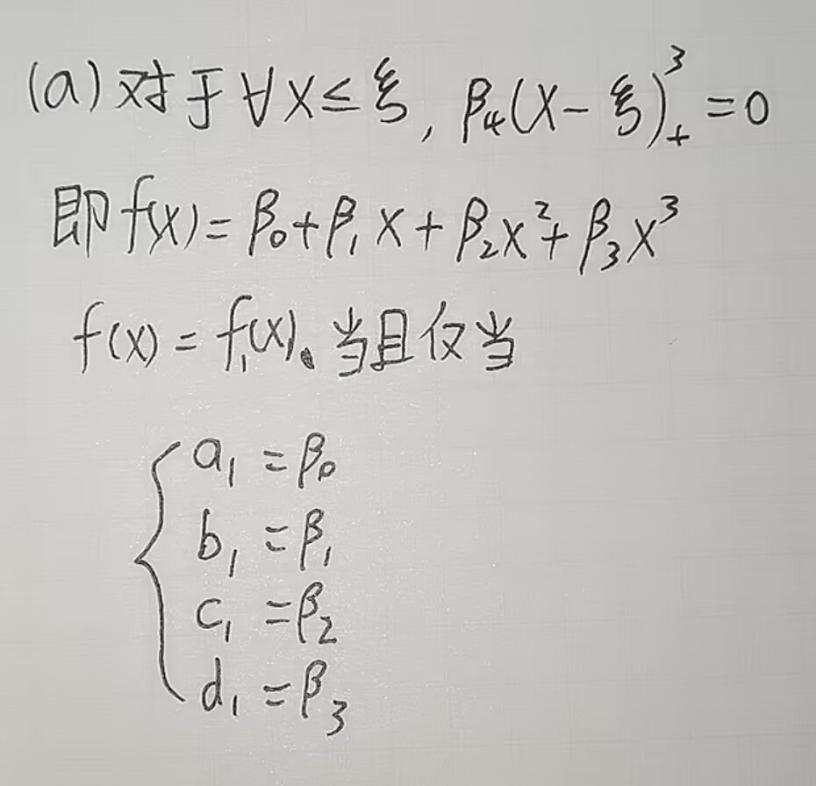
高级统计方法 第8次作业:

序号：1 姓名：杨程瑜 学号：20202241107 班级：网2001

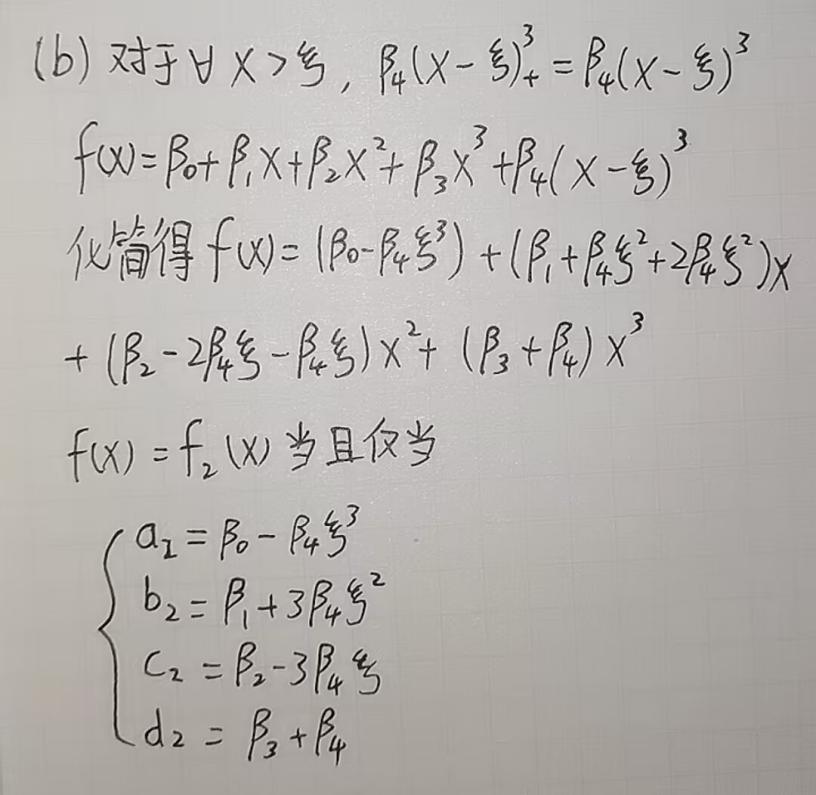
**概念**

1.问题（略）

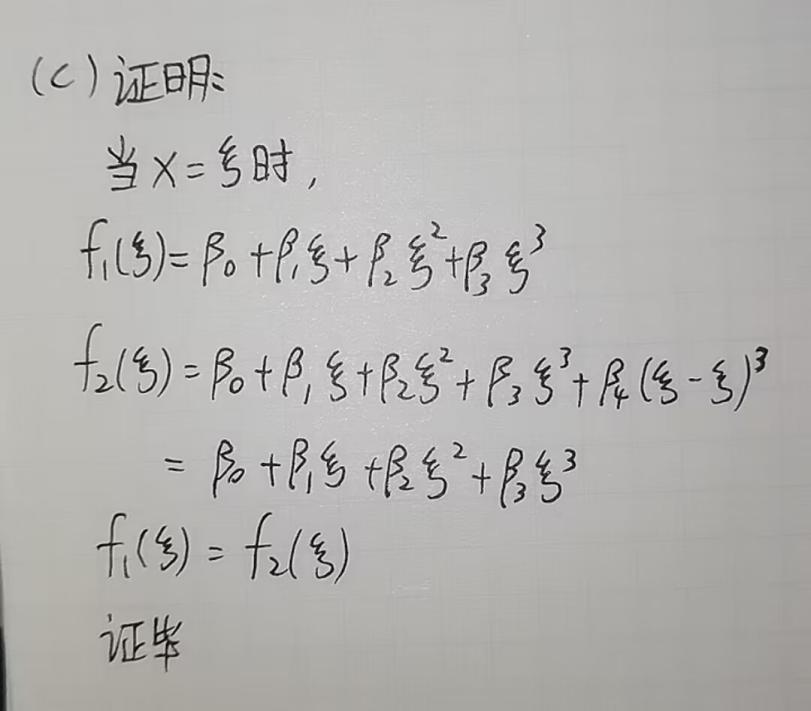
（a）问题（略）



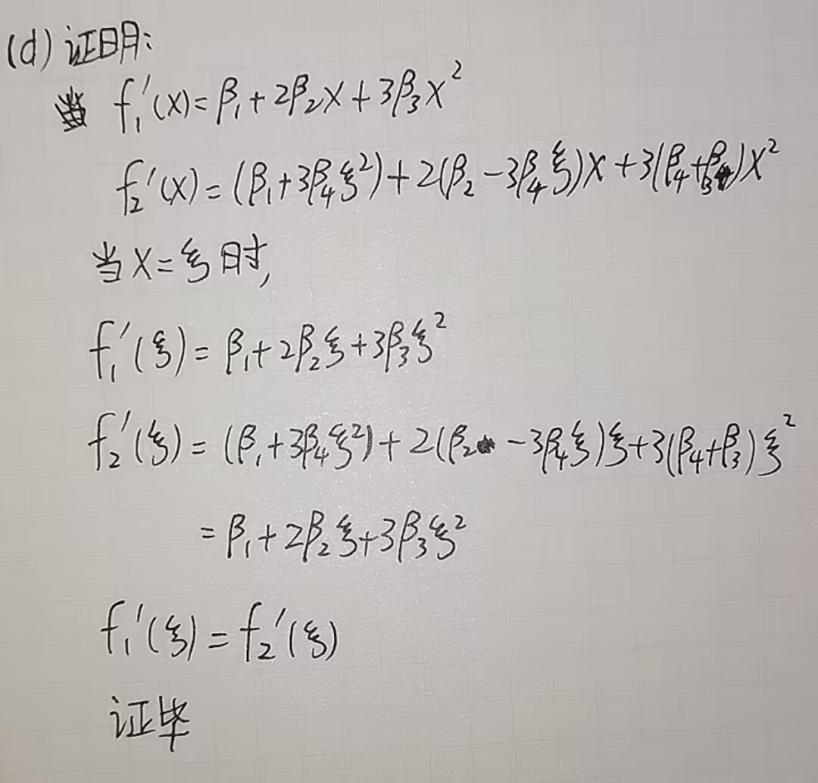
（b）问题（略）



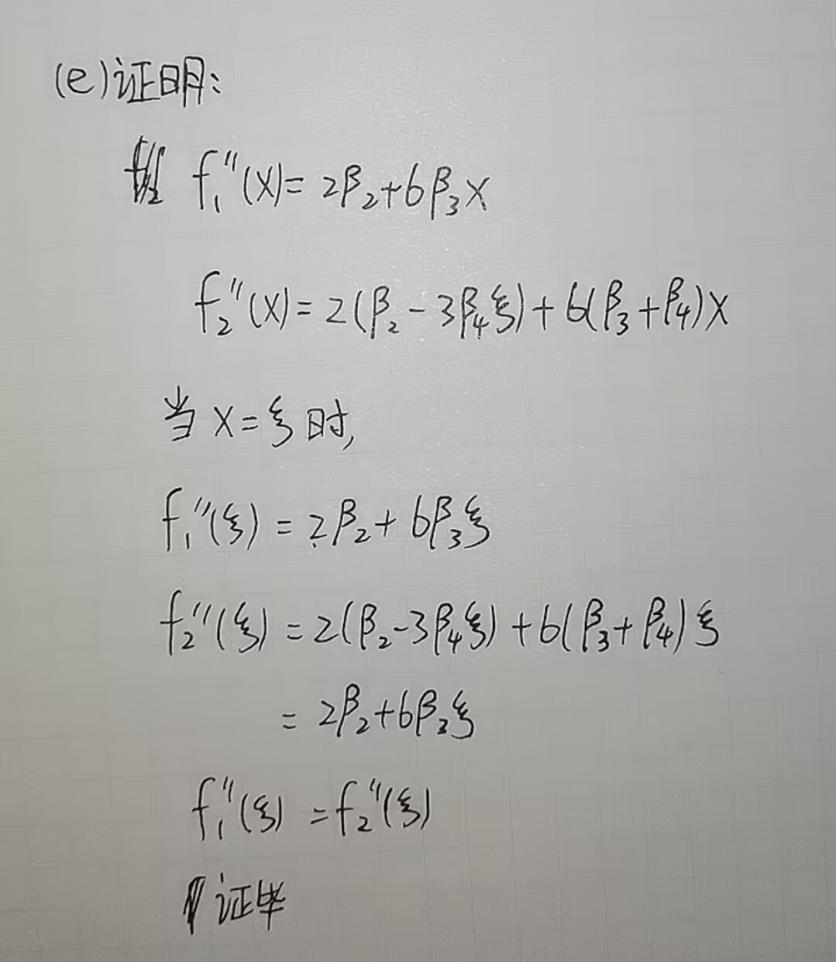
（c）问题（略）



（d）问题（略）

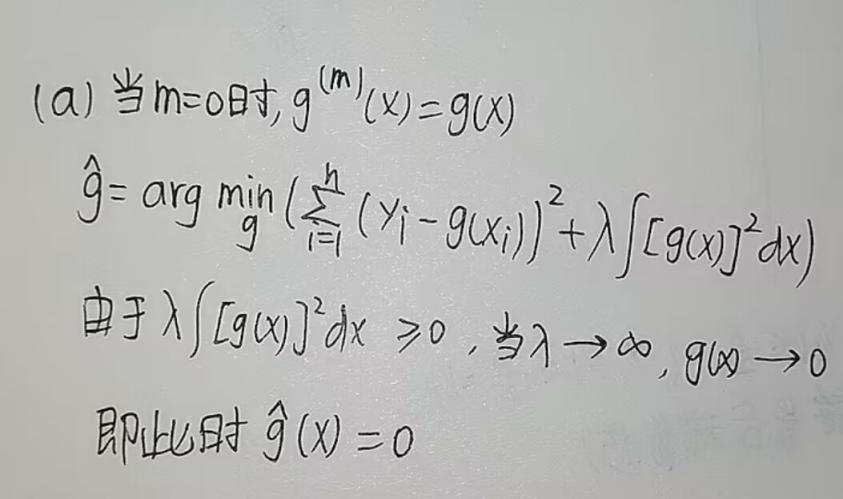


（e）问题（略）

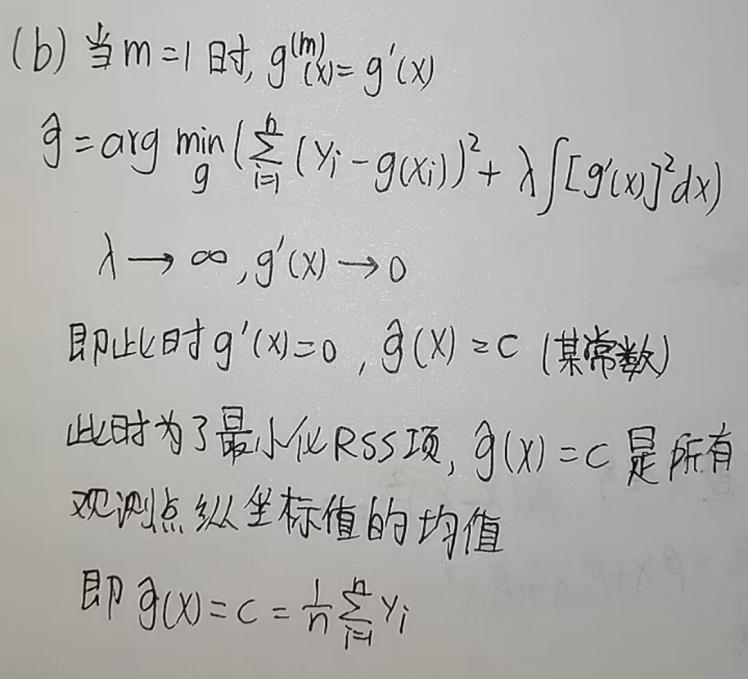


2.问题（略）

