统计模型及其 R 实现笔记

Statistical Models with R

马学俊

献给我的家人、恩师和所有在学术道路上帮助我的人

目录

第一部分

面板数据

第 1 章 固定效应和随机效应模型

面板数据分析一直以来是计量经济学中的重要组成部分。

1.1 引言

EmplUK 描述英国 (United Kingdom) 的工人和工资 (Employment, Wages) 从 1976 年到 1984 年 140 个体的情况，共计有 1031 个观测值。需要注意的时，这组数据是不平衡的，因为不同组的观测值不同，也就是说不通的个体的观察值不同。从下面的输出结果可以看出，第 1–103 个体有 7 个观测值，第 104–127 个体有 8 个观测值，其它个体有 9 个观测值。从年份的观察频数也可以看出数据是不平衡的。

R 代码

```
library(plm)
?EmplUK
data("EmplUK",package="plm")
head(EmplUK)#查看数据前6行
attach(EmplUK)
table(year)
table(firm)
```

输出结果

```
> head(EmplUK)#查看数据前6行
  firm year sector   emp    wage capital   output
1    1 1977      7 5.041 13.1516  0.5894  95.7072
2    1 1978      7 5.600 12.3018  0.6318  97.3569
3    1 1979      7 5.015 12.8395  0.6771  99.6083
4    1 1980      7 4.715 13.8039  0.6171 100.5501
5    1 1981      7 4.093 14.2897  0.5076  99.5581
6    1 1982      7 3.166 14.8681  0.4229  98.6151
> table(year)
1976 1977 1978 1979 1980 1981 1982 1983 1984
  80  138  140  140  140  140  140   78   35
> table(firm)
  1   2   3   4   5   6   7   8   9  10  11  12  13  14
  7   7   7   7   7   7   7   7   7   7   7   7   7   7
 15  16  17  18  19  20  21  22  23  24  25  26  27  28
  7   7   7   7   7   7   7   7   7   7   7   7   7   7
 29  30  31  32  33  34  35  36  37  38  39  40  41  42
  7   7   7   7   7   7   7   7   7   7   7   7   7   7
 43  44  45  46  47  48  49  50  51  52  53  54  55  56
```

```
  7   7   7   7   7   7   7   7   7   7   7   7   7   7
 57  58  59  60  61  62  63  64  65  66  67  68  69  70
  7   7   7   7   7   7   7   7   7   7   7   7   7   7
 71  72  73  74  75  76  77  78  79  80  81  82  83  84
  7   7   7   7   7   7   7   7   7   7   7   7   7   7
 85  86  87  88  89  90  91  92  93  94  95  96  97  98
  7   7   7   7   7   7   7   7   7   7   7   7   7   7
 99 100 101 102 103 104 105 106 107 108 109 110 111 112
  7   7   7   7   7   8   8   8   8   8   8   8   8   8
113 114 115 116 117 118 119 120 121 122 123 124 125 126
  8   8   8   8   8   8   8   8   8   8   8   8   8   8
127 128 129 130 131 132 133 134 135 136 137 138 139 140
  9   9   9   9   9   9   9   9   9   9   9   9   9   9
```

表1.1摘要 `EmplUK` 数据的变量的名称、类型等。我们的目的是分析 `wage`、`capital` 对 `emp` 的影响。

表 1.1: `EmplUK` 数据摘要

变量名	描述	类型
`firm`	工厂	分类
`year`	年	分类
`sector`	the sector of activity	分类
`emp`	employment	连续
`wage`	工资	连续
`capital`	资本	连续
`output`	输出	连续

1.2 模型介绍和 R 实现

`EmplUK` 数据是面板数据，它包含个时间和个体两个维度。这一章，我们主要介绍线性面板数据模型：

$$y_{it} = \alpha_{it} + \boldsymbol{\beta}_{it}^{\top}\boldsymbol{x}_{it} + u_{it} \tag{1.1}$$

其中 $\boldsymbol{x}_{it} = (x_{1,it}, \ldots, x_{p,it})^{\top}$，$\boldsymbol{\beta}_{it} = \beta_{1,it}, \ldots, \beta_{p,it}$；$i = 1, \ldots, n$ 表示个体 (Subject)，如个人、城市、国家等；$t = 1, \ldots, T$ 是时间 (Time)。$\mathrm{E}(u_{it}|\boldsymbol{x}_{it}) = 0$。记 $N = n \times T$。模型1.1是一个非常一般的模型，根据需要，它的变化种类很多。通常存在如下变形：

- 池子模模型 (Pooling Model)，又称无效应模型 (None Effect Model)：

  $$y_{it} = \alpha + \boldsymbol{\beta}^{\top}\boldsymbol{x}_{it} + u_{it} \tag{1.2}$$

  将所有数据放到一个池子里面“同等”对待，所以称为池子模型；不存在个体和时间效应，所以称为无效应模型，即模型1.1：α_{it} 和 $\boldsymbol{\beta}_{it}$ 不随个体 i 和时间 t 变化
- 固定效应 (Fixed Model)：

- 个体效应模型 (Individual model)：模型1.1中 $\boldsymbol{\beta}_{it} = \beta_i$ 不随时间 t 变化。

$$y_{it} = \alpha_i + \boldsymbol{\beta}^\top \boldsymbol{x}_{it} + u_{it} \tag{1.3}$$

  这一章的固定效应，我们默认为个体性效应模型.

- 时间效应模型 (Individual model)：模型1.1中 $\boldsymbol{\beta}_{it} = \beta_t$ 不随个体 i 变化。

$$y_{it} = \alpha_t + \boldsymbol{\beta}^\top \boldsymbol{x}_{it} + u_{it} \tag{1.4}$$

- 随机效应 (Fandom Model)：模型1.1中 $\boldsymbol{\beta}_{it} = \beta + \eta_i$。它的效应是随机的，因为 η_i 不可观测。

我们以 EmplUK 数据为例讲解固定效应和随机效应模型在 R 的实现。

R 代码

```
library(plm)
data("EmplUK",package="plm")
EmplUK <- plm.data(EmplUK, index = c("firm", "year"))
fit.fe <- plm(emp ~ wage + capital, data = EmplUK, model = "within")#固定效应
summary(fit.fe)#
summary(fixef(fit.fe))#查看个体的效应
fit.re <- plm(emp ~ wage + capital, data = EmplUK, model = "random")#随机效应
summary(fit.re)
```

输出结果

```
> summary(fit.fe)#
Oneway (individual) effect Within Model

Call:
plm(formula = emp ~ wage + capital, data = EmplUK, model = "within")

Unbalanced Panel: n=140, T=7-9, N=1031

Residuals :
    Min.  1st Qu.   Median  3rd Qu.     Max.
-17.1000  -0.3060   0.0137   0.3070  27.3000

Coefficients :#没有截距项
         Estimate Std. Error t-value  Pr(>|t|)
wage    -0.143626   0.032790 -4.3802 1.327e-05 ***
capital  0.801495   0.064088 12.5062 < 2.2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Total Sum of Squares:    5030.6
Residual Sum of Squares: 4207.8
R-Squared:      0.16356
```

```
Adj. R-Squared: 0.14103
F-statistic: 86.9179 on 2 and 889 DF, p-value: < 2.22e-16
> summary(fixef(fit.fe))#查看个体的效应
    Estimate Std. Error t-value  Pr(>|t|)
1    5.87703    0.93545  6.2826 3.330e-10 ***
2   59.87144    1.45426 41.1697 < 2.2e-16 ***
3   17.07241    1.20391 14.1808 < 2.2e-16 ***
4   21.49019    1.12716 19.0658 < 2.2e-16 ***
5   68.58195    1.85658 36.9400 < 2.2e-16 ***
6    3.90465    1.11012  3.5173 0.0004359 ***
> summary(fit.re)
Oneway (individual) effect Random Effect Model
   (Swamy-Arora's transformation)

Call:
plm(formula = emp ~ wage + capital, data = EmplUK, model = "random")

Unbalanced Panel: n=140, T=7-9, N=1031

Effects:
                 var std.dev share
idiosyncratic  4.733   2.176 0.061
individual    72.655   8.524 0.939
theta  :
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
 0.9040  0.9040  0.9040  0.9064  0.9101  0.9152

Residuals :
    Min.  1st Qu.   Median     Mean  3rd Qu.     Max.
-14.0000  -0.6260  -0.3390  -0.0113   0.0856  29.9000

Coefficients :
             Estimate Std. Error t-value  Pr(>|t|)
(Intercept)  9.072269   1.116282  8.1272 1.253e-15 ***
wage        -0.157677   0.033526 -4.7031 2.912e-06 ***
capital      1.132022   0.059248 19.1066 < 2.2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Total Sum of Squares:    7396.1
Residual Sum of Squares: 5366.2
R-Squared:      0.27448
Adj. R-Squared: 0.27369
```

```
F-statistic: 194.44 on 2 and 1028 DF, p-value: < 2.22e-16
```

用 fixef 函数调出固定效应中的个体效应。需要注意的是固定效应没有截距项，因为每一个个体都需要估计，截距项包含着个体效应中提取不出来。随机效应的估计方法 plm 函数可以实现四种，详见 plm 帮助文档。

Hausman 检验主要对比固定效应模型和随机效应模型。

R 代码

```
summary(fit.re)
phtest(fit.fe,fit.re)
```

输出结果

```
Hausman Test
data:  emp ~ wage + capital
chisq = 181.68, df = 2, p-value < 2.2e-16
alternative hypothesis: one model is inconsistent
```

说明拒绝原假设，说明两个模型存在显著差异。

第 2 章 异方差的面板数据分析

`phtt` 可以实现。

2.1 参考文献

1. Croissant, Y., & Millo, G. (2008). Panel data econometrics in R: The plm package. Journal of Statistical Software, 27(2), 1-43.

第 3 章 广义线性模型

广义线性模型 (Generalized Linear Models) 是线性模型的推广，假设模型的误差项来自于指数分布族 (Exponential family of distributions)。在研究模型之前，我们先介绍指数分布族的概念。

3.1 指数分布族

若 Y 是随机变量，并且其密度函数或者质量函数 (Probability density or mass function) 有如下形式：

$$f(y|\theta,\phi) = \exp\left\{\frac{y\theta - b(\theta)}{a(\psi)} + c(y,\theta)\right\} \tag{3.1}$$

其中：

- θ 是感兴趣自然参数 (natural parameter)。
- ψ 是附加的参数或者冗余参数 (nuisance/dispersion parameter).
- $a(\cdot), b(\cdots)$ 和 $c(\cdots)$ 是已知函数。

则称 Y 是指数分布族。

自然参数是我们实际关心的参数，比如均值参数 μ。冗余参数是所以是冗余，因为它不是我们感兴趣的。我们认为它多余，比如方差。其实这里的多余只是我们此时不需要它，并不代表它不重要。已知函数表示函数的形式已知，比如单位映射。不同的分布他们的形式不一样。下面研究常见的几种指数分布族的分布。

例题 3.1 正态分布 (Normal distribution) 设 $Y \sim N(\mu,\sigma)$，则它的分布函数是

$$\begin{aligned}
f(y|\theta,\phi) &= \frac{1}{\sqrt{2\pi\sigma^2}}\exp\left\{-\frac{(y-\mu)^2}{2\sigma^2}\right\} \\
&= \exp\left\{-\frac{(y-\mu)^2}{2\sigma^2} - \frac{1}{2}\log(2\pi\sigma^2)\right\} \\
&= \exp\left\{-\frac{(y^2 2y\mu + \mu^2)}{2\sigma^2} - \frac{1}{2}\log(2\pi\sigma^2)\right\} \\
&= \exp\left\{-\frac{y\mu + \frac{\mu^2}{2}}{\sigma^2} + \left[-\frac{y^2}{2\sigma^2} - \frac{1}{2}\log(2\pi\sigma^2)\right]\right\}
\end{aligned}$$

对照 (3.1)，我们可以得到：

$$\begin{aligned}
\theta &= \mu \\
a(\phi) &= \sigma^2 \\
b(\theta) &= \frac{\mu^2}{2} = \frac{\theta^2}{2} \\
c(y,\phi) &= -\frac{y^2}{2\sigma^2} - \frac{1}{2}\log(2\pi\sigma^2).
\end{aligned}$$

参考文献

[Fan and Li(2001)] Fan J. and Li R. (2001) Variable selection via nonconcave penalized likelihood and its oracle properties. Journal of the American Statistical Association, 96(456): 1348–1360.

第 4 章 变量选择

假设回归模型是

$$Y = X^\top \beta + \varepsilon \tag{4.1}$$

其中 Y 是一维随机变量，$X = (X_1, \ldots, X_p)$ 是 p 维随机变量，ε 是一维随机变量。$\beta = (\beta_1, \ldots, \beta_p)^\top$ 是未知参数。假设

$$E(\varepsilon | X = x) = 0, \tag{4.2}$$

模型 (4.1) 可以表示为：

$$E(Y | X = x) = x^\top \beta. \tag{4.3}$$

上述模型是均值回归 (Mean regression)，其参数可以通过下面得到:

$$\min E(Y - x\beta)^2 \tag{4.4}$$

备注 假条条件 (4.2) 不同，可以得到不同类别的统计模型，比如分位数回归 (Quantile regression) 和众数回归 (Mode regression)。这里主要讨论均值回归。

假设 $(\boldsymbol{x}_1, y_1)$,...,$(\boldsymbol{x}_n, y_n)$ 是一组样本，其中 $\boldsymbol{x}_i = (x_{i1}, \ldots, x_{ip})$，表达式 (4.4) 的样本实现值为

$$\min_\beta \sum_{i=1}^n (y_i - \boldsymbol{x}_i^\top \beta)^2 \tag{4.5}$$

经过简单运算，$\hat{\beta}_{ols} = (\boldsymbol{x}^\top \boldsymbol{x})^{-1} \boldsymbol{x}^\top y$，其中 $y = (y_1, \ldots, y_n)^\top$。$\boldsymbol{x} = (\boldsymbol{x}_1^\top, \ldots, \boldsymbol{x}_n^\top)^\top$ 是 $n \times p$ 的设计矩阵 (Design matrix，矩阵是设计好的，换句话为是给定，直白点说就是已知的) 在条件 (4.2)，$\hat{\beta}_{ols}$ 是无偏估计（注意没有其它假设条件，如果想证明其它的性质，如渐近性质或，需要其它条件。

现在，如果 $\boldsymbol{x}^\top \boldsymbol{x}$ 不可逆，也就是说 $\boldsymbol{x}$ 不是列满秩，那么 $\hat{\beta}_{ols}$ 的不存在。上面这种线性成为完全共线性。这也是为什么研究变量选择的一个重要原因。下面我们来讨论另一个原因，假如研究儿童身高的影响因素，我们收集了性别、体重、父亲体重、母亲体重、家里花草的数量等几百个因素，目的是找到主要影响因素。大家注意，我们这里其实有一个假设，那就是儿童身高的影响因素是很少的，也只有几个。换句统计的词汇就是 “稀疏性假设”。“家里花草的数量” 显然就是需要排除的因素。排除因素就是变量选择。除了上述原因外，还有

- 估计量的方差变大，预测的精度较低；
- 过拟合，保留大量的解释变量会降低模型的可解释性。

怎么进行变量选择或者消去共线性，我们学习了很多方法，比如最有子集方法 (best subset method)，逐步回归和岭回归 (Ridge regression) 等。下面简单介绍这几种方法：

最优子集方法对 p 个变量的所有可能组合分别进行拟合，选择残差平方和 (Residual square sum) 或者 R^2 最小的模型。最优子集的优点是简单直观，但效率太低，当 p 很大时，从一个巨大的搜索空间中得到的模型通常会有过拟合和系数估计方差高的问题；改进的子集选择还有逐步选择 (向前、向后)，与全子集相比限制了搜索空间，提高了运算效率，但是无法保证找到的模型是 2^p 个模型中最优的。

岭回归是求带有约束的凸优化问题：

$$\min_\beta \sum_{i=1}^n (y_i - \boldsymbol{x}_i^\top \beta)^2 \tag{4.6}$$

$$\text{s.t.} \quad \sum_{j=1}^p \beta_j^2 \le t \tag{4.7}$$

其中 s.t. 是 subject to 的缩写，表示约束条件。$t > 0$。引入 Lagrange 乘子可以转化为，上面的优化问题可以转化为

$$\min_{\beta} \left(\sum_{i=1}^{n} (y_i - \boldsymbol{x}_i^\top \beta)^2 + \lambda \sum_{j=1}^{p} \beta_j^2 \right) \tag{4.8}$$

岭回归得到的估计量是有偏的，但方差小了，得到的均方误差小，也就是其牺牲了无偏性，降低了方差。

4.1 LASSO 及其拓展

LASSO(Least absolute shrinkage and selection operator) 是[Tibshirani1996] 提出，其求下面目标函数最小值

$$\min_{\beta} \sum_{i=1}^{n} (y_i - \boldsymbol{x}_i^\top \beta)^2 \tag{4.9}$$

$$\text{s.t.} \quad \sum_{j=1}^{p} |\beta_j| \le t \tag{4.10}$$

上式等价于

$$\min_{\beta} \left(\sum_{i=1}^{n} (y_i - \boldsymbol{x}_i^\top \beta)^2 + \lambda \sum_{j=1}^{p} |\beta_j| \right) \tag{4.11}$$

其中 λ 是截断参数 (Tuning parameter)。我们可以采用 ([Wang2007]) 方法，

$$\text{BIC}(\lambda_n) = \log \left(\sum_{i=1}^{n} (Y_i - Z_i^\top \beta) \right)^2 + \frac{\log n}{n} \times df$$

其中 df 估计非零的参数的个数。$\lambda_{opt} = \arg\min_{\lambda_n} \text{BIC}(\lambda_n)$.

相比岭回归，LASSO 只是将约束条件修改为绝对值。这样做为什么可以选择变量？从图 **??**，LASSO 更有可能得到稀疏的解，即某一个解为 0。这是由于解易出现菱角或者边缘。对于岭回归而言是约束域是圆，所以每一点的可能性相同，而矩阵有几个角，角的可能性更大些。

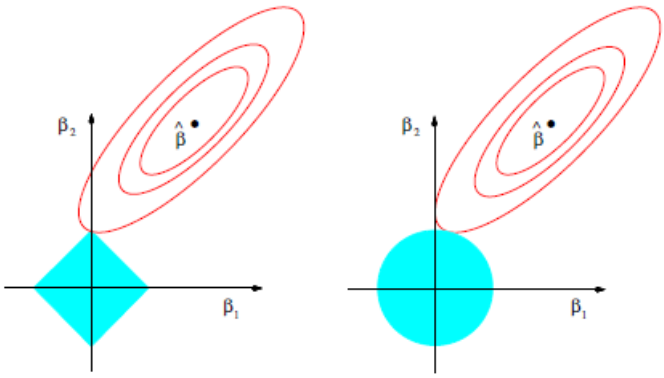

图 4.1: LAOO 和岭回归的几何解释；左边是 LASSO，右图是岭回归

LASSO 被提出后，后面有很多文章提出了不同的方法，如 SCAD ([Fan2001]) 和 Adaotive LASSO

([Zou2006]). 一般而言，后者更为简单，其为：

$$\min_{\beta}\Big(\sum_{i=1}^{n}(y_i-\boldsymbol{x}_i^{\top}\beta)^2+\lambda\sum_{j=1}^{p}w_j|\beta_j|\Big) \tag{4.12}$$

其中 $w_j=\dfrac{1}{|\widetilde{\beta}_j|^{\kappa}}$, and $\kappa>0$. $\widetilde{\beta}$ 最小二乘的解[Zou2006] 建议 $\kappa=1$。

下面我们讨论几种变量选择的关系。假设 $\boldsymbol{x}^{\top}\boldsymbol{x}=I$，其中 I 是单位矩阵。所以 $\hat{\beta}_{ols}=\boldsymbol{x}^{\top}y, \hat{y}_{ols}=\boldsymbol{x}\boldsymbol{x}^{\top}y$，且 $\boldsymbol{x}^{\top}(y-\hat{y}_{ols})=0$。我们考虑一般的变量选择的惩罚函数形式分析：

$$\frac{1}{2}\|y-\boldsymbol{x}\beta\|^2+\lambda\sum_{j=1}^{p}p_{\lambda}(|\beta_j|) \tag{4.13}$$

其中 $p_{\lambda}(\cdot)$ 是罚函数。经过计算，我们可以得到：

$$\begin{aligned}&\frac{1}{2}\|y-\boldsymbol{x}\beta\|^2+\lambda\sum_{j=1}^{p}p_{\lambda}(|\beta_j|)\\&=\frac{1}{2}\|y-\hat{y}_{ols}|^2+\frac{1}{2}\sum_{j=1}^{p}\|\hat{\beta}_{ols,j}-\beta_j\|^2+\lambda\sum_{j=1}^{p}p_{\lambda}(|\beta_j|)\end{aligned}$$

这是因为:

$$\begin{aligned}&(y-\boldsymbol{x}\beta)^{\top}(y-\boldsymbol{x}\beta)\\&=(y-\hat{y}+\hat{y}-\boldsymbol{x}\beta)^{\top}(y-\hat{y}+\hat{y}-\boldsymbol{x}\beta)\\&=(y-\hat{y})^{\top}(y-\hat{y})+(\hat{y}-\boldsymbol{x}\beta)^{\top}(\hat{y}-\boldsymbol{x}\beta)+2(\hat{y}-\boldsymbol{x}\beta)^{\top}(y-\hat{y})\\&=\|y-\hat{y}\|^2+(\hat{\beta}_{ols}-\beta)^{\top}\boldsymbol{x}^{\top}\boldsymbol{x}(\hat{\beta}_{ols}-\beta)+2(\hat{y}-\boldsymbol{x}\beta)^{\top}(y-\hat{y})\end{aligned}$$

模型 (4.13) 可以转化为

$$\frac{1}{2}(\hat{\beta}_{ols,j}-\beta_j)^2+\lambda p_{\lambda}(|\beta_j|),$$

更为一般的形式为：

$$\frac{1}{2}(z-\theta)^2+\lambda p_{\lambda}(|\theta|),$$

$p_{\lambda}(\cdot)$ 取不同的形式对应不同的方法：

1. $p_{\lambda}(\theta)=\lambda^2-(|\theta|-\lambda)I(|\theta|<\lambda)|\theta|^2$ 是最优子集估计量。
2. $p_{\lambda}(\theta)=\lambda|\theta|^2$ 是岭回归估计量
3. $p_{\lambda}(\theta)=\lambda|\theta|$ 是 LASSO 估计量
4. $p'_{\lambda}(\theta)=\lambda\Big\{I(\theta<\lambda)+\frac{(a\lambda-\theta)_+}{(a-1)\lambda}I(\theta\geq\lambda)$ 是 SCAD 估计量

经过推断，我们可以得到如下结论：

1. 最优子集的估计量 $\hat{\theta}=zI(|z|>\lambda)$
2. 岭回归估计量 $\hat{\theta}=\frac{z}{1+2\lambda}$
3. LASSO 估计量 $\hat{\theta}=\mathrm{sgn}(z)(|z|-\lambda)_+$
4. SCAD 估计量

$$\hat{\theta}=\begin{cases}\mathrm{sgn}(z)(|z|-\lambda)_+ & |z|\leq 2\lambda\\ \frac{(a-1)z-\mathrm{sgn}(z)a}{\lambda}a-2 & 2\lambda\leq|z|\leq a\lambda\text{x!=0}\\ z & |z|\geq a\lambda\end{cases}$$

下面我们介绍[Fan2001] 提出好的罚函数应该具备如下性质：

1. 无偏性 (Unbiasedness)：对于较大的 θ，$p'_{\lambda}(|\theta|)=0$，则 $\hat{\theta}=z$。
2. 稀疏性 (Sparsity)：$\min_{\theta\neq0}\{|\theta|+p'_{\lambda}(|\theta|)\}>0$，则解具有稀疏性，即当 $|z|<\min_{\theta\neq0}\{|\theta|+p'_{\lambda}(|\theta|)\}$

时,$\hat{\theta}=0$.

