HW3

Nan Tang

4/28/2020

Problem 1

(a)

```
truncated.power.design.matrix <- function(x) {
  n <- length(x)
  results <- matrix(0, n, n)
  results[, n] <- 1
  knots <- sort(x)
  for (i in 1:n) {
    x_i = x[i]
    for (j in 1:(n-1)) {
      knot_j <- knots[j]
      if (x_i > knot_j) {
        results[i, j] <- x_i - knot_j
      } else {
        break
      }
    }
  }
  return(results)
}
```

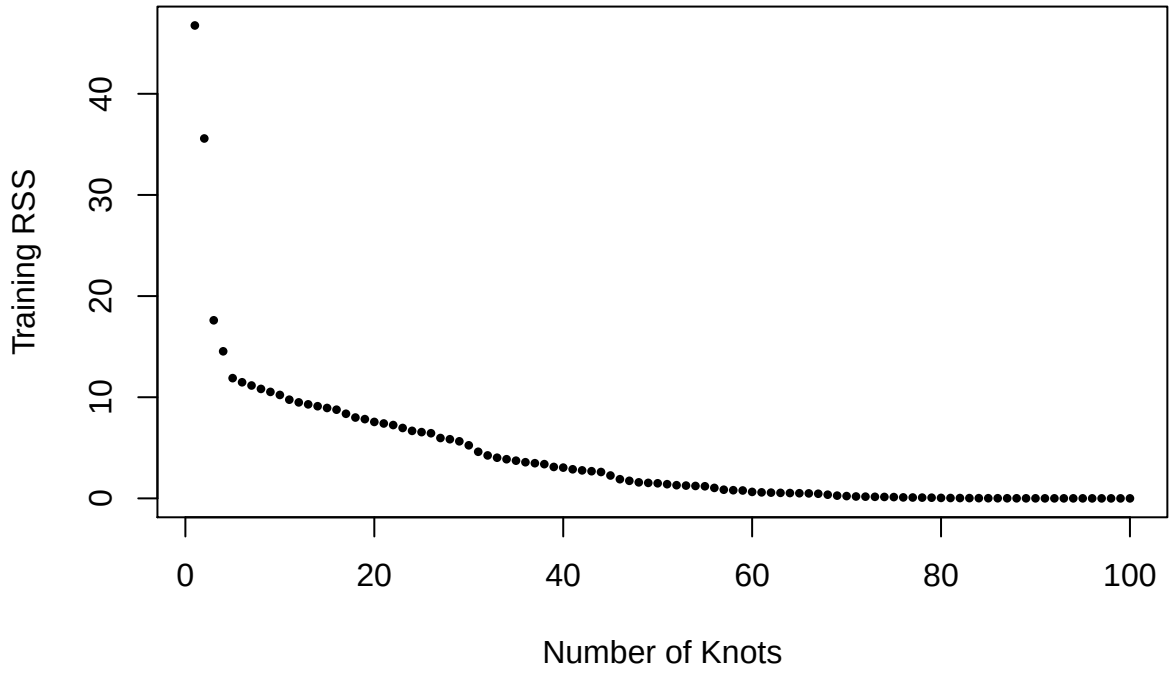
(b)

```
regsubset.fitted.values <- function(X, y, nterm) {
  reg_out <- regsubsets(X, y, nvmax=nterm, method='forward', intercept=FALSE)
  knot_dm <- X[, which(summary(reg_out)$which[nterm,])]
  yhat <- knot_dm %*% solve(t(knot_dm) %*% knot_dm) %*% t(knot_dm) %*% y
  return(yhat)
}
```

(c)

