Livestock Breeding and Genomics - Solution 2

Peter von Rohr

2020-10-02

Problem 1: Breeding Values For a Monogenic Trait

We assume that the absorption of cholesterol is determined by a certain enzyme. The level of enzyme production is determined by a single bi-allelic locus E. The genotype frequencies and the genotypic values for the two dairy cattle populations `Original Braunvieh` and `Brown Swiss` are given in the following table.

Variable	Original Braunvieh	Brown Swiss
$f(E_1E_1)$	0.0625	0.01
$f(E_1E_2)$	0.3750	0.18
$f(E_2E_2)$	0.5625	0.90
a	15.0000	29.00
d	3.0000	0.00

Hints

- Assume that allele E_1 is the allele with the positive effect on the enzyme level
- Assume that the Hardy-Weinberg Equilibrium holds in both populations

Your Task

Compute the breeding values for all three genotypes in both populations.

Solution

The breeding values are computed as shown in the following table.

Genotype	Breeding Value
E_1E_1	$BV_{11} = 2q\alpha$
E_1E_2	$BV_{12} = (q-p)\alpha$
E_2E_2	$BV_{22} = -2p\alpha$

with $\alpha = a + (q-p)d$. The values for a and d are given in the task and the allele frequencies p and q can be computed from the given genotype frequencies.

$$p = f(E_1) = f(E_1E1) + \frac{1}{2}f(E_1E_2)$$

and $q = 1 - p$

For the two populations we get

Variable	Original Braunvieh	Brown Swiss
p	0.25	0.1
q	0.75	0.9
α	16.50	29.0

The breeding values for the two breeds are given in the following table

Genotype	Breeding Value	Original Braunvieh	Brown Swiss
E_1E_1	BV_{11}	24.75	52.2
E_1E_2	BV_{12}	8.25	23.2
E_2E_2	BV_{22}	-8.25	-5.8

Problem 2: Matrices in R

In R, matrices are constructed using the function `matrix()`. This function accepts different options. We want to see, how these options work.

Your Task: Construct matrices using the different options to better understand the meaning of the different options.

Parameter `data`

- `data`: Specify the different matrix elements

```
(matA <- matrix(data = c(1:9), nrow = 3, ncol = 3))
```

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

- `data`: without specifying the matrix elements

```
(matB <- matrix(nrow = 3, ncol = 3))
```

```
##      [,1] [,2] [,3]
## [1,]   NA   NA   NA
## [2,]   NA   NA   NA
## [3,]   NA   NA   NA
```

- `data`: specifying not all matrix elements

```
(matC <- matrix(data = c(1,2,3), nrow = 3, ncol = 3))
```

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

```
(matC2 <- matrix(data = c(1,2,3,4), nrow = 3, ncol = 3))
```

```
## Warning in matrix(data = c(1, 2, 3, 4), nrow = 3, ncol = 3): data length [4] is
```

```
## not a sub-multiple or multiple of the number of rows [3]
```

```
##      [,1] [,2] [,3]
## [1,]    1    4    3
## [2,]    2    1    4
## [3,]    3    2    1
```

Parameters `nrow` and `ncol`

- Leaving out one of both parameters

```
(matD <- matrix(data = c(1:9), nrow = 3))
```

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

```
(matE <- matrix(data = c(1:9), ncol = 3))
```

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

Parameter `byrow`

```
(matF <- matrix(data = c(1:9), nrow = 3, ncol = 3, byrow = TRUE))
```

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

```
(matG <- matrix(data = c(1:9), nrow = 3, ncol = 3, byrow = FALSE))
```

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

Problem 3: Matrix multiplication in R

In R, matrices can be multiplied using the operator `%*%` or with the functions `crossprod()` or `tcrossprod()`. With `crossprod()` and `tcrossprod()` vectors and matrices can be multiplied directly. The conversion of vectors to matrices is done automatically inside of these functions. The result will always be a matrix. When doing matrix-vector multiplications with `%*%` the vector has to be converted first into a matrix using the function `as.matrix()`.

