Review Week Six Answers

1. Chapter 13, question 2.
   (a) A_j has only two possible outcomes each having a fixed probability, α of rejection and 1-α of acceptance. Accordingly, A_j is a Bernoulli random variable – like a single coin flip.
   (b) The $\sum_{j=1}^{m} A_j$ is the count of "successes", hypothesis rejections, and has a binomial distribution.
   (c) The expected number of type-I errors is $m\alpha$ and the standard deviation is $\sqrt{m\alpha(1-\alpha)}$.
2. Chapter 6, question 11(a) but compare the lasso to PCR only.

```
# Problem 6-11(a)
library(ISLR2)
library(glmnet)
attach(Boston)
names(Boston)

[1] "crim"    "zn"      "indus"   "chas"    "nox"     "rm"
"age"     "dis"     "rad"
[10] "tax"     "ptratio" "lstat"   "medv"

# "crim" is the reposonse variable per-capita crimne per town.
# Some of the other variables are "indus" proportion of non-
retail business acres per town,
# "age" proportion of owner-occupied units built prior to 1940,
"medv"
# median value of owner-occupied homes in $1000s
# There are a total of 506 observations.

# First we will use the Lasso

x <- model.matrix(crim~ ., Boston)[, -1]
y<- Boston$crim
grid <- 10^seq(10, -2, length = 100)
# generate a list of training samples that is 80% of the total
data set
train<- sample(1:506,round(506*0.8))
test<- (-train)
# Use cross-validation to choose the best lambda
set.seed(5)
cv.lasso<- cv.glmnet(x[train,],y[train],alpha=1,lambda=grid)
best.lam<- cv.lasso$lambda.min
```

```
lasso.fit<- glmnet(x[train,],y[train],alpha=1,lambda=grid)
lasso.coef<- predict(lasso.fit,type="coefficients",s=best.lam)
lasso.coef

13 x 1 sparse Matrix of class "dgCMatrix"
                     s1
(Intercept) 10.81716387
zn           0.02814192
indus       -0.04395290
chas        -0.89206094
nox         -7.93473135
rm          -0.29456628
age          .
dis         -0.62792395
rad          0.51797594
tax          .
ptratio     -0.23577132
lstat        0.20046353
medv        -0.0747466

onesd.lam<-cv.lasso$lambda.1se # The +1sd lambda
lasso.coef.1sd<-
predict(lasso.fit,type="coefficients",s=onesd.lam)
lasso.coef.1sd

13 x 1 sparse Matrix of class "dgCMatrix"
                     s1
(Intercept) 0.782102175
zn          .
indus       .
chas        .
nox         .
rm          .
age         .
dis         .
rad         0.275124204
tax         .
ptratio     .
lstat       0.005796587
medv        .

# Predict crime based on test set features
lasso.pred.best<- predict(lasso.fit,s=best.lam,newx=x[test,])
lasso.pred.1sd<- predict(lasso.fit,s=onesd.lam,newx=x[test,])
```

