QUANTILE REGRESSION IN R: A VIGNETTE

ROGER KOENKER

Abstract. Quantile regression is an evolving body of statistical methods for estimating and drawing inferences about conditional quantile functions. An implementation of these methods in the R language is available in the package `quantreg`. This vignette offers a brief tutorial introduction to the package. R and the package `quantreg` are open-source software projects and can be freely downloaded from CRAN: `http://cran.r-project.org`.

1. Introduction

Beran's (2003) provocative definition of statistics as "the study of algorithms for data analysis" elevates computational considerations to the forefront of the field. It is apparent that the evolutionary success of statistical methods is to a significant degree determined by considerations of computational convenience. As a result, design and dissemination of statistical software has become an integral part of statistical research. Algorithms are no longer the exclusive purview of the numerical analyst, or the proto-industrial software firm; they are an essential part of the artisanal research process. Fortunately, modern computing has also transformed the software development process and greatly facilitated collaborative research; the massive collective international effort represented by the R project exceeds the most idealistic Marxist imagination.

Algorithms have been a crucial part of the research challenge of quantile regression methods from their inception in the 18th century. Stigler (1984) describes an amusing episode in 1760 in which the itinerant Croatian Jesuit Rudjer Boscovich sought computational advice in London regarding his nascent method for median regression. Ironically, a fully satisfactory answer to Boscovich's questions only emerged with dawn of modern computing. The discovery of the simplex method and subsequent developments in linear programming have made quantile regression methods competitive with traditional least squares methods in terms of their computational effort. These computational developments have also played a critical role in encouraging a deeper appreciation of the statistical advantages of these methods.

Since the early 1980's I have been developing software for quantile regression: initially for the S language of Chambers and Becker (1984), later for its commercial manifestation Splus, and since 1999 for the splendid open source dialect R , initiated by Ihaka and Gentleman (1996) and sustained by the R Development Core Team (2003). Although there is now some functionality for quantile regression in most of the major commercial statistical packages, I have a natural predilection for the R environment and the software that I have developed for R . In what follows, I have tried to provide a brief tutorial introduction to this environment for quantile regression.

Version: April 2, 2007.

2. What is a vignette?

This document was written in the Sweave format of Leisch (2003). Sweave is an implementation designed for R of the literate programming style advocated by Knuth (1992). The format permits a natural interplay between code written in R , the output of that code, and commentary on the code. Sweave documents are preprocessed by R to produce a LaTeX document that may then be processed by conventional methods. Many R packages now have Sweave vignettes describing their basic functionality. Examples of vignettes can be found for many of the R packages including this one for the `quantreg` packages in the source distribution directory `inst/doc`.

3. Getting Started

I will not attempt to provide another introduction to R . There are already several excellent resources intended to accomplish this task. The books of Dalgaard (2002) and Venables and Ripley (2002) are particularly recommended. The CRAN website link to contributed documentation also offers excellent introductions in several languages.

R is a open source software project and can be freely downloaded from the CRAN website along with its associated documentation. For unix-based operating systems it is usual to download and build R from source, but binary versions are available for most computing platforms and can be easily installed. Once R is running the installation of additional packages is quite straightward. To install the quantile regression package from R one simply types,

```
> install.packages("quantreg")
```

Provided that your machine has a proper internet connection and you have write permission in the appropriate system directories, the installation of the package should proceed automatically. Once the `quantreg` package is installed, it needs to be made accessible to the current R session by the command,

```
> library(quantreg)
```

These procedures provide access to an enormous variety of specialized `packages` for statistical analysis. As we proceed a variety of other packages will be called upon.

Online help facilities are available in two modalities. If you know precisely what you are looking for, and would simply like to check the details of a particular command you can, for example, try:

```
> help(package = "quantreg")
> help(rq)
```

The former command gives a brief summary of the available commands in the package, and the latter requests more detailed information about a specific command. An convenient shorthand for the latter command is to type simply `?rq`. More generally one can initiate a web-browser help session with the command,

```
> help.start()
```

and navigate as desired. The browser approach is better adapted to exploratory inquiries, while the command line approach is better suited to confirmatory ones.

A valuable feature of R help files is that the examples used to illustrate commands are executable, so they can be pasted into an R session, or run as a group with a command like,

```
> example(rq)
```

The examples for the basic `rq` command include an analysis of the Brownlee stackloss data: first the median regression, then the first quantile regression is computed, then the full quantile regression process. A curious feature of this often analysed data set, but one that is very difficult to find without quantile regresion fitting, is the fact the 8 of the 21 points fall exactly on a hyperplane in 4-space.

The second example in the `rq` helpfile computes a weighted univariate median using randomly generated data. The original Engel (1857) data on the relationship between food expenditure and household income is considered in the third example. The data is plotted and then six fitted quantile regression lines are superimposed on the scatterplot. The final example illustrates the imposition of inequality constraints on the quantile regression coefficients using a simulated data set.

Let's consider the median regression results for the Engel example in somewhat more detail. Executing,

```
> data(engel)
> fit1 <- rq(foodexp ~ income, tau = 0.5, data = engel)
```

assigns the output of the median regression computation to the object `fit1`. In the command `rq()` there are also many options. The first argument is a "formula" that specifies the model that is desired. In this case we wanted to fit a simple bivariate linear model so the formula is just `y` $\sim$ `x`, if we had two covariates we could say, e.g. `y` $\sim$ `x+z`. Factor variables, that is variables taking only a few discrete values, are treated specially by the formula processing and result in a group of indicator (dummy) variables.

If we would like to see a concise summary of the result we can simply type,

```
> fit1
Call:
rq(formula = foodexp ~ income, tau = 0.5, data = engel)

Coefficients:
(Intercept)      income
 81.4822474   0.5601806

Degrees of freedom: 235 total; 233 residual
```

By convention for all the R linear model fitting routines, we see only the estimated coefficients and some information about the model being estimated. To obtain a more detailed evaluation of the fitted model, we can use,

```
> summary(fit1)
Call: rq(formula = foodexp ~ income, tau = 0.5, data = engel)

tau: [1] 0.5

Coefficients:
            coefficients lower bd  upper bd
```

```
(Intercept)  81.48225     53.25915 114.01156
income        0.56018      0.48702   0.60199
```

The resulting table gives the estimated intercept and slope in the first column and confidence intervals for these parameters in the second and third columns. By default, these confidence intervals are computed by the rank inversion method described in Koenker (2005), Section 3.4.5. To extract the residuals or the coefficients of the fitted relationship we can write,

```
> r1 <- resid(fit1)
> c1 <- coef(fit1)
```

They can then be easily used in subsequent calculations.