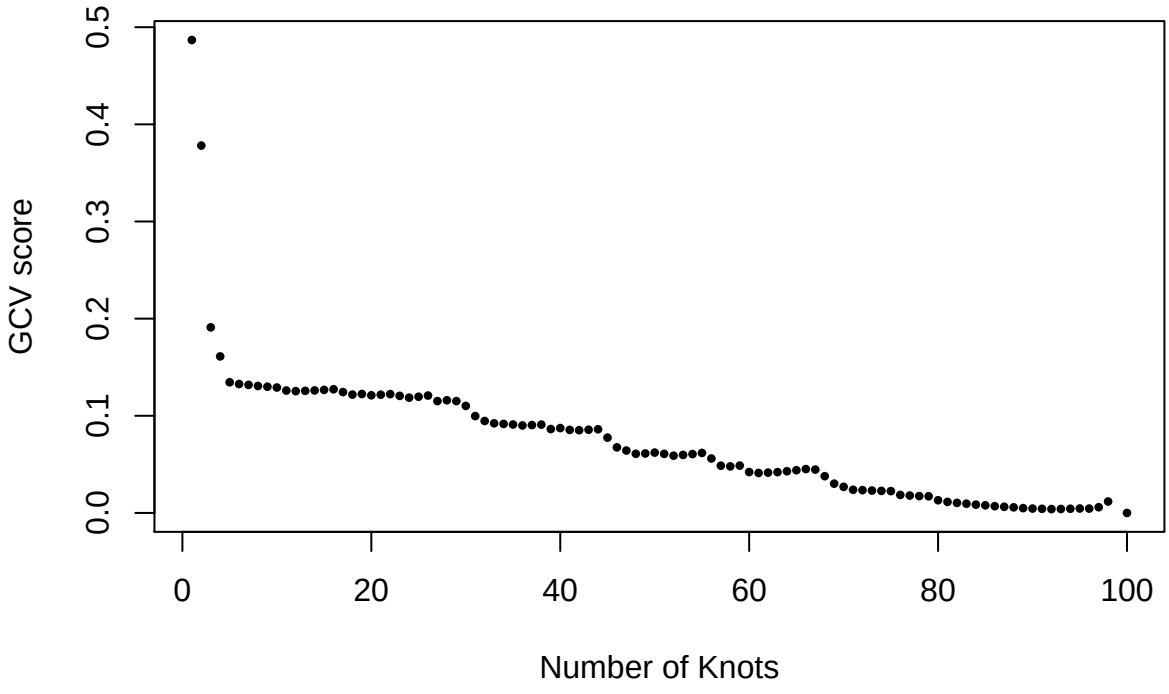
```
train_size <- length(x)
knot_rss <- numeric(train_size)
X <- truncated.power.design.matrix(x)

for (i in 1:train_size) {
```

```
  yhat <- regsubset.fitted.values(X, y, i)
  rss_temp <- sum((y - yhat)^2)
  knot_rss[i] <- rss_temp
}

plot(knot_rss, type='p', pch=16, cex=0.6,
     xlab='Number of Knots', ylab='Training RSS')
```



(d)

```
gcv <- numeric(train_size)

for (i in 1:train_size) {
  dk = i + 1
  gcv_temp <- knot_rss[i] / (train_size * (1 - dk/train_size)^2)
  gcv[i] <- gcv_temp
}

plot(gcv, type='p', pch=16, cex=0.6,
     xlab='Number of Knots', ylab='GCV score')
```

