STAT435 HW3

Liyuan Tang

5/3/2020

1

(a)

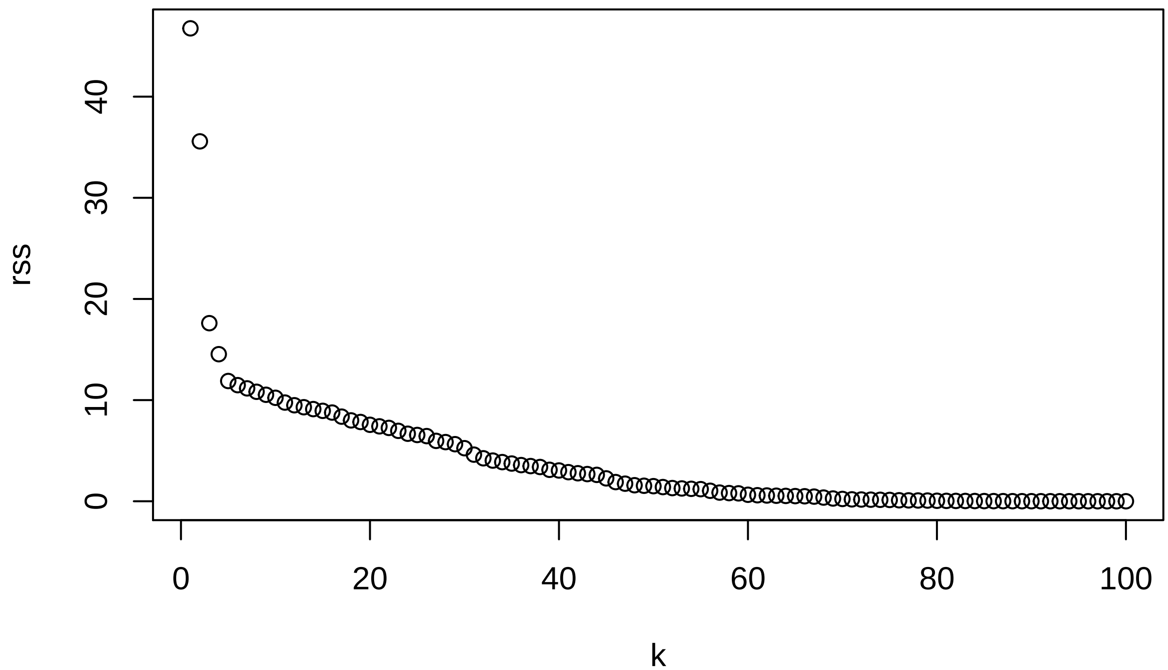
```r
source('test-data.r')
library(leaps)
library(glmnet)
truncated.power.design.matrix <- function(x){
  n = length(x)
  M = matrix(data = NA, n, n)
  for (i in 1:n) {
    for (j in 1:n) {
      M[i, j] = ifelse(x[i] - x[j] >= 0, x[i]-x[j], 0)
    }
    M[i, n] = 1
  }
  return(M)
}
```

(b)

```r
regsubsets.fitted.values <- function(X, regsubsets.out, nterm){
  coef.reg = coef(regsubsets.out, id=nterm)
  pred = as.matrix(X[,(names(coef.reg))]) %*% coef.reg
  return(pred)
}
```