4. Object Orientation

A brief digression on the role of object orientation in R is perhaps worthwhile at this juncture. Expressions in R manipulate objects, objects may be data in the form of vectors, matrices or higher order arrays, but objects may also be functions, or more complex collections of objects. Objects have a class and this clsss identifier helps to recognize their special features and enables functions to act on them appropriately. Thus, for example, the function `summary` when operating on an object of class `rq` as produced by the function `rq` can act quite differently on the object than it would if the object were of another class, say `lm` indicating that it was the product of least squares fitting. Summary of a data structure like a matrix or data.frame would have yet another intent and outcome. In the earlier dialects of S and R methods for various classes were distinguished by appending the class name to the method separated by a ".". Thus, the function `summary.rq` would summarize an `rq` object, and `summary.lm` would summarize an `lm` object. In either case the main objective of the summary is to produce some inferential evidence to accompany the point estimates of parameters. Likewise, plotting of various classes of R objects can be carried out by the expression `plot(x)` with the expectation that the plot command will recognize the class of the object `x` and proceed accordingly. More recently, Chambers (1998) has introduced an elegant elaboration of the class, method-dispatch framework for S and R .

Assignment of objects is usually accomplished by the operator `<-`, and once assigned these new objects are available for the duration of the R session, or until they are explicitly removed from the session. R is an open source language so *all* of the source files describing the functionality of the language are ultimately accessible to the individual user, and users are free to modify and extend the functionality of the language in any way they see fit. To accomplish this one needs to be able to find functions and modify them. This takes us somewhat beyond the intended tutorial scope of this vignette, however suffice it to say that most of the functions of the `quantreg` package you will find used below, can be viewed by simple typing the name of the function perhaps concatenated with a class name.

5. Formal Inference

There are several alternative methods of conducting inference about quantile regression coefficients. As an alternative to the rank-inversion confidence intervals, one can obtain a more conventional looking table of coefficients, standard errors, t-statistics, and p-values using the `summary` function:

```
> summary(fit1, se = "nid")
Call: rq(formula = foodexp ~ income, tau = 0.5, data = engel)

tau: [1] 0.5

Coefficients:
            Value    Std. Error t value  Pr(>|t|)
(Intercept) 81.48225 19.25066    4.23270  0.00003
income       0.56018  0.02828   19.81032  0.00000
```

The standard errors reported in this table are computed as described in Section 3.2.3 for the quantile regression sandwich formula, and using the Hall-Sheather bandwidth rule. To obtain the Powell kernel version of the covariance matrix estimate, one specifies the option `se="ker"` in the `summary` command. It is also possible to control the bandwidths employed with the `bandwidth` option. Another option available in `summary.rq` is to compute bootstrapped standard errors. This is accomplished by specifying the option `se="boot"`. There are currently three flavors of the bootstrap available: the standard xy-pair bootstrap, the Parzen, Wei, and Ying (1994) version, and the Markov chain marginal bootstrap of He and Hu (2002) and Kocherginsky, He, and Mu (2004). There is also the ability to specify m out n versions of the bootstrap in which the sample size of the bootstrap samples is different from (typically smaller than) the original sample size. This "subsampling" approach has a number of advantages, not the least of which is that it can be considerably faster than the full n out of n version. By default `summary` also produces components estimating the full covariance matrix of the estimated parameters and its constituent pieces. For further details, see the documentation for `summary.rq`. In the case of the bootstrap methods the full matrix of bootstrap replications is also available.

There are several options to the basic fitting routine `rq`. An important option that controls the choice of the algorithm used in the fitting is `method`. The default is `method = "br"` which invokes a variant of the Barrodale and Roberts (1974) simplex algorithm described in Koenker and d'Orey (1987). For problems with more than a few thousand observations it is worthwhile considering `method = "fn"` which invokes the Frisch-Newton algorithm described in Portnoy and Koenker (1997). Rather than traversing around the exterior of the constraint set like the simplex method, the interior point approach embodied in the Frisch-Newton algorithm burrows from within the constraint set toward the exterior. Instead of taking steepest descent steps at each intersection of exterior edges, it takes Newton steps based on a log-barrier Lagrangian form of the objective function. Special forms of Frisch-Newton are available for problems that include linear inequality constraints and for problems with sparse design matrices. For extremely large problems with plausibly exchangeable observations `method = "pfn"` implements a version of the Frisch-Newton algorithm with a preprocessing step that can further speed things up considerably.

In problems of moderate size where the default simplex option is quite practical, the parametric programming approach to finding the rank inversion confidence intervals can be rather slow. In such cases it may be advantageous to try one of the other inference methods based on estimation of the asymptotic covariance matrix, or to consider the bootstrap. Both approaches are described in more detail below.

To provide a somewhat more elaborate visualization of the Engel example consider an example that superimposes several estimated conditional quantile functions on the Engel data scatterplot. In the resulting figure the median regression line appears as a solid (blue) line, and the least squares line as a dashed (red) line. The other quantile regression lines appear in grey. Note that the plotting of the fitted lines is easily accomplished by the convention that the command `abline` looks for a pair of coefficients, which if found are treated as the slope and intercept of the plotted line. There are many options that can be used to further fine tune the plot. Looping over the quantiles is also conveniently handled by R 's `for` syntax.

Often it is useful to compute quantile regressions on a discrete set of τ's; this can be accomplished by specifying `tau` as a vector in `rq`:

```
> xx <- income - mean(income)
> fit1 <- summary(rq(foodexp ~ xx, tau = 2:98/100))
> fit2 <- summary(rq(foodexp ~ xx, tau = c(0.05,
+     0.25, 0.5, 0.75, 0.95)))
```

The results can be summarized as a plot.

```
> postscript("engelcoef.ps", horizontal = FALSE,
+     width = 6.5, height = 3.5)
> plot(fit1, nrow = 1, ncol = 2)
> dev.off()
```

or by producing a latex-formatted table.

```
> latex(fit2, caption = "Engel's Law", transpose = TRUE)
```

The `postscript` command preceding the plot tells R that instructions for the plotting should be written in encapsulated postscript (EPSF) format and placed in the file `engelcoef.ps`. Such files are then conveniently included in LaTeX documents, for example. The `dev.off()` command closes the current postscript device and concludes the figure. The horizontal lines in the coefficient plots represent the least squares fit and its associated confidence interval.