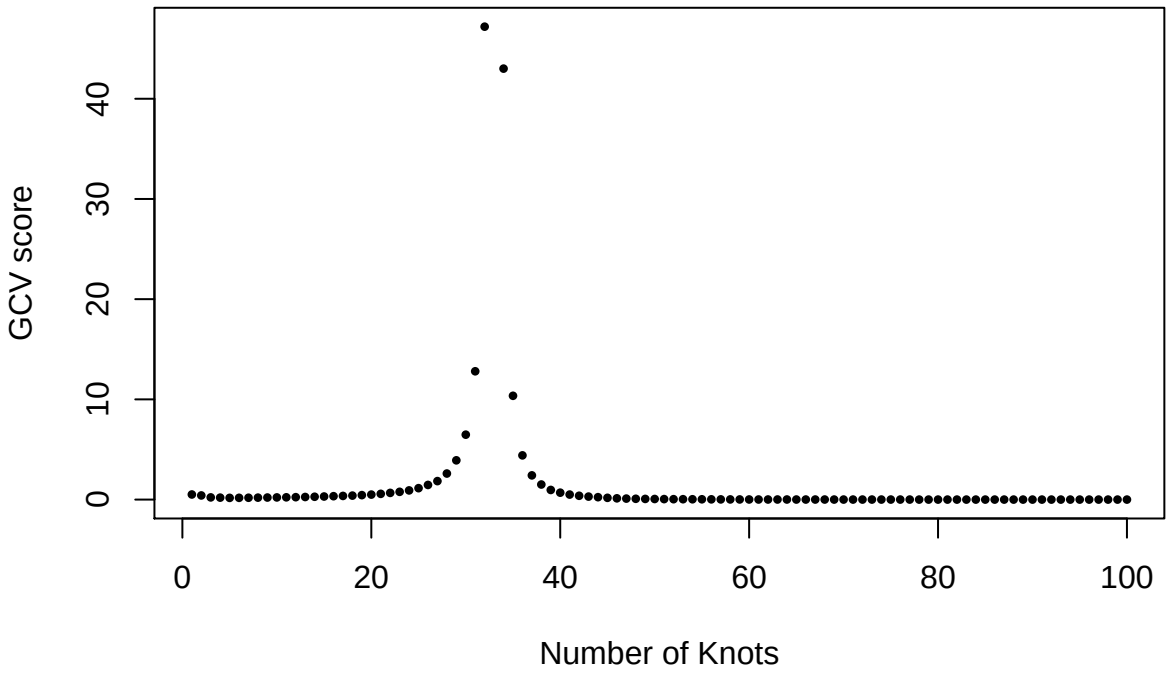


The plot of GCV scores shows that more knots we apply, lower GCV scores we will get, i.e. the model that includes all points of data as knot are the best model. It is conterintuitive, because such model generally has high model variance. In general, along with increasing number of predictor, GCV score should drop first and rise again, forming a 'U' shape.

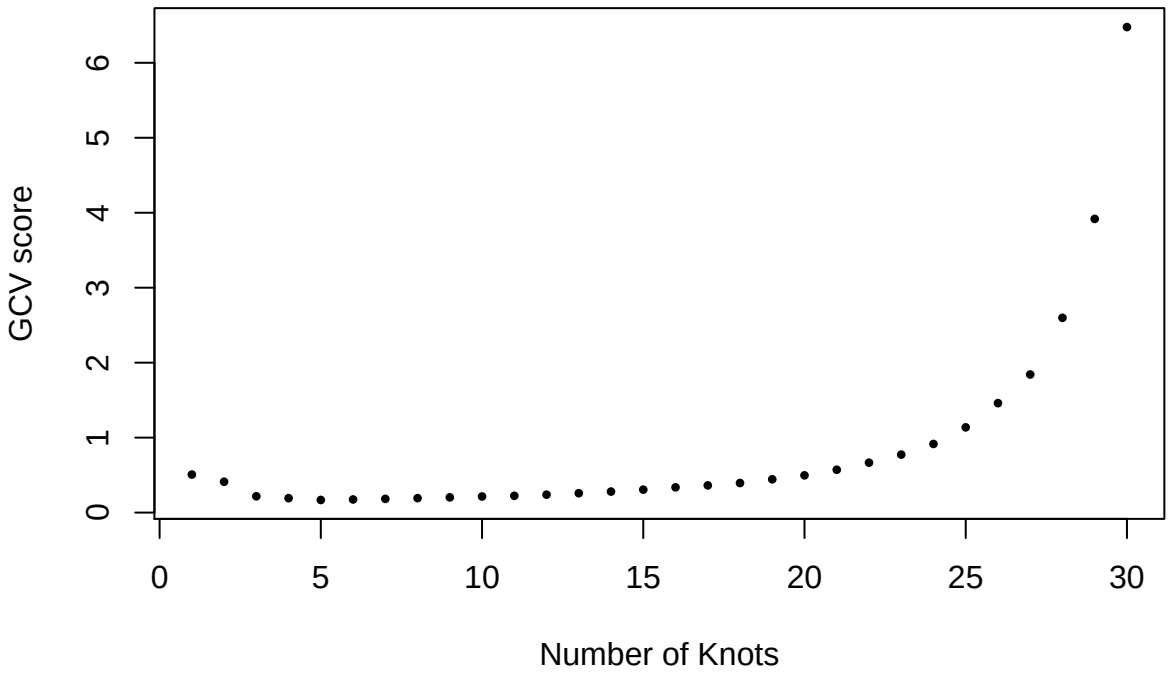
Explain: subset model selection depends on all values of y, therefore assumptions of independence for cross validation is violated.

(e)

```
new_gcv <- numeric(train_size)

for (i in 1:train_size) {
  dk = 3*i + 1
  gcv_temp <- knot_rss[i] / (train_size * (1 - dk/train_size)^2)
  new_gcv[i] <- gcv_temp
}

plot(new_gcv, type='p', pch=16, cex=0.6,
     xlab='Number of Knots', ylab='GCV score')
```




```
## for k = 30 at most
plot(new_gcv[1:30], type='p', pch=16, cex=0.6,
     xlab='Number of Knots', ylab='GCV score')
```



