Appendix C

Quantile regression and surroundings using Stata

Introduction

Following the same line of the previous two appendices, we present the Stata features to conduct a complete data analysis, starting from the data loading until the exporting of the results. Also in this case we assume the data are stored in the file *example. {txt, csv, xls, dta, sav}*. The file extension denotes the format of the data file, as follows:

txt tab delimited text file;
csv comma separated values file;
xls spreadsheet (Microsoft Excel) file;
dta Stata file;
sav Spss data file.

The variables stored in the file will be denoted as follows:

y dependent variable;
$x1$, $x2$, $x3$, $x4$ set of explanatory numerical variables;
$z1, z2$ set of explanatory categorical variables.

Throughout the appendix, Stata commands are shown in bold font and comments using regular font. The commands shown are fully explained in the Stata official documentation (Stata 2011a; Stata 2011c). Among the different books dealing with Stata, the books by Acock (2012), Hamilton (2012), and Scott Long (2008) offer a complete description of the use of the software for carrying out a statistical analysis.

Quantile Regression: Theory and Applications, First Edition. Cristina Davino, Marilena Furno and Domenico Vistocco.

Companion website: www.wiley.com/go/quantile_regression

C.1 Loading data

The most common formats for text data are tab delimited files and comma separated values files. The typical extension of the former files is '.txt' while for the latter the '.csv' extension is generally used. We assume the data file is in the current working directory.

```
// to obtain the current working directory
pwd

// to set the working directory by specifying the
// complete path
// in the example the working directory is set to the
// bookQR folder contained in the folder Documents
cd "c:\Documents\bookQR"
```

C.1.1 Text data

The *infile* command allows to import an ASCII file where the columns can be delimited by different characters.

```
// the infile command reads into memory an ASCII file
// in which the values are separated by one or more
// whitespace characters or by commas
// the str# statement precedes the string variable z1
// and z2 the number # denotes the string length
infile y x1 x2 x3 x4 str30 z1 str30 z2 using
       example.txt, clear
// to automatically change all variables to their most
// memory-efficient storage type
compress
describe

// the option automatic allows to manage non-numeric
// variables so to use strings as labels for the
// levels of the categorical variables
infile y x1 x2 x3 x4 z1 z2 using example.txt, automatic
```

An alternative command is the *insheet* command, that allows to specify the column delimiter as well as to manage the name of variables if present on the first row of the file to import.

```
// the insheet command allows to specify the column
// delimiter
// the tab option is used for tab delimited files
insheet y x1 x2 x3 x4 z using example.txt, automatic tab

// the comma option is used for comma separated files
insheet y x1 x2 x3 x4 z using example.txt, automatic
        comma

// the option names is used to denote that the
// variables' names are on the first row
// the variables' list can be omitted
insheet using example.txt, automatic tab names
```

C.1.2 Spreadsheet data

While in the past the official Stata documentation tooltip for reading Excel files consisted of first saving them as comma separated values (.csv) files and then using the *insheet* command, starting from *Stata 12* the *import excel* command allows Excel (.xls and .xlsx) files to be read directly.

```
// the import excel command allows to directly read
// spreadsheets to use
// to read the first sheet of the
// workbook whereas there are not column names on
// the first row
import excel y x1 x2 x3 x4 z example.xls, clear

// the firstrow option allows to indicate that the
// first row contains the variable names
import excel example.xls, firstrow clear

// the sheet option indicates the name of the worksheet
// hosting the data to import
import excel using hospital.xls, sheet(Sheet1) clear

// it is possible to specify a range of cells to import
// using a spreadsheet-like syntax exploiting the
// cellrange option
import excel example.xls, firstrow cellrange(A1:F101)
clear
```

C.1.3 Files from other statistical packages

The *Stat/Transfer* and *DBMS/Copy* softwares offer specialized conversion tools to convert files in different formats to Stata. It is also possible to import files in different formats exploiting an ODBC driver, whereas a driver for the required software is correctly installed and configured.

```
// to load a data stored in a stata format file
use example.dta, clear

// to load a Spss data file
// it does not require any other software installed
// on the PC
usespss example.sav, clear
// convert the Spss format to a stata native format
usespss example.sav, saving(example.dta)

// to import a SAS Xport format file
fdause example.xpt, clear
// to import a SAS data file
// it requires SAS to be installed on the PC
usesas example.ssd, clear
```

C.2 Exploring data

This section shows some of the main commands for exploring data through graphical representation and summary tables. Although in the text we refer to the commands with respect to the dependent variable *y*, they can be useful for any numerical variable. The choice to refer to the dependent variable is due to the typical approach in a regression analysis, where the study of the *y* characteristics can suggest the best model to use.