3. 连续性 (Continuity)：$\min\{|\theta|+p'_\lambda(|\theta|)\}$

经过计算，我们可以得到

表 4.1: 几种罚函数的比较

方法	无偏性	稀疏性	连续性
最优子集	√	√	
岭回归			√
LASSO		√	
SCAD	√	√	√

4.2 算法

4.2.1 二次逼近算法

关于惩罚函数求解的方法有很多。我们主要介绍局部二次逼近算法 (Local quadratic approximation,[Fan2001])。该算法采用二次逼近罚函数。假设 β^* 比较接近最小值。

- 如果 β_j^* 非常接近 0，我们设定 $\hat{\beta}_j=0$.
- 否则，我们使用二次逼近罚函数 $p_\lambda(|\beta_j|)$

由于，当 $\beta_j\neq 0$，且 $\beta_j^*\approx\beta_j$

$$[p_\lambda(|\beta_j|)]'=p'_\lambda(|\beta_j|)\mathrm{sgn}\beta_j\approx\frac{p'_\lambda(|\beta_j^*|)}{|\beta_j^*|}\beta_j.$$

换句话说：

$$p_\lambda(|\beta_j|)\approx p_\lambda(|\beta_j^*|)+\frac{1}{2}\frac{p'_\lambda(|\beta_j^*|)}{|\beta_j^*|}\left(\beta_j^2-(\beta_j^*)^2\right) \tag{4.14}$$

上面的式子是 $\beta_j^2-(\beta_j^*)^2$(不考虑 1/2)，不是 $|\beta_j|-|\beta_j^*|$。根据 Taloy 展开式这一项应该是绝对值的差，但作者使用了平方，因为平方可导。这也是称为二次逼近的原因。能不能用绝对值？当然可以。它的名字是局部线性逼近 (Local linear approximation,[Zou2008])。二次逼近可导，线性逼近不可导；二次逼近的解不稀疏，而线性逼近的解稀疏。关于二次逼近的讨论，详见[Lee2016]。

将上面逼近代入一般的惩罚函数的表达式为：

$$\min\left\{(y-\boldsymbol{x}\beta)^\top(y-\boldsymbol{x}\beta)+\sum_{j=1}^{p}\left[p_\lambda(|\beta_j^*|)+\frac{1}{2}\frac{p'_\lambda(|\beta_j^*|)}{|\beta_j^*|}\left(\beta_j^2-(\beta_j^*)^2\right)\right]\right\}$$

由于 β_j^* 是给定的，所以上面的式子进一步转化为：

$$\min\left\{(y-\boldsymbol{x}\beta)^\top(y-\boldsymbol{x}\beta)+\sum_{j=1}^{p}\frac{1}{2}\frac{p'_\lambda(|\beta_j^*|)}{|\beta_j^*|}\beta_j^2\right\} \tag{4.15}$$

为了记号方便，令 $u_i(\beta_j^*)=\frac{1}{2}\frac{p'_\lambda(|\beta_j^*|)}{|\beta_j^*|}$，则

$$\beta=(\boldsymbol{x}^\top\boldsymbol{x}+U(\beta^*))^{-1}\boldsymbol{x}^\top y \tag{4.16}$$

其中 $U(\beta^*)=\mathrm{diag}\{u_1(\beta_j^*),\ldots,u_p(\beta_j^*)\}$.

具体算法如下：

1. 第一步：给定初始值 $\beta^{(0)}$,

2. 第二步：令 $\beta^{(m)} = \beta^{(0)}$ 根据 (4.16) 更新 $\beta^{(m+1)}$

3. 第三步：重复第二步，直到其收敛。

对于 Adaptive LASSO，而言，$u_i(\beta_j^*) = \lambda \frac{1}{2|\hat{\beta}_{ols,j}|} \frac{1}{|\beta_j^*|}$。我们进行下面模型模型看看算法效果如何。

在进行分析前，我们需要解决常数项的问题，变量惩罚不会对其对其进行惩罚。下面我们简单的说明一下。回归模型为

$$y = \beta_0 + \beta_1 x_1 + \cdots + \beta_p x_p + \varepsilon$$

其经验的回归方程是：

$$\hat{y} = \hat{\beta}_0 + \hat{\beta}_1 x_1 + \cdots + \hat{\beta}_1 x_1 \tag{4.17}$$

经过中心化 $(\overline{x}_1, \ldots, \overline{x}_p, \overline{y})$ 的变量为 $(\widetilde{x}_1, \ldots, \widetilde{x}_p, \widetilde{y})$，则令

$$\widetilde{y} = \hat{\beta}_1^\star \widetilde{x}_1 + \cdots + \hat{\beta}_p^\star \widetilde{x}_p$$

进行得到：

$$y - \overline{y} = \hat{\beta}_1^\star (x_1 - \overline{x}_1) + \cdots + \hat{\beta}_p^\star (x_p - \overline{x}_p)$$

从而得到

$$y = (\overline{y} - \hat{\beta}_1^\star \overline{x}_1 - \cdots - \hat{\beta}_p^\star \overline{x}_p) + \hat{\beta}_1^\star x_1 + \cdots + \hat{\beta}_p^\star x_p \tag{4.18}$$

比较 (4.17) 和 (4.18)，我们可以得到：

$$\beta_0 = \overline{y} - \hat{\beta}_1^\star \overline{x}_1 - \cdots - \hat{\beta}_p^\star \overline{x}_p$$
$$\hat{\beta}_j = \hat{\beta}_j^\star, \quad j = 1, \ldots, p$$

下面是二次逼近的代码，我们可以修改阈值 (Threshold) 看看变量选择结果，默认 `thre=1e-2`，如果估计参数值的绝对值小于 0.01 设置为 0。从输出的结果来看，算法的效果还可以。

Adaptive LASSO 二次逼近代码

```
rm(list=ls())
library(MASS)

data1 <- function(n, p, sig0=0.25, rho=0.5){
  sigm <- sig0^abs(outer(1:(p),1:(p),"-"))
  muz <- rep(0, p)
  x <- mvrnorm(n = n, mu=muz, Sigma=sigm)
  e <- rnorm(n, 0, 1)
  beta <-c(3, 1.5, 0, 0, 2, rep(0, p-5))
  y <-  x %*% beta + rho * e
  return(list(y=y, x=x, beta=beta))
}

p <- 8
n <- 200
dat <- data1(n = n, p = p)
x <- dat$x
y <- dat$y
beta.ture <- dat$beta
```

```
#lam:lambda
#eps:
#itemax: maximum iteration time
#thre:  threshold
adalasso.my <- function(x, y, lam=0.1, eps=1e-5, itemax=1000, thre=1e-2, intercept=TRUE){
  p <- dim(x)[2]
  n <- dim(x)[1]
  x_colname <- colnames(x)
  beta_ols <- coef(lm(y~x-1))
  BB <- beta_ols
  if(is.null(x_colname)==TRUE){x_colname <- paste("x", 1:p, sep = "")}
  #
  if(intercept==TRUE){
    ###################################inclued intercept begin
    x_mean <- apply(x, 2, mean)
    y_mean <- mean(y)
    x_c <- scale(x=x, center=TRUE, scale=FALSE)
    y_c <- y - mean(y)
    txx <- t(x_c) %*% x_c
    txy <- t(x_c) %*% y_c
    iter <- 0
    juli <- 1
    ############### loop begin
    while( juli > eps ){##eps <- 0.1 stop
      u_beta <- lam * 1 / (abs(beta_ols) * abs(BB))
      U_beta <- diag(u_beta)
      BB_new <- solve( txx + U_beta) %*% txy
      cha  <-  BB_new  -  BB
      juli  <- ( t(cha) %*% cha ) ^ 0.5
      iter <- iter + 1
      BB   <- BB_new
      if(iter > itemax) break;
    }
    #############loop end
    beta0 <- c(y_mean - x_mean %*% BB)
    index_zero <- which(BB <= thre)
    BB_threshold <- BB
    BB_threshold[index_zero] <- 0

    beta_full <- c(beta0, BB_threshold)
    names(beta_full) <- c("Intercept", x_colname)
    res <- y - (beta0 + x %*% BB_threshold)
```

```
    BIC_my <- log(sum(res^2)) + log(n) / n * sum(BB_threshold!=0)
    return(list(beta=beta_full, BIC=BIC_my, iter=iter))
    ###################################inclued intercept end
  }else{
    ###################################not inclued intercept begin
    txx <- t(x) %*% x
    txy <- t(x) %*% y
    iter <- 0
    juli <- 1
    ##########loop begin
    while( juli > eps ){##eps <- 0.1 stop
      u_beta <- lam * 1 / (abs(beta_ols) * abs(BB))
      U_beta <- diag(u_beta)
      BB_new <- solve( txx + U_beta) %*% txy
      cha  <-  BB_new  -  BB
      juli  <- ( t(cha) %*% cha ) ^ 0.5
      iter <- iter + 1
      BB   <- BB_new
      if(iter > itemax) break;
    }
    ########loop end
    index_zero <- which(BB <= thre)
    BB_threshold <- BB
    BB_threshold[index_zero] <- 0
    beta_full <- c(BB_threshold)
    names(beta_full) <- x_colname
    res <- y -  x %*% BB_threshold
    BIC_my <- log(sum(res^2)) + log(n) / n * sum(BB_threshold!=0)
    return(list(beta=beta_full, BIC=BIC_my, iter=iter))
    ###################################not inclued intercept end
  }
}

adalasso.my(x=x, y=y, intercept=TRUE)
adalasso.my(x=x, y=y, intercept=FALSE)

library(doParallel)
library(foreach)

cl <- makeCluster(8)
registerDoParallel(cl)

nlam <- 100
```

```
lam_min <- 0.05
lam_max <- 4
lam_v   <- seq(from=lam_min, to=lam_max, length=nlam)
BIC_lam <- foreach(lam=lam_v, .combine="rbind") %dopar%
  {adalasso.my(y=y, x=x, lam=lam, intercept=FALSE)$BIC}

##plot
plot(lam_v, BIC_lam)
index_optlam <- min(which(BIC_lam==min(BIC_lam)))
lam_opt <- lam_v[index_optlam]
print(lam_opt)
fit <- adalasso.my(y=y, x=x, lam=lam_opt)
##compared
cbind(fit$beta[-1], beta.ture)
stopCluster(cl)
```

输出的结果

```
> cbind(fit$beta[-1], beta.ture)
              beta.ture
x1 3.01404086       3.0
x2 1.50588042       1.5
x3 0.01028767       0.0
x4 0.00000000       0.0
x5 2.00791256       2.0
x6 0.00000000       0.0
x7 0.00000000       0.0
x8 0.00000000       0.0
```

4.2.2 坐标下降法

上面的算法不适合 $p > n$ 的情况，下面我介绍一种常见的算法坐标下降法 (Coordinate descent algorithm,[Wu2008])，该算法一个分量一个分量计算。

给定 $\beta_1^{(k)}, \ldots, \beta_{j-1}^{(k)}, \beta_{j+1}^{(k)}, \ldots, \beta_p^{(k)}$，我们求解

$$\beta_j^{(k+1)} = \arg\min_{b_j} \left\{ Q(\beta_1^{(k)}, \ldots, \beta_{j-1}^{(k)}, \beta_j, \beta_{j+1}^{(k)}, \beta_p^{(k)}) + \lambda_n \frac{1}{2|\beta_j^{(k)}||\hat{\beta}_{ols,j}|} \beta_j^2 \right\} \tag{4.19}$$

其中 $Q(\beta)$ 是 $\|y - \boldsymbol{x}\beta\|^2$。算法的具体步骤如下：**大家可以尝试编写一下代码试试)**

- 第一步：初始值 $\beta^{(0)}$
- 第二步：$k \geq 0$, 给定 $\beta^{(k)}$

2.1 对于 $j \in 1, \ldots, p$, 利用 (4.19) 更新 $b_j^{(k+1)}$

2.2 重复上面不步骤直到 $b_j^{(k+1)}$ 收敛，从得到 $\beta^{(k+1)}$

- 第三步：重复第二步直到 $\beta^{(k)}$ 收敛。

4.3 组变量选择

顾名思义，组变量是一组变量。如分类变量具有 3 个水平，其需要转化为 2 个虚拟变量 (Dummy variable)。这 2 个虚拟变量是一组。我们进行变量选择，不能只选择其中的一个变量保留另一个变量。为了解决这个问题，**YuanLi2006** 提出了组变量 LASSO，**HuangMa2012** 详细总结了组变量选择方法 LASSO、SCAD 和 MCP，并且简单介绍了其在可加模型和变系数模型的应用。假设 $(X_1, \ldots, X_p)$ 可以分成 K 组，其中每一组的自变量个数为 d_k，则一般表达式为：

$$\frac{1}{2n}\|y - \sum_{k=1}^{K} X_k\beta_k\|_2^2 + \sum_{k=1}^{K} p_\lambda(\|\beta_k\|_{R_k})$$

其中 $\|v\|_R^2 = v^\top Rv$. 通常 R 是一个单位矩阵。`grpreg` 包中的函数 `grpreg` 可以实现 LASS，SCAD 和 MCP 的组变量选。

```
grpreg(X, y, group=1:ncol(X), penalty=c("grLasso", "grMCP", "grSCAD"),
            family=c("gaussian", "binomial", "poisson"), .....
```

下面是利用 CV 准则选择 λ 的命令。

组变量选择代码

```
rm(list=ls())
library(grpreg)
data(Birthwt)
summary(Birthwt)
X <- Birthwt$X
y <- Birthwt$bwt
group <- Birthwt$group

cvfit <- cv.grpreg(X, y, group)
plot(cvfit)
summary(cvfit)
coef(cvfit) ## Beta at minimum CVE

```

参考文献

[Fan and Li(2001)] Fan J. and Li R. (2001) Variable selection via nonconcave penalized likelihood and its oracle properties. Journal of the American Statistical Association, 96(456): 1348–1360.

[Huang, Breheny & Ma (2012)] Huang, J., Breheny, P., & Ma, S. (2012). A selective review of group selection in high-dimensional models. Statistical science: a review journal of the Institute of Mathematical Statistics, 27(4).

[Lee, Kwon and Kin (2016)] Lee, S., Kwon, S. and Kim, Y. (2016). A modified local quadratic approximation algorithm for penalized optimization problems. Computational Statistics & Data Analysis, 94, 275-286.

[Yuan & Lin.(2006)] Yuan, M., & Lin, Y. (2006). Model selection and estimation in regression with grouped variables. Journal of the Royal Statistical Society: Series B (Statistical Methodology), 68(1), 49-67.

[Wang et al.(2007)] Wang H., Li R. and Tsai. C. (2007) Tuning parameter selectors for the smoothly clipped absolute deviation method. Biometrika, 94(3): 553–568, 2007.

[Wang et al.(2009)] Wang H. Li B. and Leng C. (2009) Shrinkage tuning parameter selection with a diverging number of parameters, Journal of the Royal Statistical Society. Series B (Statistical Methodology), 71(3): 671–683.

[Tibshirani(1996)] Tibshirani R. (1996) Regression shrinkage and selection via the LASSO. Journal of the Royal Statistical Society. Series B (Statistical Methodology), 58(1): 267–288.

[Wu and Lang (2008)] Wu, T. and Lange, K. (2008). Coordinate descent algorithms for lasso penalized regression. The Annals of Applied Statistics, 2(1), 224-244.

[Zhang and Huang(2008)] Zhang C. and Huang J.(2008) The sparsity and bias of the LASSO selection in highdimensional linear regression, The Annals of Statistics, 36(4): 1567–1594.

[Zou(2006)] Zou H.(2006) The adaptive LASSO and its oracle property. Journal of the American Statistical Association, 101(476): 1418–1429.

[Zou and Li(2008)] Zou H. and Li R. One-step sparse estimates in nonconcave penalized likelihood models. Annals of statistics, 36(4):1509–1533.

第 5 章 核密度估计

以前统计学习中，总假设密度函数 f 是已知的，但是现实生活中，它是未知的。这一章我们将介绍估计 f 的方法。

5.1 faithful 数据

例题 5.1 faithful 数据来自与 R 包 datasets，其描述美国黄石公园的间隙式温泉，该温泉的喷发有一定规律，大约 70 分钟喷发一次，实际上是 43 分钟到 96 分钟不等。该数据有 272 个观测值，2 个变量：

- eruptions：数值变量，温泉喷发持续时间（分钟为单位）；
- waiting：数值变量，温泉喷发的间隔时间（分钟为单位）。

目的：估计这两个变量的密度函数。

我们知道高中学习的直方图可以大概显示密度函数的趋势，但需要选择区间数目和大小，并且这种描述比较简单。我们下面介绍核密度估计（kernel density estimate）。

faithful 数据摘要

```
> rm(list = ls()) # 清除所有对象
> data(faithful)  # 加载数据
> summary(faithful) # 数据摘要
   eruptions        waiting
 Min.   :1.600   Min.   :43.0 # 最小值
 1st Qu.:2.163   1st Qu.:58.0 # 四分之一分位数
 Median :4.000   Median :76.0 # 中位数
 Mean   :3.488   Mean   :70.9 # 均值
 3rd Qu.:4.454   3rd Qu.:82.0 # 四分之三分位数
 Max.   :5.100   Max.   :96.0 # 最大值
```

5.2 核密度估计

设 $X_1, \cdots, X_n \overset{\text{iid}}{\sim} f$，则 f 的核密度估计的表达式为：

$$\widehat{f_h}(x) = \frac{1}{n}\sum_{i=1}^{n} K_h(x - X_i)$$

其中，

$$K_h(\bullet) = \frac{1}{h}K(\frac{\bullet}{h})$$

$K(\bullet)$ 是核函数 (Kernel function)，h 是宽带（Bandwidth）。h 是光滑参数，其越大，拟合曲线越平滑。如图 5.1，随着 h 的增大，拟合曲线越来越光滑。在估计 x 点的密度时，核密度估计其实利用其周围的点的信息。离其越近，其作用越大，即其权重越大。如图5.2所示，总共由 10 个样本点（+ 表示点的位置）。在估计某一个点的密度时，利用其它点的信息，即以 + 为中心的小的正态曲线。

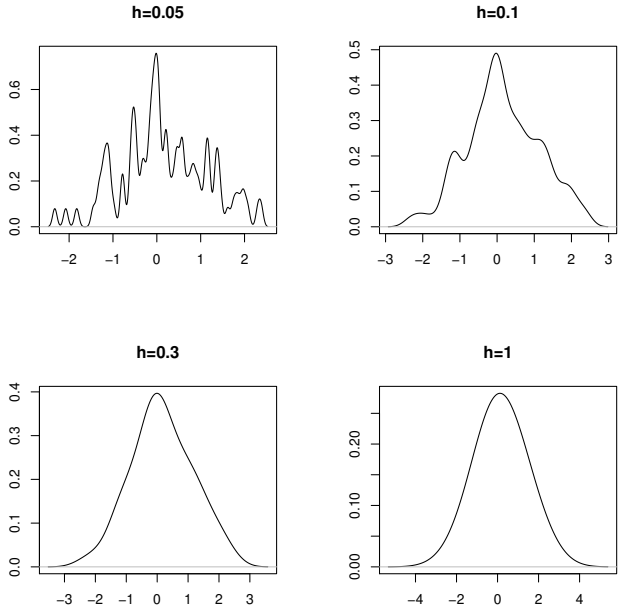

图 5.1: 四种 h 对应的核估计图形

图5.1的 R 代码

```
x <- rnorm(100)
par(mfrow=c(2,2))
plot(density(x,bw=0.05),main="h=0.05", xlab="", ylab="")
plot(density(x,bw=0.2 ),main="h=0.2",  xlab="", ylab="")
plot(density(x,bw=0.4 ),main="h=0.4",  xlab="", ylab="")
plot(density(x,bw=1   ),main="h=1",    xlab="", ylab="")
```

