MA 575 HW 9

Benjamin Draves

November 21

Exercise 9.1

(a)

First, to find the eigenvalues of X^TX we solve

$$\begin{aligned} X^TXv &= \gamma v \\ (X^TX - \gamma I)v &= 0 \end{aligned}$$

Here γ is the eigenvalue and v is the eigenvector corresponding to X^TX. Now, let $Y^TY = X^TX + \lambda I$. Then we can find the eigenvalues of Y^TT by solving the equation

$$\begin{aligned} Y^TYu &= \theta u \\ (Y^TY - \theta I)u &= 0 \end{aligned}$$

Here θ is the eigenvalue and u is the eigenvector corresponding to Y^TY. But notice, by the way we defined Y^TY we can reduce this further by

$$\begin{aligned} (Y^TY - \theta I)u &= 0 \\ (X^TX + \lambda I - \theta I)u &= 0 \\ (X^TX - (\theta - \lambda)I)u &= 0 \end{aligned}$$

Notice that this was just the system we were trying to solve above. That is the eigenvalues and eigenvectors (γ_i, v_i) corresponding to X^TX are in direct correspondence to the $(\theta_i - \lambda, u_i)$ pairs. But since λ is constant in all of these equations, we have $\lambda_i(X^TX) = \lambda_i(Y^TY) - \lambda$. This gives

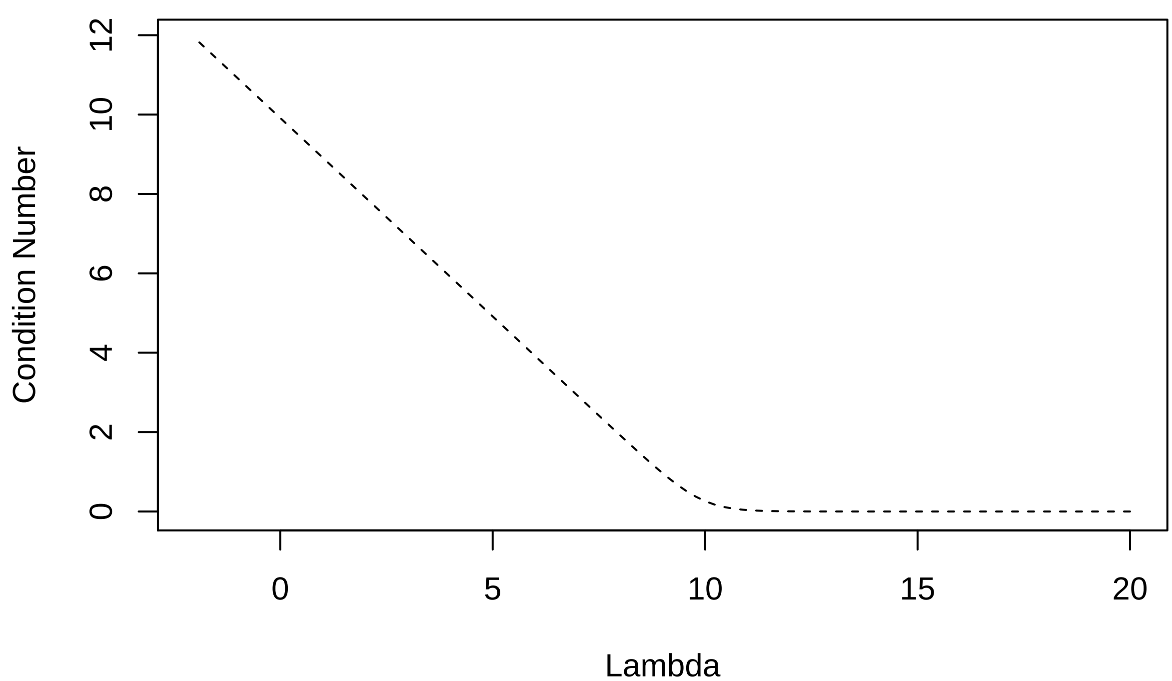
$$\lambda_i(X^TX + \lambda I) = \lambda_i(X^TX) + \lambda$$

Therefore, we can write the condition number as

$$\kappa(X^TX + \lambda I) = \frac{\lambda_m(X^TX + \lambda I)}{\lambda_1(X^TX + \lambda I} = \frac{\lambda_m(X^TX) + \lambda}{\lambda_1(X^TX) + \lambda}$$

