Assignment 1

David Fleischer – 260396047

Last Update: 16 January, 2018

Question 1

We wish to show that $\hat{\beta} = \left(\hat{\beta}_1,\, \hat{\beta}_{-1}^T\right)^T$ given by

$$
\begin{aligned}
\hat{\beta}_{-1} &= \underset{\beta \in \mathbb{R}^{p-1}}{\arg\min}\, ||\tilde{Y} - \tilde{X}\beta||_2^2 \\
\hat{\beta}_1 &= \bar{Y} - \bar{x}^T \hat{\beta}_{-1}
\end{aligned}
$$

is a global minimizer of the least squares problem

$$
\hat{\beta} = \underset{\beta \in \mathbb{R}^{p}}{\arg\min}\, ||Y - X\beta||_2^2.
$$

Solution 1

Recall our definitions of $\tilde{X}$ and $\tilde{Y}$

$$
\begin{aligned}
\tilde{X} &= X_{-1} - \mathbf{1}_n \bar{x}^T \\
\tilde{Y} &= Y - \mathbf{1}_n^T \bar{Y}
\end{aligned}
$$

Then

$$
\begin{aligned}
\hat{\beta}_{-1} &= \underset{\beta \in \mathbb{R}^{p-1}}{\arg\min}\, ||\tilde{Y} - \tilde{X}\beta||_2^2 \\
&= \underset{\beta \in \mathbb{R}^{p-1}}{\arg\min}\, ||Y - \mathbf{1}_n\bar{Y} - \left(X_{-1} - \mathbf{1}_n\bar{x}^T\right)\beta_{-1}||_2^2 \\
&= \underset{\beta \in \mathbb{R}^{p-1}}{\arg\min}\, ||Y - X_{-1}\beta_{-1} - \mathbf{1}_n\left(\bar{Y} - \bar{x}^T\beta_{-1}\right)||_2^2 \\
&= \underset{\beta \in \mathbb{R}^{p-1}}{\arg\min}\, ||Y - X_{-1}\beta_{-1} - \mathbf{1}_n\beta_1||_2^2 \quad \text{(by definition of } \beta_1 \text{ above)} \\
&= \underset{\beta \in \mathbb{R}^{p-1}}{\arg\min}\, ||Y - [\mathbf{1}_n,\, X_{-1}]\, [\beta_1,\, \beta_{-1}]||_2^2 \\
&\equiv \underset{\beta \in \mathbb{R}^{p-1}}{\arg\min}\, ||Y - X\beta||_2^2
\end{aligned}
$$

Therefore, if $\hat{\beta} = \left(\hat{\beta}_1,\, \hat{\beta}_{-1}^T\right)^T \in \mathbb{R}^p$ and

$$
\hat{\beta}_1 = \bar{Y} - \bar{x}^T \hat{\beta}_{-1}
$$

then $\hat{\beta}$ also solves the uncentered problem

$$\hat{\beta} = \left(\hat{\beta}_1,\, \hat{\beta}_{-1}^T\right)^T = \arg\min_{\beta \in \mathbb{R}^p} \|Y - X\beta\|_2^2$$

as desired.

Question 2

Consider the (centered) ridge regression problem of estimating β_* with the ℓ_2 penalized least squares regression coefficients $\hat{\beta}^{(\lambda)} = \left(\hat{\beta}_1^{(\lambda)},\, \hat{\beta}_{-1}^{(\lambda)\,T}\right)^T$ defined by

$$\begin{aligned}
\hat{\beta}_{-1}^{(\lambda)} &= \arg\min_{\beta \in \mathbb{R}^{p-1}} \|\tilde{Y} - \tilde{X}\beta\|_2^2 + \lambda\|\beta\|_2^2 \\
\hat{\beta}_1^{(\lambda)} &= \bar{Y} - \bar{x}^T \hat{\beta}_{-1}^{(\lambda)}
\end{aligned}$$

(a)

We define our objective function $f : \mathbb{R}^p \to \mathbb{R}$ by

$$\begin{aligned}
f(\beta) &= \|\tilde{Y} - \tilde{X}\beta\|_2^2 + \lambda\|\beta\|_2^2 \\
&= \left(\tilde{Y} - \tilde{X}\beta\right)^T \left(\tilde{Y} - \tilde{X}\beta\right)^T + \lambda\beta^T\beta \\
&= \tilde{Y}^T\tilde{Y} - \tilde{Y}^T\tilde{X}\beta - \beta^T\tilde{X}^T\tilde{Y} + \beta^T\tilde{X}^T\tilde{X}\beta + \lambda\beta^T\beta \\
&\equiv \tilde{Y}^T\tilde{Y} - 2\beta^T\tilde{X}^T\tilde{Y} + \beta^T\tilde{X}^T\tilde{X}\beta + \lambda\beta^T\beta
\end{aligned}$$

Therefore, taking the gradient of our function $\nabla f(\beta)$ we find

$$\nabla f(\beta) = -2\tilde{X}^T\tilde{Y} + 2\tilde{X}^T\tilde{X}\beta + 2\lambda\beta$$

as desired.