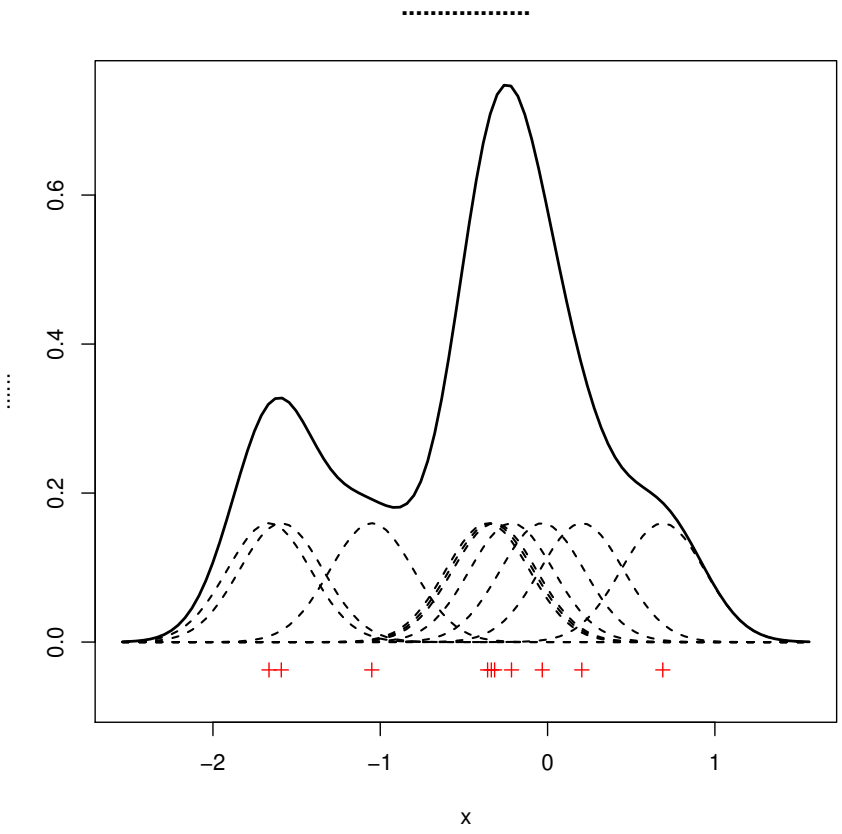

图 5.2: 核估计的构建

图5.2的 R 代码

```
# 带宽
h <- 0.25
# 样本量
n <- 10
# 产生 100 个正态随机数
x  <- rnorm(n)
# x 的最小值和最大值的差
xr <- diff(range(x))
ng <- 100
xg <- (xr + 7 * h) * (0:(ng-1)) / (ng - 1) + min(x) - 3.5 * h
fk <- matrix(0, nrow=ng, ncol=n)
# 核估计
for (j in 1:n){
  #Gaussian kernel 高斯核函数
  fk[ ,j] <- dnorm((xg - x[j]) / h) / (n * h)
}

#Kernel density estimate 核密度估计
fh <- rowSums(fk)
```

```
20
21 # 画图
22 ylim <- c(-0.1 * max(fh), max(fh)) # 设置 y 轴的范围
23 plot(xg,fh,type="l",lwd=2,ylim=ylim, main="",xlab="x",ylab="")
24 points(cbind(x,rep(-0.05 * max(fh),n)), col="red",pch=3)
25 for (j in 1:n){
26   lines(xg,fk[,j],lty=2,lwd=1.5)
27 }
```

5.3 R 实现

R 函数 density 可以实现。

```
density(x, bw = "nrd0",
        kernel = c("gaussian", "epanechnikov", "rectangular",
                   "triangular", "biweight", "cosine",......)
```

其中

- bw：带宽选择的方法。
- kernel：核函数的选择。默认是 gaussian。

5.3.1 核函数的选择

一般来说，核函数是概率密度，并且对称。主要原因如下：

1. 保证 $\widehat{f}$ 是密度函数，其必须满足

$$\begin{aligned}\int \widehat{f_h}(x)dx &= \int \frac{1}{n}\sum_{i=1}^{n} K_h(x-X_i)dx \\ &= \int \frac{1}{nh}\sum_{i=1}^{n} K(\frac{x-X_i}{h})dx \\ &= \int K(u)du \\ &= 1\end{aligned}$$

   从而 $\int K(u)du = 1$

2. 保证离 x 的距离相等点的权重相同，核函数必须是对称的，即 $\int uK(u)du = 0$。这两个性质非常重要，将在推到核密度估计的偏差和方差中应到。

表5.1列举了常见的核函数。核函数的选择没有那么重要，他们之间可以通过调整带宽使得估计几乎相当。

表 5.1: R 中常见的核函数

核函数	$K(u)$
Gaussian	$\frac{1}{2\pi}\exp(-\frac{1}{2}u^2)$
Uniform	$\frac{1}{2}I(\|u\| \le 1)$
Epanechnikov	$\frac{3}{4}(1-u^2)I(\|u\| \le 1)$
Triangle	$(1-\|u\|)I(\|u\| \le 1)$
Biweight	$\frac{15}{16}(1-u^2)^2I(\|u\| \le 1)$
Triweight	$\frac{35}{32}(1-u^2)^3I(\|u\| \le 1)$
Cosine	$\frac{\pi}{4}cos\left(\frac{\pi}{2}u\right)I(\|u\| \le 1)$

5.3.2 带宽的选择

相对于核函数的选择，带宽的选择比较重要。下面列举几种常见的方法。

1. Silverman 拇指法 (Silverman's rule of thumb)，也称插入法 (Plug-in)。`stats` 包的 `bw.nrd` 函数可以实现。这种方法适合单峰对称的分布，缺点不适合厚稳分布，对异常值比较敏感。这主要原因该方法使用正态分布代替密度函数，并且使用变得是高斯核函数。
2. 改进的插入法。`stats` 包的 `bw.nrd0` 函数可以实现或者 `MASS` 包的 bandwith.nrd。

5.4 数据分析

图5.3显示 `eruptions` 和 `waiting` 都是多峰分布。

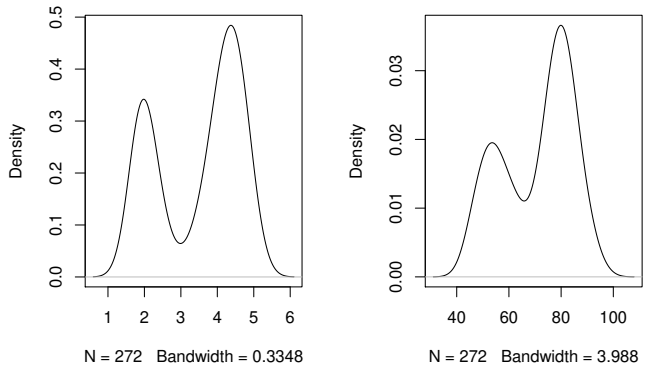

图 5.3: faithful 数据的密度估计图

图5.3的代码

```
par(mfrow=c(1,2))
plot(density(faithful$eruptions), main=" ")
plot(density(faithful$waiting),   main=" ")
```

5.5 附录

我们往往关注它的偏差（bias）、方差和均方误差（mean squared error）。

5.5.1 偏差

$$
\begin{aligned}
Bias\{\widehat{f_h}(x)\} &= E\{\widehat{f_h}(x)\} - f(x) \\
&= E\Big\{\frac{1}{n}\sum_{i=1}^{n} K_h\Big(x - X_i\Big)\Big\} - f(x) \\
&= E\Big\{\frac{1}{nh}\sum_{i=1}^{n} K\Big(\frac{x - X_i}{h}\Big)\Big\} - f(x) \\
&= \frac{1}{nh}\sum_{i=1}^{n} E\{K(\frac{x - X_i}{h})\} - f(x) \\
&\overset{iid}{\underset{X_1,\ldots,X_n}{=\!=\!=}} \frac{1}{h}E\{K(\frac{x - X}{h})\} - f(x) \\
&= \frac{1}{h}\int K(\frac{x - u}{h}) f(u)du - f(x)
\end{aligned}
$$

令 $s = \frac{u-x}{h}$。则 $s = sh + x$，$dx = hds$，从而

$$f(x + sh) = f(x) + f'(x)sh + f''(x)\frac{(sh)^2}{2} + o(sh)^2$$

所以

$$
\begin{aligned}
&Bias\{\widehat{f_h}(x)\} \\
&= \int K(s) f(x + sh)ds - f(x) \\
&= \int K(s)[f(x) + f'(x)sh + f''(x)\frac{(sh)^2}{2} + o(sh)^2]ds - f(x) \\
&= f(x)\int K(s)ds + f'(x)h\int sK(s)ds + \frac{f''(x)}{h^2}\int s^2K(s)ds + o(h^2) - f(x)^1 \\
&= f(x) + 0 + \frac{h^2 f''(x)}{2}\int s^2K(s)ds + o(h^2) - f(x) \\
&= \frac{h^2 f''(x)}{2}\int s^2K(s)ds + o(h^2)
\end{aligned}
$$

令 $\int s^2K(s)ds = \mu_2(K)$, 所以

核密度估计偏差
$Bias\{\widehat{f_h}(x)\} = \frac{h^2}{2}f''(x)\mu_2(K) + o(h^2)\ \ h \to 0$

可以看出核密度估计的方差与 h 成正比，即 h 越大，偏差越大。

5.5.2 方差

$$
\begin{aligned}
Var\{\widehat{f_h}(x)\} &= Var\{\frac{1}{n}\sum_{i=1}^{n}K_h(x-X_i)\} \\
&= Var\{\frac{1}{nh}\sum_{i=1}^{n}K(\frac{x-X_i}{h})\} \\
&\overset{iid}{=\!=\!=\!=\!=\!=}_{X_1,\ldots,X_n} \frac{1}{n}Var\{\frac{1}{h}K(\frac{x-X}{h})\} \\
&= \frac{1}{n}[E\{\frac{1}{h^2}K^2(\frac{x-X}{h})\} - E\{\frac{1}{h}K(\frac{x-X}{h})\}^2] \\
&= \frac{1}{nh}\int K^2(s)f(x+sh)ds - \frac{1}{n}E\{\widehat{f_h}(x)\} \\
&= \frac{1}{nh}\int K^2(s)[f(x)+o(h)]ds - \frac{1}{n}[f(x)+o(h)] \\
&= \frac{1}{nh}f(x)\int K^2(s)ds + 0(\frac{1}{nh}) \\
&= \frac{1}{nh}f(x)||K||_2^2 + 0(\frac{1}{nh})
\end{aligned}
$$

其中 $||K||_2^2 = \int K^2(s)ds$. 即

核密度估计方差
$Var\{\widehat{f_h}(x)\} = \frac{1}{nh}f(x)\|\|K\|\|_2^2 + 0(\frac{1}{nh})\ \ nh \to \infty$

可以看出核密度估计的方差与 h 成反比，即 h 越大，方差越小。

5.5.3 均方误差

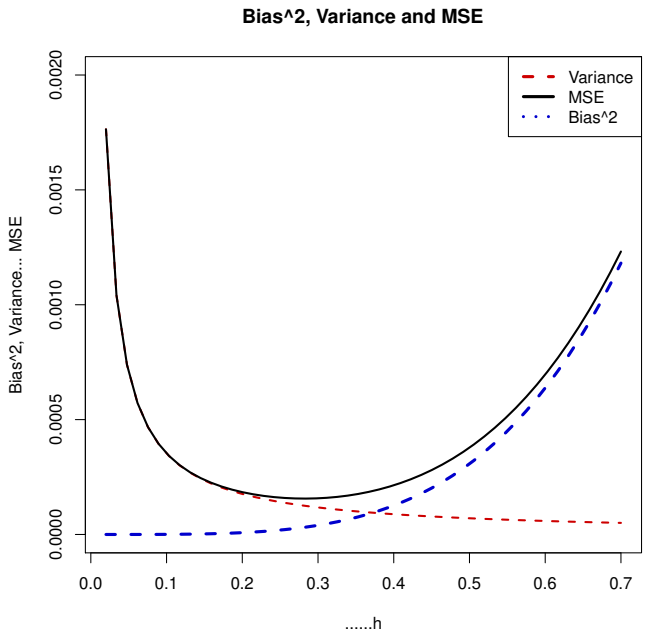

图 5.4: 偏差、方差和均方误差

图5.4的 R 代码

```
phi <-  0.4 # 分布某一点的概率
mu  <-  -3  # 均值
mu2 <-  1   # 二阶距
x0  <-  -1.2 # 待估点
h   <-  seq(0.02, 0.7,length=50) # 宽带
n   <-  1000 # 样本量
ck  <-  1/(2*sqrt(pi)) # 核函数的 L2
ff  <-  phi*dnorm(x0)+(1-phi)*dnorm(x0-mu) # 真实的密度
f2  <-  phi*(x0^2)*dnorm(x0)+(1-phi)*((x0-mu)^2)*dnorm(x0-mu)
# 真实密度的二阶倒数 its second derivative
f2 <-  f2 - phi*dnorm(x0) - (1 - phi) * dnorm(x0 - mu) # 减去真实密度
b  <-  h ^ 2 * f2 * mu2 / 2 # 偏差
b2 <- b^2 # 偏差的平方
v  <-  ff*ck/(n*h) # 方差
mse<-  b2 + v # 均方误差
plot(h,v,ylim=c(0,0.002),type="l",lwd=2,lty=2,
     col="red3",xlab=" 宽带 h", ylab="Bias^2, Variance 和 MSE")
title("Bias^2, Variance and MSE")
lines(h,mse,lwd=2,lty=1,col="black")
lines(h,b2,lwd=3,lty=2,col="blue3")
legend("topright",c("Variance","MSE","Bias^2"),lty=c(2,1,3),
       lwd=c(3,3,3),col=c("red3","black","blue3"))
```

第 6 章 非参数回归

6.1 mcycle 数据

例题 6.1 mcycle 数据来自与 R 包 MASS，主要描述摩托车模拟碰撞的数据，其有 133 个观测值，2 个变量：

- times：数值变量，摩托车碰撞之后的时间（百万分子一为单位）；自变量 X
- accel：数值变量，头部的加速度（重力加速度 g 单位）；因变量 Y。

目的：利用 times 对 accel 进行建模分析。

我们从图6.1可以看出 times 和 accel 存在关系，但不是线性关系。下面我们将介绍常见的几种非参数模型。

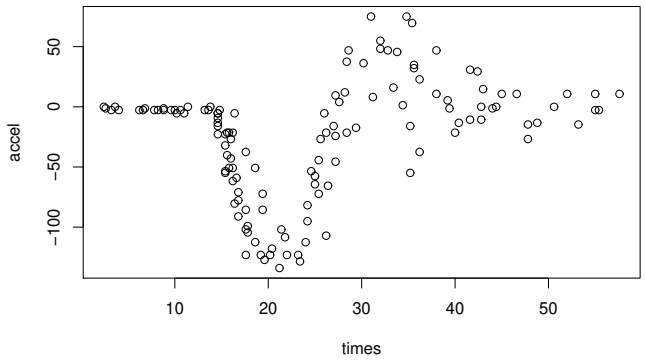

图 6.1: mcycle 的散点图

图6.1的代码

```
rm(list = ls())
library(MASS)
data(mcycle)
par(mfrow=c(1,1))
plot(mcycle)
```

6.2 核回归

核回归 (Kernel Rgression) 是非参数回归最基本的内容。它象线性回归在参数回归的地位。理解它有利于我们理解非参数回归。一般来说，非参数回归是参数回归的推广，而其的未知参数个数是无穷的，参数是有限的。如果有 n 个相互独立的观测值 $\{(X_i, Y_i)\}_{i=1}^n$，则回归寻找关系

$$Y_i = m(X_i) + \varepsilon_i \quad i = 1, 2, \cdots, n$$

其中 ϵ 是随机变量。参数回归 $m(X_i)$ 形式已知，而非参数回归其未知。我们通常关注 Y_i 在 X_i 给定的条件均值，即非参数均值回归（简称非参数回归，当然我们也可以不是均值是分位数，非参数分位数回归）。即

$$E[Y_i|X_i = x_i] = m(x_i)$$

这种形式假定 $E(\varepsilon_i|X_i = x_i) = 0$。所以，我们的目标是估计给定 $X = x$ 下 Y 的期望:

$$m(x) = E(Y|X = x) = \int yf(y|x)dy = \frac{\int yf(x,y)dy}{f_X(x)}$$

其中 $f(x,y)$ 是 x 和 Y 的条件分布,$f_X(x)$ 是 X 的边际分布。

记 $r(x) = \int yf(x,y)dy$, 则

$$m(x) = \frac{r(x)}{f_X(x)} \tag{6.1}$$

核回归根据 X 是不随机（$f_X(x)$ 未知）或随机（$f_X(x)$ 已知），可以分为 Nadaraya Watson(NG) 估计和 Gasser Müller(GM) 估计。我们这里主要讨论前者。

6.2.1 Nadaraya Watson(NW) 估计

NG 估计是对于（6.1）分子和分母分别估计。

（1）对于分子估计

$f(x,y)$ 可以使用乘积多元核估计估计，即

$$\widehat{f}_{h,g}(x,y) = \frac{1}{n}\sum K_h(\frac{x-X_i}{h})K_g(\frac{y-Y_i}{g})$$

从而

$$\begin{aligned}
\int y\widehat{f}_{h,g}(x,y)dy &= \int y\frac{1}{n}\sum K_h(\frac{x-X_i}{h})K_g(\frac{y-Y_i}{g})dy \\
&= \int y\frac{1}{n}\sum_{i=1}^{n}\frac{1}{h}K(\frac{x-X_i}{h})\frac{1}{g}K(\frac{y-Y_i}{g})dy \\
&= \frac{1}{n}\sum_{i=1}^{n}\frac{1}{h}K(\frac{x-X_i}{h})\int \frac{y}{g}K(\frac{y-Y_i}{g})dy \\
&\xlongequal[s=\frac{y-Y_i}{g}]{y=sg+Y_i} \frac{1}{n}\sum_{i=1}^{n}\frac{1}{h}K(\frac{x-X_i}{h})\int (sg+Y_i)K(s)ds \\
&= \frac{1}{n}\sum_{i=1}^{n}\frac{1}{h}K(\frac{x-X_i}{h})\Big(g\int sK(s)ds + Y_i\int K(s)ds\Big) \\
&= \frac{1}{n}\sum_{i=1}^{n}\frac{1}{h}K(\frac{x-X_i}{h})(0+Y_i) \\
&= \frac{1}{n}\sum_{i=1}^{n}\frac{1}{h}K(\frac{x-X_i}{h})Y_i
\end{aligned}$$

（2）对分母估计对于 $\widehat{f}_X(x)$ 采用核估计，即

$$\widehat{f}_X(x) = \frac{1}{n}\sum_{j=1}^{n}\frac{1}{h_1}K(\frac{x-X_i}{h})$$

上面这种引入估计方法没有利用分子的信息，显然不好。应该利用 $f_X(x) = \int f(x,y)dy$，所以

$$\begin{aligned}\widehat{f_X}(x) = \widehat{f}_X(x) &= \int \widehat{f}(x,y)dy \\ &= \int \frac{1}{n}\sum K_h(\frac{x-X_i}{h})K_g(\frac{y-Y_i}{g})dy \\ &= \frac{1}{n}\sum K_h(\frac{x-X_i}{h})\int \frac{1}{g}K(\frac{y-Y_i}{g})dy \\ &= \frac{1}{n}\sum K_h(\frac{x-X_i}{h})\int K(t)dt \\ &= \frac{1}{n}\sum K_h(\frac{x-X_i}{h})\end{aligned}$$

所以 NW 的估计是

$$\widehat{m}_h(x) = \frac{\widehat{r}(x)}{\widehat{f}_X(x)} = \frac{n^{-1}\sum_{i=1}^n K_h(x-X_i)Y_i}{n^{-1}\sum_{j=1}^n K_h(x-X_j)} \tag{2.2}$$

- 如果，我们重新写（2.2），

$$\widehat{m}_h(x) = \sum_{i=1}^n \frac{n^{-1}K_h(x-X_i)}{n^{-1}\sum_{j=1}^n K_h(x-X_j)}Y_i = \sum_{i=1}^n W_{hi}(x)Y_i$$

  可以看出，NW 的估计是 Y_i 的加权估计。
- 数据稀疏时，$W_{hi}(x)$ 可能为 0。
- h 选择决定了 NW 估计的光滑程度。如果 $h \to 0$，则 $W_{hi}(x) \to n$; 如果 $h \to \infty$，则 $W_{hi}(x) \to 1$

为了加深对于算法的了解，最好的方法就是编写代码。为此，我们利用 R 软件，进行模型。

$$Y_i = \sin^3(2\pi X_i^3) + \varepsilon_i \; i = 1, 2 \cdot \cdot, 256$$

其中，X_i 来自于 $U[0,1]$，ε_i 来自于标准正态分布。模拟结果如图 2.1 所示。从图6.2可以看出拟合（实线）效果并不好。这和我们选择的宽带有关系。因为我们选择的宽带是 Silver 宽带。Silver 宽带是基于正态假设下得到，而我们实际的数据是均分分布。关于宽带的选择，后面将会介绍。

