CHAPTER 8

EXERCISES FOR LINEAR REGRESSION: LEAST SQUARES ESTIMATION

In this homework assignment you will write an R function that estimates the regression parameter $\boldsymbol{\beta}$ in the model

$$\mathbf{y} = \mathbf{X}\boldsymbol{\beta} + \boldsymbol{\epsilon}, \tag{8.1}$$

where $\mathbf{y}$ is an N-dimensional vector of predictands, and $\mathbf{X}$ is an $N \times M$ dimensional matrix of predictors. The core of this function is to compute an estimate of $\boldsymbol{\beta}$ based on solution of the normal equations

$$\hat{\boldsymbol{\beta}} = \left(\mathbf{X}^T\mathbf{X}\right)^{-1}\mathbf{X}^T\mathbf{y}. \tag{8.2}$$

In practice, most packages use the SVD or QR algorithm to estimate $\boldsymbol{\beta}$. You will not be asked to use these algorithms because we are interested in focusing on *statistics*, not *numerics*. Having said that, you should be aware that you are solving the least squares problem inefficiently and perhaps inaccurately.

To calculate (8.2), we need to calculate $\mathbf{X}^T\mathbf{X}$ and $\mathbf{X}^T\mathbf{y}$. In R, the transpose operation is `t()`, and matrix multiplication is `%*%`. Thus, these terms are obtained by the commands

```
1 xtx      = t(x) %*% x
2 xty      = t(x) %*% y
```

The inverse of a general matrix can be calculated using the `solve` command. However, the matrix we want to invert is symmetric, and it is faster and more accurate to invert

a symmetric matrix using the *Cholesky decomposition*. R can invert a matrix using the Cholesky decomposition as follows:

```
xtx.inv = chol2inv(chol(xtx))
```

Verify that `xtx %*% xtx.inv` equals the identity matrix (to within roundoff error).

In addition to estimating the parameters $\boldsymbol{\beta}$, you also should estimate the sum square error SSE and coefficient of determination R^2. The sum square error is

$$\text{SSE} = \left(\mathbf{y} - \mathbf{X}\hat{\boldsymbol{\beta}}\right)^T \left(\mathbf{y} - \mathbf{X}\hat{\boldsymbol{\beta}}\right). \tag{8.3}$$

In the notes, R^2 was estimated from *centered* variables. It is not necessary to center the variables *as long as you include the intercept as a predictor*. Therefore, your R function should *automatically insert* the intercept as a predictor. That is, you will give the function all the predictors except the intercept, and the function will insert the intercept among the predictors. You can generate a vector of 1s using the command `rep`, and then join this vector with the predictor matrix using the command `cbind`.

In general, R^2 can be calculated as

$$R^2 = 1 - \frac{\text{SSE}}{\text{SST}}, \tag{8.4}$$

where

$$\text{SST} = \sum_n \left(y_n - \hat{\mu}\right)^2. \tag{8.5}$$

Exercise 8.1. Write an R function that sets up the normal equations for solving (8.1), solves the normal equations to obtain the least squares estimates, and computes R^2. The function call based on the normal equations should be `regress.normal(y,x)`, where `y` and `x` are the appropriate vector and matrix of the regression model (8.1). The preamble of this function should be the following:

```
regress.normal = function(y,x,include.intercept=TRUE) {
#####################################################################
## DETERMINES THE LEAST SQUARES ESTIMATE OF B IN THE EQUATION Y = XB + E
## BASED ON THE NORMAL EQUATIONS
## INPUT:
##   Y[NTOT]:      N-DIMENSIONAL VECTOR OF PREDICTANDS
##   X[NTOT,MTOT]: N X M DIMENSIONAL MATRIX OF PREDICTORS
##   INCLUDE.INTERCEPT: INCLUDE THE INTERCEPT? (DEFAULT=TRUE)
## OUTPUT:
##   BHAT: M-DIMENSIONAL VECTOR OF ESTIMATES OF B
##   R2:  R-SQUARED
##   SSE: SUM SQUARE ERROR OF THE LEAST SQUARES PREDICTION
##   DOF: DEGREES OF FREEDOM OF THE SSE.
##   RES.SE: STANDARD ERROR OF THE RESIDUALS
#####################################################################
```

□

Exercise 8.2. Apply your function to the following random numbers:

```
> set.seed(1)
> ntot  = 20
> y     = rnorm(ntot); pred1 = rnorm(ntot); pred2 = rnorm(ntot)
> x     = cbind(pred1,pred2)
```

After running your function, print out $\hat{\beta}$, R^2, SSE, and dof produced by your function.

