Problem Set 2

INF 511

Purnabhishek Sripathi

1 Manual OLS analysis to estimate $\hat{B}$

In this problem set, you will conduct OLS analysis on a data set that you will simulate.

1.1 Use the following parameters to generate the X covariate matrix.

First, you will need to generate the matrices and vectors that are needed to generate the data. Remember the X matrix is n x p, where n is the number of data observations, and p is the number of parameters. Here, $p = 2$, for the intercept and one slope (i.e., we are only dealing with one covariate, x).

```
n = 50
p = 2

x0 = rep(1, times = n)

# Randomly draw from a probability distribution to generate
# n values for x1.
# x1 = ?

# Create the matrix, xmat, using the x0 and x1 column vectors
x1 <- rnorm(50, mean = 86, sd = 36)
xmat_data <-cbind(x0,x1)
print(xmat_data)
```

```
      x0        x1
 [1,]  1  82.82193
 [2,]  1  88.19979
 [3,]  1  97.17624
 [4,]  1  62.28689
 [5,]  1 101.18066
 [6,]  1 -52.85621
 [7,]  1  77.89471
 [8,]  1  62.94068
 [9,]  1 115.18577
[10,]  1 132.32062
[11,]  1 113.56459
[12,]  1 163.99988
[13,]  1  22.26779
[14,]  1  91.91864
[15,]  1  98.69326
[16,]  1  61.66827
[17,]  1  16.55149
```

```
[18,]   1  59.65937
[19,]   1  74.41931
[20,]   1  72.28812
[21,]   1  94.38544
[22,]   1  81.22974
[23,]   1  49.08375
[24,]   1  68.19614
[25,]   1  71.94110
[26,]   1  80.06162
[27,]   1 101.61428
[28,]   1  72.85389
[29,]   1  24.79419
[30,]   1 150.80705
[31,]   1 104.98928
[32,]   1  92.09207
[33,]   1  64.13385
[34,]   1 100.90840
[35,]   1 132.28723
[36,]   1  68.00960
[37,]   1  28.34622
[38,]   1 102.94074
[39,]   1  76.82070
[40,]   1 163.90000
[41,]   1 108.32537
[42,]   1 113.98379
[43,]   1 108.77790
[44,]   1  55.14160
[45,]   1 116.96751
[46,]   1  90.14358
[47,]   1 153.77519
[48,]   1  92.84909
[49,]   1  63.60285
[50,]   1  51.73725
```

1.2 Create an array of residual error values

Remember that $\epsilon_i \sim N(0, \sigma^2)$. Draw ϵ_i values randomly from a normal distribution.

```r
# Assign a value for sigma, the residual standard deviation
# sigma = ?
sigma = 2.86
# Assign values for epsilon, drawing randomly from a normal distribution
# epsilon = ?
epsilon_data <- rnorm(1,mean=86, sd=26)
print(epsilon_data)
```

```
[1] 123.4202
```

1.3 Calculate the observed values of Y.

```r
# Use the following values of intercept and slope
# betas[1]: intercept
# betas[2]: slope
betas = c(2.5, 2.8)

# Calculate the values of y, using xmat, betas, and epsilon
# y = ?
y <- xmat_data*betas+epsilon_data
print(y)
```

```
            x0        x1
 [1,] 125.9202 330.47504
 [2,] 126.2202 370.37963
 [3,] 125.9202 366.36080
 [4,] 126.2202 297.82350
 [5,] 125.9202 376.37187
 [6,] 126.2202 -24.57717
 [7,] 125.9202 318.15699
 [8,] 126.2202 299.65412
 [9,] 125.9202 411.38465
[10,] 126.2202 493.91794
[11,] 125.9202 407.33168
[12,] 126.2202 582.61988
[13,] 125.9202 179.08968
[14,] 126.2202 380.79240
[15,] 125.9202 370.15336
[16,] 126.2202 296.09136
[17,] 125.9202 164.79894
[18,] 126.2202 290.46644
[19,] 125.9202 309.46848
[20,] 126.2202 325.82694
[21,] 125.9202 359.38382
[22,] 126.2202 350.86347
[23,] 125.9202 246.12959
[24,] 126.2202 314.36939
[25,] 125.9202 303.27296
[26,] 126.2202 347.59275
[27,] 125.9202 377.45591
[28,] 126.2202 327.41111
[29,] 125.9202 185.40568
[30,] 126.2202 545.67996
[31,] 125.9202 385.89341
[32,] 126.2202 381.27802
[33,] 125.9202 283.75483
[34,] 126.2202 405.96372
[35,] 125.9202 454.13829
[36,] 126.2202 313.84708
[37,] 125.9202 194.28576
[38,] 126.2202 411.65430
[39,] 125.9202 315.47196
[40,] 126.2202 582.34021
[41,] 125.9202 394.23365
```

```
[42,] 126.2202 442.57482
[43,] 125.9202 395.36496
[44,] 126.2202 277.81670
[45,] 125.9202 415.83900
[46,] 126.2202 375.82223
[47,] 125.9202 507.85818
[48,] 126.2202 383.39766
[49,] 125.9202 282.42733
[50,] 126.2202 268.28453
```

1.4 Calculate $\hat{B}$

Now we have the data observations. Using the example code already provided, calculate the coefficients that we estimate from the data using OLS, stored in matrix, $\hat{B}$. Use the `solve()` function.

```
# Enter your code here

hatB <- solve(t(xmat_data) %*% xmat_data)
hat1B <- solve(hatB %*% t(xmat_data)%*% (y))

print(hat1B)
```

```
              x0         x1
x0  7.932539e-03 -0.3565944
x1 -1.636817e-07  0.3726134
```

```
column_1 <- hat1B[1]
column_2 <- hat1B[2]
```