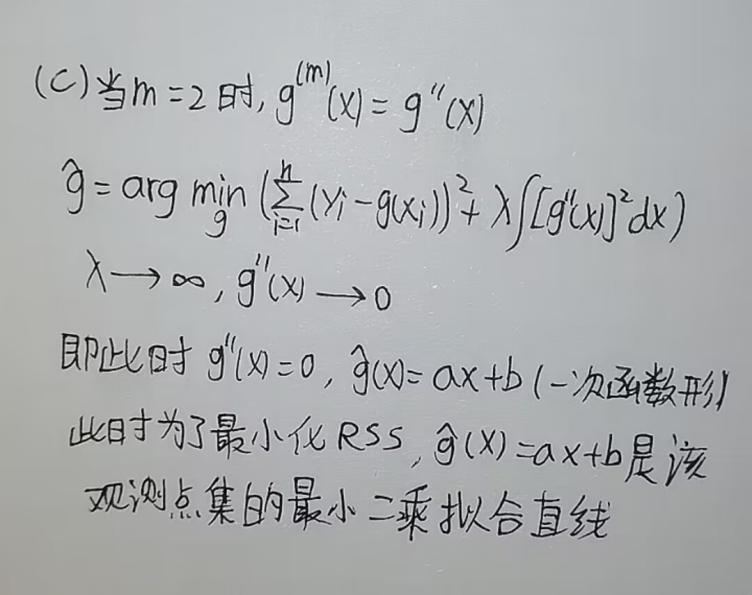
（a）问题（略）



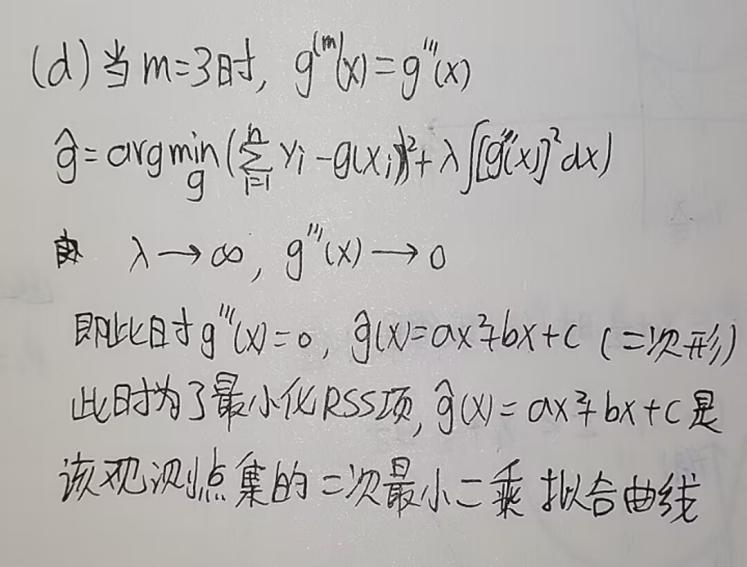
（b）问题（略）



（c）问题（略）



（d）问题（略）

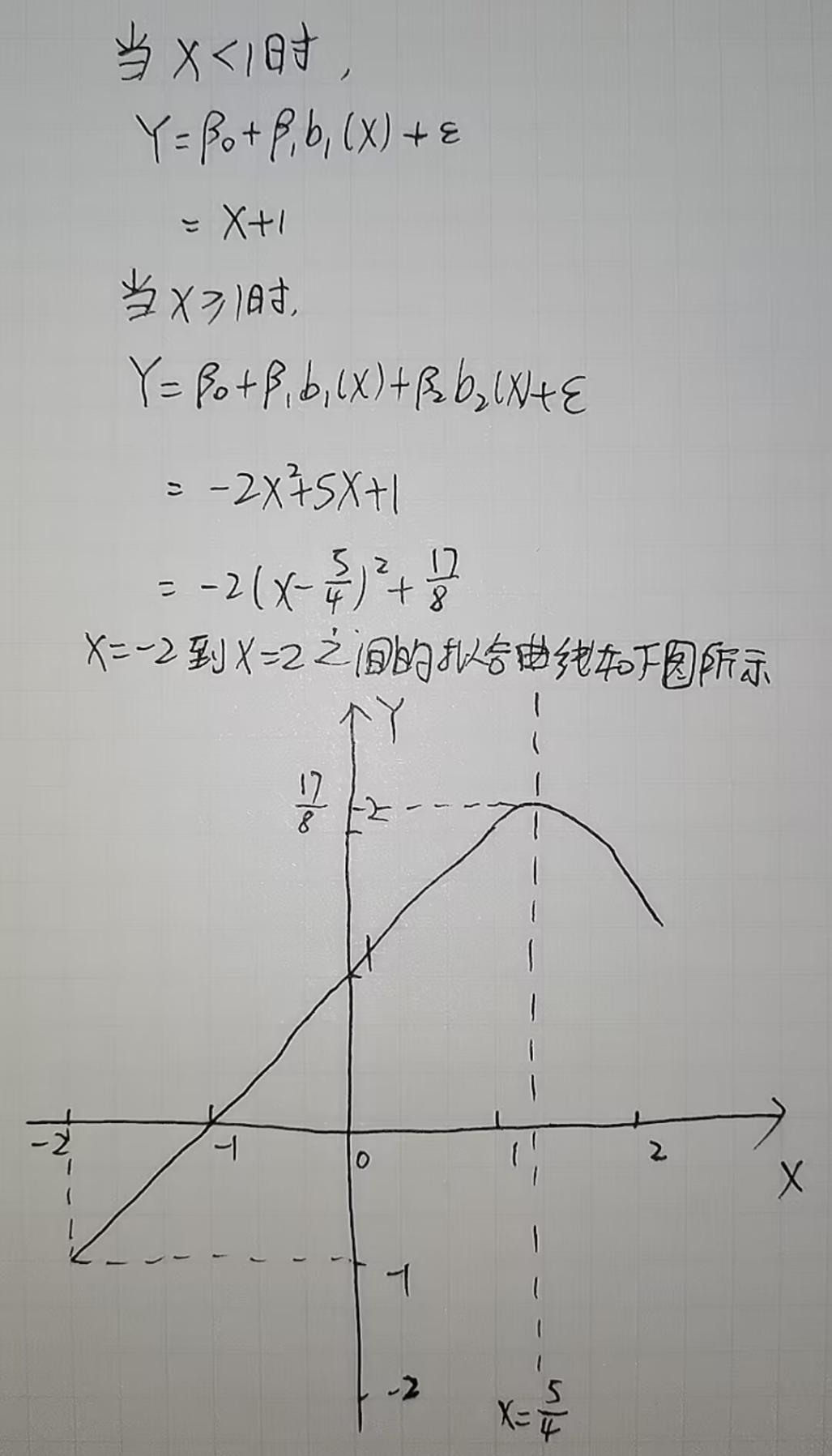


（e）问题（略）

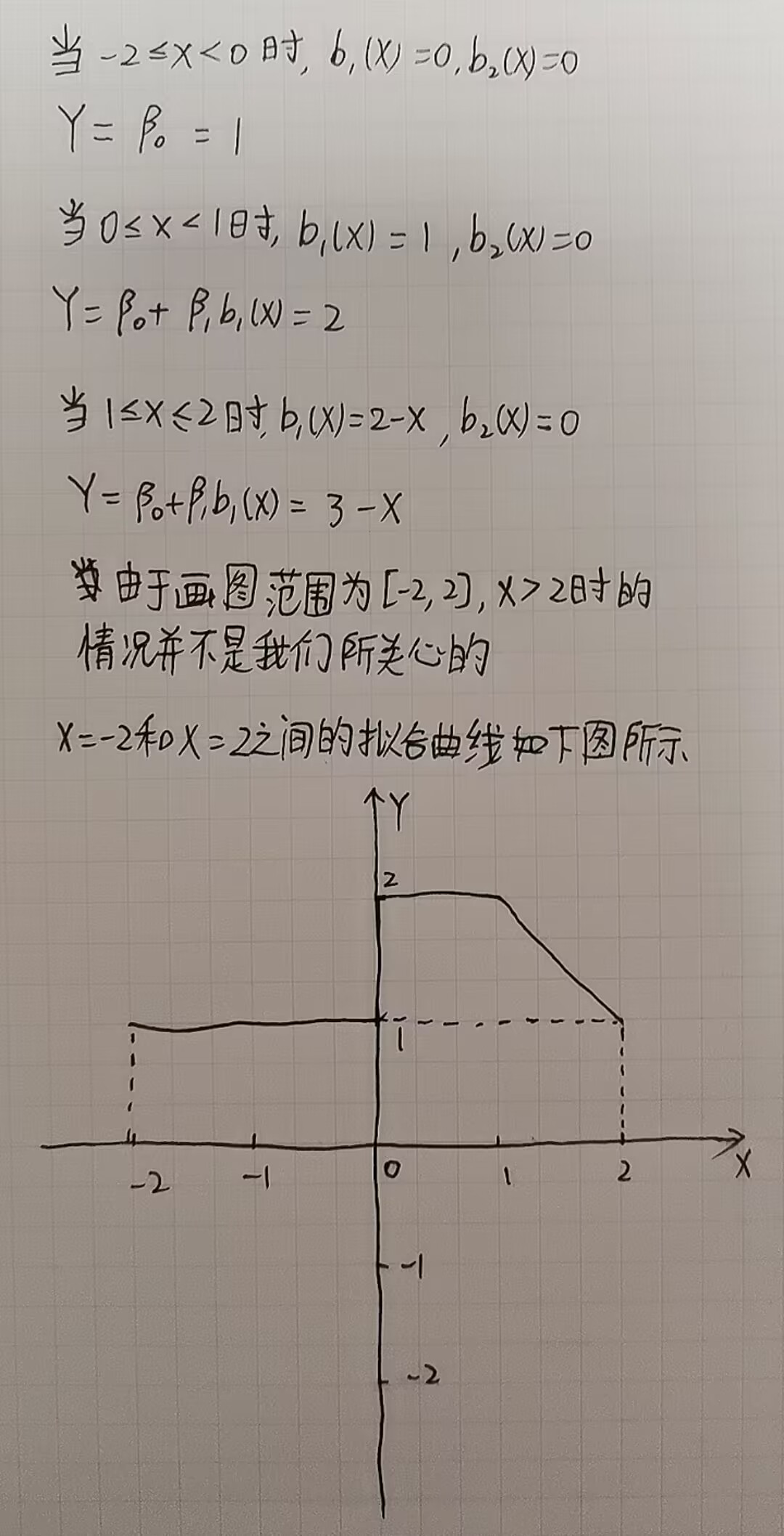
m=3但是λ=0

意味着惩罚项在g(x)中不再起作用，只有RSS一个指标。为了最小化RSS值，此时g(x)可以穿过所有观测点从而达到RSS=0的最低值。

3.问题（略）



4.问题（略）



5.问题（略）

（a）问题（略）

g2的训练RSS更小。

