中国人民大學
RENMIN UNIVERSITY OF CHINA

非参数回归的**R**语言实现

中国人民大学统计学院
陈堰平

背景

- 回归模型

$$E(Y \mid \mathbf{X}) = f(\mathbf{X})$$

  - 回归函数形式已知---参数回归
  - 回归函数形式未知---非参数回归

参数回归

Example:

```
> x=sort(runif(200))
> y=2*x+1+rnorm(200,0,0.1)
> fit.lin<-lm(y~x)
```

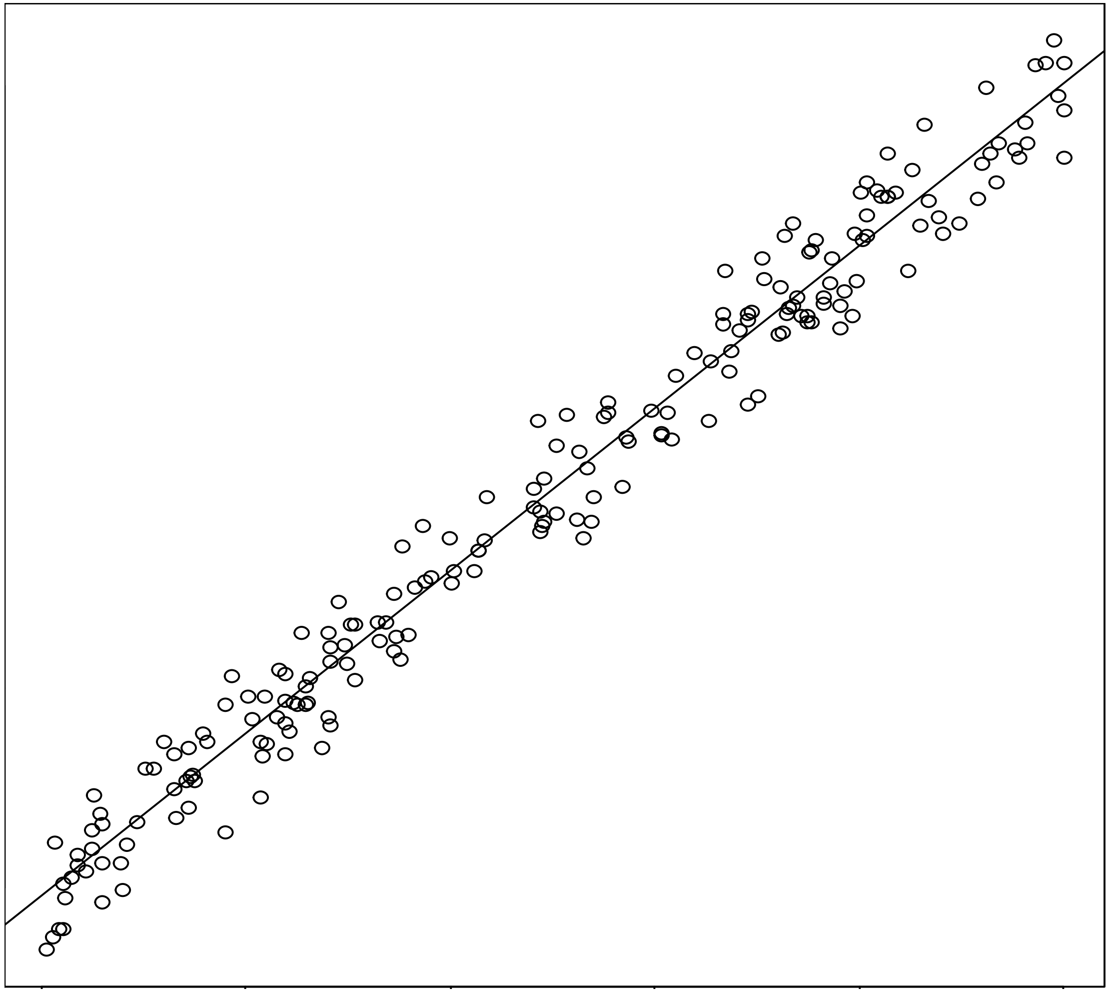
```
> summary(fit.lin)

Call:
lm(formula = y ~ x)

Residuals:
      Min        1Q    Median        3Q       Max
-0.200168 -0.066969 -0.003402  0.070464  0.208087

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)  0.97997    0.01277   76.75   <2e-16 ***
x            2.02368    0.02236   90.50   <2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.09269 on 198 degrees of freedom
Multiple R-squared: 0.9764,     Adjusted R-squared: 0.9763
F-statistic:  8189 on 1 and 198 DF,  p-value: < 2.2e-16
```



