Applied Statistical Methods - Solution 5

Peter von Rohr

2022-03-26

Problem 1: Helmert Contrasts

Use the dataset of `Body Weight` and `Breed` to fit a linear model of `Body Weight` on `Breed`. The aim of this exercise is to use the `Helmert`-contrasts instead of the defautl `Treatment` contrasts. What are the estimable functions used in the `Helmert`-Contrasts and what are the effects that are reported for the different levels of the factor `Breed`? Verify your answer by comparing estimable functions of solutions of the least squares normal equations to the effects of `lm()`.

The dataset is available under

```
## https://charlotte-ngs.github.io/asmss2022/data/asm_bw_flem.csv
```

Hint

- Use `options(contrasts = c("contr.helmert", "contr.helmert"))` to change the default contrasts to the desired `Helmert`-Contrasts

Solution

- Compute solutions to least squares normal equation. First the data is read from the given file

```
tbl_e05p01 <- readr::read_csv(file = s_ex05p01_data_path)
```

```
## Rows: 10 Columns: 6
## -- Column specification ------------------------------------------------------------------------
## Delimiter: ","
## chr (1): Breed
## dbl (5): Animal, Breast Circumference, Body Weight, BCS, HEI
##
## i Use `spec()` to retrieve the full column specification for this data.
## i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

The data is sorted according to the breed

```
tbl_e05p01 <- tbl_e05p01[order(tbl_e05p01$Breed),]
tbl_e05p01
```

```
## # A tibble: 10 x 6
##    Animal `Breast Circumference` `Body Weight`   BCS   HEI Breed
##     <dbl>                  <dbl>         <dbl> <dbl> <dbl> <chr>
##  1      1                    176           471   5     161 Angus
##  2      2                    177           463   4.2   121 Angus
##  3      4                    179           470   3     165 Angus
##  4      7                    181           518   4.4   163 Limousin
##  5      8                    182           511   4.4   149 Limousin
```

```
## 6      9                         183           510   3.5   143 Limousin
## 7     10                         184           541   4.7   130 Limousin
## 8      3                         178           481   4.9   157 Simmental
## 9      5                         179           496   6.8   136 Simmental
## 10     6                         180           491   4.9   123 Simmental
```

A solution vector depends on the matrix X and on the vector y. The vector y is directly obtained from the column `Body Weight` of the dataframe.

```
vec_y <- tbl_e05p01$`Body Weight`
```

The matrix X can be obtained from the function `model.matrix()`.

```
mat_X <- model.matrix(lm(`Body Weight` ~ 0 + Breed, data = tbl_e05p01))
mat_X <- cbind(matrix(1, nrow = nrow(mat_X), ncol = 1), mat_X)
mat_X
```

```
##      BreedAngus BreedLimousin BreedSimmental
## 1  1          1             0              0
## 2  1          1             0              0
## 3  1          1             0              0
## 4  1          0             1              0
## 5  1          0             1              0
## 6  1          0             1              0
## 7  1          0             1              0
## 8  1          0             0              1
## 9  1          0             0              1
## 10 1          0             0              1
```

A solution for the least squares normal equations is obtained by

```
mat_xtx_ginv <- MASS::ginv(crossprod(mat_X))
mat_xty <- crossprod(mat_X, vec_y)
mat_b0 <- crossprod(mat_xtx_ginv, mat_xty)
mat_b0
```

```
##           [,1]
## [1,] 369.33333
## [2,]  98.66667
## [3,] 150.66667
## [4,] 120.00000
```

The solutions correspond to the vector b^0 with the components

$$b^0 = \begin{bmatrix} \mu \\ \alpha_1 \\ \alpha_2 \\ \alpha_3 \end{bmatrix} = \begin{bmatrix} 369.333 \\ 98.667 \\ 150.667 \\ 120 \end{bmatrix}$$

- Change contrasts from default to Helmert. Start by saving away the existing options

```
opts <- options()
```

Change contrasts

```
options(contrasts = c("contr.helmert", "contr.helmert"))
getOption("contrasts")
```

```
## [1] "contr.helmert" "contr.helmert"
```

- Find estimable functions associated to Helmert contrasts. Estimable functions are derived from the contrasts matrix. First the Breed column must be converted to a factor.

