Peter von Rohr
Institute of Agricultural Sciences
D-USYS
ETH Zurich

751-7602-00 V
Solutions for Exam in
Livestock Breeding
and Genomics
Fall Semester 2024

Date: 2024-12-20

Name:

Legi-Nr:

Problem	Maximum Number of Points	Number of Points Reached
1	26	
2	18	
3	21	
4	42	
5	50	
Total	157	

Questions in German are in italics

Problem 1: One Locus Model

Given are two datasets from a one-locus-model of two different populations showing genotypes and response values of a quantitative trait.

Gegeben sind zwei Datensätze eines Ein-Lokus-Modells für zwei verschiedene Populationen. Der Datensatz umfasst Genotypen und Werte als Zielgrösse eines quantitativen Merkmals.

Table 1: Population A

Animal	Genotype	Response
1	1	3.79
2	1	-15.22
3	2	93.09
4	2	62.72
5	0	-85.43
6	1	-6.23
7	1	18.68
8	0	-72.51
9	1	-7.62
10	2	85.32
11	1	-1.63
12	1	17.04
13	0	-82.63
14	1	-7.51
15	1	17.96
16	0	-52.86
17	0	-65.37
18	0	-74.57
19	0	-55.05
20	1	13.68
21	0	-58.26
22	0	-56.12
23	0	-57.68
24	1	10.32
25	2	51.21
26	0	-69.20
27	0	-108.64
28	2	74.17
29	0	-50.38
30	0	-69.28

Table 2: Population B

Animal	Genotype	Response
1	0	-19.93
2	1	54.83
3	0	-24.48
4	1	13.40
5	0	3.06
6	1	4.42
7	0	-33.18
8	1	20.48
9	0	-51.38
10	2	16.49
11	0	10.99
12	1	13.13
13	1	25.40
14	0	-7.18
15	1	23.62
16	2	23.08
17	0	-50.22
18	1	48.13
19	1	2.86
20	1	8.22
21	1	30.31
22	1	7.88
23	1	-8.30
24	1	4.89
25	1	24.62
26	2	22.83
27	0	-41.49
28	1	7.63
29	0	-28.48
30	0	0.61

The datasets are available from

```
## https://charlotte-ngs.github.io/lbgfs2024/data/exam_p1_popA.csv

## https://charlotte-ngs.github.io/lbgfs2024/data/exam_p1_popB.csv
```

a) Compute the following quantities for both populations

- Minor-Allele-Frequency (frequency of the less frequent allele)
- Genotypic values a and d
- Breeding values for all animals based on the one-locus model, assuming the Hardy-Weinberg equilibrium for genotype frequencies
- Dominance deviations for all animals, assuming the Hardy-Weinberg equilibrium for genotype frequencies

Berechnen Sie die folgenden Grössen für beide Populationen

- Minor-Allele-Frequency (*Frequenz des selteneren Allels*)
- *Genotypische Werte a und d*
- *Zuchtwerte aller Tiere basierend auf dem Ein-Lokus-Modell, wobei das Hardy-Weinberg Gleichgewicht für die Genotypenfrequenzen angenommen wird.*
- *Dominanzabweichung aller Tiere, wobei das Hardy-Weinberg Gleichgewicht für die Genotypenfrequenzen angenommen wird.*

18

Solution

- Minor-Allele-Frequency (frequency of the less frequent allele)

```
# function to compute minor-allele-frequency
get_maf <- function(pvec_genotype_code){
  # check whether, codes are on 0-1-2 coding
  vec_genotype_code <- pvec_genotype_code
  if (min(vec_genotype_code) < 0){
    vec_genotype_code <- vec_genotype_code + 1
  }
  # get maf via mean/2
  return(mean(vec_genotype_code)/2)
}
```

Population A

```
tbl_pop_A <- readr::read_delim(s_p1_data_path_popA, delim = ",")
maf_A <- get_maf(pvec_genotype_code = tbl_pop_A$Genotype)
maf_A
```

```
## [1] 0.35
```

Population B

```
tbl_pop_B <- readr::read_delim(s_p1_data_path_popB, delim = ",")
maf_B <- get_maf(pvec_genotype_code = tbl_pop_B$Genotype)
maf_B
```

```
## [1] 0.3666667
```

- Genotypic values a and d

```r
# function to compute genotypic values
get_genotype_values <- function(ptbl_one_locus,
                                ps_colname_genotype = "Genotype",
                                ps_colname_response = "Response"){
  # homozygous genotype value a
  tbl_hom_pos <- ptbl_one_locus[ptbl_one_locus[[ps_colname_genotype]] == 2,]
  n_group_mean_hom_pos <- mean(tbl_hom_pos[[ps_colname_response]])
  tbl_hom_neg <- ptbl_one_locus[ptbl_one_locus[[ps_colname_genotype]] == 0,]
  n_group_mean_hom_neg <- mean(tbl_hom_neg[[ps_colname_response]])
  n_value_a <- (n_group_mean_hom_pos - n_group_mean_hom_neg) / 2
  # heterozygous genotype value d
  tbl_het <- ptbl_one_locus[ptbl_one_locus[[ps_colname_genotype]] == 1,]
  n_group_mean_het <- mean(tbl_het[[ps_colname_response]])
  n_value_d <- n_group_mean_het - n_group_mean_hom_neg - n_value_a
  # return result
  return(list(a = n_value_a,
              d = n_value_d))
}
```

Population A

```r
n_genotype_value_popA <- get_genotype_values(ptbl_one_locus = tbl_pop_A)
n_genotype_value_popA
```

```
## $a
## [1] 70.86457
##
## $d
## [1] 1.495299
```

Population B

```r
n_genotype_value_popB <- get_genotype_values(ptbl_one_locus = tbl_pop_B)
n_genotype_value_popB
```

```
## $a
## [1] 21.38545
##
## $d
## [1] 18.18045
```

- Breeding values for all animals based on the one-locus model

```r
# function to compute breeding values
get_breeding_values <- function(ptbl_one_locus){
  # get maf
  n_maf <- get_maf(pvec_genotype_code = ptbl_one_locus$Genotype)
  # get genotypic values
  l_geno_val <- get_genotype_values(ptbl_one_locus = ptbl_one_locus)
  # compute alpha
  n_alpha <- l_geno_val$a + (1-2*n_maf) * l_geno_val$d
```

```
  # compute breeding values for three genotypes
  vec_bv_geno <- c(-2 * n_maf * n_alpha,
                   (1 - 2 * n_maf) * n_alpha,
                   2 * (1-n_maf) * n_alpha)
  # set breeding values
  tbl_bv_all_ani <- tibble::tibble(Animal = ptbl_one_locus$Animal,
                                   BV = rep(NA, nrow(ptbl_one_locus)))
  for (idx in 1:nrow(tbl_bv_all_ani)){
    tbl_bv_all_ani$BV[idx] <- vec_bv_geno[ptbl_one_locus$Genotype[idx] + 1]
  }
  return(tbl_bv_all_ani)
}
```

Population A

```
tbl_bv_popA <- get_breeding_values(ptbl_one_locus = tbl_pop_A)
print(tbl_bv_popA, n = nrow(tbl_bv_popA))
```

```
## # A tibble: 30 x 2
##    Animal    BV
##     <dbl> <dbl>
##  1      1  21.4
##  2      2  21.4
##  3      3  92.7
##  4      4  92.7
##  5      5 -49.9
##  6      6  21.4
##  7      7  21.4
##  8      8 -49.9
##  9      9  21.4
## 10     10  92.7
## 11     11  21.4
## 12     12  21.4
## 13     13 -49.9
## 14     14  21.4
## 15     15  21.4
## 16     16 -49.9
## 17     17 -49.9
## 18     18 -49.9
## 19     19 -49.9
## 20     20  21.4
## 21     21 -49.9
## 22     22 -49.9
## 23     23 -49.9
## 24     24  21.4
## 25     25  92.7
## 26     26 -49.9
## 27     27 -49.9
## 28     28  92.7
## 29     29 -49.9
## 30     30 -49.9
```

Population B

```r
tbl_bv_popB <- get_breeding_values(ptbl_one_locus = tbl_pop_B)
print(tbl_bv_popB, n = nrow(tbl_bv_popB))
```

```
## # A tibble: 30 x 2
##    Animal     BV
##     <dbl>  <dbl>
##  1      1 -19.2
##  2      2   7.00
##  3      3 -19.2
##  4      4   7.00
##  5      5 -19.2
##  6      6   7.00
##  7      7 -19.2
##  8      8   7.00
##  9      9 -19.2
## 10     10  33.2
## 11     11 -19.2
## 12     12   7.00
## 13     13   7.00
## 14     14 -19.2
## 15     15   7.00
## 16     16  33.2
## 17     17 -19.2
## 18     18   7.00
## 19     19   7.00
## 20     20   7.00
## 21     21   7.00
## 22     22   7.00
## 23     23   7.00
## 24     24   7.00
## 25     25   7.00
## 26     26  33.2
## 27     27 -19.2
## 28     28   7.00
## 29     29 -19.2
## 30     30 -19.2
```

- Dominance deviations for all animals

```r
# compute dominance deviations
get_dom_dev <- function(ptbl_one_locus){
  # get maf
  n_maf <- get_maf(pvec_genotype_code = ptbl_one_locus$Genotype)
  # get genotypic values
  l_geno_val <- get_genotype_values(ptbl_one_locus = ptbl_one_locus)
  # dominance deviation
  vec_dom_dev <- c(-2 * n_maf^2 * l_geno_val$d,
                   2 * n_maf * (1-n_maf) * l_geno_val$d,
                   -2 * (1-n_maf)^2 * l_geno_val$d)

  # set dominance_deviations
  tbl_dd_all_ani <- tibble::tibble(Animal = ptbl_one_locus$Animal,
                                   DD = rep(NA, nrow(ptbl_one_locus)))
```

```
  for (idx in 1:nrow(tbl_dd_all_ani)){
    tbl_dd_all_ani$DD[idx] <- vec_dom_dev[ptbl_one_locus$Genotype[idx] + 1]
  }
  return(tbl_dd_all_ani)
}
```