In the one-sample setting we know that integrating the quantile function over the entire domain [0,1] yields the mean of the (sample) distribution,

$$\mu = \int_{-\infty}^{\infty} x dF(x) = \int_0^1 F^{-1}(t) dt.$$

Similarly, in the coefficient plots we may expect to see that integrating individual coefficients yields roughly mean effect as estimated by the associated least squares coefficient. One should be cautious, however, about this interpretation in very heterogeneous situations. For the Engel data, note that the least squares intercept is significantly above any of the fitted quantile regression curves in our initial scatter plot. The least squares fit is strongly affected by the two outlying observations with relatively low food expenditure; their attraction tilts the fitted line so its intercept drawn upward. In fact, the intercept for the Engel model is difficult to interpret since it asks us to consider food expenditure for households with zero income. Centering the covariate observations so they have mean zero, as we have done prior to computing `fit1` for the coefficient plot restores a reasonable interpretation of the intercept parameter. After centering the least squares estimate of the intercept is a prediction of mean food expenditure for a household with mean income, and the quantile regression intercept, $\hat{\alpha}(\tau)$ is a prediction of the τth quantile of food

```
> library(quantreg)
> data(engel)
> attach(engel)
> plot(income, foodexp, cex = 0.25, type = "n",
+     xlab = "Household Income", ylab = "Food Expenditure")
> points(income, foodexp, cex = 0.5, col = "blue")
> abline(rq(foodexp ~ income, tau = 0.5), col = "blue")
> abline(lm(foodexp ~ income), lty = 2, col = "red")
> taus <- c(0.05, 0.1, 0.25, 0.75, 0.9, 0.95)
> for (i in 1:length(taus)) {
+     abline(rq(foodexp ~ income, tau = taus[i]),
+         col = "gray")
+ }
```

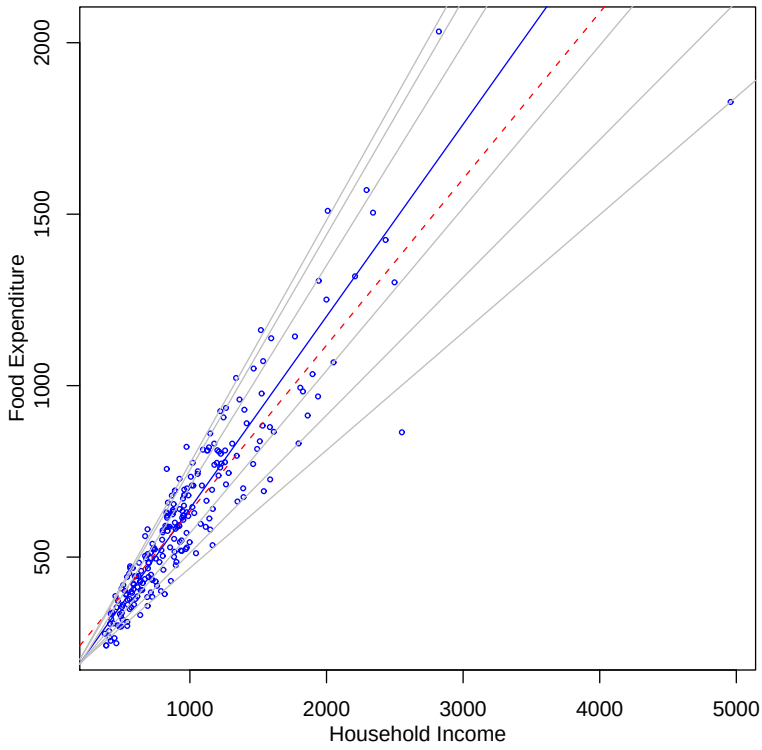

FIGURE 1. Scatterplot and Quantile Regression Fit of the Engel Food Expenditure Data: The plot shows a scatterplot of the Engel data on food expenditure vs household income for a sample of 235 19th century working class Belgian households. Superimposed on the plot are the $\{.05, .1, .25, .75, .90, .95\}$ quantile regression lines in gray, the median fit in solid black, and the least squares estimate of the conditional mean function as the dashed (red) line.

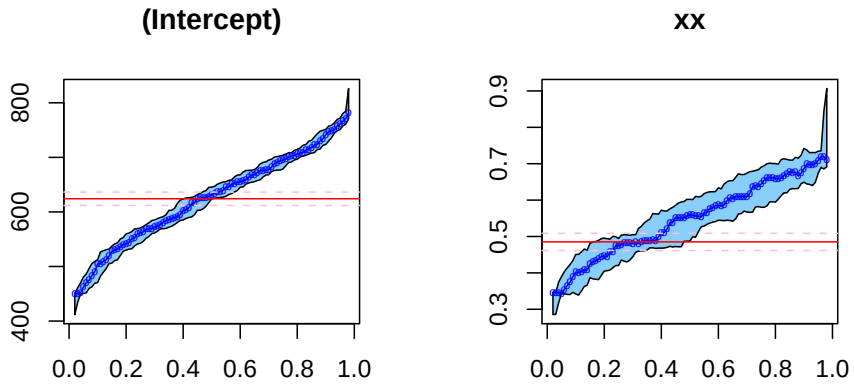

FIGURE 2. Engel Coefficient Plots: the slope and intercept of the estimated linear quantile regression linear for the Engel food expenditure data are plotted as a function of τ. Note that the household income variable has been centered at its mean value for this plot, so the intercept is really a centercept and estimates the quantile function of food expenditure conditional on mean income.

expenditure for households with mean income. In the terminology of Tukey, the "intercept" has become a "centercept."