C.2.1 Graphical tools

The graphical commands shown in this section are detailed in the Stata Graphics documentation (Stata 2011b). An additional useful reading for the different representations available in Stata is the book written by Mitchell (2012), which offers a visual tour of the different Stata graphical tools.

C.2.1.1 Histogram of the dependent variable

```
// to draw histogram of the y variable using frequencies
// on the vertical axis
```

```
histogram y, frequency

// the width option allows to specify the width of
// the bins
histogram y, width(5) frequency

// to draw histogram of the y variable using fractions
// of the data on the vertical axis
histogram y, fraction

// to draw histogram of the y variable using fractions
// of the data on the vertical axis and adding a
// normal curve based on the sample mean and sample
// standard deviation
histogram y, norm fraction
```

C.2.1.2 Conditional histograms of the dependent variable

```
// to draw separate histograms of the y variable for
// each value of the z1 variable, showing the fraction
// of the data on the vertical axis
histogram y, by(z1) frequency

// to draw separate histograms of the y variable for
// each value of the z1 variable along with a total
// histogram for the whole sample
histogram y, by(z1, total) frequency

// to draw separate histograms of the y variable for
// each value of the cartesian product between
// the variables z1 and z2
egen group = group(z1 z2), label
histogram y, by(group) frequency
```

C.2.1.3 Boxplot of the dependent variable

```
// boxplot of the y variable
graph box y

// boxplot of a set of variables
graph box y x1 x2 x3 x4
```

C.2.1.4 Conditional boxplots of the dependent variable

```
// boxplot of the y variable for each values of the
// z1 variable
graph box y, over(z1)

// to draw separate boxplots of the y variable for
// each value of the cartesian product between the z1
// and z2 variables
egen group = group(z1 z2), label
graph box y, by(group) frequency
```

C.2.1.5 Quantile plot of the dependent variable

```
// normal probability plot: quantiles of y vs
// corresponding quantiles of a normal distribution
qnorm y
```

C.2.1.6 Conditional quantile plots of the dependent variable

```
// normal probability plot: quantiles of y vs
// corresponding quantiles of a normal distribution for
// each value of z1 variable
qnorm y, by(z1)
```

C.2.1.7 Scatter plot

```
// to display a scatterplot of x1 vs y
twoway (scatter y x1)
```

C.2.1.8 Conditional scatter plots

```
// to display a scatterplot of x1 vs y
// using colours for the different levels
// of a categorical variable (z1)
// we suppose that z1 has the following
// three levels: "lev1", "lev2", "lev3"
twoway (scatter y x if z1=="lev1",
                mcolor(red) msymbol(circle_hollow))
       (scatter y x if z1=="lev2",
          mcolor(midgreen) msymbol (triangle_hollow))
       (scatter y x if z1=="lev3",
                mcolor(blue) msymbol(plus))
```

```
// using different panels for the levels of a single
// categorical variable
twoway (scatter y x1), by(z1)

// using different panels for the levels of the
// cartesian product between the variables z1 and z2
egen group = group(z1 z2), label
twoway (scatter y x1), by(group)
```

C.2.2 Summary statistics

In this section the commands for obtaining the main summary statistics are reported. In particular, we present univariate summary statistics and synthesis tables for one and two variables.

C.2.2.1 Summary statistics of the dependent variable

```
// summary statistics (means, standard deviations,
// minimum and maximum values, and number of
// observations) for the listed variable
summarize y
// detailed summary statistics, including percentiles,
// median, mean, standard deviation, variance, skewness
// and kurtosis
summarize y, details

// detailed summary statistics for a set of variables
summarize y x1 x2, details

// specified summary statistics for a single variable
tabstat y, stats(mean sd skewness kurtosis)

// location indexes for a single variable
tabstat y, stats(min p5 p25 p50 p75 p95 max)
```

C.2.2.2 Conditional summary statistics of the dependent variable

```
// summary statistics (means, standard deviations,
// minimum and maximum values, and number of observation)
```

```
// for the listed variable within categories of the
// z1 variable
summarize y, by(z1)

// detailed summary statistics, including percentiles,
// median, mean, standard deviation, variance, skewness
// and kurtosis within categories of the z1 variable
summarize y, details by(z1)

// specified summary statistics for a single variable
// within categories of the z1 variable
tabstat y, stats(mean sd skewness kurtosis) by(z1)

// location indexes for a single variable
// within categories of the z1 variable
tabstat y, stats(min p5 p25 p50 p75 p95 max) by(z1)

// location indexes for a single variable
// within categories of the cartesian product
// between the variables z1 and z2
egen group = group(z1 z2), label
tabstat y, stats(mean sd skewness kurtosis) by(group)
```

