Homework10

Jared Andreatta

2025-04-03

```r
suppressMessages(library(fields))
```

```
## Warning: package 'fields' was built under R version 4.4.2
```

```
## Warning: package 'spam' was built under R version 4.4.2
```

```r
knitr::opts_chunk$set(echo = TRUE)
```

The linear model and background on constrained OLS

Throughout assume a linear model

$$\mathbf{y} = X\beta + \mathbf{e}$$

where $\mathbf{y}$ is an vector of length n, β of length p, and X is a known, full rank matrix, $n \times p$. $\mathbf{e}_i$ are independent $N(0, \sigma^2)$ (aka $\mathbf{e} \sim MN(0, \sigma^2 I_n)$). β and σ are (of course) unknown.

Suppose we have q linear constraints on β as $A\beta = 0$ where A is $q \times p$. For convenience we will call this the null hypothesis on β and when it satisifes this constraint β_H. From Seber and Lee (3.38) we have the (amazing) formula for constrained OLS in terms of the *unconstrained* OLS estimate.

$$\hat{\beta}_H = \hat{\beta} - (X^T X)^{-1} A^T (A(X^T X)^{-1} A^T)^{-1} A\hat{\beta}$$

Problem 1

- Show that $A\hat{\beta}_H = 0$. (This is a sanity check that we have satified the constraint!)

Proof. First, we left multiply the expression for $\hat{\beta}_H$, as given above, by A.

$$A\hat{\beta}_H = A\hat{\beta} - [A(X^T X)^{-1} A^T][A(X^T X)^{-1} A^T]^{-1} A\hat{\beta}$$

Clearly, this becomes

$$A\hat{\beta}_H = A\hat{\beta} - IA\hat{\beta} = A\hat{\beta} - A\hat{\beta} = 0$$

Hence, $A\hat{\beta}_H = 0$. □

- Let $\mathbf{u} = X\hat{\beta}_H - X\hat{\beta}$ and so $SS_1 = \mathbf{u}^T\mathbf{u}$ is part of the numerator of the F statistic for testing $H_0 : A\beta = 0$.

$$F = \frac{(SS_1/q)}{\hat{\sigma}^2}$$

  Simply SS_1.

Proof. We have $SS_1 = u^T u$ where $u = X\hat{\beta}_H - X\hat{\beta}$. We can rewrite u as $u = X(\hat{\beta}_H - \hat{\beta})$. Therefore, we have

$$u^T u = (\hat{\beta}_H - \hat{\beta})^T X^T X(\hat{\beta}_H - \hat{\beta})$$

Note that when we take the difference between $\hat{\beta}_H$ and $\hat{\beta}$ we get

$$\hat{\beta}_H - \hat{\beta} = -(X^T X)^{-1} A^T (A(X^T X)^{-1} A^T)^{-1} A\hat{\beta}$$

We can use this to expand $u^T u$. Note that the terms $(X^T X)^{-1}$ and $(A(X^T X)^{-1} A^T)^{-1}$ are symmetric. After a lot of ugly algebra, we have

$$\begin{aligned}
u^T u &= (-(X^T X)^{-1} A^T (A(X^T X)^{-1} A^T)^{-1} A\hat{\beta})^T (X^T X)(-(X^T X)^{-1} A^T (A(X^T X)^{-1} A^T)^{-1} (A\hat{\beta})) \\
&= (A\hat{\beta})^T (A(X^T X)^{-1} A^T)^{-1} A(X^T X)^{-1} (X^T X)(X^T X)^{-1} A^T (A(X^T X)^{-1} A^T)^{-1} (A\hat{\beta}) \\
&= (A\hat{\beta})^T (A(X^T X)^{-1} A^T)^{-1} (A(X^T X)^{-1} A^T)(A(X^T X)^{-1} A^T)^{-1} (A\hat{\beta}) \\
&= (A\hat{\beta})^T (A(X^T X)^{-1} A^T)^{-1} (A\hat{\beta})
\end{aligned}$$

Hence, $SS_1 = (A\hat{\beta})^T (A(X^T X)^{-1} A^T)^{-1} (A\hat{\beta})$. □

- Now let A be the special case of $A = [0, \ldots, 1]$ so that $A\hat{\beta}_H = 0$ is just $\beta_k = 0$ Show that SS_1 has the simplified form: $SS1 = \hat{\beta}_k{}^2 / H_k$ where H_k is the k^{th} diagonal element of $(X^T X)^{-1}$

