Median

Consider a sample \{y_1,\cdots,y_n\}. To compute the median, solve\min_\mu \left\lbrace\sum_{i=1}^n|y_i-\mu|\right\rbracewhich can be solved using linear programming techniques. More precisely, this problem is equivalent to\min_{\mu,\mathbf{a},\mathbf{b}}\left\lbrace\sum_{i=1}^na_i+b_i\right\rbracewith a_i,b_i\geq 0 and y_i-\mu=a_i-b_i, \forall i=1,\cdots,n. Heuristically, the idea is to write y_i=\mu+\varepsilon_i, and then define a_i's and b_i's so that \varepsilon_i=a_i-b_i and |\varepsilon_i|=a_i+b_i, i.e. a_i=(\varepsilon_i)_+=\max\lbrace0,\varepsilon_i\rbrace=|\varepsilon|\cdot\boldsymbol{ 1}_{\varepsilon_i>0}andb_i=(-\varepsilon_i)_+=\max\lbrace0,-\varepsilon_i\rbrace=|\varepsilon|\cdot \boldsymbol{1}_{\varepsilon_i<0}[/latex]denote respectively the positive and the negative parts.</p> <p>Unfortunately (that was the error in my previous post), the expression of linear programs is[latex display="true"]\min_{\mathbf{z}}\left\lbrace\boldsymbol{c}^\top\mathbf{z}\right\rbrace\text{ s.t. }\boldsymbol{A}\mathbf{z}=\boldsymbol{b},\mathbf{z}\geq\boldsymbol{0}In the equation above, with the a_i's and b_i's, we're not far away. Except that we have \mu\in\mathbb{R}, while it should be positive. So similarly, set \mu=\mu^+-\mu^- where \mu^+=(\mu)_+ and \mu^-=(-\mu)_+.

Thus, let\mathbf{z}=\big(\mu^+;\mu^-;\boldsymbol{a},\boldsymbol{b}\big)^\top\in\mathbb{R}_+^{2n+2}and then write the constraint as \boldsymbol{A}\mathbf{z}=\boldsymbol{b} with \boldsymbol{b}=\boldsymbol{y} and \boldsymbol{A}=\big[\boldsymbol{1}_n;-\boldsymbol{1}_n;\mathbb{I}_n;-\mathbb{I}_n\big]And for the objective function\boldsymbol{c}=\big(\boldsymbol{0},\boldsymbol{1}_n,-\boldsymbol{1}_n\big)^\top\in\mathbb{R}_+^{2n+2}

To illustrate, consider a sample from a lognormal distribution,

```
n = 101
set.seed(1)
y = rlnorm(n)
median(y)
[1] 1.077415
```

For the optimization problem, use the matrix form, with 3n constraints, and 2n+1 parameters,

```
library(lpSolve)
X = rep(1,n)
A = cbind(X, -X, diag(n), -diag(n))
b = y
c = c(rep(0,2), rep(1,n),rep(1,n))
equal_type = rep("=", n)
r = lp("min", c,A,equal_type,b)
head(r$solution,1)
[1] 1.077415
```

It looks like it's working well…

Quantile

Of course, we can adapt our previous code for quantiles

```
tau = .3
quantile(y,tau)
      30%
0.6741586
```

The linear program is now\min_{q^+,q^-,\mathbf{a},\mathbf{b}}\left\lbrace\sum_{i=1}^n\tau a_i+(1-\tau)b_i \right\rbracewith a_i,b_i,q^+,q^-\geq 0 and y_i=q^+-q^-+a_i-b_i, \forall i=1,\cdots,n. The R code is now

```
c = c(rep(0,2), tau*rep(1,n),(1-tau)*rep(1,n))
r = lp("min", c,A,equal_type,b)
head(r$solution,1)
[1] 0.6741586
```

So far so good…

Quantile Regression

Consider the following dataset, with rents of flat, in a major German city, as function of the surface, the year of construction, etc.

```
base=read.table("http://freakonometrics.free.fr/rent98_00.txt",header=TRUE)
```