C.2.2.3 Contingency tables

```
// frequency table for a categorical variable
tabulate z1

// contingency table for two categorical variables
tabulate z1 z2
```

C.3 Modeling data

C.3.1 Ordinary least squares regression analysis

C.3.1.1 Parameter estimates

```
// OLS estimation
regress y x1

// OLS fitted values
```

```
regress y x1
predict yfit

// OLS residuals
regress y x1
predict res, residuals

// regression introducing a categorical variable
// the option gen of the tabulate command generates
// a dummy variable for each level of the variable
tabulate z1, gen(dumZ)
// assuming that the z1 variable has three levels
describe dumZ1-dumZ3
// regression using the generated dummy variables
regress y dumZ1 dumZ2 dumZ3

// multiple regression model
regress y x1 x2 x3 x4
```

C.3.1.2 Main graphical representations

```
// to draw a residual versus fitted plot, using the
// most recent regression
rvfplot
// residual versus fitted plot, adding an horizontal
// line at y=0
rvfplot, yline(0)

// residuals versus the values of the predictor x1
rvpplot x1

// to draw the simple regression line
graph twoway lfit y x ||  scatter y x

// scatterplot of the residuals
twoway (scatter res x)

// component-plus-residual-plot useful for screening
// nonlinearities
cprplot x1

// augmented component-plus-residual-plot useful for
// screening nonlinearities
```

```
acprplot x1

// added-variable plot useful for detecting
// influence points
avplot x1

// combines in one image all the added-variable plots
// from the most recent regress command
avplots

// leverage-versus-squared-residual plot
lvr2plot
```

C.3.1.3 Confidence intervals and hypothesis tests

```
// confidence intervals and hypothesis tests are
// built in results
// however the vector of estimated coefficients and
// the variance covariance matrix can be saved and
// printed by adding the following instructions
// immediately after the regression instruction
reg y x1 x2
matrix coefname=e(b)
matrix list coefname
matrix varname=e(V)
matrix list varname

// there are additional elements that can be saved and
// printed, like the explained sum of squares and the
// residuals sum of squares
scalar modelname=e(mss)
scalar list modelname
scalar residualname=e(rss)
scalar list residualname
```

C.3.2 Quantile regression analysis

C.3.2.1 Parameter estimates

```
// QR estimation: inside parenthesis is the
// selected quantile (in this case the only median)
qreg y x, q(.5)

// variance covariance matrix
```

```
matrix var=e(V)
matrix list var

// QR fitted values
predict qfit

// QR residuals
predict qres

// QR estimation for more quantiles
qreg y x, q(.10 .25 .5 .75 .9)
```

C.3.2.2 Main graphical representations

In order to obtain a graphical representation of the QR coefficients, it is possible to exploit the *GRQREG* module (Azevedo 2011).

```
// to install the grqreg module
ssc install grqreg
// after the installation, the grqreg command allows
// to plot the QR coefficients
// it works after the commands: qreg, bsqreg, sqreg
// it has the option to graph the confidence interval,
// the OLS coefficient and the OLS confidence interval
// on the same graph

// QR estimation
qreg y x
// QR coefficient plot for the slope
// by default the graph for all the estimated
// coefficients except the intercept are produced
grqreg
// QR coefficient plot for the intercept
grqreg, cons

// to set the minimum and maximum values, and the
// steps for the quantiles
// minimum (qmin) default = .05
// maximum (qmax) default = .95
// increment (qstep) default = .05
gqreg y x, qmin = .01 qmax=.99 qstep=.01
// to draw the QR confidence intervals
gqreg, ci level=0.05

// to draw the OLS line, the OLS confidence intervals
// along with their QR counterpart
gqreg, ols olsci ci level=0.05
```

In the case of a multiple QR, the *gqreg* command allows to specify the coefficients to plot through the *varlist* option. It supports also the comparison of two coefficients exploiting the *compare* option.

```
// QR multiple regression
qreg y x1 x2 x3 x4

// graph for all the estimated coefficients
// except the intercept
grqreg

// graph for all the estimated coefficients
// along with the intercept
grqreg, cons

// graph for the specified coefficients
// along with the intercept
grqreg x1 x4, cons

// the compare option graphs two QR coefficients
// it supports two variables at the time: the first
// variable is plotted on the y-axis and the second
// variable on the x-axis
grqreg  x1 x2, compare
```

```
// scatter plot of the residuals
twoway (scatter qres x)

// scatter plot of the data together with OLS and QR
// estimated regressions
// yfit are the OLS fitted values and qfit are the
// QR fitted values
// these values are computed by the OLS and the QR
// regressions
twoway (scatter y x) (line yfit x) (line qfit x)
```