Proof. A is a row vector in which the $k-th$ element is 1, while the rest are simply 0. So when we multiply A by $\hat{\beta}_H$, this effectively "picks out" the $k-th$ element of $\hat{\beta}_H$, namely $\hat{\beta}_k$. Therefore, we have

$$A\hat{\beta}_H = \hat{\beta}_k$$

From the last problem, we know that $SS_1 = (A\hat{\beta})^T (A(X^T X)^{-1} A^T)^{-1} (A\hat{\beta})$, therefore, we can say in this case that

$$SS_1 = (\hat{\beta}_k))^T (A(X^T X)^{-1} A^T)^{-1} (\hat{\beta}_k)$$

As A "picked out the $k-th$ element of $\hat{\beta}_H$, then it follows that it will pick out the k, k element of $(X^T X)^{-1}$, which we can call H_k. Therefore, we have

$$(A(X^T X)^{-1} A^T)^{-1} = (H_k)^{-1} = \frac{1}{H_k}$$

Putting it all together, we have

$$SS_1 = \frac{1}{H_k}(\hat{\beta}_k)^T(\hat{\beta}_k) = \frac{\hat{\beta_k}^2}{H_k}$$

$\square$.

Problem 2

- Explain why the F statistic in this special case from Problem 1 is just

  $$F = \hat{\beta_k}^2 / SE_k^2$$

  where SE_k is the standard error for $\hat{\beta}_k$ (and substituting in $\hat{\sigma}$ for σ).

For this case, we have 1 restriction, i.e. $q = 1$, so we can rewrite the F statistic as

$$F = \frac{SS_1}{\hat{\sigma}^2} = \frac{\hat{\beta}_k^2}{\hat{\sigma}^2 H_k}$$

For $\hat{\beta}_H$, the standard error is

$$SE_{\hat{\beta}_H} = \sqrt{\hat{\sigma}^2 (X^T X)^{-1}}$$

So it follows that for $\hat{\beta}_k$, we have

$$SE_{\hat{\beta}_k} = \sqrt{\hat{\sigma}^2 H_k}$$

Therefore, it is clear that

$$F = \frac{\hat{\beta}_k^2}{\hat{\sigma}^2 H_k} = \frac{\hat{\beta}_k^2}{SE_{\hat{\beta}_k}^2}$$

- What is the distribution of $\hat{\beta}/SE_k$?

Under OLS assumptions, $\hat{\beta}/SE_k$ follows a t-distribution with $n - p$ degrees of freedom.

Problem 3

Setup for Audi A4 data.

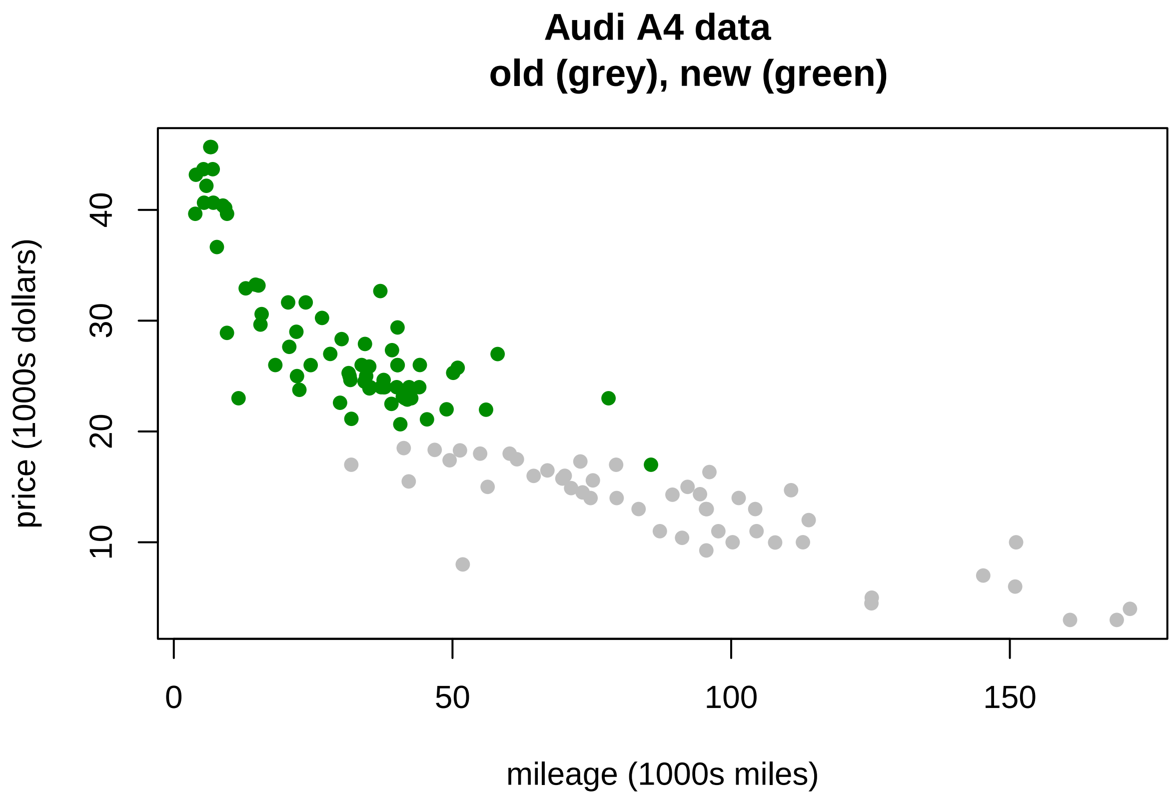
Adjust the directory path to your laptop.

