Stats790 A3

Hanwen Ju

2023-03-05

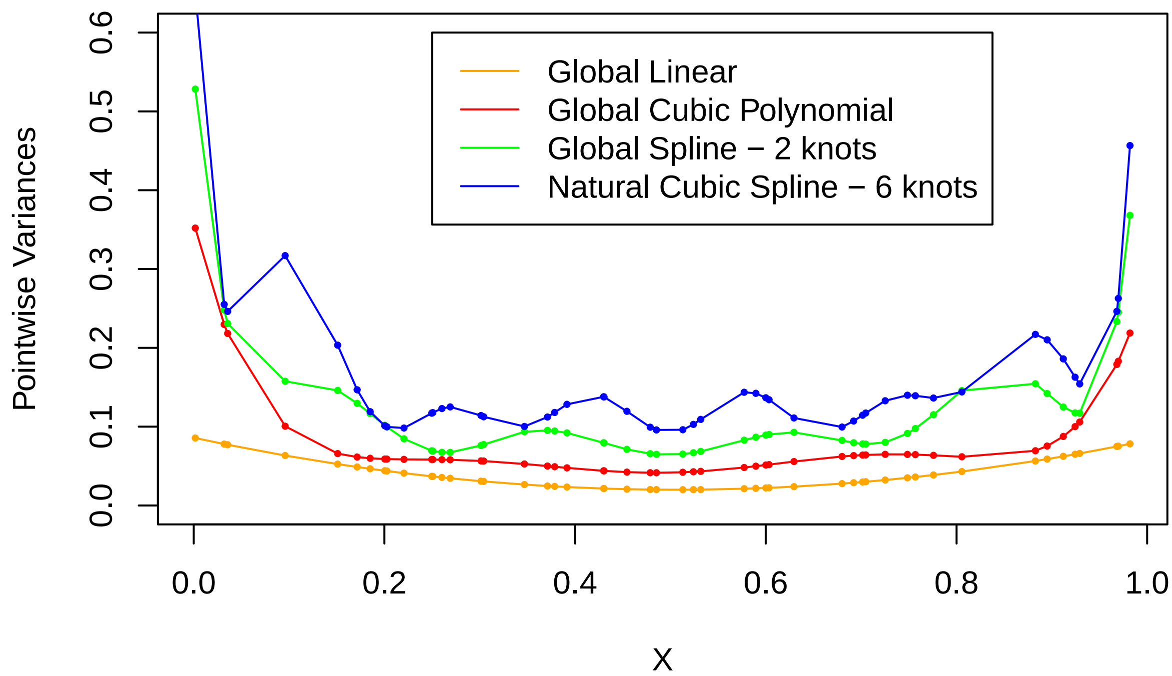
Question 1

```r
#ESL figure 5.3
library(splines)

x <- runif(50)

#Pointwise variance

pvar <- function(X) {diag(X%*%solve(t(X)%*%X)%*%t(X))}


#linear
bs_linear <- bs(x, degree = 1, df=2, intercept = TRUE)
var_linear <- pvar(bs_linear)

#cubic polynomial
bs_cubic <- bs(x, degree = 3, df=4, intercept = TRUE)
var_cubic <- pvar(bs_cubic)

#2 knots cubic
bs_cubic2 <- bs(x, degree = 3, df=6, intercept = TRUE, knots = c(0.33, 0.66))
var_cubic2 <- pvar(bs_cubic2)

#6 knots natural
knots <- seq(0.1, 0.9, length.out = 6)[2:5]
bs_ns <- bs(x, degree = 3, intercept = TRUE, Boundary.knots = c(0.1,0.9), knots = knots)
```

```
## Warning in bs(x, degree = 3, intercept = TRUE, Boundary.knots = c(0.1, 0.9), :
## some 'x' values beyond boundary knots may cause ill-conditioned bases
```

```r
var_ns <- pvar(bs_ns)



plot(x, var_cubic, ylim = c(0,0.6),cex = 0.5, pch = 16, col = 'red',
     ylab = 'Pointwise Variances', xlab = 'X')
lines(x[order(x)], var_cubic[order(x)], col = "red")
points(x, var_linear, col='orange', cex = 0.5,pch = 16)
lines(x[order(x)], var_linear[order(x)], col = "orange")
points(x, var_cubic2, col = 'green', cex = 0.5,pch = 16)
lines(x[order(x)], var_cubic2[order(x)], col = "green")
points(x, var_ns, col = 'blue',cex = 0.5, pch = 16)
```

```
lines(x[order(x)], var_ns[order(x)], col = "blue")
legend(x=0.25,y= 0.6,legend = c('Global Linear', 'Global Cubic Polynomial', 'Global Spline - 2 knots',
```