Population A

```
tbl_dd_popA <- get_dom_dev(ptbl_one_locus = tbl_pop_A)
print(tbl_dd_popA, n = nrow(tbl_dd_popA))
```

```
## # A tibble: 30 x 2
##    Animal     DD
##     <dbl>  <dbl>
##  1      1  0.680
##  2      2  0.680
##  3      3 -1.26
##  4      4 -1.26
##  5      5 -0.366
##  6      6  0.680
##  7      7  0.680
##  8      8 -0.366
##  9      9  0.680
## 10     10 -1.26
## 11     11  0.680
## 12     12  0.680
## 13     13 -0.366
## 14     14  0.680
## 15     15  0.680
## 16     16 -0.366
## 17     17 -0.366
## 18     18 -0.366
## 19     19 -0.366
## 20     20  0.680
## 21     21 -0.366
## 22     22 -0.366
## 23     23 -0.366
## 24     24  0.680
## 25     25 -1.26
## 26     26 -0.366
## 27     27 -0.366
## 28     28 -1.26
## 29     29 -0.366
## 30     30 -0.366
```

Population B

```
tbl_dd_popB <- get_dom_dev(ptbl_one_locus = tbl_popB)
print(tbl_dd_popB, n = nrow(tbl_dd_popB))
```

```
## # A tibble: 30 x 2
##    Animal     DD
```

```
##     <dbl>  <dbl>
##  1      1  -4.89
##  2      2   8.44
##  3      3  -4.89
##  4      4   8.44
##  5      5  -4.89
##  6      6   8.44
##  7      7  -4.89
##  8      8   8.44
##  9      9  -4.89
## 10     10 -14.6
## 11     11  -4.89
## 12     12   8.44
## 13     13   8.44
## 14     14  -4.89
## 15     15   8.44
## 16     16 -14.6
## 17     17  -4.89
## 18     18   8.44
## 19     19   8.44
## 20     20   8.44
## 21     21   8.44
## 22     22   8.44
## 23     23   8.44
## 24     24   8.44
## 25     25   8.44
## 26     26 -14.6
## 27     27  -4.89
## 28     28   8.44
## 29     29  -4.89
## 30     30  -4.89
```

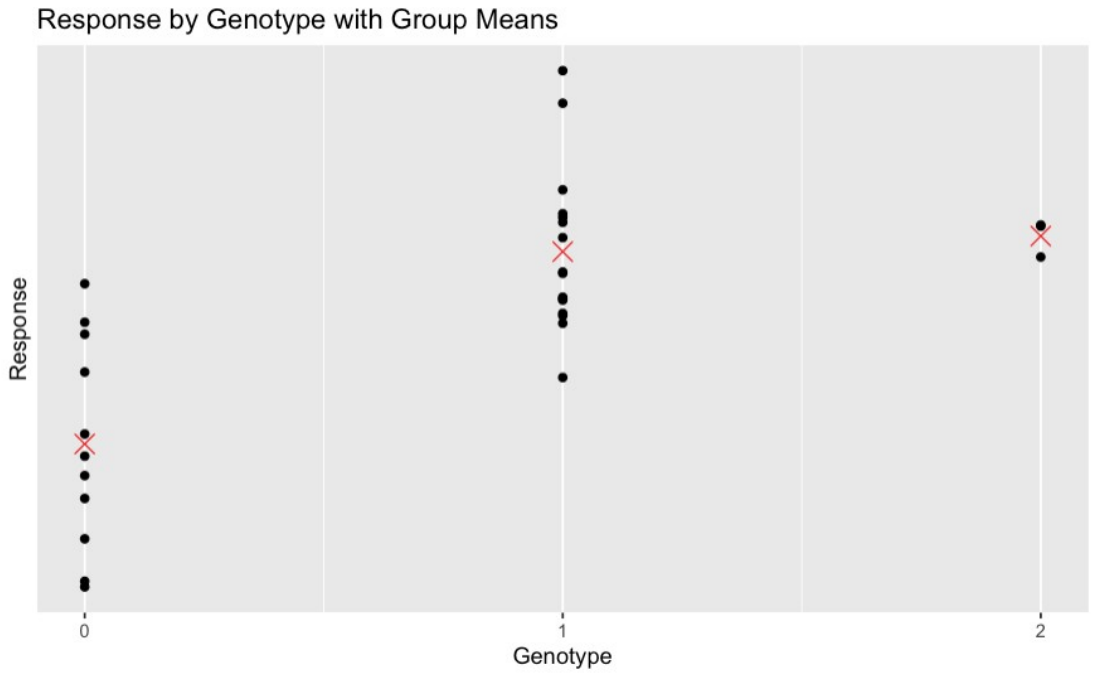
b) Which of the following plots belongs to which population? Insert the genotypic values a and d into the plots.

Welcher der folgenden Plots gehört zu welcher Population? Tragen Sie die genotypischen Werte a und d in die Plots ein.

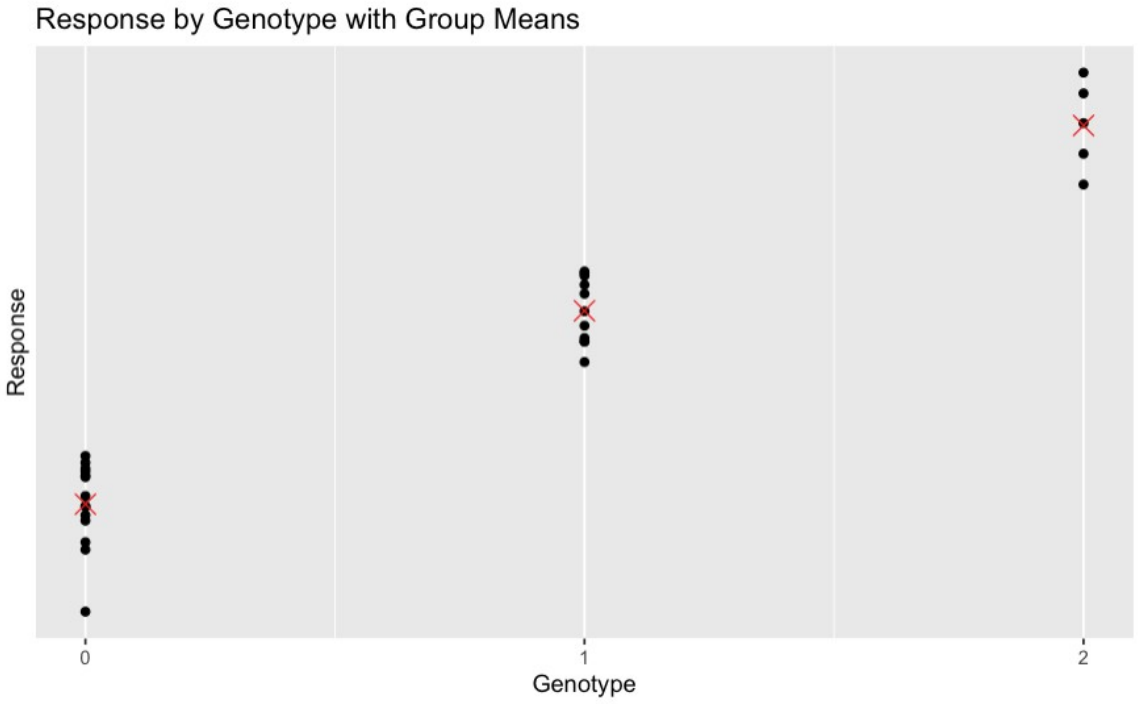
6

Solution

Population:

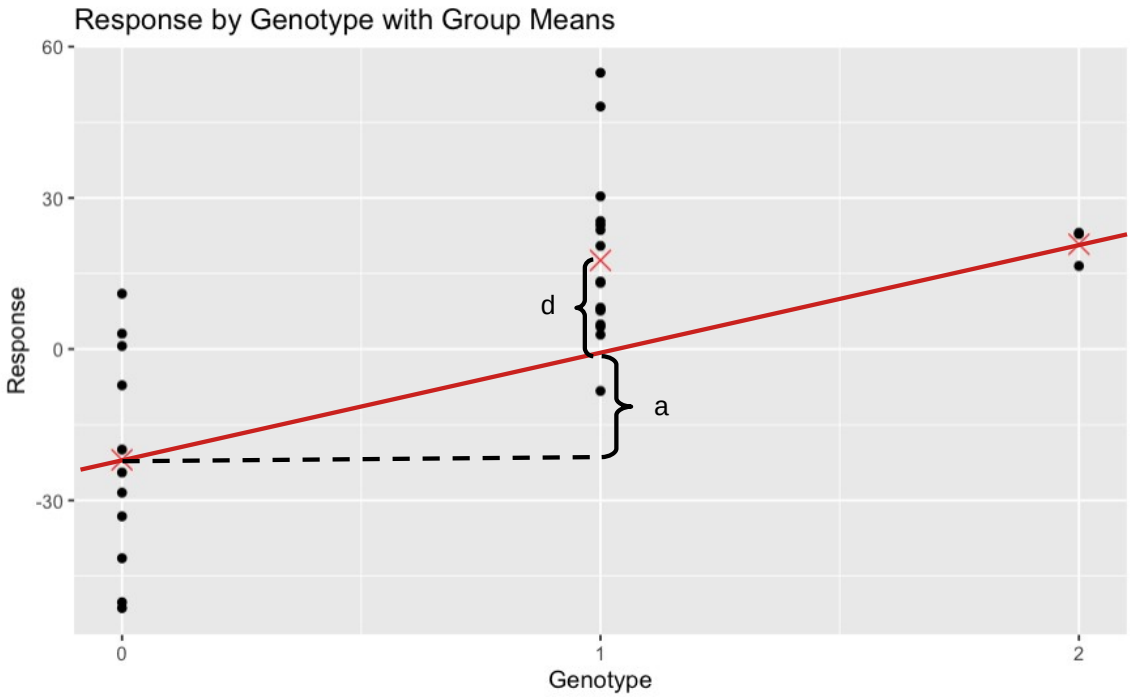


Population:

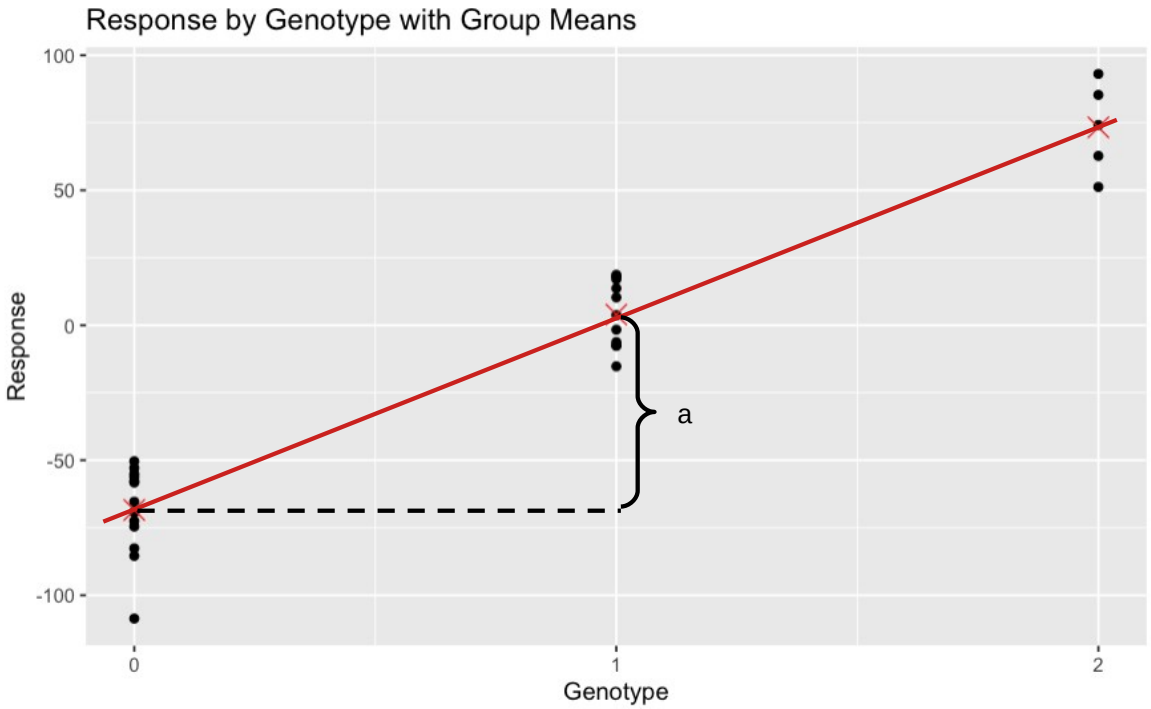

Response by Genotype with Group Means
Response
0
1
2
Genotype

Solution

Population: B



Population: A



d can be assumed to be 0

c) When using the marker effect model to predict genomic breeding values, it is assumed that alleles have a purely additive effect on the response variable. For which the above shown population (A or B) is this assumption better met?

Bei der Verwendung eines Markereffektsmodells bei der Schätzung von genomischen Zuchtwerten wird angenommen, dass die Allele einen rein additiven Effekt auf die Zielgrösse haben. Für welche der beiden Populationen (A oder B) ist diese Annahme besser erfüllt?

2

Solution

Population A has a d value that is much closer to 0

Problem 2: Numerator Relationship Matrix

Given is the following pedigree.

Gegeben ist der Folgende Stammbaum

Animal	Sire	Dam
4	2	1
5	2	1
6	2	3
7	2	3
8	2	3
9	5	6
10	7	4
11	7	9
12	8	11

The pedigree is available from

```
## https://charlotte-ngs.github.io/lbgfs2024/data/exam_pedigree_p2.csv
```

a) Compute the additive numerator relationship matrix A for the above given pedigree.
Berechnen Sie die additiv genetische Verwandtschaftsmatrix A auf für den oben gegebenen Stammbaum.

6

Solution

- Read the pedigree

```
tbl_ped_p2 <- readr::read_delim(s_ped_p2, delim = ",")
```

- Find founder animals

```
# function to get founder animals
get_founders <- function(ptbl_ped){
  # sire founders
  vec_sire_fnd <- setdiff(ptbl_ped$Sire, ptbl_ped$Animal)
  # dam founders
  vec_dam_fnd <- setdiff(ptbl_ped$Dam, ptbl_ped$Animal)
  # combine
  vec_founders <- c(vec_sire_fnd, vec_dam_fnd)
  # remove NA
  vec_founders <- vec_founders[!is.na(vec_founders)]
  # order
  vec_founders <- vec_founders[order(vec_founders)]
  # return
  return(vec_founders)
}
```

For the given pedigree

```
vec_fnds <- get_founders(ptbl_ped = tbl_ped_p2)
vec_fnds
```

```
## [1] 1 2 3
```

- Augment pedigree

```
# add founders to pedigree
augment_pedigree <- function(ptbl_ped){
  # get founders
  vec_founders <- get_founders(ptbl_ped = ptbl_ped)
  n_nr_founders <- length(vec_founders)
  # add founder records
  tbl_aug_ped <- dplyr::bind_rows(
    tibble::tibble(Animal = vec_founders,
                   Sire = rep(NA, n_nr_founders),
                   Dam = rep(NA, n_nr_founders)),
    ptbl_ped)
  # return result
  return(tbl_aug_ped)
}
```

The augmented pedigree is

```
tbl_ped_p2_aug <- augment_pedigree(ptbl_ped = tbl_ped_p2)
tbl_ped_p2_aug
```

```
## # A tibble: 12 x 3
##    Animal  Sire   Dam
##     <dbl> <dbl> <dbl>
##  1      1    NA    NA
##  2      2    NA    NA
##  3      3    NA    NA
##  4      4     2     1
##  5      5     2     1
##  6      6     2     3
##  7      7     2     3
##  8      8     2     3
##  9      9     5     6
## 10     10     7     4
## 11     11     7     9
## 12     12     8    11
```

- Numerator relationship matrix

```
pem_p2 <- pedigreemm::pedigree(sire = tbl_ped_p2_aug$Sire,
                               dam = tbl_ped_p2_aug$Dam,
                               label = tbl_ped_p2_aug$Animal)
mat_A_p2 <- pedigreemm::getA(pem_p2)
mat_A_p2
```

```
## 12 x 12 sparse Matrix of class "dsCMatrix"
##
## 1  1.0000 .   .      0.50000 0.50000 .       .       .       0.2500 0.2500
## 2  .      1.0 .      0.50000 0.50000 0.50000 0.50000 0.50000 0.5000 0.5000
## 3  .      .   1.0000 .       .       0.50000 0.50000 0.50000 0.2500 0.2500
## 4  0.5000 0.5 .      1.00000 0.50000 0.25000 0.25000 0.25000 0.3750 0.6250
## 5  0.5000 0.5 .      0.50000 1.00000 0.25000 0.25000 0.25000 0.6250 0.3750
## 6  .      0.5 0.5000 0.25000 0.25000 1.00000 0.50000 0.50000 0.6250 0.3750
## 7  .      0.5 0.5000 0.25000 0.25000 0.50000 1.00000 0.50000 0.3750 0.6250
## 8  .      0.5 0.5000 0.25000 0.25000 0.50000 0.50000 1.00000 0.3750 0.3750
## 9  0.2500 0.5 0.2500 0.37500 0.62500 0.62500 0.37500 0.37500 1.1250 0.3750
## 10 0.2500 0.5 0.2500 0.62500 0.37500 0.37500 0.62500 0.37500 0.3750 1.1250
## 11 0.1250 0.5 0.3750 0.31250 0.43750 0.56250 0.68750 0.43750 0.7500 0.5000
## 12 0.0625 0.5 0.4375 0.28125 0.34375 0.53125 0.59375 0.71875 0.5625 0.4375
##
## 1  0.1250 0.06250
## 2  0.5000 0.50000
## 3  0.3750 0.43750
## 4  0.3125 0.28125
## 5  0.4375 0.34375
## 6  0.5625 0.53125
## 7  0.6875 0.59375
## 8  0.4375 0.71875
## 9  0.7500 0.56250
## 10 0.5000 0.43750
## 11 1.1875 0.81250
## 12 0.8125 1.21875
```

b) Given is the following inverse numerator relationship matrix A^{-1}. Determine which animal is a founder and which animal has which parents based on the given matrix. Write down the parent-offspring relationship in tabular form and add the missing animal IDs to the graphical representation of the pedigree shown below.

Gegeben ist die folgende Inverse A^{-1} einer additive genetischen Verwandtschaftsmatrix. Bestimmen Sie den zu dieser Matrix welche Tiere Gründertiere sind und welche Tiere welche Eltern haben. Schreiben Sie die Eltern-Nachkommen-Beziehungen in Tabellenform auf und tragen Sie die fehlenden Tier-IDs in die grafische Repräsentation des Stammbaums ein.