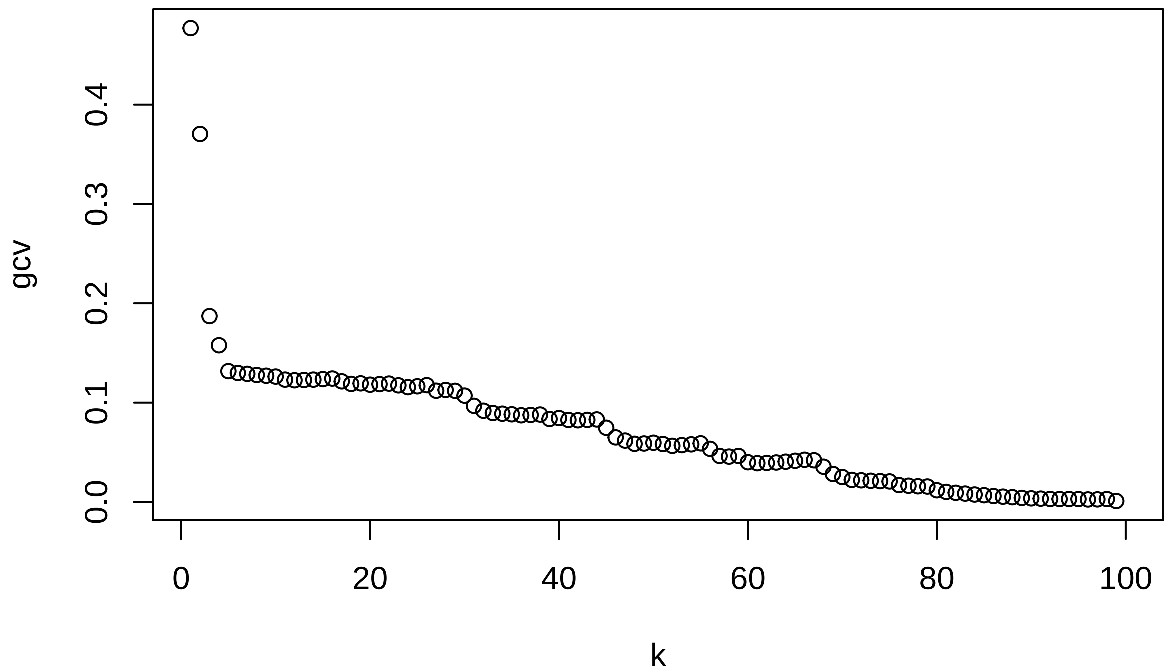
(c)

```r
M = truncated.power.design.matrix(x)
n = 100
df = data.frame(M, y)
mat = df[,-101]
regsubsets.out = regsubsets(y~., data = df, nvmax = n, intercept = FALSE, method = 'forward')
rss = numeric(n)
for (i in 1:n) {
  y.pred = regsubsets.fitted.values(mat, regsubsets.out, i)
  rss[i] = sum((y - y.pred)^2)
}
k = 1:n
plot(k, rss)
```



(d)

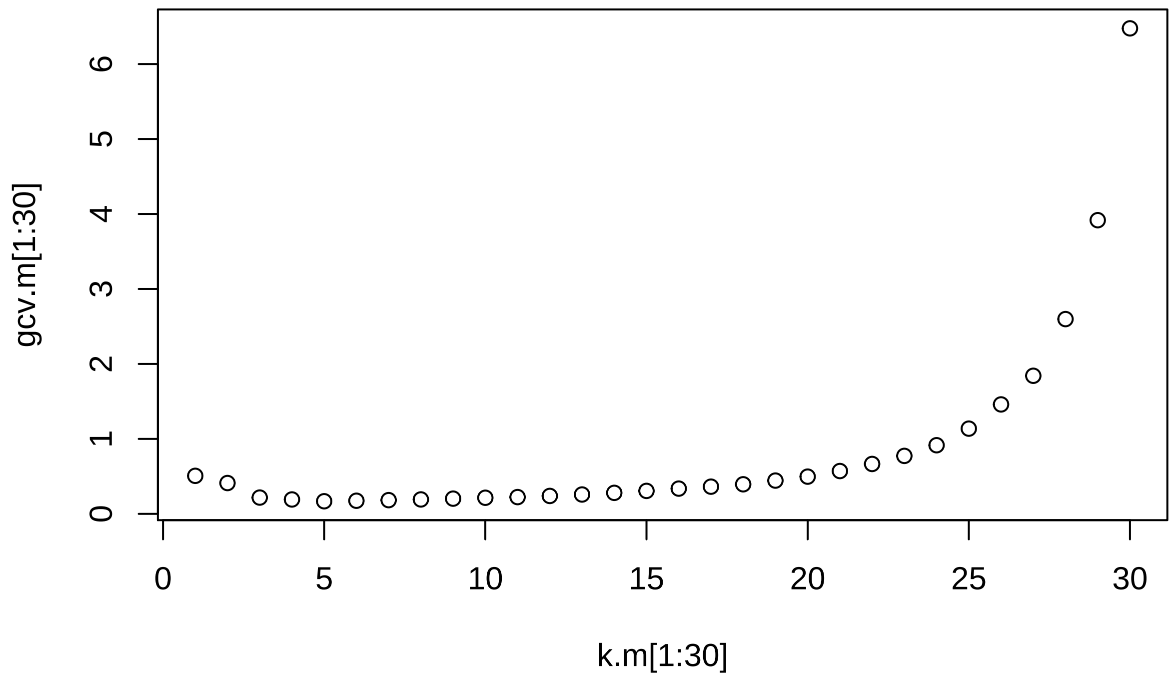
```
gcv = numeric(n)
for (i in 1:n) {
  gcv[i] = 1/n * rss[i] /(1-i/n)^2
}
plot(k, gcv)
```



