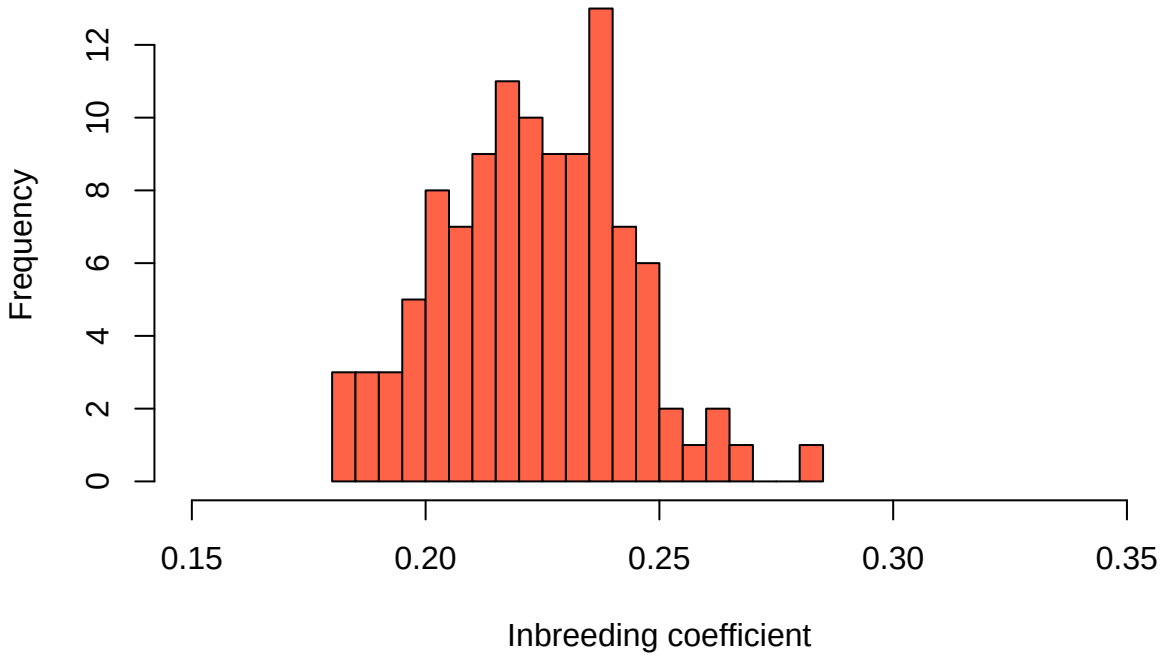
7.3 Defining inbreeding coefficients (with respect to a base population)

The objective is not always to obtain the partitioning of autozygosity in different HBD classes (contribution from ancestors tracing back to different generations in the past). We are sometimes interested in the inbreeding coefficient (or the overall autozygosity). To estimate an inbreeding coefficient, we must define a base population. Segments inherited from ancestors present in generations more remote than the reference population will no longer be consider autozygous. With our model, we can decide that all classes with a rate larger than a threshold T are not consider as autozygous. This amounts to set the base population at approximately 0.5*T generations in the past and to estimate an inbreeding coefficient F_{G-T}, for instance F_{G-50} would include all HBD classes with $R_k \leq 50$.

To obtain an inbreeding coefficient including all HBD classes, autozygosity must be summed over all HBD

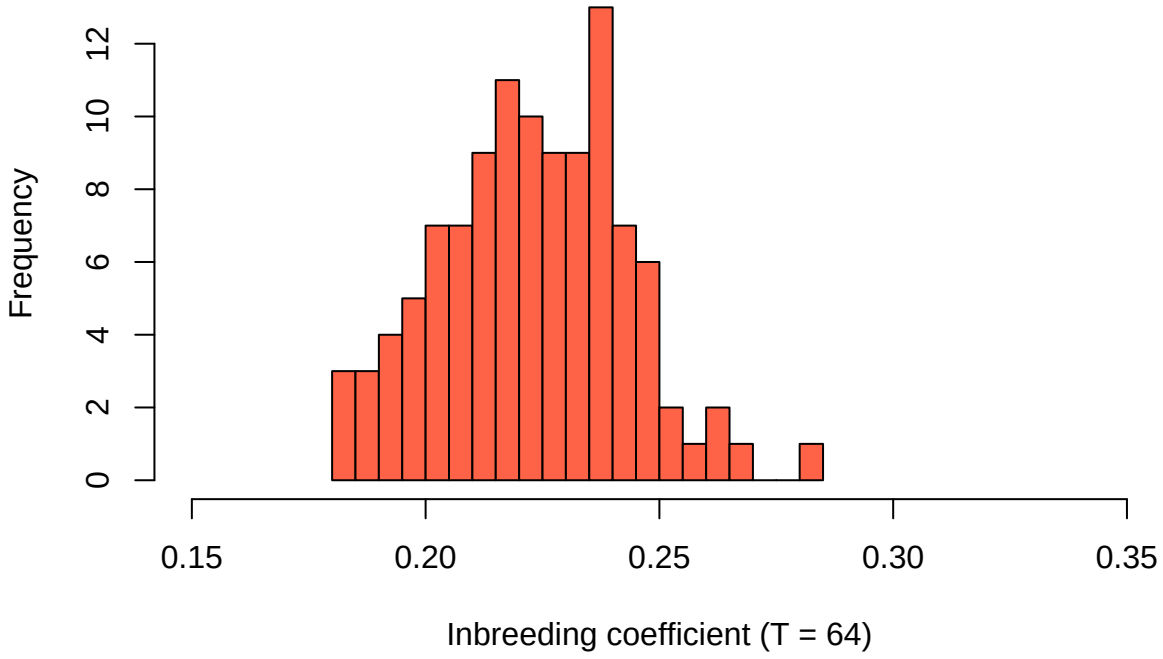
classes or can be obtained as 1 minus the non-HBD proportion.

```
x <- 1-soay_mix10r@realized[,10]
hist(x,nc=20,main="",xlab="Inbreeding coefficient",xlim=c(0.15,0.35),col='tomato')
```



To obtain an inbreeding coefficient with respect to a threshold T, for instance at 64, autozygosity must be summed over the corresponding classes, with the cumsum function (or other functions).

```
x <- t(apply(soay_mix10r@realized[,1:6],1,cumsum))
hist(x[,6],nc=20,main="",xlab="Inbreeding coefficient (T = 64)",
     xlim=c(0.15,0.35),col='tomato')
```



7.4 Local HBD probabilities

The locus specific HBD probabilities (probabilities to belong to a state at a marker position) are stored in the ..@hbdp slot. This slot is a list of matrices, one matrix per individual in the analysis. Each matrix has a dimension K x $nsnp$ (the number of classes x number of SNPs). Element[i,j] represent the probability to be in class i at marker j. To access the matrix of one individual in the list, you must use "[[x]]". For instance, to extract HBD probabilities for the first 15 markers for individual 3:

```
t(bbb_mod3r@hbdp[[3]][,1:15])
#>               [,1]         [,2]      [,3]
#>  [1,] 7.666530e-07 3.324423e-05 0.9999660
#>  [2,] 1.418597e-05 6.511638e-04 0.9993347
#>  [3,] 1.024341e-05 4.691709e-04 0.9995206
#>  [4,] 3.109336e-08 1.211845e-06 0.9999988
#>  [5,] 3.935139e-07 1.114691e-05 0.9999885
#>  [6,] 1.176953e-04 3.479645e-03 0.9964027
#>  [7,] 5.911055e-04 2.068023e-02 0.9787287
#>  [8,] 6.582677e-04 2.314747e-02 0.9761943
#>  [9,] 7.519357e-04 2.678557e-02 0.9724625
#> [10,] 8.128736e-04 2.928319e-02 0.9699039
#> [11,] 8.440785e-04 3.087533e-02 0.9682806
#> [12,] 6.402972e-07 2.057224e-05 0.9999788
#> [13,] 1.076755e-06 4.552411e-05 0.9999534
#> [14,] 1.531827e-08 6.627845e-07 0.9999993
```

```
#> [15,] 4.075094e-08 1.845258e-06 0.9999981
```

We obtain three columns, the probabilities for HBD class 1 ($R_k = 10$), for HBD class 2 ($R_k = 100$) and the non-HBD class. The HBD probabilities remain below 0.01 for most markers. An example of HBD segments (with high HBD probability) is found for individual 6:

```
t(bbb_mod3r@hbdp[[6]][,4700:4720])
#>             [,1]         [,2]         [,3]
#>  [1,] 0.9996134 2.823171e-09 0.0003866408
#>  [2,] 0.9996514 2.784829e-09 0.0003485868
#>  [3,] 0.9996540 2.694001e-09 0.0003460391
#>  [4,] 0.9996302 2.648607e-09 0.0003697920
#>  [5,] 0.9990251 2.254577e-09 0.0009749052
#>  [6,] 0.9996994 2.757016e-09 0.0003006117
#>  [7,] 0.9997337 2.756408e-09 0.0002663251
#>  [8,] 0.9997022 2.528708e-09 0.0002977747
#>  [9,] 0.9997316 2.443144e-09 0.0002683974
#> [10,] 0.9996904 2.331482e-09 0.0003095887
#> [11,] 0.9997023 2.311485e-09 0.0002977464
#> [12,] 0.9996111 2.305251e-09 0.0003888756
#> [13,] 0.9996657 2.345161e-09 0.0003343234
#> [14,] 0.9996564 2.344858e-09 0.0003436412
#> [15,] 0.9996438 2.335597e-09 0.0003561949
#> [16,] 0.9994132 2.401454e-09 0.0005868349
#> [17,] 0.9994442 2.589676e-09 0.0005557811
#> [18,] 0.9994122 2.578863e-09 0.0005877946
#> [19,] 0.9989891 2.538244e-09 0.0010109419
#> [20,] 0.9988259 2.506320e-09 0.0011740477
#> [21,] 0.9987728 2.432523e-09 0.0012272289
```

7.5 HBD segments

The HBD segments identified with the Viterbi algorithm are stored in the ..@hbdseg, a data frame with one line per identified HBD segment and with nine columns accessed with the $ symbol:

- ..@hbdseg$id is the number of the individual in which the HBD segments is located;
- ..@hbdseg$chrom is the chromosome number of the HBD segments (this chromosome number refers to the position of the chromosome in the list of all chromosomes present in the input genotype data);
- ..@hbdseg$start_snp is the number of the SNP at which the HBD segment starts (the SNP number within the chromosome);
- ..@hbdseg$start_end is the number of the SNP at which the HBD segment ends (the SNP number within the chromosome);
- ..@hbdseg$start_pos is the position at which the HBD segment starts (within the chromosome);
- ..@hbdseg$start_end is the position at which the HBD segment ends (within the chromosome);
- ..@hbdseg$number_snp is the number of consecutive SNPs in the HBD segment;
- ..@hbdseg$length is the length of the HBD segment (for instance in bp or in cM/1000000);
- ..@hbdseg$HBDclass is the HBD class associated with the HBD segment.

