线性分位数回归模型在 r 中的实现

1　回归算法

1.1　说明

模型：$\boldsymbol{Y} = \boldsymbol{X}\boldsymbol{\beta} + \boldsymbol{\varepsilon}$

$\boldsymbol{Y} = (y_1, y_2, \ldots, y_n)^{\mathrm{T}}$

$\boldsymbol{X} = (\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_n)^{\mathrm{T}}$　$\boldsymbol{x}_i = (x_{i1}, x_{i2}, \ldots, x_{ip})^{\mathrm{T}}$

$\boldsymbol{\beta} = (\beta_1, \beta_2, \ldots, \beta_p)^{\mathrm{T}}$

$\boldsymbol{\varepsilon} = (\varepsilon_1, \varepsilon_2, \ldots, \varepsilon_n)^{\mathrm{T}}$

可识别条件：$P\{\varepsilon_i < 0 | X_i\} = \tau.$

目标函数：$\min \sum_{i=1}^{n} \rho_\tau(y_i - \boldsymbol{x}_i^{\mathrm{T}}\boldsymbol{\beta})$

其中 $\rho_\tau(t) = t(\tau - I_{\{t<0\}})$

由于 lp 函数假定了变量非负，所以在使用 lp 函数求解时，只能采用分解正负部的方法。但是对于对偶问题，我们要按照 β 无约束的情况考虑.

1.2　处理 $\rho_\tau(t)$

根据实变函数的知识，可以将任意函数 f 分解成正部 f^+ 和负部 f^- 相减的形式:

$$f = f^+ - f^-$$

$$|f| = f^+ + f^-$$

其中 $f^+ = \max(f, 0), f^- = \max(-f, 0) = -\min(f, 0).$

$$\begin{aligned}
\rho_\tau(t) &= t\tau - tI_{\{t<0\}} \\
&= t\tau - \min(t, 0) \\
&= \tau(t^+ - t^-) + t^- \\
&= \tau t^+ + (1-\tau)t^-
\end{aligned}$$

1.3 线性规划求解

u_i, v_i 分别表示 $y_i - \boldsymbol{x}_i^{\mathrm{T}}\boldsymbol{\beta}$ 的正部与负部，$\boldsymbol{e}_n = (1, 1, \ldots, 1)^{\mathrm{T}}$ 表示 n 个 1 的列向量. 则目标函数可以写成:

$$\begin{aligned}\sum_{i=1}^{n} \rho_\tau(y_i - \boldsymbol{x}_i^{\mathrm{T}}\boldsymbol{\beta}) &= \sum_{i=1}^{n} (\tau u_i + (1-\tau) v_i) \\ &= \tau \boldsymbol{e}_n^{\mathrm{T}} \boldsymbol{u} + (1-\tau) \boldsymbol{e}_n^{\mathrm{T}} \boldsymbol{v}\end{aligned}$$

其中 $\boldsymbol{u} = (u_1, u_2, \ldots, u_n)^{\mathrm{T}}, \boldsymbol{v} = (v_1, v_2, \ldots, v_n)^{\mathrm{T}}$.

进一步可以将问题归结为如下线性规划问题:

$$\begin{aligned}&\min \boldsymbol{c\theta} \\ s.t.&\boldsymbol{A\theta} = \boldsymbol{Y}(\text{等式约束}) \\ &\boldsymbol{B\theta} \geqslant \boldsymbol{0}_{2p+2n}(\text{不等式约束})\end{aligned} \tag{1}$$

其中 $\boldsymbol{\theta} = (\boldsymbol{\beta}_+^{\mathrm{T}}, \boldsymbol{\beta}_-^{\mathrm{T}}, \boldsymbol{u}^{\mathrm{T}}, \boldsymbol{v}^{\mathrm{T}})^{\mathrm{T}}$, 前 $2p$ 个值为我们关心的模型系数.

记 $\boldsymbol{0}_{m\times n}$ 表示 $m \times n$ 个 0 的矩阵,$\boldsymbol{0}_m = \boldsymbol{0}_{m\times 1}$, $\boldsymbol{c} = (\boldsymbol{0}_{2p}^{\mathrm{T}}, \tau\boldsymbol{e}_n^{\mathrm{T}}, (1-\tau)\boldsymbol{e}_n^{\mathrm{T}})$.

等式约束由 $\boldsymbol{Y} - \boldsymbol{X\beta} = \boldsymbol{u} - \boldsymbol{v}$ 整理而来, 记 $\boldsymbol{I}_n$ 为 n 阶单位矩阵,$\boldsymbol{A} = (\boldsymbol{X}, -\boldsymbol{X}, \boldsymbol{I}_n, -\boldsymbol{I}_n)$.

不等式约束由 $\boldsymbol{u} \geqslant \boldsymbol{0}_n, \boldsymbol{v} \geqslant \boldsymbol{0}_n$ 以及 $\boldsymbol{\beta}$ 正负部非负整理而来.

$$\boldsymbol{B} = \boldsymbol{I}_{2p+2n}$$

至此，用 R 中的 lpSolve 包中的 lp 函数即可求解.

R 语言实现如下:

```r
IPRQ<-function(X,Y,tau){
  X=as.matrix(X)
  n=length(Y)
  p=ncol(X)
  library(lpSolve)
  e.n=rep(1,n)
  I.n=diag(n)
  n.t=2*p+2*n
  AE=cbind(X,-X,I.n,-I.n)
  AI=diag(n.t)
  A=rbind(AE,AI)
  f.obj <- c(rep(0,2*p),tau*e.n,(1-tau)*e.n)
  f.con <-A
```

```
  ce=rep("=",n)
  ci=rep(">=",n.t)
  f.dir=c(ce,ci)
  f.rhs <- c(Y,rep(0,n.t))
  out=lp ("min", f.obj, f.con, f.dir, f.rhs)
  theta.hat=out$solution
  beta.hat= theta.hat[1:(p)]-theta.hat[(p+1):(2*p)]
  result=list(beta.hat=beta.hat)
  return(result)
}
```

1.3.1 变量选择分位数回归

我们前面提到的方法，在样本量较大时，会导致机器内存溢出不可解。首先可以考虑加入变量选择，将部分系数压缩为 0. 目标函数:$\sum_{i=1}^{n}\rho_\tau(y_i-\boldsymbol{x_i}^{\mathrm{T}}\boldsymbol{\beta})+\sum_{j=1}^{p}\lambda_j\,|\beta_j|$

对于 $|\beta_j|$，我们仍然采用正负部的方法处理。

$$\begin{aligned}\sum_{i=1}^{n}\rho_\tau(y_i-\boldsymbol{x}_i^{\mathrm{T}}\boldsymbol{\beta})+\sum_{j=1}^{p}\lambda_j\,|\beta_j| &= \sum_{i=1}^{n}(\tau u_i+(1-\tau)v_i)+\sum_{j=1}^{p}(\lambda_j\beta_j^{+}+\lambda_j\beta_j^{-})\\ &= \tau\boldsymbol{e}_n^{\mathrm{T}}\boldsymbol{u}+(1-\tau)\boldsymbol{e}_n^{\mathrm{T}}\boldsymbol{v}+\boldsymbol{\lambda}^{\mathrm{T}}\boldsymbol{\beta}^{+}+\boldsymbol{\lambda}^{\mathrm{T}}\boldsymbol{\beta}^{-}\end{aligned} \tag{2}$$

进一步可以将问题归结为如下线性规划问题：

$$\begin{aligned}&\min \boldsymbol{c\theta}\\ &s.t.\boldsymbol{A\theta}=\boldsymbol{Y}(\text{等式约束})\\ &\boldsymbol{B\theta}\geqslant\boldsymbol{0}_{2p+2n}(\text{不等式约束})\end{aligned} \tag{3}$$

此时的 $\boldsymbol{\theta}=(\boldsymbol{\beta^+}^{\mathrm{T}},\boldsymbol{\beta^-}^{\mathrm{T}},\boldsymbol{u}^{\mathrm{T}},\boldsymbol{v}^{\mathrm{T}})^{\mathrm{T}}$, 我们关心的 $\boldsymbol{\beta}$ 由 $\boldsymbol{\beta}=\boldsymbol{\beta}^+-\boldsymbol{\beta}^-$ 获得 ($\boldsymbol{\theta}$ 中前 $2p$ 个值).

