Homework 1

Alex Romriell

Computational Problems

First import all of the libraries and data required for this script

```
# Install the needed packages for this script
# and set the working directory and read in the data.
# setwd("~/Dropbox/School/Fall Module II/Machine Learning/hw1")
library(plyr, quietly = TRUE)
library(ggplot2, quietly = TRUE)
library(reshape2, quietly = TRUE)
library(car, quietly = TRUE)
library(lmtest, quietly = TRUE)
library(MASS, quietly = TRUE)
library(glmnet, quietly = TRUE)
library(ISLR, quietly = TRUE)
college <- read.csv("College.csv")
```

Problem 1

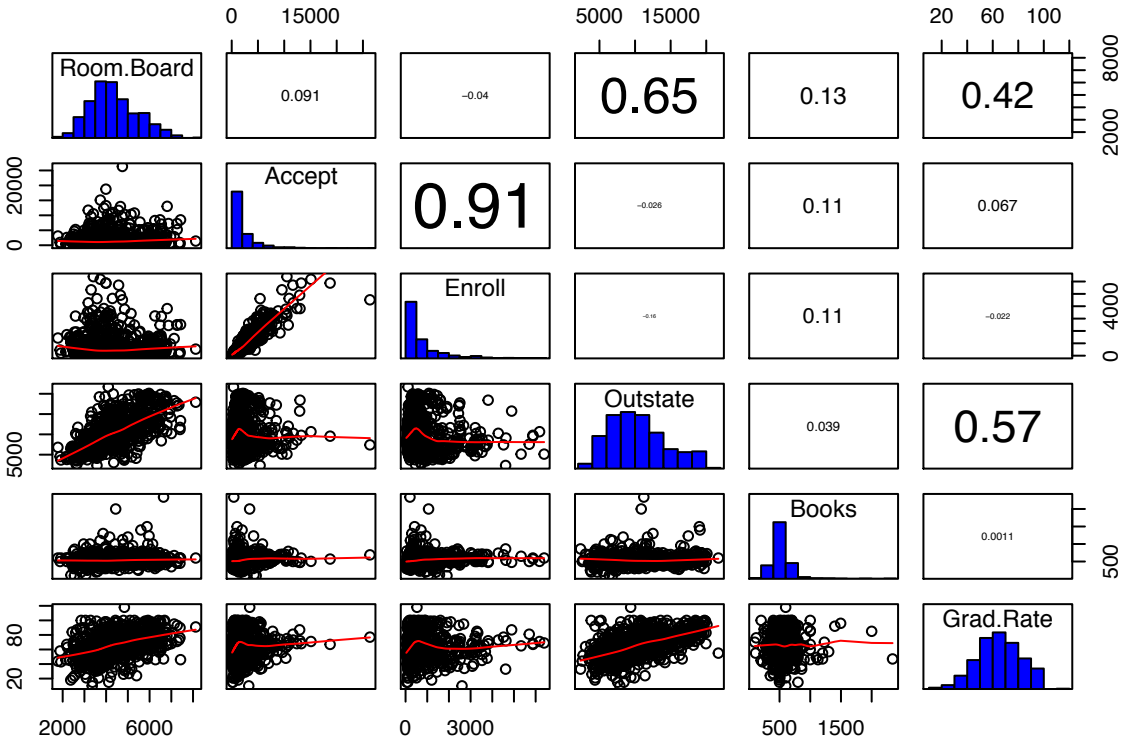
Write a function *kfold.cv.lm()* that outputs average MSE and average MSPE

```
kfold.cv.lm <- function(k, seed, X, y, which.betas) {

  # split the data into k random subsets
  n <- nrow(X)
  set.seed(seed)
  groups <- split(X, sample(1:k, n, replace=T))

  # create model based on 'which.betas' and 'y'
  response <- colnames(X[y])
  predictors <- colnames(X[which.betas])
  model_formula <- formula(paste(response, "~", paste(predictors, collapse=" + ")))

  # use ldply to create dataframe and calculate k MSE's and MSPE's for training/test sets
  MSE_results <- ldply(1:k, function(k){
    test <- groups[k]
    test <- ldply(test, data.frame) # combine list into one dataframe

    train <- groups[-k]
    train <- ldply(train, data.frame)

    fit <- lm(model_formula, data=train)
    MSE <- mean(fit$residuals^2)
    MSPE <- mean((predict(fit, test) - test[response])^2)
    return(c("MSE" = MSE, "MSPE" = MSPE))
  })
```

```
  # calculate the average of MSE/MSPE from k-fold validation.
  avg_MSE <- mean(MSE_results$MSE)
  avg_MSPE <- mean(MSE_results$MSPE)

  return(c("avg_MSE" = avg_MSE, "avg_MSPE" = avg_MSPE))
}
```

Problem 2

part (a) Plot a pairwise scatterplot of the five predictors indicated: Accept, Enroll, Outstate, Books, and Grad.Rate and comment on any trends.



Grad.Rate, and Outstate seem to provide the best trend data for Room.Board. There is a high degree of collinearity between Accept and Enroll. This could be a problem for the model if both are included.

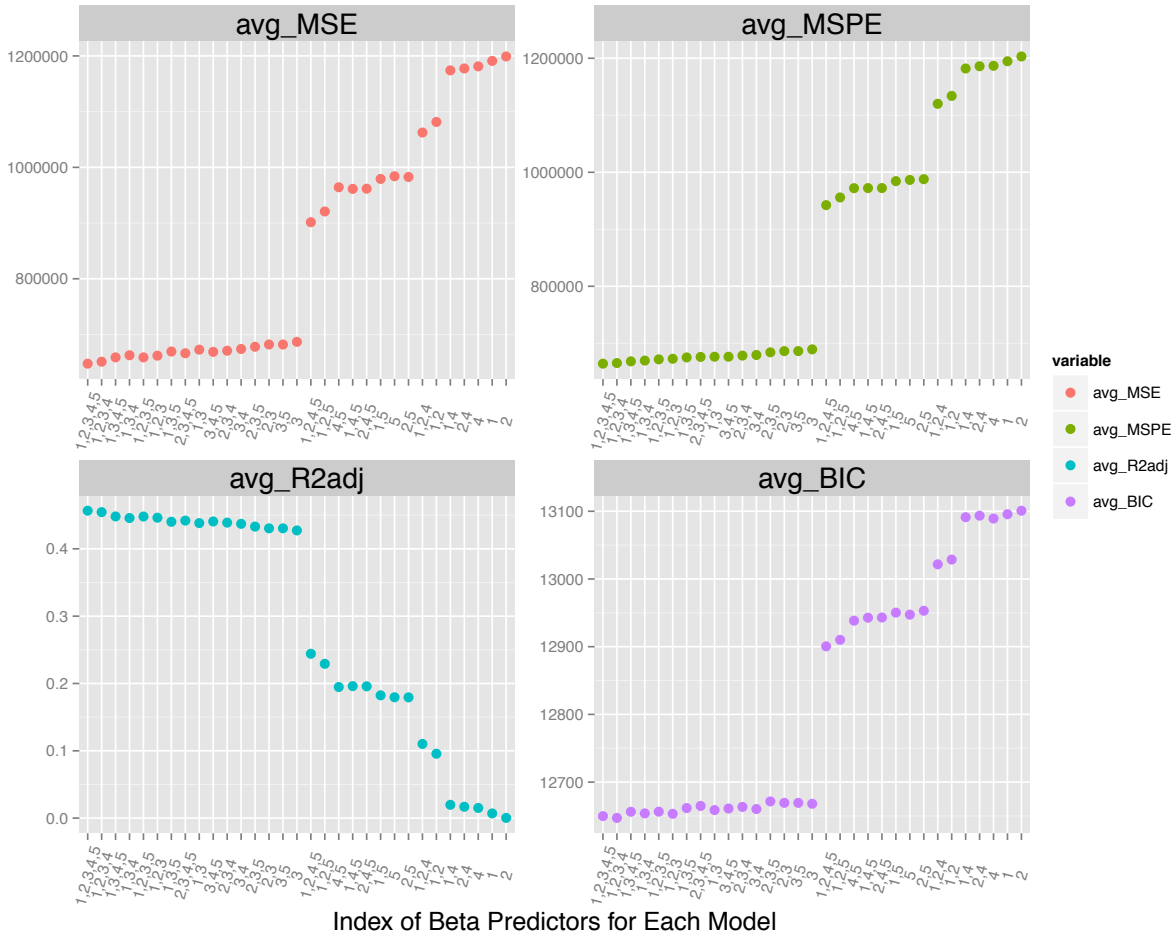
part (b and c)

Use *kfold.cv.lm()* to run 10-fold cross validation on all combinations of the 5 predictors listed in part(a). Run each model 100 times to get a distribution of the average MSPE. Select the best model and generate a histogram from the 100 runs of the 10-fold cross validation of that model.