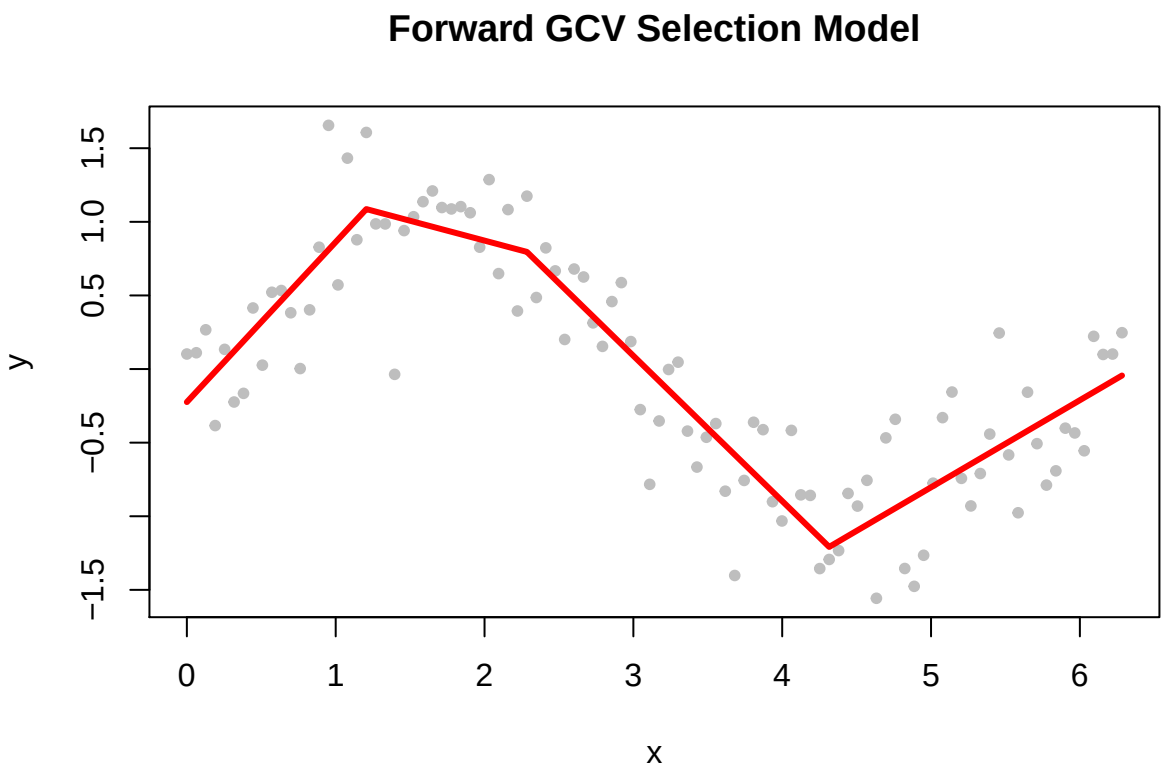
Yes, I was surprised, since there are one "U" shape along with additional "L" shape. Meaning that GCV scores still drop for increasing number of knots, which is counter-intuitive.

(f)

