a3

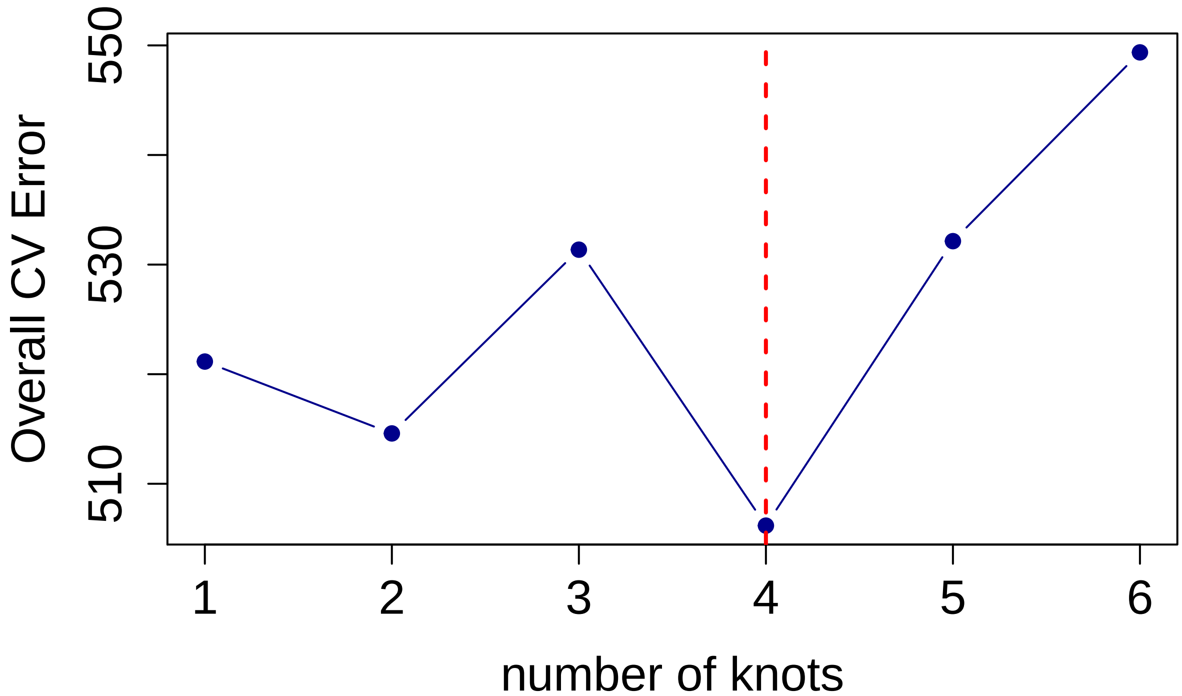
Mushi Wang

13/07/2020

1.(a)

```
ozone = read.table("ozone.txt", header = TRUE)
knots_num = c(1:6)
library(boot)
library(splines)
x = ozone$temperature
y = ozone$ozone
cv.err = vector("numeric", length(knots_num))
data = data.frame('x' = x, 'y' = y)
set.seed(444)
for(i in 1:length(knots_num)){
  knots = quantile(x, seq(1 / (i + 1), i / (i + 1), 1 / (i + 1)))
  glm.fit = glm(y ~ bs(x, degree = 3, knots = knots))
  cv.err[i] = cv.glm(data, glm.fit, K = 10)$delta[1]
}
```

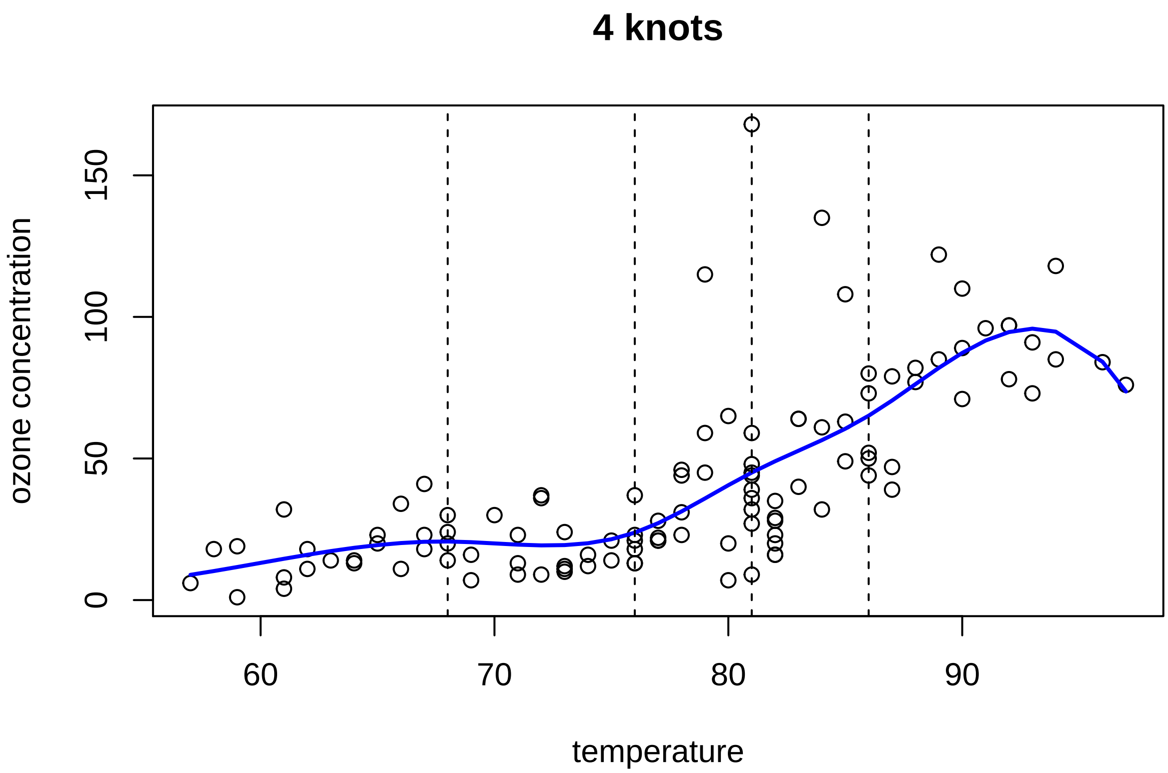
```
plot(knots_num, cv.err, pch=19, col="darkblue", type="b",
     cex.axis = 1.5, cex.lab=1.5, ylab="Overall CV Error", xlab = "number of knots")
indx = which.min(cv.err)
abline(v=indx, lty=2, lwd=2, col='red')
```