The table of HBD segments identified in the Belgian Blue cattle subset with the `mod3r` model is:

```
dim(bbb_mod3r@hbdseg)[1]
#> [1] 45
head(bbb_mod3r@hbdseg[,1:8])
```

```
#>    id chrom start_snp end_snp start_pos   end_pos number_snp  length
#> 2   1     1         7      19    845494   2049400         13 1203907
#> 21  1     4       230     252 101677815 111368926         23 9691112
#> 1   1     6         1       9    202769   1014246          9  811478
#> 22  1     8       112     117  47179793  49524000          6 2344208
#> 23  1    10        47      50  13909586  14200836          4  291251
#> 24  1    11       204     210  94101466  95637394          7 1535929
```

There are 45 identified HBD segments (approximately 7 per individual). The first and second HBD segments are respectively 1.2 and 9.7 MB long and contain 13 and 23 SNPs. They belong both to the first individual and are located at the beginning of chromosome 1 and the end of chromosome 4.

To have summary statistics of length of HBD segments (in bp or in number of SNPs):

```
summary(bbb_mod3r@hbdseg$length)
#>     Min.  1st Qu.   Median     Mean  3rd Qu.     Max.
#>   291251  2782886  9000333 10866304 12141604 48659487
summary(bbb_mod3r@hbdseg$number_snp)
#>    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
#>    4.00   16.00   23.00   30.47   34.00  145.00
```

The largest HBD segment is more than 48 Mb long, the average length is 10.87 Mb and the mean number of SNP per identified HBD segment is 30.47.

In the sheep population genotyped at higher density, the distribution of HBD segment presents some differences.

```
dim(soay_mix10r@hbdseg)[1]
#> [1] 14547
head(soay_mix10r@hbdseg[,1:8])
#>    id chrom start_snp end_snp start_pos   end_pos number_snp  length
#> 2   1     1       314     369  15944770  18966270         56 3021501
#> 4   1     1       390     417  19945652  22120789         28 2175138
#> 6   1     1       755     906  40342997  48373831        152 8030835
#> 8   1     1       945     963  50614635  51923225         19 1308591
#> 10  1     1      1465    1512  80119112  82137439         48 2018328
#> 12  1     1      1969    2038 107797839 111474679         70 3676841
```

There are 14,547 identified HBD segments (approximately 132 per individual). The first and second HBD segments are respectively 3.0 and 2.2 MB long and contain 56 and 28 SNPs. They belong both to the first individual and are located on chromosome 1. The first individual has several segments on the first chromosome.

To obtain summary statistics of length of HBD segments (in bp or in number of SNPs):

```
summary(soay_mix10r@hbdseg$length)
#>     Min.  1st Qu.   Median     Mean  3rd Qu.     Max.
#>   116689  1756868  2811065  3790554  4620455 73719189
summary(soay_mix10r@hbdseg$number_snp)
#>    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
#>    5.00   35.00   53.00   71.08   86.00 1402.00
```

The largest HBD segment is more than 73 Mb long, the average length is 3.79 Mb and the mean number of SNP per identified HBD segment is 71.08.

7.6 Accessor functions (easier access to slots)

We propose four accessor functions to interact more conveniently with the zres object, in particular with the most useful slots (..@realized, ..@hbdp and ..@hbdseg).

7.6.1 For realized autozygosity

To extract the table of realized autozygosity, you can use the function `realized` that takes as argument the name of the zres object and the numbers of the classes to be extracted (all by default). The function returns a data frame with one row per individual and one column per extracted classes. In addition, it gives names to the columns. For a pre-defined model, the names of HBD classes are "R_X" where X is the value of the corresponding R_k rate. For a model with rate estimation, the names of the HBD classes are "HBDclassX" where X is the number of the HBD class. For non-HBD classes, we use "NonHBD". To extract the entire table for the results in BBB cattle with the mod3r model:

```
realized(bbb_mod3r)
#>           R_10        R_100    NonHBD
#> 1 0.0002451684 6.283497e-02 0.9369199
#> 2 0.0429870128 4.589439e-03 0.9524235
#> 3 0.0276528577 2.482290e-02 0.9475242
#> 4 0.0228609411 1.137513e-02 0.9657639
#> 5 0.0133014561 2.845040e-02 0.9582481
#> 6 0.1003679577 5.268664e-07 0.8996315
```

To extract the contribution of the first six HBD classes for the sheep analysis:

```
head(round(realized(soay_mix10r,seq(1,6)),5))
#>     R_2     R_4     R_8    R_16    R_32    R_64
#> 1 3e-05 0.05948 0.00000 0.00000 0.07063 0.11653
#> 2 0e+00 0.00000 0.00000 0.00006 0.08329 0.09915
#> 3 0e+00 0.00000 0.00000 0.00000 0.12816 0.10630
#> 4 1e-05 0.00003 0.00019 0.01993 0.09581 0.10143
#> 5 0e+00 0.00000 0.00000 0.01670 0.10734 0.09015
#> 6 0e+00 0.00000 0.00001 0.00044 0.16462 0.07180
```

7.6.2 For inbreeding coefficients

To obtain the inbreeding coefficient F_{G-T} as the sum of contributions from all HBD classes with a $R_k \leq T$, you can use the `cumhbd` function that takes as arguments the name of the zres object and the value of the threshold T (all HBD classes by default, when no threshold is specified).

To estimate the inbreeding coefficient in the sheep analysis by setting T equal to 100:

```
F100 <- cumhbd(soay_mix10r, 100)
summary(F100)
#>    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
#>  0.1805  0.2094  0.2228  0.2228  0.2369  0.2812
```

To estimate the inbreeding coefficient in the BBB cattle data and the analysis with the mod3r model including either the first HBD class or both of them:

```
F10 <- cumhbd(bbb_mod3r, 10)
F100 <- cumhbd(bbb_mod3r, 100)
cbind(F10,F100)
#>               F10       F100
#> [1,] 0.0002451684 0.06308014
#> [2,] 0.0429870128 0.04757645
#> [3,] 0.0276528577 0.05247576
#> [4,] 0.0228609411 0.03423607
#> [5,] 0.0133014561 0.04175186
#> [6,] 0.1003679577 0.10036848
```

To estimate the inbreeding coefficient in the BBB cattle data but for the analysis performed with the 1 HBD class model. Computations are performed with respect to three values for T (20, 50 and 100) and with an estimated rate R_1 for each individual:

```
F20 <- cumhbd(bbb_results3, 20)
#> [1] "cumhbd is not recommended for a model with different rates per individual!"
F50 <- cumhbd(bbb_results3, 50)
#> [1] "cumhbd is not recommended for a model with different rates per individual!"
F100 <- cumhbd(bbb_results3, 100)
#> [1] "cumhbd is not recommended for a model with different rates per individual!"
cbind(F20,F50,F100,bbb_results3@krates[,1])
#>             F20        F50       F100
#> [1,] 0.00000000 0.00000000 0.05852398 73.280
#> [2,] 0.04253227 0.04253227 0.04253227  6.218
#> [3,] 0.00000000 0.04469971 0.04469971 24.439
#> [4,] 0.00000000 0.03333425 0.03333425 24.021
#> [5,] 0.00000000 0.03563496 0.03563496 35.496
#> [6,] 0.10070006 0.10070006 0.10070006 10.420
```

We first observe that the function does not recommend to estimate an inbreeding coefficient when the R_k rates are estimated. This is because in that situation, each individual has different HBD classes and then the number of classes included in the estimation would vary according to the individuals. Here, only individuals 2 and 6 have a non-zero inbreeding coefficient when setting T equal to 20. All individuals have a non-zero inbreeding coefficient with T equals to 50 with the exception of the first individual. Finally, all the estimated autozygosity is considered when T equals 100.

7.6.3 For HBD segments

The `rohbd` function can help to extract all HBD segments for a group of individuals and a genomic region. By default HBD segments for all individuals and the entire genome are extracted. The individuals are specified by ids = x where x is a vector with the individual numbers. The region is specified by chrom = y where y is the chromosome number. The coordinates can also be specified with the startPos and endPos arguments (these are not used if the chrom is NULL).

Two method of extraction can be used with the inside logical arguments. The function extracts only those segments with start and end positions within the specified region (inside = TRUE) or extracts all HBD segments that overlap the region (inside = FALSE). Output is a data.frame with the same format as the ..@hbdseg slot (see description in 7.4).