当λ->∞时，对于g1来说，，即三阶导数为0，g(x)=ax^2+bx+c(二次型)。此时为了最小化RSS项，g(x)=ax^2+bx+c将是该观测点集的二次最小二乘拟合曲线。

当λ->∞时，对于g2来说，，即四阶导数为0，g(x)=ax^3+bx^2+cx+d(三次型)。此时为了最小化RSS项，g(x)=ax^3+bx^2+cx+d将是该观测点集的三次最小二乘拟合曲线。

三次最小二乘拟合比二次最小二乘拟合灵活性更高，因而g2的训练RSS更小

（b）问题（略）

不一定。取决于x和y的真实关系是否更接近二次型或是更接近三次型。当x和y的关系更接近二次型的时候，g1的测试RSS更小；当x和y的关系更接近三次型的时候，g2的测试RSS更小。

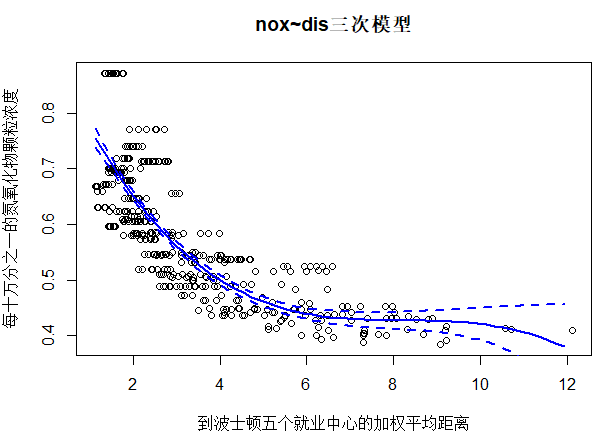
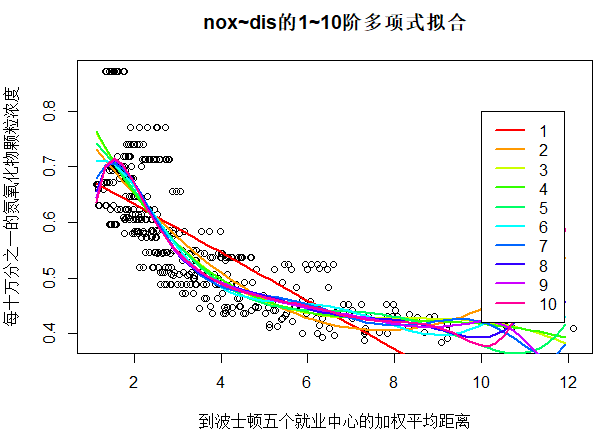
（c）问题（略）