The GCV scores continues decreasing as the number of k increases. I think it is because we used y during the best subsets selection so we cannot use cross validation.

(e)

```
n=100
gcv.m = numeric(n)
for (i in 1:n) {
  gcv.m[i] = rss[i] / ( (1-(3*i+1)/100)^2 * n)
}
k.m = 1:n
# First 30 terms
plot(k.m[1:30], gcv.m[1:30])
```



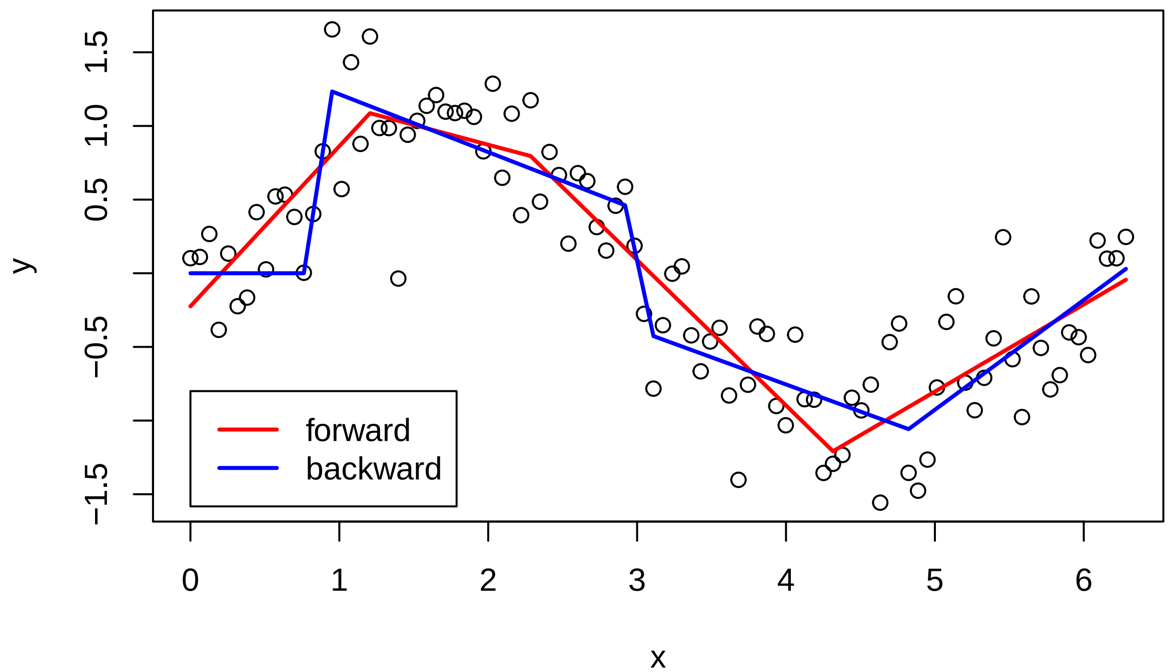
(f)

```
k.forward = which.min(gcv.m[1:30])
gcv.for.m = min(gcv.m[1:30])

# backward

regsubsets.bck = regsubsets(y~., data = df, nvmax = 100, intercept = FALSE, method = 'backward')
gcv.m2 = numeric(n)
for (i in 1:n) {
  y.pred = regsubsets.fitted.values(mat, regsubsets.bck, i)
  gcv.m2[i] = 1/100 * sum((y - y.pred)^2) / (1-(3*i+1)/100)^2
}
k.backward = which.min(gcv.m2[1:30])
gcv.back.m = min(gcv.m2[1:30])

cat('The smallest GCV score for forward method is', gcv.for.m, 'at k =', k.forward, '.', 'The smallest (
```

```
## The smallest GCV score for forward method is 0.1684513 at k = 5 . The smallest GCV score for backwar
```

```
y.pred.back = regsubsets.fitted.values(mat, regsubsets.bck, k.backward)
y.pred.for = regsubsets.fitted.values(mat, regsubsets.out, k.forward)

plot(x, y)
```

```r
lines(x, y.pred.for, col = "red", lwd = 2)
lines(x, y.pred.back, col = "blue", lwd = 2)
legend(0, -0.8, legend=c("forward", "backward"), col=c("red", "blue"), lwd =2)
```