C.3.2.3 Bootstrap estimates

```
// to compute 100 replicates of 25-th quantile
// regression coefficients, and save the estimates to
// the file outBootstrap
bootstrap, qreg y x1, q(.25) _b, reps(100)
sa(outBootstrap)
```

C.3.2.4 Confidence intervals and hypothesis tests

```
// confidence intervals and hypothesis tests are
// built in results
// however the vector of estimated coefficients and the
// variance covariance matrix can be saved and printed
// by adding the following instructions immediately
// after the regression instruction
// these values are computed by the OLS and the QR
// regressions
qreg y x z1, q(.9)
matrix coefnameqr=e(b)
matrix list coefnameqr
matrix varnameqr=e(V)
matrix list varnameqr

// there are additional elements that can be saved and
// printed, like the height of the error density at
// the selected quantile and the estimated objective
// function
// the error density at the specified quantile is
// relevant to compute sparsity and scale
scalar densityname=e(f_r)
scalar list densityname
scalar objectivename=e(sum_adev)
scalar list objectivename

// to test more than one exclusion restriction
// unconstrained median regression
qreg y x1 x2 x3 x4, q(.5)

// to verify the null H0:b2=b3=0
test x2 x3
// to verify the null H0:b2=b3=b4=0
test x2 x3 x4

// test for QR heteroskedasticity
// for five conditional quantiles
sqreg y x1, q(.10 .25 .5 .75 .9)

// test for the equality of two slopes: 10-th conditional
// quantile vs 90-th conditional quantile
test [q10]x1 = [q90]x1
// it is possible to test the equality of two slopes
// also exploiting the lincom command
```

```
lincom [q90]x1-[q10]x1

// the test of equality between two slopes can also be
// directly obtained using the iqreg command
iqreg y x1, q(.10, .90)

// use the option accum of the command test
// to test the equality of more than two slopes
// together
sqreg y x1, q(.10 .25 .5 .75 .9)
test [q25]x1 = [q10]x1
test [q5]x1 = [q25]x1, accum
test [q75]x1 = [q5]x1, accum
test [q9]x1 = [q75]x1, accum
```

C.4 Exporting figures and tables

C.4.1 Exporting figures

```
// to save a graph in stata graphic format
graph save graph1.gph

// to export a graph in pdf format
graph export graph1.pdf, as(pdf)
//NOTE: main available format
//  ps: Adobe PostScript file
// eps: Encapsuled PostScript
// wmf: Windows Metafile
// emf: Windows Enhanced Metafile
// png: PNG bitmap file
// tif: TIFF file
```

C.4.2 Exporting tables

```
// to compute the 10-th conditional quantile
qreg y x z1, q(.1)

// to save in tablename coefficients and standard errors
outreg using tablename, se ct(.10) replace
//se=standard error, ct=column name in the table
outreg using tablename, se ct(OLS) append
```

```
// NOTE 1:
// the output file is tab-delimited and has the
// extension .out
// open the file in Word, highlight the rows of
// results, click on Insert, Table, Convert
// text to table.

// NOTE 2:
// the outreg2 command is an extension of the outreg
// command in order to enhance its capabilities with
// multiple models and to provide more format options
```

Among the other available commands for exporting tables, the following are worth mentioning:

xmlsave allows to export data to Microsoft Excel exploiting an XML format;

tabout produces publication–quality cross tabulations, allowing to export the table both in tab-delimited and in HTML format;

xml_tab converts stored estimates in Excel's XML format;

logout captures the results that are printed to the Results Window (log) and writes them to Excel, Word, or other formats.

References

Acock AC 2012 *A Gentle Introduction to Stata*. Stata Press.

Azevedo JP 2011 *GRQREG: Stata Module to Graph the Coefficients of a Quantile Regression*. http://econpapers.repec.org/software/bocbocode/s437001.htm (accessed April 2013).

Hamilton LW 2012 *Statistics with Stata: Version 12*. Duxbury Press.

Mitchell N 2012 *A Visual Guide to Stata Graphics*. Stata Press.

Scott Long J 2008 *The Workflow of Data Analysis with Stata*. Stata Press.

Stata 2011a *Stata 12. Base Reference Manual (4 volumes)*. Stata Press.

Stata 2011b *Stata 12. Graphics Reference Manual*. Stata Press.

Stata 2011c *Stata 12. User's Guide*. Stata Press.