We chose the model with 4 knots.

```
plot(ozone$temperature, ozone$ozone, xlab = "temperature",
     ylab = "ozone concentration", main = "4 knots")

ozone_sorted = ozone[order(ozone$temperature),]
best_knots = quantile(ozone$temperature, seq(1 / 5, 4 / 5, 1 / 5))
lines(ozone_sorted$temperature,
      predict(lm(ozone_sorted$ozone ~ bs(ozone_sorted$temperature, degree= 3, knots = best_knots))),
      type="l", col="blue", lwd=2)
abline(v= best_knots, lty=2 )
```

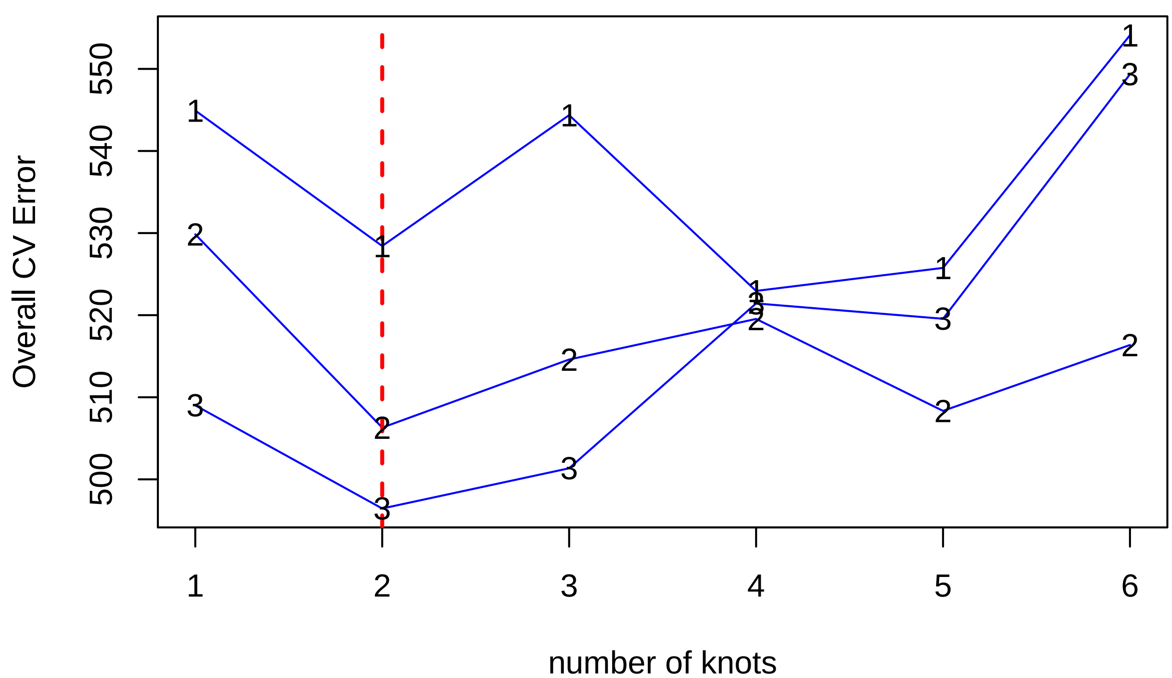

4 knots
ozone concentration
0
50
100
150
60
70
80
90
temperature

1.(b)

```
set.seed(444)
complexity = c(1:3)
cv.err2 = c()

for (j in 1:length(complexity)) {
  for(i in 1:length(knots_num)){
    knots = quantile(x, seq(1 / (i + 1), i / (i + 1), 1 / (i + 1)))
    glm.fit = glm(y ~ bs(x, degree = j, knots = knots))
    cv.err2 = append(cv.err2, cv.glm(data, glm.fit, K = 10)$delta[1])
  }
}
```

```
plot(rep(1:6, 3), cv.err2, cex=2, ylab="Overall CV Error", xlab = "number of knots",  type="n")
lines(1:6, cv.err2[1:6], type="l", col="blue", lwd=1)
lines(1:6, cv.err2[7:12], type="l", col="blue", lwd=1)
lines(1:6, cv.err2[13:18], type="l", col="blue", lwd=1)
indx = which.min(cv.err2)
abline(v=rep(1:6, 3)[indx], lty=2, lwd=2, col='red')
text(rep(1:6, 3), cv.err2, labels = c(rep(1,6), rep(2,6), rep(3,6)))
```