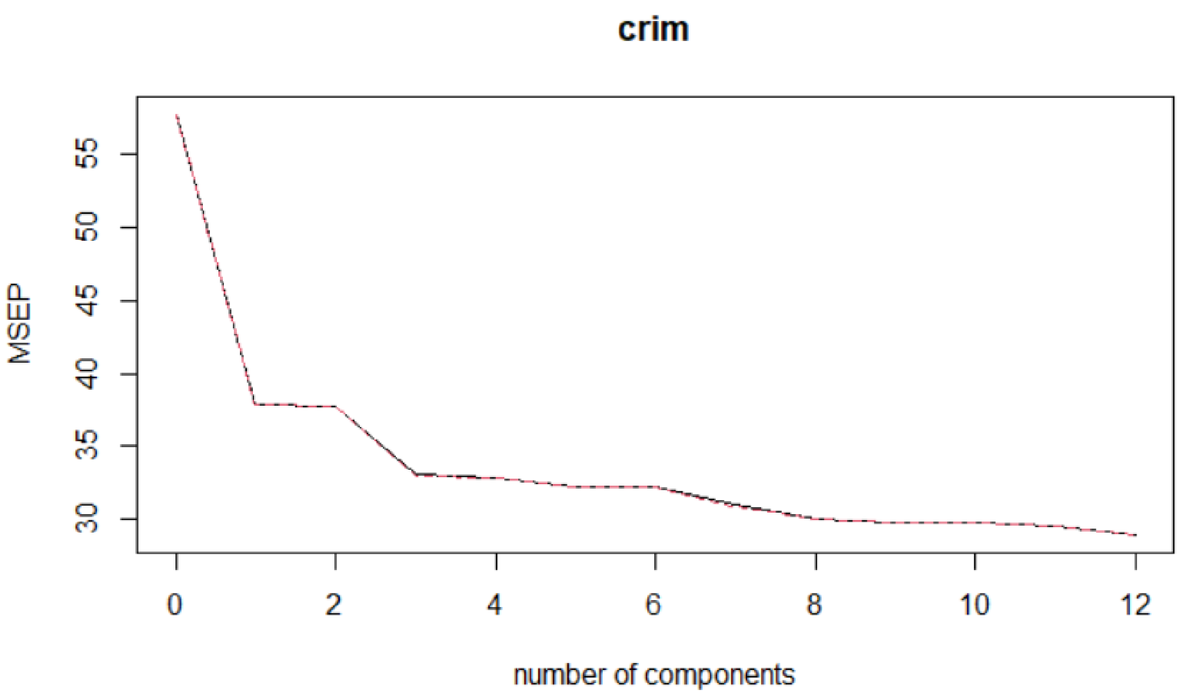
```
# Compute the MSE of the test set predictions
mean(((lasso.pred.best-y[test]))^2) # This is the best lambda
96.20112
mean(((lasso.pred.1sd-y[test]))^2) # This is for the +1sd lambda
114.8725

############
# Next we do principal components
###########
library(pls)
set.seed(6)
pcr.fit <- pcr(crim~ ., data = Boston , subset = train,scale =
TRUE , validation = "CV")
Data:     X dimension: 405 12
     Y dimension: 405 1
Fit method: svdpc
Number of components considered: 12

VALIDATION: RMSEP ###These numbers are the square root of the
MSE
Cross-validated using 10 random segments.
       (Intercept)  1 comps  2 comps  3 comps  4 comps  5 comps
6 comps  7 comps  8 comps
CV           7.601    6.157    6.145    5.745    5.728    5.673
5.677    5.563    5.478
adjCV        7.601    6.154    6.143    5.740    5.725    5.670
5.674    5.551    5.473
       9 comps  10 comps  11 comps  12 comps
CV       5.455     5.456     5.436     5.379
adjCV    5.449     5.451     5.431     5.374

TRAINING: % variance explained
      1 comps  2 comps  3 comps  4 comps  5 comps  6 comps  7
comps  8 comps  9 comps
X       50.64    64.43    73.28    80.42    86.81    90.27
92.77    95.09    96.78
crim    35.11    35.51    43.50    44.22    45.30    45.36
47.83    49.35    49.85
      10 comps  11 comps  12 comps
X        98.39     99.51    100.00
crim     50.01     50.62     51.63
```

```
validationplot(pcr.fit , val.type = "MSEP") # This hits a
minimum at 7 PC's and then goes up
```




```
# The y-axis on this plot is the MSE
pcr.pred <- predict(pcr.fit , x[test , ], ncomp = 5)
mean(((pcr.pred-y[test]))^2) # MSE
101.5411

# If we use all 12 PC's the MSE on the test data is 95.65012
```

In these examples Principal components does better than the lasso using all 12 PC's while the lasso used 10 variables.

In general, the lasso may be better in situations where $p \gg n$. Additionally, using all 12 PC's does not aide in understanding which variables are most important.