NOTE: I combined parts b and c. Then plotted the average MSE, MSPE, R^2_{adj} and BIC in order to choose the best model. Once the best model was chosen, a histogram was generated from running that chosen model through 10-fold validation once more. (Compuationally expensive, I know. . .)

```
y <- colnames(college) == "Room.Board"
which.betas <- colnames(college) %in% c("Accept","Enroll","Outstate","Books","Grad.Rate")
predictors <- names(college[which.betas])

all_MSE <- data.frame()
for (i in 1:length(predictors)){
  combos <- combn(predictors, m = i)
```

```
    for (j in 1:length(combos[1,])){

      hundred_reps <- data.frame()
      for (k in 1:100){
        sub_MSE <- kfold.cv.lm(k=10, seed=sample(1:50,1), X=college2,
                               y=colnames(college2)=="Room.Board",
                               which.betas=colnames(college2) %in% combos[,j])
        # get adjR2 and BIC...
        tmp_response <- colnames(college2["Room.Board"])
        tmp_which.betas=colnames(college2) %in% combos[,j]
        tmp_predictors <- colnames(college2[tmp_which.betas])
        model_formula <- formula(paste(tmp_response, "~", paste(tmp_predictors, collapse=" + ")))
        lm1 <- lm(model_formula, data=college2)
        tmp_adjR2 <- summary(lm1)$adj.r.squared
        tmp_bic <- BIC(lm1)

        #get beta index and compile dataframe of MSPE, MSE, R2adj, BIC
        beta_index <- which(predictors %in% combos[,j])
        # convert to a single value for naming in df
        beta_index <- as.character(paste(beta_index, collapse = ","))
        tmp_df <- data.frame(beta_index = beta_index, avg_MSE = sub_MSE[1],
                             avg_MSPE=sub_MSE[2], adjR2 = tmp_adjR2,
                             BIC = tmp_bic, row.names = NULL)
        #compute averages
        hundred_reps <- rbind(tmp_df, hundred_reps)
        avg_df <- data.frame(beta_index = beta_index, avg_MSE = mean(hundred_reps$avg_MSE),
                             avg_MSPE = mean(hundred_reps$avg_MSPE),
                             avg_R2adj = mean(hundred_reps$adjR2),
                             avg_BIC = mean(hundred_reps$BIC))
      }
      all_MSE <- rbind(avg_df, all_MSE)
    }
}

# order the data by lowest MSPE for pretty plotting, then melt the data for ggplot2
all_MSE_order <- transform(all_MSE, beta_index=reorder(beta_index, avg_MSPE))
all_MSE_order <- melt(all_MSE_order)

p <- ggplot(data=all_MSE_order, aes(x=beta_index, y=value, group=variable, color=variable))
p <- p + geom_point(size=3) + xlab("Index of Beta Predictors for Each Model") + ylab("") +
     theme(axis.text.x=element_text(angle=75, size=5, vjust=0.5))
p <- p + facet_wrap(~ variable, scales="free")
p <- p + theme(axis.text.x=element_text(size=14),
               axis.text.y=element_text(size=14),
               axis.title=element_text(size=14),
               strip.text = element_text(size=18))
#p --- formatting looked weird in R-Markdown. Inserting .pdf file instead.
```



This plot shows the average MSE, MSPE, R^2_{adj}, and BIC for all combinations of the 5 chosen predictor variables. Based on the lowest MSPE, the model with all 5 predictors preforms best. This makes sense since MSPE will be lowest when the model contains all variables.

This is also true for R^2_{adj} - the model with all 5 predictors gives the highest R^2_{adj} value. Of note, however, is that using BIC as the model selection criterion only 4 predictors are suggested for best fit. "Grad.Rate" is dropped and only Accept, Enroll, Outstate, and Books are used. The best model, based on MSPE is:

Best Model by MSPE:

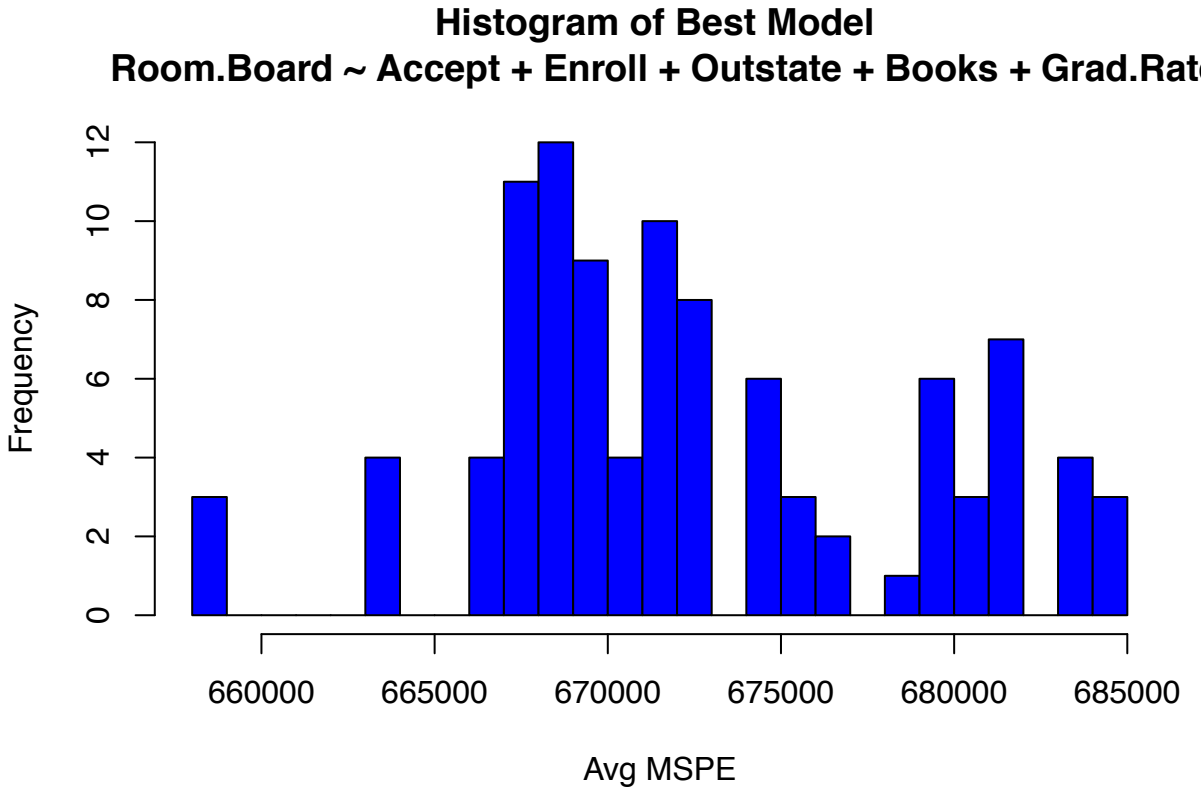
$$Y = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + \beta_3 X_3 + \beta_4 X_4 + \beta_5 X_5$$

where:

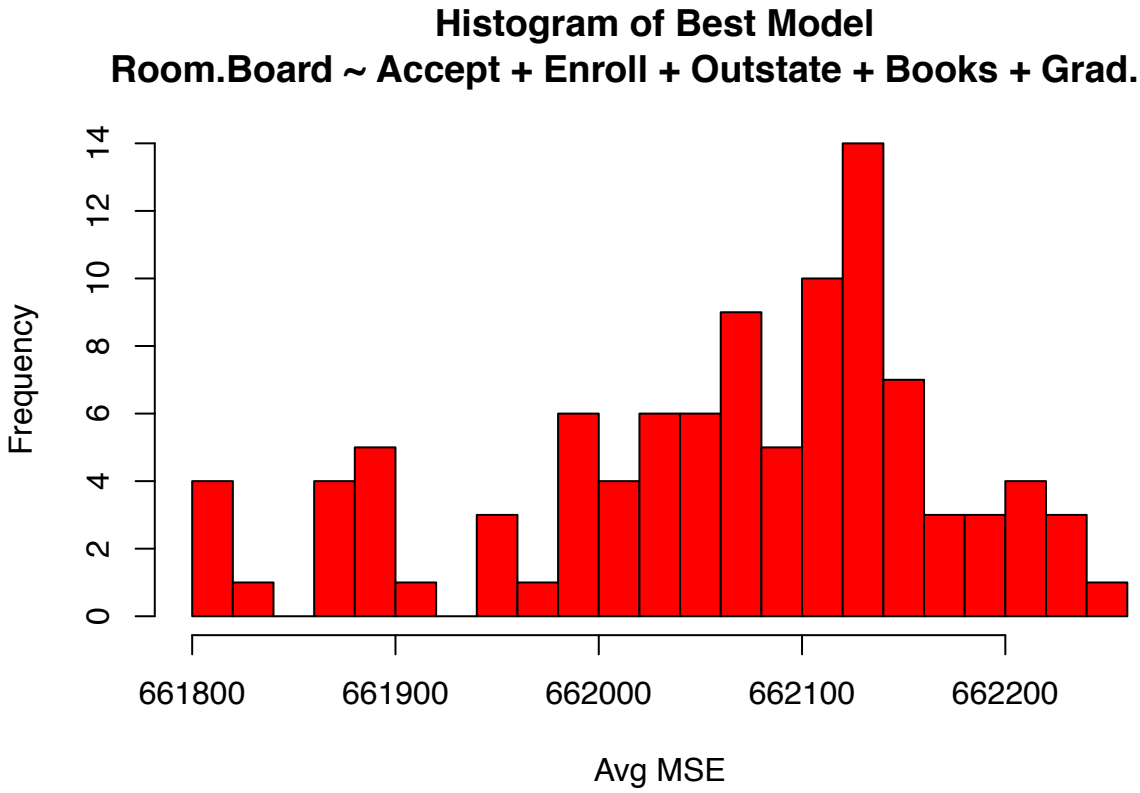
- β_1 = "Accept"
- β_2 = "Enroll"
- β_3 = "Outstate"
- β_4 = "Books"
- β_5 = "Grad.Rate