```
tbl_e05p01$Breed <- as.factor(tbl_e05p01$Breed)
c_mat_helmert <- contrasts(tbl_e05p01$Breed)
c_mat_helmert
```

```
##            [,1] [,2]
## Angus        -1   -1
## Limousin      1   -1
## Simmental     0    2
```

Add a columns of all ones to `c_mat_helmert`.

```
c_mat_helmert <- cbind(matrix(1, nrow = nrow(c_mat_helmert), ncol = 1), c_mat_helmert)
c_mat_helmert
```

```
##            [,1] [,2] [,3]
## Angus         1   -1   -1
## Limousin      1    1   -1
## Simmental     1    0    2
```

Compute the inverse of `c_mat_helmert`

```
est_mat_helmert <- solve(c_mat_helmert)
est_mat_helmert
```

```
##             Angus   Limousin Simmental
## [1,]  0.3333333  0.3333333 0.3333333
## [2,] -0.5000000  0.5000000 0.0000000
## [3,] -0.1666667 -0.1666667 0.3333333
```

The first row tells us how the intercept is computed. The intercept ($\hat{b}_0$) here corresponds to

$$\hat{b}_0 = \frac{1}{3}\left(E(y_{1.}) + E(y_{2.}) + E(y_{3.})\right)$$

where $E(y_{1.})$, $E(y_{2.})$ and $E(y_{3.})$ are the mean values of `Body Weight` for Angus, Limousin and Simmental animals, respectively.

```
n_mean_angus <- mean(tbl_e05p01[tbl_e05p01$Breed == "Angus", ]$`Body Weight`)
n_mean_limousin <- mean(tbl_e05p01[tbl_e05p01$Breed == "Limousin", ]$`Body Weight`)
n_mean_simmental <- mean(tbl_e05p01[tbl_e05p01$Breed == "Simmental", ]$`Body Weight`)
mean(c(n_mean_angus, n_mean_limousin,n_mean_simmental))
```

```
## [1] 492.4444
```

The second row of `est_mat_helmert` shows the first estimable function that is used. It corresponds to

$$\hat{b}_1 = \frac{1}{2}(\alpha_2 - \alpha_1)$$

where $\hat{b}_1$ measures the difference between the breeds `Limousin` and `Angus` corresponding to

```
1/2*(mat_b0[3] - mat_b0[2])
```

```
## [1] 26
```

The third row of `est_mat_helmert` shows how the `Body Weight` of the breed `Simmental` is compared to the two other breeds. It is

$$\hat{b}_2 = \frac{1}{6}(2\alpha_3 - \alpha_2 - \alpha_1)$$

which measures the difference between `Simmental` and `Limousin` and `Angus` together.

```
1/6 * (2*mat_b0[4] - mat_b0[3] - mat_b0[2])
```

```
## [1] -1.555556
```

- Check back with effects of `lm()`. The estimate for the intercept is

```
lm_helmert <- lm(`Body Weight` ~ Breed, data = tbl_e05p01)
coefficients(lm_helmert)["(Intercept)"]
```

```
## (Intercept)
##    492.4444
```

The estimates for the `Breed` effects can be seen from the list of all coefficients.

