Homework 2

TJ Sipin

2022-07-06

Problem 1 (Statistical thinking).

a. Consider $\text{Cov}(\mathbf{A}y) = \mathbf{A}\text{Cov}(y)\mathbf{A}^T$, where $\mathbf{A} = (X^TX)^{-1}X^T$ and $y = \beta X + \epsilon$. Taking the notation

$$\hat{\beta} = (X^TX)^{-1}X^Ty,$$

we have

$$\begin{aligned}
\text{Cov}(\hat{\beta}) &= \text{Cov}((X^TX)^{-1}X^Ty) \\
&= \text{Cov}((X^TX)^{-1}X^T(X\beta + \epsilon)) \\
&= \text{Cov}((X^TX)^{-1}X^TX\beta + (X^TX)^{-1}X^T\epsilon) \\
&= \text{Cov}(I\beta + (X^TX)^{-1}X^T\epsilon) \\
&= \text{Cov}(X^TX)^{-1}X^T\epsilon) \\
&= \text{Cov}(\mathbf{A}\epsilon) \\
&= \mathbf{A}\text{Cov}(\epsilon)\mathbf{A}^T \\
&= (X^TX)^{-1}X^T\sigma^2((X^TX)^{-1}X^T)^T \\
&= \sigma^2(X^TX)^{-1}X^TX\left((X^TX)^{-1}\right)^T \\
&= \sigma^2(X^TX)^{-1}.
\end{aligned}$$

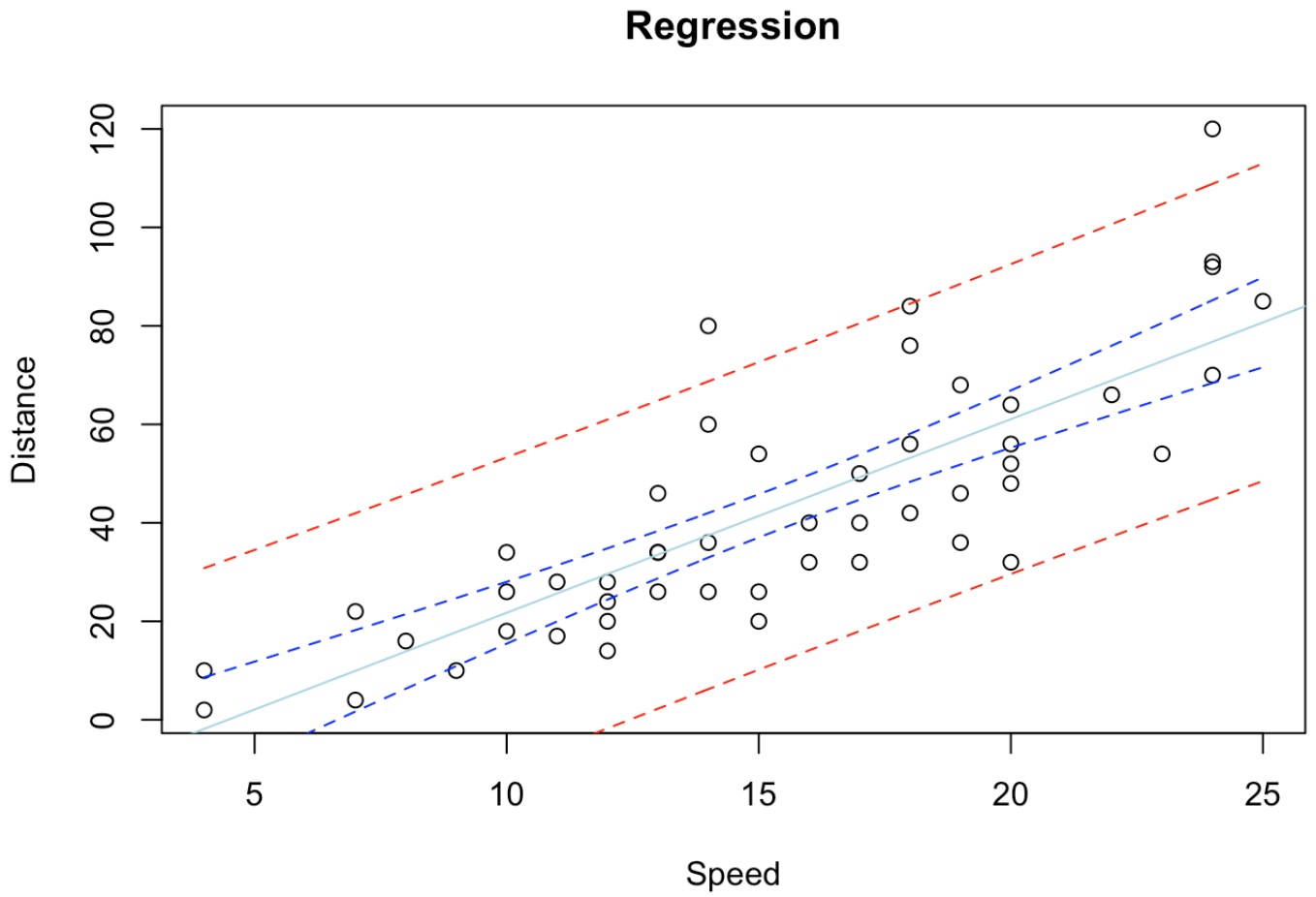
The variance of $\hat{\beta}_1$ is given by the second diagonal of the above matrix. That is,