此时的 $\boldsymbol{c}=(\boldsymbol{\lambda}^{\mathrm{T}},\boldsymbol{\lambda}^{\mathrm{T}},\tau\boldsymbol{e}_n^{\mathrm{T}},(1-\tau)\boldsymbol{e}_n^{\mathrm{T}})$, 其中 $\boldsymbol{\lambda}=(\lambda_1,\lambda_2,\ldots,\lambda_p)^{\mathrm{T}}$ 为变量选择的参数. 考虑 adplasso，我们选取

$$\boldsymbol{\lambda}=\frac{\log n}{\left|\hat{\boldsymbol{\beta}}\right|}$$

等式约束由 $\boldsymbol{Y}-\boldsymbol{X\beta}=\boldsymbol{u}-\boldsymbol{v}$ 整理而来, 利用 $\boldsymbol{\beta}=\boldsymbol{\beta}^+-\boldsymbol{\beta}^-$ 代入. 此时的 $\boldsymbol{A}=(\boldsymbol{X},-\boldsymbol{X},\boldsymbol{I}_n,-\boldsymbol{I}_n)$.

不等式约束由 $\boldsymbol{u}\geqslant\boldsymbol{0}_n,\boldsymbol{v}\geqslant\boldsymbol{0}_n,\boldsymbol{\beta}^+\geqslant\boldsymbol{0}_p,\boldsymbol{\beta}^-\geqslant\boldsymbol{0}_p$ 整理而来, 此时 $\boldsymbol{B}=I_{2p+2n}$.

R 语言实现如下：

```
VSRQ<- function(X,y,tau,lambda){
  require(SparseM)
  require(quantreg)
  X <- as.matrix(x)
  p <- ncol(X)
  n <- length(y)
  pn=2*p+2*n
  if(length(lambda)==1) lambda=rep(lemabda,p)
  e.n=rep(1,n)
  I.n=diag(n)
  f.obj=c(lambda,lambda,tau*e.n,(1-tau)*e.n)
  XE=cbind(X,-X,I.n,-I.n)
  f.con=rbind(XE,diag(pn))
  f.dir= c(rep("=", n), rep(">=", pn))
  f.rhs=c(y,rep(0,pn))
  out=lp ("min", f.obj, f.con, f.dir, f.rhs)
  theta.hat=out$solution
  beta.hat=theta.hat[1:p]-theta.hat[(p+1):(2*p)]
  result = list(beta.hat= beta.hat)
  return(result)
}
```

1.3.2 对偶问题

此外，在线性规划问题中，可以构造和原问题等价的对偶问题。当原问题求解困难时，可以求解对偶问题。根据[1] 中的介绍，首先考虑不含惩罚中1.3中 (1)的情况。

首先将原问题整理成标准形式 (左), 对偶问题如下 (右)。不同参数的约束条件不同，同一行的约束条件在原问题与对偶问题下相互对应.

[1]最优化技术与数学建模 http://www.tup.com.cn/upload/books/yz/035790-01.pdf

$$\begin{aligned} &\max -\boldsymbol{c\theta} \\ &s.t. -\boldsymbol{A\theta}=-\boldsymbol{Y} \\ &\boldsymbol{\beta}\text{无约束} \\ &\boldsymbol{u}\geqslant 0 \\ &\boldsymbol{v}\geqslant 0 \end{aligned} \qquad\qquad \begin{aligned} &\min -\boldsymbol{Y}^{\mathrm{T}}\boldsymbol{a} \\ &s.t.\boldsymbol{a}\text{无约束} \\ &-\boldsymbol{X}^{\mathrm{T}}\boldsymbol{a}=-\boldsymbol{0}_p \\ &-\boldsymbol{I}_n^{\mathrm{T}}\boldsymbol{a}\geqslant -\tau\boldsymbol{e}_n \\ &\boldsymbol{I}_n^{\mathrm{T}}\boldsymbol{a}\geqslant -(1-\tau)\boldsymbol{e}_n \end{aligned}$$

其中 $\boldsymbol{a}$ 为对偶问题的参数，可以对约束条件进行变形，使问题简单。

令 $\boldsymbol{b}=\boldsymbol{a}+(1-\tau)\boldsymbol{e}_n$, 加减常数项不影响极值点，所以可以将问题转化为:

$$\begin{aligned} &\max \boldsymbol{Y}^{\mathrm{T}}\boldsymbol{b} \\ &s.t.\boldsymbol{X}^{\mathrm{T}}\boldsymbol{b}=(1-\tau)\boldsymbol{X}^{\mathrm{T}}\boldsymbol{e}_n \\ &\boldsymbol{0}_n\leqslant \boldsymbol{b}\leqslant \boldsymbol{e}_n \end{aligned} \tag{4}$$

至此，可以利用 rq.fit.sfn 求解, 实现如下:

```
DRQ <- function(X,y,tau){
  require(SparseM)
  require(quantreg)
  X <- as.matrix(x)
  p <- ncol(X)
  n <- length(y)
  D=as.matrix.csr(X)
  a <- (1-tau)*(t(X)%*%rep(1,n))
  fit=rq.fit.sfn(D,y,rhs=a)
  result = list(beta.hat=fit$coef)
  return(result)
}
```

同样，我们可以写出1.3.1中(3)的对偶问题. 在此处，我们采用伪回归变量的方法.

记 $\boldsymbol{\Sigma}_\lambda=diag(\boldsymbol{\lambda}),\tilde{\boldsymbol{Y}}=(\boldsymbol{Y}^{\mathrm{T}},\boldsymbol{0}_p^{\mathrm{T}})^{\mathrm{T}},\tilde{\boldsymbol{X}}=(\boldsymbol{X}^{\mathrm{T}},\boldsymbol{\Sigma}_\lambda)^{\mathrm{T}}$.

目标函数(2)可以整理成以下形式:

$$\begin{aligned} \sum_{i=1}^{n}\rho_\tau(\tilde{y}_i-\tilde{\boldsymbol{x}}_i^{\mathrm{T}}\boldsymbol{\beta})+\sum_{j=n+1}^{n+p}\left|\tilde{y}_j-\tilde{\boldsymbol{x}}_j^{\mathrm{T}}\boldsymbol{\beta}\right| &= \sum_{i=1}^{n}(\tau u_{1i}+(1-\tau)v_{1i})+\sum_{j=n+1}^{n+p}(u_{2j}+v_{2j}) \\ &= \tau\boldsymbol{e}_n^{\mathrm{T}}\boldsymbol{u}_1+(1-\tau)\boldsymbol{e}_n^{\mathrm{T}}\boldsymbol{v}_1+\boldsymbol{e}_p^{\mathrm{T}}\boldsymbol{u}_2+\boldsymbol{e}_p^{\mathrm{T}}\boldsymbol{v}_2 \end{aligned}$$

从而将问题归结为:

$$\begin{aligned} &\min \boldsymbol{c\theta} \\ s.t.&\boldsymbol{A\theta} = \tilde{\boldsymbol{Y}}(\text{等式约束}) \\ &\boldsymbol{B\theta} \geqslant \mathbf{0}_{4p+2n}(\text{不等式约束}) \end{aligned} \tag{5}$$

此时的 $\boldsymbol{\theta} = (\boldsymbol{\beta}_+^{\mathrm{T}}, \boldsymbol{\beta}_-^{\mathrm{T}}, \boldsymbol{u}_1^{\mathrm{T}}, \boldsymbol{v}_1^{\mathrm{T}}, \boldsymbol{u}_2^{\mathrm{T}}, \boldsymbol{v}_2^{\mathrm{T}})^{\mathrm{T}}, \boldsymbol{c} = (\mathbf{0}_{2p}^{\mathrm{T}}, \tau\boldsymbol{e}_n^{\mathrm{T}}, (1-\tau)\boldsymbol{e}_n^{\mathrm{T}}, \boldsymbol{e}_p^{\mathrm{T}}, \boldsymbol{e}_p^{\mathrm{T}})$.

