Modeling Ozone data with Splines

```r
data <- read.table("ozone.txt", header = T)
head(data)
```

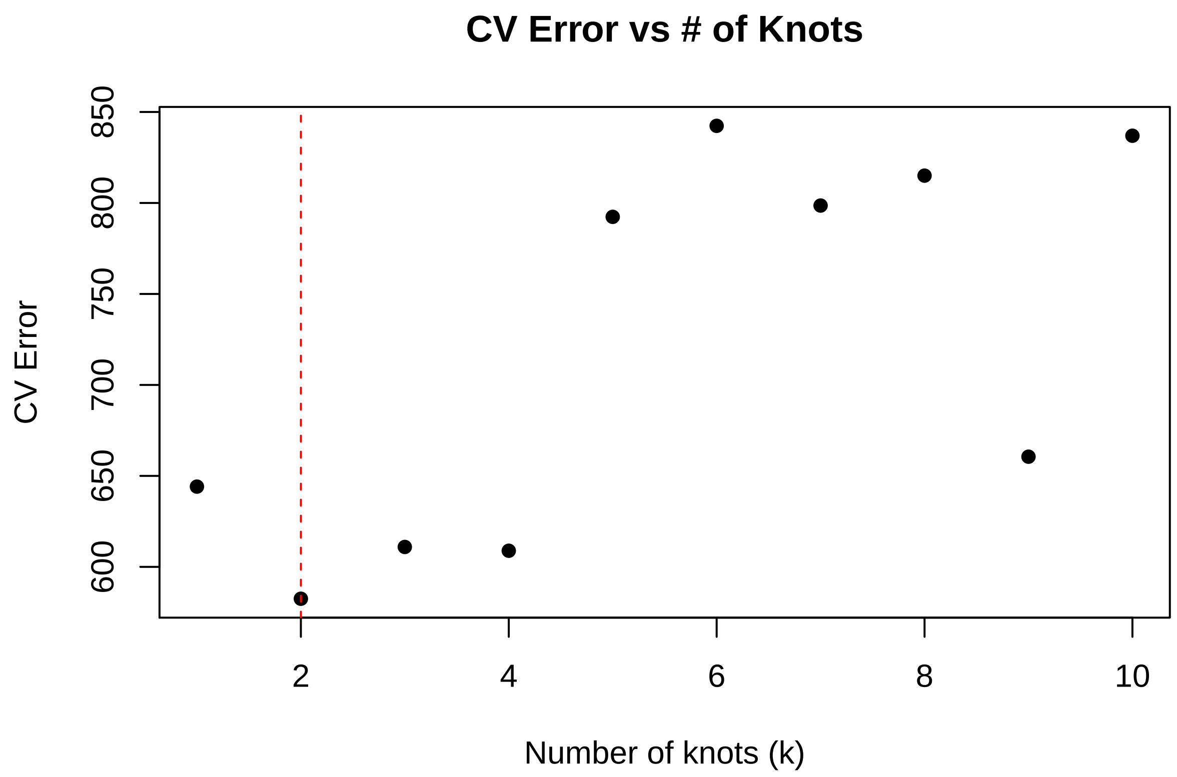
```
##   ozone radiation temperature wind
## 1    41       190          67  7.4
## 2    36       118          72  8.0
## 3    12       149          74 12.6
## 4    18       313          62 11.5
## 5    23       299          65  8.6
## 6    19        99          59 13.8
```

The dataset ozone contains 111 measurements taken daily from May until September 1973 in New York on 4 variates. Two of these variables are ozone: the ozone concentration in parts per billion and wind: the average wind speed in miles per hour. I will be modeling ozone as a function of wind.