Using the Best Model, k-fold validation was performed 100 times to get an idea of the distribution of the MSPE for this model.

```r
# from plot, best betas = 1,2,3,4,5 == Accept, Enroll, Outstate, Books, Grad.Rate
which.betas <- colnames(college) %in% c("Accept","Enroll","Outstate","Books","Grad.Rate")
predictors <- names(college[which.betas])
response <- colnames(college2["Room.Board"])
best_model_form <- formula(paste(response, "~", paste(predictors, collapse=" + ")))
which.beta_best <- colnames(college) %in% c("Accept","Enroll","Outstate","Books, Grad.Rate")

# run k-fold 100 times to get distribution of MSPE and MSE for Best Model
best_model <- data.frame()
for (i in 1:100){
  tmp_MSE <- kfold.cv.lm(k=10, seed=sample(1:1000,1), X=college,
              y=colnames(college)=="Room.Board",
              which.betas=which.beta_best)
  tmp_df <- data.frame(avg_MSE = tmp_MSE[1], avg_MSPE = tmp_MSE[2], row.names=NULL)
  best_model <- rbind(tmp_df, best_model)
}

hist(best_model$avg_MSPE, n=20,
     main="Histogram of Best Model \n Room.Board ~ Accept + Enroll + Outstate + Books + Grad.Rate",
     xlab = "Avg MSPE", col='blue')
```




```r
hist(best_model$avg_MSE, n=20,
     main="Histogram of Best Model \n Room.Board ~ Accept + Enroll + Outstate + Books + Grad.Rate",
     xlab = "Avg MSE", col='red')
```



part (d)

Do the normal assumptions hold for the chosen model?

No they do not. While there are 4 assumptions under OLS. At least two of the four assumprtion do not hold. For the Best Model defined above:

1. The residuals are not normally distributed.
   - The Shapiro-Wilk test formally tests for normally distributed data.

$$H_0 : \epsilon \sim N(0, \sigma^2)$$
$$H_a : \epsilon \not\sim N(0, \sigma^2)$$

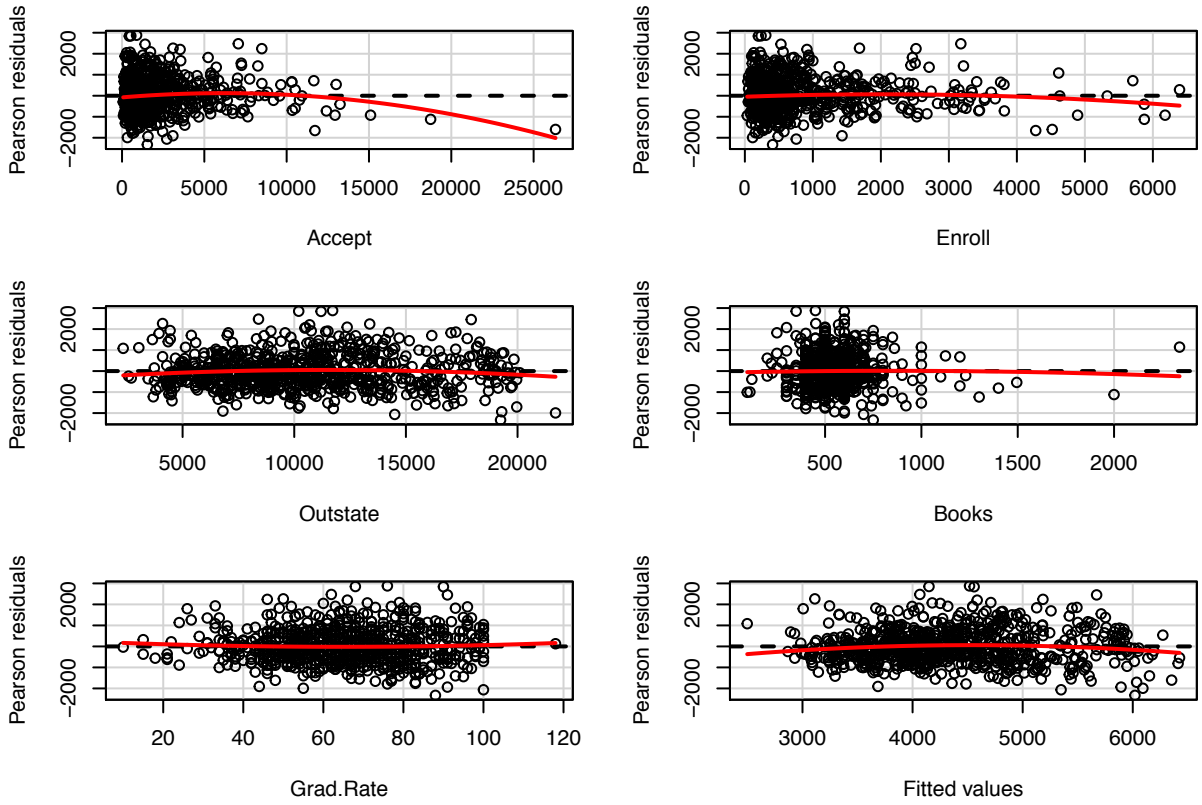
   - This test resulted in a p-value = 1.896e-07, which is $<< \alpha = 0.05$, which means we reject the null hypothesis that the residuals are normally distributed.
2. The residuals are not homoscedastic.
   - The Breush-Pagan test can test the hypothesis whether the residuals have constant variance.

$$H_0 : \gamma = 0$$
$$H_a : \gamma \neq 0$$
$$\text{where } \gamma \text{ is related to the variance of the data by } log(\sigma^2) = \gamma_0 + \gamma_1 X$$

   - This test resulted in a p-value of 0.0046 which is also $< \alpha = 0.05$ and we therefore reject the null hypothesis that the residuals are homoscedastic.
3. The residual plots and qqPlot help visualize the non-normality of the data, as well as the non-constant variance.

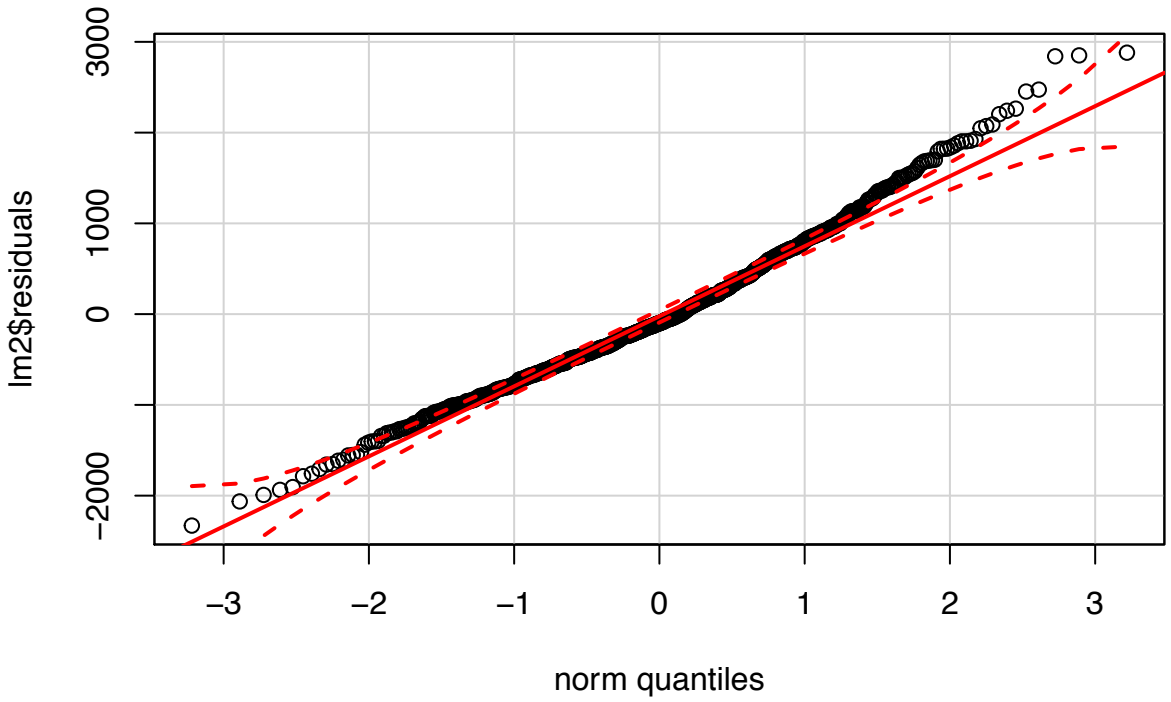
```
lm2 <- lm(best_model_form, data=college)
summary(lm2)
shapiro.test(lm2$residuals) # p-value = 3.707e-07, not normally distributed residuals
bptest(lm2) # p-value = 0.004959, heteroscedastic at alpha = 0.05
```

```
lm2 <- lm(best_model_form, data=college)
residualPlots(lm2)
```




```
##             Test stat Pr(>|t|)
## Accept         -3.772    0.000
## Enroll         -1.601    0.110
## Outstate       -2.032    0.042
## Books          -0.461    0.645
## Grad.Rate       0.859    0.391
## Tukey test     -2.481    0.013
```