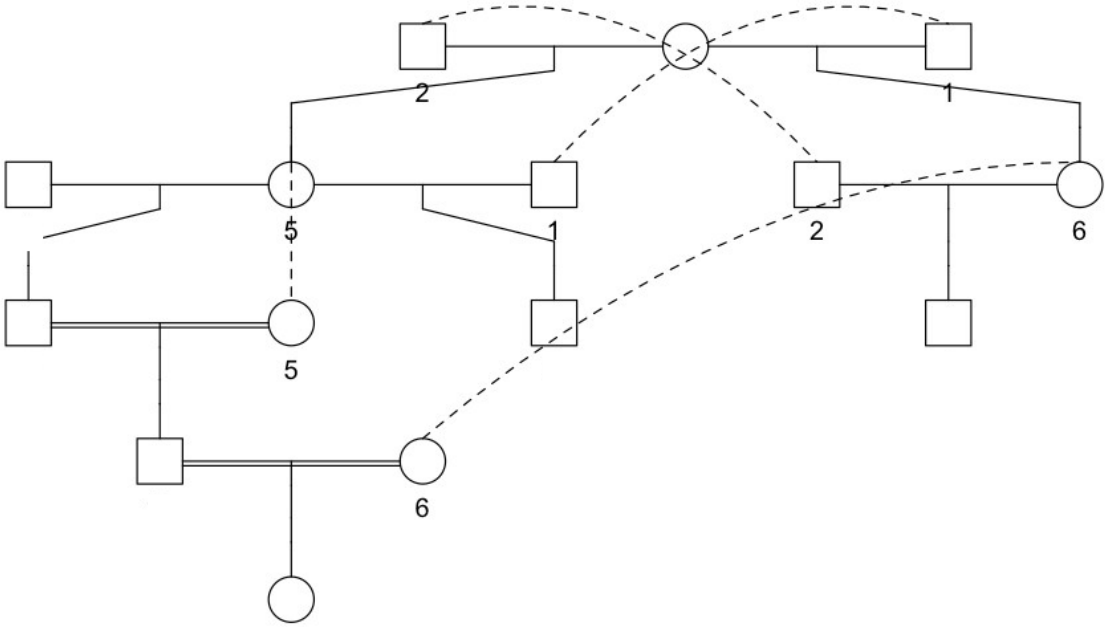
12

$$A^{-1} = \begin{bmatrix} 2 & 0 & 0.5 & 0 & 0.5 & -1 & 0 & -1 & 0 & 0 & 0 \\ 0 & 2 & 0.5 & 0 & -1 & 0.5 & 0 & 0 & -1 & 0 & 0 \\ 0.5 & 0.5 & 2 & 0 & -1 & -1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1.5 & 0.5 & 0 & -1 & 0 & 0 & 0 & 0 \\ 0.5 & -1 & -1 & 0.5 & 3.5 & 0 & -0.5 & -1 & 0 & -1 & 0 \\ -1 & 0.5 & -1 & 0 & 0 & 3.071429 & 0 & 0 & -1 & 0.571429 & -1.142857 \\ 0 & 0 & 0 & -1 & -0.5 & 0 & 2.5 & 0 & 0 & -1 & 0 \\ -1 & 0 & 0 & 0 & -1 & 0 & 0 & 2 & 0 & 0 & 0 \\ 0 & -1 & 0 & 0 & 0 & -1 & 0 & 0 & 2 & 0 & 0 \\ 0 & 0 & 0 & 0 & -1 & 0.571429 & -1 & 0 & 0 & 2.571429 & -1.142857 \\ 0 & 0 & 0 & 0 & 0 & -1.142857 & 0 & 0 & 0 & -1.142857 & 2.285714 \end{bmatrix}$$

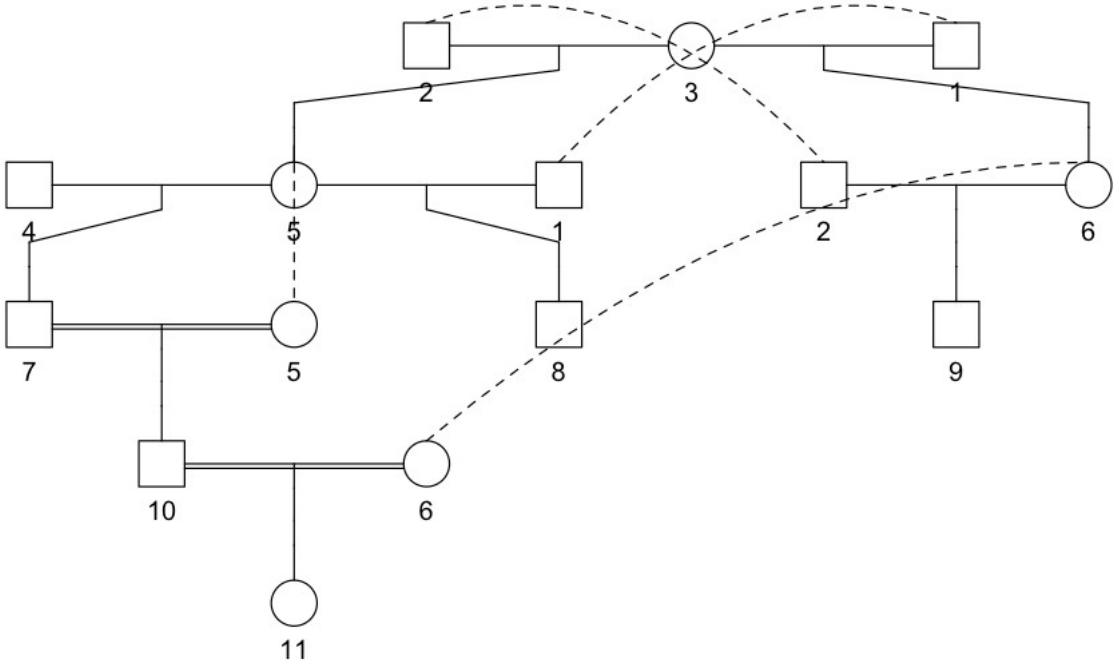
The matrix is available from

```
## https://charlotte-ngs.github.io/lbgfs2024/data/exam_inv_num_rel_mat_p2.csv
```

Solution



Solution



Setting up the pedigree via Matrix decomposition of A. The cholesky decomposition of A is given by

$$A = R \cdot R^T$$

where R is a lower triangular matrix. The matrix R is decomposed further into

$$R = L \cdot S$$

where S is a diagonal matrix with the same diagonal elements as R. Matrix L is a lower-triangular matrix with diagonal elements all equal to 1 and is computed from

$$L = R \cdot S^{-1}$$

The offspring-parent relationships can be seen from the matrix P which is

$$P = I - L^{-1} = I - (R \cdot S^{-1})^{-1} = I - S \cdot R^{-1}$$

```
# read matrix into a tibble
s_mat_p2 <- file.path(s_data_root, "exam_inv_num_rel_mat_p2.csv")
tbl_mat_p2 <- readr::read_delim(s_mat_p2, delim = ",")
# convert to matrix
mat_A_inv <- as.matrix(tbl_mat_p2)
# compute inverse
```

```
mat_A <- solve(mat_A_inv)
# cholesky decomposition
mat_R <- t(chol(mat_A))
mat_R_inv <- solve(mat_R)
# matrix S
mat_S <- diag(diag(mat_R), nrow = nrow(mat_R))
# matrix P
mat_P <- diag(nrow = nrow(mat_A)) - mat_S %*% mat_R_inv
mat_P_r <- round(mat_P, digits = 8)
```

The following matrix shows parent-offspring relations with entries being equal to 0.5.

```
mat_P_r <- round(mat_P, digits = 8)
mat_P_r
```

```
##       [,1] [,2] [,3] [,4] [,5] [,6] [,7] [,8] [,9] [,10] [,11]
##  [1,]  0.0  0.0  0.0  0.0  0.0  0.0  0.0    0    0   0.0     0
##  [2,]  0.0  0.0  0.0  0.0  0.0  0.0  0.0    0    0   0.0     0
##  [3,]  0.0  0.0  0.0  0.0  0.0  0.0  0.0    0    0   0.0     0
##  [4,]  0.0  0.0  0.0  0.0  0.0  0.0  0.0    0    0   0.0     0
##  [5,]  0.0  0.5  0.5  0.0  0.0  0.0  0.0    0    0   0.0     0
##  [6,]  0.5  0.0  0.5  0.0  0.0  0.0  0.0    0    0   0.0     0
##  [7,]  0.0  0.0  0.0  0.5  0.5  0.0  0.0    0    0   0.0     0
##  [8,]  0.5  0.0  0.0  0.0  0.5  0.0  0.0    0    0   0.0     0
##  [9,]  0.0  0.5  0.0  0.0  0.0  0.5  0.0    0    0   0.0     0
## [10,]  0.0  0.0  0.0  0.0  0.5  0.0  0.5    0    0   0.0     0
## [11,]  0.0  0.0  0.0  0.0  0.0  0.5  0.0    0    0   0.5     0
```

The tabular form of the pedigree can be produced by

```
n_nr_ani <- nrow(mat_A_inv)
for (idx in 1:n_nr_ani){
  vec_par_idx <- which(mat_P_r[idx,] > 0, arr.ind = T)
  if (length(vec_par_idx) > 0){
    cat(" * Animal", idx, " has parents: ", paste0(vec_par_idx, collapse = " and "), "\n")
  } else {
    cat (" * Animal ", idx, " is a founder \n")
  }
}
```

```
##  * Animal  1  is a founder
##  * Animal  2  is a founder
##  * Animal  3  is a founder
##  * Animal  4  is a founder
##  * Animal 5  has parents:  2 and 3
##  * Animal 6  has parents:  1 and 3
##  * Animal 7  has parents:  4 and 5
##  * Animal 8  has parents:  1 and 5
##  * Animal 9  has parents:  2 and 6
##  * Animal 10  has parents:  5 and 7
##  * Animal 11  has parents:  6 and 10
```

Problem 3: Inbreeding

Given is the pedigree shown below.