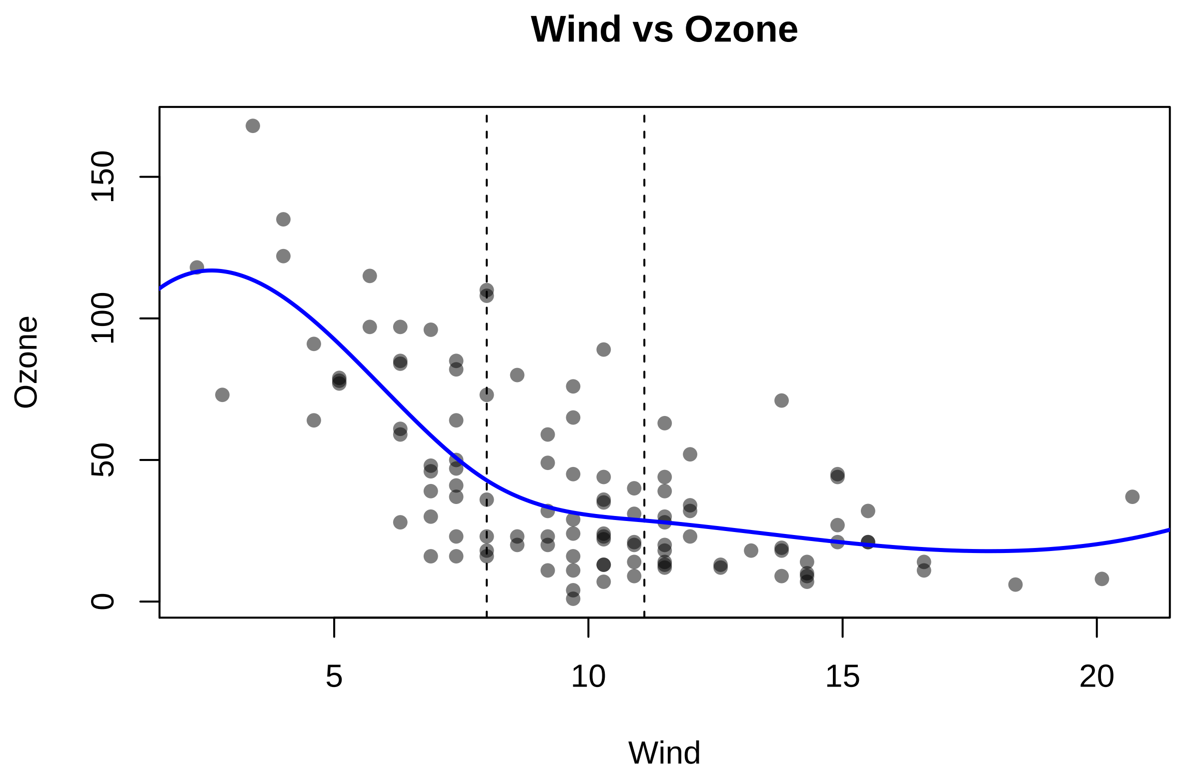
Cross validation to choose number of knots for B-splines

We get that the optimal number of knots is 2, after using 10-fold cross validation with MSE as the loss function to select the optimal number of knots,

```r
library(splines)
set.seed(444)

ozone <- data$ozone
wind <- data$wind

kfold <- function(N, k=N, indices=NULL){
  # get the parameters right:
  if (is.null(indices)) {
    # Randomize if the index order is not supplied
    indices <- sample(1:N, N, replace=FALSE)
  } else {
    # else if supplied, force N to match its length
    N <- length(indices)
  }
  # Check that the k value makes sense.
  if (k > N) stop("k must not exceed N")
  #

  # How big is each group?
  gsize <- rep(round(N/k), k)

  # For how many groups do we need odjust the size?
  extra <- N - sum(gsize)

  # Do we have too few in some groups?
  if (extra > 0) {
    for (i in 1:extra) {
      gsize[i] <- gsize[i] +1
    }
  }
  # Or do we have too many in some groups?
  if (extra < 0) {
    for (i in 1:abs(extra)) {
      gsize[i] <- gsize[i] - 1
    }
  }

  running_total <- c(0,cumsum(gsize))

  # Return the list of k groups of indices
  lapply(1:k,
         FUN=function(i) {
           indices[seq(from = 1 + running_total[i],
                       to = running_total[i+1],
                       by = 1)
                   ]
         }
  )
}

getKfoldSamples <- function (x, y, k, indices=NULL){
  groups <- kfold(length(x), k, indices)
  Ssamples <- lapply(groups,
                     FUN=function(group) {
                       list(x=x[-group], y=y[-group])
                     })
  Tsamples <- lapply(groups,
                     FUN=function(group) {
                       list(x=x[group], y=y[group])
                     })
  list(Ssamples = Ssamples, Tsamples = Tsamples)
}

samples_10fold = getKfoldSamples(wind, ozone, 10)

Ssamples <- samples_10fold$Ssamples
Tsamples <- samples_10fold$Tsamples

cv.error <- c()


for(k in 1:10) {
  knots_k <- quantile(wind, seq(0, 1, length=(k+2)))
  knots_k = knots_k[-c(1,length(knots_k))]
  cv.error.k = c()
  for(folds in 1:10) {
    fit.bs <- lm(y ~ bs(x, knots=knots_k,degree = 3), data = Ssamples[[folds]])
    ypred.bs <- predict(fit.bs, newdata = data.frame(x = Tsamples[[folds]]$x))
    cv.error.k[folds] = mean((Tsamples[[folds]]$y-ypred.bs)^2)
  }
  cv.error[k] = mean(cv.error.k)

}



plot(cv.error, pch = 16, xlab = "Number of knots (k)",
     ylab = "CV Error", main = "CV Error vs # of Knots")

best_k <- which.min(cv.error)

abline(v=best_k, col = "red", lty = 2)
```



```r
knots_cv <- quantile(wind, seq(0,1,length = (best_k + 2)))

knots_cv = knots_cv[-c(1,length(knots_cv))]

fit.bs <- lm(ozone ~ bs(wind, knots = knots_cv), data = data)

xrange <- extendrange(wind)
xnew <- seq(min(xrange), max(xrange), length.out=500)
ypred.bs <- predict(fit.bs, newdata = data.frame(wind = xnew))

plot(wind,ozone, pch =16, col =adjustcolor("black", 0.5),
     xlab = "Wind", ylab = "Ozone", main = "Wind vs Ozone",
     lim = xrange)

abline(v=knots_cv, lty = 2)
lines(xnew, ypred.bs, col="blue", lwd = 2)
```