To extract all HBD segments identified in the sheep population overlapping the region from 10 to 20 Mb on chromosome 25.

```
roh25_10_20 <- rohbd(zres = soay_mix10r, chrom = 25,
                     startPos= 10000000,endPos = 20000000, inside = FALSE)
dim(roh25_10_20)
#> [1] 84  9
head(roh25_10_20[,1:8])
#>      id chrom start_snp end_snp start_pos  end_pos number_snp   length
#> 4171  2    25       159     237   9657834 13809614         79  4151781
#> 616   2    25       272     338  15416899 18657658         67  3240760
#> 4183  5    25       280     409  15856692 21919108        130  6062417
#> 2272  7    25       318     409  17689768 21919108         92  4229341
#> 618   9    25       299     352  16816914 19159330         54  2342417
#> 337  10    25       267     488  15231602 26224691        222 10993090
summary(roh25_10_20$length)
#>     Min.  1st Qu.   Median     Mean  3rd Qu.     Max.
#>  1005736  2423824  3489860  4809936  6062417 24322495
```

There are 84 HBD segments overlapping the region, with a mean length of 4.8 Mb and the longest segments is more than 24 Mb. If we restrict the extraction on HBD segments inside the region:

```
roh25_10_20 <- rohbd(zres = soay_mix10r, chrom = 25,
                     startPos= 10000000,endPos = 20000000, inside = TRUE)
dim(roh25_10_20)
#> [1] 43  9
head(roh25_10_20[,1:8])
#>       id chrom start_snp end_snp start_pos  end_pos number_snp  length
#> 616    2    25       272     338  15416899 18657658         67 3240760
#> 618    9    25       299     352  16816914 19159330         54 2342417
#> 4218  11    25       270     365  15326949 19823535         96 4496587
#> 4195  12    25       303     362  17029368 19655167         60 2625800
#> 41612 13    25       272     326  15416899 17997283         55 2580385
#> 22413 14    25       218     237  12792777 13809614         20 1016838
summary(roh25_10_20$length)
#>    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
#> 1005736 2278147 2446597 2651727 3173355 4755436
```

There are only 43 HBD segments and the longest is less than 5 Mb in size. Extraction, can also be performed per individual for the entire genome. For instance, for individuals 56, 15, 97, 103 and 108 we find 679 HBD segments (the first individual column is now 15):

```
roh25_10_20 <- rohbd(zres = soay_mix10r, ids = c(15,56,97,103,108))
dim(roh25_10_20)
#> [1] 676   9
head(roh25_10_20[,1:8])
#>      id chrom start_snp end_snp start_pos   end_pos number_snp   length
#> 2180 15     1       228     262  11540961  13301343         35  1760383
#> 4100 15     1       324     403  16471993  21164217         80  4692225
#> 690  15     1      1260    1385  68918632  76227410        126  7308779
#> 890  15     1      2241    2459 124296584 137186574        219 12889991
#> 1080 15     1      2467    2511 137752062 139783950         45  2031889
#> 1268 15     1      3078    3163 171842250 176779276         86  4937027
```

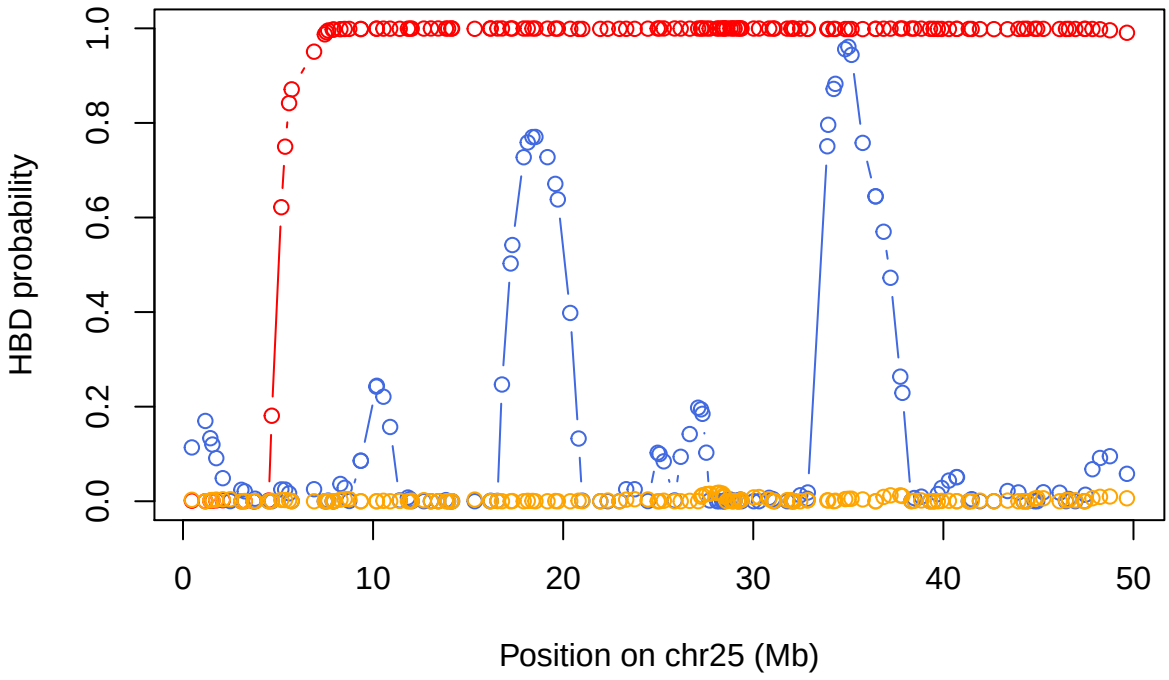
7.6.4 For local HBD probabilities (locus specific)

The `probhbd` function help to extract the local HBD probabilities for one individual in a specific region. HBD probabilities represent large amount of data. Therefore, we extract only a total HBD probability by summing over all HBD classes. As for the inbreeding coefficient, the user can specify a threshold T to determine which HBD classes should be used in the sum and considered as autozygous (all by default).

The function takes as arguments the `zres` object but also the `zooin` object (that contains some information on the markers, their position and the chromosome number). The individual is indicated with the id = . argument. The same arguments as for the `rohbd` function are used to declare the genomic region (chrom, startPos, endPos). By default all positions are extracted.

Here is an example to extract HBD probabilities for the sixth individual for the BBB cattle data, on chromosome 19 from position 10 Mb to 20 Mb and plot it (we need also to extract the position from the zooin@bp slot). In addition, we extract HBD probabilities for two other individuals:

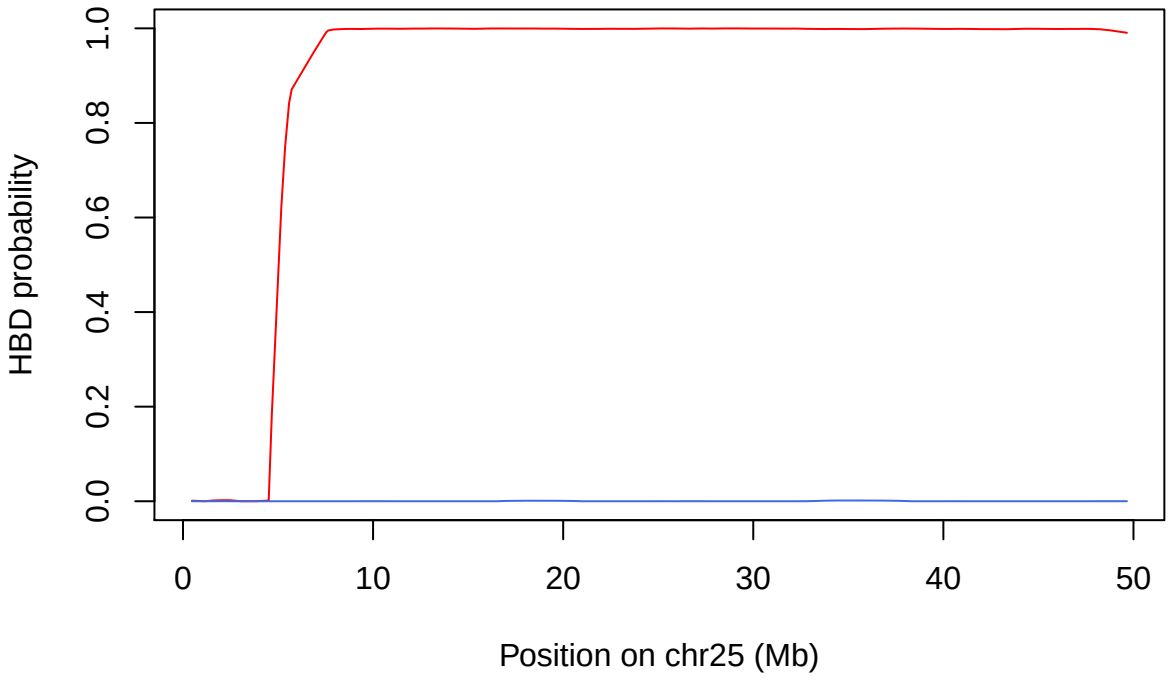
```
y6 <- probhbd(zres = bbb_mod3r, zooin = bbb, id = 6, chrom = 19,
              startPos = 0, endPos = 50000000)
x <- bbb@bp[bbb@chrbound[19,1]:bbb@chrbound[19,2]]
x <- x[x >=0 & x<= 50000000]/1000000
plot(y6~x,type='b',ylim=c(0,1),ylab='HBD probability',col='red',
     xlab='Position on chr25 (Mb)')
y1 <- probhbd(zres = bbb_mod3r, zooin = bbb, id = 1, chrom = 19,
              startPos = 0, endPos = 50000000)
y2 <- probhbd(zres = bbb_mod3r, zooin = bbb, id = 2, chrom = 19,
              startPos = 0, endPos = 50000000)
par(new=TRUE)
plot(y1~x,type='b',ylim=c(0,1),ylab='',col='royalblue',xlab='',axes=FALSE)
par(new=TRUE)
plot(y2~x,type='b',ylim=c(0,1),ylab='',col='orange',xlab='',axes=FALSE)
```



We observe that individual 6 has a long HBD segment, whereas there is no evidence of HBD segments in individual 2. For the first individual, we see two small regions were the HBD probabilities reach approximately 0.80 and 0.95. These probabilities suggest presence of HBD segment that would not have been captured with window-based approaches because they contain too few markers.