Gegeben sei der nachfolgende Stammbaum.

Animal	Sire	Dam
4	2	1
5	2	1
6	2	1
7	2	3
8	2	4
9	7	8
10	9	6

The pedigree is available from

```
## https://charlotte-ngs.github.io/lbgfs2024/data/exam_pedigree_p3.csv
```

a) Compute inbreeding coefficients for all animals in the above shown pedigree. Use the results to fill out the following table. Indicate in the second column whether a given animal is inbred or not. Use the third column of the table to list the inbreeding coefficients of all animals.

Berechnen Sie die Inzuchtkoeffizienten für alle Tiere im oben gezeigten Stammbaum. Verwenden Sie die nachfolgende Tabelle für die Resultate. In der zweiten Kolonne können Sie angeben, ob ein bestimmtes Tier ingezüchtet ist oder nicht. In der dritten Kolonnen tragen Sie die Inzuchtkoeffizienten aller Tiere ein.

5

Solution

Animal	Inbred	Inbreeding Coefficient
1		
2		
3		
4		
5		
6		
7		
8		
9		
10		

Solution

- Augment given pedigree
- Pedigreemm-Format

```
pem_p3 <- pedigreemm::pedigree(sire = tbl_ped_p3_aug$Sire,
                               dam = tbl_ped_p3_aug$Dam,
                               label = tbl_ped_p3_aug$Animal)
pem_p3
```

```
##    sire  dam
## 1  <NA> <NA>
## 2  <NA> <NA>
## 3  <NA> <NA>
## 4     2    1
## 5     2    1
## 6     2    1
## 7     2    3
## 8     2    4
## 9     7    8
## 10    9    6
```

- Inbreeding computation

```
vec_inb <- pedigreemm::inbreeding(ped = pem_p3)
```

- Results to table

```
vec_status <- sapply(vec_inb, function(x) ifelse(x > 0, "Y", "N"), USE.NAMES = F)
tbl_inb_sol <- tibble::tibble(Animal = tbl_ped_p3_aug$Animal,
                              Inbred = vec_status,
                             `Inbreeding Coefficient` = vec_inb)
knitr::kable(tbl_inb_sol, booktabs = T, longtable = T)
```

Animal	Inbred	Inbreeding Coefficient
1	N	0.0000
2	N	0.0000
3	N	0.0000
4	N	0.0000
5	N	0.0000
6	N	0.0000
7	N	0.0000
8	Y	0.2500
9	Y	0.1875
10	Y	0.1875

b) For each of the sires (2, 7, 9) shown in the above pedigree, find those mates among all dams shown in the pedigree (1, 3, 4, 6, 8) such that the potential offspring has an inbreeding coefficient smaller than 0.25.

Finde für jeden Vater (2, 7, 9) im oben gezeigten Pedigree unter allen Müttern (1, 3, 4, 6, 8) diese Paarunspartnerinnen so dass der Inzuchtkoeffizient der potentiellen Nachkommen unter einem Grenzwert von 0.25 sind.

16

Solution

- Get numerator relationship matrix of given pedigree. This uses the prepared pedigree in pedigreemm-format.

```
mat_A <- as.matrix(pedigreemm::getA(ped = pem_p3))
mat_A
```

```
##         1      2     3    4      5      6       7       8       9      10
## 1  1.0000 0.0000 0.000 0.50 0.5000 0.5000 0.00000 0.25000 0.12500 0.31250
## 2  0.0000 1.0000 0.000 0.50 0.5000 0.5000 0.50000 0.75000 0.62500 0.56250
## 3  0.0000 0.0000 1.000 0.00 0.0000 0.0000 0.50000 0.00000 0.25000 0.12500
## 4  0.5000 0.5000 0.000 1.00 0.5000 0.5000 0.25000 0.75000 0.50000 0.50000
## 5  0.5000 0.5000 0.000 0.50 1.0000 0.5000 0.25000 0.50000 0.37500 0.43750
## 6  0.5000 0.5000 0.000 0.50 0.5000 1.0000 0.25000 0.50000 0.37500 0.68750
## 7  0.0000 0.5000 0.500 0.25 0.2500 0.2500 1.00000 0.37500 0.68750 0.46875
## 8  0.2500 0.7500 0.000 0.75 0.5000 0.5000 0.37500 1.25000 0.81250 0.65625
## 9  0.1250 0.6250 0.250 0.50 0.3750 0.3750 0.68750 0.81250 1.18750 0.78125
## 10 0.3125 0.5625 0.125 0.50 0.4375 0.6875 0.46875 0.65625 0.78125 1.18750
```

- Set of unique sire and dam ids

```
vec_sire <- unique(tbl_ped_p3$Sire)
(vec_sire <- vec_sire[order(vec_sire)])
```

```
## [1] 2 7 9
```

```
vec_dam <- unique(tbl_ped_p3$Dam)
(vec_dam <- vec_dam[order(vec_dam)])
```

```
## [1] 1 3 4 6 8
```

- Check each potential pair

```
for (sidx in vec_sire){
  vec_dam_mates <- which(mat_A[sidx,vec_dam]/2 < n_inb_limit)
  if (length(vec_dam_mates) > 0){
    cat (" * Mates for sire ", sidx, ": ", paste0(vec_dam[vec_dam_mates], collapse = ", "), "\n")
  } else {
    cat (" * No mates found for sire: ", sidx, "\n")
  }
}
```

```
##  * Mates for sire  2 :  1, 3
##  * Mates for sire  7 :  1, 4, 6, 8
##  * Mates for sire  9 :  1, 3, 6
```

Problem 4: Genomic Breeding Values

Given is the following dataset with genomic information. Use `Weight` as response variable. The variable `Height` is to be modelled as regression covariate. The sex of an animal is to be treated as fixed effect.

Gegeben ist der folgende Datensatz mit genomischer Information. Verwenden Sie 'Weight' als Zielgrösse. Modellieren Sie 'Height' als Regressionscovariable und das Geschlecht des Tieres ist als fixer Effekt zu behandeln.

Animal	Sex	S1	S2	S3	S4	S5	S6	S7	S8	S9	S10	S11	S12	S13	S14	S15	S16	S17	Height	Weight
1	2	1	2	1	0	0	1	1	1	1	1	0	2	0	0	1	0	0	148	411
2	1	1	0	0	2	1	0	0	1	0	1	1	0	1	1	2	0	1	143	400
3	2	1	0	0	1	1	1	1	0	0	1	1	2	1	1	1	0	1	152	461
4	1	2	1	0	1	1	0	1	1	1	0	0	1	1	1	2	0	0	153	525
5	2	0	1	0	1	0	0	1	2	0	1	1	1	1	1	1	0	1	139	326
6	2	2	1	1	1	0	1	1	1	0	1	0	1	1	1	2	0	1	151	494
7	1	0	0	0	2	2	0	1	0	0	1	2	1	1	1	2	0	2	159	603
8	2	1	0	0	1	1	0	0	0	0	0	2	1	1	1	2	0	1	147	377
9	1	2	0	0	1	1	1	1	0	0	1	1	1	0	1	2	0	0	153	489
10	2	1	0	0	2	1	0	0	2	0	2	1	0	2	1	2	0	1	150	475
11	1	1	0	0	1	1	1	1	0	0	1	1	2	2	2	2	0	1	149	565
12	2	1	0	0	2	1	1	1	1	0	2	1	1	1	1	2	0	0	155	544
13	1	2	0	0	2	1	0	1	0	0	1	1	2	2	1	2	0	1	146	515

The dataset is available from

```
## https://charlotte-ngs.github.io/lbgfs2024/data/exam_data_p4.csv
```

a) Use a marker-effects model to predict genomic breeding values for the response variable 'Weight'. Use 'Height' as regression covariable and 'Sex' as fixed effects. Columns S1 to S17 contain marker information which is to be used for the prediction of genomic breeding values. The ratio λ_q is assumed to be 9. Specify the model using a formula and explain all the variables used in the model. Write down expected values and variance-covariance matrices for all random effects. Construct mixed-model equations and obtain solutions for estimated fixed effects and predicted breeding values. Rank all animals according to the predicted genomic breeding values.

   Verwenden Sie ein Marker-Effekt Modell zur Schätzung von genomischen Zuchtwerten für die Zielgrösse 'Weight'. Verwenden Sie 'Height' als Regressionscovariable und 'Sex' als fixen Effekt. Die Kolonnen S1 bis S17 enthalten SNP-Markerinformationen, welche für die Schätzung der genomischen Zuchtwerte verwendet werden soll. Das Verhältnis λ_q betrage 9. Schreiben Sie die Erwartungswerte und die Varianz-Covarianz-Matrizen für alle zufälligen Effekte auf. Stellen Sie die Mischmodellgleichungen auf und lösen Sie diese, damit Sie Schätzungen für fixe Effekte und Zuchtwerte erhalten. Rangieren Sie die Tiere aufgrund der geschätzten genomischen Zuchtwerte.