(b)

The second order gradient $\nabla^2 f(\beta)$ yields

$$\nabla^2 f(\beta) = 2\tilde{X}^T\tilde{X} + 2\lambda\mathbb{I}_{p-1}$$

where $\mathbb{I}_{p-1}$ is the $(p-1) \times (p-1)$ identity matrix. Note that $2\tilde{X}^T\tilde{X} \in \mathbb{S}_+^{p-1}$ is positive semi-definite and, with $\lambda > 0$, $2\lambda\mathbb{I}_{p-1} \in \mathbb{S}_+^{p-1}$, i.e. $2\lambda\mathbb{I}_{p-1}$ is also positive semi-definite. Therefore, since a sum of positive semi-definite matrices is also positive semi-definite, we find

$$\nabla^2 f(\beta) = 2\tilde{X}^T\tilde{X} + 2\lambda\mathbb{I}_{p-1} \in \mathbb{S}_+^{p-1}$$

and so f must be strictly convex in β.

(c)

Strict convexity implies that the global minimizer must be unique, and so for $\lambda > 0$ we are guaranteed that the above solution will be the unique solution to our penalized least squares problem.

(d)

To write our function solving for the ridge coefficients we first note that setting $\nabla f(\beta) = 0$ yields

$$\hat{\beta}_{-1}^{(\lambda)} = \left(\tilde{X}^T\tilde{X} + \lambda\mathbb{I}_{p-1}\right)^{-1}\tilde{X}^T\tilde{Y}$$

where $\left(\tilde{X}^T\tilde{X} + \lambda\mathbb{I}_{p-1}\right)$ is guaranteed to be nonsingular (for $\lambda \neq 0$) because it will have have full rank via the identity matrix. For the purpose of computational efficiency we make use of the singular value decomposition on $\tilde{X}$

$$\tilde{X} = UDV^T$$

for $U \in \mathbb{R}^{n\times n}$ and $V \in \mathbb{R}^{(p-1)\times(p-1)}$ both orthogonal matrices, $U^TU = \mathbb{I}_n$, $V^TV = \mathbb{I}_{p-1}$, and $D \in \mathbb{R}^{n\times(p-1)}$ a diagonal matrix with entries $\{d_j\}_{j=1}^{\min(n,\,p-1)}$ along the main diagonal. Then

$$\begin{aligned}
\hat{\beta}_{-1}^{(\lambda)} &= \left(\tilde{X}^T\tilde{X} + \lambda\mathbb{I}_{p-1}\right)^{-1}\tilde{X}^T\tilde{Y} \\
&= \left(\left(UDV^T\right)^T UDV^T + \lambda VV^T\right)^{-1}\left(UDV^T\right)^T\tilde{Y} \\
&= \left(VD^TU^TUDV^T + \lambda VV^T\right)^{-1}VD^TU^T\tilde{Y} \\
&= \left(V\left(D^TD + \lambda\mathbb{I}_{p-1}\right)V^T\right)^{-1}VD^TU^T\tilde{Y} \\
&= V\left(D^TD + \lambda\mathbb{I}_{p-1}\right)^{-1}V^TVD^TU^T\tilde{Y} \\
&= V\left(D^TD + \lambda\mathbb{I}_{p-1}\right)^{-1}D^TU^T\tilde{Y}
\end{aligned}$$

Note that $D^TD + \lambda\mathbb{I}_{p-1}$ is a diagonal $(p-1)\times(p-1)$ matrix with entries $\left\{d_j^2 + \lambda\right\}_{j=1}^{p-1}$ along the main diagonal, and so the inverse $\left(D^TD + \lambda\mathbb{I}_{p-1}\right)^{-1}$ will also be diagonal with entries $\left\{\frac{1}{d_j^2+\lambda}\right\}_{j=1}^{p-1}$. We exploit this to avoid performing a matrix inversion in our code. To this end, see the function below.

```
ridge_coef <- function(X, y, lam) {
  ytilde <- y - mean(y)
  xbar <- colMeans(X)
  Xtilde <- sweep(X, 2, xbar)

  Xtilde_svd <- svd(Xtilde)
  U <- Xtilde_svd$u
  d <- Xtilde_svd$d
  V <- Xtilde_svd$v

  Dstar <- diag(d/(d^2 + lam))

  b1 <- mean(y) - crossprod(xbar, b)
  b <- V %*% (Dstar %*% crossprod(U, ytilde))
  return (list(b1 = b1, b = b))
}
```