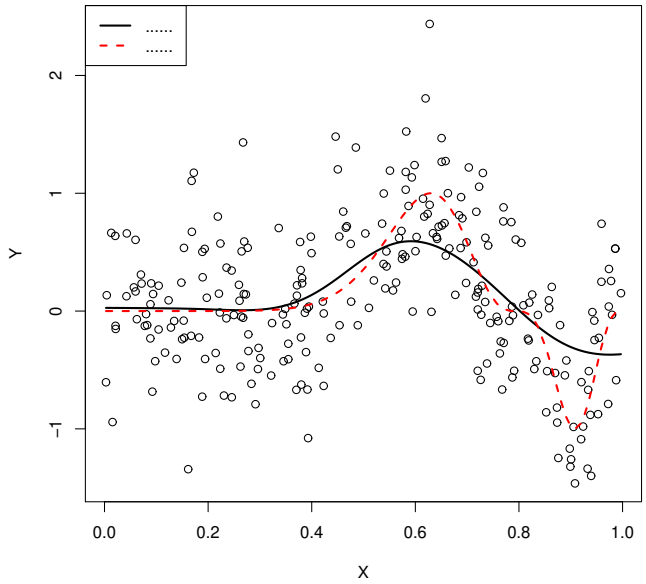

图 6.2: 模拟结果，实线是模拟结果，虚线是真实。

图6.2代码

```
n <- 256
X <- sort(runif(n))# 大写 X
h <- 1.06 * sd(X) * n ^ (-1 / 5)#Silver 宽带
e <- rnorm(n,mean=0, sd=0.5)
m <- (sin(2 * pi * X ^ 3)) ^ 3
Y <- m + e
ker <- function(s){1 / sqrt(2 * pi) * exp(- s ^ 2 / 2)}
mhat <- NULL
x <- seq(min(X), max(X),length=200)# 小写 x
for(i in 1:200){
  fenzi <- 1 / h * mean(ker((x[i] - X) / h)*Y)# 分子
  fenmu <- 1 / h * mean(ker((x[i] - X) / h))# 分母
  mhat[i] <- fenzi / fenmu
}
plot(X,Y)
lines(x,mhat,col=1,lty=1,lwd=2)
lines(X,m,col=2,lty=2,lwd=2)
legend("topleft",c(" 真实"," 估计"),lty=c(1,2),lwd=c(2,2),col=c(1,2))
```

6.2.2 NW 核回归 R 实现

NW 核回归 R 实现的的命令是:

```
ksmooth(x, y, kernel = c("box", "normal"), bandwidth = 0.5)
```

- x 表示自变量
- y 表示因变量
- kernel 表示核函数
- bandwith 表示宽带

6.2.3 mcycle 数据的 NW 建模

如图6.3所示。当 h 取不同的值时，NW 核回归的光滑程度不一样，其越大越光滑。至于怎么选择合适的 h，下一次我们将讨论。

图6.3代码

```
rm(list = ls())
library(MASS)
data(mcycle)
h <-  c(0.1,0.2,0.3,0.5) # 宽带
kernel <- "normal"# 高斯核
mh <-  list()# 存储核回归估计结果
for (i in 1:length(h)){
  mh[[i]] <-  ksmooth(mcycle$times, mcycle$accel,kernel=kernel,h[i])#NW 核回归
}

```

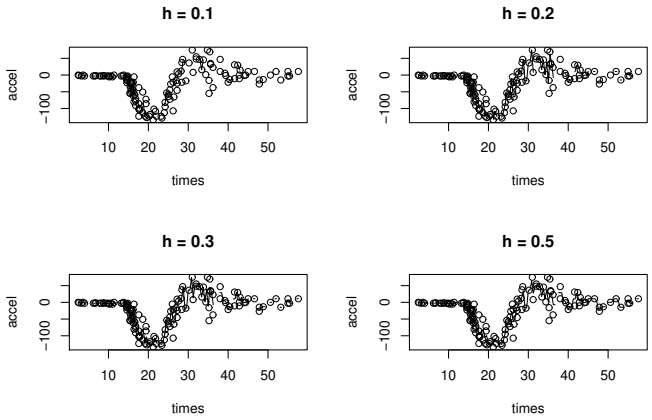

图 6.3: mcycle 核回归拟合图

```
par(mfrow=c(2,2))# 画 2x2 的图
for(i in 1:length(h)){
  plot(mcycle$times,mcycle$accel,type="n",main=paste("h =", h[i]),
       xlab="times",ylab="accel")
  points(mcycle$times, mcycle$accel)
  lines(mh[[i]])
}
```

6.3 局部线性回归

局部线性回归 (Locally Linear Regression) 是非参数回归最基本的理论之一，它的思想广泛应用于非参数回归求解，如变系数模型等。本文主要介绍局部线性回归的思想、软件实现和应用，至于大样本理论性质，详见 (田茂再, 2014; Fan 等,1994)

设 $\{Y_i, X_i\}_{i=1}^n$ 是样本序列，我们考虑下面模型：

$$Y_i = m(X_i) + \varepsilon_i \tag{6.2}$$

其中 ε_i 是独立同分布，且分布未知。$m(\bullet)$ 是未知函数。如果 $m(\bullet)$ 是线性函数，那么模型6.2就是线性模型。

目前比较流行的拟合方式是局部线性拟合，即对于任意一点 x

$$(\hat{a}, \hat{b}) = \text{argmin}_{a,b\in\mathbb{R}} \sum_{i=1}^{n} l\Big(Y_i - a - b(X_i - x)\Big) K\Big(\frac{X_i - x}{h}\Big) \tag{6.3}$$

其中 $l(\bullet)$ 是一个凸函数，并且在原点有唯一的最小值。$\hat{m}(x) = \hat{a}$，$\hat{m}'(x) = \hat{b}$。当 $l(u) = u^2$ 得到的回归是均值回归 (locally Linear Mean Regression)；当 $l(u) = \rho_\tau(u)$ 得到的回归是分位数回归 (locally Linear Quantile Regression)。h 带宽 (Bandwidth)。它决定估计曲线的光滑程度，其越大越光滑。$K(\bullet)$ 是核函数 (Kernel Function)，它满足 $\int_{-\infty}^{+\infty} K(z)dz = 1$ 且 $\int_{-\infty}^{+\infty} zK(z)dz = 0$。经常使用的核函数详见表6.1

表 6.1: 核函数

名称	$K(\bullet)$
均匀核 (Uniform)	$\frac{1}{2}I(\lvert z\rvert \le 1)$
Epanechnikov	$\frac{3}{4}(1-z^2)I(\lvert z\rvert \le 1)$
高斯核 (Gaussian)	$\frac{1}{2\pi}\exp\left(-\frac{1}{2}z^2\right)$

下面简单介绍局部线性回归的思想 (根据田茂再老师上课内容整理):

如图6.4所示，要估计 $m(x)$ 时，需要利用 $(x, m(x))$ 附近 (X_i, Y_i) 信息使得

$$\min \sum_{i=1}^{n} l(Y_i - m(X_i)) \tag{6.4}$$

由于 $m(X_i)$ 未知，通常用 $\widehat{Y}_i$ 代替。$(X_i, \widehat{Y}_i)$ 在过 $(x, m(x))$ 的切线上 (图6.4中实直线)。可以证明 $\widehat{Y}_i$ 其实是 $m(X_i)$ 在 x 的一阶 Taylor 展开式，即

$$m(X_i) \approx m(x) + m'(x)(X_i - x) \tag{6.5}$$

所以6.4可以转化为

$$\sum_{i=1}^{n} l\Big(Y_i - m(x) - m'(x)(X_i - x)\Big) \tag{6.6}$$

考虑到 (X_i, Y_i) 与 $(x, m(x))$ 距离不同，起的作用不同，越近的作用越大，所以采用局部加权的方法，6.6式进一步演变为

$$\sum_{i=1}^{n} l\Big(Y_i - m(x) - m'(x)(X_i - x)\Big) K\Big(\frac{X_i - x}{h}\Big) \tag{6.7}$$

6.7即是局部线性回归的求解式。比较6.7式和6.3式，可以得到 $a = m(x)$，$b = m'(x)$。

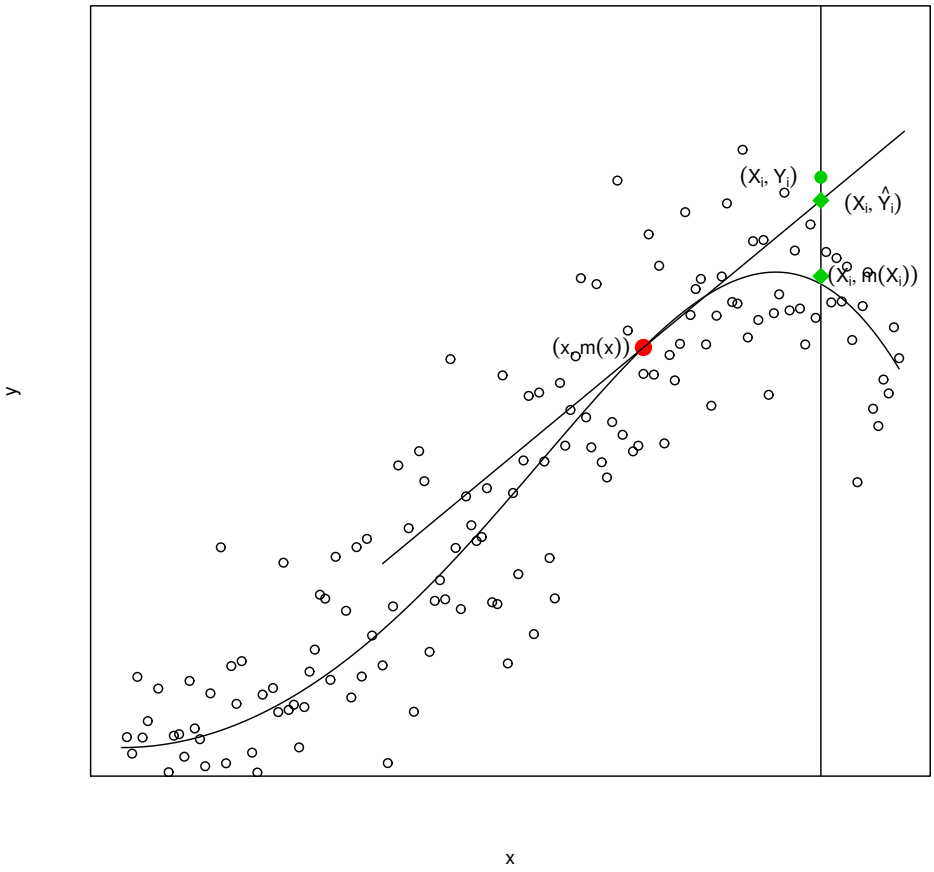

图 6.4: 局部线性回归

6.3.1 带宽的选择

h 的选择方法非常多，可以详见 Härdle 等 (2004)。下面我们介绍经常使用的方法：交叉核实 (Cross Validation, CV)。交叉核实被广泛应用于核密度估计、核回归以及样条光滑等，其定义为：

$$CV(h) = \sum_{i=1}^{n} \left(Y_i - \widehat{Y}_{(-i)} \right)^2$$

其中 $\widehat{Y}_{(-i)}$ 是去掉 Y_i 的估计。h 可以最小 $CV(h)$ 得到。

相对于局部线性分位数回归的 h_τ，局部线性均值回归的 h_{mean} 比较容易获得。R 包 `KernSmooth` 中的函数 `dpill` 可以实现 Ruppert 等 (1995) 提出的插入方法 (Plug-in)。Yu 和 Jone(1998) 得到 h_τ 和 h_{mean} 的关系是

$$h_\tau = \frac{\tau(1-\tau)}{\phi^2(\Phi)^{-1}(\tau))} h_{mean}$$

其中 ϕ 和 Φ 分别是标准正态分布的密度函数和分布函数。

表 6.2: 常用的 h_τ 和 h_{mean} 之间的关系

τ	0.05 或 0.95	0.25 或 0.75	0.5
h_τ	$1.34h_{mean}$	$1.13h_{mean}$	$1.10h_{mean}$

6.3.2 R 实现

局部线性均值回归可以使用 KernSmooth 包中的 locpoly 实现；局部线性分位数回归可以用 R 包 quantreg 中的 lprq 实现，它使用时高斯核函数。本文主要介绍局部线性分位数回归实现。

```
lprq(x, y, h, tau = .5, m = 50)

用法
x 自变量
y 因变量
h 带宽
tau 分为点
m 需要拟合的点数

输出结果
xx 需要估计的自变量
fv xx拟合值
dev xx拟合值一阶导数
```

lprq 得到不是 x 的拟合值，而是 xx = seq(min(x),max(x), length=m) 的拟合值，也即 x 最小值和最大值之间 m 个等距插值的拟合值。?lprq 查看 lprq 的源代码，修改部分参数实现 x 的拟合值。修改后的函数命名为 lprq.my。

```
lprq.my <- function (x, y, h, tau = 0.5, m = 50) {
  #xx <- seq(min(x), max(x), length = m) 去掉
  xx <- x #修改的参数
  fv <- xx
  dv <- xx
  for (i in 1:length(xx)) {
    z <- x - xx[i]
    wx <- dnorm(z/h)
    r <- rq(y ~ z, weights = wx, tau = tau, ci = FALSE)
    fv[i] <- r$coef[1]
    dv[i] <- r$coef[2]
  }
  list(xx = xx, fv = fv, dv = dv)
}
```

6.3.3 实例分析

数据 `fossil` 来自于 R 包 `SemiPar`，它有 106 个观测值，两个变量年龄 (age，单位是百万年) 和锶同位素的百分比 (strontium.ratio) (Bralower 等,1997; Chaudhuri 和 Marron, 1999)。锶同位素的百分比随着年龄而变化。它们的散点图如图6.5所示。

我们将对其进行局部线性分位数回归拟合。取分位点分别是 0.05，0.5，0.95。从图6.5 可以看出，局部线性分位数回归拟合的效果是非常好的，它能刻画锶同位素的百分比随年龄的变化。另外，利用局部线性分位数回归，我们可以得到某年的锶同位素的百分比的概率值表，这有利于地质学家查阅年龄和锶同位素的百分比 (见输出结果，本文只是粗糙的列举了 5 个，读者可以试试全部的)。如当年龄为 112.33691，即 112.33691 百万年前，锶同位素的百分比不超过 0.7072158519 的概率为 5%，不超过 0.7072603496 的概率为 50%，不超过 0.70732 的概率为 95%。需要注意的是从 `summary` 得到 `strontium.ratio` 数据前几位相同，即都是 0.707，所以我们利用 `(strontium.ratio - 0.707) * 10^3` 命令对 `strontium.ratio` 进行变换。

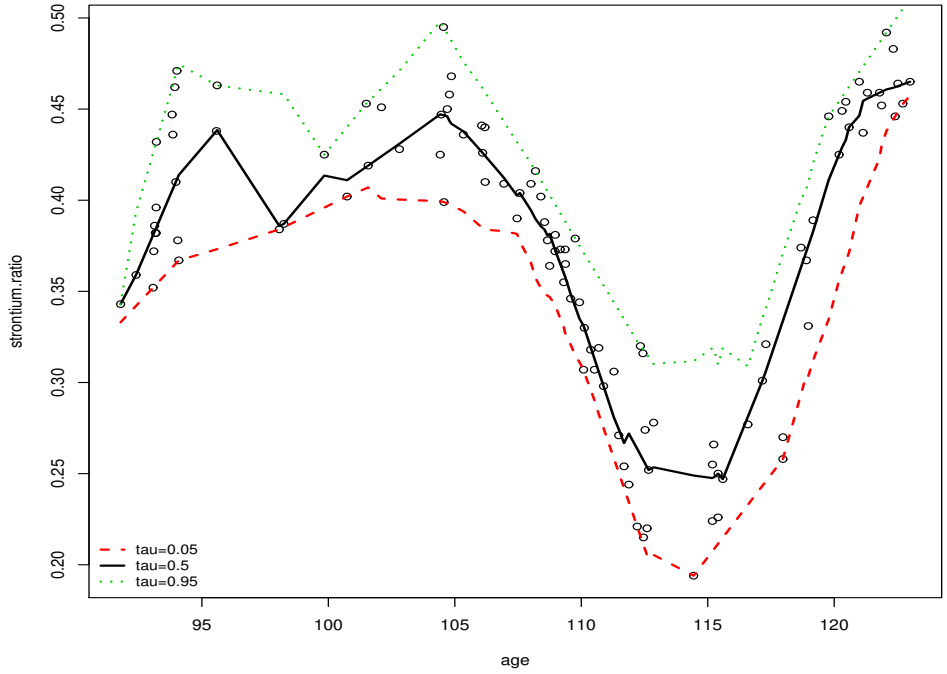

图 6.5: 数据 `fossil` 拟合图

R 代码

```
library(SemiPar)
library(KernSmooth)
library(quantreg)
data(fossil)
attach(fossil)
summary(fossil)
strontium.ratio <- (strontium.ratio - 0.707) * 10^3
#画图
```

```
win.graph(width=13, height=10,pointsize=10)
plot(age,strontium.ratio)
#带宽
h.mean <- dpill(x=age, y=strontium.ratio)
h.05 <- 1.34 * h.mean
h.95 <- h.05
h.50 <- 1.10 * h.mean
fit.05 <- lprq.my(age, strontium.ratio, h=h.05, tau=.05)
index <- order(fit.05$xx)
lines(fit.05$xx[index], fit.05$fv[index], col=2, lwd=2, pch=2, lty=2)
fit.50 <- lprq.my(age, strontium.ratio, h=h.50, tau=0.5)
lines(fit.50$xx[index], fit.50$fv[index], col=1, lwd=2, pch=1, lty=1)
fit.95 <- lprq.my(age, strontium.ratio, h=h.95, tau=0.95)
lines(fit.95$xx[index], fit.95$fv[index], col=3, lwd=2, pch=3, lty=3)
legend("bottomleft",legend=c("tau=0.05","tau=0.5","tau=0.95"),
       col=c(2,1,3),lty=c(2,1,3),lwd=c(2,2,2),cex=0.9, box.lty=0)
#查看部分拟合值
head(cbind(fit.05$xx,fit.05$fv,fit.50$fv,fit.95$fv))
```

输出结果

```
> summary(fossil)
      age         strontium.ratio
 Min.   : 91.79   Min.   :0.7072
 1st Qu.:104.43   1st Qu.:0.7073
 Median :109.48   Median :0.7074
 Mean   :108.78   Mean   :0.7074
 3rd Qu.:115.41   3rd Qu.:0.7074
 Max.   :123.00   Max.   :0.7075

> head(cbind(fit.05$xx,fit.05$fv,fit.50$fv,fit.95$fv))
          [,1]      [,2]      [,3]      [,4]
[1,]  91.78525 0.3330150 0.3430000 0.3430000
[2,]  92.39579 0.3420259 0.3590000 0.3933725
[3,]  93.97061 0.3652687 0.4100689 0.4690135
[4,]  95.57577 0.3730354 0.4380000 0.4631363
[5,]  95.60286 0.3731550 0.4384726 0.4630000
[6,] 112.33691 0.2158519 0.2603496 0.3200000
```

6.4 B 样条逼近

B 样条 (B spline) 是常见的非常熟方法。假设节点 (Knot) 为 $\xi_1, \ldots, \xi_K$，则 B 样条的基为

$$B_0(X) = 1,\ B_1(X) = X,\ B_2(X) = X^2,\ B_3(X) = X^3,$$
$$B_4(X) = (X - \xi_1)_+^3, \ldots, B_{K+4}(X) = (X - \xi_k)_+^3$$

其中 $B_0(X) = 1$ 是常数基。上面这个实际上是横截面基。关于 B 样本还有复杂形势的基函数。splines 包的 bs 可以实现。

```
bs(x, df = NULL, knots = NULL, degree = 3, intercept = FALSE, .....)
```

df: 表示基的个数
knots: 内点
intercept: 表示基中是否含有常数项。

大家可以试试下面命令看看出现什么情况。

比较基的两种形式

```
rm(list = ls())
library(splines)
n <- 1000
x <- rnorm(n)
y <- 3 * sin(x) + rnorm(n)

xik <- quantile(x, probs=c(0.3, 0.6))
BX_bs <- bs(x=x, knots=xik, intercept = FALSE) ##3+2=5

plot(x, BX_bs[, 1], ylim=c(min(BX_bs), max(BX_bs)), type="n",
     ylab=" B-spline basis")
for(k in 1:5){
  index <- order(x)
  points(x[index], BX_bs[index, k], type="l", col=k)
}

BX <- matrix(0, n, 5)
BX[, 1] <- x
BX[, 2] <- x ^ 2
BX[, 3] <- x ^ 3
BX[, 4] <- pmax(0, (x - xik[1]) ^ 3)
BX[, 5] <- pmax(0, (x - xik[2]) ^ 3)
plot(x, BX[, 1], ylim=c(min(BX), max(BX)), type="n",
     ylab=" B-spline basis")
for(k in 1:5){
  index <- order(x)
  points(x[index], BX[index, k], type="l", col=k)
}
```

```
29
30 fit_bs <- lm(y~BX_bs)
31 fit <- lm(y~BX)
32 ## 看看两者的模拟效果的差异
33 summary(abs(fit_bs$residuals - fit$residuals))
```

6.5 参考文献

1. 田茂再. 复杂数据统计推断理论、方法及应用. 科学出版社, 2014.
2. Bralower T. 等.(1997). Mid-Cretaceous Strontium-Isotope Stratigraphy of Deep-Sea Sections. Geological Society of America Bulletin, 109(11), 1421-1442
3. Chaudhuri P., Marron J. (1999). Sizer for Exploration of Structures in Curves. Journal of the American Statistical Association, 94(447), 807-823
4. Fan J., Hu T., Truong Y.(1994) Robust Non-Parametric Function Estimation. Scandinavian Journal of Statistics, Vol. 21(4), 433-446.
5. Hjϕrt N. , Pollard D.(1993). Asymptotics for Minimizers of Convex Processes. Statistical Research Report.
6. Härdle W. 等. Nonparametric and Semiparametric Models. Springer, 2004.
7. Ruppert D., Sheather, S., Wand M. (1995). An effective bandwidth selector for local least squares regression. Journal of the American Statistical Association, 90, 1257–1270.
8. Yu K. , Jones M. (1998).Local Linear Quantile Regression. Journal of the American Statistical Association, 93(441):228–237.

