Applied Genetic Evaluation - Solution 3

Peter von Rohr

2019-05-13

Problem 1: Prediction Of Breeding Values Using A Sire Model

Use the data set from Exercise 2 available from

`https://charlotte-ngs.github.io/GELASMSS2019/ex/w12/data_sire_w12.csv`

to predict breeding values using a sire model. The sire model is a mixed linear effects model where the sire effects are random effects. This leads to the following model

$$y = Xb + Zs + e \tag{1}$$

where

- y vector of length n of observations
- b vector of length p of fixed effects
- s vector of length q of random sire effects
- e vector of length n of random errors

For the random effects, we have to specify the expected values and the variance-covariance matrices. Because the random errors e and the sire effects s are deviations from a common mean and hence their expected values are defined as

$$\begin{aligned} E[e] &= 0 \\ E[s] &= 0 \\ E[y] &= Xb \end{aligned}$$

The expected value $E[y]$ is computed using the defined expected values for e and s and from the model (1).

The random error terms e_i are uncorrelated and hence the variance-covariance matrix $var(e)$ is given by

$$var(e) = I * \sigma_e^2$$

In the case where the sires are unrelated, the sire effects are also uncorrelated and the variance-covariance matrix $var(s)$ corresponds to

$$var(s) = I * \sigma_s^2$$

The values for σ_e^2 and σ_s^2 are taken from the results of the variance components estimation from last weeks exercise. The variance-covariance matrix of the observations y can be computed as

$$var(y) = ZZ^T * \sigma_s^2 + I * \sigma_e^2$$

Hints

- Use the package `pedigreemm` to predict the breeding values for all the sires.
- The function `ranef()` can be used to extract realised values of random effects.

Solution

The solution is the same as in Exercise 2, but we will be interested in a different part of the solutions. We start by reading the data and with re-formatting the fixed effects as factors.

```
s_data_file <- "https://charlotte-ngs.github.io/GELASMSS2019/ex/w12/data_sire_w12.csv"
tbl_pb_sire <- readr::read_csv2(file = s_data_file)
```

```
## Using ',' as decimal and '.' as grouping mark. Use read_delim() for more control.
```

```
## Parsed with column specification:
## cols(
##   Id = col_double(),
##   slh = col_double(),
##   hrd = col_double(),
##   age = col_double(),
##   cw = col_double(),
##   sire = col_double()
## )
```

Reformat the fixed effects to factors

```
tbl_pb_sire$slh <- as.factor(tbl_pb_sire$slh)
tbl_pb_sire$hrd <- as.factor(tbl_pb_sire$hrd)
```

Setup the pedigree for this sire model

```
vec_sire <- unique(tbl_pb_sire$sire)
vec_sire <- vec_sire[order(vec_sire)]
n_nr_sire <- length(vec_sire)
ped_sire <- pedigreemm::pedigree(sire = rep(NA, n_nr_sire),
                                 dam = rep(NA, n_nr_sire),
                                 label = c(1:n_nr_sire))
```

Fitting the sire-model is done as follows. The summary function can be used to obtain the result for the variance components and for the fixed effects.

```
require(pedigreemm)
```

```
## Loading required package: pedigreemm
```

```
## Loading required package: lme4
```

```
## Loading required package: Matrix
```

```
lmem_sire <- pedigreemm(cw ~ slh + hrd + age + (1|sire),
                        pedigree = list(sire = ped_sire),
                        data = tbl_pb_sire)
summary(lmem_sire)
```

```
## Linear mixed model fit by REML ['lmerpedigreemm']
## Formula: cw ~ slh + hrd + age + (1 | sire)
##    Data: tbl_pb_sire
##
## REML criterion at convergence: 12610.8
##
## Scaled residuals:
##     Min      1Q  Median      3Q     Max
## -3.2011 -0.6731  0.0137  0.6539  3.4813
##
```

```
## Random effects:
##  Groups   Name        Variance Std.Dev.
##  sire     (Intercept)  6.257   2.501
##  Residual             90.181   9.496
## Number of obs: 1716, groups:  sire, 10
##
## Fixed effects:
##              Estimate Std. Error t value
## (Intercept) -77.17270   16.67579  -4.628
## slh2         22.36751    0.56484  39.600
## slh3          4.27798    0.56818   7.529
## hrd2         88.81545    0.73294 121.176
## hrd3          9.28428    0.72408  12.822
## hrd4         58.98147    0.71719  82.239
## hrd5         20.36389    0.72889  27.938
## age           0.68269    0.04161  16.405
##
## Correlation of Fixed Effects:
##      (Intr) slh2   slh3   hrd2   hrd3   hrd4   hrd5
## slh2 -0.011
## slh3 -0.072  0.513
## hrd2 -0.018  0.014 -0.023
## hrd3 -0.012 -0.001 -0.003  0.493
## hrd4  0.009 -0.008 -0.025  0.497  0.501
## hrd5  0.004  0.010 -0.028  0.490  0.495  0.500
## age  -0.998 -0.007  0.056 -0.004 -0.009 -0.031 -0.026
```

The realised values of the sire-effects correspond to the predicted breeding values for all sires. They can be obtained by

```
(pb_sire <- ranef(lmem_sire))
```

```
## $sire
##    (Intercept)
## 1    3.7549839
## 2   -1.1362988
## 3    1.3937496
## 4   -2.7762615
## 5   -2.1658114
## 6    0.4890680
## 7    0.9537987
## 8    3.5275015
## 9   -2.4808416
## 10  -1.5598885
```

Problem 2: Compare Offspring Of Sires

For the purpose of livestockbreeding the realised values themselves are not so interesting. But for the selection decision require a ranking of the sires according to their breeding values. Find the ranking of the sires according to their breeding values.