We can see here the degree 3 B-spline model with 2 knots superimposed on the data.

Cross validation to choose number of knots and degree

Now we will use 10-fold cross validation with MSE as the loss function to choose the number of knots along with the degree of the B-splines.

```r
cv.error1 <- c()
cv.error2 <- c()

for(k in 1:10) {
  knots_k <- quantile(wind, seq(0, 1, length=(k+2)))
  knots_k = knots_k[-c(1,length(knots_k))]
  cv.error1.k = c()
  for(folds in 1:10) {
    fit1.bs <- lm(y ~ bs(x, knots=knots_k, degree = 1), data = Ssamples[[folds]])
    ypred1.bs <- predict(fit1.bs, newdata = data.frame(x = Tsamples[[folds]]$x))
    cv.error1.k[folds] = mean((Tsamples[[folds]]$y-ypred1.bs)^2)

  }
      cv.error1[k] = mean(cv.error1.k)


}


for(k in 1:10) {
  knots_k <- quantile(wind, seq(0, 1, length=(k+2)))
  knots_k = knots_k[-c(1,length(knots_k))]
  cv.error2.k = c()
  for(folds in 1:10) {
    fit2.bs <- lm(y ~ bs(x, knots=knots_k, degree = 2), data = Ssamples[[folds]])
    ypred2.bs <- predict(fit2.bs, newdata = data.frame(x = Tsamples[[folds]]$x))
    cv.error2.k[folds] = mean((Tsamples[[folds]]$y-ypred2.bs)^2)

  }
      cv.error2[k] = mean(cv.error2.k)


}

new <- c(cv.error1, cv.error2, cv.error)

plot(cv.error1, pch = "1", xlab = "Number of knots", ylim = c(400,850))
points(cv.error2, pch = "2")
points(cv.error, pch = "3")


best_k1 <- min(cv.error1)
best_k2 <- min(cv.error2)
best_k3 <- min(cv.error)

best_k1
```

```
## [1] 551.3952
```

```
best_k2
```

```
## [1] 571.7163
```

