高级统计方法 第N次作业:

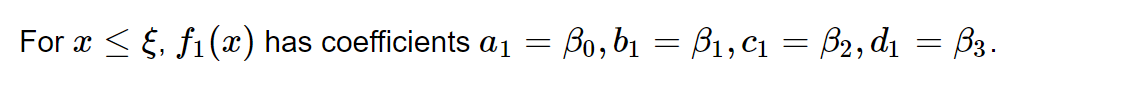
序号： 08 姓名： 潘晨楷 学号： 20212241116 班级：软2107

**作业评阅：**

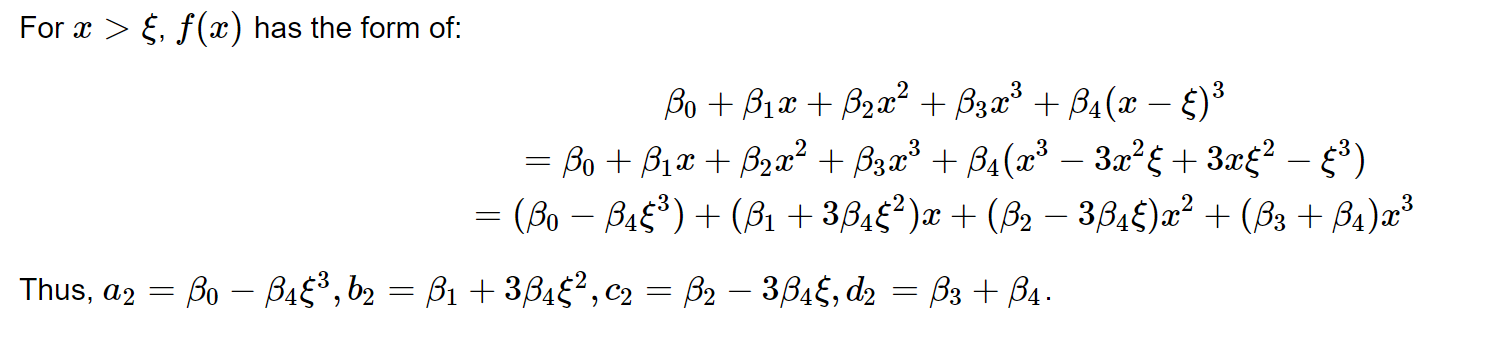
**概念**

1.问题（略）

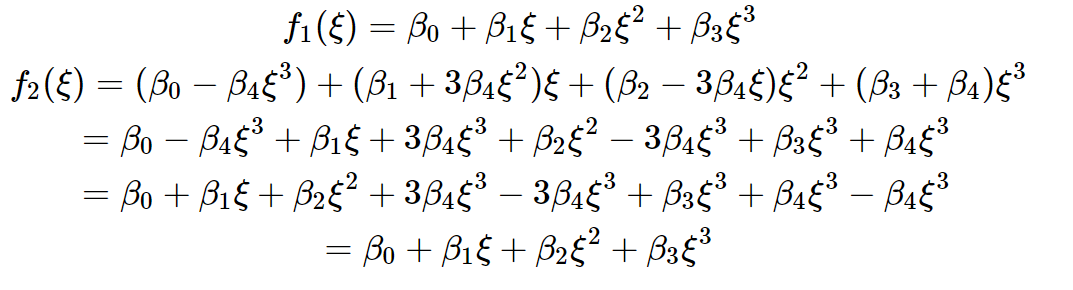
（a）问题（略）



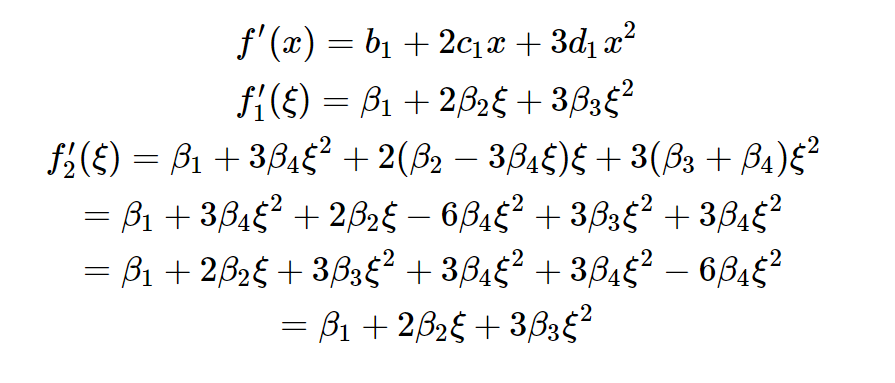
（b）问题（略）



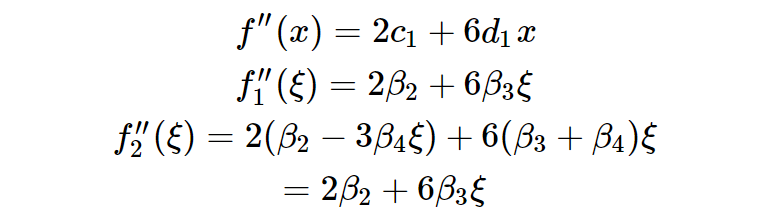
（c）问题（略）



（d）问题（略）



（e）问题（略）



2.问题（略）

（a）问题（略）

λ为无穷，忽略RSS, m=0，此时g应为0，即y=0，使其最小。

（b）问题（略）

λ为无穷，忽略RSS, m=1，此时g应为一条直线，例如y=k，使其最小。

（c）问题（略）

λ为无穷，忽略RSS, m=2，此时g的一阶导数为常数，g为一次函数，例如y=kx+c，使其最小。

（d）问题（略）