$$(i) \qquad \text{Var}(\hat{\beta}_1) = \left[\sigma^2(X^TX)^{-1}\right]_{2,2}.$$

b. Splitting the possible range of x into 100 mutually exclusive windows and collecting the equal amount of sample from each slot is more efficient in terms of estimating β_1. So when we split the possible range of x into 100 *mutually exclusive* windows, then we decrease variance of x, which in turn, decreases the variance of $\hat{\beta}$, as we see in (i).

Problem 2 (Prediction and confidence intervals)

```r
# Regression for cars data
lm.out <- lm(dist ~ speed, data = cars)

# create new predictor
newx = seq(min(cars$speed), max(cars$speed), by = 0.1)

# get confidence interval for each element in newx
conf_interval <- predict(lm.out,
                         newdata=data.frame(speed=newx),
                         interval = "confidence",
                         level = 0.95)

# get prediction interval for each element in newx
pred_interval <- predict(lm.out,
                         newdata = data.frame(speed=newx),
                         interval = "prediction",
                         level = 0.95)
# plot data points
plot(cars$speed, cars$dist,
     main = "Regression",
     xlab = "Speed",
     ylab = "Distance")

# regression line
abline(lm.out, col = 'lightblue')

# add lower end of CI
lines(newx, conf_interval[,2], col = 'blue', lty = 2)
# add upper end of CI
lines(newx, conf_interval[,3], col = 'blue', lty = 2)
# add lower end of PI
lines(newx, pred_interval[,2], col = 'red', lty = 2)
# add upper end of PI
lines(newx, pred_interval[,3], col = 'red', lty = 2)
```



The prediction interval is wider than the confidence interval since the confidence interval pertains to the median (at a 95% level), while the prediction interval pertains to all points in the relationship of distance and speed within a 95% level.

Problem 3 (Weighted least squares).

a. Derive the BLUE of β_0 and β_1 for

$$y_i \sim^{\text{indep}} N(\beta_0 + \beta_1 x_i, x_i^2 \sigma^2), \qquad i = 1, \dots, n.$$

Solution. Here, we have $\epsilon \sim N(0, \sigma^2 \mathbf{V})$, where $\mathbf{V} = x_i^2 I, \text{ for } i = 1, \dots, n$. The OLS estimator is BLUE when $\epsilon \sim (0, \sigma^2 I)$. Since $\mathbf{V}$ can be decomposed as $\mathbf{V} = (x_i I)(x_i I)^T$, it follows that

$$\text{Cov}((x_i I)^{-1} \epsilon) = \sigma^2 (x_i I)^{-1} x_i I (x_i I)^T ((x_i I)^{-1})^T = \sigma^2 I.$$

This allows us to change the OLS problem to

$$(x_i I)^{-1} Y = (x_i I)^{-1} X\beta + (x_i I)^{-1} \epsilon, \quad (x_i I)^{-1} \sim N(0, \sigma^2 I).$$