非参数回归

- 回归函数未知，要根据观测值估计给定点的估计值
  - 假设观测为(Y_i, X_i)，$i=1,\ldots,n$，假设模型为

$$Y = f(X) + \varepsilon$$

其中 $\varepsilon \sim N(0,\sigma^2)$，给定 $X = x$，

$$f(x) = E(Y \mid X = x)$$

核函数法

非参数回归的基本方法有核函数法，最近邻函数法，样条函数法，小波函数法。这些方法尽管起源不一样，数学形式相距甚远，但都可以视为关于 Y_i 的线性组合的某种权函数。也就是说，回归函数 $f(X)$ 的估计 $f_n(X)$ 总可以表为下述形式：

$$f_n(X) = \sum_{i=1}^{n} W_i(X) Y_i$$

在一般实际问题中，权函数都满足下述条件：

$$W_i(X; X_1, \cdots, X_n) \geq 0, \sum_{i=1}^{n} W_i(X; X_1, \cdots, X_n) = 1$$

- 核函数法(Nadaraya-Watson)

$$W_i(X;X_1,\cdots,X_n)=K\left(\frac{X-X_i}{h}\right)\Big/\sum_{i=1}^{n}\left(\frac{X-X_i}{h}\right)$$

$$Y=f(X)=\sum_{i=1}^{n}W_i(X)Y_i=\sum_{i=1}^{n}\frac{K\left(\frac{X-X_i}{h}\right)}{\sum_{j=1}^{n}K\left(\frac{X-X_i}{h}\right)}Y_i$$

局部多项式估计

利用局部展开的思想，在待估计点，将函数泰勒展开

$$f(x) = f(x_0) + f'(x_0)(x - x_0) + \cdots$$

距离x_0较近的点，提供的信息多，距离远的点，提供的信息少

$$(a,b) = \arg\min_{a,b} \sum_{i=1}^{n} [Y_i - (a + b(X_i - x_0))]^2 K\left(\frac{X_i - x_0}{h}\right)$$

$$\hat{f}(x_0) = a$$

可以转化为加权最小二乘的问题

```r
Ker <- function(u)
    {1/sqrt(2*pi)*exp(-u^2/2) }

x=sort(runif(200))
y=sin(10*x)+rnorm(200,0,0.1)
x0=0.5
h=0.02          # bandwidth
z=x-x0
wx=Ker(z/h) # weights
fit0 <- lm(y~z,weights = wx)
y.est = fit0$coef[1]
y.est           # estimation
sin(10*0.5) # True value
```