Reformat/Wrangle Audi data

```r
load("AudiA4.rda" )
#head( AudiA4)
#convenient scalings and naming
mileage<- AudiA4$mileage/1000
price<- AudiA4$price/1000
old<- ifelse( AudiA4$year<= 2016,1,0)
new<- ifelse( AudiA4$year > 2016,1,0)
y<- price
```

A plot of the data – always a good idea even for a "theory" class.

```r
plot( mileage,price,
      col= ifelse(old ==1, "grey", "green4"),
      pch=16,
      xlab="mileage (1000s miles)", ylab="price (1000s dollars)")
title("Audi A4 data
      old (grey), new (green) ")
```



Setup a constraint on a mixed cubic and linear fit

```r
mileCut<- 38
indC<- ifelse( mileage <= mileCut,1,0)
indL<- ifelse( mileage <= mileCut,0,1)

# mixed cubic/linear X matrix
X<- cbind(indC,mileage*indC,mileage^2*indC,mileage^3*indC,
```

```
          indL,mileage*indL)
A<- rbind(
  c(1,mileCut,  mileCut^2,   mileCut^3, -1, -mileCut),
  c(0,      1,  2*mileCut, 3*mileCut^2,  0,       -1),
  c(0,      0,          2,   6*mileCut,  0,        0)
)
```

- Based on this X matrix, explain the form for the unconstrained mileage curve.

The **indC** and **indL** are activated based on mileCut to indicate if it is high or low mileage (i.e. if its greater than or equal to/less than 38). If **indC** is activated, then this is the cubic part of the piecewise function. If **indL** is activated, then this is the linear part of the function.