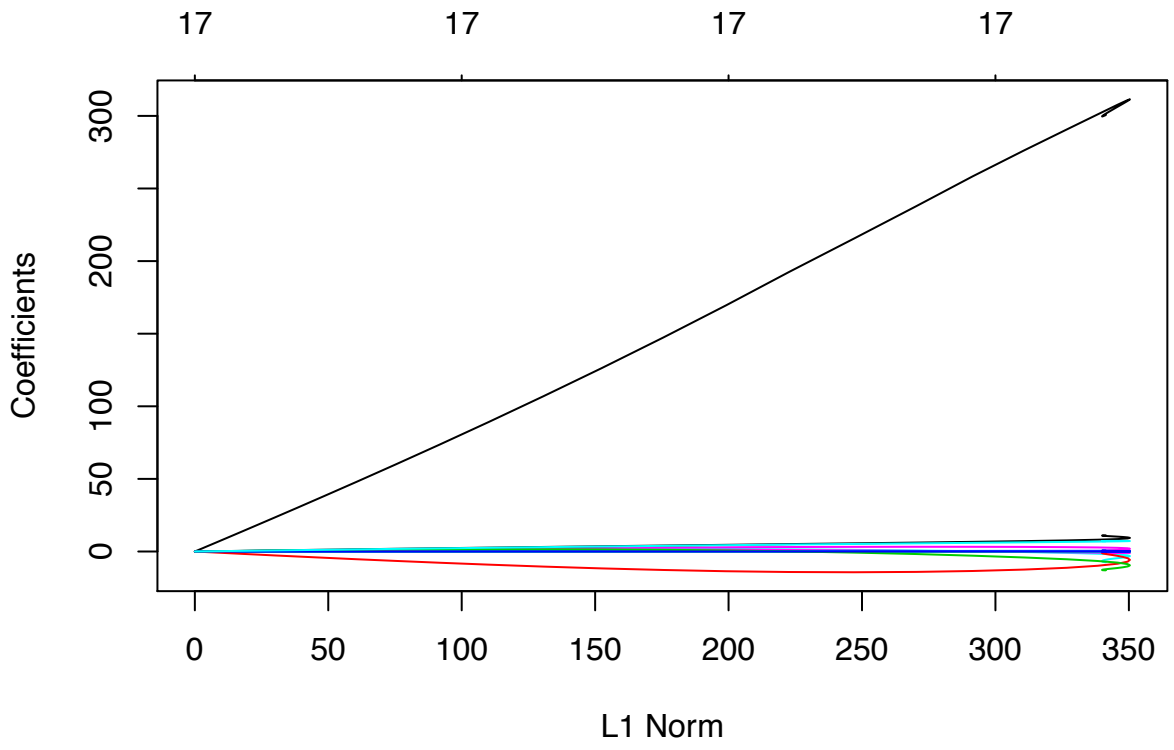
```
qqPlot(lm2$residuals)
```



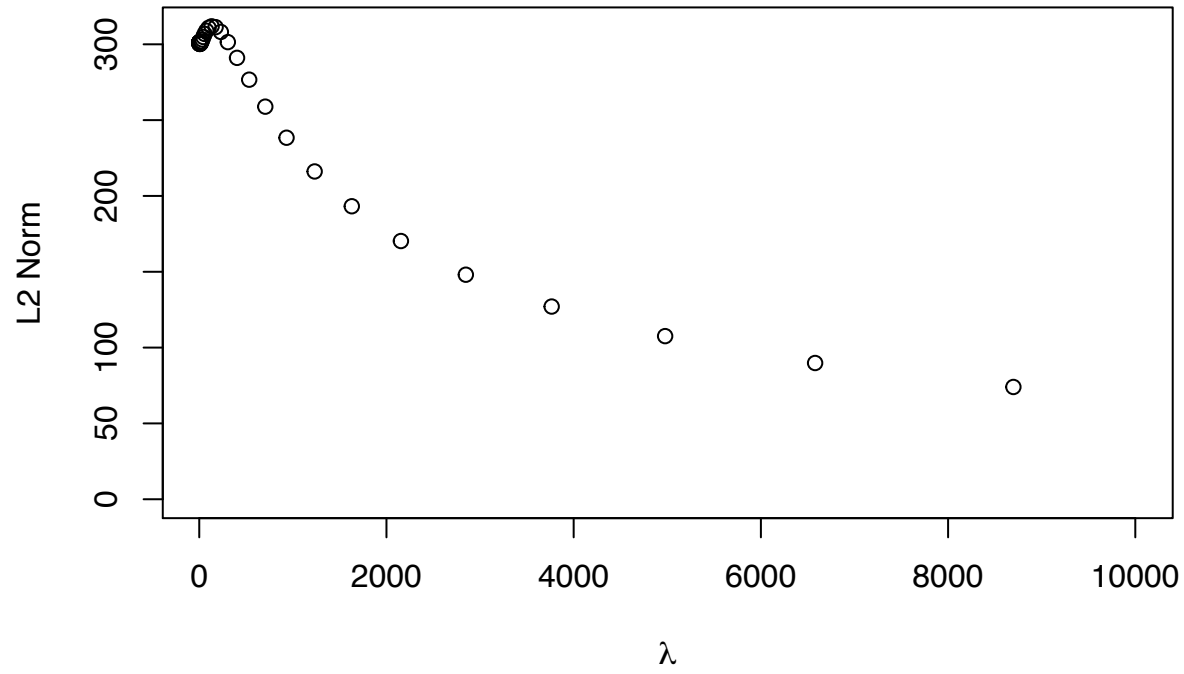
part (e, f, g)

Run Ridge, Lasso, and Elastic Net Regressions on all 17 predictors in the College data set. Comment on similarities and differences of the final model selected by each with its associated MSPE.

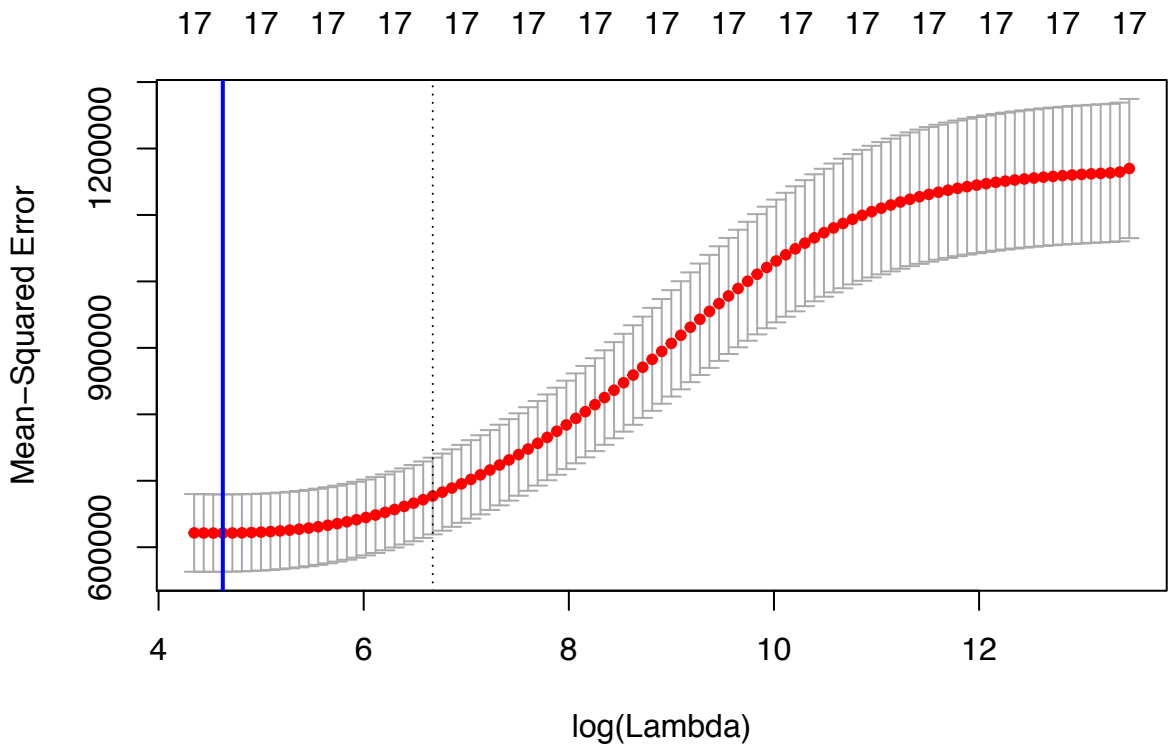
```
# First, set a grid of lambda to search over. We want to include lambda = 0
# for standard linear regression
grid.lambda <- 10^seq(10, -2, length = 100)
college3 <- college[,-1] # removing college name from data
x <- model.matrix(Room.Board ~., data=college3)[,-1]
y <- college3$Room.Board

#Fit the model across the grid of lambda values
ridge.model <- glmnet(x, y, alpha = 0, lambda = grid.lambda)

#Plot the L1 norm of the coefficents
plot(ridge.model)
```




```
#Let's repeat this for all lambda values and plot the results
ell2.norm <- numeric()
for(i in 1:length(grid.lambda)){
  ell2.norm[i] <- sqrt(sum(coef(ridge.model)[-1, i]^2))
}

plot(x = grid.lambda, y = ell2.norm, xlab = expression(lambda),
        ylab = "L2 Norm",xlim = c(10,10000))
```



```r
set.seed(1) #for reproducability
#Randomly select a training and test set.
#Here, we leave half of the data out for later model assessment
train <- sample(1:nrow(x), nrow(x) / 2)
test <- (-train)
y.train <- y[train]
y.test <- y[test]

#Now, fit a Ridge regression model to the training data
ridge.model.train <- glmnet(x[train, ], y.train, alpha = 0, lambda = grid.lambda)

#Perform cross validation on the training set to select the best lambda
set.seed(1) #for reproducability
cv.out <- cv.glmnet(x[train, ], y.train, alpha = 0)

#Find the best lambda value
best.lambda <- cv.out$lambda.min

plot(cv.out)
abline(v = log(best.lambda), col = "blue", lwd = 2)
```

