Applied Statistical Methods - Solution 11

AUTHOR
Peter von Rohr

PUBLISHED
May 20, 2024

WEBR STATUS
Ready!

Problem 1: Animal Model

Use the same dataset as in Exercise 10 for the sire model and predict breeding values for all animals in the dataset using an animals model. The dataset is available at

https://charlotte-ngs.github.io/asmasss2024/data/asm_ped_sim_data.csv

Hints

- The variance component σ_u^2 of the sire effect can be assumed to be 9.
- The variance component σ_e^2 of the random resiudals is 36.
- Sex is modelled as a fixed effect.
- The inverse sire relationship matrix can be computed using the function `getAInv()` from the `pedigreemm` package.

Solution

Specify the model

$$y = Xb + Zu + e$$

with vectors

- y of length n containing known phenotypic observations
- b of length p containing unknown fixed effects
- u of length q containing unknown random breeding values for all animals
- e of length n containing unknown random residuals

Known design matrices

- X of dimension $n \times p$ linking fixed effects to observations and
- Z of dimension $n \times q$ linking random breeding values to observations

The expected values and co-variance matrices of the random effects are

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

$$var\begin{bmatrix} y \\ u \\ e \end{bmatrix} = \begin{bmatrix} V & ZG & R \\ GZ^T & G & 0 \\ R & 0 & R \end{bmatrix}$$

with $R = I * \sigma_e^2$, $G = A * \sigma_u^2$ and $V = ZGZ^T + R$.

Read the data

Run Code

```
# read data to data.frame
s_ex11_p01 <- "https://charlotte-ngs.github.io/asmasss2024/data/asm_pe
```

```
df_lme <- read.table(s_ex11_p01, header = T, sep = ",")
df_lme
```

```
  ID SIRE DAM SEX    P
1  5    1   4   f 16.7
2  6    2   4   f 13.9
3  7    1   3   m 26.0
4  8    2   3   m  4.3
5  9    1   6   m 18.8
6 10    8   5   m  5.2
7 11    1   6   m  6.6
8 12    8   5   f 27.5
```

- Inverse Numerator Relationship Matrix

Run Code

```
# determine founder animals
vec_sire <- unique(df_lme$SIRE)
vec_fnd_sire <- setdiff(vec_sire, df_lme$ID)
vec_dam <- unique(df_lme$DAM)
vec_fnd_dam <- setdiff(vec_dam, df_lme$ID)
vec_fnd <- c(vec_fnd_sire, vec_fnd_dam)
vec_fnd <- vec_fnd[order(vec_fnd)]
n_nr_fnd <- length(vec_fnd)
# define pedigree
library(pedigreemm)
ped_am <- pedigree(sire = c(rep(NA, n_nr_fnd), df_lme$SIRE),
                   dam  = c(rep(NA, n_nr_fnd), df_lme$DAM),
                   label = as.character(c(vec_fnd, df_lme$ID)))
# inverse numerator relationship matrix
mat_A_inv <- as.matrix(getAInv(ped_am))
mat_A_inv
```

```
      1    2    3    4  5  6  7  8  9 10 11 12
1   3.0  0.0  0.5  0.5 -1  1 -1  0 -1  0 -1  0
2   0.0  2.0  0.5  0.5  0 -1  0 -1  0  0  0  0
3   0.5  0.5  2.0  0.0  0  0 -1 -1  0  0  0  0
4   0.5  0.5  0.0  2.0 -1 -1  0  0  0  0  0  0
5  -1.0  0.0  0.0 -1.0  3  0  0  1  0 -1  0 -1
6   1.0 -1.0  0.0 -1.0  0  3  0  0 -1  0 -1  0
7  -1.0  0.0 -1.0  0.0  0  0  2  0  0  0  0  0
8   0.0 -1.0 -1.0  0.0  1  0  0  3  0 -1  0 -1
9  -1.0  0.0  0.0  0.0  0 -1  0  0  2  0  0  0
10  0.0  0.0  0.0  0.0 -1  0  0 -1  0  2  0  0
11 -1.0  0.0  0.0  0.0  0 -1  0  0  0  0  2  0
12  0.0  0.0  0.0  0.0 -1  0  0 -1  0  0  0  2
```

Setup mixed model equations

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

Get the known components from the data into the mixed-model equations

- Design matrix X

Run Code

```r
# matrix X
mat_X <- model.matrix(P ~ SEX, data = df_lme)
attr(mat_X, "assign") <- NULL
attr(mat_X, "contrasts") <- NULL
dimnames(mat_X) <- NULL
mat_X
```

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

- Design matrix Z

Run Code

```r
# matrix Z
mat_Z <- model.matrix(P ~ 0 + as.factor(ID), data = df_lme)
attr(mat_Z, "assign") <- NULL
attr(mat_Z, "contrasts") <- NULL
dimnames(mat_Z) <- NULL
mat_Z <- cbind(matrix(0, nrow = nrow(df_lme), ncol = n_nr_fnd), mat_Z)
mat_Z
```

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

- Variance ration $\lambda_s = \sigma_e^2 / \sigma_s^2$

Run Code

```r
# variance components
sigma_u2 <- 0
```

```
sigma_u2 <- 9
sigma_e2 <- 36
# lambda
n_lambda <- sigma_e2 / sigma_u2
n_lambda
```

```
[1] 4
```

- Mixed model equations

Run Code

```
# coefficient matrix
mat_xtx <- crossprod(mat_X)
mat_xtz <- crossprod(mat_X, mat_Z)
mat_ztx <- t(mat_xtz)
mat_ztz_lam_a_inv <- crossprod(mat_Z) + n_lambda * mat_A_inv
mat_coef <- rbind(cbind(mat_xtx, mat_xtz),
                  cbind(mat_ztx, mat_ztz_lam_a_inv))
# right-hand side
mat_rhs <- rbind(crossprod(mat_X, df_lme$P),
                 crossprod(mat_Z, df_lme$P))
# solutions
mat_sol <- solve(mat_coef, mat_rhs)
mat_sol
```

```
            [,1]
   19.7175571343
   -7.5651720632
1   1.2950766779
2  -1.2250000000
3   0.6784481962
4  -0.7485248741
5  -0.0007843862
6  -1.4612270230
7   2.4157460473
8  -1.0238113159
9   0.6647792832
10 -1.2278630978
11 -0.6907762724
12  0.4093400063
```

Results

The first two numbers of the solutions correspond to estimates $\hat{b}$ which contains the intercept and the difference between group means of sex `f` and `m`. The remaining numbers in the solutions are the predicted breeding values of all animals in the dataset. At this point the numeric values of the predicted breeding values are not interesting. What we are interested is the ranking of the animals according to the breeding values. This is obtained by

Run Code

```
vec_ani_pbv <- mat_sol[3:nrow(mat_sol),1]
vec_ani_pbv[order(vec_ani_pbv, decreasing = T)]
```

```
            7             1             3             9            12 
 2.4157460473  1.2950766779  0.6784481962  0.6647792832  0.4093400063 
            5            11             4             8             2 
-0.0007843862 -0.6907762724 -0.7485248741 -1.0238113159 -1.2250000000 
           10             6 
-1.2278630978 -1.4612270230 
```