6.6 附录 1：NW 性质

因为

$$\widehat{m}_h(x) = \frac{\widehat{r}(x)}{\widehat{f}_X(x)}$$

讨论 NW 性质，可以先讨论 $\widehat{r}(x)$ 和 $\widehat{f_X}(x)$ 性质，再利用 Slutzky 定理，便可以得到 $\widehat{m}_h(x)$。由于 $\widehat{f_X}(x)$ 性质上一章已经讨论，所以只需要讨论 $\widehat{r}(x)$ 性质。

$$\begin{aligned}
E[\widehat{r}_h(x)] &= E[n^{-1}\sum_{i=1}^{n} K_h(x-X_i)Y_i] \\
&= E[K_h(x-X)Y] \\
&= \int\int yK_h(x-u)f(u,y)dudy \\
&= \int\int yK_h(x-u)f(y|u)f(u)dudy \\
&= \int K_h(x-u)f(u)\int yK_h(x-u)f(y|u)dy)du \\
&= \int K_h(x-u)f(u)\{E(Y|X=x)\}du \\
&= \int K_h(x-u)f(u)\{m(u)\}du \\
&= \int K_h(x-u)f(u)\{\frac{r(u)}{f(u)}\}du \\
&= \int K_h(x-u)r(u)du
\end{aligned}$$

令 $s=\frac{x-u}{h}$，则

$$\begin{aligned}
E[\widehat{r}_h(x)] &= \int \frac{1}{h}K(x-u)r(u)du \\
&= \int K(s)r(sh+x)ds \\
&= \int K(s)\{r(x)+r'(x)sh+r''(x)\frac{(sh)^2}{2}+o((sh)^2)\}ds \\
&= r(x)\int K(s)ds + r'(x)h\int sK(s)ds + \frac{h^2}{2}r''(x)\int s^2K(s)ds + o(h^2) \\
&= r(x)+0+\frac{h^2}{2}r''(x)\mu_2(K)+o(h^2)
\end{aligned}$$

$\widehat{r}(x)$ **的期望**
$E[\widehat{r}_h(x)] = r(x)+\frac{h^2}{2}r''(x)\mu_2(K)+o(h^2)\ \ h\to 0$

令 $s^2(x) = E[Y^2|X = x]$，则

$$
\begin{aligned}
& Var[\widehat{r}_h(x)] \\
&= Var[n^{-1}\sum_{i=1}^{n} K_h(x - X_i)Y_i] \\
&= \frac{1}{n^2} Var[\sum_{i=1}^{n} K_h(x - X_i)Y_i] \\
&= \frac{1}{n} Var[K_h(x - X)Y] \\
&= \frac{1}{n}\{E[K_h^2(x - X)Y^2] - E^2[K_h(x - X)Y]\} \\
&= \frac{1}{n}\{\int\int K_h^2(x - u)y^2 f(y|u)f(u)dydu - (\int\int K_h(x - u)yf(y|u)f(u)dydu)^2\} \\
&= \frac{1}{n}\{\int K_h^2(x - u)f(u)[\int y^2 f(y|u)dy]du - (\int K_h(x - u)r(u)du)^2\} \\
&= \frac{1}{n}\{\int K_h^2(x - u)f(u)[E[Y^2|X = x]]du - (\int K_h(x - u)r(u)du)^2\} \\
&= \frac{1}{n}\{\int K_h^2(x - u)f(u)[s^2(x)]du - (\int K_h(x - u)r(u)du)^2\} \\
&= \frac{1}{nh}\int K^2(t)f(x + th)s^2(x + th)dt + o(\frac{1}{nh}) \\
&= \frac{1}{nh} f(x)s^2(x)||K||_2^2 + o(\frac{1}{nh})
\end{aligned}
$$

$\widehat{r}(x)$ **的方差**
$Var[\widehat{r}_h(x)] = \frac{1}{nh} f(x)s^2(x)\|\|K\|\|_2^2 + o(\frac{1}{nh})\ \ nh \to \infty$

所以

$\widehat{r}_h(x)$ **的均方误差**
$MSE[\widehat{r}_h(x)] = \frac{1}{nh} f(x)s^2(x)\|\|K\|\|_2^2 + \frac{h^4}{2}(r''(x)\mu_2(K))^2 + o(h^4) + o(\frac{1}{nh})\ \ nh \to \infty h \to 0$

如果 $nh \to \infty$ 使得 $h \to 0$，则

$$MSE[\widehat{r}_h(x)] \to 0$$

所以

$$\widehat{r}_h(x) \xrightarrow{p} r(x)$$

上一章我们可以得到 $\widehat{f}_h(x) \xrightarrow{p} f(x)$。根据 Slutzky 定理得

$$\widehat{m}_h(x) = \frac{\widehat{r}(x)}{\widehat{f}_X(x)} \xrightarrow{p} \frac{r(x)}{f_X(x)} = m(x)\ \ nh \to \infty h \to 0$$

NW 估计的一致性
当 $nh \to \infty\ h \to 0$，$\widehat{m}(x)$ 是一致估计

我们计算 $\widehat{m}(x)$ 的均方误差有点困难，我们可以借助 $\widehat{r}_h(x)$ 得到。

$$
\begin{aligned}
\widehat{m}_h(x) - m(x) &= (\frac{\widehat{r}_h(x)}{\widehat{f}_X(x)} - m(x))(\frac{\widehat{f}_h(x)}{f_X(x)} + (1 - \frac{\widehat{f}_h(x)}{f_X(x)})) \\
&= (\frac{\widehat{r}_h(x)}{\widehat{f}_X(x)} - m(x))\frac{\widehat{f}_h(x)}{f_X(x)} + (\frac{\widehat{r}_h(x)}{\widehat{f}_X(x)} - m(x))\frac{f(x) - \widehat{f}_h(x)}{f(x)} \\
&= (\frac{\widehat{r}_h(x) - m(x)\widehat{f}_X(x)}{f_X(x)}) + (\widehat{m}_h(x) - m(x))\frac{f(x) - \widehat{f}_h(x)}{f(x)}
\end{aligned}
$$

取 $h \sim n^{-1/5}$，则

$$
\begin{aligned}
&\widehat{r}_h(x) - m(x)\widehat{f}_h(x) \\
&= \widehat{r}_h(x) - r(x) - m(x)\widehat{f}_h(x) + r(x) \\
&= \widehat{r}_h(x) - r(x) - m(x)\widehat{f}_h(x) + m(x)f(x) \\
&= \widehat{r}_h(x) - r(x) - m(x)(\widehat{f}_h(x) + f(x)) \\
&= r(x) + \frac{1}{h^2}r''(x)\mu_2(K) + o(h^2) - r(x) - m(x)(f(x) + \frac{h^2}{2}f^{(2)}(x)\mu_2(K) + o(h^2) - f(x)) \\
&= \frac{1}{h^2}r''(x)\mu_2(K) + o(h^2) - m(x)(\frac{h^2}{2}f^{(2)}(x)\mu_2(K) + o(h^2)) \\
&= O_p(n^{-2/5}) - m(xO_p(n^{-2/5}) \\
&= O_p(n^{-2/5})
\end{aligned}
$$

上面讨论计算 $\widehat{m}(x)$ 有点复杂，我们下面介绍另一种方法，直接证明：

$$
\begin{aligned}
&E[n^{-1}\sum_{i=1}^{n} K_h(x - X_i)Y_i] \\
&= E[K_h(x - X)Y] \\
&= \int\int yK_h(x-u)f(u,y)dudy \\
&= \int\int yK_h(x-u)f(y|u)f(u)dudy \\
&= \int K_h(x-u)f(u)\int yK_h(x-u)f(y|u)dy)du \\
&= \int K_h(x-u)f(u)\{E(Y|X=x)\}du \\
&= \int K_h(x-u)f(u)\{m(u)\}du \\
&= \int K(s)f(x+sh)m(x+sh)ds \\
&= \int K(s)[f(x) + shf(x) + \frac{s^2h^2}{2}f''(x) + o(h^2)][m(x) + shm(x) + \frac{s^2h^2}{2}m''(x) + o(h^2)]ds \\
&= \int K(s)[f(x)m(x) + shf(x)m(x) + \frac{s^2h^2}{2}f(x)m''(x) + o(h^2) \\
&+ shf(x)m(x) + shf(x)shm(x) + shf(x)\frac{s^2h^2}{2}m''(x) + shf(x)o(h^2) \\
&+ \frac{s^2h^2}{2}f''(x)m(x) + \frac{s^2h^2}{2}f''(x)shm(x) + \frac{s^2h^2}{2}f''(x)\frac{s^2h^2}{2}m''(x) + o(h^2)ds \\
&\approx f(x)m(x) + 0 + f(x)m''(x)\frac{h^2}{2}\mu_2(K) + 0 + f'(x)m'(x)h^2\mu_2(K) + 0 \\
&+ f''(x)m(x)\frac{h^2}{2}\mu_2(K) + 0 + \frac{h^4}{4}f''(x)m''(x)\int s^4K(s)ds \\
&= f(x)m(x) + h^2\mu_2(K)[f(x)m''(x)/2 + f'(x)m'(x) + f''(x)m(x)/2] + o(h^2)
\end{aligned}
$$

因为

$$E\{\widehat{f}_h(x)\} = f(x) + \frac{h^2}{2}f''(x)\mu_2(K) + o(h^2)$$

所以：

$$
\begin{aligned}
E\widehat{m}(x) &\approx \frac{E\int y\widehat{f}(x,y)dy}{E\widehat{f}_X(x)} \\
&\approx \frac{f(x)m(x)+h^2\mu_2(K)[f(x)m''(x)/2+f'(x)m'(x)+f''(x)m(x)/2]}{f(x)+\frac{h^2}{2}f''(x)\mu_2(K)} \\
&= \frac{f(x)[m(x)+h^2\mu_2(K)[f(x)m''(x)/2f(x)+f'(x)m'(x)/f(x)+f''(x)m(x)/2f]]}{f(x)[1+\frac{h^2}{2f(x)}f''(x)\mu_2(K)]} \\
&= \frac{m(x)+h^2\mu_2(K)[m''(x)/2+f'(x)m'(x)/f(x)+f''(x)m(x)/2f]}{1+\frac{h^2}{2f(x)}f''(x)\mu_2(K)} \\
&= \frac{m(x)+h^2\mu_2(K)[m''(x)/2+f'(x)m'(x)/f(x)]+m(x)h^2f''(x)\mu_2(K)/2f(x)}{1+\frac{h^2}{2f(x)}f''(x)\mu_2(K)}
\end{aligned}
$$

令

- $E=h^2\mu_2(K)[m''(x)/2+f'(x)m'(x)/f(x)]$
- $F=h^2f''(x)\mu_2(K)/2f(x)$

因为当 $h\to 0$ 时，$(1+ch^2)^{-1}\approx 1-ch^2$，所以 $(1+F)^{-1}\approx(1-F)$，从而

$$
\begin{aligned}
E\widehat{m}(x) &\approx [m(x)+E+m(x)F][1-F] \\
&= m(x)+E+m(x)F-m(x)F-EF-m(x)F^2 \\
&= m(x)+E-EF-m(x)F^2 \\
&= m(x)+E+o(h^2)
\end{aligned}
$$

NW 核估计的期望
$E\widehat{m}(x)=m(x)+\frac{h^2}{2}\mu_2(K)[m''(x)+\frac{2f'(x)m'(x)}{f(x)}]+o(h^2)\ h\to 0$

第 7 章 多元参数回归

前面介绍了一维的非参数回归，下面介绍多维模型。在这之前我们先介绍“维度灾难”。一般说，维度越高，需要恢复或者还原真实模型越难，需要的样本量越大。假设我们在 $[0,1]$ 抽取样本，要求两点之间的距离小于 0.01，则至少需要 10^2 个点。将二维空间进行分格，要求每一个格子之间的距离不小于 0.01，则至少需要 10^4 个点。一直到 p 维，那么需要至少 10^{2p} 个点。我们可以看出维数越大，样本量越大。

为了处理这个灾难，统计学通过降维方法进行解决。假设设 Y 是因变量，$\boldsymbol{X} = (X_1, \ldots, X_p)^\top$ 是 p 维自变量,U 是指示变量 (Index Variable)。

- 单指标模型 (Single index models):

$$E(Y|X) = g(X^\top \beta)$$

  其中 β 是未知 p 维参数，g 是未知函数。该模型通过 $X^\top\beta$ 将 p 维变量转化成一维变量，从而达到降低维度的效果。

- 可加模型 (Additive models)

$$E(Y|X) = f_1(X_1) + \cdots + f_p(X_p)$$

  该模型通过每一个维度建模，不是 p 维全部建模。

- 变系数模型 (Varying Coefficient Model)

$$E(Y|X,U) = \alpha_1(U)X_1 + \cdots + \alpha_p(U)X_p$$

第 8 章 变系数模型

变系数模型 (Varying Coefficient Model) 是线性模型的推广，最早由 Shumway (1988) 提出，Hastie 和 Tibshirani (1993) 进行了详细的研究，其拓展了局部回归技术从一维到高维的应用 (Fan 和 Zhang 2008)。与经典的线性模型相比，变系数模型具有以下优点：（1）不需要假设误差项服从某个分布，而经典线性模型需要误差项来自于正态分布；（2）它不像经典线性回归那样，假设自变量与因变量成线性关系，它只假设自变量对因变量有影响，但是这种影响不再是固定的和线性的，它是随着某个指示变量（如时间）而变化，并且这种变化是未知的。可见，变系数模型放宽了模型的假设，更加自由和灵活。

令 $\{(Y_i, \boldsymbol{X}_i, U_i), i = 1, 2, \ldots, n\}$ 是来自 $\{Y, \boldsymbol{X}, U\}$ 的观测值，其中 $Y_i \in R$ 是因变量，$\boldsymbol{X}_i = (X_{i1}, X_{i2}, \ldots, X_{ip})^\top \in \boldsymbol{R}^p$ 是 p 维自变量。如果需要可以假定 $X_{i1} \equiv 1$。$U_i \in R$ 是一维的指示变量 (Univariate Index Variable)，如测量时间等。我们考虑如下变系数函数模型 (Varying Coefficient Models,VCM):

$$Y_i = \boldsymbol{X}_i^\top \boldsymbol{a}(U_i) + \varepsilon_i \tag{8.1}$$

其中 ε_i 是随机误差项，并且满足 $E(\varepsilon_i | \boldsymbol{X}_i, U_i) = 0$。

对于变系数函数模型，我们利用局部线性光滑技术，即将 $a_j(v)$ 在 u 点展开

$$a_j(v) \approx a_j(u) + b_j(u)(v - u),$$

其中 $b(\bullet) = a'(\bullet)$。变系数模型是求下面目标函数的最小值:

$$\sum_{i=1}^{n} \left\{ Y_i - \boldsymbol{X}_i^\top \boldsymbol{a}(U) - (U_i - U)\boldsymbol{X}_i^T \boldsymbol{b}(U) \right\}^2 K_h(U_j - U)$$

其中 $K_h(\bullet) = \frac{1}{h} K(\frac{\bullet}{h})$ 是核函数，h 是带宽 (Bandwidth)。

令 W_t=diag$\Big((U_1 - U)^t K_h(U_2 - U), \ldots, (U_n - U)^t K_h(U_i - U)\Big)$，通过求导等运算，可以得到:

$$\begin{pmatrix} \mathbf{a}(u) \\ \mathbf{b}(u) \end{pmatrix} = \begin{pmatrix} \mathbf{X}^\top W_0 \mathbf{X} & \mathbf{X}^\top W_1 \mathbf{X} \\ \mathbf{X}^\top W_1 \mathbf{X} & \mathbf{X}^\top W_2 \mathbf{X} \end{pmatrix}^{-1} \begin{pmatrix} \mathbf{X}^\top W_0 \mathbf{Y} \\ \mathbf{X}^\top W_1 \mathbf{Y} \end{pmatrix}$$

h 比较重要，其越大估计越光滑，但估计偏差会增大。我们利用 Cai 等 (2000) 提出的多核实交叉核实的方法 (Modified Multi-fold Cross-validation) 选择 h，其是最小化:

$$AMS(h_2) = \sum_{q=1}^{Q} AMS_q(h_2)$$

其中

$$AMS_q(h_2) = \frac{1}{m} \sum_{i=n-qm+1}^{n-qm+m} \left\{ Y_i - \boldsymbol{X}_i^\top \widehat{\boldsymbol{a}}_q(U_i) \right\}^2$$

和 $\widehat{\boldsymbol{a}}_q$ 利用 $\{\boldsymbol{X}_i, Y_i, U_i\}_{i=1}^{n-qm}$ 计算得到。Cai 等 (2000) 建议 $m = [0.1n]$ 和 $Q = 4$ ($[c]$ 表示 c 的整数部分)。主要 $n > Qm$。下面我们以一个模拟的例子说明如何进行编程。

我们考虑如下变系数模型

$$Y_i = 8\exp(-(U_i - 0.5)^2)X_{i1} + 2\sin(2\pi U_i) + 5U_i(1 - U_i)X_{i2} + \varepsilon_i$$

其中 $X_{i1} = 1$，$(X_{i2}, \ldots, X_{i3})^T$ 是独立同分布，来自于标准正态分布，$\varepsilon_i \overset{\text{iid}}{\sim} N(0, 1)$，$U$ 是指示变量，其来自于 [0,1] 区间的均匀分布。我们设置样本量 $n = 200$。

R 代码

```
library(MASS)
```

```
library(limSolve)
#产生数据开始
Data1 <- function(n)
{
  p <- 2
  #sigm=0.5^abs(outer(1:3,1:3,"-"))
  sigm  <- diag(2)
  muz <- rep(0,2)
  x=mvrnorm(n = n, mu=muz, Sigma=sigm)
  x <- cbind(1,x)
  e <- rnorm(n,0,0.8)
  u <-  runif(n,0,1)
  beta0 <- 8*exp(-(u-0.5)^2)
  beta1 <- 2*cos(2*pi*u)
  beta2 <- 5*(u-0.5)^2
  yture <- beta0 + beta1 * x[,2] + beta2 * x[,3]
  y <- yture + e
  dat = list(y=y, u=u, x=x,beta0=beta0,beta1=beta1,beta2 =beta2)
  return(dat)
}
#产生数据结束


####变系数函数大函数开始
#核函数开始
kern <- function(t){
  0.75*pmax(1-t^2,0)
}
#核函数结束

#W函数开始
#这个函数可以不需要for循环，这样编写方便理解
wwi=function(n,s,j,h){
  ww=NULL
  for(i in 1:n){
    ww[i]=(u[i]-u[j])^s*kern((u[i]-u[j])/h)
  }
  ww
}
#W函数结束

#变系数函数开始
VCM=function(y,x,u,h){
```

```
  n=dim(x)[1]
  p=dim(x)[2]
  AB=matrix(0, nrow =2*p, ncol = n)
  for(j in 1:n){
    w0=w1=w2=matrix(0,n,n)
    diag(w0)=wwi(n=n,s=0,j=j,h=h)
    diag(w1)=wwi(n=n,s=1,j=j,h=h)
    diag(w2)=wwi(n=n,s=2,j=j,h=h)
    #合并矩阵a
    a11=t(x)%*%w0%*%x
    a12=t(x)%*%w1%*%x
    a22=t(x)%*%w2%*%x
    aa1=cbind(a11,a12)
    aa2=cbind(a12,a22)
    a=rbind(aa1,aa2)
    ###合并b
    b11=t(x)%*%w0%*%y
    b21=t(x)%*%w1%*%y
    b=rbind(b11,b21)
    AB[,j]=Solve(a) %*% b
  }
  list(AB=AB)
}
#变系数函数结束

##编写固定的h的MCV函数开始
VCM.MCV=function(y,x,u,h){
  n <- dim(x)[1]
  p <- dim(x)[2]
  m <- floor(0.1*n)
  Q <- 4
  AMS <- NULL
  for (q in 1:Q){
    ind  <- n - q * m
    xx <- x[1:ind, ]
    yy <- y[1:ind]
    uu <- u[1:ind]
    AB=matrix(0, nrow =2*p, ncol = m)
    for(j in 1:m){
      w0=w1=w2=matrix(0,ind,ind)
      diag(w0)=wwi(n=ind,s=0,j=ind+j,h=h)
      diag(w1)=wwi(n=ind,s=1,j=ind+j,h=h)
      diag(w2)=wwi(n=ind,s=2,j=ind+j,h=h)
```

```
      #合并矩阵a
      a11=t(xx)%*%w0%*%xx
      a12=t(xx)%*%w1%*%xx
      a22=t(xx)%*%w2%*%xx
      aa1=cbind(a11,a12)
      aa2=cbind(a12,a22)
      a=rbind(aa1,aa2)
      ###合并b
      b11=t(xx)%*%w0%*%yy
      b21=t(xx)%*%w1%*%yy
      b=rbind(b11,b21)
      AB[,j]=Solve(a) %*% b
    }
    AMS0=NULL
    for(k in 1:m){
      AMS0[k] <- y[ind+k] - x[ind+k,] %*% AB[1:3,k]
    }
    AMS[q] <- mean(AMS0^2)
  }
  list(h=h,AMS=sum(AMS))
}
##编写固定的h的MCV函数结束

##编写多个h的MCV函数开始
AMS <- function(y,x,u,h=seq(0.01,0.2,by=0.02)){
  MAMS <- NULL
  for(t in 1:length(h)){
    MAMS[t] <- VCM.MCV(y=y,x=x,u=u,h=h[t])$AMS
  }

  idex=which(MAMS==min(MAMS))
  h.opt <- h[idex]
  list(h=h.opt)
}
##编写多个h的MCV函数结束

n <- 200
dat <- Data1(n)
x <- dat$x
y <- dat$y
u <- dat$u
beta0 <- dat$beta0
beta1 <- dat$beta1
```

```
beta2 <- dat$beta2

hopt <- AMS(y=y,x=x,u=u)$h #选择h
hopt#查看h

fit=VCM(y=y,x=x,u=u,h=hopt)#拟合VCM

par(mfrow=c(1,3))#图省略
uu <- order(u)
plot(u[uu],fit$AB[1,][uu],ylim=c(6,8.5),ylab="a1",type="o",xlab="")
points(u[uu],beta0[uu],col=2)
plot(u[uu],fit$AB[2,][uu],ylim=c(-2,3), ylab="a2",type="o",xlab="")
points(u[uu],beta1[uu],col=2)
plot(u[uu],fit$AB[3,][uu],ylim=c(-1,3),ylab="a3",type="o",xlab="")
points(u[uu],beta2[uu],col=2)

mean((fit$AB[1,]-beta0)^2)#查看估计差异
mean((fit$AB[2,]-beta1)^2)
mean((fit$AB[3,]-beta2)^2)
```