2

(a) $\hat{a} = argmin_a[||y - Xa||^2 + \lambda a^T \Omega a]$

$\frac{\partial}{\partial a}(||y - Xa||^2 + \lambda a^T \Omega a) = 0$

$-2X^T y + 2X^T X a + 2\lambda \Omega a = 0$

$(X^T X + \lambda \Omega) a = X^T y$

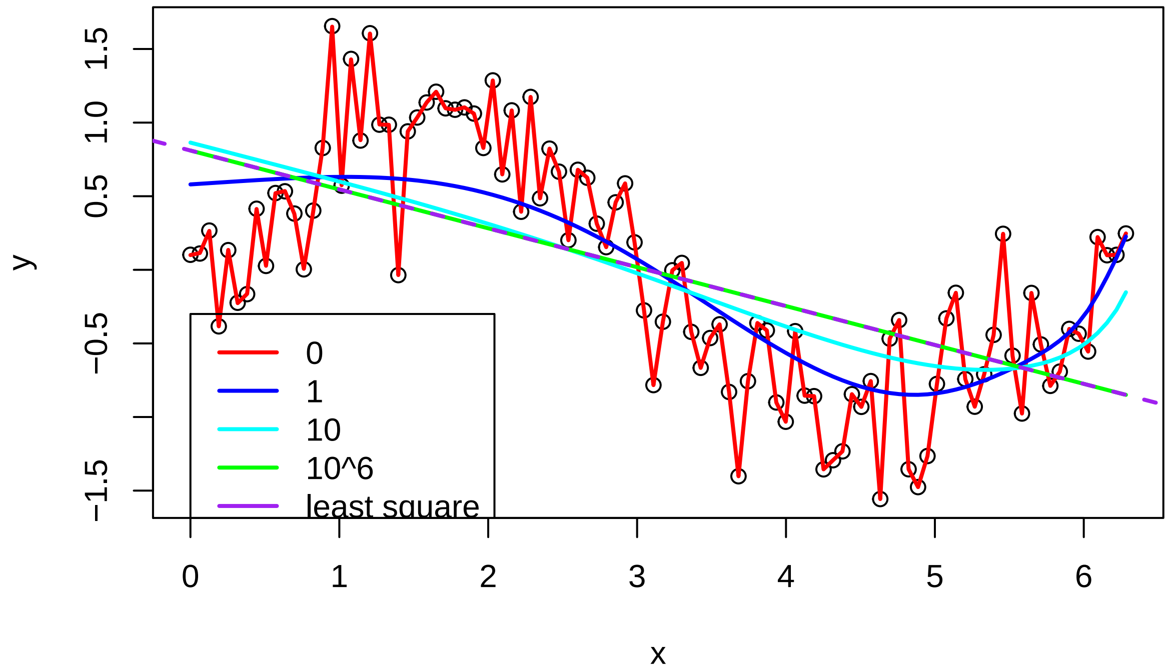
$a = (X^T X + \lambda \Omega)^{-1} X^T y$