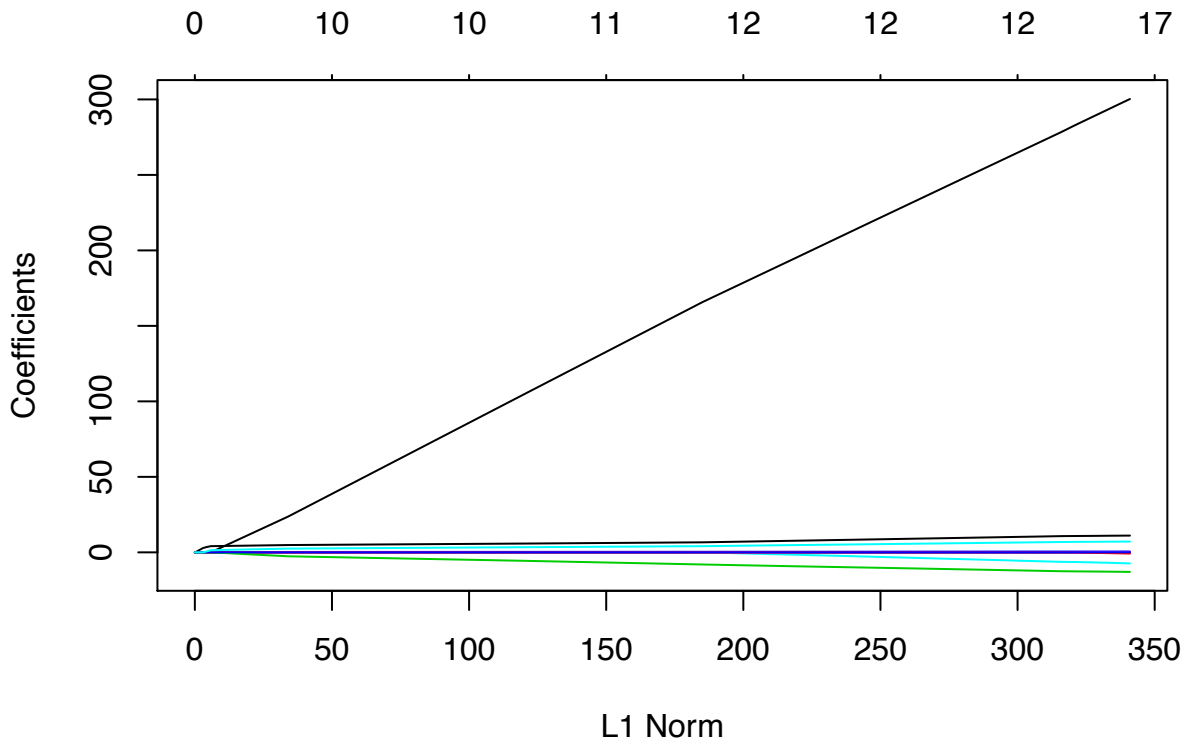



```r
best.lambda
```

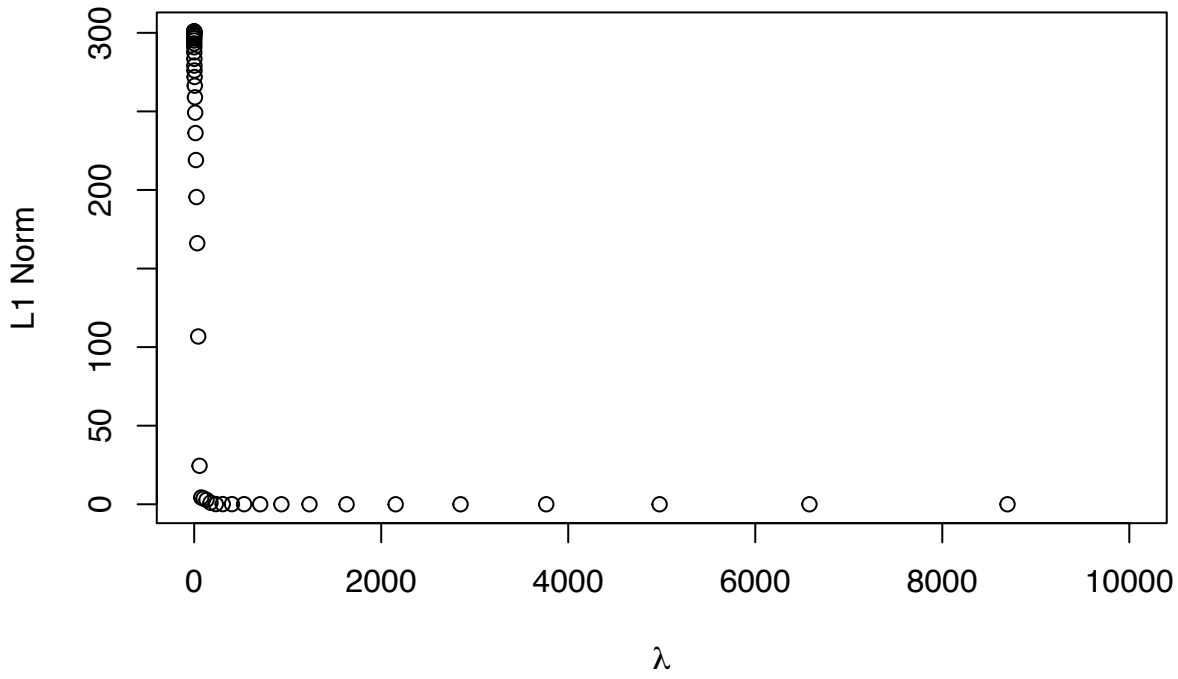
```
## [1] 102.1428
```

```r
#Calculate the MSPE of the model on the test set
ridge.pred <- predict(ridge.model.train, s = best.lambda, newx = x[test, ])
mspe.ridge <- mean((ridge.pred - y.test)^2)
```

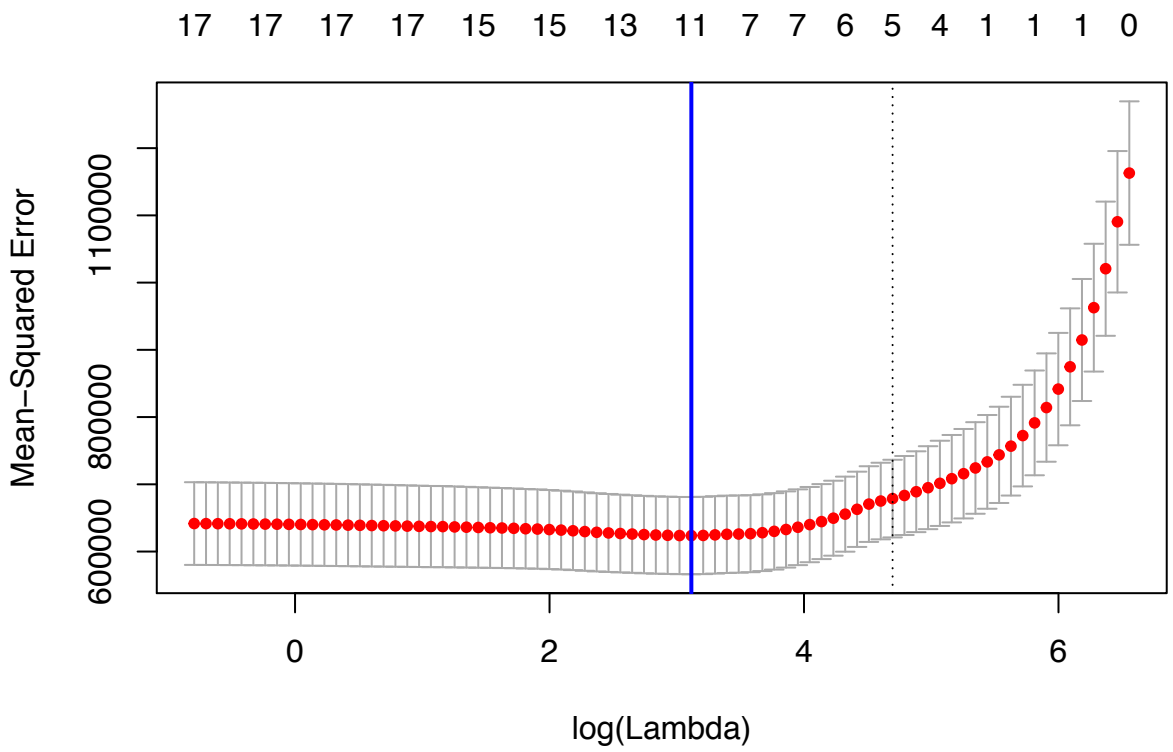
```r
#Fit the final model to the entire data set using the chosen lambda
final.model <- glmnet(x, y, alpha = 0, lambda = best.lambda)
Coef.Ridge <- coef(final.model)[1:17, ]

############################
###### PROBLEM 2f ##########
############################

# Lasso Regression

#First, let's look at the shrinkage effects of Lasso on the entire data set
lasso.model <- glmnet(x, y, alpha = 1, lambda = grid.lambda)
plot(lasso.model)
```




```r
#Let's repeat this for all lambda values and plot the results
ell1.norm <- numeric()
for(i in 1:length(grid.lambda)){
  ell1.norm[i] <- sqrt(sum(coef(lasso.model)[-1, i]^2))
}

plot(x = grid.lambda, y = ell1.norm, xlab = expression(lambda),
     ylab = "L1 Norm", xlim = c(0,10000))
```



