The R-Package ’synbreed’

Valentin Wimmer[*]

August 30, 2010

Abstract

This document gives an introduction to the R-Package ’synbreed’ which contains tools and methods for plant and animal breeding. The goal is the creation of an analysis pipeline for genomic selection. This comprises tools for genotypic, phenotypic and pedigree data. The steps of a typical analysis are presented in this document. This starts with the coding of the marker data, followed by the construction of relationship matrizes according to pedigree or genomic relationship matrizes, i.e. according to vanRaden (2008).

Keywords: synergistic plant and animal breeding, kinship, pedigree, marker data, animal model

1 Introduction

In this document the steps of an analysis pipline for genotypic and phenotypic data in plant or animal breeding with the R-package ’synbreed’ are presented. As in illustration, a simulated data set for maize called `maize` which is part of the package is used. To load `maize` data set, use

```
> library(synbreed)
> data(maize)
```

This data set contains genotypic and phenotypic data for 1250 genotpes of maize. When loading the `maize` data set, in fact the following data sets are loaded in the workspace

maize.geno This is a `data.frame` containing the marker data of 696 biallelic SNP-markers for the 1250 genotypes. The first column of the data set contains the `ID` for the identification of the genotypes. This variable should be used for the merge with the phenotypic data. The marker data is coded with 0/1 and because of simulated data no missing values are present. Note that the coding does not contain any information if an allels is the minor or major allele at one locus.

[*]Author of correspondance: Institute for plant breeding, Technical University of Munich, Germany, Email: `Valentin.Wimmer@wzw.tum.de`

maize.pheno This is a `data.frame` with one column `ID` and a column `Trait` containing the measured phenotypic trait. The order of the genotypes is the same as the order of rows in `maize.geno`.

maize.ped This `data.frame` contains the pedigree information of 1301 genotypes.

maize.marker.pos This `data.frame` contains additional information for the SNP markers. The first column `pos` gives the position of the marker on the chromosome in cM. The second column `chr` sepecifies the chromosome (linkage group) the marker belongs to. The order of the markers is the same as the order of columns in `maize.geno`.

2 Marker data

In the first step the marker data has to be coded in a way that it could be used for the construction of genomic relationship matrices. For biallelic marker data, the minor allele should be coded as 2 and the major allele as 0. This task is done by the function `codeGeno`. If no missing values are present and all markers should be used in the following analyses, this function does the simply the recoding of the alleles. For the `maize` data, use

```
> marker <- codeGeno(maize.geno[, -1])
```

to obtain a object `marker` which contains the recoded marker data. Note that the first column is not used because it contains the `ID`. Now, the minor allele frequencies are easily obtained by dividing the column means of `marker` by 2. A histogramm of the minor allele frequencies is shown in Figure 1.

In real data usually missing values occour in genotypic data due to different reasons (i.e. heterozygous genotypes for one locus in homozygous lines). The function `codeGeno` provides the possibility to impute missing values by chance or according to family structure by the following rules:

with population structure Suppose an observation i is missing (NA) for a marker j in population k. If marker j is fixed in population k, the imputed value will be the fixed allele. If marker j is segregating for the population k, the value is 0 with probability 0.5 and 2 with probability 0.5.

without population structure The missing values for a marker j are sampeled from the allele distribution of marker j.

For illustration, 200 entries of the marker matrix are selected, the values saved and these entries are coded as `NA`.

```
> marker <- as.matrix(marker)
> ind1 <- sample(1:nrow(marker), 200)
```