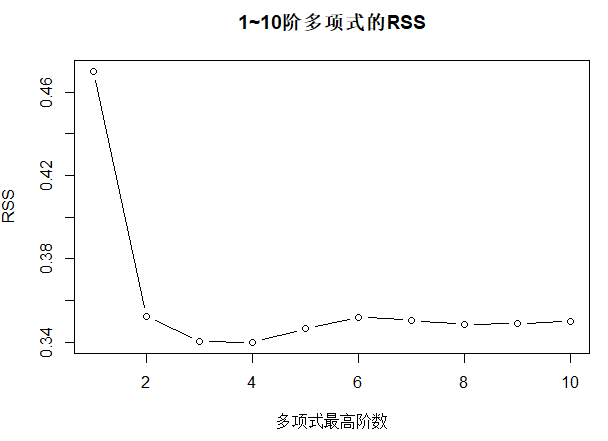
训练RSS相同，一定情况下测试RSS也会相同。

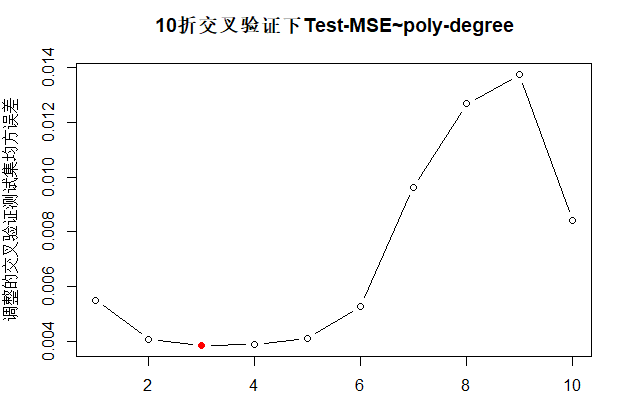
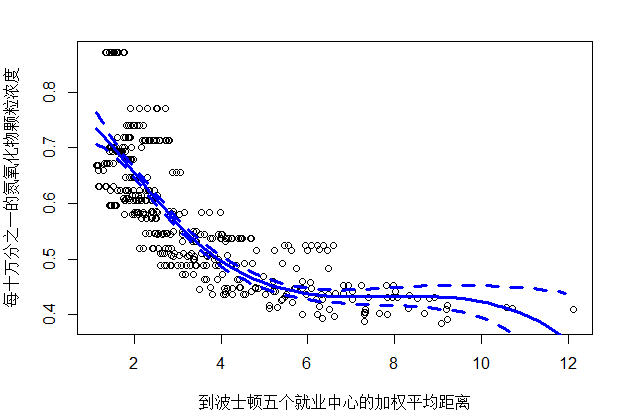
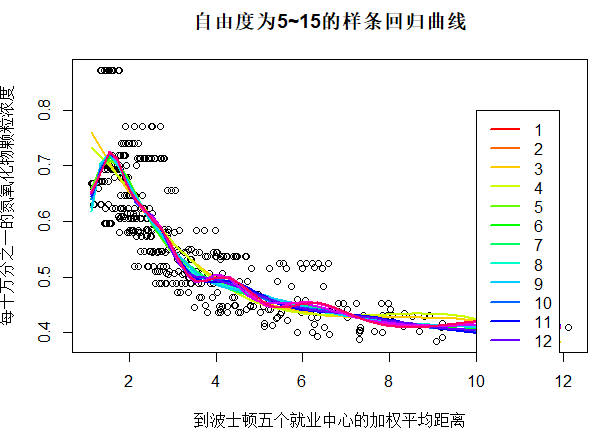
当λ=0时，意味着惩罚项不再起作用，对g(x)没有任何限制，g(x)函数曲线可以穿过每一个观测点来达到RSS最小值0。并且当g1和g2都这么做并且他们的测试集相同时，显然他们的测试RSS也会相同，并且由于严重过拟合，测试RSS会非常大。

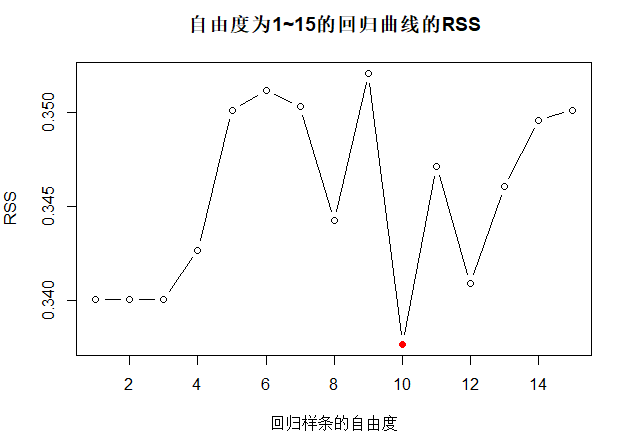
**应用**

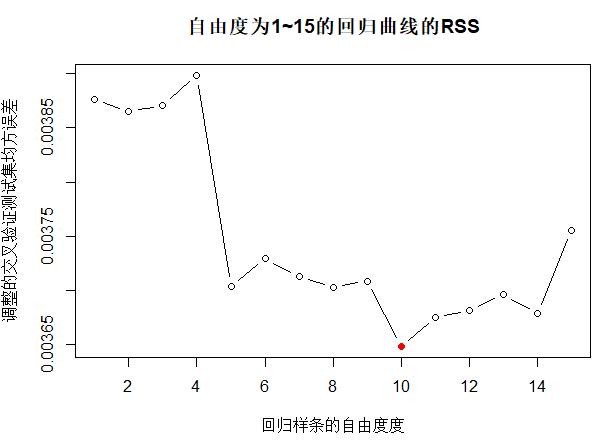
9.问题（略）

# 7.9.9
  
library(MASS)
  
library(caTools)
  
library(splines)
  
library(boot)
  
library(ggplot2)
  
  
（a）问题（略）  
# poly函数拟合nox~dis三次模型
  
glm.fit = glm(nox~poly(dis,3), data=Boston)
  
summary(glm.fit)
  
# dis区间序列
  
dis.grid = seq(from=min(Boston$dis),to=max(Boston$dis),0.2)
  
preds=predict(glm.fit,newdata=list(dis=dis.grid), se=T)
  