$\boldsymbol{A} = \begin{pmatrix} \boldsymbol{X} & -\boldsymbol{X} & I_n & -I_n & \mathbf{0}_{n\times p} & \mathbf{0}_{n\times p} \\ \boldsymbol{\Sigma}_\lambda & -\boldsymbol{\Sigma}_\lambda & \mathbf{0}_{p\times n} & \mathbf{0}_{p\times n} & I_p & -I_p \end{pmatrix}$

$\boldsymbol{B} = \boldsymbol{I}_{2n+4p}$

(5)的 R 实现如下:

```
VSIPRQ1<-function(X,Y,tau,lambda){
  X=as.matrix(X)
  n=length(Y)
  p=ncol(X)
  library(lpSolve)
  e.n=rep(1,n)
  I.n=diag(n)
  if(length(lambda)==1) lambda=rep(lambda,p)
  AE1=cbind(X,-X,I.n,-I.n,matrix(0,n,2*p))
  AE2=cbind(diag(lambda),-diag(lambda), matrix(0,p,2*n
      ),diag(p),-diag(p))
  AE=rbind(AE1,AE2)
  AI=diag(2*n+4*p)
  A=rbind(AE,AI)
  f.obj <- c(rep(0,2*p),tau*e.n,(1-tau)*e.n,rep(1,2*p)
      )
  f.con <-A
  ce=rep("=",n+p)
  ci=rep(">=",2*n+4*p)
  f.dir=c(ce,ci)
  f.rhs <- c(Y,rep(0,5*p+2*n))
  out=lp ("min", f.obj, f.con, f.dir, f.rhs)
  theta.hat=out$solution
  beta.hat= theta.hat[1:p]-theta.hat[(p+1):(2*p)]
  result=list(beta.hat=beta.hat)
  return(result)
```

}

此问题的对偶问题为:

$$
\begin{aligned}
&\max \tilde{\boldsymbol{Y}}^{\mathrm{T}}\boldsymbol{a}\\
&s.t. \boldsymbol{X}^{\mathrm{T}}\boldsymbol{a}_1 + \boldsymbol{\Sigma}_\lambda \boldsymbol{a}_2 = \boldsymbol{0}_p\\
&-(1-\tau)\boldsymbol{e}_n \leqslant \boldsymbol{a}_1 \leqslant \tau \boldsymbol{e}_n\\
&-\boldsymbol{e}_p \leqslant \boldsymbol{a}_2 \leqslant \boldsymbol{e}_p
\end{aligned}
$$

$\boldsymbol{a} = (\boldsymbol{a}_1^{\mathrm{T}}, \boldsymbol{a}_2^{\mathrm{T}})^{\mathrm{T}}, \boldsymbol{a}_1 = (a_1, a_2, \ldots, a_n)^{\mathrm{T}}, \boldsymbol{a}_2 = (a_{n+1}, a_{n+2}, \ldots, a_{n+p})^{\mathrm{T}}$.

同样,我们可以取 $\boldsymbol{b} = (\boldsymbol{b}_1^{\mathrm{T}}, \boldsymbol{b}_2^{\mathrm{T}})^{\mathrm{T}}, \boldsymbol{b}_1 = \boldsymbol{a}_1 + (1-\tau)\boldsymbol{e}_n, \boldsymbol{b}_2 = \frac{\boldsymbol{a}_2+\boldsymbol{e}_p}{2}$, 将问题进一步整理为:

$$
\begin{aligned}
&\max \tilde{\boldsymbol{Y}}^{\mathrm{T}}\boldsymbol{b}\\
&s.t. \boldsymbol{X}^{\mathrm{T}}\boldsymbol{b}_1 + 2\boldsymbol{\Sigma}_\lambda \boldsymbol{b}_2 = (1-\tau)\boldsymbol{X}^{\mathrm{T}}\boldsymbol{e}_n + \boldsymbol{\lambda}\\
&\boldsymbol{0}_{n+p} \leqslant \boldsymbol{b} \leqslant \boldsymbol{e}_{n+p}
\end{aligned}
\tag{6}
$$

(6)的 R 实现如下:

```r
VSDRQ <- function(X,y,tau,lambda,acc=1e-3){
  require(SparseM)
  require(quantreg)
  if(length(lambda)==1) lambda=rep(lemabda,p)
  temp=X
  X=rbind(X,2*diag(lambda))
  X <- as.matrix(X)
  p <- ncol(X)
  n <- length(y)
  D <- as.matrix.csr(X)
  Y <- c(y,rep(0,p))
  a <- (1-tau)*(t(temp)%*%matrix(1,n,1)) +lambda
  fit=rq.fit.sfn(D,Y,rhs=a)

  beta.hat=fit$coef
  beta.hat[abs(beta.hat)<=acc]=0
  result=list(beta.hat=beta.hat)

  return(result)
}
```

1.3.3 复合分位数回归

我们可以对简单分位数回归进行拓展，同时考虑 M 个分位数. 问题在于对于不同的 τ, 识别条件不同. 对于线性模型 $y_i = \boldsymbol{x_i}\boldsymbol{\beta} + \varepsilon_i$, 我们有 $P\{\epsilon_i < 0|\boldsymbol{x}_i\} = 0 a.s.$ 因此，我们可以通过调节截距项，使对不同的 τ 均满足识别条件.

$$\begin{aligned} y_i =& \boldsymbol{x_i}\boldsymbol{\beta} + \varepsilon_i \\ =& \boldsymbol{x_i}\boldsymbol{\beta} + \beta_0(\tau) + \varepsilon_i(\tau) \end{aligned}$$

对于固定的 τ, 有 $\sqrt{n}(\hat{\boldsymbol{\beta}} - \boldsymbol{\beta}) \xrightarrow{L} N(0, \frac{\tau(1-\tau)}{f^2(0)}\boldsymbol{\Sigma}^{-1})$.

对于 $\boldsymbol{\tau} = (\tau_1, \tau_2, \ldots, \tau_k)$, 目标函数:

$$\sum_{k=1}^{M}\sum_{i=1}^{n}\rho_{\tau_k}(y_i - \boldsymbol{x_i}\boldsymbol{\beta} - \beta_0(\tau_k))$$

此时要考虑的参数为 $\bar{\boldsymbol{\beta}} = (\boldsymbol{\beta}^{\mathrm{T}}, \beta_0(\tau_1), \beta_0(\tau_2), \ldots, \beta_0(\tau_M))^{\mathrm{T}}$

$$\begin{aligned} \sum_{k=1}^{M}\sum_{i=1}^{n}\rho_{\tau_k}(y_i - \boldsymbol{x_i}\boldsymbol{\beta} - \beta_0(\tau_k)) =& \sum_{k=1}^{M}\sum_{i=1}^{n}\rho_{\tau_k}(y_i - \tilde{\boldsymbol{x}}_i\tilde{\boldsymbol{\beta}}_k) \\ =& \sum_{k=1}^{M}\sum_{i=1}^{n}\tau_k u_{ki} + (1-\tau_k)v_{ki} \\ =& \sum_{k=1}^{M}\tau_k\boldsymbol{e}_n^{\mathrm{T}}\boldsymbol{u}_k + (1-\tau_k)\boldsymbol{e}_n^{\mathrm{T}}\boldsymbol{v}_k \\ =& \boldsymbol{\tau}\boldsymbol{U} + (\boldsymbol{e}_M^{\mathrm{T}} - \boldsymbol{\tau})\boldsymbol{V} \end{aligned}$$

其中

$$\boldsymbol{U} = \begin{pmatrix} \boldsymbol{e}_n^{\mathrm{T}}\boldsymbol{u}_1 \\ \boldsymbol{e}_n^{\mathrm{T}}\boldsymbol{u}_2 \\ \vdots \\ \boldsymbol{e}_n\boldsymbol{u}_M \end{pmatrix} = \begin{pmatrix} \boldsymbol{e}_n^{\mathrm{T}} & & & \\ & \boldsymbol{e}_n^{\mathrm{T}} & & \\ & & \ddots & \\ & & & \boldsymbol{e}_n^{\mathrm{T}} \end{pmatrix}(\boldsymbol{u}_1^{\mathrm{T}}, \boldsymbol{u}_2^{\mathrm{T}}, \ldots, \boldsymbol{u}_M^{\mathrm{T}})^{\mathrm{T}} = \boldsymbol{D}_M\tilde{\boldsymbol{u}}$$

$$\boldsymbol{V} = \begin{pmatrix} \boldsymbol{e}_n^{\mathrm{T}}\boldsymbol{v}_1 \\ \boldsymbol{e}_n^{\mathrm{T}}\boldsymbol{v}_2 \\ \vdots \\ \boldsymbol{e}_n\boldsymbol{v}_M \end{pmatrix} = \begin{pmatrix} \boldsymbol{e}_n^{\mathrm{T}} & & & \\ & \boldsymbol{e}_n^{\mathrm{T}} & & \\ & & \ddots & \\ & & & \boldsymbol{e}_n^{\mathrm{T}} \end{pmatrix}(\boldsymbol{v}_1^{\mathrm{T}}, \boldsymbol{v}_2^{\mathrm{T}}, \ldots, \boldsymbol{v}_M^{\mathrm{T}})^{\mathrm{T}} = \boldsymbol{D}_M\tilde{\boldsymbol{v}}$$

问题归结为:

$$\begin{aligned} &\min \boldsymbol{c\theta} \\ s.t.&\boldsymbol{A\theta} = \tilde{\boldsymbol{Y}}(\text{等式约束}) \\ &\boldsymbol{B\theta} \geqslant \boldsymbol{0}_{2Mn+2(p+M)}(\text{不等式约束}) \end{aligned} \tag{7}$$