So, $\hat{y} = X\hat{a} = X(X^T X + \lambda \Omega)^{-1} X^T y$

(b)

```r
fit = glmnet(M, y, alpha = 0, lambda=c(0, 1, 10, 10^6), intercept = TRUE, thresh = 1e-12, maxit = 10^7,
pred = predict(fit,newx=M, s=c(0, 1, 10, 10^6))
lm.fit = lm(y~x)
plot(x, y)
points(x, pred[, 1], col = 'red', type = 'l', lwd = 2)
```

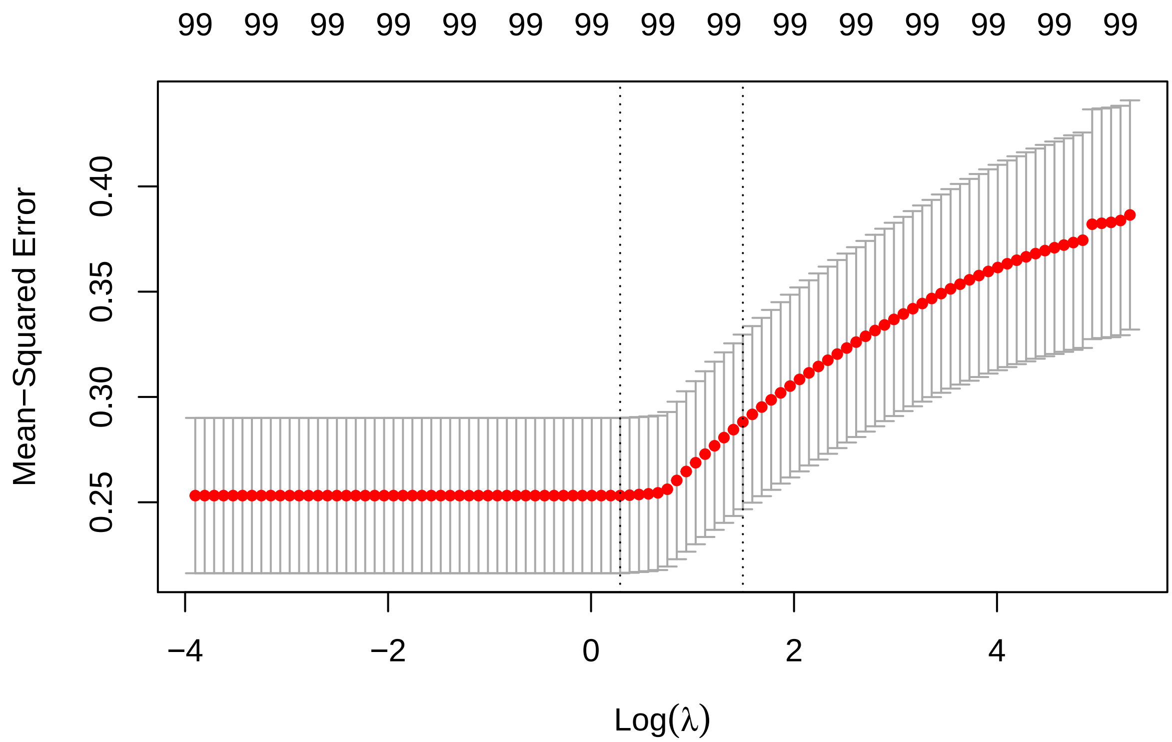
```r
points(x, pred[, 2], col = 'blue', type = 'l', lwd = 2 )
points(x, pred[, 3], col = 'cyan', type = 'l', lwd = 2)
points(x, pred[, 4], col = 'green', type = 'l', lwd = 2)
abline(lm.fit, col = 'purple', lty=2, lwd = 2)
legend(0, -0.3, legend=c("0", "1", "10", "10^6", "least square"),col=c("red", "blue", "cyan", "green",
```



From the graph, we can see the green line ($\lambda = 10^6$) and purple line (least squares line) are almost overlapping.