# 散点图
  
plot(Boston$dis,Boston$nox, main="nox~dis三次模型", xlab="到波士顿五个就业中心的加权平均距离", ylab="每十万分之一的氮氧化物颗粒浓度")
  
# 拟合曲线
  
lines(dis.grid,preds$fit,col="blue",lwd=2)
  
# 置信区间预测曲线
  
lines(dis.grid,preds$fit+2\*preds$se,col="blue",lwd=2,lty=2)
  
lines(dis.grid,preds$fit-2\*preds$se,col="blue",lwd=2,lty=2)
  
  
（b）问题（略）  
set.seed(123)
  
# 分割 训练集、测试集 dataframe 数据(随机4:1)
  
boston\_sample = sample.split(Boston$dis, SplitRatio = 0.80)
  
boston\_train = subset(Boston, boston\_sample==TRUE)
  
boston\_test = subset(Boston, boston\_sample==FALSE)
  
  
rss = rep(0,10)
  
colours = rainbow(10)
  
# 画散点图
  
plot(Boston$dis,Boston$nox, main="nox~dis的1~10阶多项式拟合", xlab="到波士顿五个就业中心的加权平均距离", ylab="每十万分之一的氮氧化物颗粒浓度")
  
# 循环：从1~10阶
  
for (i in 1:10){
  
 # 拟合i阶模型
  
 glm.fit2 = glm(nox~poly(dis,i), data=boston\_train)
  
  
 # 计算i阶模型的残差平方和
  
 rss[i] = sum((boston\_test$nox - predict(glm.fit2,newdata=list(dis=boston\_test$dis)))^2)
  
 # 预测曲线并画图
  
 preds=predict(glm.fit2,newdata=list(dis=dis.grid))
  
 lines(dis.grid,preds,col=colours[i], lwd=2, lty=1)
  
}
  
legend(10,0.8,legend=1:10, col= colours[1:10],lty=1,lwd=2)
  
# 测试集上的RSS
  
print(rss)
  
# 画测试集RSS曲线
  
plot(1:10,rss,xlab="多项式最高阶数", ylab="RSS", main="1~10阶多项式的RSS", type='b')
  
  
# RSS从线性模型的0.469降低到三次、四次模型的0.340左右，之后又慢慢上升
  


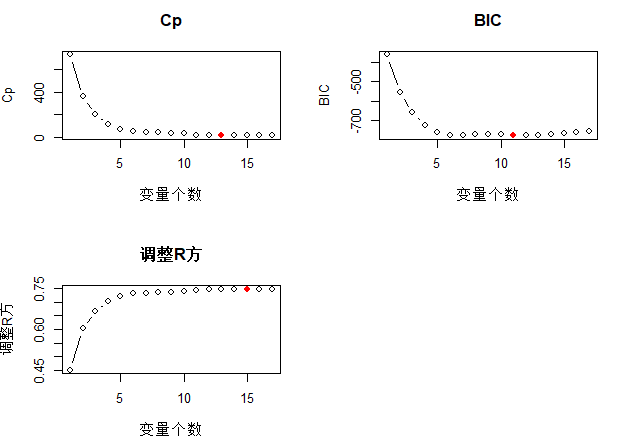


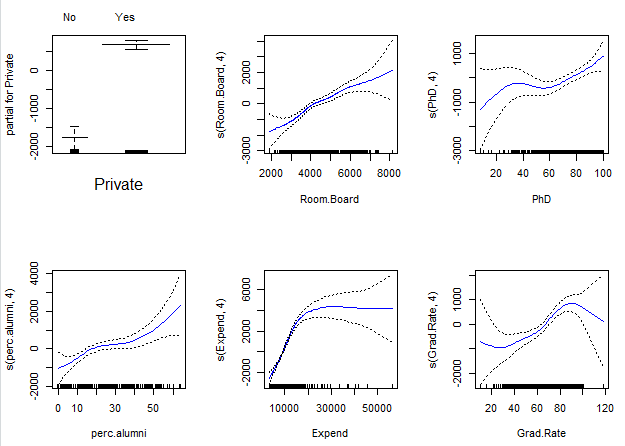
  
（c）问题（略）  
cv.err = rep(0, 10)
  
set.seed(123)
  
# 循环：从1~10阶
  
for (i in 1:10) {
  
 glm.fit = glm(nox ~ poly(dis, i), data = Boston)
  
 # 10折交叉验证
  
 # 结果包含原始交叉验证值和调整的交叉验证值，使用调整的交叉验证值更具说服力
  
 cv.err[i] = cv.glm(Boston, glm.fit, K = 10)$delta[2]
  
}
  
plot(1:10,cv.err,xlab="多项式最高阶数", ylab="调整的交叉验证测试集均方误差", main="10折交叉验证下Test-MSE~poly-degree", type='b')
  
# 红点标注最小值
  
points(which.min(cv.err), cv.err[which.min(cv.err)], col = "red", pch = 16)
  
  
# 通过标注MSE的最小点，我们发现当阶数为3时均方误差最小
  
# 故选择最高阶数为3最合适
  
  
（d）问题（略）  
# df=4 自由度为4的回归样条
  
spline.fit = lm(nox~bs(dis,df=4), data=Boston)
  
summary(spline.fit)
  
# 查看
  
attr(bs(Boston$dis,df=4),"knots")
  
# 当df=4时，只有一个结点(knots)，它的位置在整个dis点集的50%的位置，即中位数(3.20745)
  
# 注意，这个50%并不是dis的序列区间的50%位置，而是点集横坐标值的50%位置
  
  
# 散点图和预测曲线以及置信区间的曲线(和a问类似)
  
plot(Boston$dis,Boston$nox,xlab="到波士顿五个就业中心的加权平均距离", ylab="每十万分之一的氮氧化物颗粒浓度")
  
preds = predict(spline.fit, newdata=list(dis=dis.grid), se=T)
  
lines(dis.grid, preds$fit,col="blue",lwd=3)
  
lines(dis.grid, preds$fit+2\*preds$se,col="blue",lwd=3,lty=2)
  
lines(dis.grid, preds$fit-2\*preds$se,col="blue",lwd=3,lty=2)
  
  
（e）问题（略）  
# 本题尝试了自由度为1~15的一组回归样条进行拟合
  
# 主要关注自由度为5~15与自由度为4的回归样条进行比较
  
set.seed(123)
  
rss = rep(0,15)
  
colours = rainbow(15)
  
plot(Boston$dis,Boston$nox,xlab="到波士顿五个就业中心的加权平均距离", ylab="每十万分之一的氮氧化物颗粒浓度",main="自由度为5~15的样条回归曲线")
  
for (i in 1:15){
  
 spline.fit = lm(nox~bs(dis,df=i), data=boston\_train)
  
 rss[i] = sum((boston\_test$nox - predict(spline.fit,newdata=list(dis=boston\_test$dis)))^2)
  
 preds=predict(spline.fit,newdata=list(dis=dis.grid))
  
 lines(dis.grid,preds,col=colours[i], lwd=2, lty=1)
  