λ为无穷，忽略RSS, m=3，此时g的二阶导数为常数，g为二次函数，例如y=kx^2+c，使其最小。

（e）问题（略）

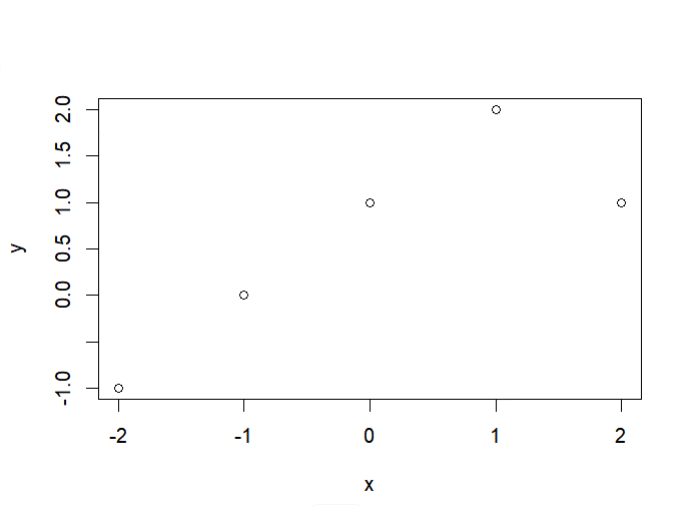
λ为0，忽略第二项, m=3，此时为最小二乘选择g。

3.问题（略）

x = -2:2

y = 1 + x + -2 \* (x-1)^2 \* I(x>1)

plot(x, y)



4.问题（略）

x = -2:2

y = c(1 + 0 + 0, # x = -2

1 + 0 + 0, # x = -1

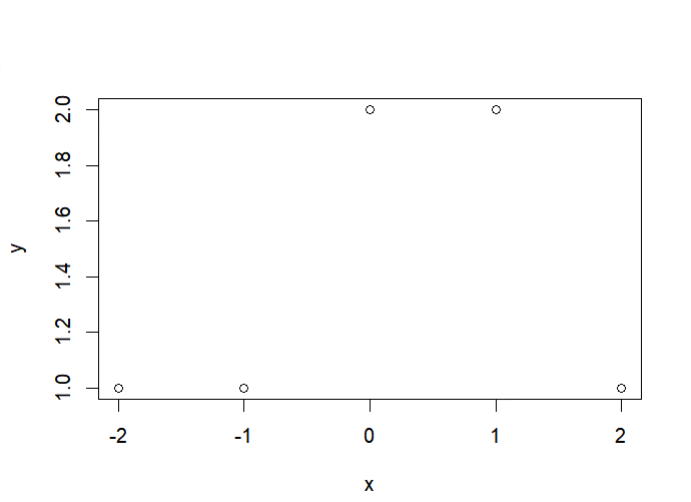
1 + 1 + 0, # x = 0

1 + (1-0) + 0, # x = 1

1 + (1-1) + 0 # x =2

)

plot(x,y)



5.问题（略）

（a）问题（略）

g2的训练RSS更小，其由于有着更高阶的导数，光滑度更高。

（b）问题（略）

g1的测试RSS更小，g2的高光滑度会造成过拟合的现象。

（c）问题（略）

当λ为0时，g1=g2，所以训练RSS和测试RSS都一样。

**应用**

9.问题（略）

set.seed(1)