The use of the threshold T determines which classes are included. In the previous example, we used two HBD classes. By setting T = 20, we would use only one HBD class with $R_k = 10$ (and not the class with $R_k = 100$):

```
y6b <- probhbd(zres = bbb_mod3r, zooin = bbb, id = 6, chrom = 19,
               startPos = 0, endPos = 50000000,T=20)
y1b <- probhbd(zres = bbb_mod3r, zooin = bbb, id = 1, chrom = 19,
               startPos = 0, endPos = 50000000,T=20)
plot(y6b~x,type='l',ylim=c(0,1),ylab='HBD probability',col='red',
     xlab='Position on chr25 (Mb)')
par(new=TRUE)
plot(y1b~x,type='l',ylim=c(0,1),ylab='',col='royal blue',xlab='',axes=FALSE)
```



The curve is almost identical for individual 6 because the long HBD segment is associated to the first HBD class (rate $R_k = 10$). The HBD probabilities for the first individual are now equal to 0, because the evidence for short segments previously observed is associated with the second HBD class, with a higher rate (100) corresponding to shorter segments.

7.6.5 Combining accessor functions with standard summary functions

The accessors can be combined with standard summary functions to obtain summary of the results (realized autozygosity, inbreeding coefficients and the HBD segments). We already had some examples with histograms of the inbreeding coefficients (see 7.6.2).

To obtain a summary of the realized inbreeding:

```
summary(realized(soay_mix10r))
#>       R_2                 R_4                 R_8
#>  Min.   :0.000e+00   Min.   :0.000e+00   Min.   :0.000e+00
#>  1st Qu.:0.000e+00   1st Qu.:0.000e+00   1st Qu.:1.000e-08
#>  Median :3.300e-09   Median :3.000e-08   Median :5.900e-07
#>  Mean   :4.285e-06   Mean   :1.891e-03   Mean   :3.310e-03
#>  3rd Qu.:1.038e-07   3rd Qu.:1.020e-06   3rd Qu.:5.146e-05
#>  Max.   :3.629e-04   Max.   :7.638e-02   Max.   :5.862e-02
#>       R_16                R_32               R_64
#>  Min.   :0.000e+00   Min.   :0.00000   Min.   :0.02755
#>  1st Qu.:2.850e-06   1st Qu.:0.07032   1st Qu.:0.09041
#>  Median :1.397e-04   Median :0.09563   Median :0.10679
#>  Mean   :1.788e-02   Mean   :0.09288   Mean   :0.10678
```

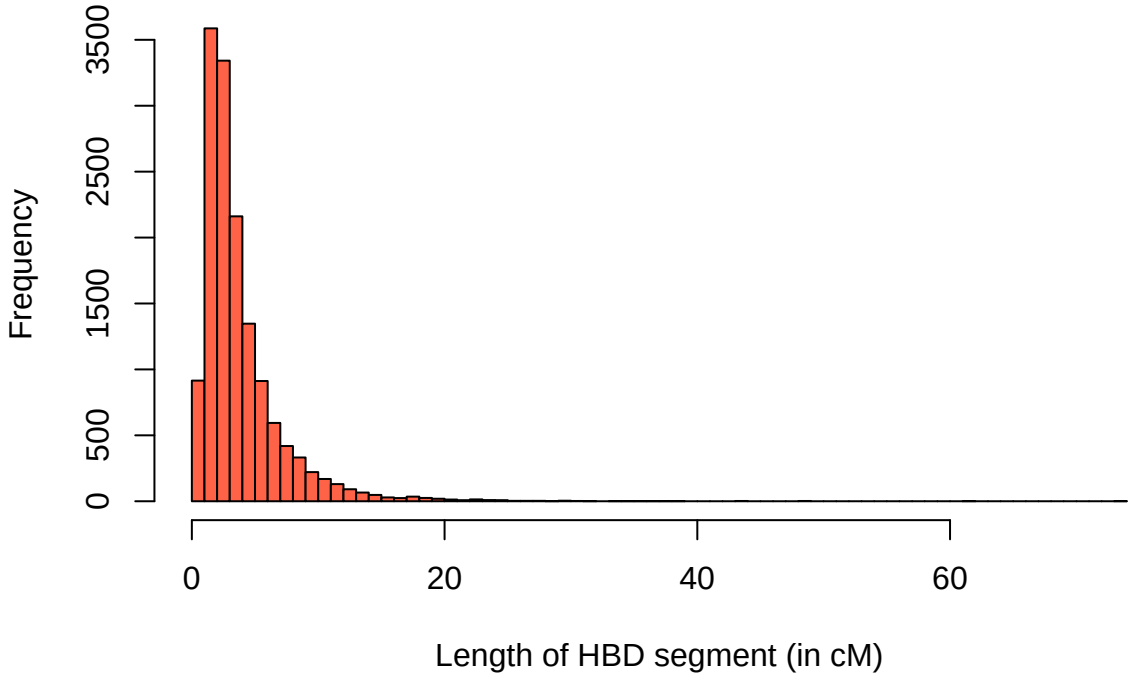
```
#>  3rd Qu.:3.308e-02   3rd Qu.:0.11857   3rd Qu.:0.12660
#>  Max.   :9.016e-02   Max.   :0.17277   Max.   :0.16842
#>      R_128               R_256               R_512
#>  Min.   :0.000e+00   Min.   :0.000e+00   Min.   :0.000e+00
#>  1st Qu.:6.200e-08   1st Qu.:4.410e-11   1st Qu.:1.810e-12
#>  Median :5.610e-07   Median :1.195e-09   Median :9.956e-11
#>  Mean   :1.353e-04   Mean   :8.266e-08   Mean   :1.121e-08
#>  3rd Qu.:3.198e-06   3rd Qu.:3.407e-08   3rd Qu.:4.397e-09
#>  Max.   :1.414e-02   Max.   :2.387e-06   Max.   :2.762e-07
#>      NonHBD
#>  Min.   :0.7188
#>  1st Qu.:0.7631
#>  Median :0.7772
#>  Mean   :0.7771
#>  3rd Qu.:0.7906
#>  Max.   :0.8195
```

Similarly, for inbreeding coefficients with T equal to 100:

```
summary(cumhbd(soay_mix10r,100))
#>    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
#>  0.1805  0.2094  0.2228  0.2228  0.2369  0.2812
```

The `summary` function can also be used with the `hbdseg` data.frame (for the entire frame or for some variables). We can also use a function to plot an histogram of length of HBD segments:

```
allseg <- rohbd(zres = soay_mix10r)
summary(allseg$length)
#>     Min.  1st Qu.   Median     Mean  3rd Qu.     Max.
#>   116689  1756868  2811065  3790554  4620455 73719189
hist(allseg$length/1000000,xlab="Length of HBD segment (in cM)",main="",col='tomato',nc=100)
```



Alternatively, we provide plot functions to obtain an overview of the results and visualize them.

7.7 Information for DoRIS

DoRIS is a software (Palamara et al., 2012) that use IBD segments to infer past demographic history in populations. To that end, it uses the number of IBD segments observed in 1Mb bins (over a range of values). The distribution can help to infer past effective population size. The software works either with counts, either with proportion of the genome in different bins ("sharing"). HBD segments are a special case of IBD segments. With IBD segments, the number of pair of haplotypes compared is large that with HBD segments (equal to the number of individuals) but with HBD segments there is no need to phase the data. If the sample is large enough and autozygous segments are frequent, HBD segments can be used for the application.

The `zoodoris` function allows to generate tables for DoRIS either as counts (`method = "counts"`) or as proportion of the genome (`method = "sharing"`). The function works on a `zres` object and need the range of HBD segments that are used (`minv` and `maxv` are the minimal and maximal length considered, in cM). Finally, the user must also provide the length of the genome in cM (using only autosomes where HBD segments have been searched for) with the `glen` argument.

To extract counts with the results from the Soay population and using bins from 5 to 10 cM long:

```
zoodoris(zres = wilt_mix10r, minv = 5, maxv = 10, glen = 2450, method = "counts")
#>   win_st win_end nseg
#> 1      5       6  148
#> 2      6       7  117
#> 3      7       8  104
```

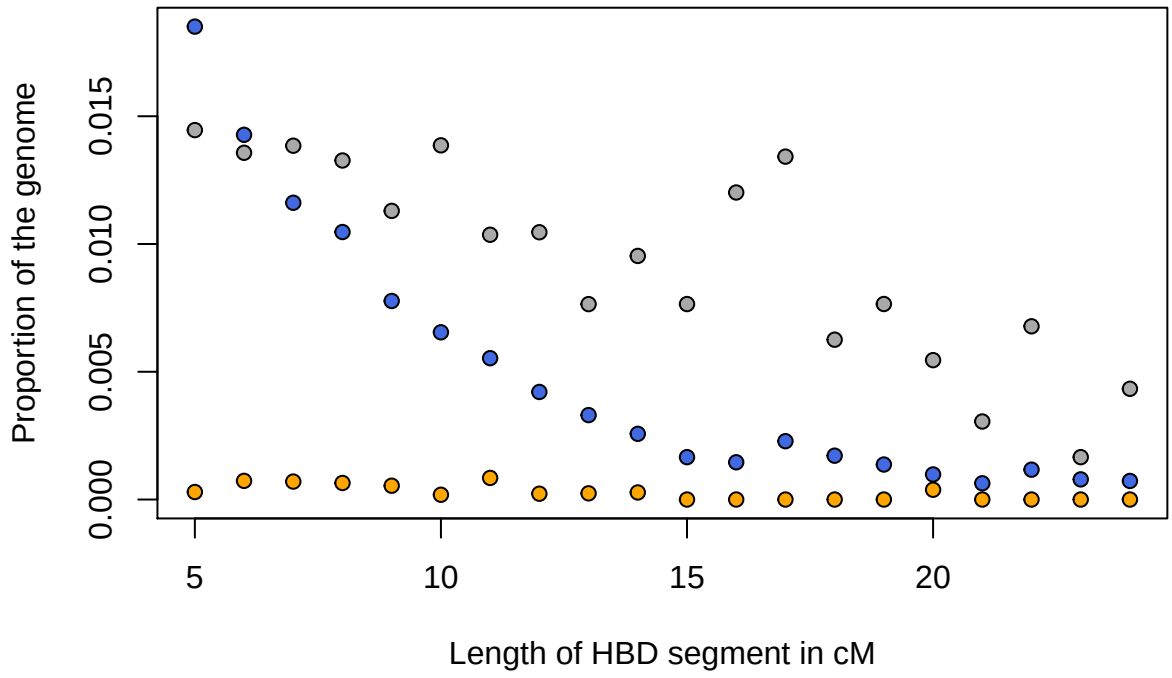
```
#> 4      8       9   88
#> 5      9      10   67
```