20

Solution

The marker effect model is given by

$$y = Xb + Wq + e$$

- with the vectors
  - y: of length n with observations
  - b: fixed effects of `Sex` - differnces, slope and intercept for `BC`
  - q: of length l with random marker effects
  - e: of length n random residual effects
- where n is the number of observations in the dataset and l the number of marker loci.
- with design Matrices X and W linking fixed effects and marker effects to observations, respectively
- Expected values of random components are given as

$$E \begin{bmatrix} y \\ q \\ e \end{bmatrix} = \begin{bmatrix} Xb \\ 0 \\ 0 \end{bmatrix}$$

- Variance-Covariance matrices

$$var \begin{bmatrix} y \\ q \\ e \end{bmatrix} = \begin{bmatrix} V & ZQ & R \\ QZ^T & Q & 0 \\ R & 0 & R \end{bmatrix}$$

- with
  - $var(q) = Q = I\sigma_q^2$, where I is the identity matrix and σ_q^2 is the marker-effects variance,
  - $var(e) = R = I\sigma_e^2$ where I is the identity matrix and σ_e^2 is the residual variance
  - $var(y) = V = ZQZ^T + R$
- Known components in the model are

```
# design matrix X
matX <- model.matrix(lm(Weight ~ Height + as.factor(Sex), data = tbl_data_p4))
attr(matX,"assign") <- NULL
attr(matX,"contrasts") <- NULL
colnames(matX) <- NULL
matX
```

```
##    [,1] [,2] [,3]
## 1     1  148    1
## 2     1  143    0
## 3     1  152    1
## 4     1  153    0
## 5     1  139    1
## 6     1  151    1
## 7     1  159    0
```

```
## 8     1  147    1
## 9     1  153    0
## 10    1  150    1
## 11    1  149    0
## 12    1  155    1
## 13    1  146    0
```

Genotype matrix

```
# matrix W is the genotype matrix in -1/0/1 coding
mat_geno <- as.matrix(tbl_data_p4 %>% select(starts_with("S")) %>% select(-Sex))
matW <- mat_geno - 1
matW
```

```
##       S1 S2 S3 S4 S5 S6 S7 S8 S9 S10 S11 S12 S13 S14 S15 S16 S17
##  [1,]  0  1  0 -1 -1  0  0  0  0   0  -1   1  -1  -1   0  -1  -1
##  [2,]  0 -1 -1  1  0 -1 -1  0 -1   0   0  -1   0   0   1  -1   0
##  [3,]  0 -1 -1  0  0  0  0 -1 -1   0   0   1   0   0   0  -1   0
##  [4,]  1  0 -1  0  0 -1  0  0  0  -1  -1   0   0   0   1  -1  -1
##  [5,] -1  0 -1  0 -1 -1  0  1 -1   0   0   0   0   0   0  -1   0
##  [6,]  1  0  0  0 -1  0  0  0 -1   0  -1   0   0   0   1  -1   0
##  [7,] -1 -1 -1  1  1 -1  0 -1 -1   0   1   0   0   0   1  -1   1
##  [8,]  0 -1 -1  0  0 -1 -1 -1 -1  -1   1   0   0   0   1  -1   0
##  [9,]  1 -1 -1  0  0  0  0 -1 -1   0   0   0  -1   0   1  -1  -1
## [10,]  0 -1 -1  1  0 -1 -1  1 -1   1   0  -1   1   0   1  -1   0
## [11,]  0 -1 -1  0  0  0  0 -1 -1   0   0   1   1   1   1  -1   0
## [12,]  0 -1 -1  1  0  0  0  0 -1   1   0   0   0   0   1  -1  -1
## [13,]  1 -1 -1  1  0 -1  0 -1 -1   0   0   1   1   0   1  -1   0
```

Mixed model equations

```
mat_xtx <- crossprod(matX)
mat_xtw <- crossprod(matX,matW)
mat_wtx <- t(mat_xtw)
mat_wtw_linv <- crossprod(matW) + lambda_q * diag(nrow = ncol(matW))
mat_coef <- rbind(cbind(mat_xtx, mat_xtw),cbind(mat_wtx, mat_wtw_linv))
mat_rhs <- rbind(crossprod(matX, tbl_data_p4$Weight),
                 crossprod(matW, tbl_data_p4$Weight))
mat_sol <- solve(mat_coef, mat_rhs)
```

Solutions for marker effects are

```
(mat_mrk_sol <- mat_sol[(ncol(matX)+1):nrow(mat_sol),])
```

```
##         S1         S2         S3         S4         S5         S6         S7
##  0.9243590 -0.1624761  1.6067004  4.3676559 -1.9520356  2.6867602  4.2445203
##         S8         S9        S10        S11        S12        S13        S14
##  2.0201180 -1.5977329  7.0811180 -3.7726585  6.9420156 13.5294352  5.3134418
##        S15        S16        S17
##  2.8266067  0.0000000  2.6591730
```

The genomic breeding values are computed by multiplying the genotype matrix with the marker solutions

```
(mat_gbv_mem <- crossprod(t(matW), mat_mrk_sol))
```

```
##              [,1]
##  [1,] -13.3654723
##  [2,]  -6.5255249
##  [3,]   5.0754063
##  [4,]  -6.5101274
##  [5,]   0.3520669
##  [6,]  11.0733927
##  [7,]  -1.3489870
##  [8,] -16.8250597
##  [9,] -14.3042518
## [10,]  16.1051463
## [11,]  26.7448899
## [12,]  11.7697163
## [13,]  24.0367028
```

Ranking animals according to genomic breeding values

```
tbl_data_p4$Animal[order(mat_gbv_mem, decreasing = T)]
```

```
##  [1] 11 13 10 12  6  3  5  7  4  2  1  9  8
```

b) Use a breeding-value based genomic BLUP model to predict genomic breeding values for the response variable 'Weight'. Use 'Height' as regression covariable and 'Sex' as fixed effect. Columns S1 to S17 contain marker information which is to be used for the prediction of genomic breeding values. The ratio λ_g is assumed to be 7. Specify the model using a formula and explain all the variables used in the model. Write down expected values and variance-covariance matrices for all random effects. Construct mixed-model equations and obtain solutions for estimated fixed effects and predicted breeding values. Rank all animals according to the predicted genomic breeding values.

Verwenden Sie ein Zuchtwert-basiertes genomisches BLUP Modell für die Schätzung von genomischen Zuchtwerten für die Zielgrösse 'Weight'. Verwenden Sie 'Height' als Regressionscovariable und 'Sex' als fixen Effekt. Die Kolonnen S1 bis S17 enthalten SNP-Markerinformationen, welche für die Schätzung der genomischen Zuchtwerte verwendet werden soll. Das Verhältnis λ_g betrage 7. Schreiben Sie die Erwartungswerte und die Varianz-Covarianz-Matrizen für alle zufälligen Effekte auf. Stellen Sie die Mischmodellgleichungen auf und lösen Sie diese, damit Sie Schätzungen für fixe Effekte und Zuchtwerte erhalten. Rangieren Sie die Tiere aufgrund der geschätzten genomischen Zuchtwerte.

22

Solution

The breeding value based model is

$$y = Xb + Zg + e$$

- with the vectors
  - y: of length n with observations
  - b: fixed effects of `Sex` - differnces, slope and intercept for `Height`
  - g: of length q with random genomic breeding values
  - e: of length n random residual effects
- where n is the number of observations in the dataset and q the number of animals with genotypes.
- with design Matrices X and Z linking fixed effects and genomic breeding values to observations, respectively
- Expected values of random components are given as

$$E \begin{bmatrix} y \\ g \\ e \end{bmatrix} = \begin{bmatrix} Xb \\ 0 \\ 0 \end{bmatrix}$$

- Variance-Covariance matrices

$$var \begin{bmatrix} y \\ g \\ e \end{bmatrix} = \begin{bmatrix} V & ZH & R \\ HZ^T & H & 0 \\ R & 0 & R \end{bmatrix}$$

- with
  - $var(g) = H = G\sigma_g^2$, where G is the genomic relationship matrix and σ_g^2 is the genomic variance,
  - $var(e) = R = I\sigma_e^2$ where I is the identity matrix and σ_e^2 is the residual variance

- $var(y) = V = ZHZ^T + R$

- Known components in the model are

```
# design matrix X
matX <- model.matrix(lm(Weight ~ Height + as.factor(Sex), data = tbl_data_p4))
attr(matX,"assign") <- NULL
attr(matX,"contrasts") <- NULL
colnames(matX) <- NULL
matX
```

```
##    [,1] [,2] [,3]
## 1     1  148    1
## 2     1  143    0
## 3     1  152    1
## 4     1  153    0
## 5     1  139    1
## 6     1  151    1
## 7     1  159    0
## 8     1  147    1
## 9     1  153    0
## 10    1  150    1
## 11    1  149    0
## 12    1  155    1
## 13    1  146    0
```

```
# design matrix Z
n_nr_obs <- nrow(tbl_data_p4)
matZ <- diag(nrow = n_nr_obs)
matZ
```

```
##       [,1] [,2] [,3] [,4] [,5] [,6] [,7] [,8] [,9] [,10] [,11] [,12] [,13]
##  [1,]    1    0    0    0    0    0    0    0    0     0     0     0     0
##  [2,]    0    1    0    0    0    0    0    0    0     0     0     0     0
##  [3,]    0    0    1    0    0    0    0    0    0     0     0     0     0
##  [4,]    0    0    0    1    0    0    0    0    0     0     0     0     0
##  [5,]    0    0    0    0    1    0    0    0    0     0     0     0     0
##  [6,]    0    0    0    0    0    1    0    0    0     0     0     0     0
##  [7,]    0    0    0    0    0    0    1    0    0     0     0     0     0
##  [8,]    0    0    0    0    0    0    0    1    0     0     0     0     0
##  [9,]    0    0    0    0    0    0    0    0    1     0     0     0     0
## [10,]    0    0    0    0    0    0    0    0    0     1     0     0     0
## [11,]    0    0    0    0    0    0    0    0    0     0     1     0     0
## [12,]    0    0    0    0    0    0    0    0    0     0     0     1     0
## [13,]    0    0    0    0    0    0    0    0    0     0     0     0     1
```