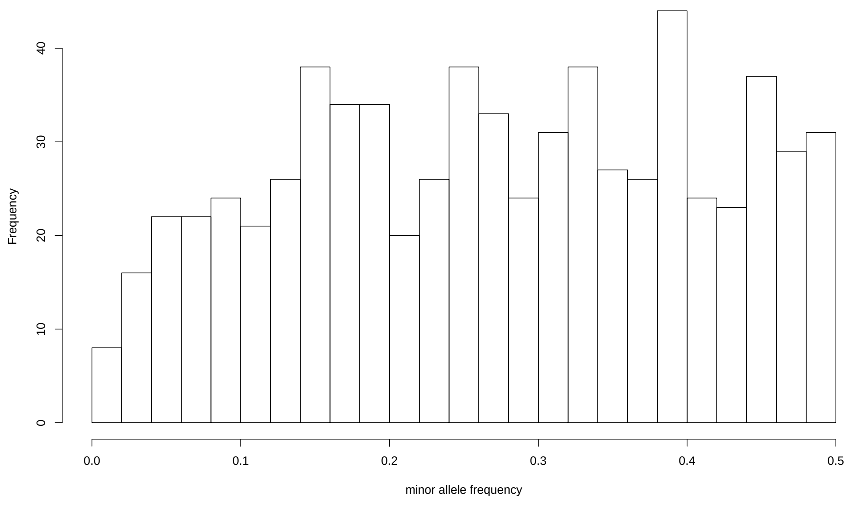


Figure 1: Histogramm of the minor allele frequency of the 696 SNP markers in `maize` data.

```
> ind2 <- sample(1:ncol(marker), 200)
> original <- marker[cbind(ind1, ind2)]
> marker[cbind(ind1, ind2)] <- NA
```

The number of 1250 genotypes in the `maize` data consist of 25 *half sib* families with 50 genotypes in each family. The genotypes are ordered according to the family structure. Recoding of the marker data and imputing of the missing values is done as follows

```
> pop <- rep(1:25, each = 50)
> marker1 <- codeGeno(marker, impute = TRUE, pop)
```

```
approximative run time  0  seconds
 ...
Total number of missing values                : 200
Number of imputations by family structure     : 123
Number of random imputations                  : 77
Approximative fraction of correct imputations : 0.808
```

A report is printed on the screen which informs about the number of imputations performed according to family structure n_F or chance n_R. The approximative fraction of correct imputations is $\frac{n_F+0.5n_R}{n_F+n_R}$. For the simulated data the original values are known. With the following commands the quality of the classification of the missing values is judged:

```
> imputed <- marker1[cbind(ind1, ind2)]
> (t1 <- table(original, imputed))
```

```
        imputed
original   0   2
       0 128  18
       2  24  30
```

The fraction of correct replacements is

```
> sum(diag(t1))/sum(t1)
```

```
[1] 0.79
```

In an analysis of genotypic data it is common to discard marker with a small minor allele frequency and/or many missing values. There are two additional arguments `maf` and `nmiss` for function `codeGeno`. Before recoding the data all marker with more than `nmiss`·100% missing values are discarded. After recoding the maker data only markers with a minor allele frequency > `maf` are returned by the function. By default, no markers are selected by one of both criteria.

Note that missing values in the marker data must be coded as `NA`. Instead of imputing the values `codeGeno` provides the possibility to replace the missing values by a certain value, i.e. 1 which is the expection for the missing values. Different codings of the alleles could easily obtained with simple operations of the resulting data.

3 Pedigree

An important source of information in breeding programs is pedigree information. Especially in plant breeding pedigree is recorded over many generations. The pedigree usually consists of a list of individuals (animals or plants) of the current generation which is the suubject of analysis and their ancestors (for which usually no additional data is available). The pedigree is sorted the generation, beginning with the individuals with unknown parents. An example for an pedigree with five individuals belonging to 4 generations is given below.

ID	Par1	Par2	gener
A	-	-	0
B	-	-	0
C	A	B	1
D	A	C	2
E	D	B	3

Note that unknown parents are coded as "0" in `synbreed` package and generation starts with 0. In `synbreed` exists the class "pedigree", which should be used for handling pedigree information. An object of class ”pedigree” consists of a `data.frame` with at least variables `ID`, `Par1`, `Par2` and `gener`. The function `create.pedigree` creates an object of class ”pedigree” for a given set of individuals and their parents. The generation can be specified by the user or otherwise is computed by the function.

Suppose we have the pedigree struture of the example. This structure is carried into `synbreed` package with the following command:

```
> id <- c("A", "B", "C", "D", "E")
> par1 <- c(0, 0, "A", "A", "D")
> par2 <- c(0, 0, "B", "C", "B")
> ped <- create.pedigree(id, par1, par2)
> ped
```

```
  ID Par1 Par2 gener
1  A    0    0     0
2  B    0    0     0
3  C    A    B     1
4  D    A    C     2
5  E    D    B     3
```

An object of class ”pedigree” could be visualised with generic plotting function for S3 class ”pedigree”.