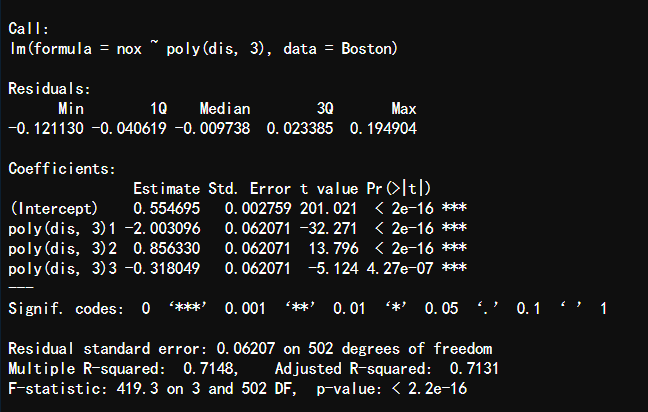
library(MASS)

attach(Boston)

（a）问题（略）

lm.fit = lm(nox ~ poly(dis, 3), data = Boston)

summary(lm.fit)



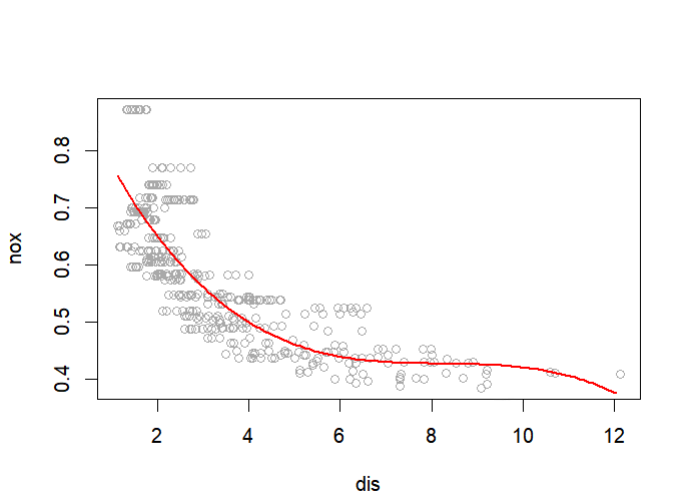
dislim = range(dis)

dis.grid = seq(from = dislim[1], to = dislim[2], by = 0.1)

lm.pred = predict(lm.fit, list(dis = dis.grid))

plot(nox ~ dis, data = Boston, col = "darkgrey")

lines(dis.grid, lm.pred, col = "red", lwd = 2)



由图所示，有着较好的拟合效果。

（b）问题（略）

all.rss = rep(NA, 10)