- Genomic relationship matrix

```
#' Compute genomic relationship matrix based on data matrix
computeMatGrm <- function(pmatData, pn_max_iter = 10, pn_min_eig_val = 0.0001) {
  matData <- pmatData
  # check the coding, if matData is -1, 0, 1 coded, then add 1 to get to 0, 1, 2 coding
  if (min(matData) < 0) matData <- matData + 1
```

```
  # Allele frequencies, column vector of P and sum of frequency products
  freq <- apply(matData, 2, mean) / 2
  P <- 2 * (freq - 0.5)
  sumpq <- sum(freq*(1-freq))
  # Changing the coding from (0,1,2) to (-1,0,1) and subtract matrix P
  Z <- matData - 1 - matrix(P, nrow = nrow(matData),
                            ncol = ncol(matData),
                            byrow = TRUE)
  # Z%*%Zt is replaced by tcrossprod(Z)
  matG_result <- tcrossprod(Z)/(2*sumpq)
  # check for positive definiteness
  n_min_eig_matG_result <- min(eigen(matG_result, only.values = TRUE)$values)
  n_iter_idx <- 0
  while (n_min_eig_matG_result < pn_min_eig_val & n_iter_idx < pn_max_iter){
    matG_result <- matG_result + 0.01 * diag(nrow = nrow(matG_result))
    n_min_eig_matG_result <- min(eigen(matG_result, only.values = TRUE)$values)
    n_iter_idx <- n_iter_idx + 1
  }
  # check for convergence
  if (n_iter_idx > pn_max_iter){
    stop(" *** ERROR: No convergence of bending genomic relationship matrix")
  }
  return(matG_result)
}
mat_geno <- as.matrix(tbl_data_p4 %>% select(starts_with("S")) %>% select(-Sex))
mat_grm_inv <- solve(computeMatGrm(pmatData = mat_geno))
```

- Set up mixed model equations

```
mat_xtx <- crossprod(matX)
mat_xtz <- crossprod(matX, matZ)
mat_ztx <- t(mat_xtz)
mat_ztz_lgrm_inv <- crossprod(matZ) + lambda_g * mat_grm_inv
mat_coef <- rbind(cbind(mat_xtx,mat_xtz),
                  cbind(mat_ztx,mat_ztz_lgrm_inv))
mat_rhs <- rbind(crossprod(matX,tbl_data_p4$Weight),
                 crossprod(matZ,tbl_data_p4$Weight))
mat_sol <- solve(mat_coef,mat_rhs)
```

- Show the solutions for breeding values

```
cat(" * Genomic breeding values: \n")
```

```
##  * Genomic breeding values:
```

```
(mat_gbv_bvm <- mat_sol[(ncol(matX)+1):nrow(mat_sol),])
```

```
##  [1] -6.6472899 -2.1065419  0.7606017 -3.4687207 -1.1398511  1.7768713
##  [7] -0.4292761 -5.4061365 -5.0117499  4.7231338  7.2723361  2.9849377
## [13]  6.6916856
```

- Rank animals according to genomic breeding values

```
tbl_data_p4$Animal[order(mat_gbv_bvm, decreasing = T)]
```

```
##  [1] 11 13 10 12  6  3  7  5  2  4  9  8  1
```

Problem 5: Pedigree Based BLUP

Given is the following dataset. The column `Weight` is to be taken as response variable. The sex of an animal shown in column entitled with `Sex` is to be treated as fixed effect. The variable `Height` is to be modeled by a regression.

Im nachfolgenden Datensatz entspricht die Kolonne 'Weight' der Zielgrösse. Das Geschlecht der Tiere soll als fixer Effekt behandelt werden. Die Variable 'Height' soll als Regression modelliert werden.

Animal	Sire	Dam	Sex	Height	Weight
4	2	1	1	142	416
5	2	1	2	153	488
6	2	1	2	154	492
7	2	3	1	144	437
8	2	3	2	150	438
9	4	3	1	155	553
10	2	5	2	156	494
11	7	6	1	157	578
12	9	10	2	153	480
13	11	12	1	155	528

The dataset is available from

```
## https://charlotte-ngs.github.io/lbgfs2024/data/exam_data_p5.csv
```

a) Use a sire model to predict breeding values for all sires in the pedigree. Specify the model using a formula and explain all the variables used in the model. Write down expected values and variance-covariance matrices for all random effects. Construct mixed-model equations and obtain solutions for estimated fixed effects and predicted breeding values. The ratio λ_s between residual variance (σ_e^2) and sire variance (σ_s^2) is given as $\lambda_s = 24$. Rank the sires according to their breeding values.

Verwenden Sie ein Vatermodell für die Schätzung der Zuchtwerte aller Väter im Pedigree. Spezifizieren Sie das Modell mit einer Formel und erklären Sie die Bedeutung aller Variablen. Schreiben Sie die Erwartungswerte und die Varianz-Covarianz-Matrizen für alle zufälligen Effekte auf. Stellen Sie die Mischmodellgleichungen auf und lösen Sie diese, damit Sie Schätzungen für fixe Effekte und Zuchtwerte erhalten. Das Verhältnis λ_s zwischen Restvariance (σ_e^2) und Vätervarianz (σ_s^2) ist gegeben als $\lambda_s = 24$. Rangieren Sie die Väter gemäss ihren geschätzten Zuchtwerten.

23

Solution

Model

The sire model is given by

$$y = Xb + Zs + e$$

with vectors

- y: vector of length n with containing observations of responses
- b: vector of length k with intercept, regression coefficient and effects for sex

- s: vector of length q_s corresponding to the number of sires with random sire breeding values
- e: vector of length n with random residuals

and matrices

- X: design matrix of dimension $n \times k$ linking fixed effects to observations
- Z: design matrix of dimension $n \times q_s$ linking sire breeding values to observations.

Expected Values and Variances

$$E \begin{bmatrix} y \\ s \\ e \end{bmatrix} = \begin{bmatrix} Xb \\ 0 \\ 0 \end{bmatrix}$$

$$var \begin{bmatrix} y \\ s \\ e \end{bmatrix} = \begin{bmatrix} V & ZG_s & R \\ G_s Z^T & G_s & 0 \\ R & 0 & R \end{bmatrix}$$

with $var(e) = R = I * \sigma_e^2$, $var(s) = G_s = A_s * \sigma_s^2$ and $var(y) = V = ZG_sZ^T + R$. The residual variance component σ_e^2 and the sire variance component σ_s^2 are given via the ration λ_s and assumed to be known.

Mixed Model Equations

The mixed model equations are

$$\begin{bmatrix} X^TX & X^TZ \\ Z^TX & Z^TZ + A_s^{-1} * \lambda_s \end{bmatrix} \begin{bmatrix} \widehat{b} \\ \widehat{s} \end{bmatrix} = \begin{bmatrix} X^Ty \\ Z^Ty \end{bmatrix}$$

Define the componenents of MME

```
# matrix X
tbl_data_p5$Sex <- as.factor(tbl_data_p5$Sex)
mat_X <- model.matrix(Weight ~ Height + Sex, data = tbl_data_p5)
attr(mat_X, "assign") <- NULL
attr(mat_X, "contrasts") <- NULL
mat_X
```

```
##    (Intercept) Height Sex2
## 1            1    142    0
## 2            1    153    1
## 3            1    154    1
## 4            1    144    0
## 5            1    150    1
## 6            1    155    0
## 7            1    156    1
## 8            1    157    0
## 9            1    153    1
## 10           1    155    0
```

Matrix Z

```
mat_Z <- model.matrix(Weight ~ 0 + as.factor(Sire), data = tbl_data_p5)
attr(mat_Z, "assign") <- NULL
attr(mat_Z, "contrasts") <- NULL
dimnames(mat_Z) <- NULL
mat_Z
```

```
##       [,1] [,2] [,3] [,4] [,5]
##  [1,]    1    0    0    0    0
##  [2,]    1    0    0    0    0
##  [3,]    1    0    0    0    0
##  [4,]    1    0    0    0    0
##  [5,]    1    0    0    0    0
##  [6,]    0    1    0    0    0
##  [7,]    1    0    0    0    0
##  [8,]    0    0    1    0    0
##  [9,]    0    0    0    1    0
## [10,]    0    0    0    0    1
```

Augment pedigree

```
tbl_p5_ped <- tbl_data_p5[,c("Animal", "Sire", "Dam")]
tbl_p5_ped_aug <- augment_pedigree(ptbl_ped = tbl_p5_ped)
tbl_p5_ped_aug
```

```
## # A tibble: 13 x 3
##    Animal  Sire   Dam
##     <dbl> <dbl> <dbl>
##  1      1    NA    NA
##  2      2    NA    NA
##  3      3    NA    NA
##  4      4     2     1
##  5      5     2     1
##  6      6     2     1
##  7      7     2     3
##  8      8     2     3
##  9      9     4     3
## 10     10     2     5
## 11     11     7     6
## 12     12     9    10
## 13     13    11    12
```