输出结果

```
> hopt#查看h
[1] 0.11
> mean((fit$AB[1,]-beta0)^2)#查看估计差异
[1] 0.01669084
> mean((fit$AB[2,]-beta1)^2)
[1] 0.01960522
> mean((fit$AB[3,]-beta2)^2)
[1] 0.03013411
```

8.1 变量选择

设 Y 是因变量，$\boldsymbol{X}$ 是 p 维自变量，U 是指示变量 (Index Variable)。其中 $\boldsymbol{X}$ 被分成两部分，$\boldsymbol{X}^1$ 和 $\boldsymbol{X}^2$。$\boldsymbol{X}^1$ 和 Y 呈非线性关系，$\boldsymbol{X}^2$ 和 Y 呈线性关系。假设 $(Y_i, \mathbf{X}_i^1, \boldsymbol{X}_i^2, U_i), i = 1, 2, \ldots, n$ 是独立同分布。我们考虑部分线性变系数模型:

$$Y_i = \mathbf{X}_i^{1T}\boldsymbol{\alpha}(U_i) + \mathbf{X}_i^{2T}\boldsymbol{\beta} + \varepsilon_i \tag{8.2}$$

其中 $\mathbf{X}_i^1 = (X_{i1}^1, X_{i2}^1, \ldots, X_{ip_1}^1)^T \in R^{p_1}$ 是 p_1 维的自变量向量，$\mathbf{X}_i^2 = (X_{i1}^2, X_{i2}^2, \ldots, X_{i,p_2}^2)^T \in R^{p_2}$ 是 p_2 维的自变量向量，且 $X_{i1}^1 \equiv 1$，$p_1 + p_2 = p$；$U_i \in R$ 是指示变量，如时间等；$\boldsymbol{\alpha}(U_i)$ 称为变系数函数或简称变系数，$\boldsymbol{\beta}$ 称为常系数函数或简称常系数。ε_i 是随机误差项，其满足 $E(\varepsilon_i|\boldsymbol{X}_i, U_i) = 0$。

对于（部分线性）变系数模型，主要分成三个研究问题：(1) 如何识别变系数和常系数函数，(2) 变系数函数自变量的选择，(3) 常系数函数自变量的选择。对于问题 (1)，Xia 等 (2004) 基于交叉验证 (CV, Cross Validation) 提出了逐步选择方法。Hu 和 Xia (2012) 结合局部多项式光滑和 GLASSO 方法提出区分变系数函数和常系函数。另外，Hohsusuk 和 Ingrid(2012) 也进行了研究。对于问题 (2)，Wang 和 Xia(2009) 结合局部多项式光滑和 LASSO 提出一种新的变系数模型（模型**??**）的变量选择方法，该方法可以在估计非参数形式的同时进行变量选择。Leng(2009) 提出利用惩罚似然对变系数模型（模型**??**）进行变量选择。对部分线性变系数模型（模型8.2），Zhao 和 Xue(2009) 结合基函数逼近和 SCAD 惩罚选择常系数函数和变系数函数对应的变量；他们解决问题 (1) 和 (2)。另外，Li 和 Liang(2008) 研究了广义部分变系数模型的变量选择，但他们只对常系数函数的自变量进行选择，没对变系数函数的自变量进行选择（解决问题 (3)）。Tang 等 (2012) 利用 B 样条逼近和 AdapLASSO，采用两部估计方法解决了问题 (1)、(2) 和 (3)。Wang 和 Kulasekera(2012) 利用局部光滑和 LASSO 也解决了问题上述三个问题。目前如何实现变系数模型的统一变量选择 (Unified Variable Selection)，即识别变系数和常系数函数，选择非零的变系数函数和常系数函数的自变量，仍是研究的难点和热点。

8.2 引言

变系数函数模型是非参数模型的重要组成部分，其可以有效的避免维数灾难。一直以来困难人们的问题，不仅仅有变系数函数和常系数函数的变量选择，还有变系数函数和常系数函数的识别，尤其是后者。一般对于后者的处理往往是根据经验得到，即经验表明某些自变量和因变量呈线性关系，某些自变量与因变量呈非线性关系。但是这种方法具有主观性。如何有效的识别变系数函数和常系数函数是研究的难点。本章利用常系数函数导数为零的思想和 SCAD 压缩技术解决这一难点问题。另外，Oracle 性质是一个非常好的性质，目前统一变量选择的方法并不具有这样的性质，本章将证明提出的方法具有该性质。

本章主要研究的问题：变系数模型的统一变量选择。

8.3 方法

令 $\{(Y_i, \boldsymbol{X}_i, U_i), i = 1, 2, \ldots, n\}$ 是来自 $\{Y, \boldsymbol{X}, U\}$ 的观测值，其中 $Y_i \in R$ 是因变量，$\boldsymbol{X}_i = (X_{i1}, X_{i2}, \ldots, X_{ip})^T \in \boldsymbol{R}^p$ 是 p 维自变量。如果需要可以假定 $X_{i1} \equiv 1$。$U_i \in R$ 是一维的指示变量 (Univariate Index Variable)，如测量时间等。本章考虑如下变系数函数模型 (Varying Coefficient Models,VCM):

$$Y_i = \boldsymbol{X}_i^T \boldsymbol{a}(U_i) + \varepsilon_i \tag{8.3}$$

其中 ε_i 是随机误差项，并且满足 $E(\varepsilon_i | \boldsymbol{X}_i, U_i) = 0$。

参数矩阵记为 $\mathbf{A}$ 和 $\mathbf{B}$，其中 $b_j(u_i) = a_j'(u_i)$。用 $\mathbf{a}(u_j) = (a_1(u_j), a_2(u_j), \ldots, a_p(u_j))^T$ 表示 $\mathbf{A}$ 第 j 个行；$\mathbf{a}_i(u) = (a_i(u_1), a_i(u_2), \ldots, a_i(u_n))^T$ 表示 $\mathbf{A}$ 第 i 个列；$\mathbf{b}_j(u) = (b_j(u_1), b_j(u_2), \ldots, b_j(u_n))^T$ 表示 $\mathbf{B}$ 第 j 个列；$||\mathbf{a}_k|| = (\sum_{i=1}^n a_k^2(u_i))^{\frac{1}{2}}$ 表示 A 的第 k 列元素的模，$||\mathbf{b}_k|| = (\sum_{i=1}^n b_k^2(u_i))^{\frac{1}{2}}$ 表示 B 的第 k 列元素的模。

$$\mathbf{A}_{n\times p}=\begin{pmatrix} a_1(u_1) & a_2(u_1) & \dots & a_p(u_1)\\ a_1(u_2) & a_2(u_2) & \dots & a_p(u_2)\\ \vdots & \vdots & \ddots & \vdots\\ a_1(u_n) & a_2(u_n) & \dots & a_p(u_n)\end{pmatrix}$$

$$\mathbf{B}_{n\times p}=\begin{pmatrix} b_1(u_1) & b_2(u_1) & \dots & b_p(u_1)\\ b_1(u_2) & b_2(u_2) & \dots & b_p(u_2)\\ \vdots & \vdots & \ddots & \vdots\\ b_1(u_n) & b_2(u_n) & \dots & b_p(u_n)\end{pmatrix}$$

调整参数用 $\boldsymbol{\lambda_1}$ 和 $\boldsymbol{\lambda_2}$ 表示，其中 $\boldsymbol{\lambda}_1=(\lambda_{11},\lambda_{12},\dots,\lambda_{1p})^T$，$\boldsymbol{\lambda}_2=(\lambda_{21},\lambda_{22},\dots,\lambda_{2p})^T$

8.3.1 惩罚最小二乘方法

对于变系数函数模型，本文利用局部线性光滑技术，即将 $a_j(v)$ 在 u 点展开

$$a_j(v)\approx a_j(u)+a_j'(u)(v-u),$$

同时对 $||\mathbf{a}_k||$ 和 $||\mathbf{b}_k||$ 进行惩罚；所以，提出的方法是求下面目标函数的最小值：

$$\begin{aligned}Q_{\lambda_1,\lambda_2}(A,B)=&\sum_{j=1}^{n}\sum_{i=1}^{n}\left\{Y_i-\boldsymbol{X}_i^T\boldsymbol{a}(U_j)-(U_i-U_j)\boldsymbol{X}_i^T\boldsymbol{b}(U_j)\right\}^2K_h(U_i-U_j)\\&+\sum_{k=1}^{p}p_{\lambda_{1k}}(||\boldsymbol{a}_k||)+\sum_{l=1}^{p}p_{\lambda_{2l}}(||\boldsymbol{b}_l||)\end{aligned}\tag{8.4}$$

其中 $K_h(\bullet)=\frac{1}{h}K(\frac{\bullet}{h})$ 是核函数，h 是带宽 (Bandwidth)，$p_\lambda(\bullet)$ 为惩罚函数。第一项惩罚 $\sum_{k=1}^{p}p_{\lambda_{1k}}(||\mathbf{a}_k||)$ 是将 $\boldsymbol{a}_i(u)$ 压缩为 0，即惩罚变系数函数模型参数 (变系数函数和常系数函数)。第二项惩罚 $\sum_{l=1}^{p}p_{\lambda_{1l}}(||\mathbf{b}_k||)$ 是将 $\boldsymbol{b}_i(u)$ 压缩为 0，即惩罚变系数函数模型参数的导数。由于常函数的导数为零，所以，第二个惩罚可以区分变系数函数和常系数函数，这也意味着第一项惩罚可以实现对于常系数函数和变系数函数对应变量的选择。

常见的罚函数有 L_2 罚函数：$p_\lambda(\theta)=\lambda|\theta|^2$, L_1 罚函数：$p_\lambda(\theta)=\lambda|\theta|$ 和 SCAD 罚函数：

$$p_\lambda'(w)=\lambda\left\{I(w\le\lambda)+\frac{(\alpha\lambda-w)_+}{(\alpha-1)\lambda}I(w>\lambda)\right\}\quad\alpha>2,w>0.$$

Fan 和 Li (2001) 曾论证 SCAD 优于 L_2 和 L_1 罚函数。所以，本文使用 SCAD 罚函数。

8.3.2 计算方法

因为 $Q_{\lambda_1,\lambda_2}(\mathbf{A},\mathbf{B})$ 中的惩罚函数在原点奇异，所以普通的求导计算方法将不能直接利用。本文根据 Fan 和 Li(2001) 二次逼近的方法提出一种迭代算法。设 $w_0\in U(w,\sigma)(\sigma>0)$ 且 $w_0\neq 0$，则 $p_\lambda(\bullet)$ 可以渐近的表示为：

$$p_\lambda(|w|)\approx p_\lambda(|w_0|)+\frac{p_\lambda'(|w_0|)}{2|w_0|}(w^2-w_0^2)$$

所以，目标函数 (8.4) 可以转化为:

$$\begin{aligned}
Q_{\lambda_1,\lambda_2}(\mathbf{A},\mathbf{B}) \approx & \sum_{j=1}^{n}\sum_{i=1}^{n}\left\{Y_i - \mathbf{X}_i^T\mathbf{a}(U_j) - (U_i-U_j)\mathbf{X}_i^T\mathbf{a}'(U_j)\right\}^2 K_h(U_i-U_j) \\
& + \sum_{k=1}^{p}\left(p_{\lambda_{1k}}(||\mathbf{a}_{0k}||) + \frac{p'_{\lambda_{1k}}(||\mathbf{a}_{0k}||)}{2||\mathbf{a}_{0k}||}(||\mathbf{a}_k||^2 - ||\mathbf{a}_{0k}||^2)\right) \\
& + \sum_{l=1}^{p}\left(p_{\lambda_{2l}}(||\mathbf{b}_{0l}||) + \frac{p'_{\lambda_{2l}}(||\mathbf{b}_{0l}||)}{2||\mathbf{b}_{0l}||}(||\mathbf{b}_l||^2 - ||\mathbf{b}_{0l}||^2)\right) \\
& \doteq \Delta_1 + \Delta_2 + \Delta_3
\end{aligned}$$

$Q_{\lambda_1,\lambda_2}(\mathbf{A},\mathbf{B})$ 关于 $\mathbf{a}(U_j)$ 和 $\mathbf{a}'(U_j)$ 求导，可以得到 $\widehat{\mathbf{A}}$ 和 $\widehat{\mathbf{B}}$。由于 Δ_3 和 $\mathbf{a}(U_j)$ 没关系，所以关于 $Q_{\lambda_1,\lambda_2}(\mathbf{A},\mathbf{B})$ 对 $\mathbf{a}(U_j)$ 求导等价于 Δ_1 和 Δ_2 对 $\mathbf{a}(U_j)$ 求导；同理
$Q_{\lambda_1,\lambda_2}(\mathbf{A},\mathbf{B})$ 对 $\mathbf{a}'(U_j)$ 求导等价于 Δ_1 和 Δ_3 对 $\mathbf{a}'(U_j)$ 求导。

由于

$$\begin{aligned}
\frac{\partial\Delta_1}{\partial\mathbf{a}(u_j)} &= -2\sum_{i=1}^{n}\mathbf{X_i}\left\{Y_i - \mathbf{X}_i^T\mathbf{a}(U_j) - (U_i-U_j)\mathbf{X}_i^T\mathbf{a}'(U_j)\right\}K_h(U_i-U_j) \\
&= -2\sum_{i=1}^{n}\mathbf{X_i}Y_iK_h(U_i-U_j) + 2\sum_{i=1}^{n}\mathbf{X_i}\mathbf{X_i}^T\mathbf{a}(U_j)K_h(U_i-U_j) \\
&\quad + 2\sum_{i=1}^{n}\mathbf{X_i}(U_i-U_j)\mathbf{X}_i^T\mathbf{a}'(U_j)K_h(U_i-U_j)
\end{aligned}$$

$$\begin{aligned}
\frac{\partial\Delta_1}{\partial\mathbf{a}'(U_j)} &= -2\sum_{i=1}^{n}(U_i-U_j)\mathbf{X_i}\left\{Y_i - \mathbf{X}_i^T\mathbf{a}(U_j) - (U_i-U_j)\mathbf{X}_i^T\mathbf{a}'(U_j)\right\}K_h(U_i-U_j) \\
&= -2\sum_{i=1}^{n}(U_i-U_j)\mathbf{X_i}Y_iK_h(U_i-U_j) + 2\sum_{i=1}^{n}(U_i-U_j)\mathbf{X_i}\mathbf{X_i}^T\mathbf{a}(U_j) \\
&\quad + 2\sum_{i=1}^{n}\mathbf{X_i}(U_i-U_j)^2\mathbf{X}_i^T\mathbf{a}'(U_j)K_h(U_i-U_j)
\end{aligned}$$

$$\begin{aligned}
\frac{d\Delta_2}{d\mathbf{a}(U_j)} &= \begin{pmatrix} \frac{p'_{\lambda_{11}}(||\mathbf{a}_{01}||)}{2||\mathbf{a}_{01}||}2a'_1(U_j) \\ \frac{p'_{\lambda_{12}}(||\mathbf{a}_{02}||)}{2||\mathbf{a}_{02}||}2a'_2(U_j) \\ \vdots \\ \frac{p'_{\lambda_{1p}}(||\mathbf{a}_{0p}||)}{2||\mathbf{a}_{0p}||}2a'_p(U_j) \end{pmatrix} \\
&= D_1\mathbf{a}(U_j)
\end{aligned}$$

其中 $D_1 = diag\left(\frac{p'_{\lambda_{11}}(||\boldsymbol{a}_1||)}{||\boldsymbol{a}_1||},\ldots,\frac{p'_{\lambda_{1p}}(||\boldsymbol{a}_p||)}{||\boldsymbol{a}_p||}\right)$

$$\begin{aligned}
\frac{d\Delta_3}{d\mathbf{a}'(u_j)} &= \begin{pmatrix} \frac{p'_{\lambda_{11}}(||\mathbf{b}_{01}||)}{2||\mathbf{b}_{01}||}2b'_1(U_j) \\ \frac{p'_{\lambda_{12}}(||\mathbf{b}_{02}||)}{2||\mathbf{b}_{02}||}2b'_2(U_j) \\ \vdots \\ \frac{p'_{\lambda_{1p}}(||\mathbf{b}_{0p}||)}{2||\mathbf{a}_{0p}||}2b'_p(U_j) \end{pmatrix} \\
&= D_2\mathbf{a}'(u_j)
\end{aligned}$$

其中 $D_2 = \mathrm{diag}\left(\frac{p'_{\lambda_{21}}(||\boldsymbol{b}_1||)}{||\boldsymbol{b}_1||},\ldots,\frac{p'_{\lambda_{2p}}(||\boldsymbol{b}_p||)}{2||\boldsymbol{b}_p||}\right)$，所以

$$\begin{pmatrix}\mathbf{a}(U_j)\\ \mathbf{a}'(U_j)\end{pmatrix}=\begin{pmatrix}\sum_{i=1}^n \mathbf{X_i}\mathbf{X_i}^T K_h(U_i-U_j)+\frac{1}{2}D_1 & \sum_{i=1}^n \mathbf{X_i}(U_i-U_j)\mathbf{X}_i^T K_h(U_i-U_j)\\ \sum_{i=1}^n (U_j-U_i)\mathbf{X_i}\mathbf{X_i}^T K_h(U_i-U_j) & \sum_{i=1}^n \mathbf{X_i}(U_i-U_j)^2\mathbf{X}_i^T K_h(U_i-U_j)+\frac{1}{2}D_2\end{pmatrix}^{-1}$$
$$\times\begin{pmatrix}\sum_{i=1}^n \mathbf{X_i}Y_i K_h(U_i-U_j)\\ \sum_{i=1}^n (U_i-U_j)\mathbf{X_i}Y_i K_h(U_i-U_j)\end{pmatrix}$$

令 $W_{t,j}$=diag$\Big((U_1-U_j)^t K_h(U_1-U_j),\ldots,(U_n-U_j)^t K_h(U_n-U_j)\Big)$，则上面求解方程可以表示为

$$\begin{pmatrix}\mathbf{a}(U_j)\\ \mathbf{a}'(U_j)\end{pmatrix}=\begin{pmatrix}\mathbf{X}^T W_{0,j}\mathbf{X}+\frac{1}{2}D_1 & \mathbf{X}^T W_{1,j}\mathbf{X}\\ \mathbf{X}^T W_{1,j}\mathbf{X} & \mathbf{X}^T W_{2,j}\mathbf{X}+\frac{1}{2}D_2\end{pmatrix}^{-1}\begin{pmatrix}\mathbf{X}^T W_0\mathbf{Y}\\ \mathbf{X}^T W_1\mathbf{Y}\end{pmatrix}$$

从而迭代式为：在给定 $\mathbf{A}^{(m)}$ 和 $\mathbf{B}^{(m)}$，

$$\begin{pmatrix}\boldsymbol{a}^{(m+1)}(U_j)\\ \boldsymbol{b}^{(m+1)}(U_j)\end{pmatrix}=\begin{pmatrix}\boldsymbol{X}^T W_{0,j}\boldsymbol{X}+\frac{1}{2}D_1^{(m)} & \boldsymbol{X}^T W_{1,j}\boldsymbol{X}\\ \boldsymbol{X}^T W_{1,j}\boldsymbol{X} & \boldsymbol{X}^T W_{2,j}\boldsymbol{X}+\frac{1}{2}D_2^{(m)}\end{pmatrix}^{-1}\begin{pmatrix}\boldsymbol{X}^T W_{0,j}\boldsymbol{Y}\\ \boldsymbol{X}^T W_{1,j}\boldsymbol{Y}\end{pmatrix}\tag{8.5}$$

综上所述，迭代计算步骤：

步骤 1 给定初始值 $\widehat{\mathbf{A}}^{(0)}$ 和 $\widehat{\mathbf{B}}^{(0)}$。

步骤 2 令 $\widehat{\mathbf{A}}^{(m)}=\widehat{\mathbf{A}}^{(0)}$ 和 $\widehat{\mathbf{B}}^{(m)}=\widehat{\mathbf{B}}^{(0)}$，通过迭代式 (8.5) 解得 $\widehat{\mathbf{A}}^{(m+1)}$ 和 $\widehat{\mathbf{B}}^{(m+1)}$。

步骤 3 重复步骤 2，直到收敛。

步骤 1 的初始值 $\mathbf{A}^{(0)}$ 和 $\mathbf{B}^{(0)}$ 可以基于目标函数 (8.4) 右边的第一项，并利用加权最小二乘得到。在迭代过程中，如果 $|a_i(U_j)|<\epsilon$(或 $|b_k(U_j)|<\epsilon$), $j=1,\ldots,n$，设定 $\boldsymbol{a}_i(U)=0$ (或 $\boldsymbol{b}_k(U)=0$)，其中 $\epsilon>0$ 是非常小的正数。如果 $b_k(U)=0$，$\boldsymbol{a}_k(U)$ 是常系数函数，并且 $a_k=\frac{1}{n}\Sigma_{j=1}^n a_k(U_j)$。在实际计算中，本文设置 $\epsilon=10^{-2}$。除此之外，根据 Fan 和 Li (2001) 的建议 $\alpha=3.7$。核函数选择高斯函数 $K(t)=\exp(-t^2/2)/\sqrt{2\pi}$。

8.3.3 带宽和调整参数的选择

在目标函数 (8.4) 中，我们需要处理截断参数 $\boldsymbol{\lambda}_1$、$\boldsymbol{\lambda}_2$ 和带宽 h 的选择。因为 $\boldsymbol{\lambda}_1$ 控制着显著自变量的个数，$\boldsymbol{\lambda}_2$ 决定识别变系数函数和常系数函数的程度。h 决定着待估曲线的光滑程度，其越大越光滑。本文利用 BIC 准则 (Hu 和 Xia, 2012) 可以选择它们，其定义如下所示：

$$BIC_{\lambda_1,\lambda_2}=\log(RSS_{\lambda_1,\lambda_2})+df_{\lambda_1,\lambda_2}\frac{\log(nh)}{nh}+(p-df_{\lambda_1,\lambda_2})\frac{\log(n)}{n}\tag{8.6}$$