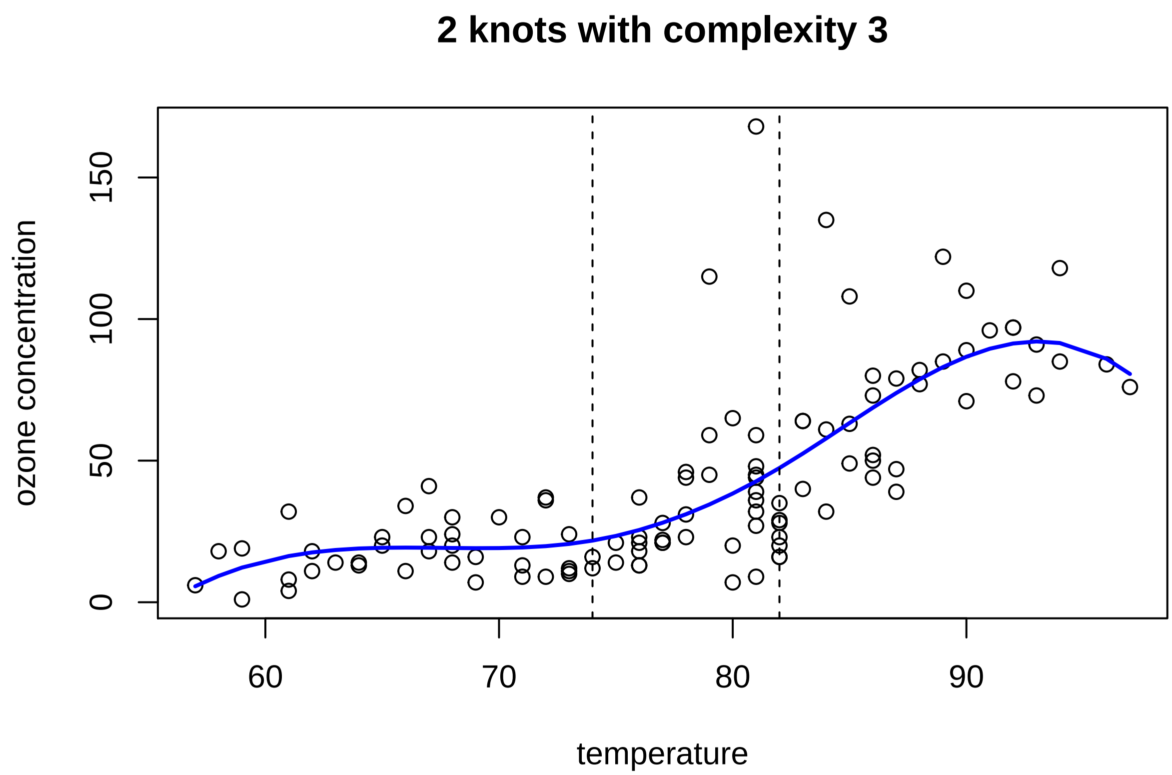


We choose the model with 2 knots and complexity 3

```
plot(ozone$temperature, ozone$ozone, xlab = "temperature",
     ylab = "ozone concentration", main = "2 knots with complexity 3")

best_knots = quantile(ozone$temperature, seq(1 / 3, 2 / 3, 1 / 3))
```

```r
lines(ozone_sorted$temperature,
      predict(lm(ozone_sorted$ozone ~ bs(ozone_sorted$temperature, degree= 3, knots = best_knots))),
      type="l", col="blue", lwd=2)
abline(v= best_knots, lty=2 )
```



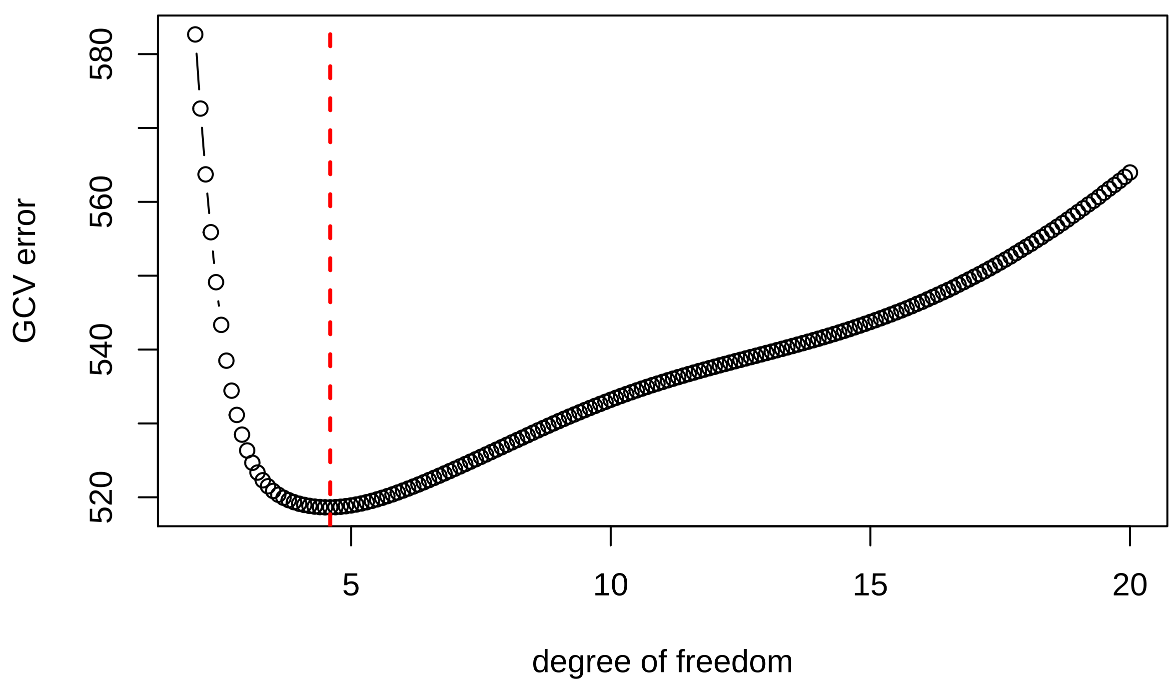
1.(c) Since the number of knots is $df - degree$. The model in (a) has $df = 7$ and the model in (b) has $df = 5$. Hence the model in (b) is smoother that the model in (a).

2.(a)

```
df = c()
error = c()

for(i in seq(2, 20, 0.1)) {
  sm.gcv = smooth.spline(x, y, df = i, cv = FALSE)
  error = append(error, sm.gcv$cv.crit)
  df = append(df, i)
}

plot(df, error, xlab = "degree of freedom", ylab = "GCV error", type = 'b')
indx = which.min(error)
abline(v=df[indx], lty=2, lwd=2, col='red')
```