In a first part of this problem, compare the results of the functions `crossprod()`, `tcrossprod()` and `%*%`.

a) Given are the following matrices

```
matA <- matrix(data = c(1:9), ncol = 3)
matB <- matrix(data = c(2:10), ncol = 3)
```

Find out which matrix multiplication with `%*%` corresponds to the following statement?

```
crossprod(matA,matB)
```

```
##      [,1] [,2] [,3]
## [1,]   20   38   56
## [2,]   47   92  137
## [3,]   74  146  218
```

Solution

The statement `crossprod(matA,matB)` corresponds to

```
t(matA) %*% matB
```

```
##      [,1] [,2] [,3]
## [1,]   20   38   56
## [2,]   47   92  137
## [3,]   74  146  218
```

Alternatively there is the function `tcrossprod()`. Find out which matrix multiplication is executed by

```
tcrossprod(matA, matB)
```

```
##      [,1] [,2] [,3]
## [1,]   78   90  102
## [2,]   93  108  123
## [3,]  108  126  144
```

Solution

```
matA %*% t(matB)
```

```
##      [,1] [,2] [,3]
## [1,]   78   90  102
## [2,]   93  108  123
## [3,]  108  126  144
```

b) Given is the vector `vecB`

```
vecB <- c(-3,16,1)
```

Multiply the matrix `matA` with the vector `vecB` once using `%*%` and then with the function `crossprod()`.

Hint: a vector can be converted to a matrix using the function `as.matrix()`.

Solution

```
matA %*% as.matrix(vecB)
```

```
##      [,1]
## [1,]   68
## [2,]   82
## [3,]   96
```

```
crossprod(t(matA), vecB)
```

```
##      [,1]
## [1,]   68
## [2,]   82
## [3,]   96
```

Problem 4: Quantitative Genetics

In a population the following numbers of genotypes were counted for a given genetic locus called A.

```
dfGenotypeFreq <- data.frame(Genotypes = c("$A_1A_1$", "$A_1A_2$", "$A_2A_2$"),
                             Numbers   = c(24, 53, 23),
                             stringsAsFactors = FALSE)
knitr::kable(dfGenotypeFreq)
```

Genotypes	Numbers
A_1A_1	24
A_1A_2	53
A_2A_2	23

a) Compute the genotype frequencies

Solution

```
nTotNrInd <- sum(dfGenotypeFreq$Numbers)
vGenoTypeFreq <- dfGenotypeFreq$Numbers / nTotNrInd
cat(paste("genotype-frequency", dfGenotypeFreq$Genotypes[1]), ": ", vGenoTypeFreq[1])
```

```
## genotype-frequency $A_1A_1$ :  0.24
```

```
cat(paste("genotype-frequency", dfGenotypeFreq$Genotypes[2]), ": ", vGenoTypeFreq[2])
```

```
## genotype-frequency $A_1A_2$ :  0.53
```

```
cat(paste("genotype-frequency", dfGenotypeFreq$Genotypes[3]), ": ", vGenoTypeFreq[3])
```

```
## genotype-frequency $A_2A_2$ :  0.23
```

b) Compute the allele frequencies

Solution

```
vGenFreqP <- vGenoTypeFreq[1] + 0.5*vGenoTypeFreq[2]
vGenFreqQ <-  vGenoTypeFreq[3] + 0.5*vGenoTypeFreq[2]
cat("allele frequency for A1: ", vGenFreqP)
```

```
## allele frequency for A1:  0.505
```

```
cat("allele frequency for A2: ", vGenFreqQ)
```

```
## allele frequency for A2:  0.495
```

c) Compute the population mean μ under the following assumptions

- the difference between the genotypic values of the homozygous genotypes is 20 and
- the genotypic value of the heterozygous genotype is 2.

Solution

```r
nDeltaHom <- 20
### # additive value A
nAddValue <- nDeltaHom / 2
nDom <- 2
### # population mean
nMu <- (vGenFreqP-vGenFreqQ) * nAddValue + 2 * vGenFreqP * vGenFreqQ * nDom
cat("Population mean: ", nMu, "\n")
```

```
## Population mean:  1.0999
```