To obtain proportion of the genome in the bins:

```
zoodoris(zres = wilt_mix10r, minv = 5, maxv = 10, glen = 2450, method = "sharing")
#>   win_st win_end    sharing
#> 1      5       6 0.01445879
#> 2      6       7 0.01357026
#> 3      7       8 0.01384652
#> 4      8       9 0.01326850
#> 5      9      10 0.01129579
```

We can compare the distribution for three different sheep populations:

```
dtw <- zoodoris(zres = wilt_mix10r, minv = 5, maxv = 25, glen = 2450, method = "sharing")
dts <- zoodoris(zres = soay_mix10r, minv = 5, maxv = 25, glen = 2450, method = "sharing")
dtr <- zoodoris(zres = rara_mix10r, minv = 5, maxv = 25, glen = 2450, method = "sharing")
matplot(dtw$win_st,cbind(dtw$sharing,dts$sharing,dtr$sharing),
        xlab = 'Length of HBD segment in cM', ylab = 'Proportion of the genome',
        type ='p', pch=c(21,21,21),bg=c('dark grey','royal blue','orange'),
        col=c('black','black','black'))
```



Clear differences are observed, with the Rasa Aragonesa individuals (orange) having few HBD segments in all bins, the Soay individuals (blue) having a high number of the shortest segments and the Wiltshire individuals (grey) presenting also large values for long HBD segments.

8 Plotting

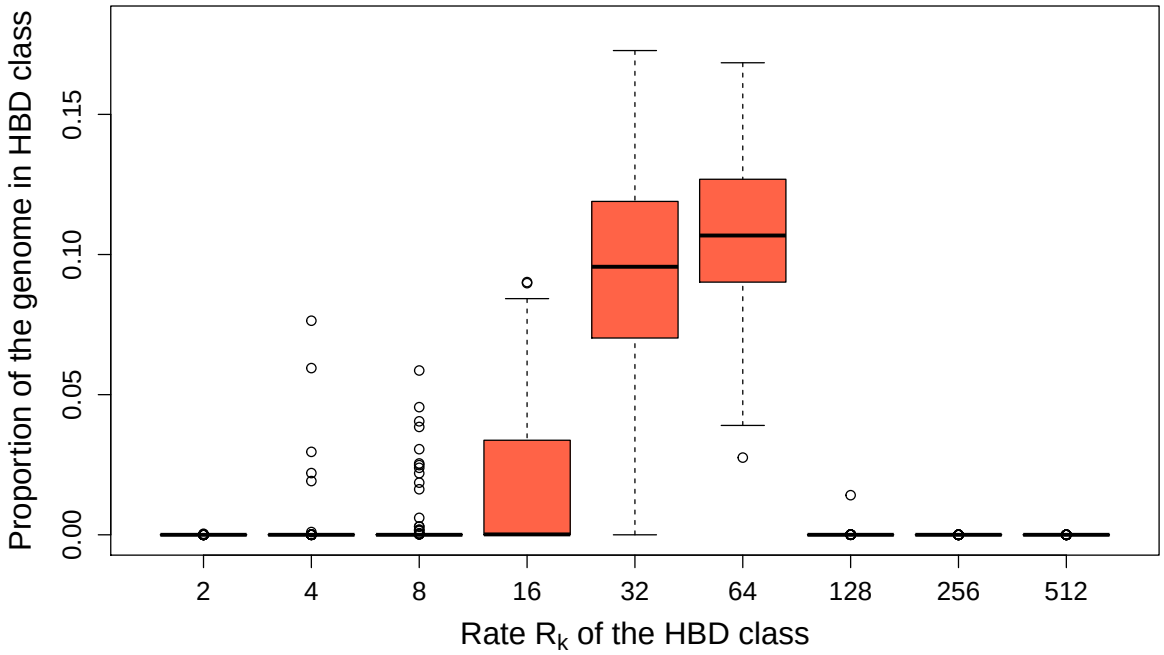
Four plotting functions are helpful to interpret the results. They plot either the partitioning of the genome in HBD class at the population or the individual level, the contribution of different classes to the genome or HBD segments. We recommend to use the functions representing proportion of the genome in different HBD classes or partitioning of the genome in HBD classes **only with models with pre-defined rates** (see section 5).

8.1. Proportion of the genome associated with different HBD classes (population)

The `zooplot_prophbd` represents the proportion of the genome associated with the different HBD classes at the population level. The input is a named list of zres object (list(name1 = zres1, name2 = zres2, ...)). The list can contain one or several populations. When provided, the names are used in the plot. Three format can be used with the `style` arguments: "barplot", "lines" and "boxplot" (boxplot can only be use with a single zres object).

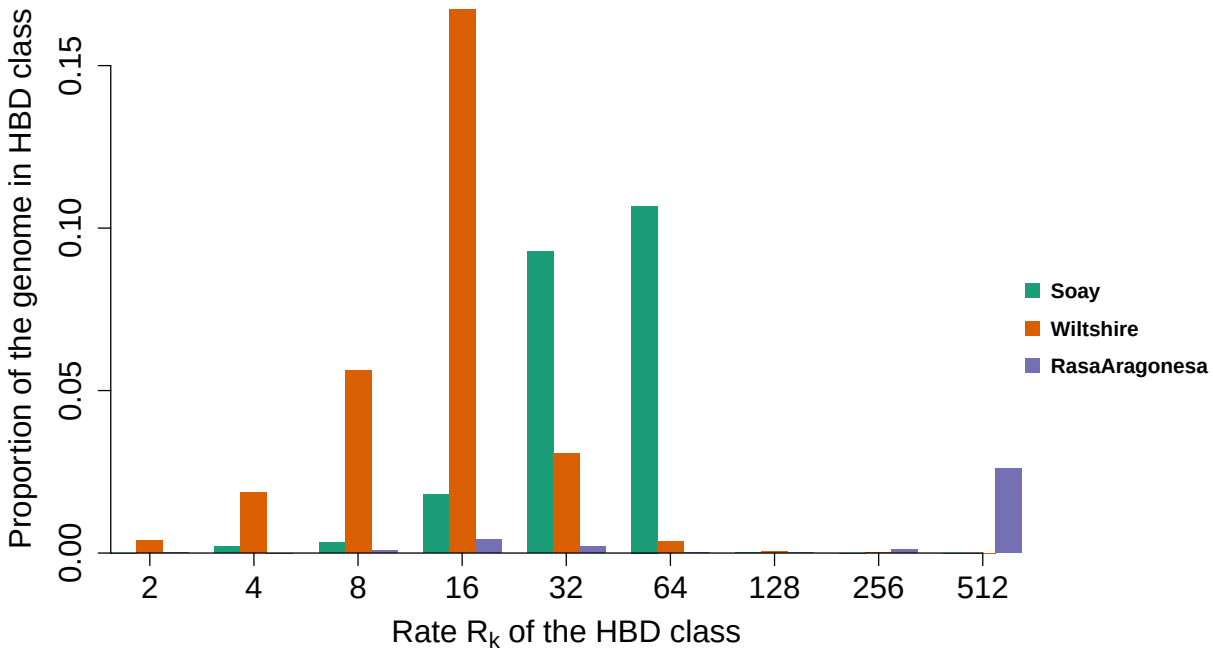
To plot a single population with boxplots:

```
zooplot_prophbd(list(Soay = soay_mix10r), cols = 'tomato', style = 'boxplot')
```



To plot three populations with barplots:

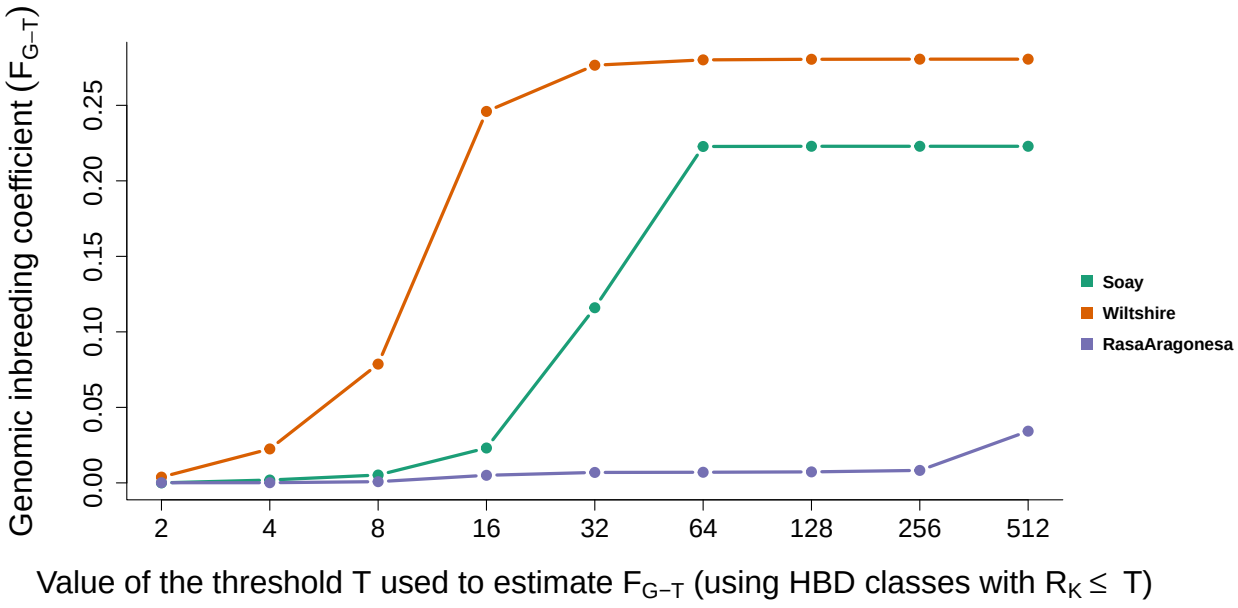
```
zooplot_prophbd(list(Soay=soay_mix10r,Wiltshire=wilt_mix10r,
                     RasaAragonesa=rara_mix10r),style='barplot')
```



The plot can also represents cumulative proportions: the fraction of the genome in all HBD classes with rates $R_k \leq T$. For the plot, we use the rate of the HBD classes as values for T. So, we obtain the fraction of the genome in the first HBD class, the two first HBD classes, the first three HBD classes, etc. These values can be interpreted as inbreeding coefficients estimated with respect to different base populations (see sections 2.4, 7.3, 7.5.2 or Druet and Gautier (2017) and Solé et al. (2017) for more details).