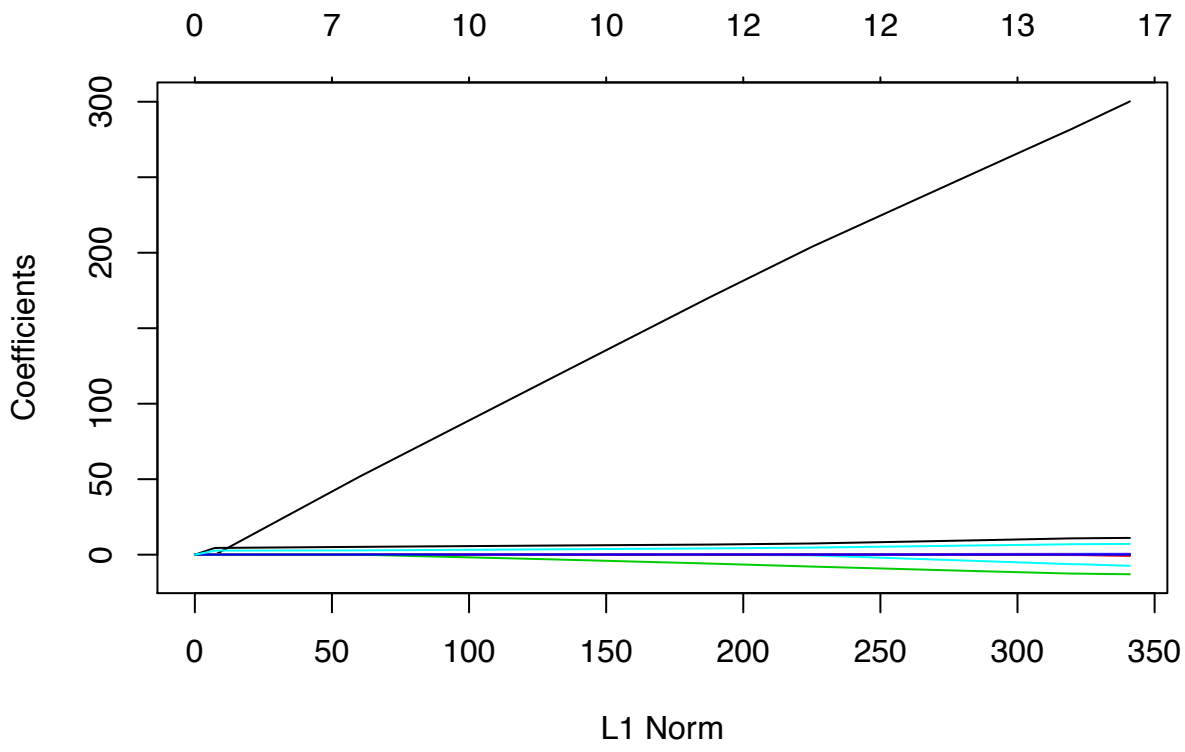

```r
#Now, fit a Lasso regression model to the training data
lasso.model.train <- glmnet(x[train, ], y.train, alpha = 1, lambda = grid.lambda)

#Perform cross validation on the training set to select the best lambda
set.seed(1) #for reproducability
cv.out <- cv.glmnet(x[train, ], y.train, alpha = 1)

#Find the best lambda value
best.lambda <- cv.out$lambda.min

plot(cv.out)
abline(v = log(best.lambda), col = "blue", lwd = 2)
```



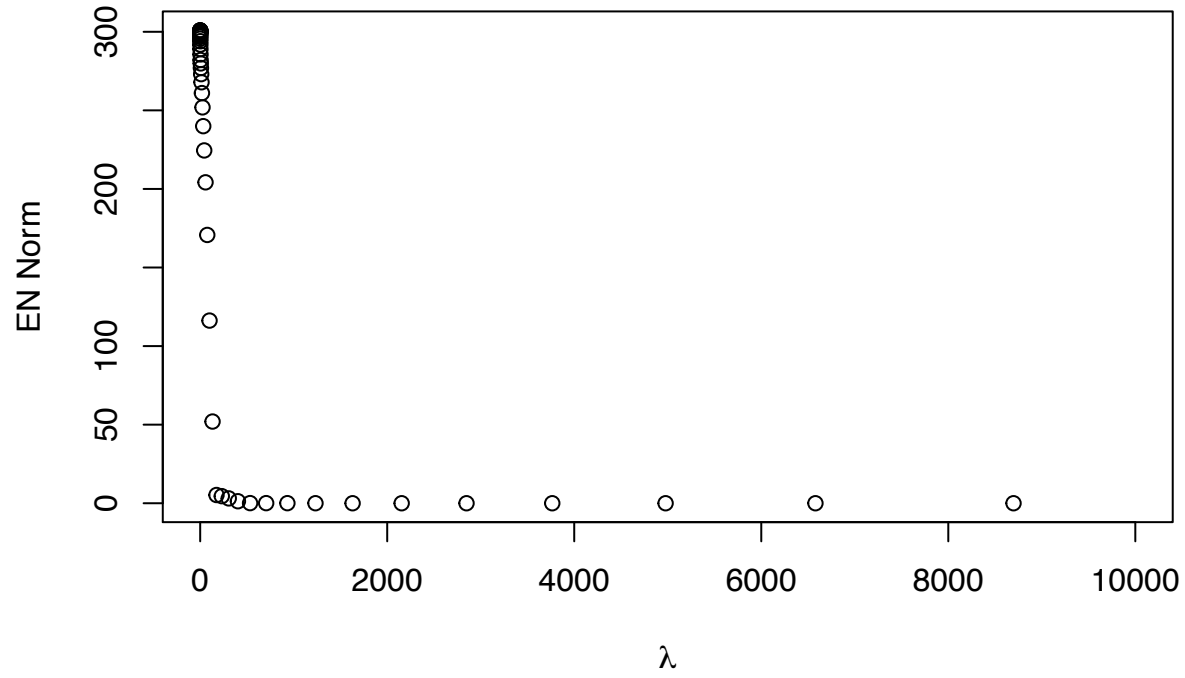

```
best.lambda
```

```
## [1] 22.52383
```

