Exercises

Arunsoumya Basu

Roll No. bs2029

1. We have been discussing about a problem related to the elongation of springs in two different labs. We used the intercept coefficients α_1 and α_2 for lab 1 and lab 2 respectively, and β for the coefficient of weight. We formulated the problem as a linear model and solved for these three parameters. Now, if α_1 and α_2, the unstretched lengths of the springs, are directly measured (i.e., they are given constants), how would one formulate the problem as a linear model and solve for β?

Solution : Let us consider the general case, where we have observations $l_{11}, l_{12}, \cdots, l_{1i_1}$ from the first lab, corresponding to the weights $w_{11}, w_{12}, \cdots, w_{1i_1}$ and observations $l_{21}, l_{22}, \cdots, l_{2i_2}$ from the second lab, corresponding to the weights $w_{21}, w_{22}, \cdots, w_{2i_2}$. The system of equations is

$$l_{11} = \alpha_1 + \beta w_{11} + \epsilon_{11}$$

$$\vdots$$

$$l_{1i_1} = \alpha_1 + \beta w_{1i_1} + \epsilon_{1i_1}$$

$$l_{21} = \alpha_2 + \beta w_{21} + \epsilon_{21}$$

$$\vdots$$

$$l_{2i_2} = \alpha_2 + \beta w_{2i_2} + \epsilon_{2i_2}$$

where α_1 and α_2 are given constants, and ϵ_{ij}s denote the random errors, as usual. After transferring α_1 and α_2 to the left side and writing this in matrix notation, we obtain

$$\underbrace{\begin{bmatrix} l_{11}-\alpha_1 \\ \vdots \\ l_{1i_1}-\alpha_1 \\ l_{21}-\alpha_2 \\ \vdots \\ l_{2i_2}-\alpha_2 \end{bmatrix}}_{\vec{y}} = \underbrace{\begin{bmatrix} w_{11} \\ \vdots \\ w_{1i_1} \\ w_{21} \\ \vdots \\ w_{2i_2} \end{bmatrix}}_{\mathbf{X}} \underbrace{\beta}_{\vec{\beta}} + \underbrace{\begin{bmatrix} \epsilon_{11} \\ \vdots \\ \epsilon_{1i_1} \\ \epsilon_{21} \\ \vdots \\ \epsilon_{2i_2} \end{bmatrix}}_{\vec{\epsilon}}$$

Now the standard procedure of obtaining $\widehat{\beta} = (\mathbf{X'X})^{-\mathbf{1}}\mathbf{X'}\ \vec{y}$ through the `lm` function in R can be performed. For an illustration, we consider the dataset of the spring problem, given by

Weight	Length	Lab
1.0	5.29	1
1.5	6.31	1
2.0	7.28	1
2.5	8.33	1
3.0	9.30	1
3.5	10.32	1
1.2	7.60	2
1.5	8.11	2
1.8	8.88	2
2.1	9.40	2
2.1	9.39	2

While solving the original problem (unknown $\alpha_1, \alpha_2, \beta$), we obtained the least squares estimates $\widehat{\alpha_1} = 3.276$, $\widehat{\alpha_2} = 5.173$ and $\widehat{\beta} = 2.013$. So it is a good idea to try with the following inputs: $\alpha_1 = 3.276$ and $\alpha_2 = 5.173$. Intuitively, it seems that we would obtain $\widehat{\beta} = 2.013$ as the output.

The series of commands

```
f=function(a,b)
{
   lab1=matrix(c(1,1.5,2,2.5,3,3.5,5.29,6.31,7.28,8.33,9.30,10.32),6,2)
   colnames(lab1)=c("weight","length")
   lab1[,2]=lab1[,2]-a
   lab2=matrix(c(1.2,1.5,1.8,2.1,2.1,7.60,8.11,8.88,9.40,9.39),5,2)
   lab2[,2]=lab2[,2]-b
   colnames(lab2)=c("weight","length")
   lab1m=data.frame(lab1,lab=1)
   lab2m=data.frame(lab2,lab=2)
   alllab=rbind(lab1m,lab2m)
   alllab$lab=factor(alllab$lab)
   fit=lm(length ~ weight-1,alllab)
   print(fit)
   cat("\nThe design matrix is:\n\n")
   print(model.matrix(fit))
}
f(3.276,5.173)
```

generate the output

```
Call:
lm(formula = length ~ weight - 1, data = alllab)

Coefficients:
weight
 2.013


The design matrix is:

   weight
1     1.0
2     1.5
3     2.0
4     2.5
5     3.0
6     3.5
7     1.2
8     1.5
9     1.8
10    2.1
11    2.1
attr(,"assign")
[1] 1
```

And yes, we obtained what we expected!