Question 2

```
library(foreign)
url <- "http://www-stat.stanford.edu/~tibs/ElemStatLearn/datasets/SAheart.data"
dd <- read.csv(url, row.names = 1)

exclude_vars <- c("chd", "famhist")
numvars <- setdiff(names(dd), exclude_vars)
library(splines)
library(nnet)

#train/test set


train_sa <- dd[1:380,]
train_tob <- as.data.frame(train_sa$tobacco)
train_tob_res <- train_sa$chd


test_sa <- dd[380:462,]
test_tob <- as.data.frame(test_sa$tobacco)
test_tob_res <- test_sa$chd
```

```
#natural cubic spline##############################################################################


#full_model, using all features:
spline_terms_ns <- sprintf("ns(%s, df = 6)", numvars)
ff_ns <- reformulate(c("famhist", spline_terms_ns), response = "chd")
```

```r
full_model_ns <- glm(ff_ns, family = binomial, data = dd)


#tobacco
knots_ns <- seq(min(train_tob), max(train_tob), length = 7)[2:6]
tob_ns <- ns(train_sa$tobacco,knots = knots_ns)


#logistic regression
tob_model_ns <- glm(chd ~ ns(tobacco,knots = knots_ns), data = train_sa, family = binomial)
```

```
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
```

```r
tob_ns_coe <- tob_model_ns$coefficients





#predict train
pred.ns.train <- as.matrix(cbind(rep(1,nrow(train_tob)),tob_ns)) %*% tob_ns_coe
pred.ns.train <- exp(pred.ns.train) / (1 + exp(pred.ns.train))

pred.ns.train<-ifelse(pred.ns.train>=0.5,1,0)

table(pred.ns.train,train_tob_res)
```

```
##              train_tob_res
## pred.ns.train   0   1
##             0 220  90
##             1  26  44
```

```r
#accuracy
sum(diag(table(pred.ns.train,train_tob_res)))/sum(table(pred.ns.train,train_tob_res))
```

```
## [1] 0.6947368
```

```r
#predict test
knots_ns.test <- seq(min(test_tob), max(test_tob), length = 7)[2:6]
tob_ns.test <- ns(test_sa$tobacco,knots = knots_ns.test)
pred.ns.test <- as.matrix(cbind(rep(1,nrow(test_tob)),tob_ns.test)) %*% tob_ns_coe
pred.ns.test <- exp(pred.ns.test) / (1 + exp(pred.ns.test))

pred.ns.test<-ifelse(pred.ns.test>=0.5,1,0)

table(pred.ns.test,test_tob_res)
```

```
##             test_tob_res
## pred.ns.test  0  1
##            0 46 19
##            1 10  8
```

```r
#accuracy
sum(diag(table(pred.ns.test,test_tob_res)))/sum(table(pred.ns.test,test_tob_res))
```

```
## [1] 0.6506024
```

```r
#b-spline######################################################################################

#full_model, using all features:
spline_terms_bs <- sprintf("bs(%s, df = 6)", numvars)
ff_bs <- reformulate(c("famhist", spline_terms_bs), response = "chd")
full_model_bs <- glm(ff_bs, family = binomial, data = dd)
#reduced_model <- MASS::stepAIC(full_model_ns, direction = "backward")
#as.formula(model.frame(reduced_model))

#tobacco
knots_bs <- seq(min(train_tob), max(train_tob), length = 7)[2:6]
tob_bs <- bs(train_sa$tobacco,knots = knots_bs)


#logistic regression
tob_model_bs <- glm(chd ~ bs(tobacco,knots = knots_bs), data = train_sa, family = binomial)
tob_bs_coe <- tob_model_bs$coefficients




#predict train
pred.bs.train <- as.matrix(cbind(rep(1,nrow(train_tob)),tob_bs)) %*% tob_bs_coe
pred.bs.train <- exp(pred.bs.train) / (1 + exp(pred.bs.train))

pred.bs.train<-ifelse(pred.bs.train>=0.5,1,0)

table(pred.bs.train,train_tob_res)
```

```
##               train_tob_res
## pred.bs.train   0   1
##             0 228  97
##             1  18  37
```

```r
#accuracy
sum(diag(table(pred.bs.train,train_tob_res)))/sum(table(pred.bs.train,train_tob_res))
```

```
## [1] 0.6973684
```

```r
#predict test
knots_bs.test <- seq(min(test_tob), max(test_tob), length = 7)[2:6]
tob_bs.test <- bs(test_sa$tobacco,knots = knots_bs.test)
pred.bs.test <- as.matrix(cbind(rep(1,nrow(test_tob)),tob_bs.test)) %*% tob_bs_coe
pred.bs.test <- exp(pred.bs.test) / (1 + exp(pred.bs.test))

pred.bs.test<-ifelse(pred.bs.test>=0.5,1,0)

table(pred.bs.test,test_tob_res)
```

```
##              test_tob_res
## pred.bs.test  0  1
##            0 51 22
##            1  5  5
```

```r
#accuracy
sum(diag(table(pred.bs.test,test_tob_res)))/sum(table(pred.bs.test,test_tob_res))
```

```
## [1] 0.6746988
```

```r
#truncated polynomial spline################################################################################

truncpolyspline <- function(x, df) {
if (!require("Matrix")) stop("need Matrix package")
knots <- quantile(x, seq(0, 1, length = df - 1))
## should probably use seq() instead of `:`
## dim: n x (df-2)
trunc_fun <- function(k) (x>=k)*(x-k)^3
S <- sapply(knots[2:(df-2)], trunc_fun)
#S <- as(S, "CsparseMatrix")
## dim: n x df
S <- cbind(x, x^2, S)
return(S)
}



#tobacco
tob_tps <- truncpolyspline(train_sa$tobacco, df = 8)
```

```
## Loading required package: Matrix
```

```r
#logistic regression
tob_model_tps <- glm(chd ~ truncpolyspline(tobacco, df = 8), data = train_sa, family = binomial)
tob_tps_coe <- tob_model_tps$coefficients





#predict train
pred.tps.train <- as.matrix(cbind(rep(1,nrow(train_tob)),tob_tps)) %*% tob_tps_coe
pred.tps.train <- exp(pred.tps.train) / (1 + exp(pred.tps.train))

pred.tps.train<-ifelse(pred.tps.train>=0.5,1,0)

table(pred.tps.train,train_tob_res)
```

```
##                 train_tob_res
## pred.tps.train   0   1
##                0 222  92
##                1  24  42
```

```r
#accuracy
sum(diag(table(pred.tps.train,train_tob_res)))/sum(table(pred.tps.train,train_tob_res))
```

```
## [1] 0.6947368
```

```r
#predict test
tob_tps.test <- truncpolyspline(test_sa$tobacco, df = 8)
pred.tps.test <- as.matrix(cbind(rep(1,nrow(test_tob)),tob_tps.test)) %*% tob_tps_coe
pred.tps.test <- exp(pred.tps.test) / (1 + exp(pred.tps.test))

pred.tps.test<-ifelse(pred.tps.test>=0.5,1,0)

table(pred.tps.test,test_tob_res)
```

```
##                test_tob_res
## pred.tps.test  0  1
##              0 35  8
##              1 21 19
```