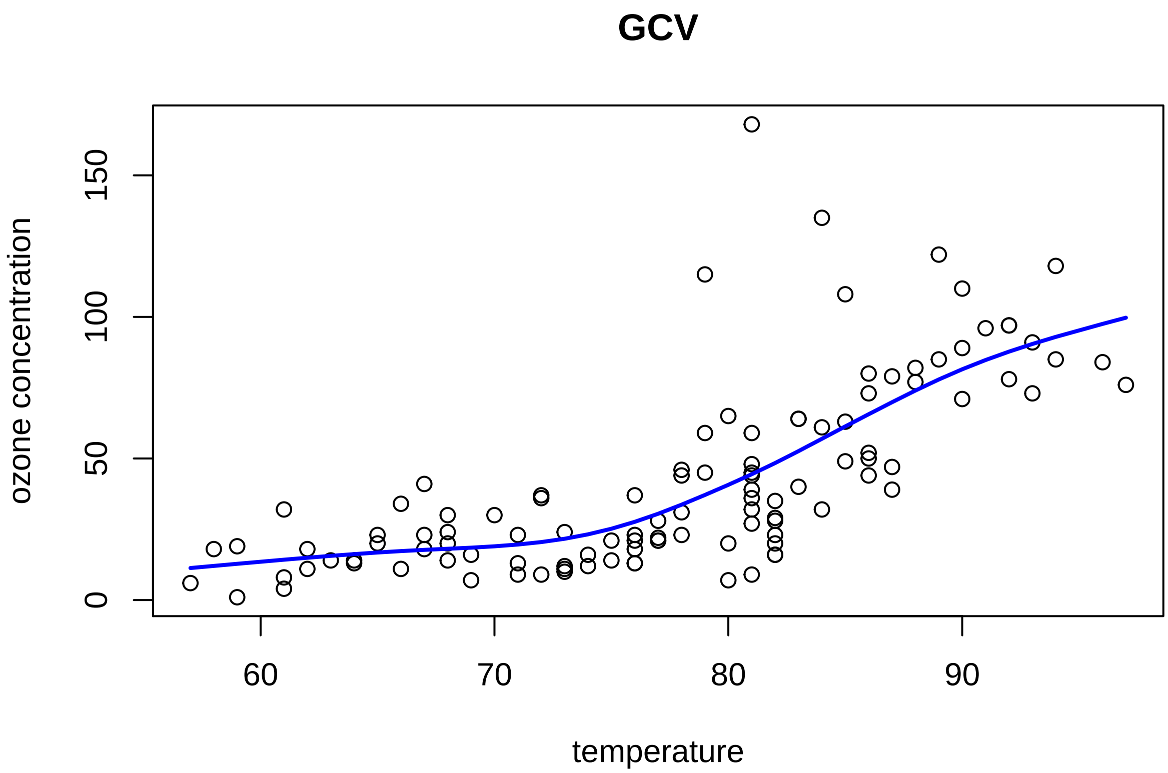

```
df[indx]
```

```
## [1] 4.6
```

We choose the model with degree of freedom 4.6.

```
plot(ozone$temperature, ozone$ozone, xlab = "temperature",
     ylab = "ozone concentration", main = "GCV")

lines(ozone_sorted$temperature,
      predict(smooth.spline(ozone_sorted$temperature, ozone_sorted$ozone,
                            df = 4.6, cv = FALSE), x = ozone_sorted$temperature)$y,
      type="l", col="blue", lwd=2)
```

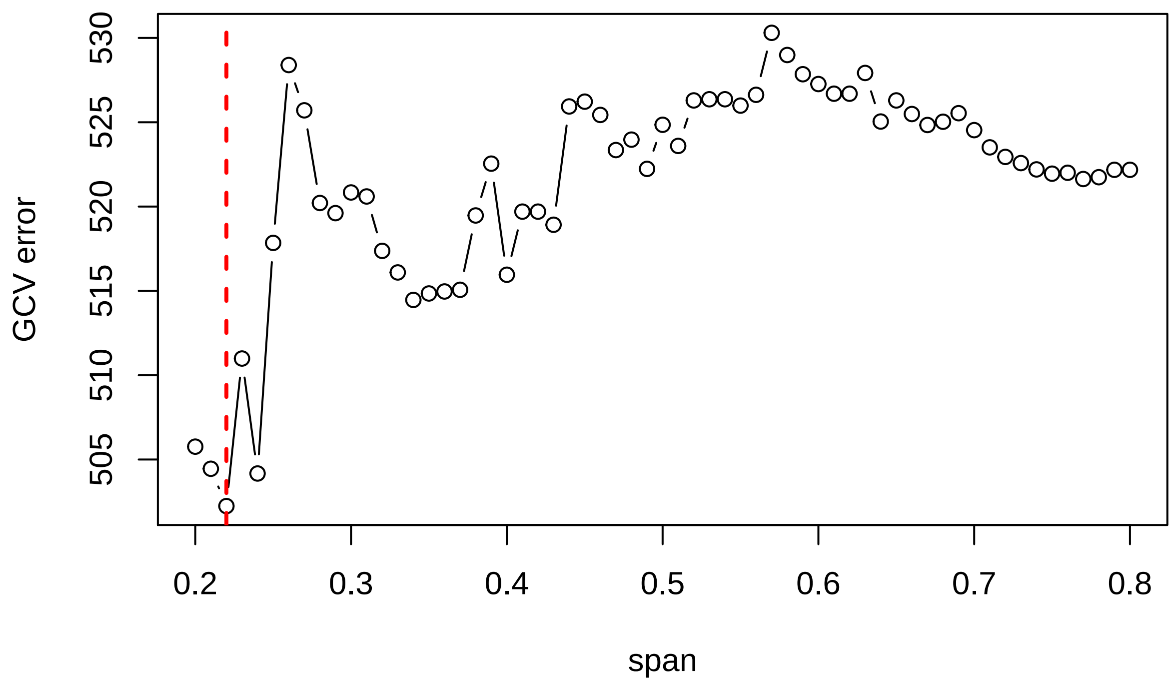

GCV
ozone concentration
0
50
100
150
60
70
80
90
temperature

2.(b)

```
span = seq(0.2, 0.8, 0.01)
error = c()

for(i in span) {
  loess_err = loess(y ~ x, span = i)
  error = append(error,
                 sum((loess_err$residuals) ^ 2) /
                   (1 - loess_err$trace.hat / length(loess_err$residuals))^2 / length(x))
}

plot(span, error, xlab = "span", ylab = "GCV error", type = 'b')
indx = which.min(error)
abline(v=span[indx], lty=2, lwd=2, col='red')
```

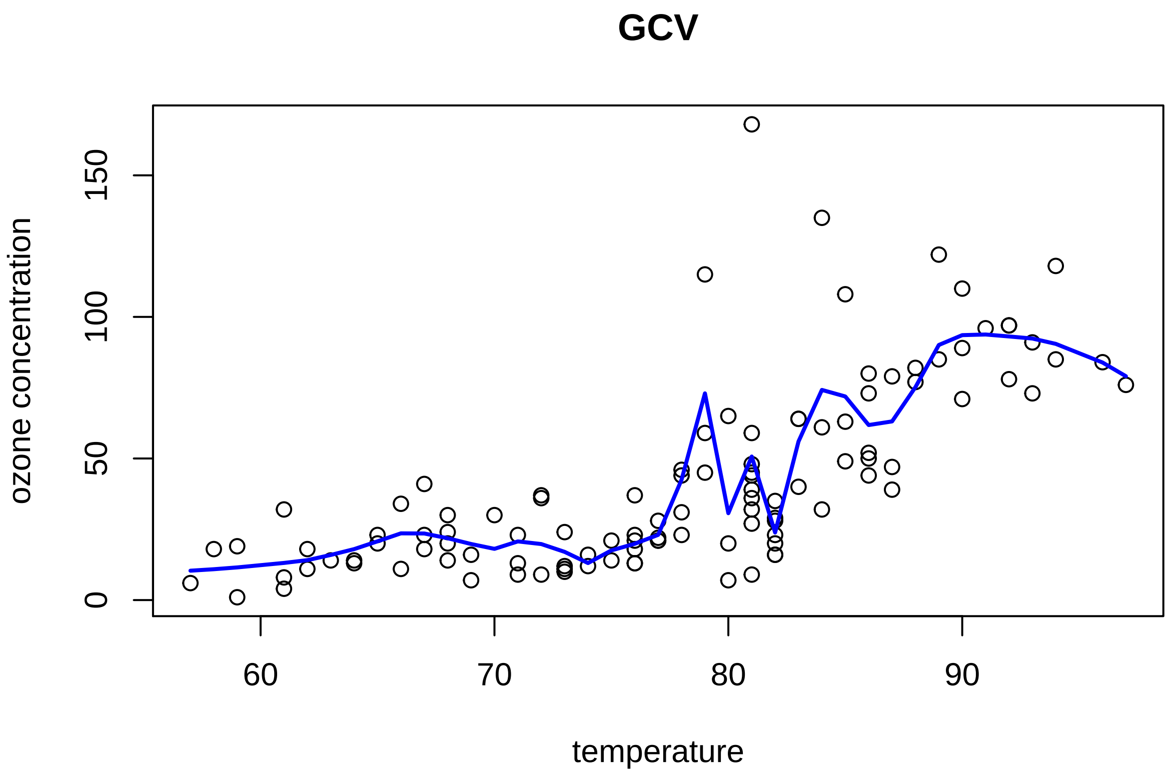



```
span[indx]
```

```
## [1] 0.22
```

```
plot(ozone$temperature, ozone$ozone, xlab = "temperature",
     ylab = "ozone concentration", main = "GCV")

lines(ozone_sorted$temperature,
      predict(loess(ozone_sorted$ozone ~ ozone_sorted$temperature, span = 0.22)),
      type="l", col="blue", lwd=2)
```