Quantiles	(Intercept)	xx
0.05	462.223 (450.572,480.503)	0.343 (0.343, 0.390)
0.25	561.277 (542.572,570.726)	0.474 (0.420, 0.494)
0.50	631.845 (623.706,645.461)	0.560 (0.487, 0.602)
0.75	695.123 (678.663,704.105)	0.644 (0.580, 0.690)
0.95	760.745 (751.092,771.916)	0.709 (0.674, 0.734)

TABLE 1. Engel's Law

The `latex` command produces a LaTeX formatted table that can be easily included in documents. In many instances the plotted form of the results will provide a more economical and informative display. It should again be stressed that since the quantile regression functions and indeed all of R is open source, users can always modify the available functions to achieve special effects required for a particular application. When such modifications appear to be of general applicability, it is desirable to communicate them to the package author, so they could be shared with the larger community.

If we want to see *all* the distinct quantile regression solutions for a particular model application we can specify a tau outside the range [0,1], e.g.

```
> z <- rq(foodexp ~ income, tau = -1)
```

This form of the function carries out the parametric programming steps required to find the entire sample path of the quantile regression process. The returned object is of class `rq.process` and has several components: the primal solution in `z$sol`, and the dual solution in `z$dsol`. In interactive mode typing the name of an R object causes the program to print the object in some reasonably intelligible manner determined by the print method designated for the object's class. Again, plotting is often a more informative means of display and so there is a special `plot` method for objects of class `rq.process`.

Estimating the conditional quantile functions of `y` at a specific values of `x` is also quite easy. In the following code we plot the estimated empirical quantile functions of food expenditure for households that are at the 10th percentile of the sample income distribution, and the 90th percentile. In the right panel we plot corresponding density estimates for the two groups. The density estimates employ the adaptive kernel method proposed by Silverman (1986) and implemented in the `quantreg` function `akj`. This function is particularly convenient since it permits unequal mass to be associated with the observations such as those produced by the quantile regression process.

Thus far we have only considered Engel functions that are linear in form, and the scatterplot as well as the formal testing has revealed a strong tendency for the dispersion of food expenditure to increase with household income. This is a particularly common form of heteroscedasticity. If one looks more carefully at the fitting, one sees interesting departures from symmetry that would not be likely to be revealed by the typical textbook testing for heteroscedasticity. One common remedy for symptoms like these would be to reformulate the model in log linear terms. It is interesting to compare what happens after the log transformation with what we have already seen.

Note that the flag `log="xy"` produces a plot with log-log axes, and for convenience of axis labeling these logarithms are base 10, so the subsequent fitting is also specified as base 10 logs for plotting purposes, even though base 10 logarithms are *unnatural* and would never be used in reporting numerical results. This looks much more like a classical iid error regression model, although again some departure from symmetry is visible. An interesting exercise would be to conduct some formal testing for departures from the iid assumption of the type already considered above.

6. More on Testing

Now let's consider some other forms of formal testing. A natural first question is: do the estimated quantile regression relationships conform to the location shift hypothesis that assumes that all of the conditional quantile functions have the same slope parameters. To begin, suppose we just estimate the quartile fits for the Engel data and look at the default output:

```
> fit1 <- rq(foodexp ~ income, tau = 0.25)
> fit2 <- rq(foodexp ~ income, tau = 0.5)
> fit3 <- rq(foodexp ~ income, tau = 0.75)
```

```
> x.poor <- quantile(income, 0.1)
> x.rich <- quantile(income, 0.9)
> ps <- z$sol[1, ]
> qs.poor <- c(c(1, x.poor) %*% z$sol[4:5, ])
> qs.rich <- c(c(1, x.rich) %*% z$sol[4:5, ])
> par(mfrow = c(1, 2))
> plot(c(ps, ps), c(qs.poor, qs.rich), type = "n",
+     xlab = expression(tau), ylab = "quantile")
> plot(stepfun(ps, c(qs.poor[1], qs.poor)), do.points = FALSE,
+     add = TRUE)
> plot(stepfun(ps, c(qs.poor[1], qs.rich)), do.points = FALSE,
+     add = TRUE, col.hor = "gray", col.vert = "gray")
> ps.wts <- (c(0, diff(ps)) + c(diff(ps), 0))/2
> ap <- akj(qs.poor, z = qs.poor, p = ps.wts)
> ar <- akj(qs.rich, z = qs.rich, p = ps.wts)
> plot(c(qs.poor, qs.rich), c(ap$dens, ar$dens),
+     type = "n", xlab = "Food Expenditure", ylab = "Density")
> lines(qs.rich, ar$dens, col = "gray")
> lines(qs.poor, ap$dens, col = "black")
> legend("topright", c("poor", "rich"), lty = c(1,
+     1), col = c("black", "gray"))
```

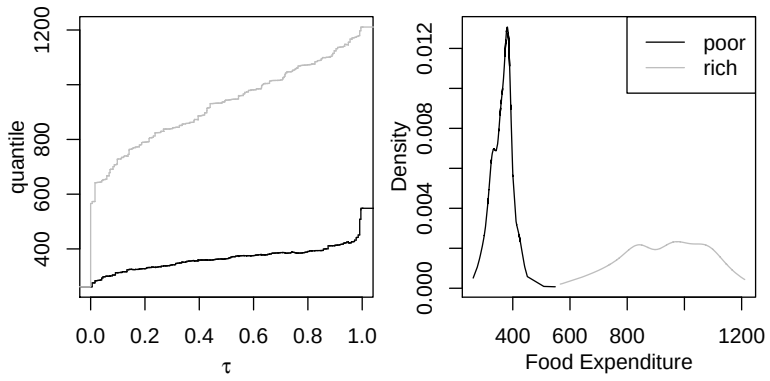

FIGURE 3. Estimated conditional quantile and density functions for food expenditure based on the Engel data: Two estimates are presented one for relatively poor households with income of 504.5 Belgian francs, and the other for relatively affluent households with 1538.99 Belgian francs.