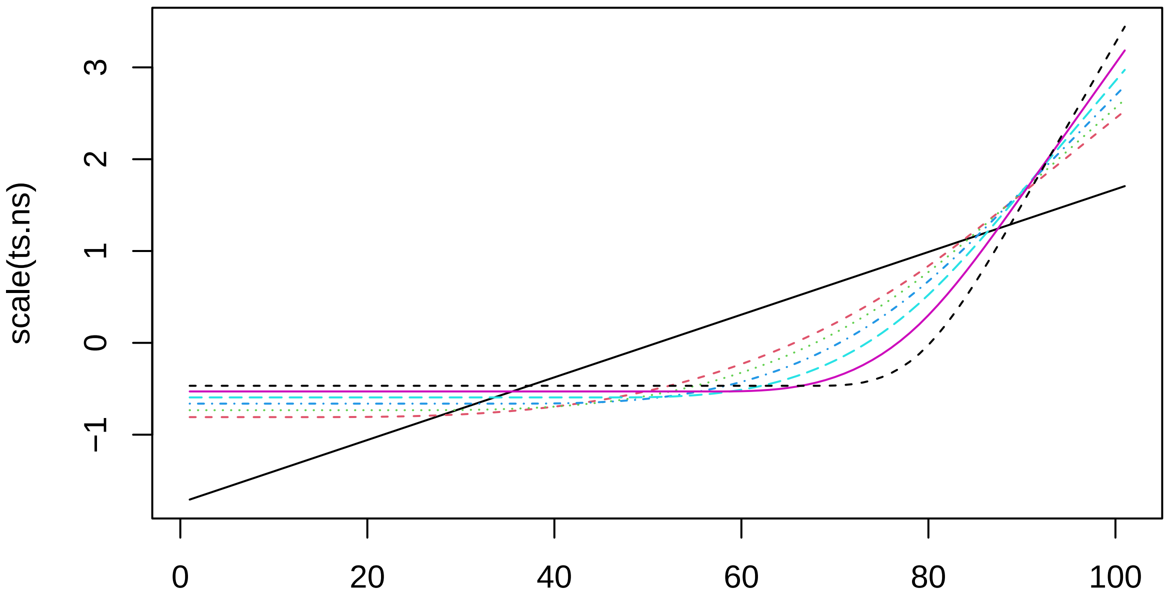
```
#accuracy
sum(diag(table(pred.tps.test,test_tob_res)))/sum(table(pred.tps.test,test_tob_res))
```

```
## [1] 0.6506024
```