To plot the average inbreeding coefficients (cumulative values) for three populations with lines:

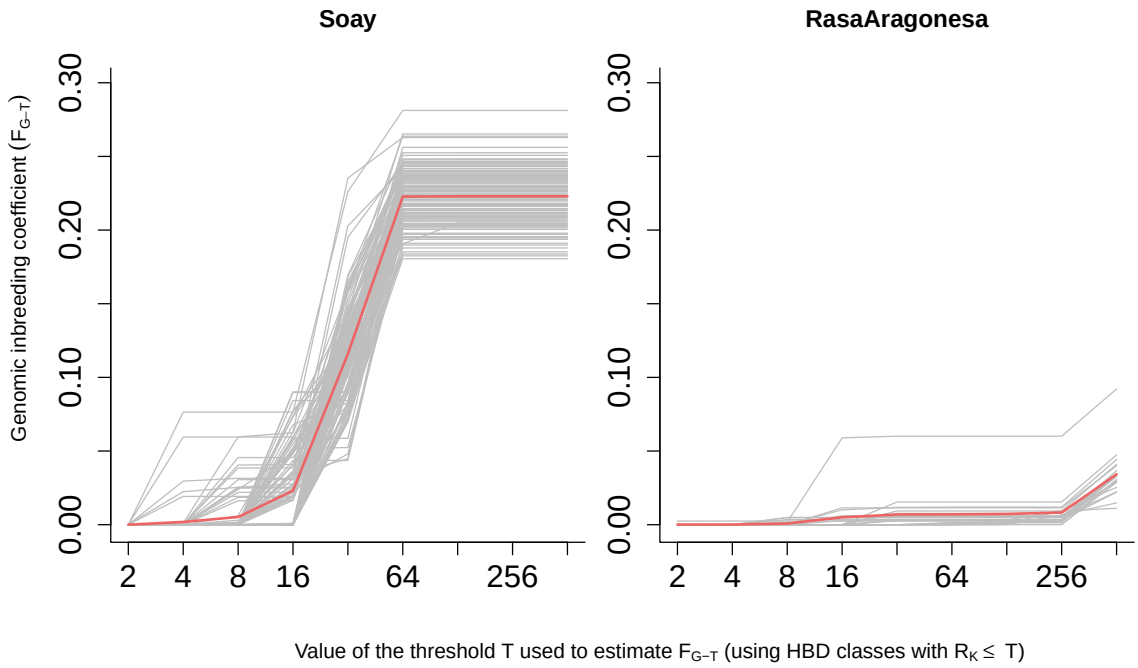
```
zooplot_prophbd(list(Soay=soay_mix10r,Wiltshire=wilt_mix10r,
                     RasaAragonesa=rara_mix10r),style='lines', cumulative = TRUE)
```



8.2. Proportion of the genome associated with different HBD classes (individuals)

The `zooplot_individuals` function represents the same results as the `zooplot_prophbd` function but at an individual level. Each individual is represented by a line (no barplots or boxplots). The function plots either the proportion of the genome in different HBD classes or cumulative proportions (see above) if `cumulative` is set to `TRUE`. The average value of the population is added in red. As before, the input is a named list of zres object (list(name1 = zres1, name2 = zres2, ...)). The list can contain one or several populations. When provided, the population names are used in the plot. The `ncols` argument determines the number of graphs (populations) plotted next to each other.

```
pop2 <- list(Soay=soay_mix10r,RasaAragonesa=rara_mix10r)
zooplot_individuals(pop2, cumulative = TRUE)
#> Warning in zooplot_individuals(pop2, cumulative = TRUE):
#> All individuals are plotted; use toplot to select individuals
```



We observe than Soay sheep present higher levels of autozygosity and that patterns are relatively homogeneous in both populations.

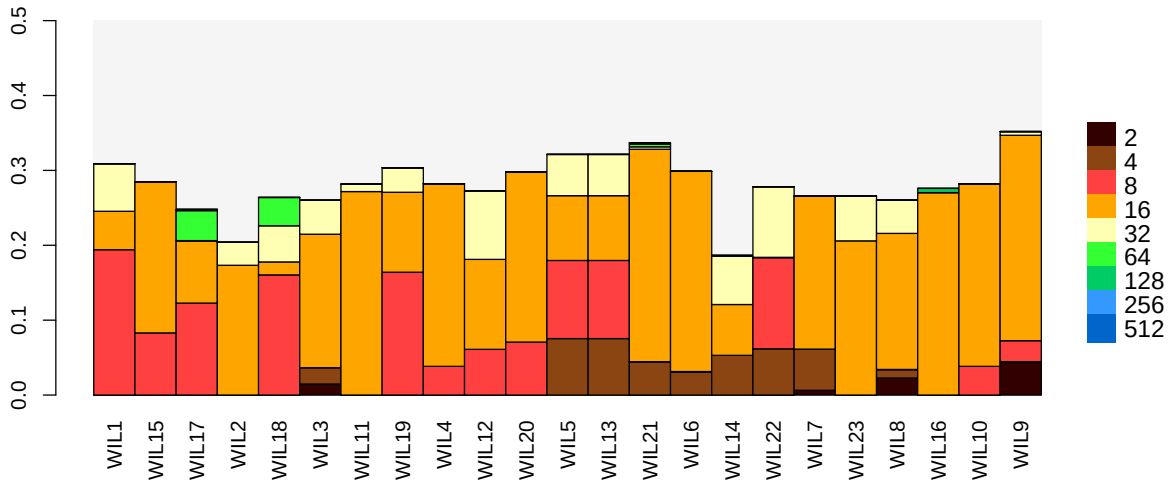
8.3. Partitioning individual genomes in different HBD classes

The `zooplot_partitioning` function represents for each individual the proportion of the genome in each HBD class. Each individual is represent as a stacked barplot. The total height represents the total autozygosity level. The contribution of each HBD class (in % of the genome) is represented as a bar of a different color (black, brown, red for the most recent HBD classes).

The input is again a named list of zres object (list(name1 = zres1, name2 = zres2, ...)). The list can contain one or several populations. When provided, the population names are used in the plot. The user can provide the colors of the bars (argument `cols`), can choice to plots the ids or not (argument `plotids`), can give a list of individuals to plot (argument `toplot`, a list of vectors containing the individuals to plot for each population), can choose to plot a random sample of individuals (argument `randomids` set to TRUE with

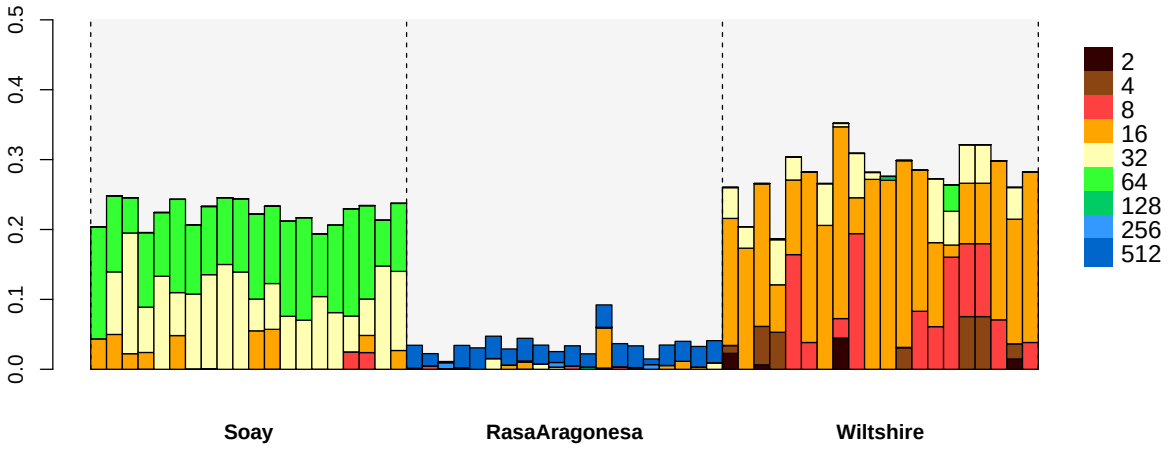
`nrandom` a vector containing the number of individuals to select for each population and `seed` being the random seed). The `ylim` argument indicates the minimal and maximal values of the y-axis, the `border` argument indicates whether a border is drawn around each bar, the `nonhbd` argument indicates whether a border is drawn around the non-hbd contribution and the `vertical` argument specifies if sample names are written vertically or not. To plot the partitioning in the Wiltshire population:

```
zooplot_partitioning(list(Wiltshire=wilt_mix10r), ylim = c(0,0.5), nonhbd = FALSE)
```



Next we plot the three sheep population. To reduce the number of individuals, we select randomly 20 individuals per population and we don't print the ids.

```
pop3 <- list(Soay=soay_mix10r,RasaAragonesa=rara_mix10r,Wiltshire=wilt_mix10r)
zooplot_partitioning(pop3, randomids = TRUE, nrandom = c(20,20,20), plotids = FALSE,
                     ylim=c(0,0.5), nonhbd = FALSE)
#> Warning in zooplot_partitioning(pop3, randomids = TRUE, nrandom = c(20, :
#> Random seed 100 is used to sample individuals.
```