The linear program for the quantile regression is now\min_{\boldsymbol{\beta}^+,\boldsymbol{\beta}^-, \mathbf{a},\mathbf{b}}\left\lbrace\sum_{i=1}^n\tau a_i+(1-\tau)b_i\right\rbracewith a_i,b_i\geq 0 and y_i=\boldsymbol{x}^\top[\boldsymbol{\beta}^+-\boldsymbol{\beta}^-]+a_i-b_i\forall i=1,\cdots,n and \beta_j^+, \beta_j^-\geq 0 \forall j=0,\cdots,k. So use here

```
require(lpSolve)
tau = .3
n=nrow(base)
X = cbind( 1, base$area)
y = base$rent_euro
K = ncol(X)
N = nrow(X)
A = cbind(X,-X,diag(N),-diag(N))
c = c(rep(0,2*ncol(X)),tau*rep(1,N),(1-tau)*rep(1,N))
b = base$rent_euro
const_type = rep("=",N)
r = lp("min",c,A,const_type,b)
beta = r$sol[1:K] -  r$sol[(1:K+K)]
beta
[1] 148.946864   3.289674
```

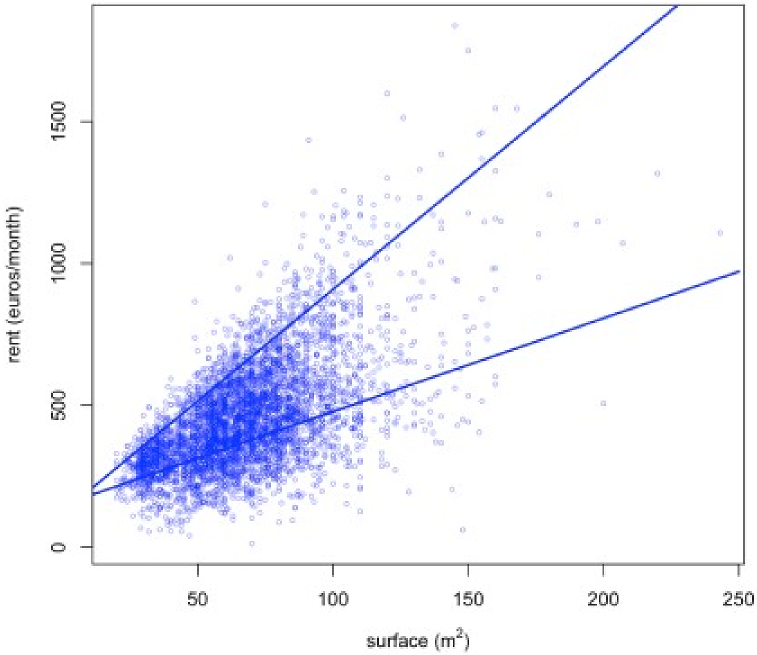
Of course, we can use R function to fit that model

```
library(quantreg)
rq(rent_euro~area, tau=tau, data=base)
Coefficients:
(Intercept)         area
 148.946864     3.289674
```

Here again, it seems to work quite well. We can use a different probability level, of course, and get a plot

```
plot(base$area,base$rent_euro,xlab=expression(paste("surface (",m^2,")")),
     ylab="rent (euros/month)",col=rgb(0,0,1,.4),cex=.5)
sf=0:250
yr=r$solution[2*n+1]+r$solution[2*n+2]*sf
lines(sf,yr,lwd=2,col="blue")
```

```
tau = .9
r = lp("min",c,A,const_type,b)
tail(r$solution,2)
[1] 121.815505   7.865536
yr=r$solution[2*n+1]+r$solution[2*n+2]*sf
lines(sf,yr,lwd=2,col="blue")
```



And we can adapt the later to multiple regressions, of course,

```
X = cbind(1,base$area,base$yearc)
K = ncol(X)
N = nrow(X)
A = cbind(X,-X,diag(N),-diag(N))
c = c(rep(0,2*ncol(X)),tau*rep(1,N),(1-tau)*rep(1,N))
b = base$rent_euro
const_type = rep("=",N)
r = lp("min",c,A,const_type,b)
beta = r$sol[1:K] -  r$sol[(1:K+K)]
beta
[1] -5542.503252     3.978135     2.887234
```

to be compared with

```
library(quantreg)
rq(rent_euro~ area + yearc, tau=tau, data=base)

Coefficients:
 (Intercept)          area         yearc
-5542.503252     3.978135     2.887234

Degrees of freedom: 4571 total; 4568 residual
```