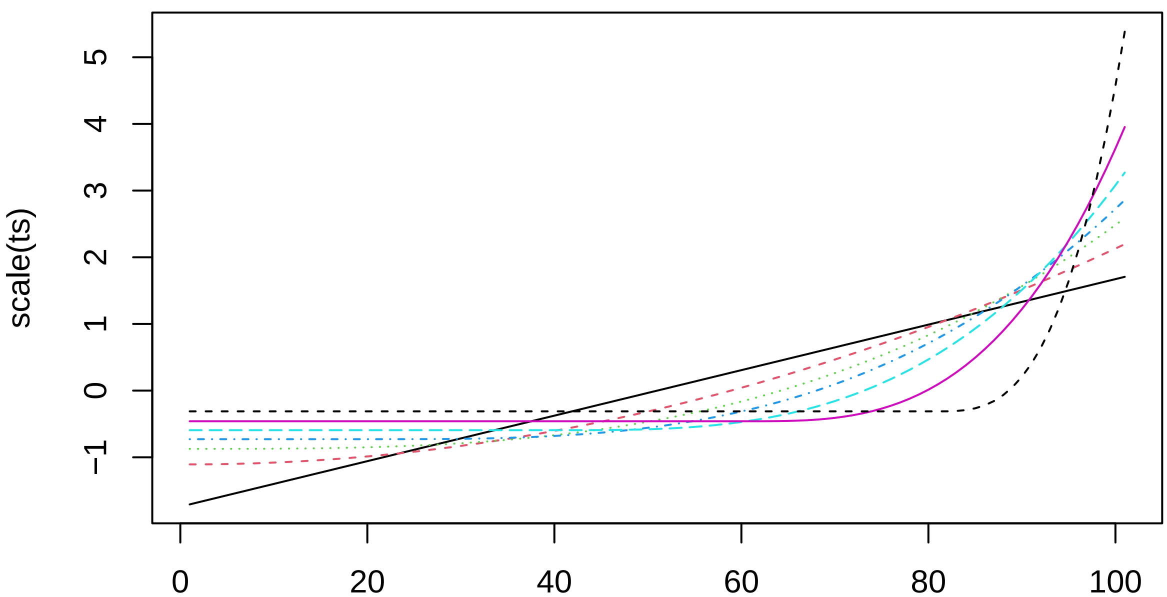
Question 3

```
library(Matrix)
truncpolyspline <- function(x, df,natrual) {
  if (!require("Matrix")) stop("need Matrix package")

# if natural cubic spline
if (natrual == TRUE){

  knots <- quantile(x, seq(0, 1, length = df+3))
  trunc_fun <- function(k) (x>=k)*(x-k)^3

  numberKnots <- df+2

  xiK <- knots[df+2] #last region

  SN <- x


  end <- df - 1
  for (i in 1:end){
    #print(j)
    dk <- (trunc_fun(knots[i+1])-trunc_fun(xiK))/(xiK-knots[i+1])
    dKm1 <- (trunc_fun(knots[df+1])-trunc_fun(xiK))/(xiK-knots[df+1])    #K-1
    SN <- cbind(SN,dk-dKm1)
  }
  return(SN)

}else{# cubic spline

  ## should probably use seq() instead of `:`
  ## dim: n x (df-2)

  knots <- quantile(x, seq(0, 1, length = df - 1))
  trunc_fun <- function(k) (x>=k)*(x-k)^3

  S <- sapply(knots[1:(df-2)], trunc_fun)
  S <- as(S, "CsparseMatrix")
  ## dim: n x df
  S <- cbind(x, x^2, S)
  return(S)
}

}
```

```
xvec <- seq(0, 1, length = 101)

ts.ns <- truncpolyspline(xvec, df = 7,natrual = TRUE)
ts <- truncpolyspline(xvec, df = 7,natrual = FALSE)

matplot(scale(ts.ns), type = "l")
```