for (i in 1:10) {

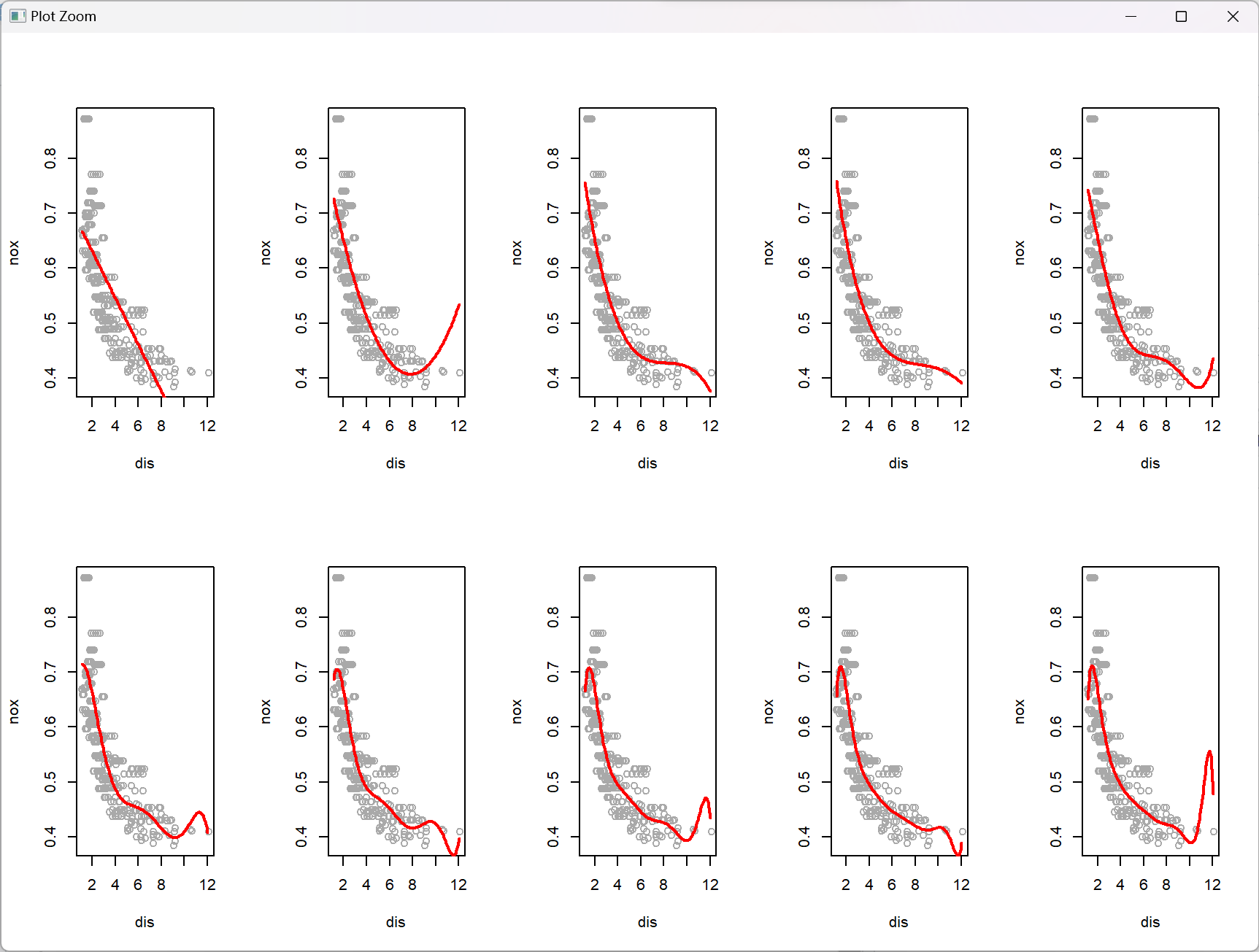
lm.fit = lm(nox ~ poly(dis, i), data = Boston)

all.rss[i] = sum(lm.fit$residuals^2)

}

all.rss





和预期的结果一样，训练RSS随多项式次数的增加而单调减小。

（c）问题（略）

library(boot)

all.deltas = rep(NA, 10)

for (i in 1:10) {

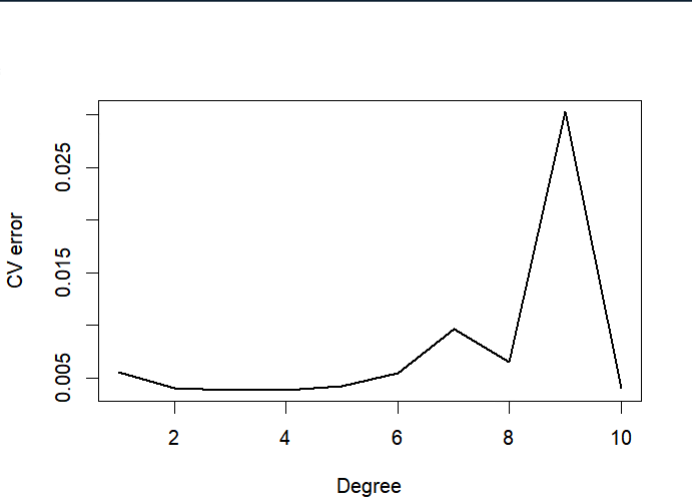
glm.fit = glm(nox ~ poly(dis, i), data = Boston)

all.deltas[i] = cv.glm(Boston, glm.fit, K = 10)$delta[2]

}

plot(1:10, all.deltas, xlab = "Degree", ylab = "CV error", type = "l", pch = 20,

lwd = 2)



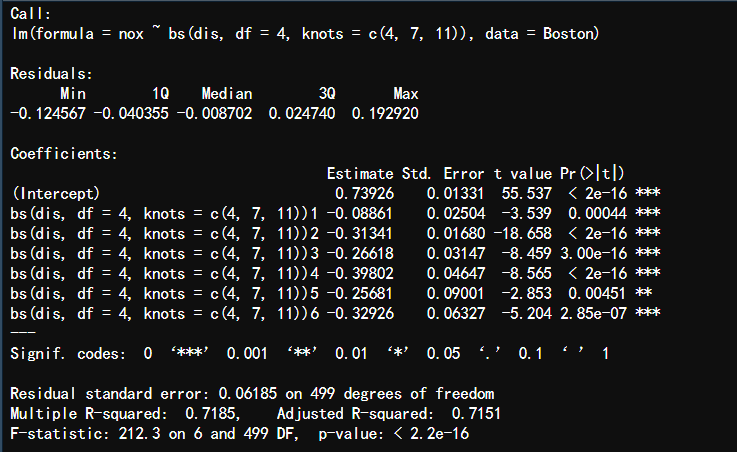
如图所示，在自由度为4的时候，交叉验证的误差最小。

（d）问题（略）

library(splines)