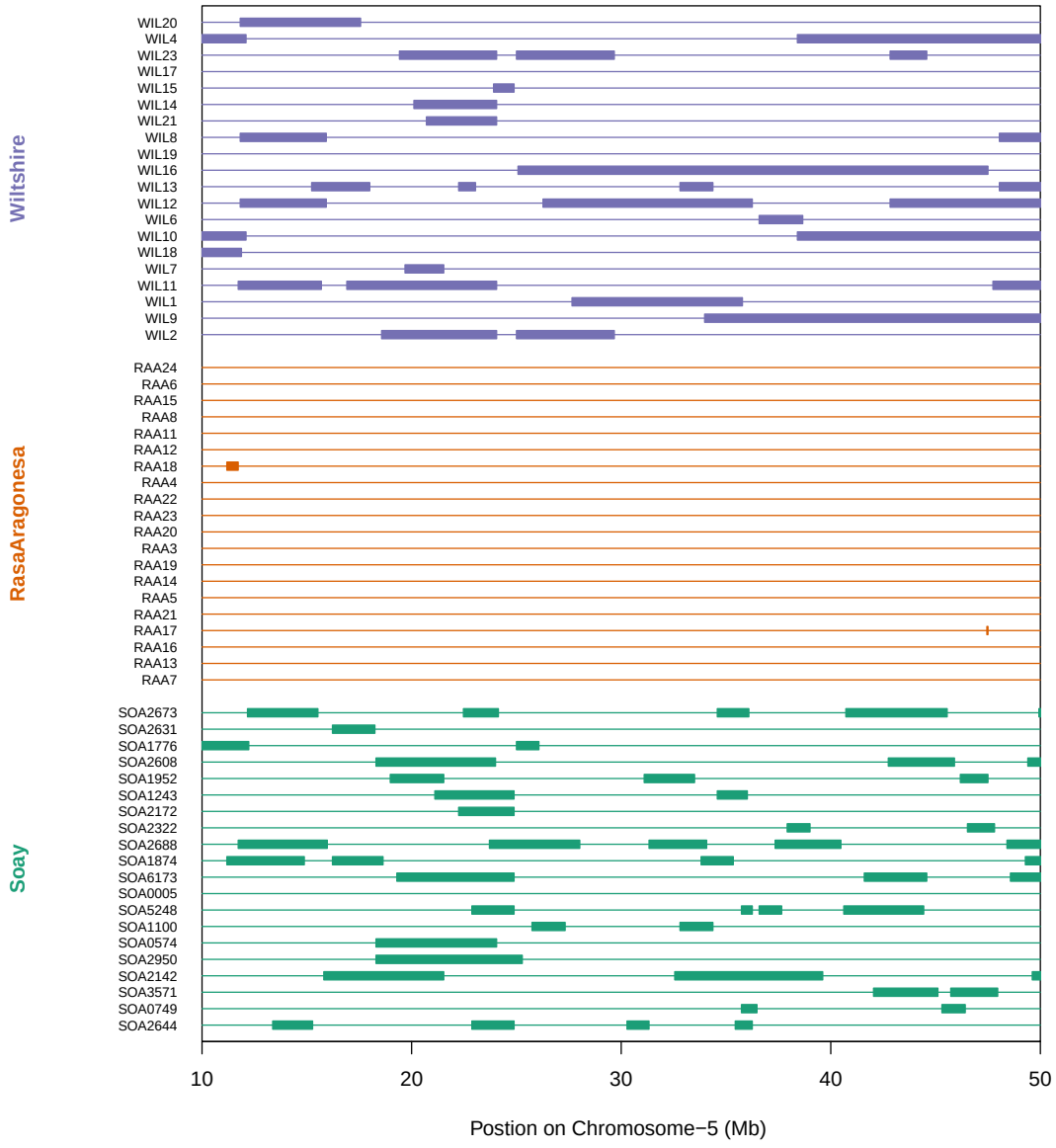
Soay and Wiltshire populations present higher levels of autozygosity. The Wiltshire individuals present more recent autozygosity (brown, red, orange bars) compared to the Soay individuals where light yellow ($R_k = 32$) and green ($R_k = 64$) dominate, indicating that the Wiltshire individual have more recent common ancestors causing autozygosity.

8.4. Plotting identified HBD segments

The `zooplot_hbdseg` function plots the identified segments for a selected region. The region is specified with the `chr` and `coord` arguments. A minimal segment size can be selected with the `minlen` arguments.

Some arguments are identical to the `zooplot_partitioning function`: those to select individuals, `randomids`, `nrandom`, `seed`, `toplot`, `plotids`. The `cols` argument allows to specify the color used for each population or zres object in the input list.

```
zooplot_hbdseg(pop3, randomids = TRUE, nrandom = c(20,20,20),
               chr=5,coord=c(10000000,50000000))
#> random seed  100  is used to sample ids
```



9 Impact of data informativity

The genotyping technology will impact the amount of available data to characterize autozygosity and identify HBD segments. The method will be more efficient with more SNPs per HBD segment, with less genotyping / sequencing errors, when markers are more informative (higher minor allele frequency - MAF) or when sequencing is performed at higher coverage. There are no simple guidelines indicating how well the method will perform as it depends on many factors specific to data sets. Still we performed some simulations to asses the impact of different factors on the mean absolute errors (MAE) of HBD probabilities, the absolute value of the difference between the HBD probability and the true HBD status (Druet and Gautier, 2017). The results are summarized in Figure 4, that can help to understand the impact of lower marker density, marker informativity or sequencing coverage and to estimate the number of SNP required per HBD segment to obtain descent estimates.

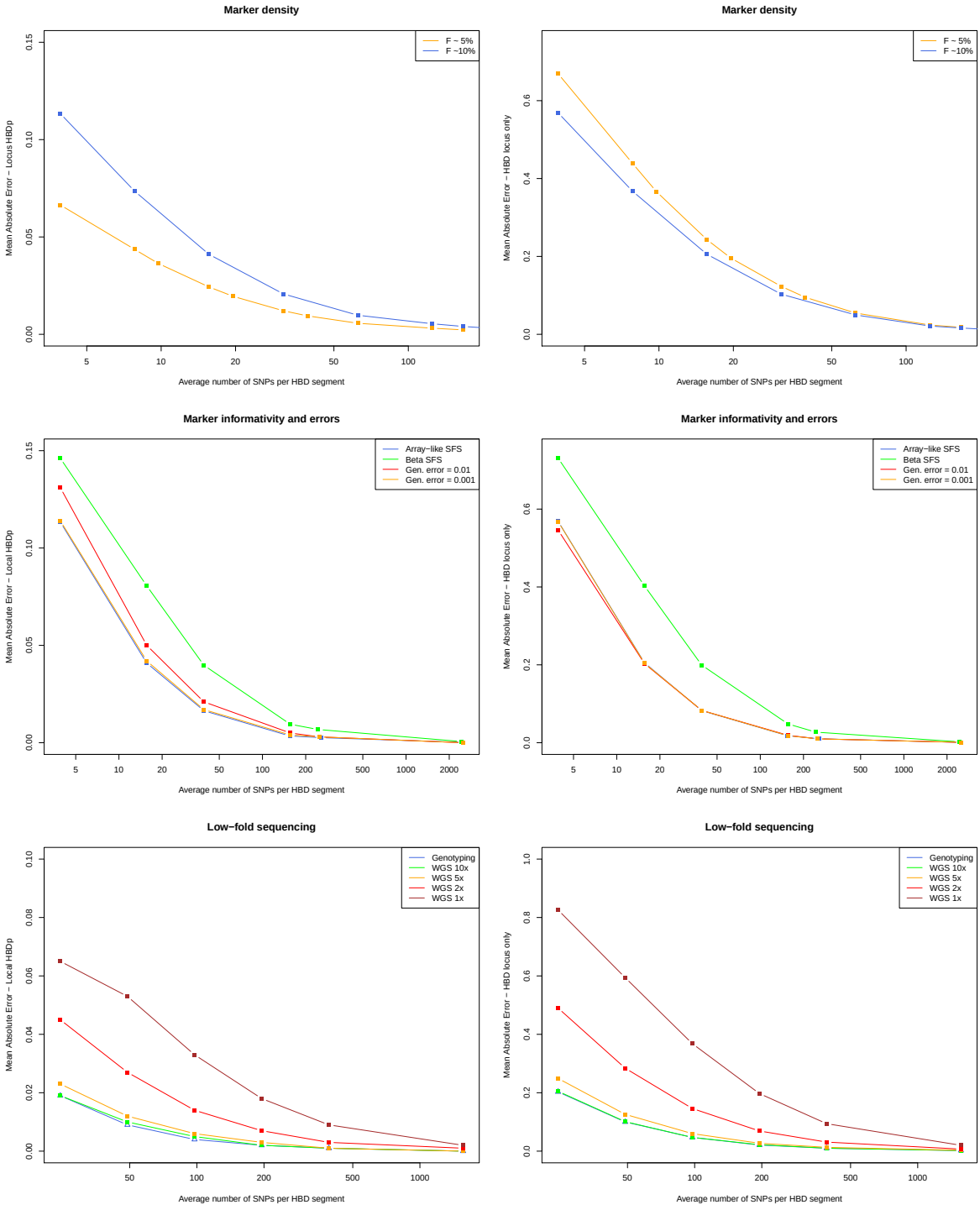


Figure 4: Impact on marker informativity on mean absolute error (MAE) for local HBD probabilities for all locus (left column) or for HBD locus only (right column) based on results from results from Druet and Gautier (2017). The figure shows the impact of average number of SNPs per HBD segment, site frequency spectrum (SFS), presence of genotyping errors or coverage with whole-genome sequencing (WGS) data.