其中 $df_{\boldsymbol{\lambda}_1,\boldsymbol{\lambda}_2}$ 是非零变系数函数的个数，且

$$RSS_{\boldsymbol{\lambda}_1,\boldsymbol{\lambda}_2}=\frac{1}{n^2}\sum_{j=1}^n\sum_{i=1}^n\Big\{Y_i-\boldsymbol{X}_i^T\boldsymbol{a}(U_j)-(U_i-U_j)\boldsymbol{X}_i^T\boldsymbol{b}(U_j))\Big\}^2 K_h(U_i-U_j).$$

截断参数可以最小化下面式子得到：

$$(\boldsymbol{\lambda}_1,\boldsymbol{\lambda}_2)=\min_{\lambda_1,\lambda_2} BIC_{\lambda_1,\lambda_2}$$

由于 $\boldsymbol{\lambda}_1,\boldsymbol{\lambda}_2\in\boldsymbol{R}^p$，所以我们需要选择 $2p$ 个参数，这非常具有挑战性。与 Wang 和 Xia (2009)，Hu 和 Xia (2012) 类似，一个非常方便有效的方法是将 $2p$ 个参数转化为一个参数，即

$$\lambda_{1k}=\frac{\lambda}{n^{-1/2}||\boldsymbol{a}_k^{(0)}||}\quad\lambda_{2l}=\frac{\lambda}{n^{-1/2}||\boldsymbol{b}_l^{(0)}||}.$$

数值模拟可以印证其有效性。

h 的选择理论上可以通过 (8.6) 式得到最优解。然而，为了方便，本章采用交叉核实 (Cross Validation, CV) 的方法选择。交叉核实被广泛应用于核密度估计、核回归以及样条光滑等，其定义

为：

$$CV(h) = \sum_{i=1}^{n} \left(Y_i - \widehat{Y}_{(-i)} \right)^2$$

其中 $\widehat{Y}_{(-i)}$ 是去掉 Y_i 没有惩罚的估计。h 可以最小 $CV(h)$ 得到。需要说明的是 h 的选择理论上应该涉及到惩罚函数，但实际上为了计算方便没有涉及到惩罚函数，只是利用目标函数 (8.4) 中的第一项，即变系数函数模型的 h 的选择。在模拟实验中，本章利用 CV 选择 h，表现非常好。

8.4 理论性质

为了研究提出方法的大样本性质，我们假设 $\boldsymbol{a}_a(u) = (a_1(u), a_2(u), \ldots, a_{p_0}(u))^T$ 是变系数函数，$\boldsymbol{a}_b(u) = (a_{p_0+1}(u), a_{p_0+2}(u), \ldots, a_{p_1}(u))^T$ 是非零的常系数函数，$\boldsymbol{a}_c(u) = (a_{p_1+1}(u), a_{p_1+2}(u), \ldots, a_p(u))^T$ 是零常系数函数；相对应的自变量定义为：$\boldsymbol{X}_{ia} = (X_{i1}, X_{i2}, \ldots, X_{ip_0})^T \in \boldsymbol{R}^{p_0}$，$\boldsymbol{X}_{ib} = (X_{i,p_0+1}, X_{i,p_0+2}, \ldots, X_{i,p_1})^T \in \boldsymbol{R}^{p_1-p_0}$ 和 $\boldsymbol{X}_{ic} = (X_{i,p_1+1}, X_{i,p_1+2}, \ldots, X_{i,p})^T \in \boldsymbol{R}^{p-p_1}$。另外，令 $a_{1n} = \max\{\lambda_{1j}, 1 \le j \le p_1\}$，$a_{2n} = \min\{\lambda_{1j}, p_1 < j \le p\}$，$b_{1n} = \max\{\lambda_{2j}, 1 \le j \le p_0\}$ 以及 $b_{2n} = \min\{\lambda_{2j}, p_0 < j \le p\}$。

定理 8.1 (估计的稀疏性)

假设附录中的 *(C1)–(C6)* 成立，$nh^{-1/2}a_{1n} \to 0$，$nh^{-1/2}b_{1n} \to 0$，$nh^{-1/2}a_{2n} \to \infty$ 以及 $nh^{-1/2}b_{2n} \to \infty$，则

(1) 对于 $p_0 < j \le p$，$P(||\hat{\boldsymbol{b}}_{j,\lambda}|| = 0) \to 1$。

(2) 对于 $p_1 < k \le p$，$P(||\hat{\boldsymbol{a}}_{k,\lambda}|| = 0) \to 1$。

♡

如果识别了变系数函数和常系数函数，则模型 (8.3) 便会成为部分线性变系数模型。为了建立 Oracle 性质，本章定义 Oracle 估计量，对于任意的 u

$$\hat{\boldsymbol{a}}_{ora}(u) = \left(\frac{1}{n} \sum_{i=1}^{n} \boldsymbol{X}_{ia} \boldsymbol{X}_{ia}^T K_h(U_i - u) \right)^{-1} \left(\frac{1}{n} \sum_{i=1}^{n} \boldsymbol{X}_{ia} (Y_i - \boldsymbol{X}_{ib}^T \boldsymbol{a}_b(u) - (U_i - u) \boldsymbol{X}_{ia}^T \boldsymbol{a}_a^T(u)) \right)$$

定理 8.2 (Oracle 性质)

假设附录中的 *(C1)–(C6)* 成立，如果 $nh^{-1/2}a_{1n} \to 0$，$nh^{-1/2}b_{1n} \to 0$，$nh^{-1/2}a_{2n} \to \infty$ 以及 $nh^{-1/2}b_{2n} \to \infty$，那么

$$\sup_u ||\hat{\boldsymbol{a}}_{a,\lambda}(u) - \hat{\boldsymbol{a}}_{ora}(u)|| = o_p(n^{-2/5}).$$

♡

8.5 模拟例子和实例分析

8.5.1 模拟例子

这部分进行蒙特卡洛模拟。令样本量 n 为 300，重复进行 500 次模拟。本文考虑如下部分线性变系数模型：

例题 8.1

$$\begin{aligned} Y_i = {} & 2\sin(2\pi U_i) X_{i1} + 4U_i(1 - U_i) X_{i2} + 3X_{i3} + 1.5X_{i4} + 2X_{i5} \\ & + 0X_{i6} + 0X_{i7} + 0X_{i8} + 0X_{i9} + 0X_{i10} + 0X_{i11} + 0.5e_i \end{aligned}$$

例题 8.2

$$Y_i = 2\sin(2\pi U_i)X_{i1} + 8U_i(1-U_i)X_{i2} + \cos^2(2\pi U_i)X_{i3} + 3X_{i4}$$
$$- 1.5X_{i5} + 4X_{i6} + 0X_{i7} + 0X_{i8} + 0X_{i9} + 0X_{i10} + 0X_{i11} + 0.5e_i$$

其中 $X_{i1} = 1$，$(X_{i2}, \ldots, X_{i,11})^T$ 来自于多元正态分布，且 $\mathrm{cov}(X_{ij_1}, X_{ij_2}) = 0.5^{|j_1 - j_2|}$，$2 \le j_1, j_2 \le 11$。$e_i \overset{\text{iid}}{\sim} N(0,1)$，$U$ 是指示变量，其来自于 $(0,1)$ 区间的均匀分布，即 $U_i \overset{\text{iid}}{\sim} U(0,1)$。

本节比较了提出方法 (简记 PPM) 和 Tang 等 (2012) 基展开方法 (简记 BSE)。为了比较估计结果，我们考虑如下指标：

(1) SV 表示变系数函数个数；
(2) SC 表示常系数函数个数；
(3) SZ 表示零系数个数；
(4) V 表示变系数函数被判为常系数函数的个数；
(5) F 表示常系数函数被判为变系数函数的个数；
(6) C 表示零系数被判为零系数的个数；
(7) I 表示非零系数被判为零系数的个数；
(8) Z 表示零系数被判为非零系数的个数。

其中 SV、SC 和 SZ 评价 $\alpha_1, \alpha_2, \ldots, \alpha_{11}$ 的效应，V、F、C、I 和 Z 评价整个效应。

表8.1—8.4总结了例8.1和8.2的模拟结果。我们可以得到：(1) 除了表8.1中的 α_2，PPM 的 SV、SC 和 SZ 均优于 BSE。(2) 在识别变系数函数方面，这两种方法表现的很相似，BSE 更好一些。(3) 相比 BSE，PPM 在识别常系数函数和非零常系数函数以及变量选择上更有效，因为它的 SV、SC 和 SZ 接近于 Oracle，并且 F 和 Z 非常小。如表8.1中 BSE 的 α_3 为 215，其误判率为 43%，而 PPM 的仅为 7，误判率仅为 1.4%。(4)BSE 更倾向于将常系数函数判别为变系数函数。如表8.2所示，BSE 的 F 为 2.938，而 PPM 的为 0.028。(5)BSE 更倾向于将零系数判为非零。如表8.3 所示，BSE 的 Z 为 1.9，而 PPM 仅为 0.324。(6) 对 PPM 和 BSE，真实为非零的别判为零的概率都很小。综合来说，PPM 优于 BSE，这也印证了提出方法有效性和准确性。

表 8.1: 例8.1模拟 SV、SC 和 SZ 结果

	BSE			PPM			Oracle		
	SV	SC	SZ	SV	SC	SZ	SV	SC	SZ
α_1	500	0	0	500	0	0	500	0	0
α_2	478	22	0	281	219	0	500	0	0
α_3	215	285	0	7	493	0	0	500	0
α_4	170	330	0	0	500	0	0	500	0
α_5	176	324	0	3	497	0	0	500	0
α_6	159	341	334	0	500	442	0	500	500
α_7	157	343	335	0	500	438	0	500	500
α_8	148	352	343	1	499	439	0	500	500
α_9	157	343	332	0	500	433	0	500	500
α_{10}	157	343	335	2	498	439	0	500	500
α_{11}	130	370	357	1	499	444	0	500	500

表 8.2: 例8.1模拟 V、F、C 和 I 的结果

方法	V	F	C	I	Z
BSE	0.044	2.938	4.072	0	1.928
PPM	0.438	0.028	5.270	0	0.730
Oracle	0	0	6	0	0

表 8.3: 例8.2模拟 SV、SC 和 SZ 结果

	BSE			PPM			Oracle		
	SV	SC	SZ	SV	SC	SZ	SV	SC	SZ
α_1	500	0	0	500	0	0	500	0	0
α_2	500	0	0	500	0	0	500	0	0
α_3	500	0	0	500	0	0	500	0	0
α_4	220	280	0	3	497	0	0	500	0
α_5	211	289	0	0	500	0	0	500	0
α_6	207	293	0	0	500	0	0	500	0
α_7	180	320	307	0	500	476	0	500	500
α_8	163	337	323	0	500	468	0	500	500
α_9	180	320	317	1	499	467	0	500	500
α_{10}	206	294	288	0	500	461	0	500	500
α_{11}	182	318	315	0	500	466	0	500	500

表 8.4: 例8.2模拟 V、F、C 和 I 的结果

方法	V	F	C	I	Z
BSE	0	3.098	3.100	0	1.900
PPM	0	0.008	4.676	0	0.324
Oracle	0	0	5	0	0

8.6 本章小结

本章讨论了变系数模型的统一变量选择，提出一种新的方法解决了变系数函数和常系数函数的识别，以及变系数函数和常系数函数的变量选择。在一定的条件性，本章证明了提出方法的稀疏性质和 Oracle 性质。模型实验印证了提出方法的有效性和准确性。该方法可以推广到分位数变系数模型的统一变量选择（详见第六章）。

8.7 附录

为了研究提出方法的渐近性质，本文令 $H = (A, B) = (\beta(U_1), \ldots, \beta(U_n))^T$，并且假设下面条件成立 (Fan 和 Huang 2005).

(C1) 对于 $s > 2$，$E|Y_i|^{2s} < \infty$ 和 $E||\boldsymbol{X}_i||^{2s} < \infty$；

(C2) U_i 的密度函数 $f(u)$ 连续，并且在 $[0,1]$ 有界，其取值不等于 0；

(C3) 矩阵 $\Omega(u) = E(\boldsymbol{X}_i \boldsymbol{X}_i^T | U_i = u)$ 是非奇异矩阵，并且在 [0,1] 存在二阶导数。另外，函数 $E(||\boldsymbol{X}_i||^4 | U_i = u)$ 也有界；

(C4) $f(u)$ 二阶导数有界，并且 $\sigma^2(u) = E(\varepsilon_i^2 | U_i = u)$ 有界；

(C5) $K(u)$ 在紧支持 (Compact Support) 是对称的密度函数；

(C6) $a_j(u), j = 1, 2, \ldots, p$ 存在连续的二阶导。

其中 (C2) 保证两个指示变量之间的最大距离阶数是 $O_p(\log(n)/n)$。对于任意指示变量的取值 $u \in [0,1]$，令 u^* 是距离观测值领域最近的值，即 $u^* = \text{argmin}_{\bar{u} \in \{U_t : 1 \le t \le n\}} |u - \bar{u}|$。结合光滑假设 (C6)，我们可以得到 $||\beta(u) - \beta(u^*)|| = O_p(\log(n)/n)$，这个收敛速度小于非参数的最优收敛速度 $(n^{-2/5})$。实际上，这意味着指示变量在其支撑上的取值很多，以至于可以用 $\{\beta(U_t) : 1 \le t \le n\}$ 逼近函数 $\beta(u)$。

引理 8.1

设 $(\xi_i, U_i), i = 1, \ldots, n$ 是独立同分布随机变量，其中 ξ_i 是刻度随机变量 (Scalar Random Variables)。假设 $E|\xi_i|^s < \infty$，$\sup_u \int |y|^s f(u, v) dv < \infty$，其中 f 表示 (ξ_i, U_i) 的联合分布函数。令 K 是一个在有界支撑上的有界正函数，且满足 Lipschitz 条件，假设存在 $\delta < 1 - s^{-1}$ 使得 $n^{2\delta - 1} h \to \infty$，则

$$\sup_{u \in [0,1]} |\frac{1}{n} \sum_{i=1}^{n} [K_h(U_i - u)\xi_i - E\{K_h(U_i - u)\xi_i\}]| = O_p\Big(\frac{log(1/h)}{nh}\Big)^{1/2}$$

♡

这个引理的证明详见 Mack 和 Silverman(1982) 或 Fan 和 Zhang(2000)。

引理 8.2

如果 (C1)–(C6) 成立，且 $nh^{-1/2} a_{1n} \to 0, nh^{-1/2} b_{1n} \to 0$，则

$$\frac{1}{n} \sum_{t=1}^{n} ||\hat{\beta}(U_t) - \beta(U_t)||^2 = O\{(nh)^{-1/2}\}$$

♡

证明. 对于任意矩阵 $G = (g_{ij})$，$||G||^2 = \sum g_{ij}^2$。我们用 $S = (s_{ij}) \in R^{n \times 2p}$ 表示任意 $n \times 2p$ 维矩阵，且行用 $\boldsymbol{s}_1^T, \ldots, \boldsymbol{s}_n^T$ 表示，列用 $\boldsymbol{v}_1^T, \ldots, \boldsymbol{v}_{2p}^T$ 表示。令 $H_0 = (\beta_0(U_1), \ldots, \beta_0(U_n))^T$，且其列用 $\boldsymbol{h}_{01}^T, \ldots, \boldsymbol{h}_{0,2p}^T$ 表示。根据 Fan 和 Li(2001)，我们只需证明对于任意 $\varepsilon > 0$，存在常数 $C > 0$ 使得

$$\liminf_n P\Big\{\Big(\inf_{n^{-1}||s||^2 = C} Q_\lambda(H_0 + (nh)^{-1/2} S) > Q_\lambda(H_0)\Big)\Big\} = 1 - \varepsilon \tag{8.7}$$

成立即可。

根据 $Q_\lambda(H)$ 定义，我们可以得到：

$$
\begin{aligned}
& hn^{-1}\left\{Q_\lambda(H_0+(nh)^{-1/2}S) > Q_\lambda(H_0)\right\} \\
&= hn^{-1}\sum_{t=1}^{n}\sum_{i=1}^{n}\left(Y_i - D_{u_t,i}^T\{\beta_0(U_t)+(nh)^{-1/2}\boldsymbol{s}_t\}\right)^2 K_h(U_i-U_t) \\
&- hn^{-1}\sum_{t=1}^{n}\sum_{i=1}^{n}\left(Y_i - D_{u_t,i}^T\beta_0(U_t)\right)^2 K_h(U_i-U_t) \\
&+ \frac{h}{n}\sum_{j=1}^{p} p_{\lambda_{1j}}(||\boldsymbol{h}_{0j}+(nh)^{-1/2}\boldsymbol{v}_j|| - ||\boldsymbol{h}_{0j}||) \\
&+ \frac{h}{n}\sum_{j=p+1}^{2p} p_{\lambda_{2,j-p}}(||\boldsymbol{h}_{0j}+(nh)^{-1/2}\boldsymbol{v}_j|| - ||\boldsymbol{h}_{0j}||) \doteq R_1
\end{aligned}
$$

其中 $D_{u_t,i} = (\boldsymbol{X}_i, (U_i-U_t)\boldsymbol{X}_i)^T$。经过简单的代数计算和 $||\boldsymbol{h}_{0j}|| = 0$ $(p_1 < j \le p,\ p+p_0 < j \le 2p)$ 的事实，有：

$$
\begin{aligned}
R_1 =& \frac{1}{n}\sum_{t=1}^{n}\left(\boldsymbol{s}_t^T\hat{\Sigma}(U_t)\boldsymbol{s}_t - 2s^T\hat{\boldsymbol{e}}_t\right) \\
&+ \frac{h}{n}\sum_{j=1}^{p_1} p_{\lambda_{1j}}(||\boldsymbol{h}_{0j}+(nh)^{-1/2}\boldsymbol{v}_j|| - ||\boldsymbol{h}_{0j}||) \\
&+ \frac{h}{n}\sum_{j=p_1+1}^{p} p_{\lambda_{1,j}}(||\boldsymbol{h}_{0j}+(nh)^{-1/2}\boldsymbol{v}_j|| - ||\boldsymbol{h}_{0j}||) \\
&+ \frac{h}{n}\sum_{j=p+1}^{p+p_0} p_{\lambda_{2,j-p}}(||\boldsymbol{h}_{0j}+(nh)^{-1/2}\boldsymbol{v}_j|| - ||\boldsymbol{h}_{0j}||) \\
&+ \frac{h}{n}\sum_{j=p+p_0+1}^{2p} p_{\lambda_{2,j-p}}(||\boldsymbol{h}_{0j}+(nh)^{-1/2}\boldsymbol{v}_j|| - ||\boldsymbol{h}_{0j}||) \\
\ge& \frac{1}{n}\sum_{t=1}^{n}\left(\boldsymbol{s}_t^T\hat{\Sigma}(U_t)\boldsymbol{s}_t - 2\boldsymbol{s}_t^T\hat{\boldsymbol{e}}_t\right) \\
&+ \frac{h}{n}\sum_{j=1}^{p_1} p_{\lambda_{1j}}(||\boldsymbol{h}_{0j}+(nh)^{-1/2}\boldsymbol{v}_j|| - ||\boldsymbol{h}_{0j}||) \\
&+ \frac{h}{n}\sum_{j=p+1}^{p+p_0} p_{\lambda_{2,j-p}}(||\boldsymbol{h}_{0j}+(nh)^{-1/2}\boldsymbol{v}_j|| - ||\boldsymbol{h}_{0j}||) \doteq R_2
\end{aligned}
$$

其中 $\hat{\Sigma}(U_t) = n^{-1}\sum_{i=1}^{n} D_{u_t,i}D_{u_t,i}^T K_h(U_i-U_t)$, $\hat{e}_t = n^{-1/2}h^{1/2}\sum_{i=1}^{n} D_{u_t,i}\left(D_{u_t,i}^T[\beta(U_t)-\beta(U_i)]+\varepsilon_i\right)K_h(U_t-U_i)$。令 $\hat{\lambda}_t^{\min}$ 是 $\hat{\Sigma}(U_t)$, $\hat{\lambda}_{\min} = \min\{\hat{\lambda}_t^{\min}, t = 1,\ldots,n\}$ 的最小特征值，以及 $\hat{\boldsymbol{e}} = (\hat{\boldsymbol{e}}_1,\ldots,\hat{\boldsymbol{e}}_n)^T \in \boldsymbol{R}^{n\times 2p}$，

可以得到:

$$
\begin{aligned}
R_2 \geq & \frac{1}{n}\sum_{t=1}^{n}\left\{||\boldsymbol{s}_t||^2\hat{\lambda}_t^{\min} - 2||\boldsymbol{s}_t||\cdot||\hat{\boldsymbol{e}}_t||\right\} \\
& - n^{-3/2}h^{1/2}\sum_{j=1}^{p_1} p_{\lambda_{1j}}(||\boldsymbol{v}_j||) - n^{-3/2}h^{1/2}\sum_{j=p+1}^{p+p_0} p_{\lambda_{2,j-p}}(||\boldsymbol{v}_j||) \\
\geq & \hat{\lambda}_{\min}\left\{n^{-1}\sum_{t=1}^{n}||\boldsymbol{s}_t||^2\right\} - 2n^{-1}\left\{\sum_{t=1}^{n}||\boldsymbol{s}_t||\cdot||\hat{\boldsymbol{e}}_t||\right\} \\
& - n^{-3/2}h^{1/2}\sum_{j=1}^{p_1} p_{\lambda_{1j}}(||\boldsymbol{v}_j||) - n^{-3/2}h^{1/2}\sum_{j=p+1}^{p+p_0} p_{\lambda_{2,j-p}}(||\boldsymbol{v}_j||) \\
\geq & \hat{\lambda}_{\min}\left\{n^{-1}||S||^2\right\} - 2(n^{-1}||S||^2)^{1/2}\cdot(n^{-1}||\hat{e}||^2)^{1/2} \\
& - n^{-3/2}h^{1/2}\sum_{j=1}^{p_1} p_{\lambda_{1j}}(||\boldsymbol{v}_j||) - n^{-3/2}h^{1/2}\sum_{j=p+1}^{p+p_0} p_{\lambda_{2,j-p}}(||\boldsymbol{v}_j||) \doteq R_3
\end{aligned}
$$