Sire relationship matrix

```
vec_sire <- unique(tbl_data_p5$Sire)
pem_p5_sire <- pedigreemm::pedigree(sire = c(NA, 2, 2, 4, 7),
                                    dam  = rep(NA, ncol(mat_Z)),
                                    label = vec_sire)
mat_A_s_inv <- as.matrix(pedigreemm::getAInv(ped = pem_p5_sire))
mat_A_s_inv
```

```
##             2          4          7          9         11
```

```
## 2   1.6666667 -0.6666667 -0.6666667  0.0000000  0.0000000
## 4  -0.6666667  1.6666667  0.0000000 -0.6666667  0.0000000
## 7  -0.6666667  0.0000000  1.6666667  0.0000000 -0.6666667
## 9   0.0000000 -0.6666667  0.0000000  1.3333333  0.0000000
## 11  0.0000000  0.0000000 -0.6666667  0.0000000  1.3333333
```

Coefficient matrix

```
mat_xtx <- crossprod(mat_X)
mat_xtz <- crossprod(mat_X, mat_Z)
mat_ztx <- t(mat_xtz)
mat_ztz_l_A_s_inv <- crossprod(mat_Z) + n_lambda_s * mat_A_s_inv
mat_coef_sire <- rbind(cbind(mat_xtx, mat_xtz), cbind(mat_ztx, mat_ztz_l_A_s_inv))
mat_coef_sire
```

```
##             (Intercept) Height Sex2
## (Intercept)          10   1519    5   6   1   1   1   1
## Height             1519 230969  766 899 155 157 153 155
## Sex2                  5    766    5   4   0   0   1   0
## 2                     6    899    4  46 -16 -16   0   0
## 4                     1    155    0 -16  41   0 -16   0
## 7                     1    157    0 -16   0  41   0 -16
## 9                     1    153    1   0 -16   0  33   0
## 11                    1    155    0   0   0 -16   0  33
```

Right-hand side

```
vec_y <- tbl_data_p5$Weight
mat_rhs <- rbind(crossprod(mat_X, vec_y),
                 crossprod(mat_Z, vec_y))
mat_rhs
```

```
##               [,1]
## (Intercept)   4904
## Height      746937
## Sex2          2392
##               2765
##                553
##                578
##                480
##                528
```

Solutions

```
mat_sol_sire <- solve(mat_coef_sire, mat_rhs)
mat_sol_sire
```

```
##                      [,1]
## (Intercept) -1015.6778233
## Height         10.0800565
## Sex2          -50.3233238
```

```
##                 0.1064882
##                 0.2945718
##                 0.1122394
##                 0.2565351
##                -0.5131848
```

Sire breeding values

```
vec_sire_bv <- mat_sol_sire[(ncol(mat_X)+1):nrow(mat_sol_sire),]
```

Ranking according to the breeding values

```
vec_sire[order(vec_sire_bv, decreasing = T)]
```

```
## [1]  4  9  7  2 11
```

b) Use an animal model to predict breeding values for all animals in the pedigree. Specify the model using a formula and explain all the variables used in the model. Write down expected values and variance-covariance matrices for all random effects. Construct mixed-model equations and obtain solutions for estimated fixed effects and predicted breeding values. The heritability of the trait 'Weight' is assumed to be $h^2 = 0.16$. Rank all animals according to their breeding values.

Verwenden Sie ein Tiermodell für die Schätzung der Zuchtwerte aller Tiere im Pedigree. Spezifizieren Sie das Modell mit einer Formel und erklären Sie die Bedeutung aller Variablen. Schreiben Sie die Erwartungswerte und die Varianz-Covarianz-Matrizen für alle zufälligen Effekte auf. Stellen Sie die Mischmodellgleichungen auf und lösen Sie diese, damit Sie Schätzungen für fixe Effekte und Zuchtwerte erhalten. Die Erblichkeit des Merkmals 'Weight' beträgt $h^2 = 0.16$. Rangieren Sie alle Tiere im Pedigree gemäss den geschätzten Zuchtwerten.

23

Solution

Model

The sire model is given by

$$y = Xb + Zu + e$$

with vectors

- y: vector of length n with containing observations of responses
- b: vector of length k with intercept, regression coefficient and effects for sex
- u: vector of length q corresponding to the number of animals in the pedigree with random breeding values
- e: vector of length n with random residuals

and matrices

- X: design matrix of dimension $n \times k$ linking fixed effects to observations
- Z: design matrix of dimension $n \times q$ linking breeding values to observations.

Expected Values and Variances

$$E \begin{bmatrix} y \\ u \\ e \end{bmatrix} = \begin{bmatrix} Xb \\ 0 \\ 0 \end{bmatrix}$$

$$var \begin{bmatrix} y \\ u \\ e \end{bmatrix} = \begin{bmatrix} V & ZU & R \\ UZ^T & U & 0 \\ R & 0 & R \end{bmatrix}$$

with $var(e) = R = I * \sigma_e^2$, $var(u) = U = A * \sigma_u^2$ and $var(y) = V = ZGZ^T + R$

Mixed Model Equations

The mixed model equations are

$$\begin{bmatrix} X^TX & X^TZ \\ Z^TX & Z^TZ + A^{-1} * \lambda \end{bmatrix} \begin{bmatrix} \widehat{b} \\ \widehat{u} \end{bmatrix} = \begin{bmatrix} X^Ty \\ Z^Ty \end{bmatrix}$$

Define the componenents of MME

```r
# matrix X
tbl_data_p5$Sex <- as.factor(tbl_data_p5$Sex)
mat_X <- model.matrix(Weight ~ Height + Sex, data = tbl_data_p5)
attr(mat_X, "assign") <- NULL
attr(mat_X, "contrasts") <- NULL
mat_X
```

```
##    (Intercept) Height Sex2
## 1            1    142    0
## 2            1    153    1
## 3            1    154    1
## 4            1    144    0
## 5            1    150    1
## 6            1    155    0
## 7            1    156    1
## 8            1    157    0
## 9            1    153    1
## 10           1    155    0
```

Matrix Z

```r
tbl_p5_ped <- tbl_data_p5[,c("Animal", "Sire", "Dam")]
vec_founders <- get_founders(ptbl_ped = tbl_p5_ped)
n_nr_founders <- length(vec_founders)
mat_Z <- cbind(matrix(0, nrow = nrow(tbl_data_p5), ncol = n_nr_founders),
               diag(nrow = nrow(tbl_data_p5)))
```

Inverse numerator relationship matrix

```r
tbl_p5_ped_aug <- augment_pedigree(ptbl_ped = tbl_p5_ped)
pem_p5_am <- pedigreemm::pedigree(sire = as.integer(tbl_p5_ped_aug$Sire),
                                  dam = as.integer(tbl_p5_ped_aug$Dam),
                                  label = tbl_p5_ped_aug$Animal)
mat_A_inv <- as.matrix(pedigreemm::getAInv(ped = pem_p5_am))
```

Coefficient matrix

```r
mat_xtx <- crossprod(mat_X)
mat_xtz <- crossprod(mat_X, mat_Z)
mat_ztx <- t(mat_xtz)
mat_ztz_l_A_inv <- crossprod(mat_Z) + n_lambda * mat_A_inv
mat_coef <- rbind(cbind(mat_xtx, mat_xtz), cbind(mat_ztx, mat_ztz_l_A_inv))
```

Right-hand side

```r
vec_y <- tbl_data_p5$Weight
mat_rhs <- rbind(crossprod(mat_X, vec_y),
                 crossprod(mat_Z, vec_y))
```

Solutions

```
mat_sol <- solve(mat_coef, mat_rhs)
mat_sol
```

```
##                       [,1]
## (Intercept) -1016.9334440
## Height         10.0900497
## Sex2          -49.9299609
##                 1.0057372
##                -0.8234443
##                -0.1822929
##                 0.2411257
##                 0.4405386
##                 0.5975123
##                -0.2952374
##                -1.2107981
##                 0.3474218
##                -1.5203362
##                 0.3261108
##                -0.9679997
##                -1.7596616
```

Breeding values

```
vec_bv <- mat_sol[(ncol(mat_X)+1):nrow(mat_sol),]
```

Ranking

```
tbl_p5_ped_aug$Animal[order(vec_bv, decreasing = T)]
```

```
##  [1]  1  6  5  9 11  4  3  7  2 12  8 10 13
```

c) Compute reliabilities (B) of predictied breeding values for all animals in the pedigree. The residual variance under the animal model is 299. Rank all animals according to the computed reliabilities.

Berechnen Sie die Bestimmtheitsmasse (B) der geschätzten Zuchtwerte für alle Tiere im Pedigree. Die Restvarianz im Tiermodell ist 299. Rangieren Sie alle Tiere gemäss den berechneten Bestimmtheitsmassen.

4

Solution

Reliabilities are computed based on the inverse of the general form of the coefficient matrix

```
mat_gen_coef_inv <- solve(mat_coef / n_var_res_r)
mat_C22 <- mat_gen_coef_inv[(ncol(mat_X)+1):nrow(mat_gen_coef_inv),
                            (ncol(mat_X)+1):ncol(mat_gen_coef_inv)]
# compute inbreeding coefficient
vec_inb <- pedigreemm::inbreeding(ped = pem_p5_am)
# get reliability B
(vec_B <- 1 - diag(mat_C22) / ((1+vec_inb) * n_var_u_r))
```

```
##  [1] 0.04833601 0.01748213 0.06014200 0.05531701 0.06137300 0.06662141
##  [7] 0.05984273 0.06669195 0.06426483 0.06656385 0.05276533 0.07075975
## [13] 0.04923162
```

```
cat(" * Reliabilities in % are: \n")
```

```
##  * Reliabilities in % are:
```

```
round(100*vec_B, digits = 2)
```

```
##  [1] 4.83 1.75 6.01 5.53 6.14 6.66 5.98 6.67 6.43 6.66 5.28 7.08 4.92
```

Ranking animals according to reliabilities

```
tbl_p5_ped_aug$Animal[order(vec_B, decreasing = T)]
```

```
##  [1] 12  8  6 10  9  5  3  7  4 11 13  1  2
```