}
  
legend(10,0.8,legend=1:15, col=colours[1:15],lty=1,lwd=2)
  
# RSS曲线图
  
plot(1:15,rss,xlab="回归样条的自由度", ylab="RSS", main="自由度为1~15的回归曲线的RSS", type='b')
  
points(which.min(rss), rss[which.min(rss)], col = "red", pch = 16)
  
# 观察5~15的自由度回归样条曲线能够发现，由于自由度都比4更大，这一组曲线灵活程度都比df=4的回归样条更高
  
# 结果发现前期RSS先随自由度增大而升高，当自由度到达12后上下起伏不稳定
  


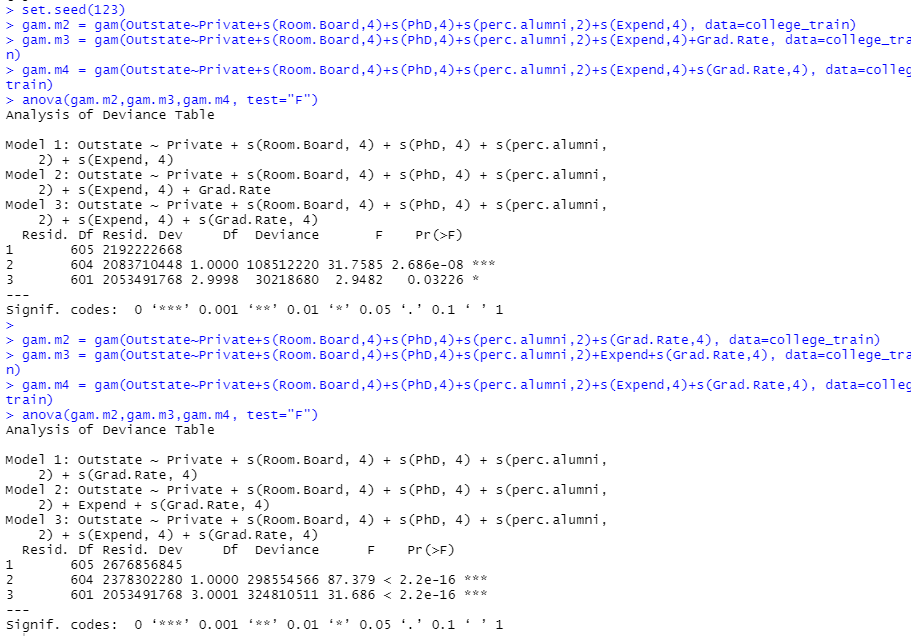
  
（f）问题（略）  
# 与(c)问一样，把poly函数换成bs即可
  
set.seed(123)
  
cv.err = rep(0, 15)
  
# 循环：从1~10阶
  
for (i in 1:15) {
  
 glm.fit = glm(nox ~ bs(dis, i), data = Boston)
  
 # 10折交叉验证
  
 # 结果包含原始交叉验证值和调整的交叉验证值，使用调整的交叉验证值更具说服力
  
 cv.err[i] = cv.glm(Boston, glm.fit, K = 10)$delta[2]
  
}
  
plot(1:15,cv.err,xlab="回归样条的自由度度", ylab="调整的交叉验证测试集均方误差", main="自由度为1~15的回归曲线的RSS", type='b')
  
# 红点标注最小值
  
points(which.min(cv.err), cv.err[which.min(cv.err)], col = "red", pch = 16)
  
# 发现自由度为10的时候，交叉验证均方误差最小
  
# 故自由度为10的时候模型最合适



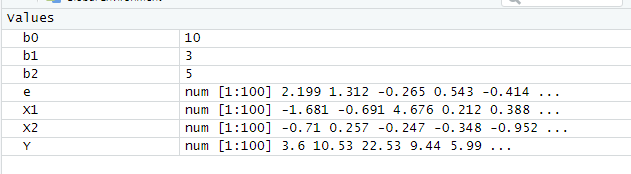
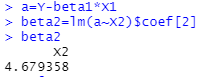
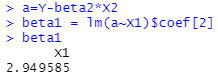
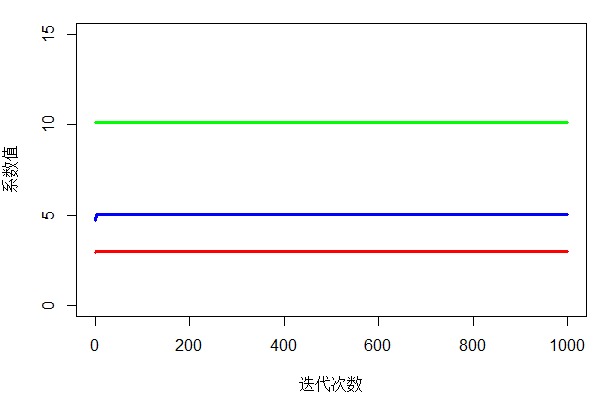
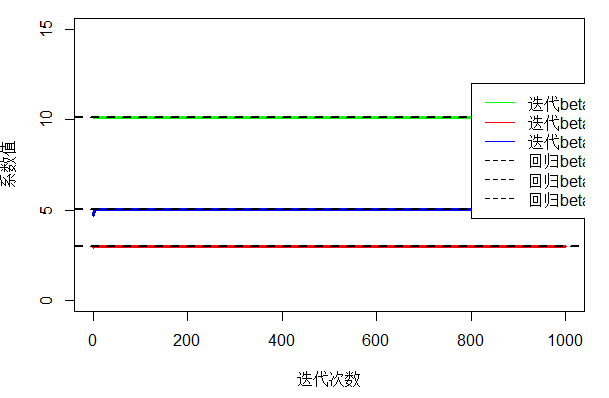
10.问题（略）

# 7.9.10
  
library(ISLR)
  
library(leaps)
  
library(gam)
  
（a）问题（略）  
set.seed(123)
  
  
# 前向逐步选择，最大变量个数为17
  
fit.fwd = regsubsets(Outstate~., data=college\_train, nvmax=17, method="forward")
  
fit.summary = summary(fit.fwd)
  
# 画图: Cp,BIC,调整R方, 以及最大/最小点
  
par(mfrow=c(2,2))
  
plot(1:17, fit.summary$cp,xlab="变量个数",ylab="Cp",main="Cp", type='b')
  
points(which.min(fit.summary$cp), fit.summary$cp[which.min(fit.summary$cp)], col = "red", pch = 16)
  
plot(1:17, fit.summary$bic,xlab="变量个数",ylab="BIC",main="BIC", type='b')
  
points(which.min(fit.summary$bic), fit.summary$bic[which.min(fit.summary$bic)], col = "red", pch = 16)
  
plot(1:17, fit.summary$adjr2,xlab="变量个数",ylab="调整R方",main="调整R方", type='b')
  
points(which.max(fit.summary$adjr2), fit.summary$adjr2[which.max(fit.summary$adjr2)], col = "red", pch = 16)
  
  
coef(fit.fwd,6)
  