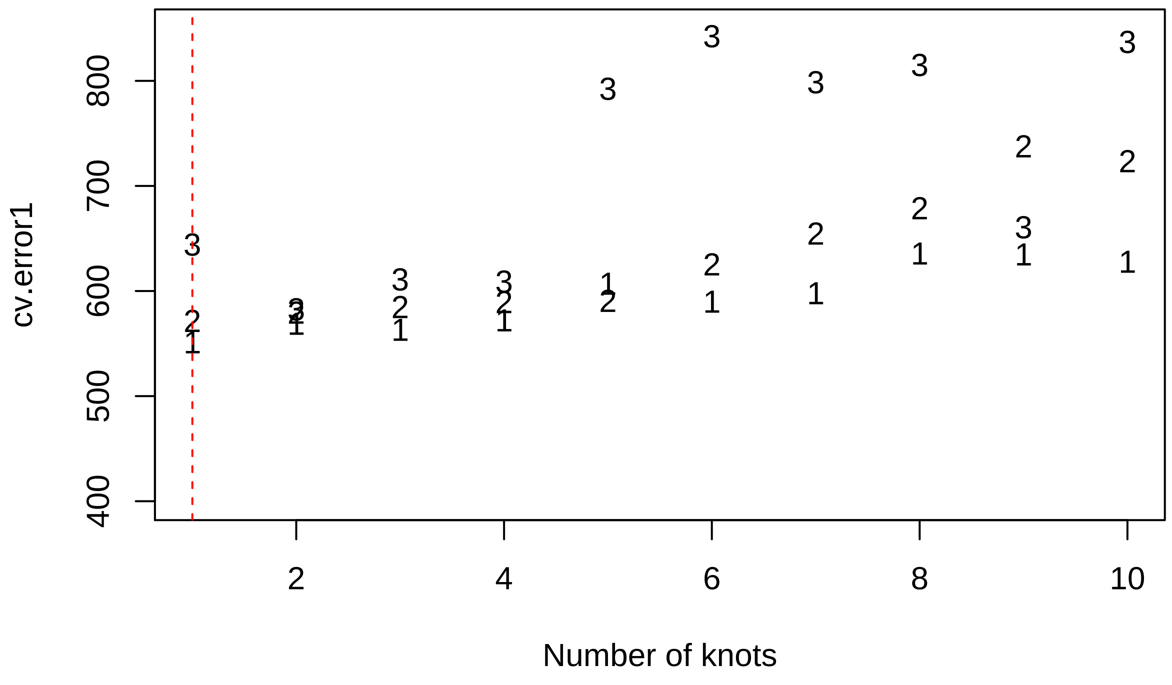
```
best_k3
```

```
## [1] 582.4962
```

```
bestk1 <- which.min(cv.error1)
bestk1
```

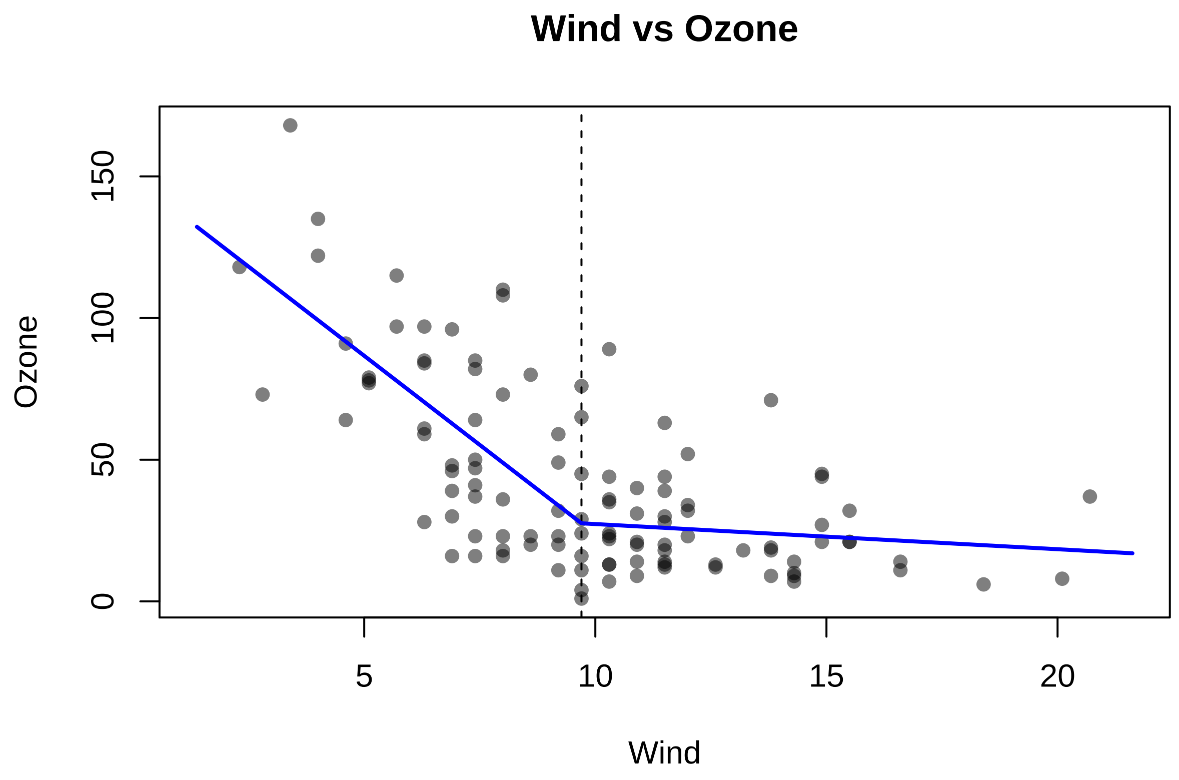
```
## [1] 1
```

```
abline(v=bestk1, col = "red", lty = 2)
```



```
knots_cv <- quantile(wind, seq(0,1,length = (bestk1 + 2)))

knots_cv = knots_cv[-c(1,length(knots_cv))]

fit1.bs <- lm(ozone ~ bs(wind, knots = knots_cv,degree=1), data = data)

xrange <- extendrange(wind)
xnew <- seq(min(xrange), max(xrange), length.out=500)
```

```r
ypred.bs <- predict(fit1.bs, newdata = data.frame(wind = xnew))

plot(wind,ozone, pch =16, col =adjustcolor("black", 0.5),
     xlab = "Wind", ylab = "Ozone", main = "Wind vs Ozone",
     xlim = xrange)

abline(v=knots_cv, lty = 2)
lines(xnew, ypred.bs, col="blue", lwd = 2)
```