(b)

```r
#read in/format data
dat = read.csv("~/Desktop/Courses/MA 575/book_data/reducedbikedata2011.csv")

#form the p + 1 data matrix
X = as.matrix(dat[,-c(1,2)])

#make the design matrix
design = t(X) %*% X

#get eigen values from design
evals = eigen(design)$values
max = max(evals)
min = min(evals)

#define data structures
lambda <- 10^seq(20, -2, length = 100)
Y = length(lambda)

#get condition values
for(i in 1:length(lambda)){
  val = (max + lambda[i])/(min + lambda[i])
  Y[i] = log(val,10)
}

#make plot of condition numbers
plot(log(lambda, 10), Y, type = "l", xlab = "Lambda", ylab = "Condition Number", main = "", lty
```



From a numerical point of view, larger λ will stabilize the inversion of $X^T X$ by adding larger and larger values to the main diagnose via the correction λI. But notice as we let $\lambda \to \infty$, we require that all β values are zero. This is because as we let any $\beta > 0$, then the penalty will inflate the RSS_{RIDGE} to infinity. Therefore, we face a trade off - numerical stability and penalization for problems with several predictors (i.e. $n << p$) and the inferential task of relating the covariates to Y.Therefore, we should choose a λ value that penalizes enough to ensure stability and provide structure to high dimensional problems while still allowing the model to fit the trends in the data.

Exercise 9.2

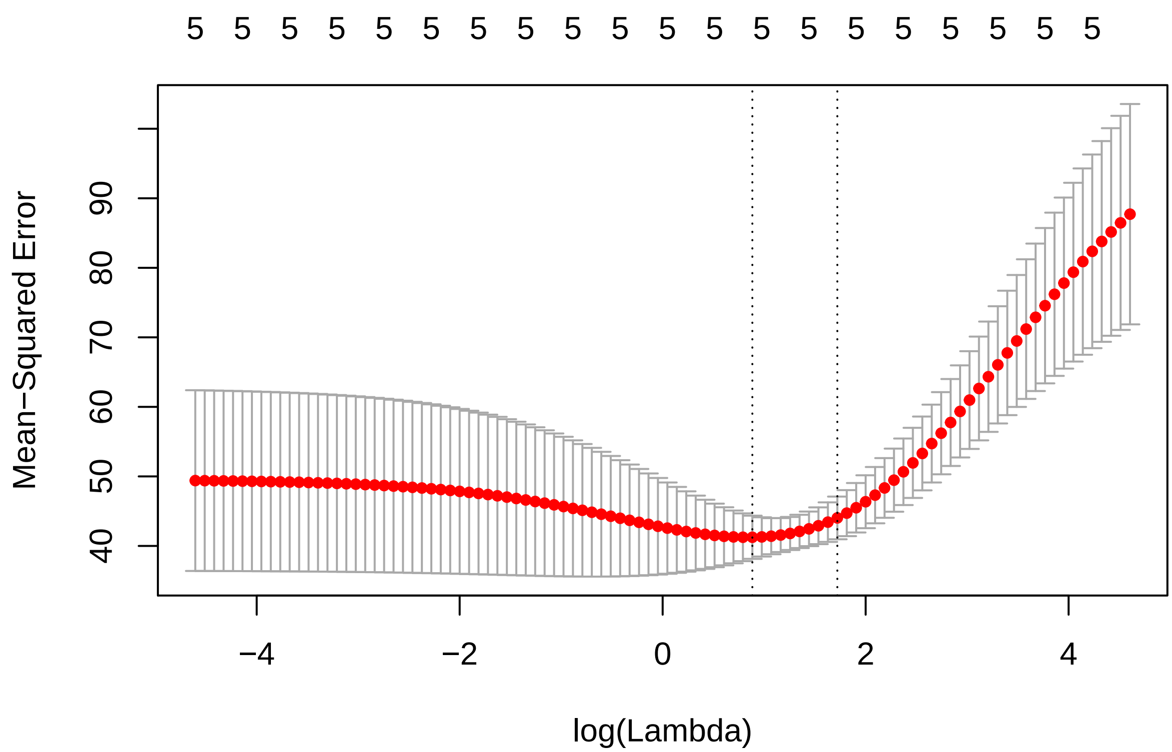
We will fit all three models then comment below.

```
#-------------------------
#
#       Data Prep
#
#-------------------------

#load in data
swiss <- datasets::swiss

#create model matrix/response vector
x = model.matrix(Fertility~., swiss)[,-1]
y = swiss$Fertility

#set up training/testing sets
set.seed(489)
train = sample(1:nrow(x), nrow(x)/2)
test = (1:nrow(x))[-train]

#-------------------------
#
#       OLS
#
#-------------------------

#OLS model/prediction
ols = lm(y~., data = swiss[,-1], subset = train)

#Prediction
pred_ols = as.numeric(predict(ols, newdata = swiss[test,-1]))

#Get MSE
mse_ols = mean((y[test] - pred_ols)^2)

#-------------------------
#
#       Ridge
```

```r
#
#-------------------------

#fit Ridge
library("glmnet")
```