此时:

$$\boldsymbol{\theta} = (\bar{\boldsymbol{\beta}}^{+}, \bar{\boldsymbol{\beta}}^{-}, \tilde{\boldsymbol{u}}^{\mathrm{T}}, \tilde{\boldsymbol{v}}^{\mathrm{T}})^{\mathrm{T}}$$

$$\boldsymbol{c} = (\boldsymbol{0}_{2(p+M)}^{\mathrm{T}}, \boldsymbol{\tau D}_M, (\boldsymbol{e}_M^{\mathrm{T}} - \boldsymbol{\tau})\boldsymbol{D}_M)$$

$$\boldsymbol{T} = \begin{pmatrix} \boldsymbol{X} & \boldsymbol{e}_n & & & \\ \boldsymbol{X} & & \boldsymbol{e}_n & & \\ \vdots & & & \ddots & \\ \boldsymbol{X} & & & & \boldsymbol{e}_n \end{pmatrix}$$

$$\boldsymbol{A} = (\boldsymbol{T}, -\boldsymbol{T}, \boldsymbol{I}_{Mn}, -\boldsymbol{I}_{Mn})$$

$$\tilde{\boldsymbol{Y}} = (\boldsymbol{Y}^{\mathrm{T}}, \boldsymbol{Y}^{\mathrm{T}}, \ldots, \boldsymbol{Y}^{\mathrm{T}})^{\mathrm{T}}$$

$$\boldsymbol{B} = \boldsymbol{I}_{2Mn+2(p+M)}$$

```
CRQ<-function(X,Y,tau){
    n=length(Y)
    p=ncol(X)
    M=length(tau)
    library(Matrix)
    e.n=t(rep(1,n))
    e.M=t(rep(1,M))
    D.M=diag(M)%x%e.n
    f.obj <- c(rep(0,2*(p+M)),tau%*%D.M,(e.M-tau)%*%D.M)
    temp=cbind(matrix(1,M,1)%x%X,diag(M)%x%t(e.n))
    AE=cbind(temp,-temp,diag(M*n),-diag(M*n))
    AI=diag(2*M*n+2*(p+M))
    A=rbind(AE,AI)
    f.con <-A
    ce=rep("=",M*n)
    ci=rep(">=",2*M*n+2*(p+M))
    f.dir=c(ce,ci)
    f.rhs <- c(rep(Y,M),rep(0,(2*(p+M)+2*M*n)))
    out=lp ("min", f.obj, f.con, f.dir, f.rhs)
```

```
    theta.hat=out$solution
    beta.hat=theta.hat[1:(2*(p+M))]
    beta=beta.hat[1:(p+M)] - beta.hat[(p+M+1):(2*(p+M))]
    result=list(beta.hat=beta)
  return(result)
}
```

此问题的对偶问题为:

$$\begin{aligned}
&\max \tilde{\boldsymbol{Y}}^{\mathrm{T}}\boldsymbol{a}\\
&s.t.\boldsymbol{T}^{\mathrm{T}}\boldsymbol{a}=\mathbf{0}_{p+M}\\
&-((\boldsymbol{e}_M^{\mathrm{T}}-\boldsymbol{\tau})\boldsymbol{D}_M)^{\mathrm{T}}\leqslant\boldsymbol{a}\leqslant(\boldsymbol{\tau}\boldsymbol{D}_M)^{\mathrm{T}}
\end{aligned}$$

同样, 取 $\boldsymbol{b}=\boldsymbol{a}+((\boldsymbol{e}_M^{\mathrm{T}}-\boldsymbol{\tau})\boldsymbol{D}_M)^{\mathrm{T}}$, 问题化简为:

$$\begin{aligned}
&\max \tilde{\boldsymbol{Y}}^{\mathrm{T}}\boldsymbol{b}\\
&s.t.\boldsymbol{T}^{\mathrm{T}}\boldsymbol{b}=\boldsymbol{T}^{\mathrm{T}}((\boldsymbol{e}_M^{\mathrm{T}}-\boldsymbol{\tau})\boldsymbol{D}_M)^{\mathrm{T}}\\
&\mathbf{0}\leqslant\boldsymbol{b}\leqslant(\boldsymbol{e}_M^{\mathrm{T}}\boldsymbol{D}_M)^{\mathrm{T}}
\end{aligned}$$

```
DCRQ <- function(X,y,tau){
  require(SparseM)
  require(quantreg)
  n=length(y)
  p=ncol(X)
  M=length(tau)
  library(Matrix)
  e.n=t(rep(1,n))
  e.M=t(rep(1,M))
  D.M=diag(M)%x%e.n
  temp=cbind(matrix(1,M,1)%x%X,diag(M)%x%t(e.n))
  X <- as.matrix(temp)
  D=as.matrix.csr(X)
  Y=rep(y,M)
  a <- t(temp)%*%t((e.M-tau)%*%(D.M) )
  fit=rq.fit.sfn(D,Y,rhs=a)
  beta.hat=fit$coefficients
  return(list(beta.hat=beta.hat))
}
```

考虑变量选择:

$$
\begin{aligned}
&\min \boldsymbol{c}\boldsymbol{\theta} \\
s.t.&\boldsymbol{A}\boldsymbol{\theta} = \tilde{\boldsymbol{Y}}(\text{等式约束}) \\
&\boldsymbol{B}\boldsymbol{\theta} \geqslant \boldsymbol{0}_{2Mn+2(p+M)}(\text{不等式约束})
\end{aligned} \tag{8}
$$

此时:

$$\boldsymbol{\theta} = (\bar{\boldsymbol{\beta}}^{+}, \bar{\boldsymbol{\beta}}^{-}, \tilde{\boldsymbol{u}}^{\mathrm{T}}, \tilde{\boldsymbol{v}}^{\mathrm{T}})^{\mathrm{T}}$$

$$\boldsymbol{\lambda} = (\lambda_1, \lambda_2, \ldots, \lambda_p, \lambda_{p+1}, \ldots, \lambda_{p+M})$$

$$\boldsymbol{c} = (\boldsymbol{\lambda}, \boldsymbol{\lambda}, \boldsymbol{\tau}\boldsymbol{D}_M, (\boldsymbol{e}_M^{\mathrm{T}} - \boldsymbol{\tau})\boldsymbol{D}_M)$$

记

$$\boldsymbol{T} = \begin{pmatrix} \boldsymbol{X} & \boldsymbol{e}_n & & & \\ \boldsymbol{X} & & \boldsymbol{e}_n & & \\ \vdots & & & \ddots & \\ \boldsymbol{X} & & & & \boldsymbol{e}_n \end{pmatrix}$$

$$\boldsymbol{A} = (\boldsymbol{T}, -\boldsymbol{T}, \boldsymbol{I}_{Mn}, -\boldsymbol{I}_{Mn})$$

$$\tilde{\boldsymbol{Y}} = (\boldsymbol{Y}^{\mathrm{T}}, \boldsymbol{Y}^{\mathrm{T}}, \ldots, \boldsymbol{Y}^{\mathrm{T}})^{\mathrm{T}}$$

$$\boldsymbol{B} = \boldsymbol{I}_{2Mn+2(p+M)}$$

R 实现如下:

```r
VSCRQ<-function(X,Y,tau,lambda){
  n=length(Y)
  p=ncol(X)
  M=length(tau)
  library(Matrix)
  e.n=t(rep(1,n))
  e.M=t(rep(1,M))
  D.M=diag(M)%x%e.n
  f.obj <- c(lambda,lambda,tau%*%D.M,(e.M-tau)%*%D.M)
  temp=cbind(matrix(1,M,1)%x%X,diag(M)%x%t(e.n))
  AE=cbind(temp,-temp,diag(M*n),-diag(M*n))
  AI=diag(2*M*n+2*(p+M))
  A=rbind(AE,AI)
  f.con <-A
  ce=rep("=",M*n)
  ci=rep(">=",2*M*n+2*(p+M))
```

```
  f.dir=c(ce,ci)
  f.rhs <- c(rep(Y,M),rep(0,(2*(p+M)+2*M*n)))
  out=lp ("min", f.obj, f.con, f.dir, f.rhs)
  theta.hat=out$solution
  beta.hat=theta.hat[1:(2*(p+M))]
  beta=beta.hat[1:(p+M)] - beta.hat[(p+M+1):(2*(p+M))]
  result=list(beta.hat=beta)
  return(result)
}
```

另外，我们也可以考虑伪回归变量的方法：

$$\begin{aligned} &\min \boldsymbol{c\theta} \\ s.t.&\boldsymbol{A\theta} = \tilde{\boldsymbol{Y}}(\text{等式约束}) \\ &\boldsymbol{B\theta} \geqslant \boldsymbol{0}_{2Mn+4(p+M)}(\text{不等式约束}) \end{aligned} \tag{9}$$