It is possible to simulate a pedigree structure with function `simul.pedigree`. As arguments, the number of generations to simulate and the number of individuals in each generation has to be specified. By default, random mating is assumed in each generation. As their are no further restrictions, it is possible that inbreeds could be generated when parent 1 equals parent 2. To simulate a pedigree with 6 generations and 4, 6, 7, 9, 10 and 10 individuals in each generation, use

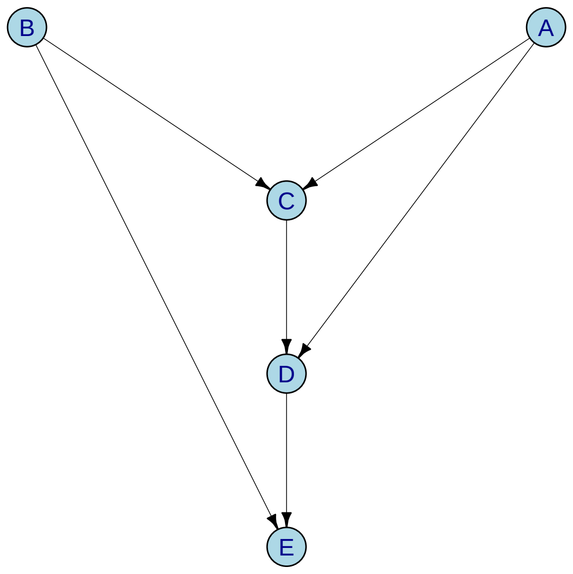
```
> set.seed(123)
> ped.simul <- simul.pedigree(gener = 6, ids = c(4, 6,
+     7, 9, 10, 10))
```

The resulting pedigree is visualized in Figure 2.

4 Relationship matrices

Pedigree information is usually used to set up relationship matrices of a number of individuals. The relationship is constructed by the *expected* fraction of alleles that are identical by descent (IBD) between relatives. Another possibility to set up a relationship is the use of marker data to compute the genomic relationship. This gives the *observed* relationship of individuals.

```
> plot(ped)
```



4.1 Based on Pedigree

The computation of the pedigree based relationship in `synbreed` starts with the gametic relationship. A gamet is the genetic unit that an individual passes to its offspring. Thus the genetic value of an individual at one locus consists of two allelels. Suppose there is an individual C with the parents A and B. Individual C has to allels C1 and C2. The source of allele C1 is Parent A, thus allele C1 could either equal A1 or A2. Allele C2 was inherited of parent B, thus it could be B1 or B2. To compute the gametic relationship start with an expanded table with two alleles for each individual.

This table is converted into the gametic relationship **G** matrix which is of order $2n$, if the number of individuals is n. The diagonal values of **G** are always 1. The off-diagonal values give the probabilty that the two allelse are identical by descent (IBD). If the parents are unknown, it is assumed that they are progeny of a random mating population. In this case the off-diagonals are zero. The other values of the gametic relations are filled in rowwise. The four values which describe the relationship of progeny A with

```
> plot(ped.simul)
```

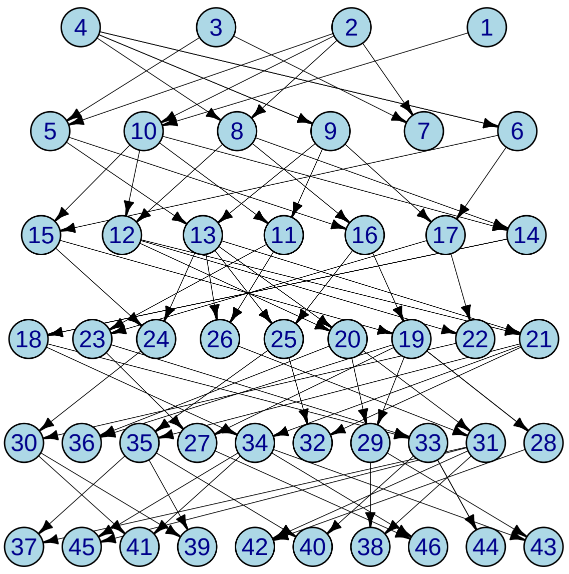


Figure 2: Simulated pedigree structure

parent C are computed as follows

$$(A1, C1) = 0.5 \cdot [(A1, A1) + (A1, A2)]$$

$$(A1, C2) = 0.5 \cdot [(A1, B1) + (A1, B2)]$$

$$(A2, C1) = 0.5 \cdot [(A2, A1) + (A2, A2)]$$

$$(A2, C2) = 0.5 \cdot [(A2, B1) + (A2, B2)]$$

The gametic relationship for a given pedigree is obtained as follows

```
> G <- kinship(ped, ret = "gam")
> G
```

```
      A_1   A_2   B_1   B_2 C_1  C_2   D_1   D_2   E_1   E_2
A_1 1.000 0.000 0.000 0.000 0.5 0.00 0.500 0.250 0.375 0.000
```