Thus, we can get BLUE from the OLS estimator from the changed problem:

$$
\begin{aligned}
\hat{\beta} &= (X^T((x_iI)^T)^{-1}(x_iI)^{-1}X)^{-1}((x_iI)^{-1}X)^T(x_iI)^{-1}Y \\
&= (X^T(x_iIx_iI^T)^{-1}X)^{-1}X^T(x_iIx_iI^T)^{-1}Y \\
&= (X^T(x_iIx_iI^T)^{-1}X)^{-1}X^T(x_iIx_iI^T)^{-1}Y \\
&= (X^T(x_i^2I)^{-1}X)^{-1}X^T(x_i^2I)^{-1}Y,
\end{aligned} \quad (ii)
$$

as long as $X^T(x_i^2I)^{-1}X$ is nonsingular. That is, if $n > 1$ and at least one $x_i \neq 0$. The BLUE of β_0 is the first row of the above matrix (ii) and the BLUE of β_1 is the second row.

b.

```
# library(matlib)
set.seed(2022)
x = rnorm(20, 30, 5)
y = rnorm(20, 40, 5)


lm.out.3b = lm(y ~ x)
xm = matrix(c(rep(1, times = 20), x), nrow = 20)
ym = as.matrix(y)
q  = diag(x = x^2, nrow = 20, ncol = 20)

beta.hat = solve(t(xm)%*%solve(q)%*%xm)%*%t(xm)%*%solve(q)%*%ym
beta.hat
```

```
##            [,1]
## [1,] 35.8811172
## [2,]  0.1709089
```

The BLUE of $\beta_0 = 38.28292797$ and the BLUE of $\beta_1 = 0.08383732$ based on what was derived in (a).

c.

```
lm.out.3c = lm(y ~ x, weights = 1/(x**2))
lm.out.3c
```

```
##
## Call:
## lm(formula = y ~ x, weights = 1/(x^2))
##
## Coefficients:
## (Intercept)            x
##     35.8811       0.1709
```

The results are the same as the results in (b). Here, the `weights =` option is giving us the inverse of V. This is since $Y \sim N(\mu, V\sigma^2) \implies \frac{Y}{V} \sim N(\mu, \sigma^2)$.

Problem 4 (Wheezing data).

```
wheeze <- read.csv('data/wheeze_wmiss.csv')
wheeze %>% head()
```

```
##   case t wheeze kingston age smoke
## 1    1 1      1        0   9     0
## 2    1 2      1        0  10     0
## 3    1 3      1        0  11     0
## 4    1 4      0        0  12     0
## 5    2 1      1        1   9     1
## 6    2 2      1        1  10     2
```

```
summary(wheeze)
```

```
##       case              t            wheeze          kingston
##  Min.   : 1.00   Min.   :1.00   Min.   :0.0000   Min.   :0.0000
##  1st Qu.: 4.75   1st Qu.:1.75   1st Qu.:0.0000   1st Qu.:0.0000
##  Median : 8.50   Median :2.50   Median :0.0000   Median :0.0000
##  Mean   : 8.50   Mean   :2.50   Mean   :0.2969   Mean   :0.4531
##  3rd Qu.:12.25   3rd Qu.:3.25   3rd Qu.:1.0000   3rd Qu.:1.0000
##  Max.   :16.00   Max.   :4.00   Max.   :1.0000   Max.   :1.0000
##
##       age            smoke
##  Min.   : 9.00   Min.   :0.00
##  1st Qu.: 9.75   1st Qu.:0.00
##  Median :10.50   Median :1.00
##  Mean   :10.50   Mean   :0.78
##  3rd Qu.:11.25   3rd Qu.:1.00
##  Max.   :12.00   Max.   :2.00
##                  NA's   :14
```

a.

```
wheeze <- wheeze %>%
  mutate(smoke = as.factor(smoke))

# %>%
#   mutate(wheeze = as.factor(wheeze)) %>%
#   mutate(kingston = as.factor(kingston))


wheeze_binom <- glm(wheeze ~ . - t, data = wheeze, family = "binomial")

summary(wheeze_binom)
```

```
##
## Call:
## glm(formula = wheeze ~ . - t, family = "binomial", data = wheeze)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -1.3820  -0.8890  -0.6402   1.2748   1.7231
##
## Coefficients:
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept)  2.00511    3.18201   0.630    0.529
## case         0.03012    0.07016   0.429    0.668
## kingston     0.61103    0.64974   0.940    0.347
## age         -0.25530    0.30083  -0.849    0.396
## smoke1      -1.16160    0.72795  -1.596    0.111
## smoke2      -0.67572    0.99893  -0.676    0.499
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 62.687  on 49  degrees of freedom
## Residual deviance: 58.756  on 44  degrees of freedom
##   (14 observations deleted due to missingness)
## AIC: 70.756
##
## Number of Fisher Scoring iterations: 4
```

