Computer Workshop

This computer workshop presents data analyses by using some of the modelling tools for continuous and discrete outcomes we covered in this course. We use the statistical package R. No prior knowledge of R is assumed.

1 Introduction to R

R is a free, powerful object oriented software for performing statistical analyses. For this workshop no prior knowledge of R is assumed. We have already written the code for you. We will use the code step by step in order to illustrate some aspects of fitting quantile, M-quantile and expectile regression with R.

2 How to download R

1 Go to http://lib.stat.cmu.edu/R/CRAN/

2 Click on operating system you are using Windows or Mac

3 Follow the installation process. R will be automatically installed and a shortcut will be created.

3 How to open R

The installation process automatically creates an R shortcut. Double click this icon to open the R environment.

4 Quantile regression in R - Installing an R library

For quantile regression in R one needs to download library *quantreg*. This can be done as follows:

1 Open R

2 Go to Packages and then select Install Package(s)

3 Select a mirror from the list and click OK

4 Select the appropriate library from the list and click OK. In our case this will be package `quantreg`.

Once the package has been installed, we need to load it for carrying out our analyses. To do this type in the R console library(`quantreg`) and press ENTER. The package has now been loaded and is ready for use.

5 The Data

Fitting quantile regression in R will be illustrated by using the Engel food expenditure data that is also used in Koenker and Bassett(1982). This is a regression data set comprising 235 observations on income and food expenditure for Belgian working class households. The dataset is part of the `quantreg` library. The aim of the modeling exercise is to study the relationship between food expenditure and income by using a least squares regression, a median regression and different quantile, M-quantile and expectile regressions.

In the R console type `data(engel)` and press ENTER.

The data has now been loaded. To see the data Type `engel` and press ENTER.

You will see a data frame consisting of 235 observations for 2 variables.

income: annual household income in Belgian francs;
foodexp: annual household food expenditure in Belgian francs.

6 The rq function

To fit quantile regression in R we need to use function rq (This is part of the *quantreg* library).

The general structure of rq is as follows:
`rq(`$y \sim x$`,tau=c(vector of quantiles))`,
where y is the dependent variable, x is the independent variable and tau are the quantiles at which we would like to fit the model. Let us fit a quantile regression model to the Engels data.

Type `f=rq((engel$foodexp)~(engel$income),tau=c(0.05,0.1,0.25,0.5,0.75,0.9,0.95)).`

The results are saved into object `f`. Type `f` and press ENTER.

This will give you estimates of the intercept and slope at the different quantiles.

To obtain confidence intervals for the quantile regression parameters, type `summary(f)` and press ENTER

To obtain standard errors of the quantile regression parameters type
`summary (rq(engel$foodexp~engel$income,tau=c(0.05,0.1,0.25,0.5,0.75,0.9,0.95) ),se= "iid")`. Note that different methods for computing standard errors of the quantile regression parameters are available.

7 Comparing least squares to quantile regression

To visually see the differences between the least squares fit, the median fit and the quantile fits, copy and paste the following commands into the R console.

```
data(engel)
attach(engel)
plot(income,foodexp,xlab="Household Income",ylab="Food Expenditure",type = "n", cex=.5)
points(income,foodexp,cex=.5,col="blue")
taus=c(.05,.1,.25,.5,.75,.9,.95)
xx = seq(min(income),max(income),100)
f = coef(rq((foodexp)~ (income),tau=taus))
yy = cbind(1,xx)%*%f
for(i in 1:length(taus)){
lines(xx,yy[,i],col = "gray")
}
abline(lm(foodexp ~  income),col="red",lty = 2)
abline(rq(foodexp ~  income), col="blue")
legend(3000,500,c("mean (OLS) fit", "median (LAE) fit"),
col = c("red","blue"),lty = c(2,1))
```

8 M-quantile Regression in R

In this part of the workshop we will illustrate how to use R code for M-quantile regression. In the R console type

```
source("M-quantile regression_1.345.R")
```

Type `results.coefficients` and Press ENTER.

Type `results.sterrors` and Press ENTER.

Study and compare the point and variance estimates.

We will now compare M-quantile regression to the quantile regression by using the Engel data. For facilitating the visual comparison we are focusing only at $q = 0.25, 0.5, 0.75$. Copy and paste the following commands into the R console.

```
data(engel)
attach(engel)
plot(income,foodexp,xlab="Household Income",ylab="Food Expenditure",type = "n ", cex=.5)
points(income,foodexp,cex=.5,col="blue")
taus=c(.25,.5,.75)
xx = seq(min(income),max(income),100)
f = coef(rq((foodexp)~ (income),tau=taus))
yy = cbind(1,xx)%*%f
MQmodel=QRLM(cbind(1,income),foodexp,maxit=100,q=c(taus),k=1.345)
f.mq=coef(MQmodel)
yy1 = cbind(1,xx)%*%f.mq
for(i in 1:length(taus)){
lines(xx,yy[,i],col = "red")
lines(xx,yy1[,i],col = "green") }
abline(rq(foodexp ~  income), col="blue")
legend(3000,500,c("mean (OLS) fit"),
col = c("blue"),lty = c(2,1))
```

Compare the fitted regression lines.