We end up with a model that has 1 knot and a degree of 1.

The first model has 2 knots and degree of 3. Thus, the degrees of freedom is $4 + 2 = 6$. The cv error is 582.5.

The second model has 1 knot and degree of 1. Thus the degrees of freedom is $2 + 1 = 3$. The cv error is 551.3952

Given the degrees of freedom and cross-validation results, it would seem that the model in b is the best since it is less complex and has lower cv error.

We then consider modeling the data using smoothing splines. We use generalized cross validation to choose the effective degrees of freedom, and plot the gcv error vs degrees of freedom.

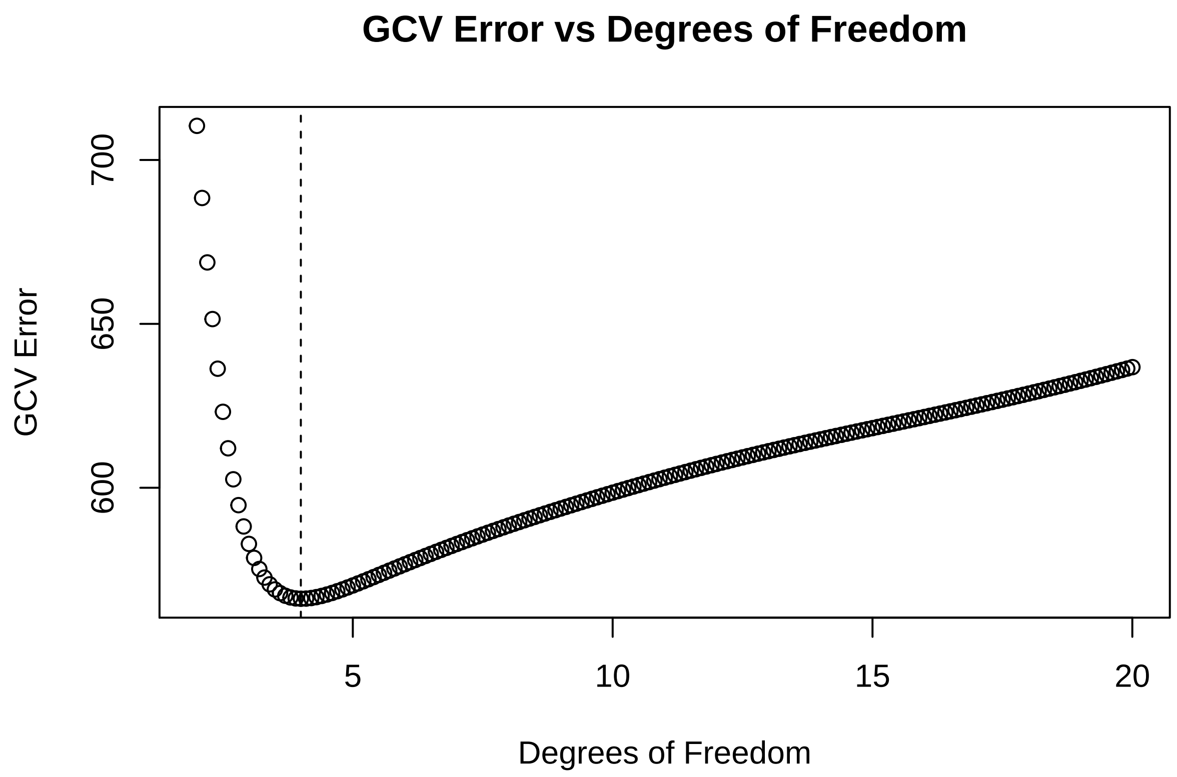
```r
data <- read.table("ozone.txt", header = T)

library(splines)

df <- seq(from= 2.0 , to= 20.0, by =0.1)
sm <- smooth.spline(data$wind, data$ozone, df = df)
x<- data$wind
y <- data$ozone

cv.error <- c()
cv.df <- c()
## Generalized cross-validation

for (i in 1:length(df)) {
sm.GCV <- smooth.spline(x, y,df=df[i], cv = FALSE)

cv.error[i] <- sm.GCV$cv
cv.df[i] <- sm.GCV$df
}

plot(cv.df, cv.error, main = "GCV Error vs Degrees of Freedom", xlab = "Degrees of Freedom",
     ylab = "GCV Error")

df[which.min(cv.error)]
```

```
## [1] 4
```