The coefficients for `smoke1` and `smoke2` both have a p-value of greater than 0.05, thus smoking is not significant at the 95% level.

b.

```
library(lme4)
```

```
## Warning: package 'lme4' was built under R version 4.1.2
```

```
## Loading required package: Matrix
```

```
## Warning: package 'Matrix' was built under R version 4.1.2
```

```
wheeze_binom_random_intercept <- glmer(wheeze ~ kingston + smoke + (1|case), family =
'binomial', data = wheeze)
summary(wheeze_binom_random_intercept)$coefficients
```

```
##               Estimate Std. Error    z value  Pr(>|z|)
## (Intercept) -0.8462974  0.9107001 -0.9292822 0.3527429
## kingston     0.9032460  1.0581880  0.8535780 0.3933389
## smoke1      -1.0606774  0.9053678 -1.1715431 0.2413805
## smoke2      -1.1177135  1.3483172 -0.8289692 0.4071218
```

Here the standard errors for `smoke1` and `smoke2` are 0.91 and 1.35, respectively, compared to 0.73 and 1.0 from the GLM. So the standard errors for the random intercept gets larger for both. Perhaps this is due to the lower number of observations as the observations go from 64 (total observations) to 4 (grouped by individual child).

The p-values are 0.24 and 0.40 for the random intercept model, compared to 0.11 and 0.50 from the GLM. The p-value for smoke1 gets bigger, while the p-value for smoke2 gets smaller.

Appendix: All code for this report

```
knitr::opts_chunk$set(echo = TRUE)
library(dplyr)
# Regression for cars data
lm.out <- lm(dist ~ speed, data = cars)

# create new predictor
newx = seq(min(cars$speed), max(cars$speed), by = 0.1)

# get confidence interval for each element in newx
conf_interval <- predict(lm.out,
                         newdata=data.frame(speed=newx),
                         interval = "confidence",
                         level = 0.95)

# get prediction interval for each element in newx
pred_interval <- predict(lm.out,
                         newdata = data.frame(speed=newx),
                         interval = "prediction",
                         level = 0.95)
# plot data points
plot(cars$speed, cars$dist,
     main = "Regression",
     xlab = "Speed",
     ylab = "Distance")

# regression line
abline(lm.out, col = 'lightblue')

# add lower end of CI
lines(newx, conf_interval[,2], col = 'blue', lty = 2)
# add upper end of CI
lines(newx, conf_interval[,3], col = 'blue', lty = 2)
```

```
# add lower end of PI
lines(newx, pred_interval[,2], col = 'red', lty = 2)
# add upper end of PI
lines(newx, pred_interval[,3], col = 'red', lty = 2)
# library(matlib)
set.seed(2022)
x = rnorm(20, 30, 5)
y = rnorm(20, 40, 5)


lm.out.3b = lm(y ~ x)
xm = matrix(c(rep(1, times = 20), x), nrow = 20)
ym = as.matrix(y)
q  = diag(x = x^2, nrow = 20, ncol = 20)

beta.hat = solve(t(xm)%*%solve(q)%*%xm)%*%t(xm)%*%solve(q)%*%ym
beta.hat
lm.out.3c = lm(y ~ x, weights = 1/(x**2))
lm.out.3c
wheeze <- read.csv('data/wheeze_wmiss.csv')
wheeze %>% head()
summary(wheeze)
wheeze <- wheeze %>%
  mutate(smoke = as.factor(smoke))

# %>%
#   mutate(wheeze = as.factor(wheeze)) %>%
#   mutate(kingston = as.factor(kingston))


wheeze_binom <- glm(wheeze ~ . - t, data = wheeze, family = "binomial")

summary(wheeze_binom)
library(lme4)
wheeze_binom_random_intercept <- glmer(wheeze ~ kingston + smoke + (1|case), family =
'binomial', data = wheeze)
summary(wheeze_binom_random_intercept)$coefficients
```