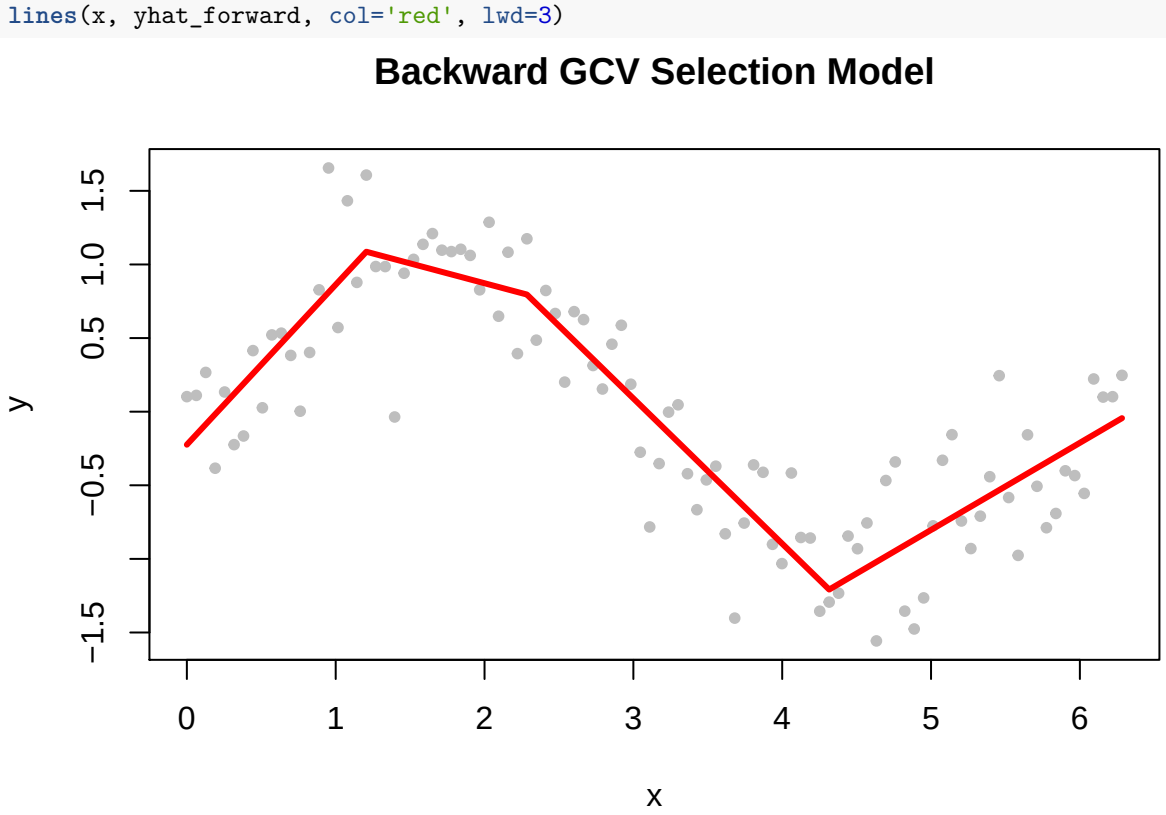
```
## restrict k to range 1:30
k_forward_opt <- which(new_gcv == min(new_gcv[1:30]))

## backward selection
back_knot_rss <- numeric(30)

regsubset.fitted.values.backward <- function(X, y, nterm) {
  reg_out <- regsubsets(X, y, nvmax=nterm, method='backward', intercept=FALSE)
  knot_dm <- X[, which(summary(reg_out)$which[nterm,])]
  yhat <- knot_dm %*% solve(t(knot_dm) %*% knot_dm) %*% t(knot_dm) %*% y
  return(yhat)
}

for(i in 1:30) {
  yhat <- regsubset.fitted.values.backward(X, y, i)
  rss_temp <- sum((y - yhat)^2)
  back_knot_rss[i] <- rss_temp
}

back_gcv <- numeric(30)

for (i in 1:30) {
  dk = 3*i + 1
  gcv_temp <- back_knot_rss[i] / (train_size * (1 - dk/train_size)^2)
  back_gcv[i] <- gcv_temp
}

k_backward_opt <- which(back_gcv == min(back_gcv))

c(k_forward_opt, k_backward_opt)
```

```
## [1] 5 5
```

```
yhat_forward <- regsubset.fitted.values(X, y, k_forward_opt)
plot(x, y, col='grey', pch=20, main='Forward GCV Selection Model')
lines(x, yhat_forward, col='red', lwd=3)
```




```
yhat_backward <- regsubset.fitted.values.backward(X, y, k_backward_opt)
plot(x, y, col='grey', pch=20, main='Backward GCV Selection Model')
lines(x, yhat_forward, col='red', lwd=3)
```



If we restrict number of knots to at most 30, then it turns out knots = 5 has lowest GCV score in both forward and backward model selection.

Problem 2

(a)

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

estimator of coefficient a is valid when partial derivative of this formula is zero.