此时:

$$\boldsymbol{\theta} = (\bar{\boldsymbol{\beta}}^{+}, \bar{\boldsymbol{\beta}}^{-}, \tilde{\boldsymbol{u}}^{\mathrm{T}}, \tilde{\boldsymbol{v}}^{\mathrm{T}}, \boldsymbol{u}_{\lambda}^{\mathrm{T}}, \boldsymbol{v}_{\lambda}^{\mathrm{T}})^{\mathrm{T}}$$

$$\boldsymbol{\Sigma}_{\lambda} = diag(\boldsymbol{\lambda})$$

$$\boldsymbol{c} = (\boldsymbol{0}_{2(p+M)}, \boldsymbol{\tau D}_M, (\boldsymbol{e}_M^{\mathrm{T}} - \boldsymbol{\tau})\boldsymbol{D}_M, \boldsymbol{e}_{p+M}^{\mathrm{T}}, \boldsymbol{e}_{p+M}^{\mathrm{T}})$$

记

$$\boldsymbol{T} = \begin{pmatrix} \boldsymbol{X} & \boldsymbol{e}_n & & & \\ \boldsymbol{X} & & \boldsymbol{e}_n & & \\ \vdots & & & \ddots & \\ \boldsymbol{X} & & & & \boldsymbol{e}_n \end{pmatrix}$$

$$\boldsymbol{A} = \begin{pmatrix} \boldsymbol{T} & -\boldsymbol{T} & \boldsymbol{I}_{Mn} & -\boldsymbol{I}_{Mn} & & \\ \boldsymbol{\Sigma}_{\lambda} & -\boldsymbol{\Sigma}_{\lambda} & & & \boldsymbol{I}_{p+M} & -\boldsymbol{I}_{p+M} \end{pmatrix}$$

$$\tilde{\boldsymbol{Y}} = (\boldsymbol{Y}^{\mathrm{T}}, \boldsymbol{Y}^{\mathrm{T}}, \ldots, \boldsymbol{Y}^{\mathrm{T}}, \boldsymbol{0}_{p+M}^{\mathrm{T}})^{\mathrm{T}}$$

$$\boldsymbol{B} = \boldsymbol{I}_{2Mn+4(p+M)}$$

R 实现如下:

```
VSCRQ2<-function(X,Y,tau,lambda){
  n=length(Y)
  p=ncol(X)
  M=length(tau)
  library(Matrix)
```

```
  e.n=t(rep(1,n))
  e.M=t(rep(1,M))
  D.M=diag(M)%x%e.n
  f.obj <- c(rep(0,2*(p+M)),tau%*%D.M,(e.M-tau)%*%D.M,
      rep(1,2*(p+M))  )
  temp=cbind(matrix(1,M,1)%x%X,diag(M)%x%t(e.n))
  AE1=cbind(temp,-temp,diag(M*n),-diag(M*n),matrix(0,M
     *n,2*(p+M))  )
  AE2=cbind(diag(lambda),-diag(lambda),matrix(0,p+M,2*
     M*n),diag(p+M),-diag(p+M))
  AI=diag(2*M*n+4*(p+M))
  A=rbind(AE1,AE2,AI)
  f.con <-A
  ce=rep("=",M*n+p+M)
  ci=rep(">=",2*M*n+4*(p+M))
  f.dir=c(ce,ci)
  f.rhs <- c(rep(Y,M),rep(0,(p+M)),rep(0,2*M*n+4*(p+M)
     ))
  out=lp ("min", f.obj, f.con, f.dir, f.rhs)
  theta.hat=out$solution
  beta.hat=theta.hat[1:(2*(p+M))]
  beta=beta.hat[1:(p+M)] - beta.hat[(p+M+1):(2*(p+M))]
  result=list(beta.hat=beta)
  return(result)
}
```

此问题的对偶问题为:

$$
\begin{aligned}
&\max \tilde{\boldsymbol{Y}}^{\mathrm{T}}\boldsymbol{h} \\
s.t.&\boldsymbol{T}^{\mathrm{T}}\boldsymbol{h}_1+\boldsymbol{\Sigma}_{\lambda}\boldsymbol{h}_2=\boldsymbol{0} \\
&-((\boldsymbol{e}_M^{\mathrm{T}}-\boldsymbol{\tau})\boldsymbol{D}_M)^{\mathrm{T}}\leqslant \boldsymbol{h}_1\leqslant(\boldsymbol{\tau}\boldsymbol{D}_M)^{\mathrm{T}} \\
&-\boldsymbol{e}_{p+M}\leqslant \boldsymbol{h}_2\leqslant \boldsymbol{e}_{p+M}
\end{aligned}
$$

同样, 取 $\boldsymbol{b}_1=\boldsymbol{h}_1+((\boldsymbol{e}_M^{\mathrm{T}}-\boldsymbol{\tau})\boldsymbol{D}_M)^{\mathrm{T}}, \boldsymbol{b}_2=\frac{\boldsymbol{e}_{p+M}+\boldsymbol{h}_2}{2}$ 问题化简为:

$$
\begin{aligned}
&\max \tilde{\boldsymbol{Y}}^{\mathrm{T}}\boldsymbol{b}\\
s.t.&\boldsymbol{T}^{\mathrm{T}}\boldsymbol{b}_1+2\boldsymbol{\Sigma}_\lambda\boldsymbol{b}_2=\boldsymbol{T}^{\mathrm{T}}((\boldsymbol{e}_M^{\mathrm{T}}-\boldsymbol{\tau})\boldsymbol{D}_M)^{\mathrm{T}}+\boldsymbol{\Sigma}_\lambda\boldsymbol{e}_{p+M}\\
&\boldsymbol{0}\leqslant\boldsymbol{b}_1\leqslant(\boldsymbol{e}_M^{\mathrm{T}}\boldsymbol{D}_M)^{\mathrm{T}}\\
&\boldsymbol{0}\leqslant\boldsymbol{b}_2\leqslant\boldsymbol{e}_{p+M}
\end{aligned}
$$

R 实现如下:

```
VSDCRQ2 <- function(X,y,tau,lambda){
  require(SparseM)
  require(quantreg)
  n=length(y)
  p=ncol(X)
  M=length(tau)
  library(Matrix)
  e.n=t(rep(1,n))
  e.M=t(rep(1,M))
  e.pM=t(rep(1,p+M))
  D.M=diag(M)%x%e.n
  temp=cbind(matrix(1,M,1)%x%X,diag(M)%x%t(e.n))
  X <- as.matrix(rbind(temp,2*diag(lambda)))
  D=as.matrix.csr(X)
  Y=c(rep(y,M),rep(0,p+M))
  a <- t(temp)%*%t((e.M-tau)%*%(D.M)  )+diag(lambda)%*%t(e.pM)
  fit=rq.fit.sfn(D,Y,rhs=a)
  beta.hat=fit$coefficients
  return(list(beta.hat=beta.hat))
}
```

下面两种算法主要参考 [2].

1.4 ADMM

Alternating Direction Method of Multipliers (ADMM) 交替方向乘子法

分位数回归将问题归结为 (首先不考虑变量选择):

$$
\begin{aligned}
\min_{\beta\in\mathbb{R}^{p+1}} \quad & \sum_{i=1}^{n}\rho_\tau(r_i)\\
\textit{subject to} \quad & X\beta+r=Y
\end{aligned}
$$

增量拉格朗日函数为:

$$L_\rho(x,\lambda)=\sum_{i=1}^{n}\rho_\tau(r_i)+\lambda^{\mathrm{T}}(X\beta+r-Y)+(\rho/2)\left\|X\beta-r-Y\right\|_2^2$$

引入松弛变量 $X\beta+r-Y=u/\rho$

$$\begin{aligned}\min_{\beta\in\mathbb{R}^{p+1}}\quad & \sum_{i=1}^{n}\rho_\tau(r_i)+\frac{\rho}{2}\left\|Y-r-X\beta+u/\rho\right\|_2^2\\ \textit{subject to}\quad & X\beta+r-Y=u/\rho\end{aligned}$$

迭代规则:

$$\begin{aligned}r^{(t+1)}=&\arg\min_{r\in\mathbb{R}^n}\sum_{i=1}^{n}\rho_\tau(r_i)+\frac{\rho}{2}\left\|Y-r-X\beta^{(t)}+u^{(t)}/\rho\right\|_2^2\\ \beta^{(t+1)}=&\arg\min_{r\in\mathbb{R}^{p+1}}\frac{\rho}{2}\left\|Y-r^{(t+1)}-X\beta^{(t)}+u^{(t)}/\rho\right\|_2^2\\ u^{(t+1)}=&u^{(t)}+\rho(Y-r^{(t+1)}-X\beta^{(t+1)})\end{aligned}$$

r 的更新：利用

$$\rho_\tau(t)=\tau t-\min(t,0)=\tau t+\frac{1}{2}(|t|-t)=(\tau-\frac{1}{2})t+\frac{1}{2}\left|t\right|$$