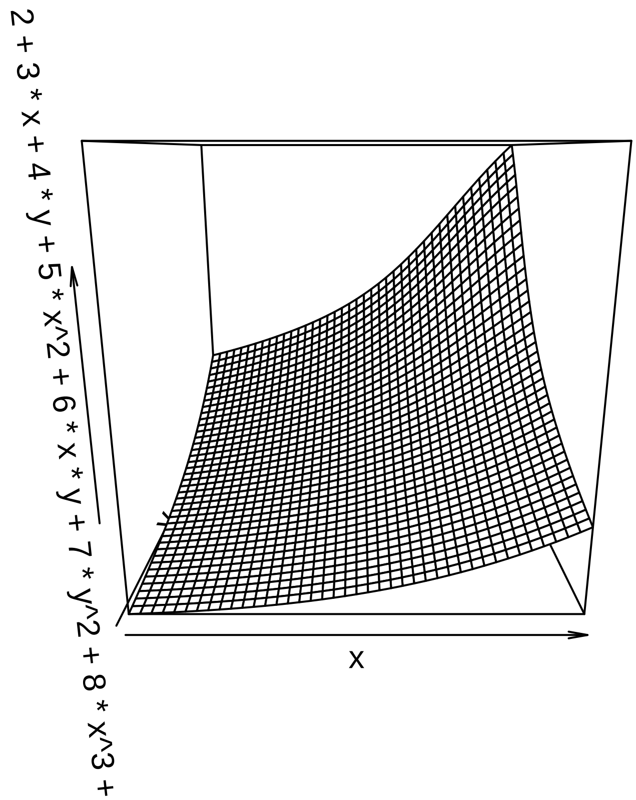



```
matplot(scale(ts), type = "l")
```