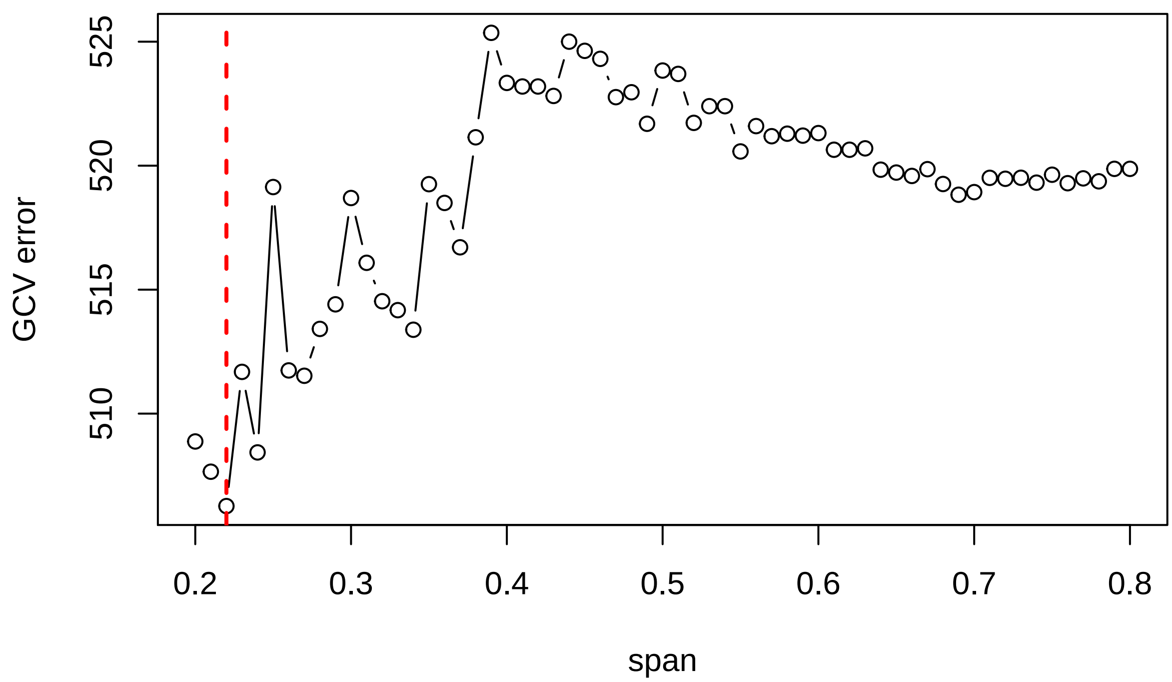

GCV
ozone concentration
0
50
100
150
60
70
80
90
temperature

2.(c)

```r
span = seq(0.2, 0.8, 0.01)
error = c()

for(i in span) {
  loess_err = loess(y ~ x, span = i, degree = 1)
  error = append(error,
                 sum((loess_err$residuals) ^ 2) /
                   (1 - loess_err$trace.hat / length(loess_err$residuals))^2 / length(x))
}

plot(span, error, xlab = "span", ylab = "GCV error", type = 'b')
indx = which.min(error)
abline(v=span[indx], lty=2, lwd=2, col='red')
```

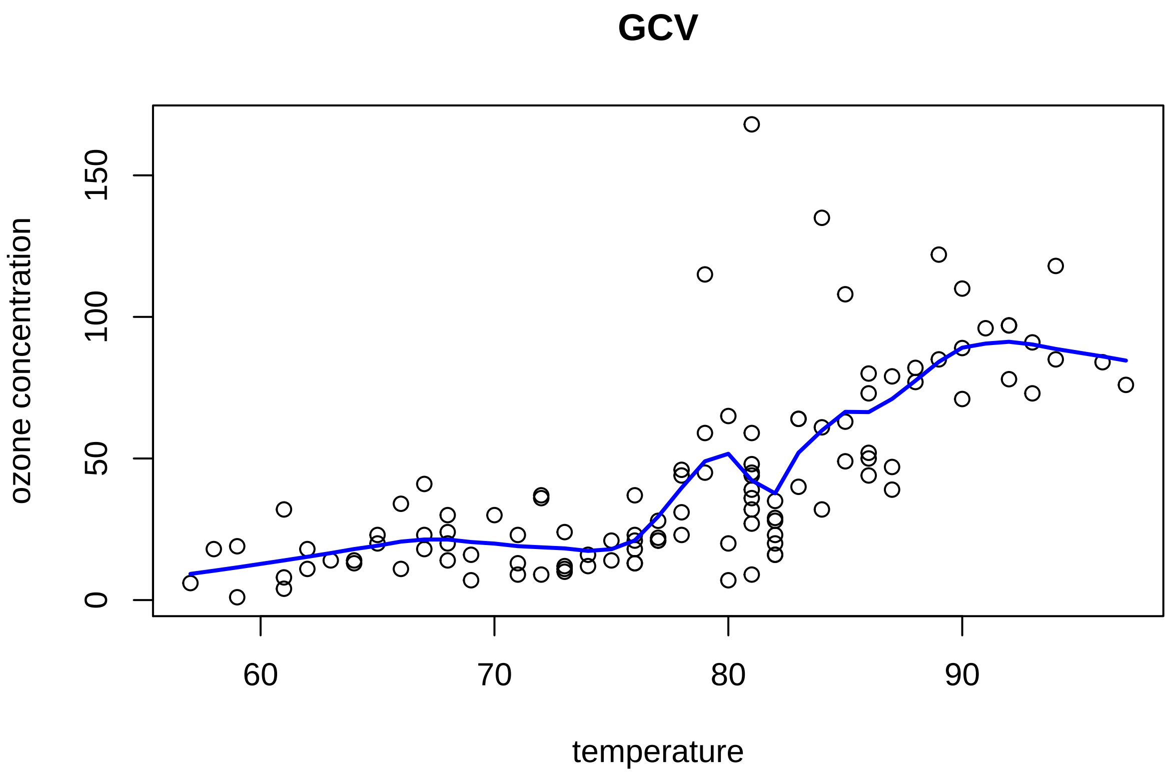



```r
span[indx]
```

```
## [1] 0.22
```

```r
plot(ozone$temperature, ozone$ozone, xlab = "temperature",
     ylab = "ozone concentration", main = "GCV")

lines(ozone_sorted$temperature,
      predict(loess(ozone_sorted$ozone ~ ozone_sorted$temperature, span = 0.22, degree = 1)),
      type="l", col="blue", lwd=2)
```