Repeat the above analysis this time replacing in the code above

`MQmodel=QRLM(cbind(1,income),foodexp,maxit=100,q=c(taus),k=1.345)` with

`MQmodel=QRLM(cbind(1,income),foodexp,maxit=100,q=c(taus),k=100)`

Compare the fitted regression lines. What do you observe?

You can play with different values of c for exploring how the fitted M-quantile lines move closer to or further from the corresponding quantile lines.

9 Quantile-type Regression for Discrete Outcomes

In this part of the workshop we will illustrate how to use R code for fitting a quantile-type regression models for count and binary outcomes with R. In particular, we will explore the use of Quantile, M-quantile and expectile-type regression.

9.1 Quantile Regression for Counts

We first consider quantile regression for count outcomes. For doing this one can use function lqmm.counts which is part of the R package `lqmm` (Geraci 2014). To start with you must download and install package `lqmm`. Do this by following the instructions in Section 4 of this workshop. The general structure of `lqm.counts` is as follows

`lqm.counts(`$y \sim x$`, tau, data, M)`, where M is the number of jittered datasets we use and tau is the vector of quantiles. Type the following commands into the R console.

```
library(MASS)
library(lqmm)
library(VGAM)
source("glm.mq.poisson.R")
n = 1000
x = runif(n)
test = data.frame(x = x, y = rpois(n,3))
modlqm=lqm.counts(y ~ x, tau = 0.5, data = test, M = 50)
coefq=coef(modlqm)
```

Type `coefq` to see the estimated quantile regression coefficients.

9.2 M-quantile Regression for Counts

For M-quantile regression with counts one can use the function `glm.mq.poisson.R`. The function has been already loaded in the previous subsection. For estimating the M-quantile regression coefficients type the following commands in the R console.

```
modmq= glm.mq.poisson(x = cbind(1,test$x),y = test$y, maxit=100,acc=0.0001,q = 0.5,weights.x=TRUE,
k=1.345)
coefmq=t(modmq$coefficients)
```

Type coefmq to see the estimated M-quantile regression coefficients.

9.3 Expectile Regression for Counts

One way to implement expectile regression is by using function amlpoisson in library `VGAM`. Library `VGAM` has been already loaded in subsection 9.1. Type the following in the R console

```
modexp = vgam(test$y ~ test$x, fam= amlpoisson(w.aml = c(1)))
```

Type `coef(modexp)` to see the estimated expectile regression coefficients

Which percentile is estimated is driven by the weight w.aml. When the weight is equal to 1, the estimates will correspond exactly to the maximum likelihood solution for the Poisson model. You can check this by comparing the estimates in `coef(modexp)` above with those obtained by using the glm function with family set equal to poisson. Smaller values of the w.aml will correspond to percentiles below 0.5 whereas higher values of w.aml will correspond to percentiles above 0.5.

As we saw in the lectures, the estimates obtained by the amlpoisson function will correspond to the estimates obtained by glm.mq.poisson when k in glm.mq.poisson is equal to a large value and w.aml in amlpoisson is equal to $\frac{q}{1-q}$. For example, the expectile regression estimates obtained by setting in the glm.mq.poisson k=100 and q=0.5 should be the same as the regression estimates obtained by setting w.aml= 1 in the amlpoisson.

Finally, by typing modexp@extra, you can find out the percentile that corresponds to the specific value of w.aml used. You can use this for further linking expectile with quantile regression.

10 Quantile Multilevel Regression

In this part of the workshop we will illustrate how to use R code for fitting a quantile multilevel regression. This can be done in R by using function `lqmm` by using maximum likelihood estimation under the Asymmetric Laplace Distribution. M-quantile multilevel regression is also possible. However, there is no R package for doing so. As we mentioned in the lectures, code is available from the presenters upon request. The results from the lqmm are compared to those obtained by fitting a conventional multilevel model in R with function `lme` in `library (nlme)`. The example we use is taken from the lqmm help menu written by Geraci (2014).

The commands below are used to fit a median random intercepts model to synthetically generated data assuming a normal distribution for the random effects. Type the following commands in the R console

```
 library(nlme)
library(lqmm)
set.seed(123)
M = 50
```

```
n = 10
test = data.frame(x = runif(n*M,0,1), group = rep(1:M,each=n))
test$y = 10*test$x + rep(rnorm(M, 0, 2), each = n) + rchisq(n*M, 3)
fit.lqmm = lqmm(fixed = y ~ x, random = ~ 1, group = group,
data = test, tau = 0.5, nK = 11, type = "normal")
```

To see the results type `fit.lqmm`

You can compare the results to the results obtained by using a mean random intercepts model. To do this type the following.

```
fit.lme=lme(y~x,random=~1|group,data=test)
```

To see the results type `fit.lme`. Compare the estimates of the fixed effects obtained by the two models and comment on the differences.