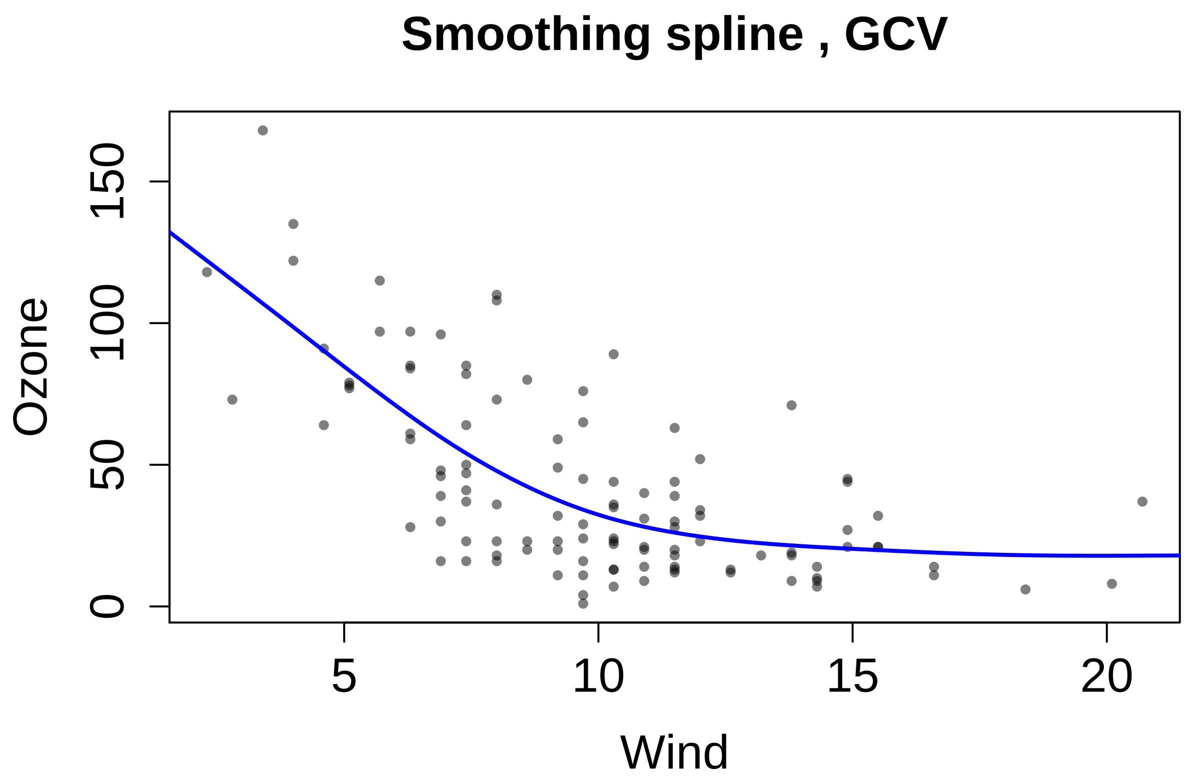
```r
cv.df[which.min(cv.error)]
```

```
## [1] 4.000597
```

```r
sm.GCV <- smooth.spline(x, y,df=df[i], cv = FALSE)

abline(v = df[which.min(cv.error)], lty =  2)
```



```r
sm1.GCV <- smooth.spline(data$wind, data$ozone, df = 4.0)

xrange <- extendrange(x)
xnew <- seq(min(xrange), max(xrange), length.out=500)

plot(x,y,
     pch=16, cex=0.7,
     main = "Smoothing spline , GCV",
     cex.axis=1.5 , cex.main=1.5, cex.lab=1.5,, xlab = "Wind", ylab = "Ozone",
     col=adjustcolor("black" , 0.5)
)
ypred.sm1.GCV <- predict(sm1.GCV, x=xnew)$y
lines(xnew, ypred.sm1.GCV, col="blue", lwd=2)
```