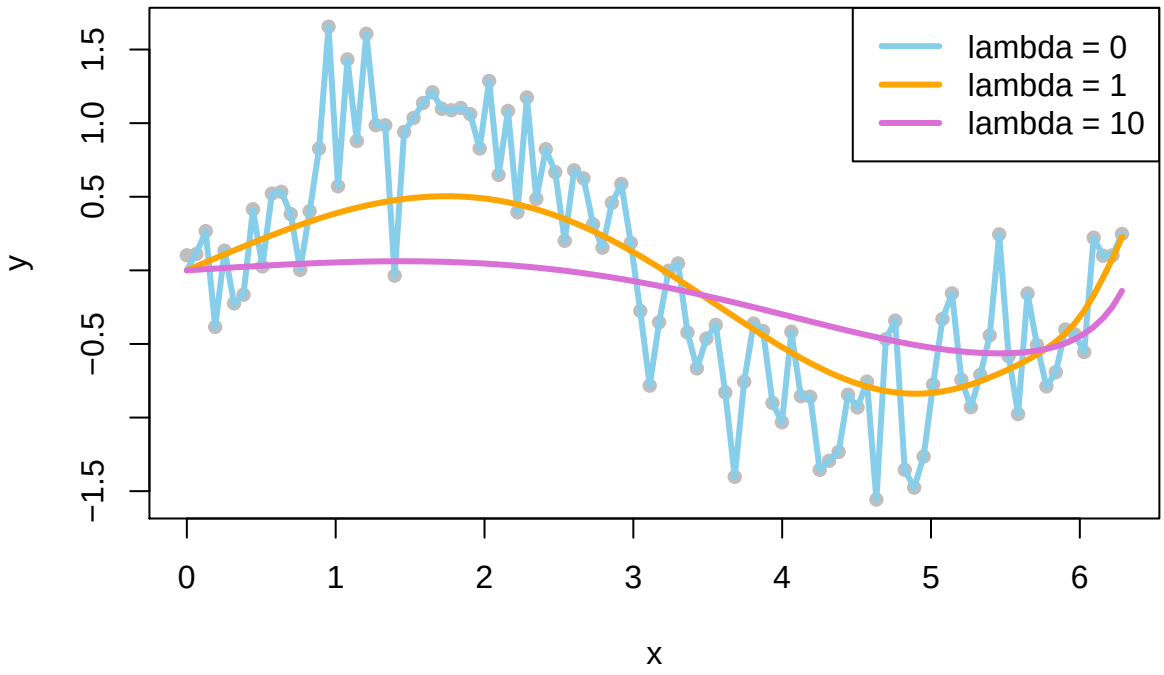
$$\begin{aligned} \partial[\|y - Xa\|^2 + \lambda a^T \Omega a]/\partial a &= 0 \\ 0 - 2X^T y + 2X^T X a + 2\lambda \Omega a &= 0 \\ (X^T X + \lambda \Omega)a &= X^T y \\ \hat{a} &= (X^T X + \lambda \Omega)^{-1} X^T y \end{aligned}$$

when design matrix X is fixed, prediction $\hat{y}$ can be represented by function of λ and y

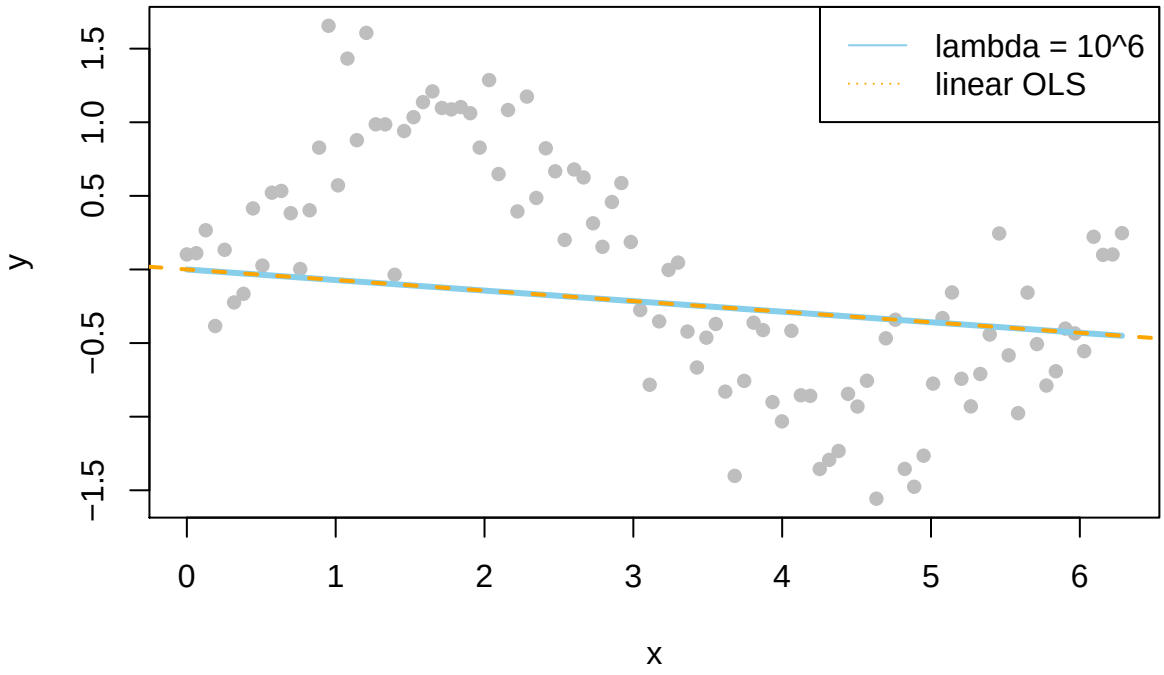
$$\begin{aligned} \hat{y} &= X\hat{a} \\ &= X(X^T X + \lambda \Omega)^{-1} X^T y \\ &= S_\lambda y \end{aligned}$$