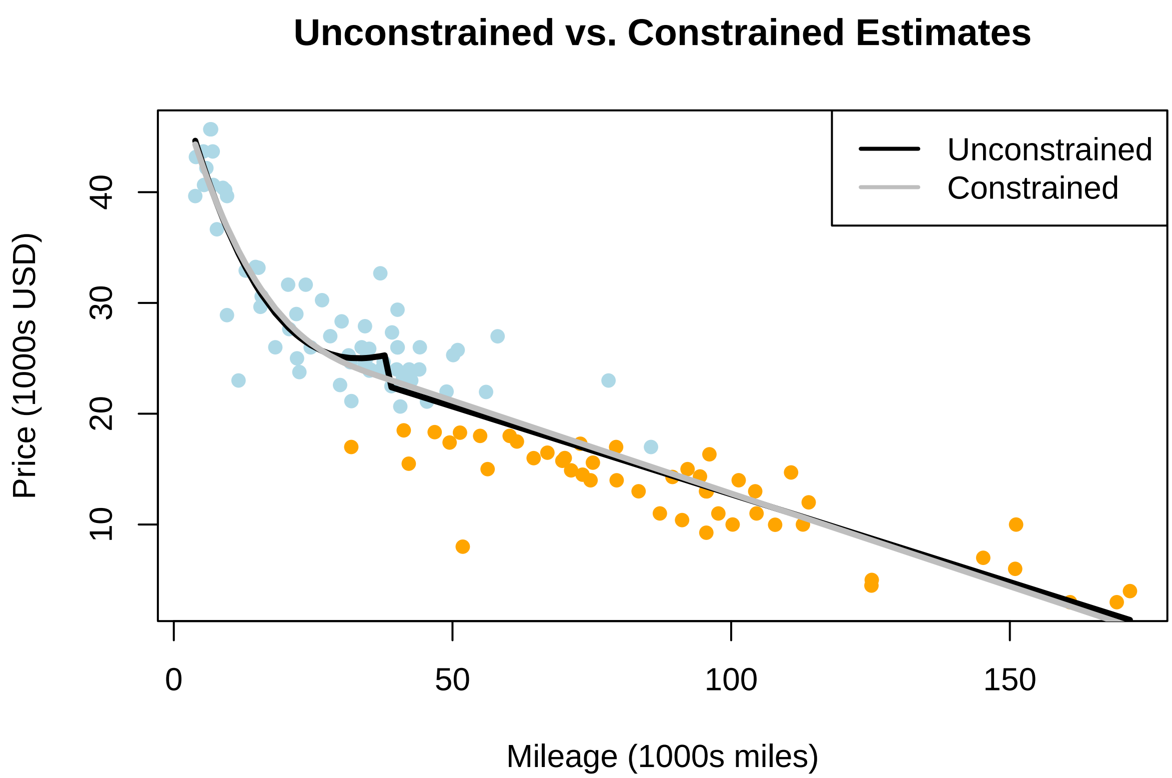
- Explain how the curve is changed based on these constraints in `A`.

The first row ensures that the cubic part of the function and the linear part of the function intersect at the same price value for some value of **mileCut**. The second row (first-order derivative of the function) ensures that the cubic and linear parts have the same slope for some value of **mileCut**. The third row (second-order derivative of the function) is a function of the curvature for the cubic segment, which is simply a linear function itself. The curvature of the linear segment is 0, since it is a linear function.

- Estimate the unconstrained and constrained OLS estimates and plot these curves on a scatterplot of the data. (I just found these explicitly using the basic linear algebra rather than using `lm`.)

```
# Estimators #
XTXinv <- solve(t(X) %*%X) # (X^TX)^-1
beta_unconstrained <- XTXinv %*% t(X) %*% y # Unconstrained OLS
A_XTX_AT <- A %*% XTXinv %*% t(A) # A(X^TX)^-1A^T
beta_constrained <- beta_unconstrained - XTXinv %*% t(A) %*% solve(A_XTX_AT) %*% (A %*% beta_unconstrai

# Y Hats #
y_hat_unconstrained <- X %*% beta_unconstrained
y_hat_constrained   <- X %*% beta_constrained

# Plotting #

# Scatterplot
plot(mileage, price,
     col = ifelse(old == 1, "orange", "lightblue"),
     pch = 16,
     xlab = "Mileage (1000s miles)",
     ylab = "Price (1000s USD)",
     main = "Unconstrained vs. Constrained Estimates")

ord <- order(mileage)

# Unconstrained Curve
lines(mileage[ord], y_hat_unconstrained[ord], col = "black", lwd = 3)

# Constrained Curve
```

```
lines(mileage[ord], y_hat_constrained[ord], col = "grey", lwd = 3)

# Legend:
legend("topright",
       legend = c("Unconstrained", "Constrained"),
       col = c("black", "grey"),
       lwd = 2)
```



- Do a test of the hypothesis $H_0 : A\beta = 0$ at the 95 percent level of confidence and also report the p-value for this test.

We have a p-value of .2756, so we fail to reject the null hypothesis. This suggests that the constrained model outperforms the unconstrained model.

```
# RSS vals
RSS_H <- t(y- X %*% beta_constrained) %*% (y- X %*% beta_constrained) # RSS for constrained
RSS <- t(y- X %*% beta_unconstrained) %*% (y- X %*% beta_unconstrained) # RSS unconstrained

# Params
q <- nrow(A)          # Cosntraints
n <- length(y)        # Sample size
p <- ncol(X)          # Number of params in the unconstrained model

# Results
F_stat <- ((RSS_H - RSS) / q) / (RSS / (n - p))
```

```r
p_value_F <- 1 - pf(F_stat, q, n - p)
print(F_stat)
```

```
##          [,1]
## [1,] 1.307253
```

```r
print(p_value_F)
```

```
##           [,1]
## [1,] 0.2756005
```

- EXTRA CREDIT – At what value of `mileCut` where the p-value maximized?