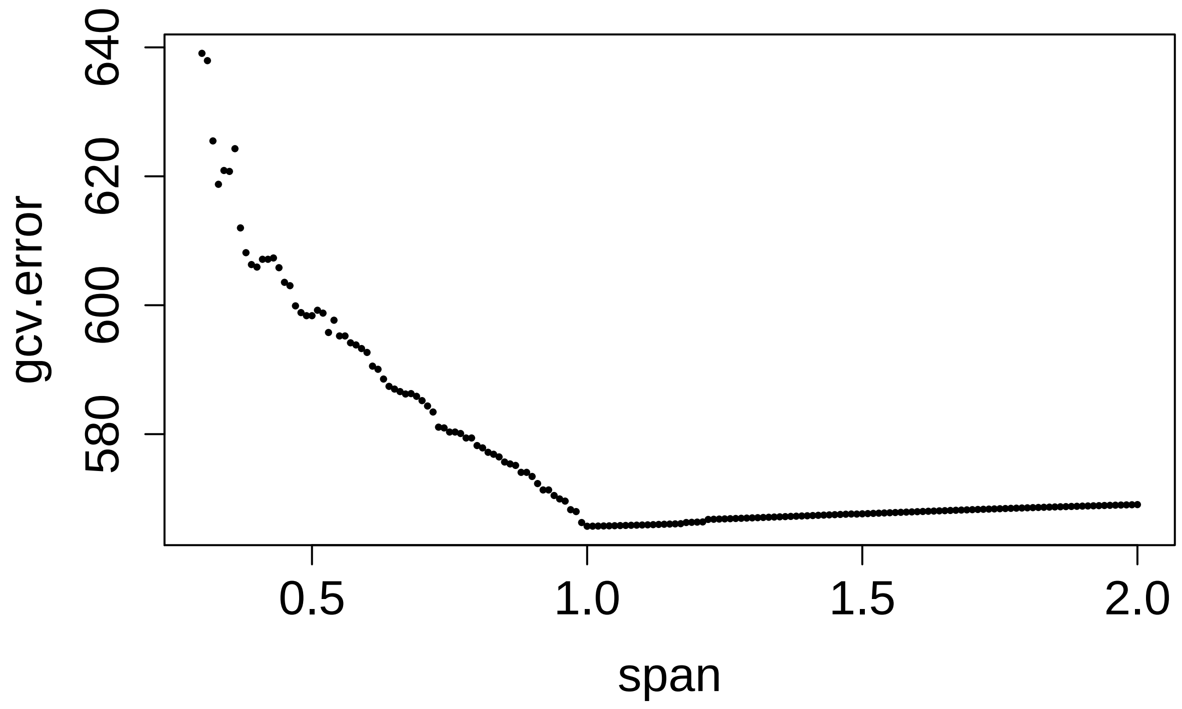
The best model based on GCV Error has df = 4.0.

We then consider modeling the data with a loess quadratic model, using generalized cross validation to select the optimum span.

```
x<- data$wind
y <- data$ozone

gcv.error <- c()

span = seq(from= 0.30 , to = 2.0, by =0.01)

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
    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,...))
          }
      }
  }
  S
}

for(j in 1:length(span)) {
        model = loess(ozone~wind, span = span[j], degrees=2)
        pred = fitted(model)
        S<- smootherMatrixLoess(wind, span = span[j])
        numerator = ozone-pred
        denominatorGCV = 1-mean(diag(S))
        gcv.error[j] = mean((numerator/denominatorGCV)^2)
}


c(span[which.min(gcv.error)], min(gcv.error))
```

```
## [1]   1.0000 565.7135
```