sp.fit = lm(nox ~ bs(dis, df = 4, knots = c(4, 7, 11)), data = Boston)

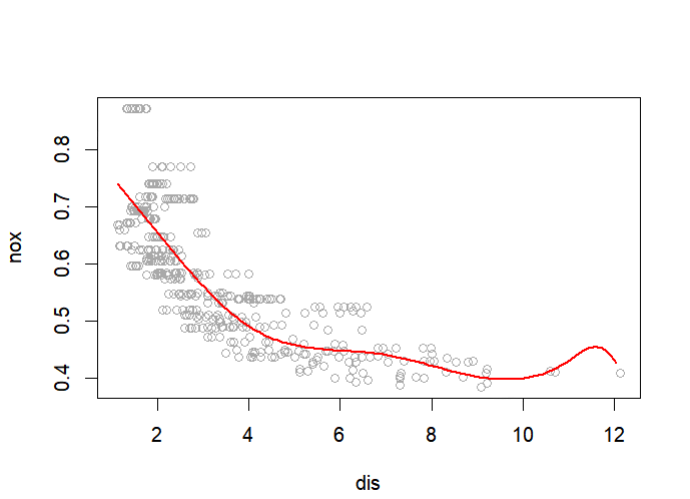
summary(sp.fit)



sp.pred = predict(sp.fit, list(dis = dis.grid))

plot(nox ~ dis, data = Boston, col = "darkgrey")

lines(dis.grid, sp.pred, col = "red", lwd = 2)



自由度为4，选择三个节点，dis的范围在1到12，分成4个间隔，选取节点为4，7，10。

（e）问题（略）

all.cv = rep(NA, 16)

for (i in 3:16) {

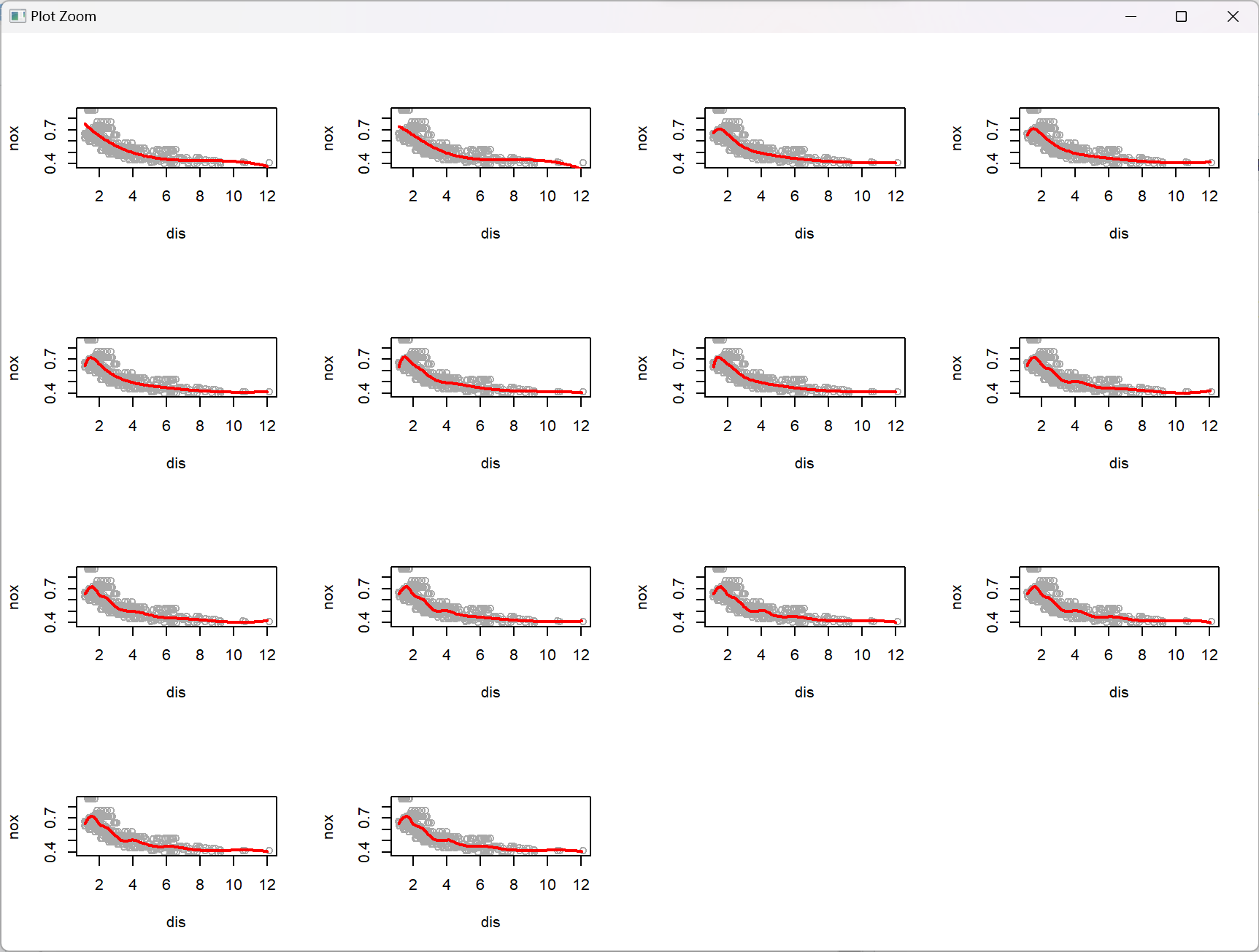
lm.fit = lm(nox ~ bs(dis, df = i), data = Boston)