Recall that `rq` just produces coefficient estimates and `summary` is needed to evaluate the precision of the estimates. This is fine for judging whether covariates are significant at particular quantiles but suppose that we wanted to test that the slopes were the same at the three quartiles? This is done with the `anova` command as follows:

```
> anova(fit1, fit2, fit3)
Quantile Regression Analysis of Variance Table
```

```
> plot(income, foodexp, log = "xy", xlab = "Household Income",
+     ylab = "Food Expenditure")
> taus <- c(0.05, 0.1, 0.25, 0.75, 0.9, 0.95)
> abline(rq(log10(foodexp) ~ log10(income), tau = 0.5),
+     col = "blue")
> abline(lm(log10(foodexp) ~ log10(income)), lty = 3,
+     col = "red")
> for (i in 1:length(taus)) {
+     abline(rq(log10(foodexp) ~ log10(income),
+         tau = taus[i]), col = "gray")
+ }
```

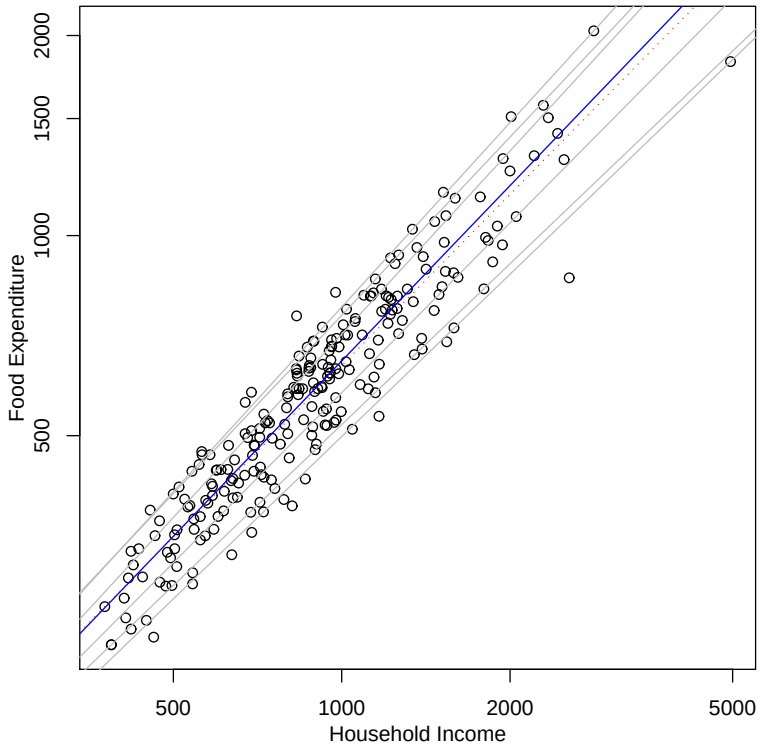

FIGURE 4. Quantile regression estimates for a log-linear version of the Engel food expenditure model.

```
Model: foodexp ~ income
Joint Test of Equality of Slopes: tau in {  0.25 0.5 0.75  }

   Df Resid Df F value    Pr(>F)
1   2      703  15.557 2.449e-07 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

This is an example of a general class of tests proposed in Koenker and Bassett (1982). It may be instructive to look at the code for the command `anova.rq` to

see how this test is carried out. The Wald approach is used and the asymptotic covariance matrix is estimated using the approach of Hendricks and Koenker (1991). It also illustrates a general syntax for testing in R adapted to the present situation.

If you have estimated two models with different covariate specifications, but the same τ then `anova(f0,f1)` tests whether the more restricted model is preferred. Note that this requires that the models with fits say `f0` and `f1` are nested. The procedure `anova.rq` attempts to check whether the fitted models are nested, but this is not foolproof. These tests can be carried out either as Wald tests based on the estimated joint asymptotic covariance matrix of the coefficients, or using the rank test approach described in Gutenbrunner, Jurečková, Koenker, and Portnoy (1993). A variety of other options are described in the documentation of the function `anova.rq`.

7. Inference on the Quantile Regression Process

In least squares estimation of linear models it is implicitly assumed that we are able to model the effects of the covariates as a pure location shift, or somewhat more generally as a location and scale shift of the response distribution. In the simplest case of a single binary treatment effect this amounts to assuming that the treatment and control distributions differ by a location shift, or a location-scale shift. Tests of these hypotheses in the two sample model can be conducted using the conventional two-sample Kolmogorov Smirnov statistic, but the appearance of unknown nuisance parameters greatly complicates the limiting distribution theory. Similar problems persist in the extension of these tests to the general quantile regression setting. Using an approach introduced by Khmaladze (1981), Koenker and Xiao (2002) consider general forms of such tests. The tests can be viewed as a generalization of the simple tests of equality of slopes across quantiles described in the previous section.

In this section we briefly describe how to implement these tests in R . The application considered is the quantile autoregression (QAR) model for weekly U.S. gasoline prices considered in **?**. The data consists of 699 weekly observations running from August 1990 to February, 2004. The model considered is a QAR(4) model utilizing four lags. We are interested in testing whether the classical location-shift AR(4) is a plausible alternative to the QAR(4) specification, that is whether the four QAR lag coefficients are constant with respect to *tau*.

To carry out the test we can either compute the test using the full quantile regression process or on a moderately fine grid of taus:

```
> source("gasprice.R")
> x <- gasprice
> n <- length(x)
> p <- 5
> X <- cbind(x[(p - 1):(n - 1)], x[(p - 2):(n -
+     2)], x[(p - 3):(n - 3)], x[(p - 4):(n - 4)])
> y <- x[p:n]
> T1 <- KhmaladzeTest(y ~ X, taus = -1, nullH = "location")
> T2 <- KhmaladzeTest(y ~ X, taus = 10:290/300,
+     nullH = "location", se = "ker")
taus: 0.03333333  0.03666667  0.04  0.04333333  0.04666667  0.05  0.05333333  0.05666667  0.0
```

When `taus` contains elements outside of the the interval $(0, 1)$ then the process is standardized by a simple form of the covariance matrix that assumes iid error. In the second version of the test Powell's kernel form of the sandwich formula estimate is used, see `summary.rq`. The function `KhmaladzeTest` computes both a joint test that *all* the covariate effects satisfy the null hypothesis, and a coefficient by coefficient version of the test. In this example the former component, `T1$Tn` is 4.8. This test has a 1 percent critical value of 5.56, so the test weakly rejects the null. For the Powell form the standardization the corresponding test statistic is more decisive, taking the value 11.03.

Tests of the location-scale shift form of the null hypothesis can be easily done by making the appropriate change in the `nullH` argument of the function.

8. Nonlinear Quantile Regression

Quantile regression models with response functions that are nonlinear in parameters can be estimated with the function `nlrq`. For such models the specification of the model formula is somewhat more esoteric than for ordinary linear models, but follows the conventions of the R command `nls` for nonlinear least squares estimation.

To illustrate the use of `nlrq` consider the problem of estimating the quantile functions of the Frank copula model introduced in Koenker (2005), Section 8.4. We begin by setting some parameters and generating data from the Frank model:

```
> n <- 200
> df <- 8
> delta <- 8
> set.seed(4003)
> x <- sort(rt(n, df))
> u <- runif(n)
> v <- -log(1 - (1 - exp(-delta))/(1 + exp(-delta *
+     pt(x, df)) * ((1/u) - 1)))/delta
> y <- qt(v, df)
```

We plot the observations, superimpose three conditional quantile functions, and then estimate the same three quantile functions and plot their estimated curves as the dashed curves.

9. Nonparametric Quantile Regression

Nonparametric quantile regression is initially most easily considered within a locally polynomial framework. Locally linear fitting is carried out by the following function:

```
> "lprq" <- function(x, y, h, m = 50, tau = 0.5) {
+     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)
```

```
> plot(x, y, col = "blue", cex = 0.25)
> us <- c(0.25, 0.5, 0.75)
> for (i in 1:length(us)) {
+     u <- us[i]
+     v <- -log(1 - (1 - exp(-delta))/(1 + exp(-delta *
+         pt(x, df)) * ((1/u) - 1)))/delta
+     lines(x, qt(v, df))
+ }
> Dat <- NULL
> Dat$x <- x
> Dat$y <- y
> deltas <- matrix(0, 3, length(us))
> FrankModel <- function(x, delta, mu, sigma, df,
+     tau) {
+     z <- qt(-log(1 - (1 - exp(-delta))/(1 + exp(-delta *
+         pt(x, df)) * ((1/tau) - 1)))/delta, df)
+     mu + sigma * z
+ }
> for (i in 1:length(us)) {
+     tau = us[i]
+     fit <- nlrq(y ~ FrankModel(x, delta, mu, sigma,
+         df = 8, tau = tau), data = Dat, tau = tau,
+         start = list(delta = 5, mu = 0, sigma = 1),
+         trace = TRUE)
+     lines(x, predict(fit, newdata = x), lty = 2,
+         col = "green")
+     deltas[i, ] <- coef(fit)
+ }
```

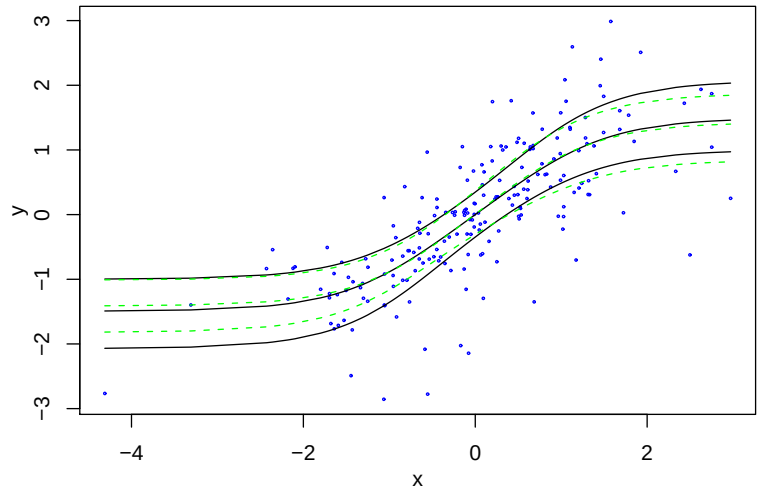

FIGURE 5. Nonlinear Conditional Quantile Estimation of the Frank Copula Model: The solid curves are the true conditional quantile functions and the corresponding estimated curves are indicated by the dashed curves.

```
+         fv[i] <- r$coef[1]
+         dv[i] <- r$coef[2]
+     }
+     list(xx = xx, fv = fv, dv = dv)
+ }
```

If you study the function a bit you will see that it is simply a matter of computing a quantile regression fit at each of m equally spaced x-values distributed over the support of the observed x points. The function value estimates are returned as `fv` and the first derivative estimates at the m points are returned as `dv`. As usual you can specify τ, but now you also need to specify a bandwidth `h`.

Let's begin by exploring the effect of the `h` argument for fitting the motorcycle data.

Fitting derivatives, of course, requires somewhat larger bandwidth and larger sample size to achieve the same precision as function fitting. It is a straightforward exercise to adapt the function `lprq` so that it does locally quadratic rather than locally linear fitting.

Another simple, yet quite general, strategy for nonparametric quantile regression uses regression splines. The function `bs()` in the package `splines` gives a very flexible way to construct B-spline basis expansions. For example you can fit a new motorcycle model like this:

The fitted conditional quantile functions do reasonably well except in the region beyond 50 milliseconds where the data is so sparse that all the quantile curve want to coalesce.

This procedure fits a piecewise cubic polynomial with 15 knots (breakpoints in the third derivative) arranged at quantiles of the x's. (You can also explicitly specify the knot sequence and the order of the spline using the optional arguments to `bs`.) In this instance we have estimated three quartile curves for the B-spline model; a salient feature of this example is the quite dramatic variable in variability over the time scale. There is essentially no variability for the first few milliseconds, but quite substantial variability after the crash. One advantage of the B-spline approach is that it is very easy to add a partially linear model component. If there were another covariate, say z, it could be added as a parametric component using the usual `formula` syntax,

```
> fit <- rq(y bs(x,df=5)+z,tau=.33)
```

Another appealing approach to nonparametric smoothing involves penalty methods. Koenker, Ng, and Portnoy (1994) describe total variation penalty methods for fitting univariate functions; Koenker and Mizera (2004), extend to approach to bivariate function estimation. Again, partially linear models are easily adapted, and there are also easy ways to impose monotonicity and convexity on the fitted functions. In some applications it is desirable to consider models that involve nonparametric fitting with several covariates. A tractable approach that has been explored by many authors is to build additive models. This approach is available in R for least squares fitting of smoothing spline components in the `mgcv` and `gss` packages. A prototype package for quantile regression models of this type is available using the `rqss` function of the `quantreg` package.

The function `rqss` offers a formula interface to nonparametric quantile regression fitting with total variation roughness penalties. Consider the running speed