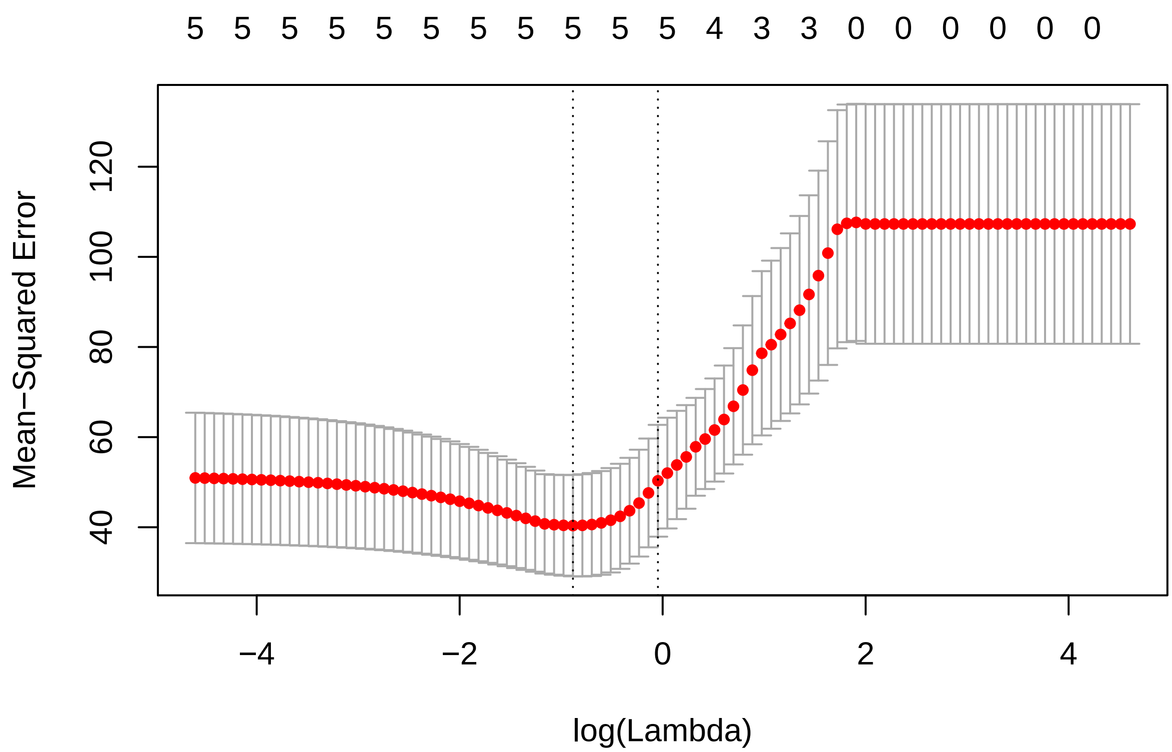
```
## Loading required package: Matrix

## Loading required package: foreach

## Loaded glmnet 2.0-13
```

```r
lam = 10^seq(2, -2, length = 100)
ridge = glmnet(x[train,], y[train], lambda = lam, alpha = 0)

#CV to find lambda_opt
cv = cv.glmnet(x[train,], y[train], lambda = lam, nfolds = 5, alpha = 0)
plot(cv)
```




```r
lam_opt_ridge = cv$lambda.min

#find predictions
pred_ridge = predict(ridge, s= lam_opt_ridge, newx = x[test,])

#MSE calculations
mse_ols = mean((y[test] - pred_ols)^2)
mse_ridge = mean((y[test] - pred_ridge)^2)
```

```r
#-------------------------
#
#       LASSO
#
#-------------------------
lasso = glmnet(x[train,], y[train], lambda = lam, alpha = 1)

#CV to find lambda_opt
cv = cv.glmnet(x[train,], y[train], lambda = lam, nfolds = 5, alpha = 1)
plot(cv)
```




```r
lam_opt_lasso = cv$lambda.min

#find predictions
pred_lasso = predict(lasso, s= lam_opt_lasso, newx = x[test,])

#get MSE
mse_lasso = mean((pred_lasso - y[test])^2)

#-------------------------
#
#       Compare Models
#
#-------------------------

#print all three MSE
```