根据条件 $n^{-1}||S||^2 = C^2$，有

$$\begin{aligned}
R_3 =& \hat{\lambda}_{min} \times C^2 - 2C \times (n^{-1}||\hat{\boldsymbol{e}}||^2)^{1/2} \\
& - n^{-3/2}h^{1/2} \sum_{j=1}^{p_1} p_{\lambda_{1j}}(||\mathbf{v}_j||) - n^{-3/2}h^{1/2} \sum_{j=p+1}^{p+p_0} p_{\lambda_{2,j-p}}(||\boldsymbol{v}_j||) \\
\geq& \hat{\lambda}_{min} \times C^2 - 2C \times (n^{-1}||\hat{\boldsymbol{e}}||^2)^{1/2} \\
& - n^{-1}h^{3/2}a_{1n}\Big(n^{-1}\sum_{j=1}^{p_1}||\boldsymbol{v}_j||^2\Big)^{1/2} - n^{-1}h^{1/2}b_{1n}\Big(n^{-1}\sum_{j=p+1}^{p+p_0}||\boldsymbol{v}_j||^2\Big)^{1/2} \\
\geq& \hat{\lambda}_{min} \times C^2 - 2C \times (n^{-1}||\hat{\boldsymbol{e}}||^2)^{1/2} - n^{-1}h^{1/2}(a_{1n} + b_{1n})\Big(n^{-1}\sum_{j=1}^{2p}||\boldsymbol{v}_j||\Big)^{1/2} \\
=& \hat{\lambda}_{min} \times C^2 - 2C \times (n^{-1}||\hat{\boldsymbol{e}}||^2)^{1/2} - n^{-1}h^{1/2}(a_{1n} + b_{1n})C
\end{aligned} \tag{8.8}$$

经过计算，我们可以得到 $n^{-1}||\hat{\boldsymbol{e}}|| = O_p(1)$。根据引理8.1和条件 (C3)，我们有 $P(\lambda_{\min} \to \lambda_0^{\min}) \to 1$，其中 $\lambda_0^{\min} = \inf_{u\in[0,1]} \lambda_{\min}(f(u)\Omega(u))$，$\lambda_{\min}(A)$ 任意正定矩阵 A 的最小特征值。根据条件 (C2)、(C3) 和引理8.1，我们有 $\lambda_0^{\min} > 0$。所以，(8.8) 式由前面两项决定，因为 $nh^{-1/2}(a_{1n} + b_{1n}) \to 0$。由于 (8.8) 式的第一项是关于 C 的二次函数，第二项是 C 的线性函数。当 C 足够大时，可以保证 (8.8) 以概率 1 的可能性大于零。从而证明了 (8.7) 式，进而证明了引理8.2。

证明 [定理 8.1 的证明] 我们先证明定理的第一部分。证明 (1) 只需要证明 $P(||\hat{\boldsymbol{b}}_{\lambda,j}|| = 0) \to 1$，其中 $j = p$。对于 $p_0 < j < p$ 的证明类似。本文利用反证法证明。如果当 $j = p$，$P(||\hat{\boldsymbol{b}}_{\lambda,j}|| = 0) \to 1$ 不成立，那么 $\hat{\boldsymbol{b}}_{\lambda,j}$ 是下面等式的解

$$0 = \frac{\partial Q(H)}{\partial \boldsymbol{b}_{\lambda,p}}|_{H=\hat{H}_\lambda} = \alpha_1 + \alpha_2 \tag{8.9}$$

其中 α_1 是 n 维的向量，且第 t 分量是

$$\alpha_{1t} = -2\sum_{i=1}^{n}(U_i - U_t)X_{ip}(Y_i - D_{u_t,i}^T\hat{v}(U_t))K_h(U_i - U_t), t = 1, 2, \ldots, n$$

且

$$\alpha_2 = \frac{p'_{2p}||\boldsymbol{b}_p^{(m)}||}{||\boldsymbol{b}_p^{(m)}||}\boldsymbol{b}_{\lambda,j}.$$

根据核光滑以及引理8.1和8.2，我们可以得到 $||\alpha_1|| = O_p(nh^{-1/2})$。另一方面，在理论条件 $nh^{-1/2}||\alpha_2|| \geq nh^{-1/2}b_{2n} \to \infty$，这意味着 $P(||\alpha_1|| < ||\alpha_2||) \to 1$。从而 (2.7) 不成立。这说明 $\hat{\boldsymbol{b}}_{\lambda,j}$ 不会落在 Q(H) 的可导区间内。而 $Q_\lambda(H)$ 只有在原点不可导，从而 $P(||\hat{\boldsymbol{b}}_{\lambda,j}|| = 0) \to 1$。

同理，我们可以证明定理8.1的第二部分，从而这定理证明完毕。

证明 [定理 8.2 的证明] 根据定理8.1，我们可以得到 $\hat{\boldsymbol{a}}_{\lambda.j} = 0$，$p_1 < j \leq p$，以及 $\hat{\boldsymbol{b}}_{\lambda.j} = 0$，$p_0 < j \leq p$ 以概率 1 成立，则 $a_{a,\lambda}(u)$ 是下面正则方程的解，即

$$-\frac{1}{n}\sum_{i=1}^{n}\boldsymbol{X}_{ia}\Big\{Y_i - \boldsymbol{X}_{ia}^T\boldsymbol{a}_{a,\lambda}(u) - (U_i - u)\boldsymbol{X}_{ia}^T\boldsymbol{a}'_{a,\lambda}(u) - \boldsymbol{X}_{ib}^T a_{b,\lambda}(u)\Big\}K_h(U_i - u) + \frac{1}{n}L = 0$$

其中 $L = \Big(p'_{\lambda_{11}}(||\hat{\boldsymbol{a}}_{1,\lambda}||)\frac{\hat{\boldsymbol{a}}_1(u)}{||\hat{\boldsymbol{a}}_{1,\lambda}||}, \ldots, p'_{\lambda_{1p_0}}(||\hat{\boldsymbol{a}}_{p_0,\lambda}||)\frac{\hat{\boldsymbol{a}}_{p_0}(u)}{||\hat{\boldsymbol{a}}_{p_0,\lambda}||}\Big)^T$。这意味着 $\hat{\boldsymbol{a}}_{a,\lambda}$ 是下面形式：

$$\hat{\boldsymbol{a}}_{a,\lambda}(u) = \{\Sigma_1(u)\}^{-1}\Big\{\frac{1}{n}\sum_{i=1}^{n}\boldsymbol{X}_{ia}\{Y_i - (U_i - u)\boldsymbol{X}_{ia}^T\hat{\boldsymbol{a}}'_{a,\lambda}(u) - \boldsymbol{X}_{ib}^T\hat{\boldsymbol{a}}_{b,\lambda}(u)\} + \frac{1}{n}L\Big\}$$

其中 $\Sigma_1(u) = n^{-1}\sum_{i=1}^{n} \boldsymbol{X}_{ia}\boldsymbol{X}_{ia}^T K_h(U_i - u)$。**与 Oracle 估计量相比，我们得到**

$$\begin{aligned}
&\max_{u\in[0,1]} ||\hat{\boldsymbol{a}}_{a,\lambda} - \hat{\boldsymbol{a}}_{ora}|| \\
&\quad = ||\{\Sigma_1(u)\}^{-1}\{\frac{1}{n}L + \Sigma_2(u)(\hat{\boldsymbol{a}}'_{a,\lambda}(u) - \boldsymbol{a}'_a(u)) + \Sigma_3(u)(\hat{\boldsymbol{a}}_{b,\lambda}(u) - \boldsymbol{a}_b(u))\}|| \\
&\quad \le \lambda_{1,\min}^{-1}||\frac{1}{n}L|| + \lambda_{1,min}^{-1}\lambda_{2,\max}||\hat{\boldsymbol{a}}'_{a,\lambda}(u) - \boldsymbol{a}'_a(u)|| \\
&\qquad + \lambda_{1,\min}^{-1}\lambda_{3,\max}||\hat{\boldsymbol{a}}_{b,\lambda}(u) - \boldsymbol{a}_a(u)|| \\
&\quad \doteq J_1 + J_2 + J_3
\end{aligned}$$

其中 $\Sigma_2(u) = n^{-1}\sum_{i=1}^{n} \boldsymbol{X}_{ia}\boldsymbol{X}_{ia}^T(u_i - u)K_h(u_i - u)$，$\Sigma_3(u) = n^{-1}\sum_{i=1}^{n} \boldsymbol{X}_{ib}\boldsymbol{X}_{ib}^T K_h(u_i - u)$，$\lambda_{1,\min} = \min\{\lambda_{\min}(\Sigma_1(u)), u \in [0,1]\}$，$\lambda_{2,\max} = \max\{\lambda_{\max}(\Sigma_2(u)), u \in [0,1]\}$，$\lambda_{3,\max} = \max\{\lambda_{\max}(\Sigma_3(u)), u \in [0,1]\}$。**对于** J_1，**应用引理8.1，我们可以得到** $J_1 \le C\sqrt{p_0}a_{1n} = o_p(n^{-2/5})$。**根据引理8.2，我们有** $J_2 = o_p(n^{-2/5})$ **和** $J_3 = o_p(n^{-2/5})$。**从而证明了定理8.2**。

8.8 参考文献

1. Cai, Z., Fan, J., & Yao, Q. (2000). Functional-coefficient regression models for nonlinear time series. Journal of the American Statistical Association, 95(451), 941-956.
2. Fan, J., & Zhang, W. (1999). Statistical estimation in varying coefficient models. Annals of Statistics, 1491-1518.
3. H*ä*rdle, W. K., M*ü*ller, M., Sperlich, S., & Werwatz, A. (2012). Nonparametric and semiparametric models. Springer Science & Business Media.
4. Shumway R. Applied statistical time series analysis. 1988, Prentice-Hall.
5. Soetaert K. 等 (2014). limSolve. R package version 1.5.5.1, http://CRAN.R-project.org/package=limSolve.

第 9 章 可加模型

9.1 引言

可加模型的表达式是：

$$E(Y|X) = c + \sum_{j=1}^{p} f_j(X_j) \tag{9.1}$$

为了可以识别，通常假设

$$E_{X_j}\Big[f_j(X_j)\Big] = 0 \tag{9.2}$$

这个假设使得 $c = E(Y)$。

从表示式，X_j 通过函数 f_j 对 Y 的影响，这样便于解释。R `gam` 包中的 `gam` 可以实现。下面我们介绍 B 样条的估计。

9.2 B 估计

假设内点的为 $\zeta_1, \ldots, \zeta_K$，对于 x，B 样条基函数可以表示为：

$$1, x, x^2, x^3, (x-\zeta_1)_+^3, \ldots, (x-\zeta_K)_+^3 \tag{9.3}$$

其中 $a_+ = \max(a, 0)$。基函数包含截距项，所以基的个数为 $K+3+1$。B 样条展开可以由 `splines` 包中 `bs` 可以实现。

对于 $f(X)$，B 样条逼近的表达式为：

$$f(X) = \beta_1 B_1(X) + \beta_2 B_2(X) + \cdots + \beta_{K-1} B_{K-1}(X) + \beta_K B_K(X) \tag{9.4}$$

其中 $B_1(\bullet), \ldots, B_K(\bullet)$ 是带有常数项的 B 样条基函数。

这种表达没有考虑假设条件 (9.2)。根据约束条件[1]，

$$\beta_K = -\frac{\beta_1\mu_1 + \beta_2\mu_2 + \cdots + \beta_{K-1}\mu_{K-1}}{\mu_K} \tag{9.5}$$

其中 $\mu_k = E_X\Big[B_k(X)\Big], k = 1, \ldots, K$。

将 (9.5) 代入 (9.4)，

$$f_j(X_j) = \beta_1\Big[B_1(X) - \frac{\mu_1}{\mu_K}B_K(X)\Big] + \beta_2\Big[B_2(X) - \frac{\mu_2}{\mu_K}B_K(X)\Big] + \cdots + \beta_{K-1}\Big[B_{K-1}(X) - \frac{\mu_{K-1}}{\mu_K}B_K(X)\Big] \tag{9.6}$$

限制条件 (9.2) 暗示了基函数的自由度 (degree) 是 $K-1$。

给定样本 $x_1, \ldots, x_n$, (9.7) 是

$$f(x_i) \approx \beta_1\Big[B_1(x_i) - \frac{\bar{b}_1}{\bar{b}_K}B_K(x_i)\Big] + \beta_2\Big[B_2(x_i) - \frac{\bar{b}_2}{\bar{b}_K}B_K(x_i)\Big] + \cdots + \beta_{K-1}\Big[B_{K-1}(x_i) - \frac{\bar{b}_{K-1}}{\bar{b}_K}B_K(x_j)\Big] \tag{9.7}$$

其中 $\bar{b}_k = \sum_{i=1}^{n} B_K(x_i)/n$。假设 $Z_k(x_i) = B_k(x_i) - \frac{\bar{b}_k}{\bar{b}_K}B_K(x_i)$，则

$$f(x_i) \approx \beta_1 Z_1(x_i) + \beta_2 Z_2(x_i) + \cdots + \beta_{K-1} Z_{K-1}(x_i) \tag{9.8}$$

[1]谢谢北京工业大学杜江老师的讨论和建议

矩阵表示为

$$\begin{bmatrix} z_1(x_1) & z_2(x_1) & \dots & z_{K-1}(x_1) \\ z_1(x_2) & z_2(x_2) & \dots & z_{K-1}(x_1) \\ \dots & \dots & \dots & \dots \\ z_1(x_n) & z_2(x_n) & \dots & z_{K-1}(x_1) \end{bmatrix}_{n\times(K-1)}$$

$$= \begin{bmatrix} B_1(x_1) & B_2(x_1) & \dots & B_K(x_1) \\ B_1(x_2) & B_2(x_2) & \dots & B_K(x_1) \\ \dots & \dots & \dots & \dots \\ B_1(x_n) & B_2(x_n) & \dots & B_K(x_1) \end{bmatrix}_{n\times K} \times \begin{bmatrix} 1 & 0 & \dots & 0 \\ \dots & \dots & \dots & \dots \\ 0 & 0 & \dots & 1 \\ -\frac{\bar{b}_1}{\bar{b}_K} & -\frac{\bar{b}_2}{\bar{b}_K} & \dots & -\frac{\bar{b}_{K-1}}{\bar{b}_K} \end{bmatrix}_{K\times(K-1)}$$

由此，可加模型转化为线性模型了。

下面我们进行模型比较 R 包实现的方法，如[2]:

$$Y = \sum_j^4 f_j(X_j) + \varepsilon$$

其中

$$f_1(X_1 = -\sin(2X_1) \qquad f_2(X_2) = X_2^2 - E(X_2^2)$$
$$f_3(X_3) = X_3 \qquad f_4 = \exp(-X_4) - E[\exp(-X_4)]$$

其中 X_i 独立同分且来自于 $[-2.5, 2.5]$。ε 来自于 $n(0,1)$。我们设置 $n = 300$，使用 MSE 比较两个方法

$$MSE_j = \frac{1}{n}\sum_{i=1}^{n}(\hat{f}_j - f_j)^2$$

结果如图 9.1 和表 9.1 所示。`gam` 和样条估计几乎一样。

表 9.1: 可加模型模拟的结果

	bs	R
f_1	0.02	0.02
f_2	0.01	0.03
f_3	0.06	0.01
f_4	0.06	0.06

[2]Hardle, W. K., Maler, M., Sperlich, S., & Werwatz, A. (2012). Nonparametric and semiparametric models: an Introduction. Springer Science & Business Media.

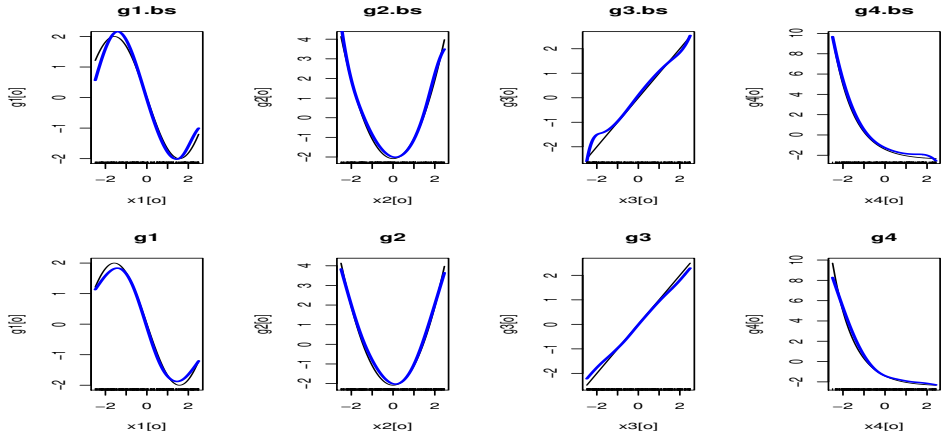

图 9.1: 两种方法的比较

模拟代码

```
graphics.off()
rm(list=ls(all=TRUE))
#setwd("C:/...")         #set your working directory
#install.packages("gam")
library(gam)

n <- 300 ####75
p <- 4
a <- 2.5
x <- matrix(runif(n*p, min=-a, max=a),n,4)

x1 <- x[,1]
x2 <- x[,2]
x3 <- x[,3]
x4 <- x[,4]

g1 <- -2*sin(x1)
g2 <- x2 ^ 2 - (2 * a) ^ 2 / 12
g3 <- x3
g4 <- exp(-x4) - (exp(a)-exp(-a))/(2 * a)
y  <- g1 + g2 + g3 + g4 + rnorm(n)

print( c(mean(g1), mean(g2), mean(g3), mean(g4)))


#####################################Bspline begin
kn <- floor(n^(1/5))
degree <- 3
Kn  <- kn + degree + 1
```

```
B.basis <- matrix(0, n, p*Kn)
Z.basis <- matrix(0, n, p*(Kn-1))
BZ <- matrix(0, Kn, p*(Kn-1))
kk.B <- 0
kk.Z = kk.B
for(j in 1:p){
  Knots.xj <- quantile(x[, j], probs=ppoints(kn))
  index.B <- (kk.B+1) : (kk.B+Kn)

  B.basis[, index.B]<- bs(x[,j], knots = Knots.xj, degree = 3, intercept =TRUE)

  ###BZ
  B.mean <- apply(  B.basis[,index.B], 2, mean)
  index.Z <- (1 + kk.Z) :(Kn-1+ kk.Z)
  BZ[, index.Z] <- rbind(diag(Kn-1), - B.mean[-Kn]/B.mean[Kn])
  ##Z
  Z.basis[,index.Z]<-  B.basis[,index.B] %*%  BZ[, index.Z]
  kk.B <- kk.B + Kn
  kk.Z <- kk.Z + Kn - 1
}

#####################################Bspline end



fit<- lm(y~Z.basis-1)

############################compute f begin
beta.full <- fit$coefficients
f.estimator <- matrix(0, n, p)
colnames(f.estimator) <- paste("f", 1:p)
kk.Z <- 0
for(j in 1:p){
  index.Z <- (1 + kk.Z) :(Kn-1+ kk.Z)
  f.estimator[,j] <- Z.basis[, index.Z] %*% beta.full[index.Z]
  kk.Z <- kk.Z + Kn-1
}

############################compute f end


par(mfrow=c(2,4))

```

```
o <- order(x1); plot(x1[o],g1[o],type="l",main="g1.bs")
lines(x1[o], f.estimator[o,1],col="blue",lwd=2)
rug(x1)



o <- order(x2); plot(x2[o],g2[o],type="l",main="g2.bs")
lines(x2[o],f.estimator[o,2],col="blue",lwd=2)
rug(x2)

o <- order(x3); plot(x3[o],g3[o],type="l",main="g3.bs")
lines(x3[o],f.estimator[o,3],col="blue",lwd=2)
rug(x3)

o <- order(x4); plot(x4[o],g4[o],type="l",main="g4.bs")
lines(x4[o],f.estimator[o,4],col="blue",lwd=2)
rug(x4)




#####################################gam
am <- gam( y ~ s(x1) + s(x2) + s(x3) + s(x4) , family=gaussian)
#print(summary(am))

b <- coef(am)
#plot(am)

#par(mfrow=c(2,2))

o <- order(x1); plot(x1[o],g1[o],type="l",main="g1")
lines(x1[o],x1[o]*b[2]+am$smooth[o,1],col="blue",lwd=2)
rug(x1)

o <- order(x2); plot(x2[o],g2[o],type="l",main="g2")
lines(x2[o],x2[o]*b[3]+am$smooth[o,2],col="blue",lwd=2)
rug(x2)

o <- order(x3); plot(x3[o],g3[o],type="l",main="g3")
lines(x3[o],x3[o]*b[4]+am$smooth[o,3],col="blue",lwd=2)
rug(x3)

o <- order(x4); plot(x4[o],g4[o],type="l",main="g4")
```

```
116 lines(x4[o],x4[o]*b[5]+am$smooth[o,4],col="blue",lwd=2)
117 rug(x4)
118
119 #par(mfrow=c(1,1))
120 ###################################MSE
121 f.mse <- matrix(0, 4,2)
122 colnames(f.mse) <- c("bs","R")
123 f.mse[1, 1] <- mean((f.estimator[,1]-g1)^2)
124 f.mse[2, 1] <- mean((f.estimator[,2]-g2)^2)
125 f.mse[3, 1] <- mean((f.estimator[,3]-g3)^2)
126 f.mse[4, 1] <- mean((f.estimator[,4]-g4)^2)
127
128 f.mse[1, 2] <- mean((x1*b[2]+am$smooth[,1]-g1)^2)
129 f.mse[2, 2] <- mean((x2*b[3]+am$smooth[,2]-g2)^2)
130 f.mse[3, 2] <- mean((x3*b[4]+am$smooth[,3]-g3)^2)
131 f.mse[4, 2] <- mean((x4*b[5]+am$smooth[,4]-g4)^2)
132 library(xtable)
133 xtable(f.mse)
```

第 10 章 海量数据分析

10.1 线性回归

假设 n 个数据被分成 K 块，记玫