GCV
ozone concentration
0
50
100
150
60
70
80
90
temperature

2.(d)

```
#smooth.spline(ozone_sorted$temperature, ozone_sorted$ozone, df = 4.6, cv = FALSE) #a
model2b = loess(ozone_sorted$ozone ~ ozone_sorted$temperature, span = 0.22) #b
model2c = loess(ozone_sorted$ozone ~ ozone_sorted$temperature, span = 0.22, degree = 1) #c
summary(model2b)
```

```
## Call:
## loess(formula = ozone_sorted$ozone ~ ozone_sorted$temperature,
##     span = 0.22)
##
## Number of Observations: 111
## Equivalent Number of Parameters: 16.35
## Residual Standard Error: 20.7
## Trace of smoother matrix: 18.08  (exact)
##
## Control settings:
##   span     :  0.22
##   degree   :  2
##   family   :  gaussian
##   surface  :  interpolate      cell = 0.2
##   normalize:  TRUE
##  parametric:  FALSE
## drop.square:  FALSE
```

```
summary(model2c)
```

```
## Call:
## loess(formula = ozone_sorted$ozone ~ ozone_sorted$temperature,
##     span = 0.22, degree = 1)
##
## Number of Observations: 111
## Equivalent Number of Parameters: 8.47
## Residual Standard Error: 21.63
## Trace of smoother matrix: 10.06  (exact)
##
## Control settings:
##   span     :  0.22
##   degree   :  1
##   family   :  gaussian
##   surface  :  interpolate      cell = 0.2
##   normalize:  TRUE
##  parametric:  FALSE
## drop.square:  FALSE
```

The degree of freedom of model in part (a) is 4.6. From summary, the degree of freedom of model in part (b) is 16.35 and the degree of freedom of model in part (c) is 8.47. The model in part (a) is the smoothest, the model in part (c) is the second smoothest and the model in part (b) is the roughest.

3.(a)

```r
LongSpanModel <- loess(ozone$ozone ~ ozone$temperature , span=0.8)
ShortSpanModel <- loess(ozone$ozone ~ ozone$temperature , span=0.2)
```

```r
smootherMatrixLoess <- function(x,
                                span=NULL,
                                enp.target=NULL,
                                ...) {
  n <- length(x)
  S <- matrix(0, n, n)
  for (i in 1:n) {
    ei = rep(0, n)
    ei[i] <- 1
    # insert the fit into the i'th column of S
    if (is.null(span) & is.null(enp.target))
    {
      S[,i]  <- predict(loess(ei ~ x, ...))
    } else {
      if (is.null(span)) {
        S[,i]  <- predict(loess(ei ~ x,
                                enp.target=enp.target,
                                ...))

      } else {
        S[,i]  <- predict(loess(ei ~ x,
                                span=span,
                                ...))

      }

    }

  }
  # For loess, the smoother matrix need
  # not be a symmetric matrix
  S
}
```

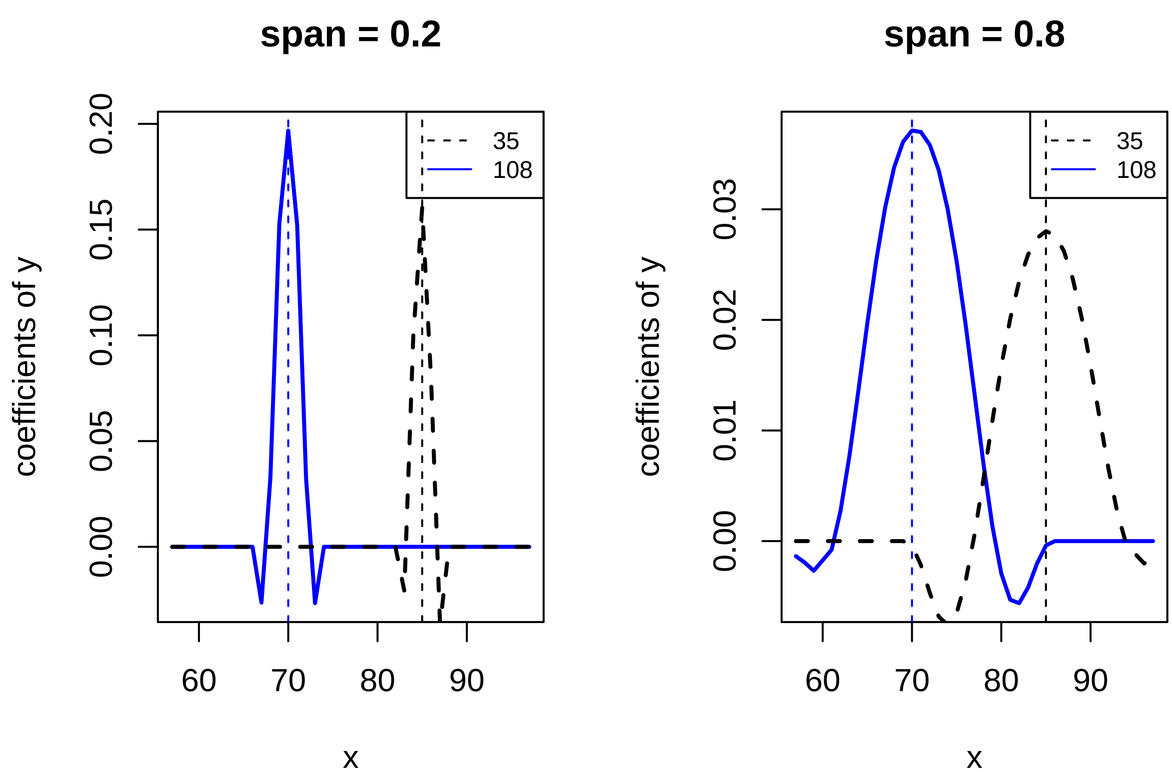
(i)

```r
s_short = smootherMatrixLoess(ozone$temperature, span = 0.2)
s_long = smootherMatrixLoess(ozone$temperature, span = 0.8)

par(mfrow=c(1,2))
coeff_y_short = data.frame(x = ozone$temperature, s_35 = s_short[35,], s_108 = s_short[108,])
coeff_y_short = coeff_y_short[order(coeff_y_short$x),,]
plot(coeff_y_short$x, coeff_y_short$s_108, lwd = 2, col = "blue", type = 'l',
     main = "span = 0.2", xlab = "x", ylab = "coefficients of y")
lines(coeff_y_short$x, coeff_y_short$s_35, lwd = 2, lty = 2)
abline(v = ozone$temperature[35], lty=2)
abline(v = ozone$temperature[108], lty=2, col = "blue")
legend("topright", legend = c("35", "108"), col = c("black", "blue"), lty = c(2, 1), cex = 0.75)

coeff_y_long = data.frame(x = ozone$temperature, s_35 = s_long[35,], s_108 = s_long[108,])
```