整理目标函数. 记

$$\begin{aligned}L=&\sum_{i=1}^{n}\rho_\tau(r_i)+\frac{\rho}{2}\left\|Y-r-X\beta+u^{(t)}/\rho\right\|_2^2\\ =&\sum_{i=1}^{n}\left[(\tau-\frac{1}{2})r_i+\frac{1}{2}\left|r_i\right|\right]+\frac{\rho}{2}\left\|Y-r-X\beta+u^{(t)}/\rho\right\|_2^2\end{aligned}$$

$$\frac{\partial L}{\partial r_i}=(\tau-\frac{1}{2})+\frac{1}{2}\frac{\partial|r_i|}{\partial r_i}+\rho(Y-X\beta-r+u^{(t)}/\rho)(-1)$$

取 $c=Y-X\beta^{(t)}+u^{(t)}/\rho$, 令 $\frac{\partial L}{\partial r_i}=0$. 可得

$$r_i=c_i-\frac{1}{\rho}(\tau-\frac{1}{2})-\frac{1}{2\rho}\frac{\partial\left|r_i\right|}{\partial r_i}$$

利用正负部分解，可以将 r 的进一步更新整理成:

$$r^{(t+1)}=S_{1/\rho}(c-(2\tau_{n\times1}-1_{n\times1})/\rho)$$

. 其中：$c=Y-X\beta^{(t)}+u^{(t)}/\rho, S_a(v)_i=(v_i-a)_+-(-v_i-a)_+$

β 的更新用最小二乘求解：$\beta^{(t+1)}=(X^{\mathrm{T}}X)^{-1}X(Y-r^{(t+1)}+u^{(t)}/\rho)$

程序实现:

```
ADMMRQ <- function(X,y,tau,iter=200,esp=1e-03){
  n=length(y)
  e.new=e.old=rep(0,n)
  X=as.matrix(X)
  p=ncol(X)
  beta.new=beta.old=rep(0,p)
  u.new=u.old=rep(0,n)
  step=0
  error=1
  while(step<iter&error>esp){
    beta.old=beta.new
    e.old=e.new
    u.old=u.new
    step=step+1
    c=y-X%*%beta.old+u.old/1.2
    c1=c-(2*tau-1)/1.2
    e.new=pmax(c1-1/1.2,0)-pmax(-c1-1/1.2,0)
    beta.new=solve(t(X)%*%X)%*%t(X)%*%(y-e.new+u.old
        /1.2)
    temp=y-e.new-X%*%beta.new
    u.new=u.old+temp*1.2
    temp1=y-e.new-X%*%beta.new
    temp2=1.2*t(X)%*%(e.new-e.old)
    error1=max(sqrt(sum((X%*%beta.new)^2)),sqrt(sum(
        e.new^2)),sqrt(sum(y^2)))
    error2=sqrt(p)*0.01+esp*sqrt(sum((X%*%beta.new)
        ^2))
    if(sqrt(sum(temp1^2))<sqrt(n)*0.01+esp*error1&
        sqrt(sum(temp2^2))<error2) step=iter+1
     error=sqrt(sum((beta.new-beta.old)^2))
  }
  e.hat=y-X%*%beta.new
  loss=sum(e.hat*(tau-1*(e.hat<0)))
   tmp.out = list(beta.hat=beta.new, loss=loss,ind=
       step)
return(tmp.out)
```

```
}
```

下面考虑复合分位数回归，需要重新设计矩阵.

$Y^\star=(Y^{\mathrm{T}},Y^{\mathrm{T}},\ldots,Y^{\mathrm{T}})^{\mathrm{T}},\beta=(\beta_1,\beta_2,\ldots,\beta_p,\beta_{p+1},\ldots,\beta_{p+K})^{\mathrm{T}}$

$$X^\star=\begin{pmatrix}\boldsymbol{X} & \boldsymbol{e}_n & & & \\ \boldsymbol{X} & & \boldsymbol{e}_n & & \\ \vdots & & & \ddots & \\ \boldsymbol{X} & & & & \boldsymbol{e}_n\end{pmatrix}$$

$\tau^\star=(\tau_{1n\times 1},\tau_{2n\times 1},\ldots,\tau_{Kn\times 1}),\boldsymbol{e}_n=(1,1,\ldots,1)^{\mathrm{T}}$ 从而问题为:

$$\min_{\beta\in\mathbb{R}^{p+K}}\quad \sum_{k=1}^{K}\sum_{i=1}^{n}\rho_{\tau_k}(r_{ik})$$
$$\textit{subject to}\quad X^\star\beta+r=Y^\star$$

类似的可以得到迭代规则:

$$\begin{aligned}
r^{(t+1)}=&\arg\min_{r\in\mathbb{R}^n}\sum_{k=1}^{K}\sum_{i=1}^{n}\rho_{\tau_k}(r_{ik})+\frac{\rho}{2}\left\|Y^\star-r-X^\star\beta^{(t)}+u^{(t)}/\rho\right\|_2^2\\
=&S_{\frac{1}{\rho}}\left(c-\frac{2\tau^\star-1_{n\times 1}}{\rho}\right)\\
\beta^{(t+1)}=&\arg\min_{r\in\mathbb{R}^{p+1}}\frac{\rho}{2}\left\|Y^\star-r^{(t+1)}-X^\star\beta^{(t)}+u^{(t)}/\rho\right\|_2^2\\
=&({X^\star}^{\mathrm{T}}X^\star)^{-1}X^\star(Y^\star-r^{(t+1)}+u^{(t)}/\rho)\\
u^{(t+1)}=&u^{(t)}+\rho(Y^\star-r^{(t+1)}-X^\star\beta^{(t+1)})
\end{aligned}$$

程序实现:

```
ADMMCRQ <- function(X,y,tau,iter=2000,esp=1e-03){
  n=length(y)
  K=length(tau)
  X=as.matrix(X)
  p=ncol(X)
  e.n=t(rep(1,n))
  X=cbind(matrix(1,K,1)%x%X,diag(K)%x%t(e.n))
  y=rep(y,K)
  e.new=e.old=rep(0,n*K)
  tau=matrix(tau%x%rep(1,n))
  beta.new=beta.old=rep(0,p+K)
  u.new=u.old=rep(0,n*K)
```

```
  step=0
  error=1
  while(step<iter&error>esp){
    beta.old=beta.new
    e.old=e.new
    u.old=u.new
    step=step+1
    c=y-X%*%beta.old+u.old/1.2
    c1=c-(2*tau-1)/1.2
    e.new=pmax(c1-1/1.2,0)-pmax(-c1-1/1.2,0)
    beta.new=solve(t(X)%*%X)%*%t(X)%*%(y-e.new+u.old/
        1.2)
    temp=y-e.new-X%*%beta.new
    u.new=u.old+temp*1.2
    temp1=y-e.new-X%*%beta.new
    temp2=1.2*t(X)%*%(e.new-e.old)
    error1=max(sqrt(sum((X%*%beta.new)^2)),sqrt(sum(e.
        new^2)),sqrt(sum(y^2)))
    error2=sqrt(p)*0.01+esp*sqrt(sum((X%*%beta.new)^2)
        )
    if(sqrt(sum(temp1^2))<sqrt(n)*0.01+esp*error1&sqrt
        (sum(temp2^2))<error2)  step=iter+1
    error=sqrt(sum((beta.new-beta.old)^2))
  }
  e.hat=y-X%*%beta.new
  loss=sum(e.hat*(tau-1*(e.hat<0)))
  tmp.out = list(beta.hat=beta.new,  loss=loss,ind=step
      )
  return(tmp.out)
}
```

1.5 MM

Majorize-Minimization (MM) 算法在 [1] 中有详细的说明. 主要思想是构造一个 $g(\theta|\theta^{(m)})$ 在 $\theta^{(m)}$ 处与 $f(\theta)$ 相切，其他处均大于 $f(\theta)$, 即:

$$\begin{aligned} g(\theta|\theta^{(m)}) &\geqslant f(\theta) \quad \forall\theta \\ g(\theta^{(m)}|\theta^{(m)}) &= f(\theta^{(m)}) \end{aligned}$$

通过求 $g(\theta|\theta^{(m)})$ 的极值，进行迭代.