# 通过趋势图能够发现三种方式都在变量数为6时到达比较稳定理想的状态
  
# 故选择变量数为6拟合效果良好，同时变量个数不会过于繁杂
  


  
（b）问题（略）  
# 划分训练集与测试集(随机4:1)
  
college\_df = College
  
college\_sample = sample.split(college\_df$Outstate, SplitRatio = 0.80)
  
college\_train = subset(college\_df, college\_sample==TRUE)
  
college\_test = subset(college\_df, college\_sample==FALSE)
  
# 将(a)问得出来的6个预测变量加入，使用光滑样条并设定自由度为4
  
gam.m1 = gam(Outstate~Private+s(Room.Board,4)+s(PhD,4)+s(perc.alumni,4)+s(Expend,4)+s(Grad.Rate,4), data=college\_train)
  
# 画出每个预测变量的函数对outstate的影响图示
  
par(mfrow=c(2,3))
  
plot(gam.m1, col="blue", se=T)
  
  
# 在其他变量不变的情况下，州外学费(outstate)随着Room.Board和perc.alumni单增
  
# 同时在其他变量的影响下也大体呈上涨趋势
  
  
（c）问题（略）  
# 测试集上预测并计算测试集均方误差
  
preds = predict(gam.m1,newdata = college\_test)
  
mse = mean((college\_test$Outstate - preds)^2)
  
mse
  
  
（d）问题（略）  
set.seed(123)
  
gam.m2 = gam(Outstate~Private+s(Room.Board,4)+s(PhD,4)+s(perc.alumni,2)+s(Expend,4), data=college\_train)
  
gam.m3 = gam(Outstate~Private+s(Room.Board,4)+s(PhD,4)+s(perc.alumni,2)+s(Expend,4)+Grad.Rate, data=college\_train)
  
gam.m4 = gam(Outstate~Private+s(Room.Board,4)+s(PhD,4)+s(perc.alumni,2)+s(Expend,4)+s(Grad.Rate,4), data=college\_train)
  
anova(gam.m2,gam.m3,gam.m4, test="F")
  
  
gam.m2 = gam(Outstate~Private+s(Room.Board,4)+s(PhD,4)+s(perc.alumni,2)+s(Grad.Rate,4), data=college\_train)
  
gam.m3 = gam(Outstate~Private+s(Room.Board,4)+s(PhD,4)+s(perc.alumni,2)+Expend+s(Grad.Rate,4), data=college\_train)
  
gam.m4 = gam(Outstate~Private+s(Room.Board,4)+s(PhD,4)+s(perc.alumni,2)+s(Expend,4)+s(Grad.Rate,4), data=college\_train)
  
anova(gam.m2,gam.m3,gam.m4, test="F")
  
  
# 从图中观察发现Expend和Grad.Rate和非线性接近
  
# 使用ANOVA方法进行验证：在广义可加模型中不添加、添加一次项以及四次项的目标变量
  
# 观察模型中该预测变量的显著程度
  
# 结果提供了令人信服的证据：
  
# s(Grad.Rate,4)的p值为0.03<0.05，s(Expend,4)的p值<2.2e-16
  
# 即证明Expend和Grad.Rate和响应变量有明显的非线性关系



11.问题（略）

# 7.9.11
  
rm(list=ls())
  
（a）问题（略）  
set.seed(123)
  
# 随机生成数据
  
X1 = rnorm(100, sd=3)
  
X2 = rnorm(100, sd=1)
  
e = rnorm(100, sd=1)
  
b0 =10; b1=3 ; b2=5
  
Y = b0 +b1\*X1 + b2\*X2 + e
  
  
（b）问题（略）  
# 随机指定β1
  
beta1 = 0.5
  
  
（c）问题（略）  
a=Y-beta1\*X1
  
beta2=lm(a~X2)$coef[2]
  
beta2
  
# β2=4.679358
  
  
（d）问题（略）  
a=Y-beta2\*X2
  
beta1 = lm(a~X1)$coef[2]
  
beta1
  
# β1=2.949585
  
  
（e）问题（略）  
beta.df = data.frame("beta0"=rep(0,1000),"beta1"=rep(0,1000),"beta2"=rep(0,1000))
  
beta1=0.5
  
# 迭代1000次
  
for (i in 1:1000){
  
 # β1 => β2
  
 a=Y-beta1\*X1
  
 model = lm(a~X2)
  
 beta2 = model$coef[2]
  
 # 记录系数值
  
 beta.df$beta2[i]= beta2
  
 # β2 => β1
  
 a=Y-beta2\*X2
  
 model = lm(a~X1)
  
 beta1 = model$coef[2]
  
 # 记录系数值
  
 beta.df$beta1[i]=beta1
  
  
 # 记录系数值
  
 beta.df$beta0[i]=model$coef[1]
  
}
  
# 画曲线图
  
plot(1:1000,beta.df$beta2,ylim=range(0:15),type='l', lwd="3", col="blue",xlab="迭代次数",ylab="系数值")
  
lines(1:1000,beta.df$beta1, col="red", lwd=3)
  
lines(1:1000,beta.df$beta0, col="green",lwd=3)
  
  
（f）问题（略）  
# 获取迭代结果
  
print(beta.df$beta0[1000])
  
print(beta.df$beta1[1000])
  
print(beta.df$beta2[1000])
  
# abline画线
  
abline(h=10.13507, lty=2, lwd=2, col=1)
  
abline(h=2.955609, lty=2, lwd=2, col=1)
  
abline(h=5.023811, lty=2, lwd=2, col=1)
  
legend(800,12, legend=c("迭代beta0", "迭代beta1", "迭代beta2", "回归beta0", "回归beta1", "回归beta2"),col=c("green","red","blue", 1, 1, 1), lty = c(1,1,1,2,2,2))
  
  
（g）问题（略）  
# 抽取前10次的结果来观察
  
plot(1:10,beta.df$beta2[1:10],ylim=range(0:15),type='l', lwd="3", col="blue",xlab="迭代次数",ylab="系数值")
  
lines(1:10,beta.df$beta1[1:10], col="red", lwd=3)
  
lines(1:10,beta.df$beta0[1:10], col="green",lwd=3)
  
  
# 发现当迭代到第2~3次时，就已经基本迭代完成
  
  
# 再将迭代到第2~3次的系数值与多元线性回归的拟合系数值对比
  
print(beta.df$beta0[3])
  
print(beta.df$beta1[3])
  
print(beta.df$beta2[3])
  
lm.fit = lm(Y~X1+X2)
  
coef(lm.fit)
  
  
# 系数的差距非常小，即逼近效果很好
  
# 故需要三次迭代就能很好地逼近多元线性回归的结果