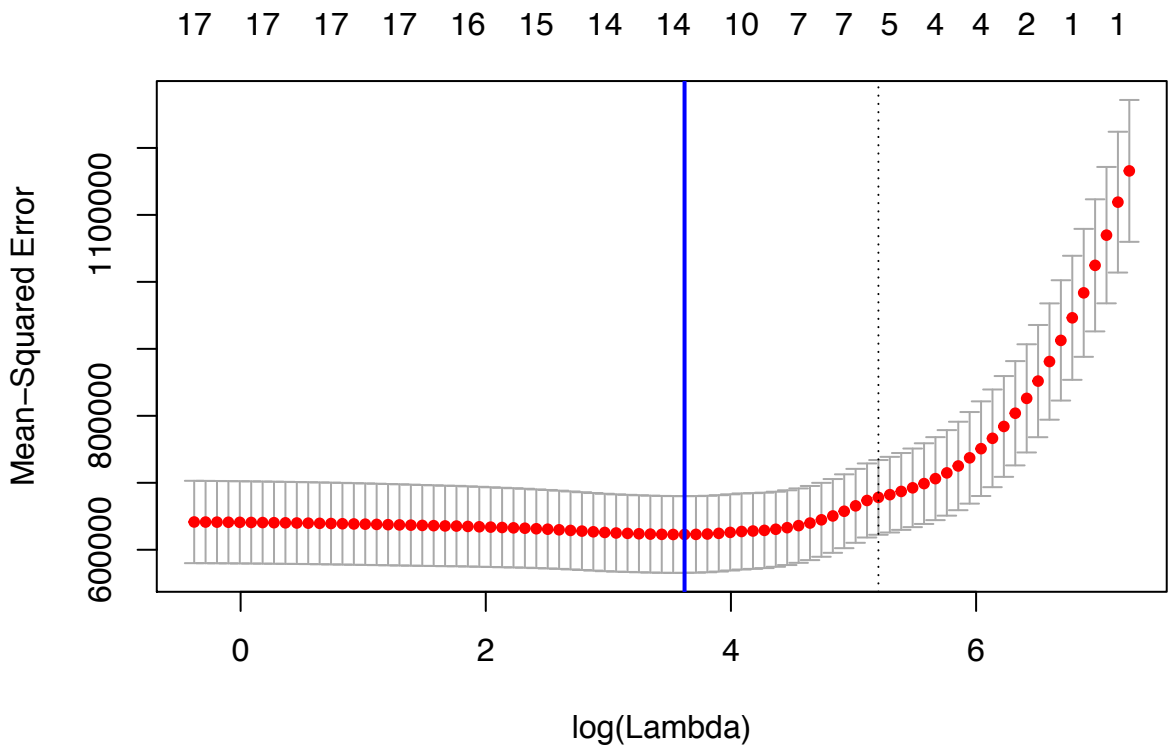
```
#Calculate the MSPE of the model on the test set
lasso.pred <- predict(lasso.model.train, s = best.lambda, newx = x[test,])
mspe.lasso <- mean((lasso.pred - y.test)^2)

#Fit the final model to the entire data set using the chosen lambda
final.model <- glmnet(x, y, alpha = 1, lambda = best.lambda)
Coef.Lasso <- coef(final.model)[1:17,]

##############################
###### PROBLEM 2g ############
##############################

# Elastic Net
#First, let's look at the shrinkage effects of Lasso on the entire data set
EN.model <- glmnet(x, y, alpha = 0.5, lambda = grid.lambda)
plot(EN.model)
```




```
#Let's repeat this for all lambda values and plot the results
ell.EN.norm <- numeric()
for(i in 1:length(grid.lambda)){
  ell.EN.norm[i] <- sqrt(sum(coef(EN.model)[-1, i]^2))
}

plot(x = grid.lambda, y = ell.EN.norm, xlab = expression(lambda),
         ylab = "EN Norm", xlim = c(0,10000))
```



```r
#Now, fit a Lasso regression model to the training data
EN.model.train <- glmnet(x[train, ], y.train, alpha = 0.5, lambda = grid.lambda)

#Perform cross validation on the training set to select the best lambda
set.seed(1) #for reproducability
cv.out <- cv.glmnet(x[train, ], y.train, alpha = 0.5)

#Find the best lambda value
best.lambda <- cv.out$lambda.min

plot(cv.out)
abline(v = log(best.lambda), col = "blue", lwd = 2)
```




```r
best.lambda
```

```
## [1] 37.39936
```

```r
#Calculate the MSPE of the model on the test set
EN.pred <- predict(EN.model.train, s = best.lambda, newx = x[test,])
mspe.EN <- mean((EN.pred - y.test)^2)

#Fit the final model to the entire data set using the chosen lambda
final.model <- glmnet(x, y, alpha = 0.5, lambda = best.lambda)
Coef.EN <- coef(final.model)[1:17,]

Coefficients <- data.frame(Ridge = Coef.Ridge, Lasso = Coef.Lasso, Elastic.Net = Coef.EN)
MSPE <- data.frame(Ridge = mspe.ridge, Lasso = mspe.lasso, Elastic.Net = mspe.EN)

Coefficients
```

```
##                     Ridge         Lasso   Elastic.Net
## (Intercept)  1.583912e+03 1672.49536836 1633.33112087
## PrivateYes   3.108435e+02  204.49244013  232.49795201
## Apps         4.360707e-02    0.03496878    0.03852575
## Accept       2.116675e-02    0.00000000    0.00000000
## Enroll      -1.442461e-01   -0.08637323   -0.10231694
## Top10perc   -3.603766e+00   -2.05315816   -2.56402303
## Top25perc   -1.052955e+00    0.00000000    0.00000000
## F.Undergrad -8.604808e-03    0.00000000    0.00000000
## P.Undergrad  6.242429e-02    0.04140081    0.04517162
## Outstate     1.113241e-01    0.14124110    0.13433481
## Books        5.317524e-01    0.41882489    0.43611333
## Personal    -9.353495e-02   -0.05303090   -0.06109570
## PhD          1.597267e+00    0.00000000    0.00000000
## Terminal     9.672045e+00    8.01455351    8.62626759
## S.F.Ratio   -4.730415e+00    0.00000000    0.00000000
## perc.alumni -1.020618e+01   -9.58628739   -9.63444322
## Expend       2.394058e-02    0.01415988    0.01680877
```

```
MSPE
```

```
##      Ridge    Lasso Elastic.Net
## 1 647638.7 660957.5    657384.3
```

```
BestSubset_Coefficients <- lm2$coefficients
best_model_MSPE <- data.frame(OLS.avg.MSE = mean(best_model$avg_MSE),
                              OLS.avg.MSPE = mean(best_model$avg_MSPE))
best_model_MSPE
```

```
##   OLS.avg.MSE OLS.avg.MSPE
## 1    662060.8     672343.8
```

```
BestSubset_Coefficients
```

```
##  (Intercept)       Accept       Enroll     Outstate        Books
## 2013.4479912    0.1408850   -0.2904891    0.1589863    0.6458207
##    Grad.Rate
##    4.1474836
```

The final model from Ridge regression includes ALL variables from the data set. This is expected because while Ridge does try to "shrink" variables, it rarely is able to shrink variables all the way down to zero. Hence, all 17 variables are included in the model. The Ridge Regression model has the *lowest* MSPE because it uses all of the variables to explain variation in the data. As such, it is expected to have the lowest MSPE.

The final model from Lasso regression is able to completely zero out 5 of the variables in the data. This reduces variance in the model but increases bias into the coefficients.

The Elastic Net model almost matches the Lasso model exactly. This is because Lasso is always a subset of Elastic Net; Elastic Net is a blend of Ridge and Lasso regression models.

part (h)

Of the models selected from ridge regression, the Lasso, and Elastic Net, which model do you prefer? Discuss this in terms of variance, interpretability, inference, and prediction.

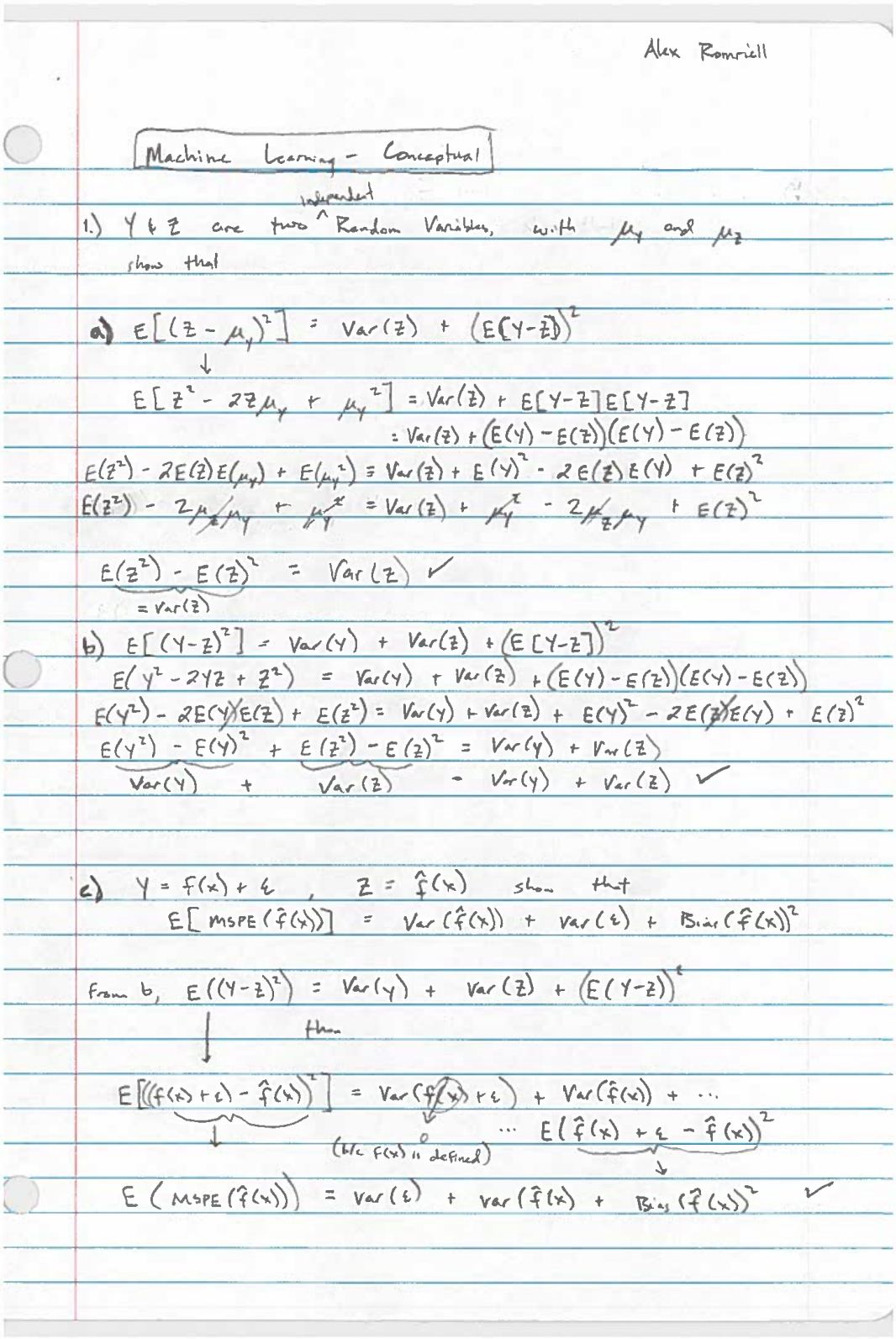
I prefer the Lasso method. Even though it has a worse MSPE than the other models, it simplifies the model by zeroing out some of the variables. This makes the model a bit easier to interpret. Also, since it has fewer explanatory variables than the other models, it will vary less with new data. The other models are subject to change more as more data is added to the model.

Ultimately I prefer the best subset model, but since it is copmutationally expensive. It takes a lot of computing power to iterate through all combinations of the variables and do k-fold validation on top of that. Taking this into consideration, Lasso seems best for selecting variables. However, Ridge regression is a useful tool when the data set is "ill-conditioned."

Elastic Net is an interesting blend of Lasso and Ridge, but interpretibility is even worse than the interpretibiiy of Lasso and Ridge, which is already biased. Interpretibility for Elastic Net is difficult because it uses a hybrid of L1 and L2 penalties to describe $\hat{\beta}$

Conceptual Problems

Problem 1



Problem 2

a. Best Subset has the smallest *training* MSE. Best subset is the most exhaustive process and will find the

combination of variables that minimizes the training MSE.

b. Depends on selection criterion. But, most likely backward stepwise regression will result in the smallest *test* MSE because it starts with all variables and drops them based on significance limits. As a result, it will most likely have a model with many variables and good MSPE.

c. TRUE. Forward-stepwise only adds on models, so the k+1 model will always contain k variables.

d. FALSE. Best subset looks at all combinations. It is possible to have a model with k and k+1 variables where k contains very different variables.