```
> library(MASS)
> data(mcycle)
> attach(mcycle)
> plot(times, accel, xlab = "milliseconds", ylab = "acceleration")
> hs <- c(1, 2, 3, 4)
> for (i in hs) {
+     h = hs[i]
+     fit <- lprq(times, accel, h = h, tau = 0.5)
+     lines(fit$xx, fit$fv, lty = i)
+ }
> legend(45, -70, c("h=1", "h=2", "h=3", "h=4"),
+     lty = 1:length(hs))
```

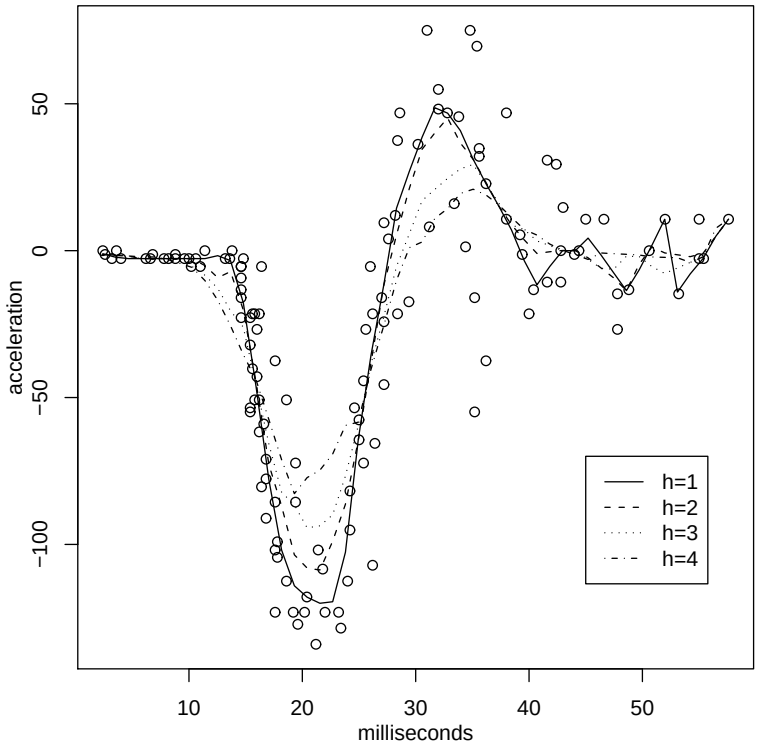

FIGURE 6. Estimation of a locally linear median regression model for the motorcycle data: four distinct bandwidths

of mammals example from Chapter 7. The objective is to estimate a model for the upper envelope of the scatterplot, a model that would reflect best evolutionary practice in mammalian ambulatory efficiency. In contrast to the least squares analysis of Chappell (1989) where they are omitted, no special allowance is made for the "specials" indicated by the plotting character `s` or "hoppers" indicated by `h`. The data is plotted on a (natural) log scale, the model is fit using $\lambda = 1$ as the penalty parameter, and the fitted curve is plotted using a special plotting function that understands the structure of the objects returned from `rqss`. The estimated

```
> library(splines)
> plot(times, accel, xlab = "milliseconds", ylab = "acceleration",
+     type = "n")
> points(times, accel, cex = 0.75)
> X <- model.matrix(accel ~ bs(times, df = 15))
> for (tau in 1:3/4) {
+     fit <- rq(accel ~ bs(times, df = 15), tau = tau,
+         data = mcycle)
+     accel.fit <- X %*% fit$coef
+     lines(times, accel.fit)
+ }
```

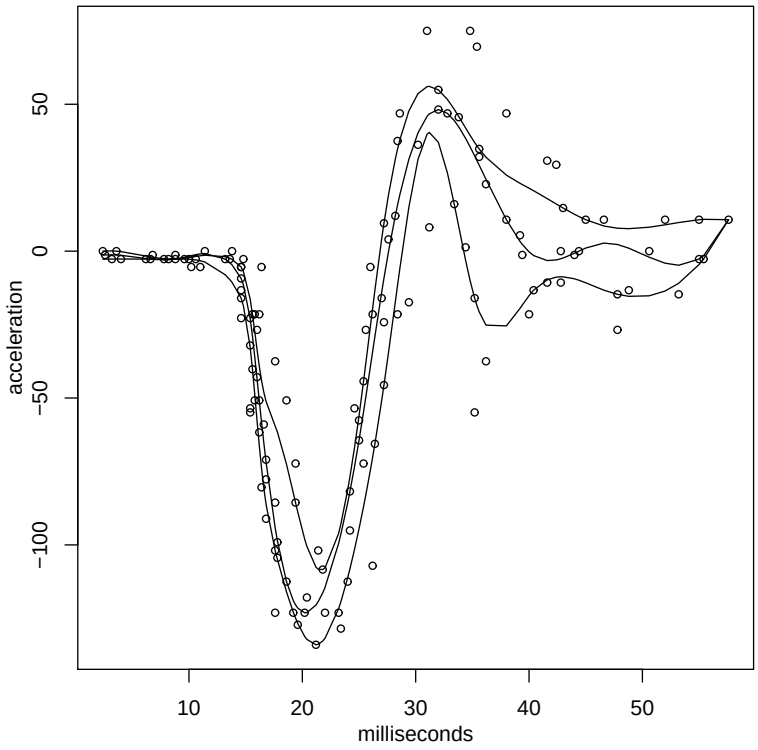

Figure 7. B-spline estimates of the three conditional quartile functions of the motorcycle data

turning point of the piecewise linear fitted function occurs at a weight of about 40 Kg.