all.cv[i] = sum(lm.fit$residuals^2)

}

all.cv[-c(1, 2)]





训练RSS在df=6前单调下降，在df=7和df=9时略有增加。

（f）问题（略）

all.cv = rep(NA, 16)

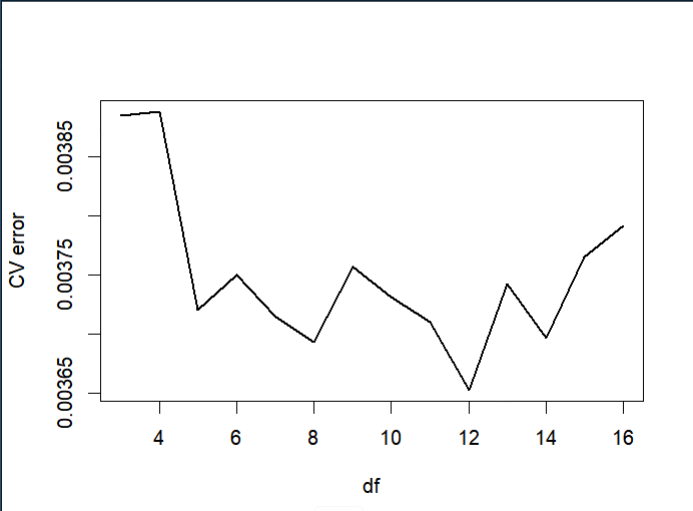
for (i in 3:16) {

lm.fit = glm(nox ~ bs(dis, df = i), data = Boston)

all.cv[i] = cv.glm(Boston, lm.fit, K = 10)$delta[2]

}

plot(3:16, all.cv[-c(1, 2)], lwd = 2, type = "l", xlab = "df", ylab = "CV error")



如图所示，在十折交叉验证下，在自由度12的时候取得最小的误差。

10.问题（略）

（a）问题（略）

set.seed(1)

library(ISLR)

library(leaps)

attach(College)

train = sample(length(Outstate), length(Outstate)/2)

test = -train

College.train = College[train, ]

College.test = College[test, ]

reg.fit = regsubsets(Outstate ~ ., data = College.train, nvmax = 17, method = "forward")

reg.summary = summary(reg.fit)

par(mfrow = c(1, 3))

plot(reg.summary$cp, xlab = "Number of Variables", ylab = "Cp", type = "l")

min.cp = min(reg.summary$cp)

std.cp = sd(reg.summary$cp)

abline(h = min.cp + 0.2 \* std.cp, col = "red", lty = 2)

abline(h = min.cp - 0.2 \* std.cp, col = "red", lty = 2)

plot(reg.summary$bic, xlab = "Number of Variables", ylab = "BIC", type = "l")

min.bic = min(reg.summary$bic)

std.bic = sd(reg.summary$bic)