(b)

```
train_size <- length(x)
X <- truncated.power.design.matrix(x)

lambda_range <- c(0, 1, 10, 10^6)

rdg_fit <- glmnet(X, y, alpha=0, lambda=lambda_range, intercept=FALSE,
                  thresh =1e-12, maxit = 10^7, penalty.factor = c(0,rep(1,train_size-2),0))

yhat1 <- predict(rdg_fit, X, s=0)

yhat2 <- predict(rdg_fit, X, s=1)

yhat3 <- predict(rdg_fit, X, s=10)

plot(x, y, col='gray', pch=16)
lines(x, yhat1, lwd=3, col='skyblue')
lines(x, yhat2, lwd=3, col='orange')
lines(x, yhat3, lwd=3, col='orchid')

legend('topright', legend = c('lambda = 0', 'lambda = 1', 'lambda = 10'),
       col=c('skyblue', 'orange', 'orchid'), lty=1, lwd=3)
```



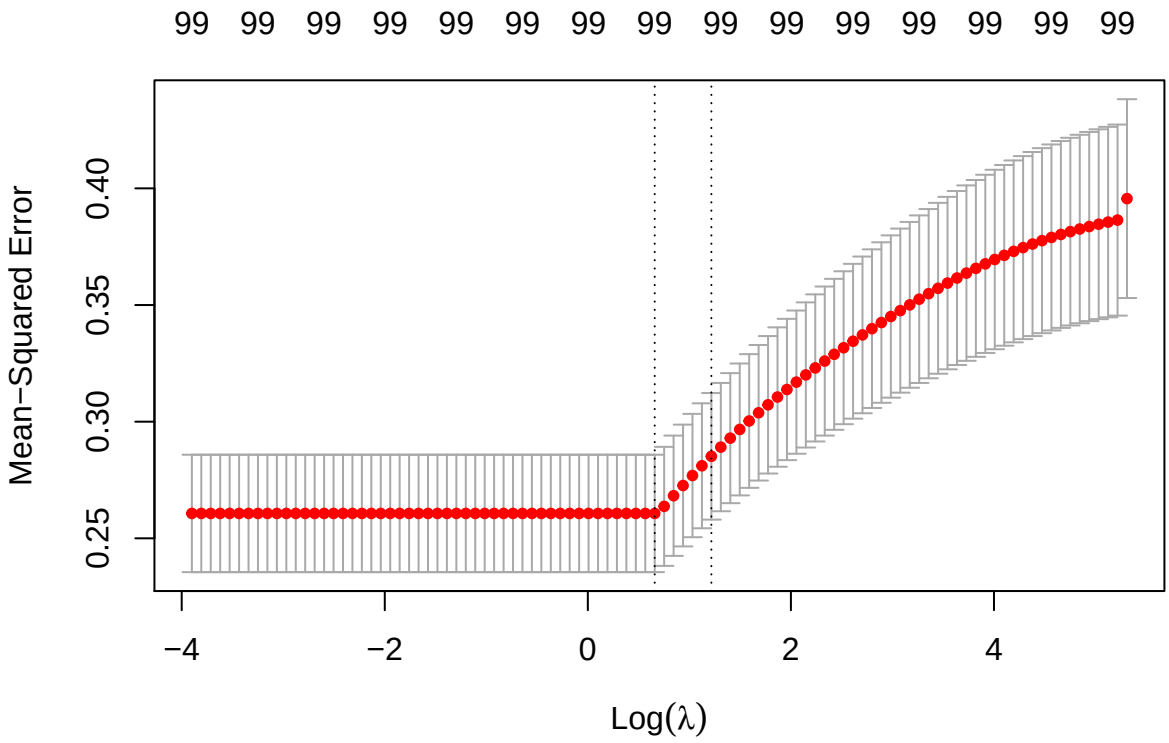

```
## verify lambda=10^6
## use OLS without intercept

yhat4 <- predict(rdg_fit, X, s=10^6)

plot(x, y, col='gray', pch=16)
lines(x, yhat4, lwd=3, col='skyblue')
abline(lm(y~0 + x), col='orange', lty=2, lwd=2)

legend('topright', legend=c('lambda = 10^6', 'linear OLS'),
       col=c('skyblue', 'orange'), lty = c(1, 3))
```



We can perceive that when $\lambda = 10^6$, the spline is similiar to linear OLS (without intercept).