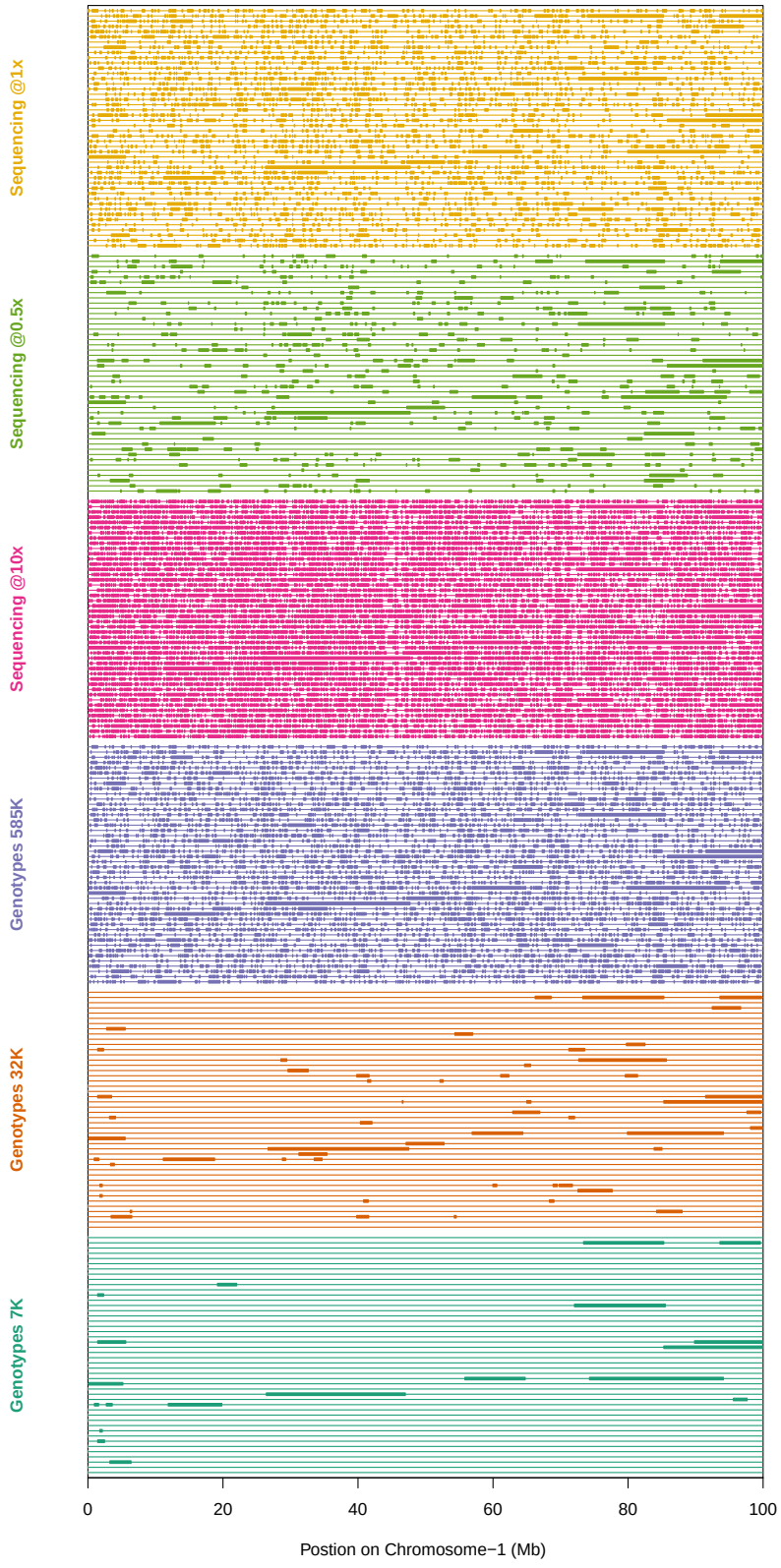


Figure 5: HBD segments identified on chromosome 1 using different genotyping densities or using whole-genome sequencing at different coverages in 46 Belgian blue sires.

With genotyping data, the HBD segments are accurately identified with 50 or 100 SNPs per segments (as for window-based RoH) and with 20 SNPs per segment the method is still efficient but in that case, false positive and false negative rates are higher and the use of HBD probabilities is recommended (it is risky to use window-based RoH with only 20 SNPs per window). If we use sequencing data, we observe that with low cover, the number of marker per HBD segments must be increased to obtain similar accuracy.

We can observe that genotyping error rates have a small impact because they are accounted for in the model. Still, with higher error rate (0.01) we observe that the number of false positive HBD segments increases because heterozygous genotypes are less penalized, the model allows approximately one heterozygous SNP per 100 SNPs (the accuracy in HBD segments remains however unchanged). Similarly, when the overall autozygosity is higher the risk of false positive HBD segments increases because since HBD segments are more frequent, the probability to observe HBD segments is higher. In that case, we also observe that the power to identify the HBD segments increases in parallel.

Regarding the genome-wide autozygosity level, its association with the number of SNPs per HBD segment is more complex as it also depends the total number of SNPs.

In addition to these simulations, we show results obtained with low-fold sequencing in Belgian Blue beef cattle to illustrate with real data that the approach can indeed capture HBD segments even with low-fold sequencing. Figure 5 represents for 46 individuals HBD segments obtained with genotyping arrays (~ 2-3 SNPs per Mb, ~ 10 SNPs per Mb and ~ 200 SNPs per Mb) and with whole genome sequencing (10x, 0.5x and 1x). The results from the 1x are similar to those of the highest SNPs density (in terms of number of segments and their length) whereas the results from the 0.5x lie in between those obtained with ~10 SNPs per Mb and those obtained with ~ 200 SNPs per Mb.

10 References

References

Broman, K. W. and Weber, J. L. (1999). Long homozygous chromosomal segments in reference families from the centre d'etude du polymorphisme humain. *The American Journal of Human Genetics*, 65(6):1493–1500.

Druet, T. and Gautier, M. (2017). A model-based approach to characterize individual inbreeding at both global and local genomic scales. *Molecular ecology*, 26(20):5820–5841.

Leutenegger, A.-L., Labalme, A., Génin, E., Toutain, A., Steichen, E., Clerget-Darpoux, F., and Edery, P. (2006). Using genomic inbreeding coefficient estimates for homozygosity mapping of rare recessive traits: application to taybi-linder syndrome. *The American journal of human genetics*, 79(1):62–66.

Leutenegger, A.-L., Prum, B., Génin, E., Verny, C., Lemainque, A., Clerget-Darpoux, F., and Thompson, E. A. (2003). Estimation of the inbreeding coefficient through use of genomic data. *The American Journal of Human Genetics*, 73(3):516–523.

Leutenegger, A.-L., Sahbatou, M., Gazal, S., Cann, H., and Génin, E. (2011). Consanguinity around the world: what do the genomic data of the hgdp-ceph diversity panel tell us? *European Journal of Human Genetics*, 19(5):583.

Magi, A., Tattini, L., Palombo, F., Benelli, M., Gialluisi, A., Giusti, B., Abbate, R., Seri, M., Gensini, G. F., Romeo, G., et al. (2014). H 3 m 2: detection of runs of homozygosity from whole-exome sequencing data. *Bioinformatics*, 30(20):2852–2859.

Marchini, J., Howie, B., Myers, S., McVean, G., and Donnelly, P. (2007). A new multipoint method for genome-wide association studies by imputation of genotypes. *Nature genetics*, 39(7):906.

McQuillan, R., Leutenegger, A.-L., Abdel-Rahman, R., Franklin, C. S., Pericic, M., Barac-Lauc, L., Smolej-Narancic, N., Janicijevic, B., Polasek, O., Tenesa, A., et al. (2008). Runs of homozygosity in european populations. *The American Journal of Human Genetics*, 83(3):359–372.

Narasimhan, V., Danecek, P., Scally, A., Xue, Y., Tyler-Smith, C., and Durbin, R. (2016). Bcftools/roh: a hidden markov model approach for detecting autozygosity from next-generation sequencing data. *Bioinformatics*, 32(11):1749–1751.

Palamara, P. F., Lencz, T., Darvasi, A., and Pe'er, I. (2012). Length distributions of identity by descent reveal fine-scale demographic history. *The American Journal of Human Genetics*, 91(5):809–822.

Pemberton, T. J., Absher, D., Feldman, M. W., Myers, R. M., Rosenberg, N. A., and Li, J. Z. (2012). Genomic patterns of homozygosity in worldwide human populations. *The American Journal of Human Genetics*, 91(2):275–292.

Purcell, S., Neale, B., Todd-Brown, K., Thomas, L., Ferreira, M. A., Bender, D., Maller, J., Sklar, P., De Bakker, P. I., Daly, M. J., et al. (2007). Plink: a tool set for whole-genome association and population-based linkage analyses. *The American Journal of Human Genetics*, 81(3):559–575.

Rabiner, L. R. (1989). A tutorial on hidden markov models and selected applications in speech recognition. *Proceedings of the IEEE*, 77(2):257–286.

Solé, M., Gori, A.-S., Faux, P., Bertrand, A., Farnir, F., Gautier, M., and Druet, T. (2017). Age-based partitioning of individual genomic inbreeding levels in belgian blue cattle. *Genetics Selection Evolution*, 49(1):92.

Thompson, E. A. (2013). Identity by descent: variation in meiosis, across genomes, and in populations. *Genetics*, 194(2):301–326.

Vieira, F. G., Albrechtsen, A., and Nielsen, R. (2016). Estimating ibd tracts from low coverage ngs data. *Bioinformatics*, 32(14):2096–2102.

Wang, S., Haynes, C., Barany, F., and Ott, J. (2009). Genome-wide autozygosity mapping in human populations. *Genetic Epidemiology: The Official Publication of the International Genetic Epidemiology Society*, 33(2):172–180.

Zucchini, W. and MacDonald, I. (2009). Hidden markov models for time series, volume 110 of monographs on statistics and applied probability.