y.est = -0.9689503 , sin(10*0.5) = -0.9589243

带宽h的选择

- Cross Validation

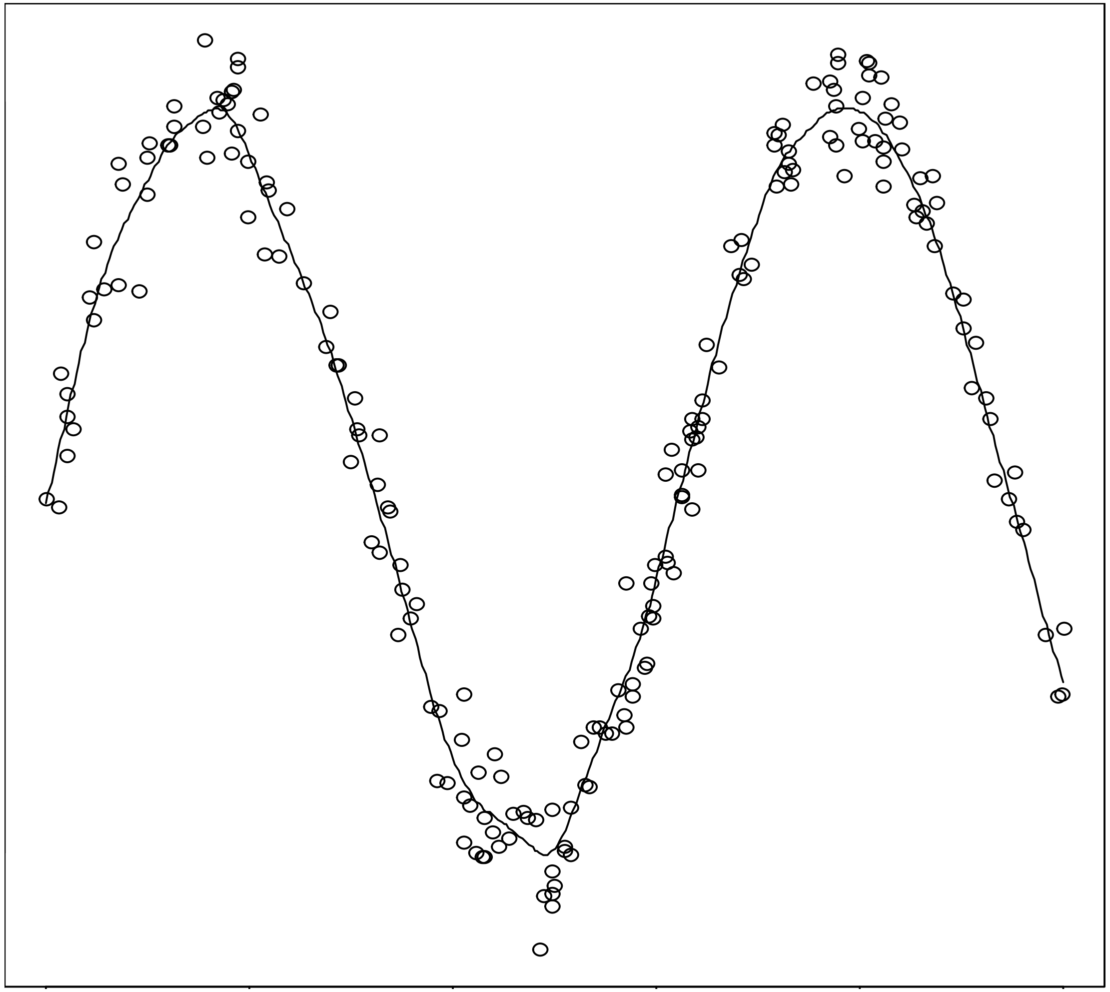
$$CV = \frac{1}{n}\sum_{i=1}^{n}[Y_i - \hat{f}_{(-i)}(X_i)]^2$$

选取一系列的h，计算相应的CV，使得CV最小的就是最优带宽

现成的包

KernSmooth，locpol，…

```
require(KernSmooth)
x=sort(runif(200))
y=sin(10*x)+rnorm(200,0,0.1)
plot(x,y)
h=dpill(x,y)
fit <- locpoly(x,y,kernel = "normal",
      bandwidth=h,)
lines(fit)
```



在分位回归的应用

- quantreg包中有lprq函数

```
lprq <- function (x, y, h, tau = 0.5, m = 50){
  xx <- seq(min(x), max(x), length = m)
  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)
}
```