(c)

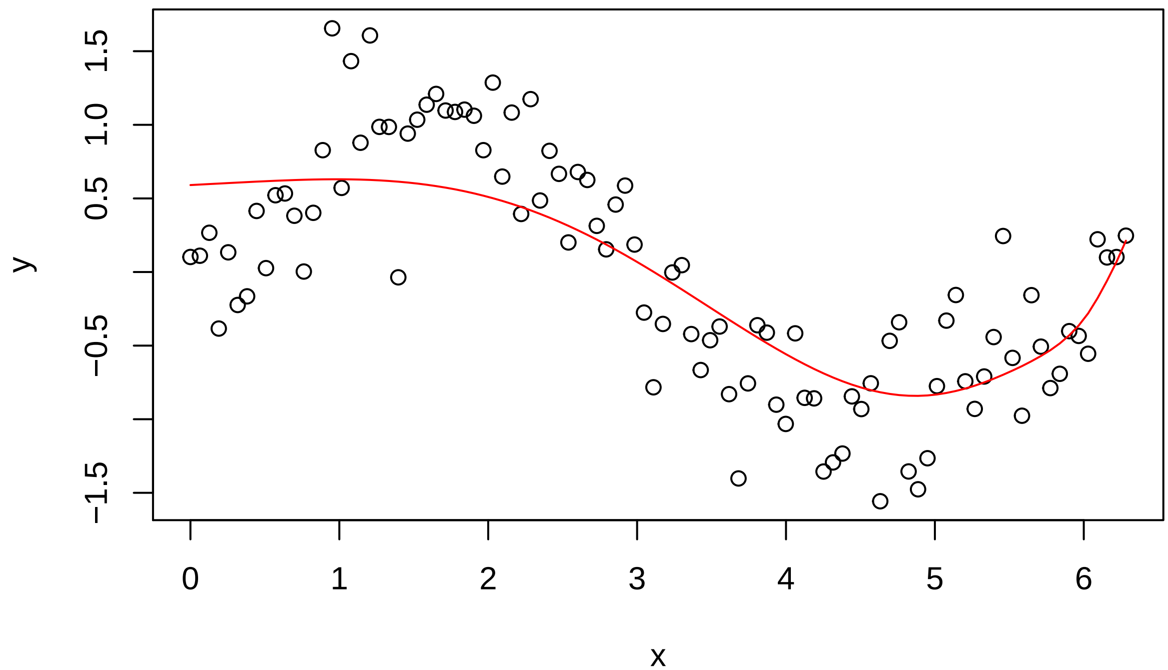
```r
cvfit = cv.glmnet(M, y, alpha = 0, thresh = 1e-12, maxit = 10^7, penalty.factor = c(0,rep(1,98),0))
plot(cvfit)
```




```
lambda.min = cvfit$lambda.min
cat('The optimal lambda is', lambda.min)
```

```
## The optimal lambda is 1.331724
```

```
pred = predict(fit,newx=M, s=lambda.min)
plot(x, y)
points(x, pred, col = 'red', type = 'l')
```



6.8

Question 1

(a). Best subset selection will have the smallest training RSS since it will consider all combination of predictors.

(b). Best subset selection may have the smallest test RSS since it will calculate all possible models. But the rest models may also get a smaller test RSS by chance.

(c). i. True. Since we add one more predictor based on the k-variable model.

ii. True. Since we remove one additional predictor from (k+1)-variable to get k-variable model.

iii. False. There is no relation between forward and backward selection.

iv. False. Same reason as the previous.

v. False. The example is the model we used during the lab section.

Question 4

(a). It will increased as the answer **iii**. Since we increase λ, we will decrease the β_j which results the increasing in training RSS.

(b). **ii** is correct. When λ is 0, we will have a high test RSS since the model is very flexible. As we increase the *lambda*, the model will become less flexible which reduces the test RSS. But as *lambda* keep increasing, all β_j will become 0, so the test RSS eventually starts to increase.

(c). **iv** is correct. Since the model becomes less flexible as we increase the λ, the variance will decrease.

(d). **iii** is correct. Bias will change in the opposite direction as the variance.

(e). **v** is correct. Irreducible error will not change as the model changes by definition.