(c)

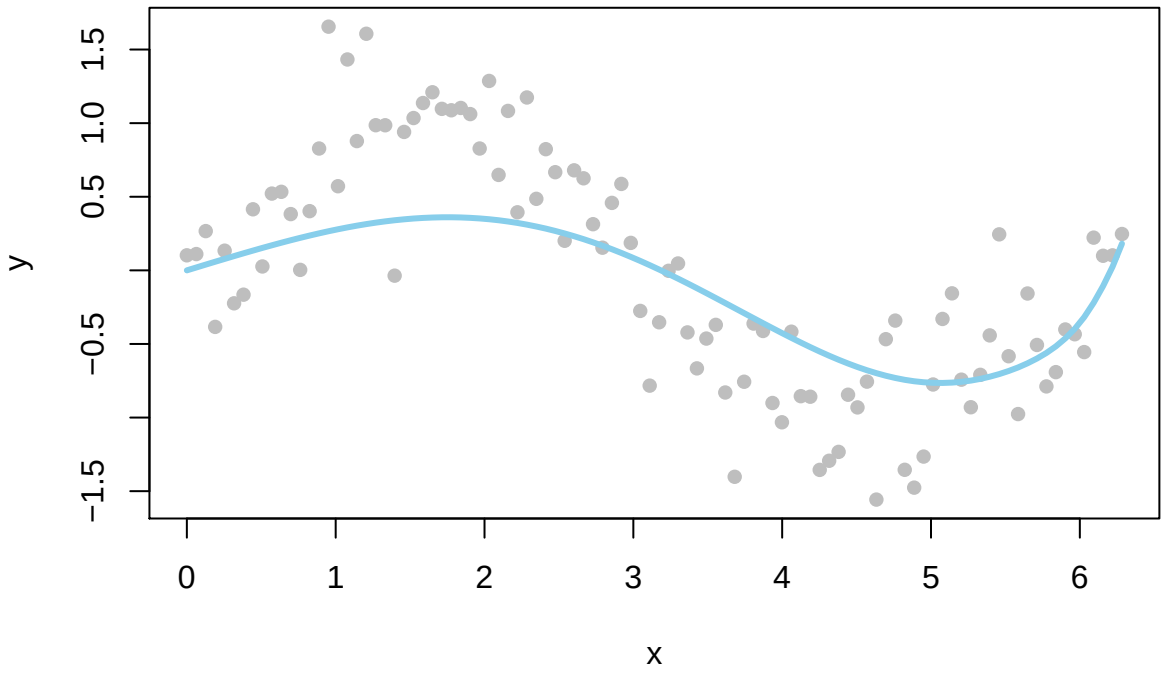
```
set.seed(123)

cv_fit <- cv.glmnet(X, y, alpha=0, thresh =1e-12, maxit = 10^7,
                    penalty.factor = c(0, rep(1, train_size - 2), 0))
plot(cv_fit)
```




```
lambda_opt <- cv_fit$lambda.min
print(lambda_opt)
```

```
## [1] 1.932104
```

```
rdg_fit_opt <- glmnet(X, y, alpha=0, lambda=lambda_opt, intercept=FALSE,
                      penalty.factor = c(0, rep(1, train_size - 2), 0))
yhat <- predict(rdg_fit_opt, X)

plot(x, y, col='gray', pch=16)
lines(x, yhat, lwd=3, col='skyblue')
```



Problem 3

(a)

In general, set of predictors chosen from best subset selection has smallest training RSS. Because both forward and backward selection leave some possible models without check, while best subset selection checked all possible models with k predictors.

(b)

In general, model selected by best subset perform better on testing data.

(c)

1. True, when choosing number (k + 1) predictor, previous selections are fixed.
2. True, definition of backward selection.
3. False, the set of (k + 1) predictors from backward selection is not necessarily same as from forward selection.
4. False, same as above.
5. False, best subset selection conducts exhaustive selection on all combinations of (k + 1) predictors. The selection is not necessarily related to best subset model having k predictors.

Problem 4

(a)

As λ increase from 0, training RSS will (iii) steadily increase.

(b)

As λ increase from 0, testing RSS will (ii) decrease initially and then increase, forming a U shape.

(c)

As λ increase from 0, variance will (iv) steadily decrease.

(d)

As λ increase from 0, variance will (iii) steadily increase.

(e)

As λ increase from 0, irreducible error will (v) remain constant, since irreducible error is independent with OLS model.