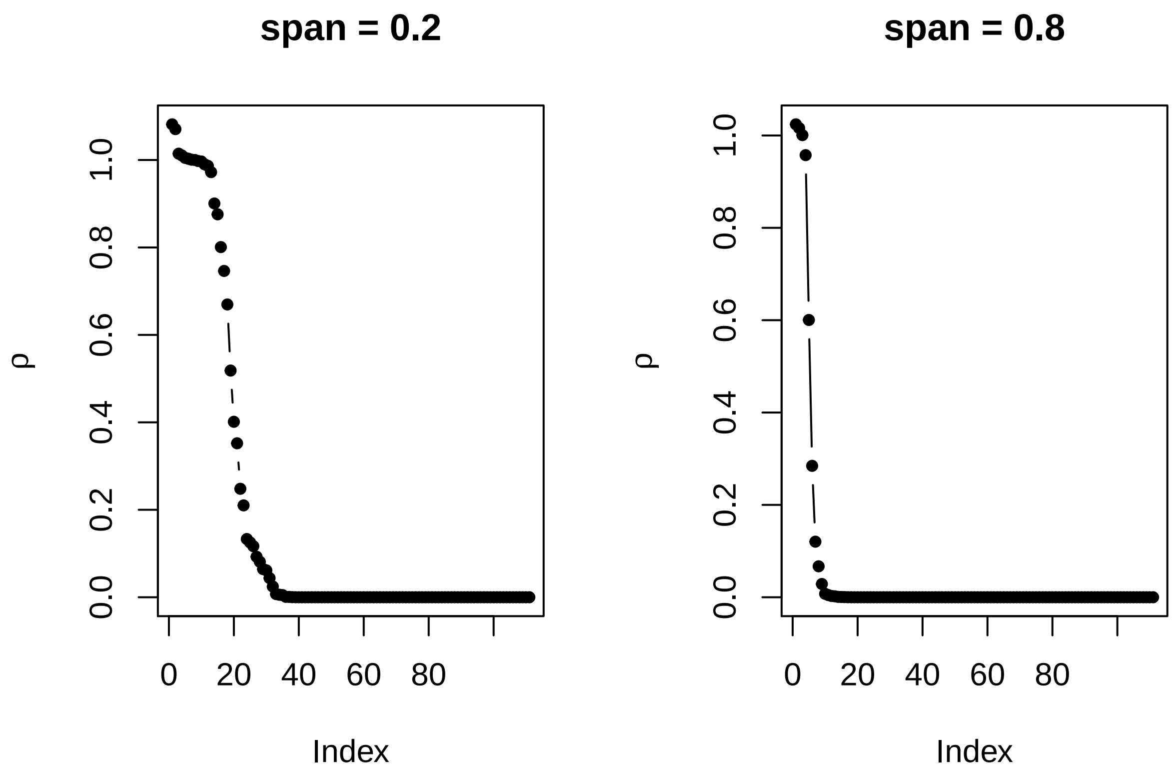
```r
coeff_y_long = coeff_y_long[order(coeff_y_long$x),,]
plot(coeff_y_long$x, coeff_y_long$s_108, lwd = 2, col = "blue", type = 'l',
     main = "span = 0.8", xlab = "x", ylab = "coefficients of y")
lines(coeff_y_long$x, coeff_y_long$s_35, lwd = 2, lty = 2)
abline(v = ozone$temperature[35], lty=2)
abline(v = ozone$temperature[108], lty=2, col = "blue")
legend("topright", legend = c("35", "108"), col = c("black", "blue"), lty = c(2, 1), cex = 0.75)
```



(ii)

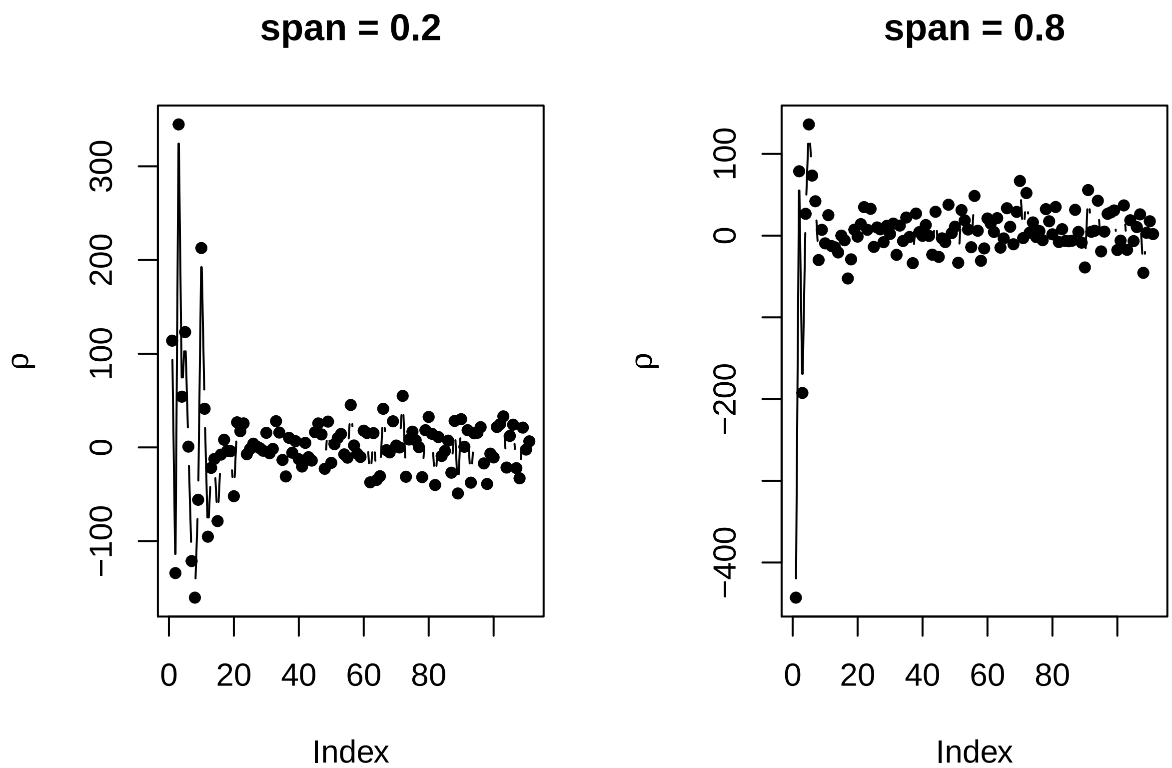
```r
par(mfrow=c(1,2))
svd_short = svd(s_short)
plot(svd_short$d, type = 'b', cex = 0.7, pch = 19, ylab = expression(rho), main = "span = 0.2")

svd_long = svd(s_long)
plot(svd_long$d, type = 'b', cex = 0.7, pch = 19, ylab = expression(rho), main = "span = 0.8")
```