2 OLS analysis using `lm()`

Use the `lm()` function to estimate the model coefficients. Store the estimated coefficients in an array.

```
# Enter your code here
model_coefficients <- lm(y ~ xmat_data)
coef(model_coefficients)
```

```
                      x0       x1
(Intercept) 1.260655e+02 120.6459
xmat_datax0           NA       NA
xmat_datax1 5.537810e-05   2.6838
```

```
#fetching columns
col1 <- model_coefficients[1]
col2 <- model_coefficients[2]

# Store the estimated coefficients in an array.
result_arr <- array(c(col1,col2))
print(result_arr)
```

```
[[1]]
                      x0       x1
```

```
(Intercept) 1.260655e+02 120.6459
xmat_datax0           NA       NA
xmat_datax1 5.537810e-05   2.6838

[[2]]
           x0         x1
1  -0.1499050 -12.448339
2   0.1497972  13.023147
3  -0.1506999 -15.086661
4   0.1512322  10.012069
5  -0.1509217 -15.822675
6   0.1576086  -3.367553
7  -0.1496321 -11.542716
8   0.1511960  10.088040
9  -0.1516972 -18.396815
10  0.1473539  18.149985
11 -0.1516075 -18.098841
12  0.1455995  21.831114
13 -0.1465516  -1.318485
14  0.1495913  13.455277
15 -0.1507839 -15.365490
16  0.1512665   9.940186
17 -0.1462351  -0.267829
18  0.1513777   9.706751
19 -0.1494397 -10.903936
20  0.1506784  11.174212
21 -0.1505454 -14.573713
22  0.1501832  12.213227
23 -0.1480366  -6.247260
24  0.1509050  10.698723
25 -0.1493024 -10.448441
26  0.1502479  12.077492
27 -0.1509457 -15.902374
28  0.1506470  11.239955
29 -0.1466915  -1.782837
30  0.1463301  20.298108
31 -0.1511326 -16.522699
32  0.1495816  13.475430
33 -0.1488701  -9.013469
34  0.1490934  14.499886
35 -0.1526443 -21.540064
36  0.1509153  10.677048
37 -0.1468882  -2.435701
38  0.1489809  14.736045
39 -0.1495727 -11.345312
40  0.1456051  21.819508
41 -0.1513173 -17.135873
42  0.1483693  16.019246
43 -0.1513424 -17.219048
44  0.1516279   9.181788
45 -0.1517959 -18.724299
46  0.1496896  13.249015
47 -0.1538342 -25.489551
48  0.1495397  13.563395
```

```
49 -0.1488407  -8.915870
50  0.1518164   8.786202
```

3 Visualize and compare the analyses

Plot the observed data in a scatter plot using `plot()`. As in the code already provided, plot three lines: (1) linear relationship with true values of coefficients, B; (2) linear relationship with coefficients estimated from manual OLS analysis; and (3) linear relationship with coefficients estimated from `lm()`. Each line should be a different color and a different line type (option `lty` in the `abline()` function.) Finally, create chunk options (i.e., using the `#|` syntax) to specify the plot's height and width.

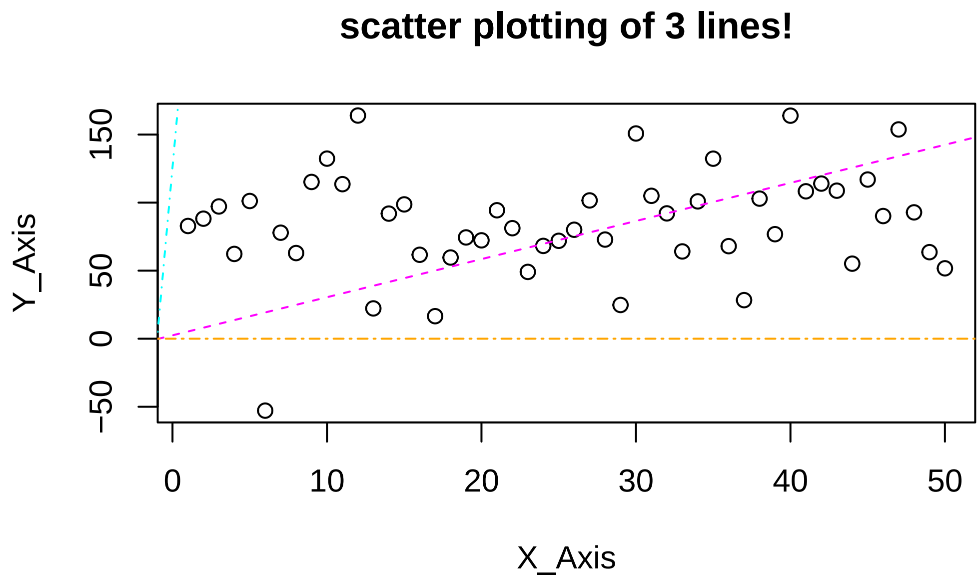
```r
# Enter your code here


#observed data in a scatter plot using `plot()`
plot(x1,xlab="X_Axis", ylab="Y_Axis", main="scatter plotting of 3 lines!")

#(1) linear relationship with true values of coefficients
abline(betas[1],betas[2], col="magenta",lty=2)

#(2) linear relationship with coefficients estimated from manual OLS analysis
abline(column_1,column_2,col="orange",lty=6)

#(3) linear relationship with coefficients estimated from `lm()`
abline(coef(model_coefficients),coef(model_coefficients),col="cyan",lty=4)
```



4 Rendering

Render this document as a `.PDF` file. Upload the rendered `.PDF` and the original `.QMD` onto BBLearn.