- 原理

线性分位回归

$$q_\tau(y) = a + bx$$

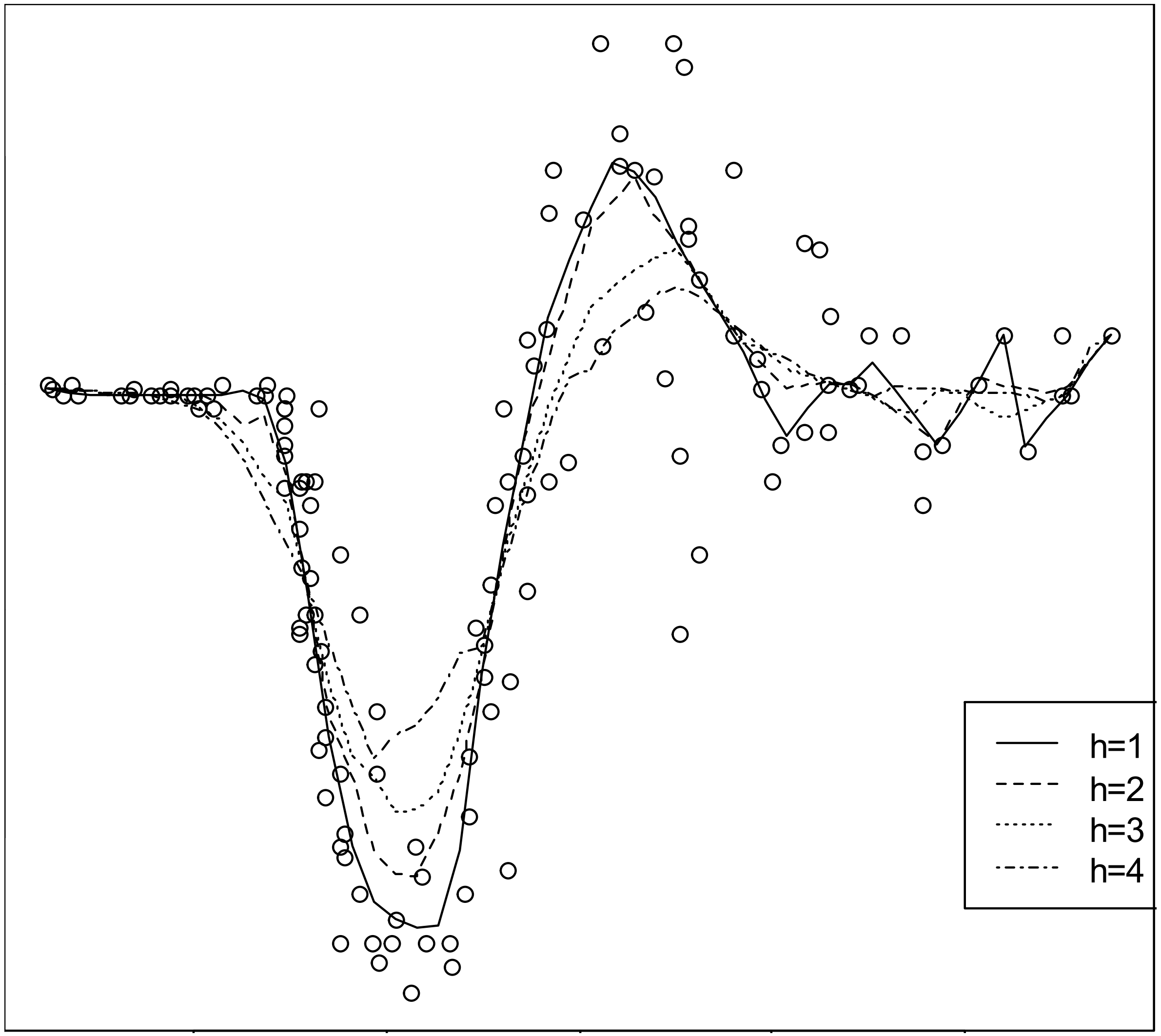
估计方程

$$(a,b) = \arg\min_{(a,b)} \sum_{i=1}^{n} \rho_\tau(Y_i - a - bX_i)$$

非参数分位回归的估计方程

$$(a,b) = \arg\min_{a,b} \sum_{i=1}^{n} \rho_\tau\left(Y_i - (a + b(X_i - x_0))\right) K\left(\frac{X_i - x_0}{h}\right)$$

```r
require(MASS)
data(mcycle)
attach(mcycle)
plot(times,accel,xlab = "milliseconds",
     ylab = "acceleration (in g)")
hs <- c(1,2,3,4)
for(i in hs){
        h = hs[i]
        fit <- lprq(times,accel,h=h,tau=.5)
        lines(fit$xx,fit$fv,lty=i)
        }
legend(50,-70,c("h=1","h=2","h=3","h=4"),
     lty=1:length(hs))
```


h=1
h=2
h=3
h=4

Why R?

灵活：研究新的模型时，可以在原有代码的基础上修改

变系数分位回归模型：

$$q_\tau(y) = c_1(u)x + c_0$$

$$\arg\min_{a,b} \sum_{i=1}^{n} \rho_\tau\left(Y_i - (a + b(U_i - u_0))X_i - c_0\right) K\left(\frac{U_i - u_0}{h}\right)$$

$$= \arg\min_{a,b} \sum_{i=1}^{n} \rho_\tau\left(Y_i - aX_i - b(U_i - u_0)X_i - c_0\right) K\left(\frac{U_i - u_0}{h}\right)$$

```
lprq0<-function (x, u, y, h, tau =
0.5,u0){
     #对单点进行估计
    require(quantreg)
    fv <- u0
    dv <- u0
    z <- u - u0
    wx <- Ker(z/h)
    r <- rq(y ~ x+I(z*x), weights = wx,
     method="br",tau = tau, ci = FALSE)
    fv <- r$coef[c(1,2)]
    dv <- r$coef[3]
    list(u0 = u0, fv = fv, dv = dv)
}
```