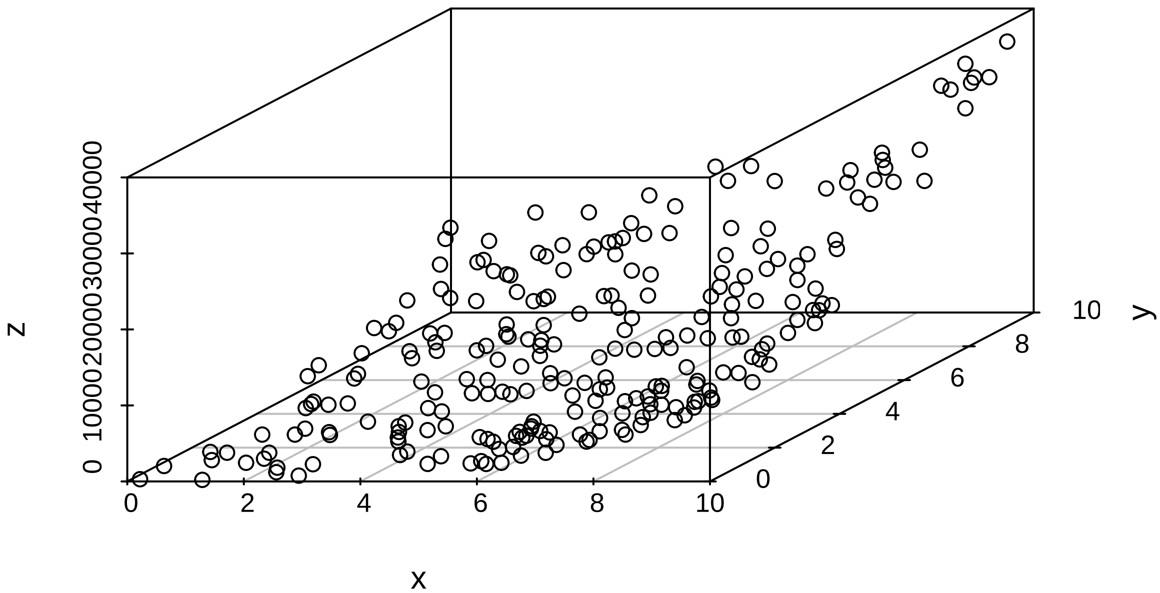


Question 4

```
library(emdbook)
library("scatterplot3d")

#3-oder bivariate polynomial
curve3d( 2 + 3*x + 4*y + 5*x^2 + 6*x*y + 7*y^2 + 8*x^3 + 9*x^2*y + 10*x*y^2 + 11*y^3 + 12*exp(-((x-1)^2
```




```
bivPoly3 <- function(x,y, sd){
  z <- 2 + 3*x + 4*y + 5*x^2 + 6*x*y + 7*y^2 + 8*x^3 + 9*x^2*y + 10*x*y^2 + 11*y^3 + 12*exp(-((x-1)^2 +

  # add gaussian noise to it
  z <- z + rnorm(length(x), mean = 0, sd = sd)
}


# create 250 data points:

x <- runif(250,0,10)
y <- runif(250,0,10)

z <- bivPoly3(x,y,0.4)

scatterplot3d(x, y,z)
```




```
library(mgcv)
```

```
## Loading required package: nlme
```

```
## This is mgcv 1.8-42. For overview type 'help("mgcv-package")'.
```

```
##
## Attaching package: 'mgcv'
```

```
## The following object is masked from 'package:nnet':
##
##     multinom
```

```
xy <- data.frame(x,y)


#model using generalized cross-validation

m1 <- gam(z ~ te(x,y,bs = "gp"),data = xy, method="GCV.Cp")
summary(m1)
```

```
##
## Family: gaussian
## Link function: identity
##
## Formula:
## z ~ te(x, y, bs = "gp")
##
## Parametric coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) 8135.6749     0.7242   11234   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Approximate significance of smooth terms:
##         edf Ref.df       F p-value
## te(x,y)  24     24 4039352  <2e-16 ***
```

```
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## R-sq.(adj) =      1   Deviance explained =  100%
## GCV = 145.69  Scale est. = 131.12    n = 250
```

```r
p1 <- predict(m1, newdata = xy)

#MSE
mean((z - p1)^2)
```

```
## [1] 118.0099
```

```r
#Bias
sum(abs(mean(p1) - z))
```

```
## [1] 1334621
```

```r
#variance
mean((p1 - mean(p1)) ^ 2)
```

```
## [1] 50845854
```

```r
#model using restricted maximum likelihood

m2 <- gam(z ~ te(x,y,bs = "gp"),data = xy, method = "REML")
summary(m2)
```

```
##
## Family: gaussian
## Link function: identity
##
## Formula:
## z ~ te(x, y, bs = "gp")
##
## Parametric coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) 8135.6749     0.7242   11234   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Approximate significance of smooth terms:
##         edf Ref.df       F p-value
## te(x,y)  24     24 4039230  <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## R-sq.(adj) =      1   Deviance explained =  100%
## -REML = 1069.1  Scale est. = 131.13    n = 250
```

```r
p2 <- predict(m2, newdata = xy)



#MSE
sum((z - p2)^2)
```

```
## [1] 29503.73
```

```r
#Bias
sum(abs(mean(p2) - z))
```

```
## [1] 1334621
```

```r
#variance
mean((p2 - mean(p2)) ^ 2)
```

```
## [1] 50845841
```

```r
# Time

library(microbenchmark)
m1 <- microbenchmark(
    gam(z ~ te(x,y,bs = "gp"),data = xy, method="GCV.Cp"),
    gam(z ~ te(x,y,bs = "gp"),data = xy, method = "REML")
    )
results <- summary(m1)
#time comparsion
rownames(results) <- c("generalized cross-validation", "restricted maximum likelihood")
results[,-1]
```

```
##                                    min       lq     mean   median       uq
## generalized cross-validation  29.38632 32.05909 35.49187 34.23362 35.51872
## restricted maximum likelihood 57.41836 61.48761 65.15445 63.41964 65.48375
##                                    max neval
## generalized cross-validation  182.9319   100
## restricted maximum likelihood 208.4050   100
```

Stats790 Assignment3

Hanwen Ju

March 2023

1 ESL 5.4

Consider the truncated power series representation for cubic splines with K interior knots:

$$f(X) = \sum_{j=0}^{3} \beta_j X^j + \sum_{k=1}^{K} \theta_k (X - \xi_k)_+^3 \tag{1}$$