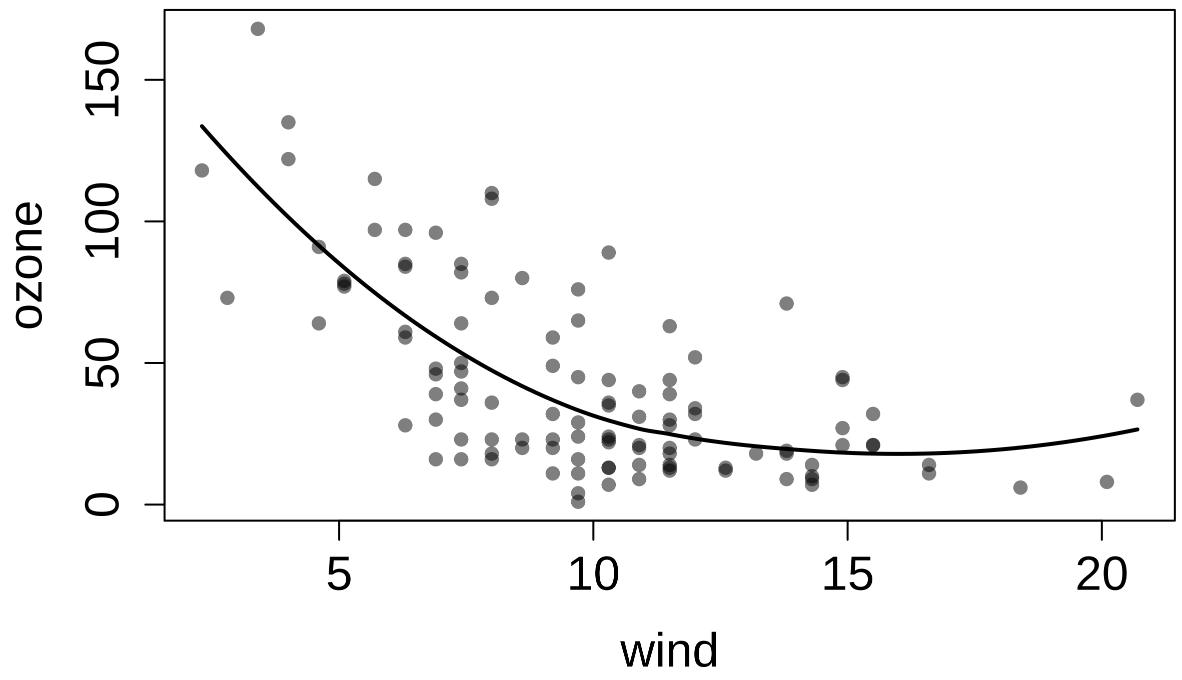
```
plot(span, gcv.error,
     pch=16, cex=0.5, col = adjustcolor("black",alpha=1),
     cex.axis=1.5, cex.main=1.5, cex.lab=1.5)
```



```r
S<- smootherMatrixLoess(wind, span = span[which.min(gcv.error)])
sum(diag(S))
```

```
## [1] 3.688232
```

```r
plot(wind, ozone,
     pch=16, cex=1, cex.axis = 1.5, cex.main = 1.5, cex.lab = 1.5,
     col=adjustcolor("black", alpha=0.5)
)

fit.loess.quadratic = loess(ozone~wind, span = span[which.min(gcv.error)])
wind.new = seq(min(wind), max(wind), length.out=500)
pred.loess.quadratic <- predict(fit.loess.quadratic, wind.new)
lines(wind.new, pred.loess.quadratic, col="black",lwd=2)
```



```
# for ( i in 1:length(span)) {
#
# fit <- loess(y~x,span = span[i], data=data, degree = 2)
#
# gcv.error[i] <- mean( (fit$residuals / ((1-fit$enp) / 1-((fit$enp)/length(span))) ) )^2
#
# }
#
# which.min(gcv.error)
# span[which.min(gcv.error)]
#
#
# plot(span, gcv.error, main = "GCV Error vs Span", xlab = "Span",
#      ylab = "GCV Error")
#
# abline(v = span[which.min(gcv.error)], lty =  2)
#
# model1 <- loess(y~x, span = span[which.min(gcv.error)], data = data, degree = 2)
#
# plot(x,y,
#      pch=16, cex=0.7,
#      main = "Local Quadratic, GCV",
#      cex.axis=1.5 , cex.main=1.5, cex.lab=1.5,, xlab = "Wind", ylab = "Ozone",
#      col=adjustcolor("black" , 0.5)
# )
#
```

```
# pred1 <- predict(model1)
# xorder <- order(x)
#
# lines(x[xorder], pred1[xorder], col = "green")
```