abline(h = min.bic + 0.2 \* std.bic, col = "red", lty = 2)

abline(h = min.bic - 0.2 \* std.bic, col = "red", lty = 2)

plot(reg.summary$adjr2, xlab = "Number of Variables", ylab = "Adjusted R2",

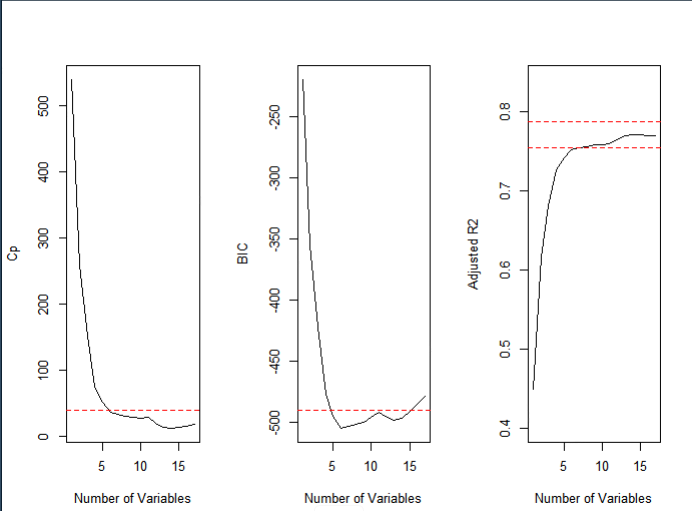
type = "l", ylim = c(0.4, 0.84))

max.adjr2 = max(reg.summary$adjr2)

std.adjr2 = sd(reg.summary$adjr2)

abline(h = max.adjr2 + 0.2 \* std.adjr2, col = "red", lty = 2)

abline(h = max.adjr2 - 0.2 \* std.adjr2, col = "red", lty = 2)



对于Cp和adjr2是14占优，而bic是6占优，根据图像可以得到，在三个图像中6和14的差距不大，为计算方便，采用6变量模型。

reg.fit = regsubsets(Outstate ~ ., data = College, method = "forward")

coefi = coef(reg.fit, id = 6)

names(coefi)



（b）问题（略）

Install.packages(“gam”)

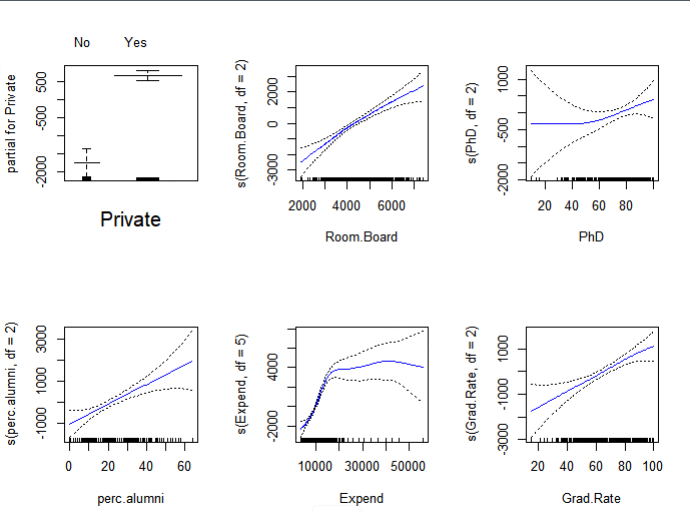
Libary(gam)

gam.fit = gam(Outstate ~ Private + s(Room.Board, df = 2) + s(PhD, df = 2) +

s(perc.alumni, df = 2) + s(Expend, df = 5) + s(Grad.Rate, df = 2), data = College.train)

par(mfrow = c(2, 3))

plot(gam.fit, se = T, col = "blue")



（c）问题（略）

gam.pred = predict(gam.fit, College.test)

gam.err = mean((College.test$Outstate - gam.pred)^2)

gam.err



gam.tss = mean((College.test$Outstate - mean(College.test$Outstate))^2)