Assume we have $x1 \leq \xi_1$ and $x_2 \geq \xi_2$. So we have the polynomial for $x1$:

$$f(x_1) = \beta_0 + \beta_1 x_1 + \beta_2 x_1^2 + \beta_3 x_1^3 \tag{2}$$

by take its 1st derivative:

$$f'(x_1) = \beta_1 + \beta_2 x_1 + \beta_3 x_1^2 \tag{3}$$

By the definition of nature boundary conditions for natural cubic splines, its second and third derivative should be zero. So its first derivative is constant. Thus, we must have:

$$\beta_2 = \beta_3 = 0 \tag{4}$$

Consider x_2, we have the polynomial for x_2:

$$\begin{aligned} f(x_2) &= \beta_0 + \beta_1 x_2 + \beta_2 x_2^2 + \beta_3 x_2^3 + \sum_{k=1}^{K} \theta_k (x_2 - \xi_k)_+^3 \\ &= \beta_0 + \beta_1 x_2 + \sum_{k=1}^{K} \theta_k (x_2 - \xi_k)_+^3 \end{aligned} \tag{5}$$

By taking its 1st derivative, we get:

$$\begin{aligned} f'(x_2) &= \beta_1 + 3 \sum_{k=1}^{K} \theta_k (x_2 - \xi_k)^2 \\ &= \beta_1 + 3 \left[\sum_{k=1}^{K} \theta_k x_2^2 - 2 \sum_{k=1}^{K} \theta_k \xi_k x_2 + \sum_{k=1}^{K} \theta_k \xi_k^2 \right] \end{aligned} \tag{6}$$

Similarly, we know $f'(x_2) = 0$. So we must have:

$$\sum_{k=1}^{K} \theta_k = 0 \tag{7}$$

and

$$\sum_{k=1}^{K} \theta_k \xi_k = 0 \tag{8}$$

Back to the equation (5), we know the general case for any $x_i \geq \xi_2$ is:

$$f(x_i) = \beta_0 + \beta_1 x_i + \sum_{k=1}^{K} \theta_k (x_i - \xi_k)_+^3 \tag{9}$$

So the first two basis for natural cubic spline is:

$$N_1 = 1, N_2 = X \tag{10}$$

We know we have $(K+1) \times 4 - 3 \times K - 4 = K$ parameters for natural cubic spline with K knots. We want to have the form $f(x) = \sum_{k=1}^{K} \beta_k' N_k(x)$. First, by using the constraints $\sum_{k=1}^{K} \theta_k = 0$ and $\sum_{k=1}^{K} \theta_k \xi_k = 0$, we can derive θ_K and θ_{K-1} from $\sum_{k=1}^{K} \theta_k = \sum_{k=1}^{K} \theta_k \xi_k$:

$$\theta_{K-1} = -\sum_{k=1}^{K-2} \theta_k \frac{\xi_K - \xi_k}{\xi_K - \xi_{K-1}} \tag{11}$$

$$\theta_K = -\sum_{k=1}^{K-2} \theta_k \frac{\xi_{K-1} - \xi_k}{\xi_{K-1} - \xi_K} \tag{12}$$

We can rewrite the $f(X)$ as:

$$\begin{aligned}
f(X) &= \beta_0 + \beta_1 X + \sum_{k=1}^{K} \theta_k (X - \xi_k)_+^3 \\
&= \beta_0 + \beta_1 X + \sum_{k=1}^{K-2} \theta_k (X - \xi_k)^3 + \theta_{K-1}(X - \xi_{K-1})^3 + \theta_K (X - \xi_K)^3 \\
&= \beta_0 + \beta_1 X + \sum_{k=1}^{K-2} \theta_k (X - \xi_k)^3 - \sum_{k=1}^{K-2} \theta_k \frac{\xi_K - \xi_k}{\xi_K - \xi_{K-1}} (X - \xi_{K-1})^3 - \sum_{k=1}^{K-2} \theta_k \frac{\xi_{K-1} - \xi_k}{\xi_{K-1} - \xi_K} (X - \xi_K)^3 \\
&= \beta_0 + \beta_1 X + \sum_{k=1}^{K-2} \theta_k \left[(X - \xi_k)^3 - \frac{\xi_K - \xi_k}{\xi_K - \xi_{K-1}} (X - \xi_{K-1})^3 - \frac{\xi_{K-1} - \xi_k}{\xi_{K-1} - \xi_K} (X - \xi_K)^3 \right] \\
&= \beta_0 + \beta_1 X + \sum_{k=1}^{K-2} \theta_k \left[(X - \xi_k)^3 - \frac{\xi_K - \xi_k}{\xi_K - \xi_{K-1}} (X - \xi_{K-1})^3 - \frac{(\xi_K - \xi_k) - (\xi_K - \xi_{K-1})}{\xi_{K-1} - \xi_K} (X - \xi_K)^3 \right] \\
&= \beta_0 + \beta_1 X + \sum_{k=1}^{K-2} \theta_k \left[(X - \xi_k)^3 - \frac{\xi_K - \xi_k}{\xi_K - \xi_{K-1}} (X - \xi_{K-1})^3 - \frac{\xi_K - \xi_k}{\xi_{K-1} - \xi_K} (X - \xi_K)^3 - (X - \xi_K)^3 \right] \\
&= \beta_0 + \beta_1 X + \sum_{k=1}^{K-2} \theta_k \left[(X - \xi_k)^3 - (X - \xi_K)^3 - \frac{\xi_K - \xi_k}{\xi_K - \xi_{K-1}} (X - \xi_{K-1})^3 - \frac{\xi_K - \xi_k}{\xi_{K-1} - \xi_K} (X - \xi_K)^3 \right] \\
&= \beta_0 + \beta_1 X + \sum_{k=1}^{K-2} \theta_k (\xi_K - \xi_k) \left[\frac{(X - \xi_k)^3 - (X - \xi_K)^3}{\xi_K - \xi_k} - \frac{1}{\xi_K - \xi_{K-1}} (X - \xi_{K-1})^3 - \frac{1}{\xi_{K-1} - \xi_K} (X - \xi_K)^3 \right] \\
&= \beta_0 + \beta_1 X + \sum_{k=1}^{K-2} \theta_k (\xi_K - \xi_k) \left[\frac{(X - \xi_k)^3 - (X - \xi_K)^3}{\xi_K - \xi_k} - \frac{(X - \xi_{K-1})^3 - (X - \xi_K)^3}{\xi_K - \xi_{K-1}} \right] \\
&= \beta_0 + \beta_1 X + \sum_{k=1}^{K-2} \theta_k (\xi_K - \xi_k) \left[\frac{(X - \xi_k)^3 - (X - \xi_K)^3}{\xi_K - \xi_k} - \frac{(X - \xi_{K-1})^3 - (X - \xi_K)^3}{\xi_K - \xi_{K-1}} \right] \\
&= \beta_0 + \beta_1 X + \sum_{k=1}^{K-2} \theta_k (\xi_K - \xi_k) \left[d_k(X) - d_{K-1}(X) \right]
\end{aligned} \tag{13}$$

where:

$$d_k = \frac{(X-\xi_k)^3 - (X-\xi_K)^3}{\xi_K - \xi_k} \tag{14}$$

And we have:

$$N_{k+2} = d_k(X) - d_{K-1}(X) \tag{15}$$

2 ESL 5.13

Suppose we have fitted a smoothing spline $\hat{f}_\lambda$ to a sample of N pairs (x_i, y_i). We have:

$$\hat{f}_\lambda(x_0) = S_\lambda y = \sum_{j=1}^{N} S_\lambda(i,j)y_j + S_\lambda(i,0)\hat{f}_\lambda(x_0) \tag{16}$$

If we add a new pair defined in the question $(x_0, \hat{f}_\lambda(x_0))$, and let $y_0 = \hat{f}_\lambda(x_0)$ so we have:

$$\hat{f}'_\lambda(x_0) = S'_\lambda y' = \sum_{j=1}^{N} S_\lambda(i,j)y_j + S_\lambda(i,0)y_0 \tag{17}$$

We can subtract equation (16) and (17) to get:

$$\begin{aligned}\hat{f}_\lambda(x_0) - \hat{f}'_\lambda(x_0) &= \sum_{j=1}^{N} S_\lambda(i,j)y_j + S_\lambda(i,0)\hat{f}_\lambda(x_0) - (\sum_{j=1}^{N} S_\lambda(i,j)y_j + S_\lambda(i,0)y_0) \\ &= S_\lambda(i,0)\hat{f}_\lambda(x_0) - S_\lambda(i,0)y_0\end{aligned} \tag{18}$$

so

$$\hat{f}_\lambda(x_0) = \frac{\hat{f}_\lambda(x_0) - S_\lambda(i,0)y_0}{1 - S_\lambda(i,0)} \tag{19}$$

Hence, we have:

$$y_0 - \hat{f}_\lambda(x_0) = y_0 - \frac{\hat{f}_\lambda(x_0) - S_\lambda(i,0)y_0}{1 - S_\lambda(i,0)} = \frac{y_0 - \hat{f}_\lambda(x_0)}{1 - S_\lambda(i,0)} \tag{20}$$