You should check your function by comparing with the R function `lm()`. To do this, apply the `lm` function as follows:

```
1  > xy.lm     = lm(y~x)
2  > summary(xy.lm)
3
4  Call:
5  lm(formula = y ~ x)
6
7  Residuals:
8      Min      1Q  Median      3Q     Max
9  -2.0455 -0.6782  0.2175  0.5049  1.4532
10
11 Coefficients:
12             Estimate Std. Error t value Pr(>|t|)
13 (Intercept)   0.1494     0.2072   0.721    0.481
14 xpred1       -0.1516     0.2504  -0.605    0.553
15 xpred2        0.2894     0.2695   1.074    0.298
16
17 Residual standard error: 0.9119 on 17 degrees of freedom
18 Multiple R-squared: 0.1078,        Adjusted R-squared: 0.002868
19 F-statistic: 1.027 on 2 and 17 DF,  p-value: 0.3791
```

In line 1, the function `lm` is called using the formula notation. Then, in line 2, the summary function is used to extract basic information about the linear model. For the purpose of the present homework, we are interested in only three parts of this summary: the coefficients, R^2, and residual standard error. The value of the coefficients are listed under `Coefficients: Estimate`, and are 0.1494, -0.1516, and 0.2894, corresponding to the three predictors: intercept, `pred1`, and `pred2`. These should match the values computed from `regress.normal` above. The R^2 is in the second to last line: 0.1078, and should agree with that calculated from `regress.normal`. Finally, the residual standard error derived from `regress.normal` should agree with the result from `lm` (line 17). □

Exercise 8.3. Apply your regression function to estimate the growth rate of atmospheric CO_2 concentration over the past half-century or so. State the growth rate in units of ppm/yr. The CO_2 concentration data can be downloaded as `co2_mm_mlo.txt` from the class website (which in turn was downloaded from http://www.esrl.noaa.gov/gmd/ccgg/trends/). This data set can be read into R as follows:

```
iyst      = 1960
iynd      = 2017

########################################
######## GET CO2 DATA
########################################
fdata = '/Users/delsole/data/indices/co2_mm_mlo.txt'
nskip = 72
col.names = c('year','month','date','average','interp','trend','#days')
co2.table = read.table(fdata,skip=nskip,col.names=col.names,na.strings=-99.99)

year.get = co2.table[,'year'] >= iyst & co2.table[,'year'] <= iynd
year.say = co2.table[year.get,'date']
month    = co2.table[year.get,'month']
co2      = co2.table[year.get,'average']
plot(year.say,co2,type="l",col="black",xlab='year',ylab='Parts Per Million')
```

You should change `fdata` to correspond to the data file on your computer. The resulting plot should reproduce the figure in the notes.

Unfortunately, there exists missing data. Therefore, *inside your R function*, you will have to strip out this missing data before applying the least squares method. I recommend including the following inside your R function:

```
ntot = length(y)
if ( length(x) %% ntot != 0) stop('x not dimensioned correctly')
mtot = length(x)/ntot
if ( ntot <= mtot ) stop('regression problem is not over-determined')

### STRIP MISSING DATA
dim(x)     = c(ntot,mtot)
is.missing = is.na(y) | is.na(rowSums(x))
x.good     = x[!is.missing,]
y.good     = y[!is.missing ]
nsamp      = sum(!is.missing)
```

Note that the correct sample size after missing data has been stripped is `nsamp`. This is important when you augment the predictor matrix $\mathbf{X}$ by a column of ones.

After running your function, print out $\hat{\beta}$, R^2, SSE, and dof produced by your function. You can check your calculations against the built-in R function `lm` (you will need to use the `na.action=na.omit` option to deal with missing data). □

Exercise 8.4. Compute the residuals of the regression equation. Make a plot of them, and *state the first 50 values* as a vector (e.g., `as.numeric(residuals[1:50])`. □

Exercise 8.5. Use your regression function to estimate the annual cycle of the CO_2 concentration. The annual cycle should be defined as the first two Fourier harmonics of the annual cycle (i.e., sin/cos function with periods of 12 months and 6 months). The predictor matrix for *just these harmonics* can be constructed in R as follows:

```
year = iyst + (1:ntot - 0.5)/12
year.shift = year - 1960
t = seq(year.shift)/12
nharm = 2
x = NULL
for ( n in 1:nharm) x = cbind(x,cos(2*pi*t*n),sin(2*pi*t*n))
colnames(x)  = c(paste(rep(c('cos','sin'),nharm),
  rep(1:nharm,each=2),sep=""))
```

State the five coefficients of this fit (i.e, the intercept, the 2 coefficients for the sin function, 2 coefficients for the cosine function). *Print out the resulting* coefficients, R^2, dof, and SSE. □

Exercise 8.6. Plot the residuals after the annual cycle has been removed. Superimpose a plot of the actual CO_2 data for comparison. To do this, you will need to add a constant term to the residuals to make them fit on the same figure; state what constant should you use and why. □