Hint: Do a find grid search!

```r
# Define Grid and initialize empty p-value list #
mileCut_grid <- seq(15, 150, by = 0.1)
p_values <- numeric(length(mileCut_grid))

# This loop just uses the code I've already written to generate p-values #
for(i in seq_along(mileCut_grid)) {
  # Temp var
  mileCut_temp <- mileCut_grid[i]

  indC <- ifelse(mileage <= mileCut_temp, 1, 0)
  indL <- ifelse(mileage <= mileCut_temp, 0, 1)

  X <- cbind(indC,
             mileage * indC,
             mileage^2 * indC,
             mileage^3 * indC,
             indL,
             mileage * indL)

  A <- rbind(
    c(1, mileCut_temp,  mileCut_temp^2,   mileCut_temp^3, -1, -mileCut_temp),
    c(0, 1, 2 * mileCut_temp, 3 * mileCut_temp^2, 0, -1),
    c(0, 0, 2, 6 * mileCut_temp, 0, 0)
  )

  XTXinv <- solve(t(X) %*% X)
  beta_unconstrained <- XTXinv %*% t(X) %*% y
  beta_constrained <- beta_unconstrained -
    XTXinv %*% t(A) %*% solve(A %*% XTXinv %*% t(A)) %*% (A %*% beta_unconstrained)

  RSS_unconstrained <- t(y - X %*% beta_unconstrained) %*% (y - X %*% beta_unconstrained)
  RSS_constrained <- t(y - X %*% beta_constrained) %*% (y - X %*% beta_constrained)

  q <- nrow(A)
  n <- length(y)
  p <- ncol(X)
```

```
  F_stat <- ((RSS_constrained - RSS_unconstrained) / q) / (RSS_unconstrained / (n - p))
  p_val <- 1 - pf(F_stat, q, n - p)

  # Store P-value
  p_values[i] <- p_val
}

max_index <- which.max(p_values)
max_mileCut <- mileCut_grid[max_index]
max_p_value <- p_values[max_index]

cat("Maximum p-value:", max_p_value, "at mileCut =", max_mileCut, "\n")
```

```
## Maximum p-value: 0.9562798 at mileCut = 34.5
```

Problem 4

Add the old/new variable to your constrained linear model based on mileage. Test whether this additional variable is significant at the 95 percent level ($\alpha = .05$).

Hint: Don't use both old and new – they will be colinear with the constant. Also reuse your code from problem 3, adding a column to X and a column of zeroes to A. I also changed the names of the matrices and estimates (e.g. `X1`, `A1`, `beta_hat1`) so not to get confused with the ones in problem 3.

```
X1 <- cbind(X, old)   # X with new old indicator var
A1 <- cbind(A, rep(0, nrow(A)))  # New constraint on A

# Unconstrained
X1TX1inv <- solve(t(X1) %*% X1)
beta_hat1 <- X1TX1inv %*% t(X1) %*% y

# Constrained
A1X1TX1invA1T <- A1 %*% X1TX1inv %*% t(A1)
beta_hat1_constrained <- beta_hat1 - X1TX1inv %*% t(A1) %*% solve(A1X1TX1invA1T) %*% (A1 %*% beta_hat1)

# RSS for unconstrained and constrained
RSS1_unconstrained <- t(y - X1 %*% beta_hat1) %*% (y - X1 %*% beta_hat1)
RSS1_constrained  <- t(y - X1 %*% beta_hat1_constrained) %*% (y - X1 %*% beta_hat1_constrained)

q <- 1  # Only adding "old" variable
p1 <- ncol(X1) # # of params

# F-stat
F_stat <- ((RSS1_constrained - RSS1_unconstrained) / q) /
          (RSS1_unconstrained / (n - p1))

# p-val
p_value_F <- 1 - pf(F_stat, q, n - p1)

# Variable is significant
cat("F statistic:", F_stat, "\n")
```

```
## F statistic: 34.55969
```

```r
cat("p-value for the 'old' variable:", p_value_F, "\n")
```

```
## p-value for the 'old' variable: 4.344041e-08
```