test.rss = 1 - gam.err/gam.tss

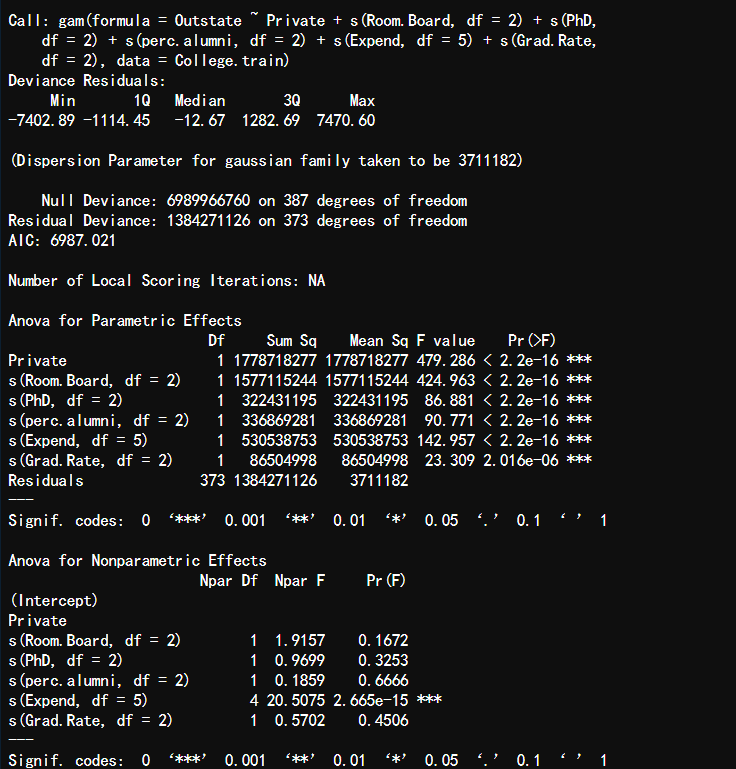
test.rss



测试RSS为3349290，测试R2为0.7660016

（d）问题（略）

summary(gam.fit)



通过观察p值，所有变量的p值都挺小，并根据（b）中图像，可以得出其与响应变量有显著的非线性关系。

11.问题（略）

（a）问题（略）

set.seed(1)

X1 = rnorm(100)

X2 = rnorm(100)

eps = rnorm(100, sd = 0.1)

Y = -2.1 + 1.3 \* X1 + 0.54 \* X2 + eps

（b）问题（略）

beta0 = rep(NA, 1000)

beta1 = rep(NA, 1000)

beta2 = rep(NA, 1000)

beta1[1] = 10

（c）（d）（e）问题（略）

for (i in 1:1000) {

a = Y - beta1[i] \* X1

beta2[i] = lm(a ~ X2)$coef[2]

a = Y - beta2[i] \* X2

lm.fit = lm(a ~ X1)

if (i < 1000) {

beta1[i + 1] = lm.fit$coef[2]

}

beta0[i] = lm.fit$coef[1]

}

plot(1:1000, beta0, type = "l", xlab = "iteration", ylab = "betas", ylim = c(-2.2,

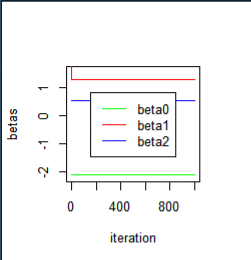
1.6), col = "green")

lines(1:1000, beta1, col = "red")

lines(1:1000, beta2, col = "blue")

legend("center", c("beta0", "beta1", "beta2"), lty = 1, col = c("green", "red",

"blue"))



如图所示，模型预测出的系数在很少的迭代之下便收敛到了真实值。

（f）问题（略）

lm.fit = lm(Y ~ X1 + X2)

plot(1:1000, beta0, type = "l", xlab = "iteration", ylab = "betas", ylim = c(-2.2,

1.6), col = "green")

lines(1:1000, beta1, col = "red")

lines(1:1000, beta2, col = "blue")

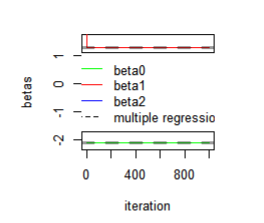
abline(h = lm.fit$coef[1], lty = "dashed", lwd = 3, col = rgb(0, 0, 0, alpha = 0.4))

abline(h = lm.fit$coef[2], lty = "dashed", lwd = 3, col = rgb(0, 0, 0, alpha = 0.4))

abline(h = lm.fit$coef[3], lty = "dashed", lwd = 3, col = rgb(0, 0, 0, alpha = 0.4))

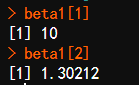
legend("center", c("beta0", "beta1", "beta2", "multiple regression"), lty = c(1,

1, 1, 2), col = c("green", "red", "blue", "black"))



如图所示，得出的结果与（e）中完全相同

（g）问题（略）



第一次迭代的前的beta1值是自设的，而一次迭代之后就接近于真实系数。所以对于线性关系，只需要一次迭代就能得到系数估计的近似结果。