算法的收敛性:

$$f(\theta^{(m)}) = g(\theta^{(m)}|\theta^{(m)}) \geqslant g(\theta^{(m+1)}|\theta^{(m)}) \geqslant f(\theta^{(m+1)})$$

分位数损失函数为:

$$\rho_\tau(t) = \begin{cases} \tau t & t \geqslant 0 \\ -(1-\tau)t & t < 0 \end{cases}$$

构造二次函数:

$$\xi_\tau^\varepsilon(r|r^{(t)}) = \frac{1}{4}\left[\frac{r^2}{\varepsilon + |r^{(t)}|} + (4\tau - 2)r + c\right]$$

通过求导可以得到更新规则为:

$$\beta^{(t+1)} = \beta^{(t)} + \left(\frac{X^{\mathrm{T}}X}{\varepsilon + |r^{(t)}|}\right)^{-1} X^{\mathrm{T}}\left(2\tau - 1 + \frac{Y - X\beta^{(t)}}{\varepsilon + |r^{(t)}|}\right)$$

$$\beta^{(t+1)} = \left(\frac{X^{\mathrm{T}}X}{\varepsilon + |r^{(t)}|}\right)^{-1} X^{\mathrm{T}}\left(2\tau - 1 + \frac{Y}{\varepsilon + |r^{(t)}|}\right)$$

两个是一样的.

程序实现:

```
MMRQ=function(x,y,tau,delta=1e-3){
  p=ncol(x)
  n=length(y)
  beta.hat=rnorm(p)
  beta.new=beta.hat
  beta.old=rep(n,length(beta.new))
  loop=0
  Loop=1000
  esp=p*0.01
  while(sum(abs(beta.new-beta.old))>esp&&loop<Loop){
```

```
      loop=loop+1
      beta.old=beta.new
      e.hat=y-x%*%beta.old
      W=diag(as.vector(1/(delta+abs(e.hat))))
      beta.new=beta.old+solve(t(x)%*%W%*%x)%*%t(x)%*%
          (2*tau-1+e.hat/(abs(e.hat)+delta))
      #beta.new=solve(t(x)%*%W%*%x)%*%t(x)%*%(2*tau-1+y
          /(abs(e.hat)+delta))
    }
    ind=sum(abs(beta.new-beta.old))#1*(loop<Loop)
    ind=loop
    loss=0
    temp=y-x%*%beta.new
    loss=sum(temp*(tau-1*(temp<0)))
    tmp.out = list(beta.hat=beta.new, loss=loss,ind=
        ind)
  return(tmp.out)
}
```

和前面一样修改矩阵，可以得到复合分位数的算法:

```
MMCRQ=function(x,y,tau,delta=1e-3){
  n=length(y)
  K=length(tau)
  x=as.matrix(x)
  p=ncol(x)
  e.n=t(rep(1,n))
  x=cbind(matrix(1,K,1)%x%x,diag(K)%x%t(e.n))
  y=rep(y,K)
  e.new=e.old=rep(0,n*K)
  tau=matrix(tau%x%rep(1,n))
  beta.new=beta.old=rep(0,p+K)
  loop=0
  Loop=1000
  esp=1e-3
  error=1
  while(error>esp&&loop<Loop){
    loop=loop+1
```

```
    beta.old=beta.new
    e.hat=y-x%*%beta.old
    W=diag(as.vector(1/(delta+abs(e.hat))))
    beta.new=beta.old+solve(t(x)%*%W%*%x)%*%t(x)%*%(2*
        tau-1+e.hat/(abs(e.hat)+delta))
    #beta.new=solve(t(x)%*%W%*%x)%*%t(x)%*%(2*tau-1+y/
        (abs(e.hat)+delta))
    error=sum(abs(beta.new-beta.old))
  }
  ind=sum(abs(beta.new-beta.old))#1*(loop<Loop)
  ind=loop
  loss=0
  temp=y-x%*%beta.new
  loss=sum(temp*(tau-1*(temp<0)))
  tmp.out = list(beta.hat=beta.new, loss=loss,ind=ind)
  return(tmp.out)
}
```

1.6 模拟

数据按 $y_i=\sum_{j=1}^{p}x_{ij}\beta_j+\varepsilon_i$ 生成, 其中 ε_i 为 i.i.d. 标准正态分布.$p=5, n=200,400,600,800,1000,2000$.$\beta_j$ 为 $[-1,1]$ 上的均匀分布.$error=\sum_{j=1}^{p}\left|\hat{\beta}_j-\beta_j\right|$.

表 1: 无惩罚分位数回归

(n,p)	IP(quantreg)		IPRQ		DRQ		ADMMRQ		MMRQ	
	error	time	error	time	error	time	error	time	error	time
(200 , 5)	0.428	0.003	0.428	0.024	0.428	0.004	0.438	0.020	0.554	0.003
(400 , 5)	0.181	0.001	0.181	0.100	0.181	0.004	0.190	0.018	0.340	0.016
(600 , 5)	0.230	0.002	0.230	0.228	0.230	0.005	0.225	0.021	0.191	0.032
(800 , 5)	0.278	0.001	0.278	0.431	0.278	0.005	0.282	0.034	0.302	0.074
(1000 , 5)	0.181	0.003	0.181	0.824	0.181	0.006	0.153	0.046	0.170	0.178
(2000 , 5)	0.145	0.007	0.145	3.349	0.145	0.013	0.146	0.044	0.350	0.386

IP(quantreg) 是 quantreg 包中方法.IPRQ 是用 lp 函数求解.DRQ 是用 rq.fit.sfn 函数解对偶问题.ADMMRQ 是 ADMM 算法.MMRQ 是 MM 算法.

对于复合分位数的模拟，$\tau=(0.2,0.5,0.8)$.

表 2: 无惩罚复合分位数

(n,p)	CRQ		DCRQ		ADMMCRQ		MMCRQ	
	error	time	error	time	error	time	error	time
(200 , 5)	0.228	0.292	0.228	0.013	0.223	0.074	0.220	0.342
(400 , 5)	0.154	1.260	0.154	0.012	0.157	0.119	0.149	0.777
(600 , 5)	0.190	3.185	0.190	0.013	0.192	0.100	0.178	2.114
(800 , 5)	0.185	6.332	0.185	0.021	0.179	0.183	0.190	3.956
(1000 , 5)	0.143	9.914	0.143	0.025	0.144	0.167	0.144	5.778

表 3: cqrReg 包中的函数求解

(n,p)	ip		mm		admm	
	error	time	error	time	error	time
(200 , 5)	0.233	0.292	0.234	0.013	0.228	0.074
(400 , 5)	0.157	1.260	0.156	0.012	0.160	0.120
(600 , 5)	0.194	3.185	0.193	0.013	0.196	0.100
(800 , 5)	0.188	6.333	0.189	0.021	0.183	0.184
(1000 , 5)	0.146	9.914	0.146	0.025	0.147	0.168

2 渐进方差的估计

核心理论:

$$\sqrt{n}(\hat{\beta}-\beta)=\left[\frac{1}{n}\sum x_i x_i^{\mathrm{T}} f(0|x_i)+o_p(1)\right]^{-1}\frac{1}{\sqrt{n}}\sum_{i=1}^{n} x_i^{\mathrm{T}}\psi(\varepsilon_i)+o_p(1)$$

$$\sqrt{n}(\hat{\beta}-\beta)\sim N(0,\Sigma_1^{-1}\Sigma_2\Sigma_1)$$

其中 $\Sigma_1=E(X^{\mathrm{T}}Xf(0|X)),\Sigma_2=\tau(1-\tau)E(X^{\mathrm{T}}X)$.

$f(0|X)$ 表示给定 X 下 ε 的概率密度函数在 0 处取值. [2]

2.1 Bootstrap

适用范围广. 通过构造多个 β 的估计值，进而计算得到方差.

取 $E(W_i)=1,Var(W_i)=1$.

$\hat{\beta}_W=\arg\min\sum W_i\rho_\tau(r_i)$

$\boldsymbol{W}=(W_1,W_2,\ldots,W_n)$

$\sqrt{n}(\hat{\beta}_{\boldsymbol{W}}-\hat{\beta})\to N(0,\Sigma_1^{-1}\Sigma_2\Sigma_1)$

[2]由于目标函数不光滑，不容易得到，可以采用下面的方法.