```
coefficients(lm_helmert)
```

```
## (Intercept)      Breed1      Breed2
##  492.444444   26.000000   -1.555556
```

- Restore original options

```
options(opts)
```

Problem 2: Simulation

Use the results of the regression of `Body Weight` on `Breast Circumference` and simulate three datasets with 10, 30 and 100 observations respectively. What is the number of observations required to obtain the same regression results from the simulated dataset that you used in the simulation?

The original dataset is available under:

```
## https://charlotte-ngs.github.io/asmss2022/data/asm_bw_flem.csv
```

Solution

- Run the regression analysis of `Body Weight` on `Breast Circumference`

```
tbl_bwbc <- readr::read_csv(file = s_ex05p02_data_path)
```

```
## Rows: 10 Columns: 6
```

```
## -- Column specification --------------------------------------------------------------------------
## Delimiter: ","
## chr (1): Breed
## dbl (5): Animal, Breast Circumference, Body Weight, BCS, HEI
```

```
##
## i Use `spec()` to retrieve the full column specification for this data.
## i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

```
lm_bwbc <- lm(`Body Weight` ~ `Breast Circumference`, data = tbl_bwbc)
b0 <- coefficients(lm_bwbc)["(Intercept)"]
b1 <- coefficients(lm_bwbc)["`Breast Circumference`"]
mean_bc <- mean(tbl_bwbc$`Breast Circumference`)
sd_bc <- sd(tbl_bwbc$`Breast Circumference`)
```

```r
sry_bwbc <- summary(lm_bwbc)
sd_res <- sry_bwbc$sigma
```

- Create the three datasets. In a first step, we create a function that takes as arguments
  - number of observations
  - intercept b_0
  - slope b_1
  - mean `Breast Circumference`
  - standard deviation `Breast Circumference`
  - standard deviation of residuals

and returns a dataset according to these input values.

```r
simulate_bwbc <- function(pn_nrobs, pn_b0, pn_b1, pn_mean_x, pn_sd_x, pn_sd_res){
  vec_bc <- rnorm(pn_nrobs, mean = pn_mean_x, sd = pn_sd_x)
  vec_bw <- pn_b0 + pn_b1 * vec_bc + rnorm(pn_nrobs, mean = 0, sd = pn_sd_res)
  tbl_result <- tibble::tibble(Animal = c(1:pn_nrobs),
                               BC = vec_bc,
                               BW = vec_bw)
  return(tbl_result)
}
```

With the above defined function, we can create a list with the three datasets

```r
set.seed(1928)
vec_nobs <- c(10,30,100)
l_data_set <- lapply(vec_nobs, simulate_bwbc,
                     pn_b0 = b0,
                     pn_b1 = b1,
                     pn_mean_x = mean_bc,
                     pn_sd_x = sd_bc,
                     pn_sd_res = sd_res)
```

Each of the datasets is analysed by `lm()` and the results are again stored in a list

```r
l_lm_result <- lapply(l_data_set, function(x) lm(BW ~ BC, data = x))
```

Collect the resutls into a table

```r
tbl_result <- NULL
for (cur_res in l_lm_result){
  sry_cur_res <- summary(cur_res)
  tbl_cur <- tibble::tibble(NrObs = length(cur_res$residuals),
             Intercept_Estimate = coefficients(sry_cur_res)["(Intercept)", "Estimate"],
             Intercept_StdErr = coefficients(sry_cur_res)["(Intercept)", "Std. Error"],
             Slope_Estimate = coefficients(sry_cur_res)["BC", "Estimate"],
             Slope_StdErr = coefficients(sry_cur_res)["BC", "Std. Error"],
             ResStdErr = sry_cur_res$sigma)
  if (is.null(tbl_result)){
    tbl_result <- tbl_cur
  } else {
    tbl_result <- dplyr::bind_rows(tbl_result, tbl_cur)
  }
}
knitr::kable(tbl_result)
```

NrObs	Intercept_Estimate	Intercept_StdErr	Slope_Estimate	Slope_StdErr	ResStdErr
10	-1321.766	223.4298	10.084512	1.2400348	9.310627
30	-1089.632	124.7779	8.801965	0.6917396	9.520995
100	-1069.614	83.3047	8.682105	0.4635349	11.905503

The true values used in the sumulation are

Intercept	Slope	ResStdErr
-1065.115	8.673235	11.0815