ID	Allele	Par1	Par2
A	A1	-	-
A	A2	-	-
B	B1	-	-
B	B1	-	-
C	C1	A1	A2
C	C2	B1	B2
D	D1	A1	A2
D	D2	C1	C2
E	E1	D1	D2
E	E2	B1	B2

```
A_2 0.000 1.000 0.000 0.000 0.5 0.00 0.500 0.250 0.375 0.000
B_1 0.000 0.000 1.000 0.000 0.0 0.50 0.000 0.250 0.125 0.500
B_2 0.000 0.000 0.000 1.000 0.0 0.50 0.000 0.250 0.125 0.500
C_1 0.500 0.500 0.000 0.000 1.0 0.00 0.500 0.500 0.500 0.000
C_2 0.000 0.000 0.500 0.500 0.0 1.00 0.000 0.500 0.250 0.500
D_1 0.500 0.500 0.000 0.000 0.5 0.00 1.000 0.250 0.625 0.000
D_2 0.250 0.250 0.250 0.250 0.5 0.50 0.250 1.000 0.625 0.250
E_1 0.375 0.375 0.125 0.125 0.5 0.25 0.625 0.625 1.000 0.125
E_2 0.000 0.000 0.500 0.500 0.0 0.50 0.000 0.250 0.125 1.000
attr(,"class")
[1] "relationshipMatrix"
```

The resulting object `G` is of class ”relationshipMatrix” which is the general class for all kinds of relationship matrices (gametic relationship, additive and dominance relationship, kinship). An object of class ”relationshipMatrix” is basically a symmetric matrix containing the relationship coefficient of two individuals. Note that the entry of allele 1 and allele 2 of an individual i equals his inbreeding coefficient F_i. For example, the inbreeding coefficent of individual D is

```
> G["D_1", "D_2"]
```

```
[1] 0.25
attr(,"class")
[1] "relationshipMatrix"
```

which is nonzero because individuals A and C which are the parents of D are relatives. Once the gametic relationship is computed, it could be converted in the additive numerator relationship matrix **A** or the dominance relationship matrix **D**. The additive relationship between the individuals A and B is given by

$$0.5 \cdot (G[A1, B1] + G[A1, B2] + G[A2, B1] + G[A2, B2]),$$

where $G[.,.]$ denotes the corresponding value of the gametic relationship matrix $\mathbf{G}$. The additive numerator relationship matrix descripes the relationship between individuals and is of order n. It is typically used in the animal model

$$y_i = \mu + a_i + e_i,$$

where a_i is the additive genetic effect and $\mathbf{a} \sim \mathrm{N}(\mathbf{0}, \mathbf{A}\sigma^2_{\mathbf{a}})$ is assumed. The additive numerator relationship matrix for a given pedigree is obtained as follows

```
> A <- kinship(ped, ret = "add")
> A
```

```
      A     B     C    D     E
A 1.000 0.000 0.500 0.75 0.375
B 0.000 1.000 0.500 0.25 0.625
C 0.500 0.500 1.000 0.75 0.625
D 0.750 0.250 0.750 1.25 0.750
E 0.375 0.625 0.625 0.75 1.125
attr(,"class")
[1] "relationshipMatrix"
```

Note that the diagonals of $\mathbf{A}$ are $1 + F_i$. Sometimes the kinship matrix is required which is half of the additive numerator relationship matrix. It is obtained by

```
> K <- kinship(ped, ret = "kin")
```