程序实现:

```
ESD<-function(x,y,tau,B=500){
  n=length(y)
  p=ncol(x)
  beta.hat=rq(y~x+0,tau)$coef
  BETA=matrix(0,B,p)
  for(i in 1:B){
    W=rexp(n)
    BETA[i,]=rq(y~x+0,tau,weight=W)$coef-beta.hat
  }
  res=list(beta.sd=sqrt(diag(cov(BETA) )))
 return(res)
}
```

2.2 估计 $f(0|X)$

模型: $y_i = X_i\beta + \varepsilon_i$

给定 X_i, y_i 的分位数函数为 $F^{-1}(y_i|X_i) = \tau, F(y_i|X_i)$ 为给定 X_i, y_i 的分布函数.

$$\begin{aligned} y_i &= X_i\beta(\tau) + \varepsilon_i(\tau) & P(\varepsilon_i(\tau) < 0|X_i) &= \tau(\text{识别条件}) \\ F(X_i\beta(\tau)|X_i) &= \tau & F^{-1}(\tau|X_i) &= X_i\beta(\tau) \end{aligned}$$

由 $\frac{\partial F^{-1}(x)}{\partial x} = \frac{1}{f(x)}$, 有

$$\frac{1}{f(0|X_i)} = \frac{\partial F^{-1}(\tau|X_i)}{\partial \tau} = \lim_{h\to 0} \frac{F^{-1}(\tau+h) - F^{-1}(\tau-h)}{2h}$$

进而得到估计:

$$\hat{f}(0|X_i) = \frac{2h}{X_i\beta(\tau+h) - X_i\beta(\tau-h)}$$

从而得到 $\hat{\Sigma}_1$. [3]

程序实现:

```
sdEST<-function(X,y,tau){
 require('quantreg')
 p=ncol(X)
 n=length(y)
```

[3] 此方法效果对 h 的选取很敏感

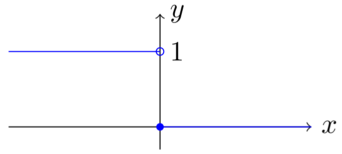
```
  h<- bandwidth.rq(tau, n, hs = TRUE)
  beta1<-IPRQ(X,y,tau+h)$beta.hat
  beta2<-IPRQ(X,y,tau-h)$beta.hat
  f.hat=2*h/(X%*%beta1-X%*%beta2 )
  Sigma1=1/n*t(X)%*%diag(as.vector(f.hat))%*%(X)
  Sigma2=1/n*tau*(1-tau)*t(X)%*%X
  sd=solve(Sigma1)%*%Sigma2%*%solve(Sigma1)
  res=list(beta.sd=diag(sd))
  return (res)
}
```

2.3 改进估计方程

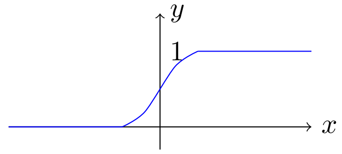
$\hat{\beta} = \arg\min \sum_{i=1}^{n} \rho_\tau(y_i - X_i\beta)$

$L(\beta) = \sum X_i^{\mathrm{T}}(\tau - I_{\{y_i - X_i\beta < 0\}}) = 0$

由于估计方程 $\rho_\tau(t) = \tau(t - I_{\{t<0\}})$ 中指示函数不是连续函数，所以不能直接得到参数的渐进分布. $I_{\{t<0\}}$ 的函数如下图:



标准正太分布的概率分布函数 $\Phi(x)$ 为:



这个方法就是用 $\Phi(-x)$ 代替 $I_{\{x<0\}}$.

$\tilde{L}(\beta) = \sum \left[X_i^{\mathrm{T}}(\tau - \Phi(-\frac{y_i - X_i\beta}{h}))\right]$. [4]

Taylor 展开有:$\tilde{L}(\hat{\beta}) = \tilde{L}(\beta_0) + \tilde{L}'(\beta_0)(\hat{\beta} - \beta_0) + o(\hat{\beta} - \beta_0)$

进而有: $\sqrt{n}(\hat{\beta} - \beta_0) = \left(\frac{1}{n}\tilde{L}'(\beta_0)\right)^{-1} \frac{1}{\sqrt{n}}\tilde{L}(\beta_0)$

$\tilde{L}'(\beta) = \frac{\partial \tilde{L}(\beta)}{\partial \beta} = \sum_{i=1}^{n} X_i^{\mathrm{T}} X_i \phi(\frac{X_i\beta - y_i}{h})\frac{1}{h}$. [5]

程序实现如下:

[4] 此处 h 越小与 $I_{\{t<0\}}$ 越接近.

[5] 相当于对 $f(0|X_i)$ 进行核估计，此时 h 越大越好. 文献中一般要求 $nh^4 \to 0$.

```
SRQ=function(x,y,tau){
    require(MASS)
    p=ncol(x)
    n=length(y)
    h=log(n)/sqrt(n)
    beta.new=rq(y~x+0,tau)$coef
    beta.old=rep(0,p)
    Loop=100
    loop=0
    error=1
    while(loop<Loop&error>1e-3){
      loop=loop+1
      beta.old=beta.new
      e.hat=(y-x%*%beta.old)/h
      L1=t(x)%*%(tau-pnorm(-e.hat))
      L2=-t(x)%*%diag(as.vector(dnorm(e.hat)))%*%x/h
      beta.new=beta.old-ginv(L2)%*%L1
      e=(y-x%*%beta.old)
      error=sum(abs(beta.old-beta.new))
    }
     D=tau*(1-tau)*t(x)%*%x/n
     H=ginv(L2/n)%*%D%*%ginv(L2/n)/n
     #H=ginv(L2/n)%*%L1%*%t(L1)%*%ginv(L2/n)
     beta.sd=sqrt(diag(H))
     tmp.out = list(beta.hat=beta.new, beta.sd= beta.
         sd)
     return(tmp.out)
}
```

2.4 求期望获得连续函数

$\sqrt{n}(\hat{\beta}-\beta_0) \xrightarrow{L} N(0,\Sigma_1^{-1}\Sigma_2\Sigma_1)$

取 $H=\Sigma_1^{-1}\Sigma_2\Sigma_1, Z\sim N(0,I_p)$

$\hat{\beta} = \beta_0 + H^{1/2}Z$

$$\begin{aligned} EL(\hat{\beta}) =& EL(\beta_0 + H^{1/2}Z) = E_Z\left[X_i^{\mathrm{T}}(\tau - I_{\{y_i - X_i\beta\}})\right] \\ =& X_i^{\mathrm{T}}\left(\tau - P\{-XH^{1/2}Z < -y_i + X_i\beta\}\right) \\ =& X^{\mathrm{T}}\left(\tau - \Phi(-(XHX^{\mathrm{T}})^{-1/2}(Y - X\beta)\right) \triangleq \bar{L}(\beta) \end{aligned}$$

$$\frac{\partial \bar{L}(\beta)}{\partial \beta} = \frac{1}{n}\sum_{i=1}^{n} X_i X_i^{\mathrm{T}} \frac{1}{h_i}\phi(\frac{y_i - X_i\beta}{h_i}), h_i = \sqrt{X_i^{\mathrm{T}} H X_i}$$

程序实现如下:

```
ISRQ=function(x,y,tau){
  p=ncol(x)
  n=length(y)
  beta.new=rq(y~x+0,tau)$coef
  beta.old=rep(0,p)
  Loop=100
  loop=0
  error=1
  H=diag(p)/n
  while(loop<Loop&error>1e-3){
    loop=loop+1
    beta.old=beta.new
    h=sqrt(diag((x)%*%H%*%t(x)))
    e.hat=-(y-x%*%beta.old)/h
    L1=t(x)%*%(pnorm(e.hat)-tau)
    L2=t(x)%*%diag(as.vector(dnorm(e.hat)/h))%*%x
    beta.new=beta.old-ginv(L2)%*%L1
    e=(y-x%*%beta.old)
    Temp= t(x)%*%e
    D=tau*(1-tau)*t(x)%*%x/n
    #H=ginv(L2)%*%L1%*%t(L1)%*%ginv(L2)
    H=ginv(L2/n)%*%D%*%ginv(L2/n)/n
    error=sum(abs(beta.old-beta.new))
  }
  beta.sd=sqrt(diag(H))
  tmp.out = list(beta.hat=beta.new, beta.sd= beta.sd
    )
  return(tmp.out)
```

}

2.5 模拟

$\beta = (-1, -1, 1, 1, 0, 0), n = 100.$

表 4: 95% 的置信区间

β	ESD		sdEST		SRQ		ISRQ	
	下界	上界	下界	上界	下界	上界	下界	上界
-1	-1.01	-0.95	-1.10	-0.87	-1.01	-0.96	-1.01	-0.96
-1	-1.08	-1.02	-1.10	-1.00	-1.08	-1.02	-1.08	-1.02
1	0.99	1.06	0.98	1.07	1.00	1.06	1.00	1.05
1	1.00	1.06	0.97	1.09	1.00	1.05	1.01	1.05
0	-0.14	-0.08	-0.18	-0.04	-0.13	-0.08	-0.13	-0.09
0	0.07	0.14	0.02	0.19	0.08	0.13	0.08	0.13

参考文献

[1] David R Hunter and Kenneth Lange. A tutorial on mm algorithms. *The American Statistician*, 58(1):30–37, 2004.

[2] Matthew Pietrosanu, Jueyu Gao, Linglong Kong, Bei Jiang, and Di Niu. cqrreg: An r package for quantile and composite quantile regression and variable selection. *arXiv preprint arXiv:1709.04126*, 2017.