Problem 3

a. Rank (lowest MSPE to highest MSPE): M =: 3, 1, 9, 0.

- 3 has the best fitted model and will provide the best MSPE for a new data point outside the domain
- 1 had the second best fitted model because it captures the overall trend and will likely have a better MSPE than the model with M=9.
- 9 is third best because it has so much variation in the model. There's no telling where the model will place the next value.
- 0 is the worst MSE because it doesn't have any predictor variables in the model. It is only the average of the domain data.

b. Rank (lowest MSE to highest MSE): M =: 9, 3, 1, 0

- 9 uses the most variables to explain variation in the data. MSE will get large as the predictors in the model consecutively get smaller and smaller.

c. Rank (least model variance to highest model variance): M =: 0, 1, 3, 9

- As new data is added to the model, the model with 0 predictors will not shift at all to adjust for the new data point. As predictors increase to 9, the model will vary more and more to fit for the new data. M=0 will wiggle around less than M=9.

Problem 4

part(a, b, and c)

4. $Y = X\beta + \varepsilon$, $\varepsilon \sim N(0, \sigma^2)$

show that

a.) $\underline{E[\hat{\beta}_{OLS}]} = \beta$

From Normal Equations,

$$X^T X \hat{\beta} = X^T Y$$

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

$$= (X^T X)^{-1} X^T (X\beta + \varepsilon)$$

$$= (X^T X)^{-1} X^T X \beta + (X^T X)^{-1} X^T \varepsilon$$

$$\hat{\beta} = \beta + (X^T X)^{-1} X^T \varepsilon$$

$$E(\hat{\beta}) = E(\beta) + E[(X^T X)^{-1} X^T \varepsilon]$$

$$= \beta + (X^T X)^{-1} X^T E(\varepsilon)$$

$E(\varepsilon) = 0$

$$E(\hat{\beta}_{OLS}) = \beta \checkmark$$

$\underline{Var(\hat{\beta}_{OLS})} = \sigma^2 (X^T X)^{-1}$

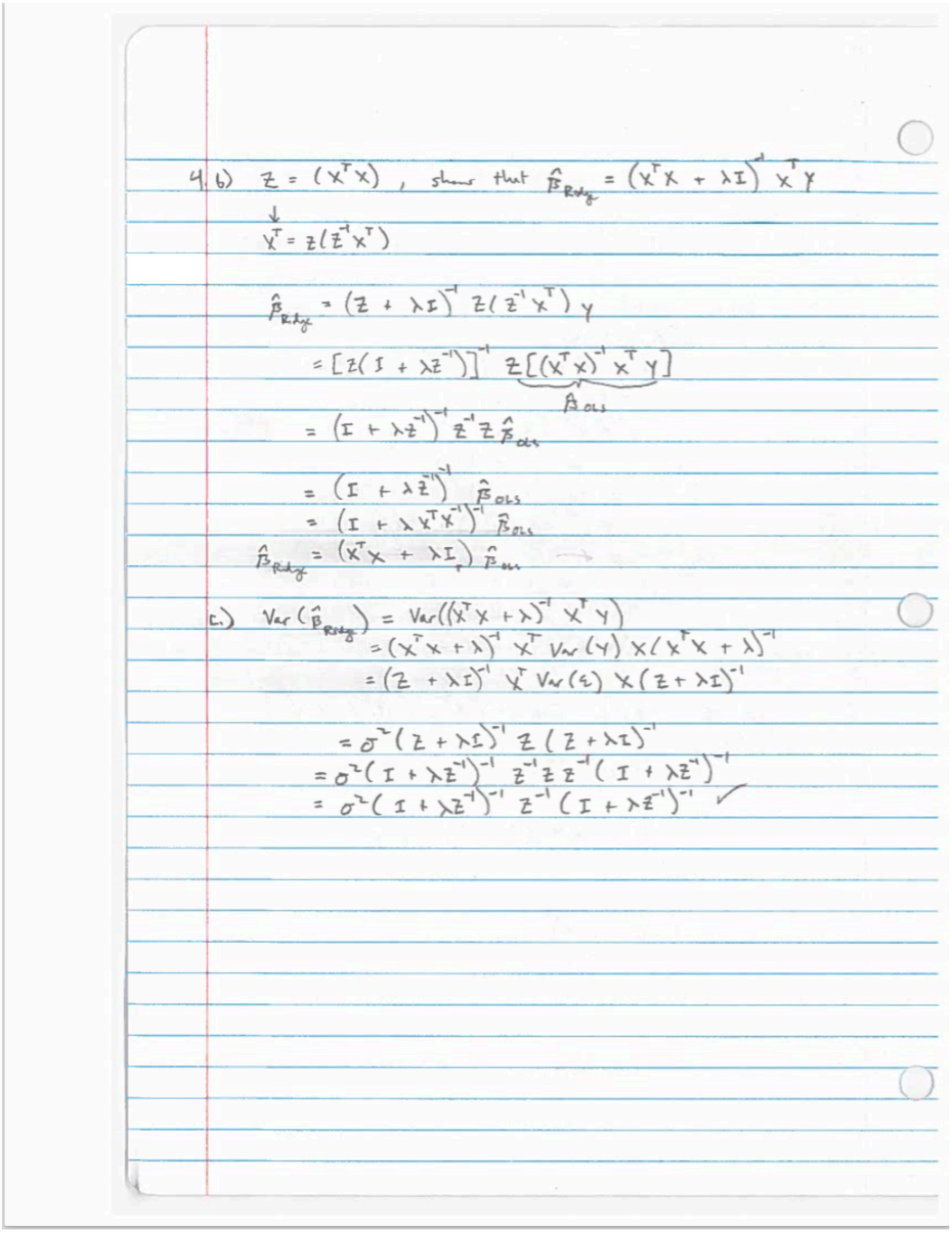
$$E[(\hat{\beta} - \beta)(\hat{\beta} - \beta)^T] = E[((X^T X)^{-1} X^T \varepsilon)((X^T X)^{-1} X^T \varepsilon)^T]$$

$$= E(\hat{\beta}^2) \longrightarrow$$

$$= (X^T X)^{-1} \varepsilon^T \cdot E(\varepsilon \cdot \varepsilon^T) \cdot (X^T X^{-1}) X$$

$$= \underbrace{(X^T X)^{-1} X^T X}_{1} E(\varepsilon \cdot \varepsilon^T) \cdot (X^T X)^{-1}$$

$$= \sigma^2 (X^T X)^{-1} \checkmark$$


4. b) $Z = (X^T X)$, show that $\hat{\beta}_{Ridge} = (X^T X + \lambda I)^{-1} X^T Y$
$X^T = Z(Z^{-1} X^T)$
$\hat{\beta}_{Ridge} = (Z + \lambda I)^{-1} Z(Z^{-1} X^T) Y$
$= [Z(I + \lambda Z^{-1})]^{-1} Z[(X^T X)^{-1} X^T Y]$
$\hat{\beta}_{OLS}$
$= (I + \lambda Z^{-1})^{-1} Z^{-1} Z \hat{\beta}_{OLS}$
$= (I + \lambda Z^{-1})^{-1} \hat{\beta}_{OLS}$
$= (I + \lambda X^T X^{-1})^{-1} \hat{\beta}_{OLS}$
$\hat{\beta}_{Ridge} = (X^T X + \lambda I) \hat{\beta}_{OLS}$
c.) $Var(\hat{\beta}_{Ridge}) = Var((X^T X + \lambda)^{-1} X^T Y)$
$= (X^T X + \lambda)^{-1} X^T Var(Y) X (X^T X + \lambda)^{-1}$
$= (Z + \lambda I)^{-1} X^T Var(\varepsilon) X (Z + \lambda I)^{-1}$
$= \sigma^2 (Z + \lambda I)^{-1} Z (Z + \lambda I)^{-1}$
$= \sigma^2 (I + \lambda Z^{-1})^{-1} Z^{-1} Z Z^{-1} (I + \lambda Z^{-1})^{-1}$
$= \sigma^2 (I + \lambda Z^{-1})^{-1} Z^{-1} (I + \lambda Z^{-1})^{-1}$ ✓

part d

```
sigma2 <- 2
lambda <- 2

mm <- matrix(rep(0), nrow=2, ncol=2)
mm[1,1] <- 1
mm[1,2] <- 0.7
mm[2,1] <- 1
mm[2,2] <- 0.69

Z <- t(mm) %*% mm
Z_inv <- solve(Z)
id_matrix <- diag(2)

Var.B.OLS <- sigma2 * solve(Z)
Var.B.OLS
```

```
##        [,1]   [,2]
## [1,]  19322 -27800
## [2,] -27800  40000
```

```
Var.B.Ridge <- sigma2*solve((id_matrix + lambda * Z_inv))*Z_inv*solve((id_matrix + lambda*Z_inv))
Var.B.Ridge
```

```
##           [,1]      [,2]
## [1,]  3133.965 -2177.884
## [2,] -2177.884  1513.943
```

The reason why these two estimates for $\hat{\beta}$ are so different is because of the *bias-variance* trade-off. With OLS, the variance of the different models is usually very high. Predictions from OLS can change dramatically for slight changes in X. Ridge estimation, however, reduces this variability substantially, but at the sacrifice of introducing bias into the models. This bias is seen in the coefficients for the selected model. The value of the coeffiecient will be conflated or deflated due to model "shrinkage." This reduces variance in the model while still providing a good predictive model. Since variance is reduced, it is easier to get a real sense for the general trends in the data because the model is less influenced by new data points.