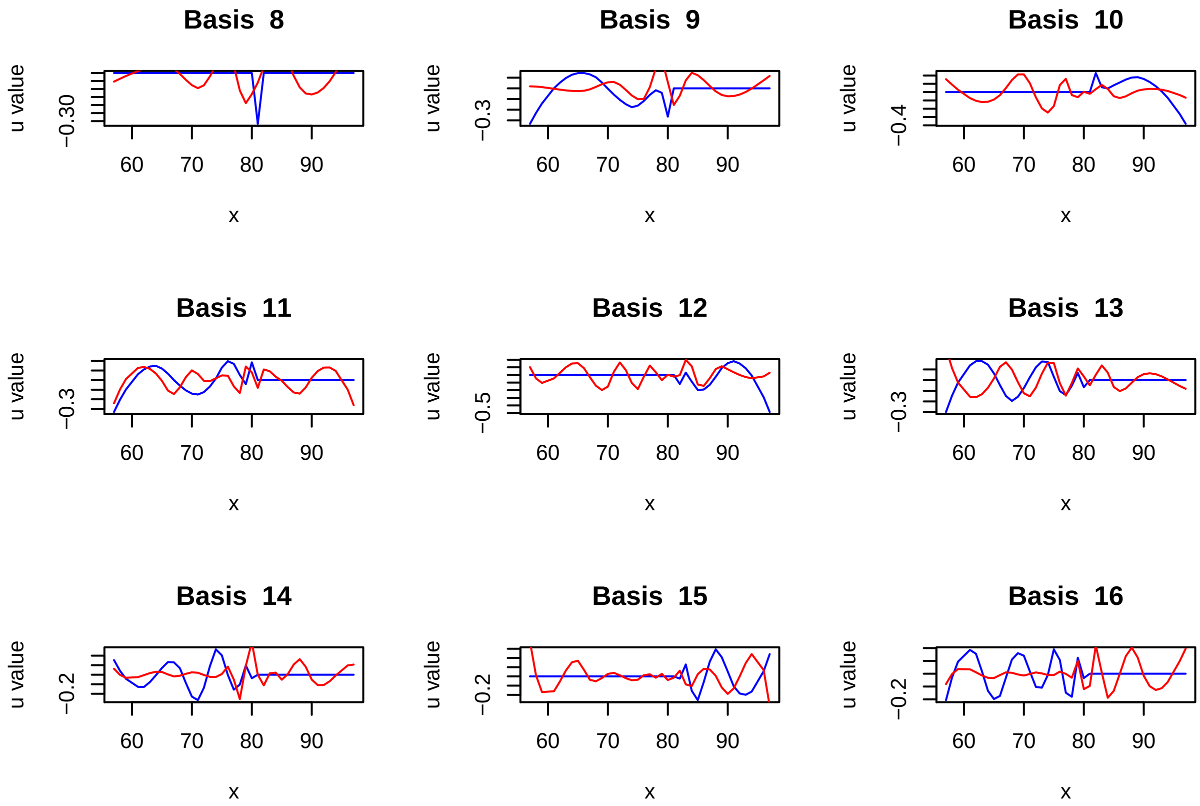
(iii)

```
par(mfrow=c(1,2))
plot(t(svd_short$v) %*% ozone$ozone, type = 'b', cex = 0.7, pch = 19, ylab = expression(rho), main = "sp
plot(t(svd_long$v) %*% ozone$ozone, type = 'b', cex = 0.7, pch = 19, ylab = expression(rho), main = "spa
```



(iv)

```
par(mfrow=c(3, 3))
plot_fun = function() {
  for(i in 8:16) {
    basis = data.frame(x = ozone$temperature, short = svd_short$u[,i], long = svd_long$u[,i])
    basis = basis[order(x),,]
    plot(basis$x, basis$short, type = 'l', xlab = "x", ylab = "u value", main = paste("Basis ", i), col
    lines(basis$x, basis$long, col = "red")
  }
}
plot_fun()
```



The red lines correspond to S_{long} and the blue lines correspond to S_{short}.

3.(b) (i)
Similarities: Coefficients of y decreases when x moves further away from x = ozone$temperature[35] andx = ozone$temperature[108] in both graphs. Since as x moves further away, the weights are reducing.
Differences: The coefficients of y have more zeros when span = 0.2 than span = 0.8. Since for span = 0.2, the model is more local that is fewer points "contribute" in estimation.
(ii)
Similarities: Both graphs have an elbow shaped curve. Majorities of the singular values are close to 0. Only a few singular values affect the result of estimations $\hat{\mu}$. Since we only need much fewer basis than the sample size to estimate.
Differences: There are more non-zero points when span is 0.2 than when span is 0.8. Since the model with span = 0.8 is smoother than the model with span = 0.2.
(iii)
Similarities: Majority of points in both graphs are very close to 0.
Differences: There are more non-zero points when span is 0.2 than when span is 0.8. Since the model with span = 0.8 is smoother than the model with span = 0.2.

3.(c) Both of them become wigglier as the basis increase. But for a certain basis, S_{long} is wigglier than S_{short} and S_{long} has less value close to 0 than S_{short}.

```
svd_short$d[8:16]
```

```
## [1] 1.0000000 0.9977208 0.9966038 0.9901450 0.9866578 0.9722642 0.9005572
## [8] 0.8757600 0.8008420
```

```
svd_long$d[8:16]
```

```
## [1] 0.0669977503 0.0286712951 0.0074513446 0.0046619054 0.0026958944
## [6] 0.0019868360 0.0008591393 0.0005437607 0.0003108313
```

S_{short} gives a higher weight to these basis directions.