Note the choice to use `V %*% (Dstar %*% crossprod(U, ytilde))` to compute the matrix product $VD^*U^T\tilde{Y}$ as opposed to the (perhaps more intuitive) `V %*% Dstar %*% t(U) %*% ytilde`. Such a choice can be justified via the following matrix multiplication benchmarks (for the cases of $n >> p$ and $p >> n$)

```
library(microbenchmark)

#===== Large n case =====#
set.seed(124)

# set parameters
n <- 1e3
p <- 1e2
lam <- 1

# generate data
X <- matrix(rnorm(n * p), nrow = n, ncol = p)
beta <- rnorm(p)
eps <- rnorm(n)
y <- X %*% beta + eps

ytilde <- y - mean(y)
xbar <- colMeans(X)
Xtilde <- sweep(X, 2, xbar)

# compute decomposition
Xtilde_svd <- svd(Xtilde)
U <- Xtilde_svd$u
d <- Xtilde_svd$d
V <- Xtilde_svd$v
Dstar <- diag(d/(d^2 + lam))

# define multiplication functions
f1 <- function() V %*% Dstar %*% t(U) %*% ytilde
f2 <- function() V %*% Dstar %*% (t(U) %*% ytilde)
f3 <- function() V %*% (Dstar %*% (t(U) %*% ytilde))
f4 <- function() V %*% (Dstar %*% crossprod(U, ytilde))
f5 <- function() V %*% crossprod(Dstar, crossprod(U, ytilde))

# test speed
microbenchmark(f1(), f2(), f3(), f4(), f5(), times = 100, unit = "us")
```

```
## Unit: microseconds
##   expr      min         lq       mean     median         uq       max neval
##   f1() 8675.897 10418.0540 11594.1290 11211.8595 11789.5385 47415.609   100
##   f2() 1096.256  1311.2580  2421.7085  1542.1120  2214.6150 35378.696   100
##   f3()  366.366   507.5965   741.4603   583.9960   831.1000  1701.947   100
##   f4()  131.109   147.8810   193.8988   160.7280   198.6690   993.283   100
##   f5()  130.856   145.4300   181.5766   155.7845   179.7705   696.934   100
```

```
#===== Large p case =====#
set.seed(124)

# set parameters
n <- 1e2
p <- 1e3
```

```r
lam <- 1

# generate data
X <- matrix(rnorm(n * p), nrow = n, ncol = p)
beta <- rnorm(p)
eps <- rnorm(n)
y <- X %*% beta + eps

# define multiplication functions
f1 <- function() V %*% Dstar %*% t(U) %*% ytilde
f2 <- function() V %*% Dstar %*% (t(U) %*% ytilde)
f3 <- function() V %*% (Dstar %*% (t(U) %*% ytilde))
f4 <- function() V %*% (Dstar %*% crossprod(U, ytilde))
f5 <- function() V %*% crossprod(Dstar, crossprod(U, ytilde))

# test speed
microbenchmark(f1(), f2(), f3(), f4(), f5(), times = 100, unit = "us")
```

```
## Unit: microseconds
##   expr      min         lq       mean     median         uq       max neval
##   f1() 9267.035 10715.2085 14139.2541 11679.2180 13612.0930 61102.292   100
##   f2() 1101.875  1415.5660  2835.8924  1950.0455  2735.9460 39304.956   100
##   f3()  374.673   509.1730   821.4771   573.5920   716.6405  6809.041   100
##   f4()  129.743   154.8745   192.6407   167.3170   205.5340   590.024   100
##   f5()  128.371   146.1310   193.4637   159.6365   190.7005   920.452   100
```

(e)

We take the expectation of $\hat{\beta}^{(\lambda)}$

$$\begin{aligned}
\mathbb{E}\left[\hat{\beta}_{-1}^{(\lambda)}\right] &= \mathbb{E}\left[\left(\tilde{X}^T\tilde{X} + \lambda\mathbb{I}_{p-1}\right)^{-1}\tilde{X}^T\tilde{Y}\right] \\
&= \mathbb{E}\left[\right] \\
&= \mathbb{E}\left[\right]
\end{aligned}$$

Question 3

Question 4

Question 5

Question 6