2. We consider a variant of the above problem, where α_1 has been measured correctly, but α_2 is unknown. (This is natural to expect in real-life problems, where a part of the data may be missing.) In that case, how would one formulate the problem as a linear model and solve for α_2 and β?

Solution : Here, too we have the same system of equations

$$l_{11} = \alpha_1 + \beta w_{11} + \epsilon_{11}$$

$$\vdots$$

$$l_{1i_1} = \alpha_1 + \beta w_{1i_1} + \epsilon_{1i_1}$$

$$l_{21} = \alpha_2 + \beta w_{21} + \epsilon_{21}$$

$$\vdots$$

$$l_{2i_2} = \alpha_2 + \beta w_{2i_2} + \epsilon_{2i_2}$$

with the usual notations; the only difference is that α_1 is a given constant, and α_2, β have to be estimated. In the matrix notation, this becomes

$$\underbrace{\begin{bmatrix} l_{11}-\alpha_1 \\ \vdots \\ l_{1i_1}-\alpha_1 \\ l_{21} \\ \vdots \\ l_{2i_2} \end{bmatrix}}_{\vec{y}} = \underbrace{\begin{bmatrix} 0 & w_{11} \\ \vdots & \\ 0 & w_{1i_1} \\ 1 & w_{21} \\ \vdots & \\ 1 & w_{2i_2} \end{bmatrix}}_{\mathbf{X}} \underbrace{\begin{bmatrix} \alpha_2 \\ \beta \end{bmatrix}}_{\vec{\beta}} + \underbrace{\begin{bmatrix} \epsilon_{11} \\ \vdots \\ \epsilon_{1i_1} \\ \epsilon_{21} \\ \vdots \\ \epsilon_{2i_2} \end{bmatrix}}_{\vec{\epsilon}}$$

after transferring α_1 to the left side. Similar to the previous problem, the estimated values $\begin{bmatrix} \widehat{\alpha_2} \\ \widehat{\beta} \end{bmatrix}$ are given by $(\mathbf{X}'\mathbf{X})^{-1}\mathbf{X}'\,\vec{y}$. To demonstrate, we consider the same dataset used in the previous problem. We could have chosen any reasonable value of α_1, but let us again choose its least square estimate, which was obtained earlier. So we start with the input $\alpha_1 = 3.276$. Once again, we expect that $\widehat{\alpha_2}$ and $\widehat{\beta}$ should be same as their previously obtained least square estimates, i.e., 5.173 and 2.013 respectively. The series of commands

```
g=function(k)
{
   lab1=matrix(c(1,1.5,2,2.5,3,3.5,5.29,6.31,7.28,8.33,9.30,10.32),6,2)
   colnames(lab1)=c("weight","length")
   lab1[,2]=lab1[,2]-k
   lab2=matrix(c(1.2,1.5,1.8,2.1,2.1,7.60,8.11,8.88,9.40,9.39),5,2)
   colnames(lab2)=c("weight","length")
   lab1m=data.frame(lab1,lab=1)
   lab2m=data.frame(lab2,lab=2)
   alllab=rbind(lab1m,lab2m)
```

```
   alllab$lab=factor(alllab$lab)

   mat=matrix(0,11,2)

   mat[,1]=c(rep(0,nrow(lab1)),rep(1,nrow(lab2)))

   mat[,2]=alllab[,1]

   colnames(mat)=c("lab2","weight")

   fit=lm(alllab$length ~ mat-1)

   print(fit)

   cat("\nThe design matrix is:\n\n")

   print(model.matrix(fit))

}

g(3.276)
```

generate the output

```
Call:
lm(formula = alllab$length ~ mat - 1, data = alllab)

Coefficients:
  matlab2  matweight
    5.174      2.013


The design matrix is:

   matlab2 matweight
1        0       1.0
2        0       1.5
3        0       2.0
4        0       2.5
5        0       3.0
6        0       3.5
7        1       1.2
8        1       1.5
9        1       1.8
10       1       2.1
11       1       2.1
attr(,"assign")
[1] 1 1
```

We note that upto rounding error (the value of α_2 is shown to be 5.174, not 5.173), the estimates match our expectations.