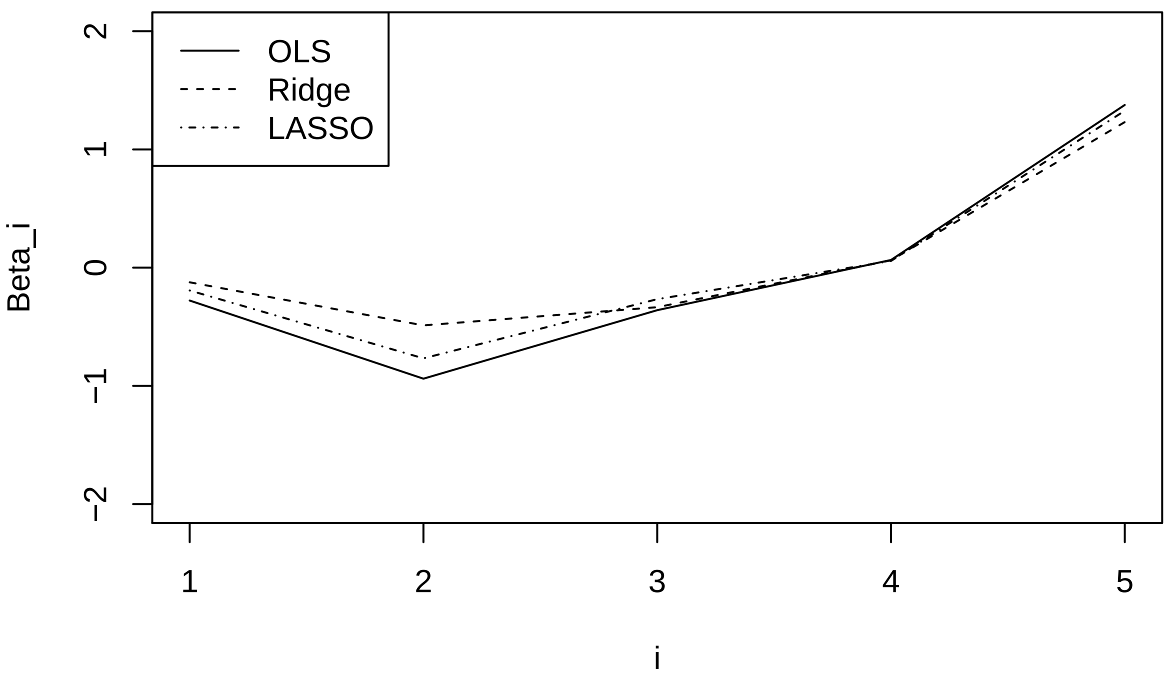
```r
message(paste("MSE OLS:", round(mse_ols,3),"\n MSE Lasso:", round(mse_lasso,3),"\n MSE Ridge:"
```

```
## MSE OLS: 106.009
##  MSE Lasso: 103.738
##  MSE Ridge: 93.136
```

```r
#get best coef
betas = data.frame(B.OLS = coef(ols))
betas$B.Ridge = predict(ridge, type = "coefficients", s = lam_opt_ridge)[1:6,]
betas$B.LASSO = predict(lasso, type = "coefficients", s = lam_opt_lasso)[1:6,]

#print all coefficents
print(betas)
```

```
##                        B.OLS     B.Ridge    B.LASSO
## (Intercept)      74.63669064 62.45392671 68.0347328
## Agriculture      -0.27810752 -0.12381476 -0.1931092
## Examination      -0.93921916 -0.48843634 -0.7672501
## Education        -0.35970838 -0.33455989 -0.2666706
## Catholic          0.06498258  0.06201879  0.0569043
## Infant.Mortality  1.37617843  1.23132670  1.3273270
```

```r
#plot coefficents (except for intercept)
plot(1:5,betas[2:6,1], lty = 1, type = "l", ylim = c(-2,2), ylab = "Beta_i", xlab = "i")
points(1:5,betas[2:6,2],lty = 2,  type = "l")
points(1:5,betas[2:6,3], lty = 4, type = "l")
legend("topleft", legend = c("OLS", "Ridge", "LASSO"), lty = c(1,2,4))
```



Here we see that the MSE decreases as we increase our penalty. That is we have

$$MSE_{OLS} > MSE_{Lasso} > MSE_{Ridge}$$

Ridge performs very well, with sizable MSE reduction. I included a table of the final β estimates from each model. Also included is a plot for the β estimates except for the intercept. We notice as λ increases, the β terms tend to zero. This matches our intuition/interpretation as Lasso and Ridge as shrinkage estimators. For instance for β_2 we see great shrinkage, but for more important terms in the model (i.e. infant mortality) that the coefficients remain the same, even though it is statistically different than zero. This is a case where the penalization schemes improve on the OLS models significantly.