According to the definition of breeding value, it corresponds to the deviation of the offspring from the population mean. Hence the offspring of the sire with the best breeding value should on average be better than the offspring of the sire with the worst breeding value. Verify the difference between the average phenotypic values of the offsprings of the sires with the best and the worst predicted breeding values.

Solution

The ranking is obtained by

```
(vec_ranking_pb_sire <- order(pb_sire$sire$`(Intercept)`, decreasing = TRUE))
```

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

The comparison of the average phenotypic values of the offsprings of the sires with the best and the worst predicted breeding values can be verified by

```
require(dplyr)
tbl_pb_sire %>%
  group_by(sire) %>%
  summarise(avg = mean(cw))
```

```
## # A tibble: 10 x 2
##     sire   avg
##    <dbl> <dbl>
##  1     1  241.
##  2     2  238.
##  3     3  241.
##  4     4  237.
##  5     5  240.
##  6     6  242.
##  7     7  240.
##  8     8  244.
##  9     9  244.
## 10    10  238.
```

Comparing the average of the phenotypic values of the sire with the best and the worst predicted breeding value leads to

```
tbl_pb_sire %>%
  group_by(sire) %>%
  summarise(avg = mean(cw)) %>%
  filter(sire == vec_ranking_pb_sire[1] |
           sire == vec_ranking_pb_sire[length(vec_ranking_pb_sire)])
```

```
## # A tibble: 2 x 2
##    sire   avg
##   <dbl> <dbl>
## 1     1  241.
## 2     4  237.
```

Additional Problem: Predict Breeding Values Using Animal Model

As in Exercise 2, we are using the full dataset to predicted breeding values with an animal model. The computations for the solution of this Problem will have a very long runtime. That is why the solution is only sketched and not explicitly computed.

The data is available from `https://charlotte-ngs.github.io/GELASMSS2019/ex/w12/data_bp_w12.csv`.

Solution

We first have to read the data

```
s_data_path_gel_ex2 <- "https://charlotte-ngs.github.io/GELASMSS2019/ex/w12/data_bp_w12.csv"
tbl_gel_ex2 <- readr::read_csv2(file = s_data_path_gel_ex2)
```

```
## Using ',' as decimal and '.' as grouping mark. Use read_delim() for more control.
```

```
## Parsed with column specification:
## cols(
##   Id = col_double(),
##   sex = col_double(),
##   slh = col_double(),
##   hrd = col_double(),
##   age = col_double(),
##   cw = col_double(),
##   sire = col_double(),
##   dam = col_double()
## )
```

```
colnames(tbl_gel_ex2);dim(tbl_gel_ex2)
```

```
## [1] "Id"   "sex"  "slh"  "hrd"  "age"  "cw"   "sire" "dam"
```

```
## [1] 5325    8
```

The fixed effects are converted into factors

```
tbl_gel_ex2$sex <- as.factor(tbl_gel_ex2$sex)
tbl_gel_ex2$slh <- as.factor(tbl_gel_ex2$slh)
tbl_gel_ex2$hrd <- as.factor(tbl_gel_ex2$hrd)
```

From the help file of `pedigreemm`, we can see that we first have to define a pedigree.

```
ped <- pedigreemm::pedigree(sire  = tbl_gel_ex2$sire,
                            dam   = tbl_gel_ex2$dam,
                            label = tbl_gel_ex2$Id)
```

Now the model can be specified as for the other functions to fit linear mixed effects model, such as `lmer`.

```
# This takes more than one hour to run.
require(pedigreemm)
# according to https://stat.ethz.ch/pipermail/r-sig-mixed-models/2014q1/021609.html
options(lmerControl=list(check.nobs.vs.nlev="ignore",
     check.nobs.vs.rankZ = "ignore",
     check.nobs.vs.nRE="ignore"))
s_lmem_file <- "lmem_gel_ex2.rds"
if (file.exists(s_lmem_file)){
  load(file = s_lmem_file)
} else {
  lmem_gel_ex2 <- pedigreemm(cw ~ sex + slh + hrd + age + (1|Id),
                             data = tbl_gel_ex2,
                             pedigree = list(Id = ped))
  saveRDS(lmem_gel_ex2, file = s_lmem_file)
}
summary(lmem_gel_ex2)
```

The predicted breeding values are obtained by

```
ranef(lmem_gel_ex2)
```