Additionally it is possible to derive the dominance relationship matrix $\mathbf{D}$ out of $\mathbf{G}$. The dominance is needed in the non-additive animal model

$$y = \mu + a_i + a_j + d_{ij} + e_i,$$

where a_i and a_j are the additive genetic effects of alleles i and j and d_{ij} is the dominance (interaction) effect of alleles i and j with $\mathbf{d} \sim \mathrm{N}(\mathbf{0}, \mathbf{D}\sigma^2_{\mathbf{d}})$. The dominance relationship matrix for the example is obtained as

```
> D <- kinship(ped, ret = "dom")
> D
```

```
     A     B    C       D        E
A 1.00 0.000 0.00 0.25000 0.000000
B 0.00 1.000 0.00 0.00000 0.125000
C 0.00 0.000 1.00 0.25000 0.250000
D 0.25 0.000 0.25 1.06250 0.156250
E 0.00 0.125 0.25 0.15625 1.015625
attr(,"class")
[1] "relationshipMatrix"
```

Heigher order interactions in the non-additive animal modal as addidive-additive, additive-dominance or dominance-dominance variance-covariance matrices can be computes as

```
> (AA <- A * A)
```

```
         A        B        C      D        E
A 1.000000 0.000000 0.250000 0.5625 0.140625
B 0.000000 1.000000 0.250000 0.0625 0.390625
C 0.250000 0.250000 1.000000 0.5625 0.390625
D 0.562500 0.062500 0.562500 1.5625 0.562500
E 0.140625 0.390625 0.390625 0.5625 1.265625
attr(,"class")
[1] "relationshipMatrix"
```

```
> (AD <- A * D)
```

```
       A        B       C         D         E
A 1.0000 0.000000 0.00000 0.1875000 0.0000000
B 0.0000 1.000000 0.00000 0.0000000 0.0781250
C 0.0000 0.000000 1.00000 0.1875000 0.1562500
D 0.1875 0.000000 0.18750 1.3281250 0.1171875
E 0.0000 0.078125 0.15625 0.1171875 1.1425781
attr(,"class")
[1] "relationshipMatrix"
```

```
> (DD <- D * D)
```

```
       A        B      C          D          E
A 1.0000 0.000000 0.0000 0.06250000 0.00000000
B 0.0000 1.000000 0.0000 0.00000000 0.01562500
C 0.0000 0.000000 1.0000 0.06250000 0.06250000
D 0.0625 0.000000 0.0625 1.12890625 0.02441406
E 0.0000 0.015625 0.0625 0.02441406 1.03149414
attr(,"class")
[1] "relationshipMatrix"
```

4.2 Based on marker data

4.3 Doubled haploid (DH)- lines

In plant breeding doubled haploid lines are common. DH lines are fully inbreed and thus have an inbreeding coefficient of 1. This has to be taken into account, when the relationship matrix in a pedigree with DH lines is computed. As an example the `maize` data is taken.

```
> data(maize)
> head(maize.ped)

  ID Par1 Par2 DH
1  1    0    0  1
2  2    0    0  1
3  3    0    0  1
4  4    0    0  1
5  5    0    0  1
6  6    0    0  1
```

First, the additive numerator relationship matrix is computed. There are 1276 DH lines and 25 non DH lines in the pedigree. For DH lines, it is necassary to set the inbreeding coefficient on 1. An argument `DH` is available for function `kinship` where for each individual in the pedigree it specified whether this is a DH line or not. This information is available for the `maize` data. To obtain the addtive numerator relationship matrix of the first 100 genotypes, use

```
> ped.maize <- create.pedigree(maize.ped$ID, maize.ped$Par1,
+     maize.ped$Par2)
> A.maize100 <- kinship(ped.maize[1:100, ], DH = maize.ped$DH[1:100],
+     ret = "add")
```

4.4 Visualisation of relationship matrices

As in most cases a relationship matrix is to big to print it on the screen. Thus there are two possibilites for visualisation of an object of class "relationshipMatrix" in `synbreed` package. A `summary` method is defined which gives the important characteristics of an relationship matrix. Use

```
> summary(A.maize100)

Dimension         : 100 x 100
Rank              : 75
Range             : 0 -- 2
# of unique values: 5
```

to get the summary for the pedigree based additive relationship matrix of the `maize` data. Another possibility is the `plot` method which could be applied to an object of class "relationshipMatrix". This gives a heatmap of the entries of the relationship matrix

5 Acknowledgements

This research was funded by the German Federal Ministry of Education and Research (BMBF) within the AgroClustEr *Synbreed Synergistic plant and animal breeding*.

References

vanRaden, P. (2008). Efficient methods to compute genomic predictions. *Journal of Dairy Science*, 91:4414–4423.