```
> data(Mammals)
> attach(Mammals)
```

Bivariate nonparametric fitting using the triogram methods described in Chapter 7 can be handled in a similar manner. If we consider the Cobar mining data from Green and Silverman (1994)

The `qss` term in this case case requires both x and y components. In addition one needs to specify a smoothing parameter λ, and the parameter `ndum` may be used

```
> x <- log(weight)
> y <- log(speed)
> plot(x, y, xlab = "Weight in log(Kg)", ylab = "Speed in log(Km/hour)",
+     type = "n")
> points(x[hoppers], y[hoppers], pch = "h", col = "red")
> points(x[specials], y[specials], pch = "s", col = "blue")
> others <- (!hoppers & !specials)
> points(x[others], y[others], col = "black", cex = 0.75)
> fit <- rqss(y ~ qss(x, lambda = 1), tau = 0.9)
> plot(fit)
```

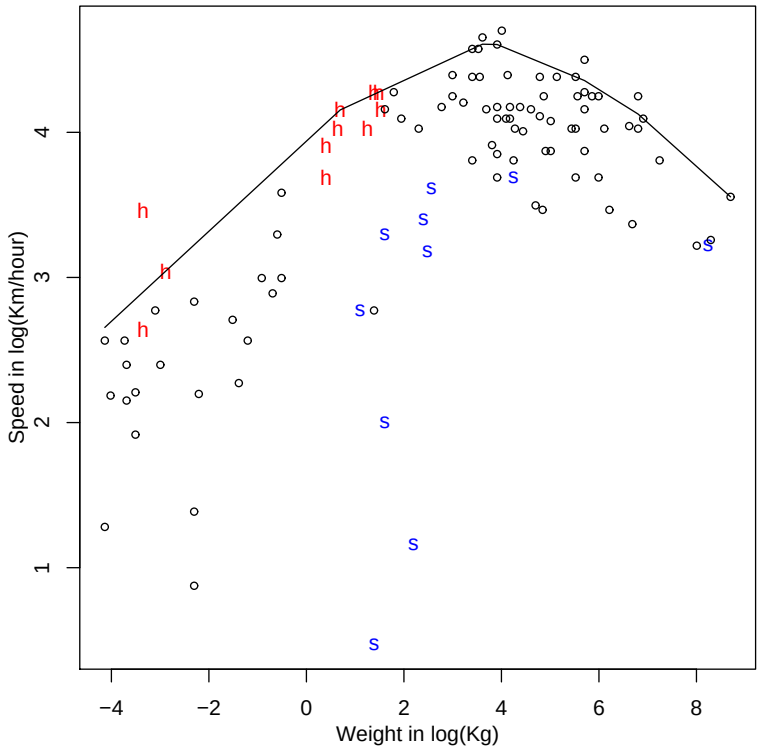

FIGURE 8. Maximal Running Speed of Terrestial Mammals: The figure illustrates the relationship between adult body mass and maximal running speed for 107 species of terrestial mammals. The piecewise linear curve is an estimate of the .90 conditional quantile function estimated subject to a constraint on the total variation of the function gradient.

to specify the number of artificial vertices introduced into the fitting procedure in addition to the actual observations. These artificial vertices contribute to the penalty term, but not to the fidelity.

By default the fit is rendered as a contour plot, but there are also two forms of perspective plots. A conventional R `persp` plot can be obtained by passing option `render = "persp"` to the plot command. More adventurous R gonauts are

```
> data(CobarOre)
> fit <- rqss(z ~ qss(cbind(x, y), lambda = 0.01,
+     ndum = 100), data = CobarOre)
> plot(fit, axes = FALSE, xlab = "", ylab = "")
> rm(list = ls())
```

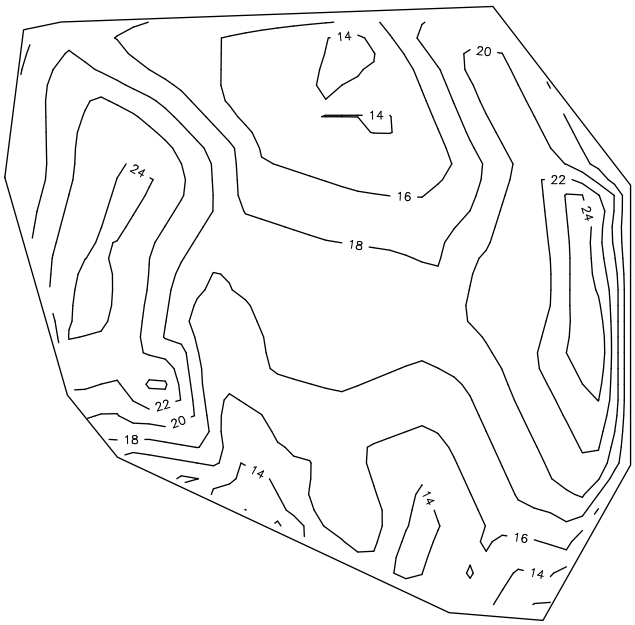

Figure 9. Contour plot of a triogram fit of the Cobar mining data.

encouraged to explore the option `render = "rgl"`, which produces a perspective plot in the dynamic graphics interface to the open GL library provided by the package `rgl`. Of course this package must be installed. A demonstration of how to create animations of `rqss` triogram output using `rgl` is available by running the command `demo(cobar)`.

Another advantage of the penalty approach embodied in `rqss` is that it is straightforward to impose additional qualitative constraints on the fitted functions. Univariate functions can be constrained to be monotone and/or convex or concave. Bivariate functions can be constrained to be either convex or concave. This functionality is implemented by simply imposing nonnegativity constraints on certain linear combinations of model parameters and illustrates one of many possible applications for such constraints. An interesting open research problem involves formal inference on such constraints.

10. Conclusion

A few of the capabilities of the R `quantreg` package have been described. Inevitably, new applications will demand new features and reveal unforseen bugs. In either case I hope that users will send me their comments and suggestions. This document will be periodically updated and the current version will be made available in the R distribution of `quantreg`. See the R command `vignette()` for details on how to find and view vignettes from within R .

References

Barrodale, I., and F. Roberts (1974): "Solution of an overdetermined system of equations in the ℓ_1 norm," *Communications of the ACM*, 17, 319–320.

Beran, R. (2003): "Impact of the Bootstrap on Statistical Algorithms and Theory," *Statistical Science*, pp. 175–184.

Chambers, J. M. (1998): *Programming with Data: A Guide to the S Language*. Springer.

Chappell, R. (1989): "Fitting best lines to data, with applications to allometry," *J. Theor. Biology*, 138, 235–256.

Dalgaard, P. (2002): *Introductory Statistics with R*. Springer-Verlag.

Engel, E. (1857): "Die Produktions- und Konsumptionsverhältnisse des Königreichs Sachsen," *Zeitschrift des Statistischen Bureaus des Königlich Sächsischen Ministeriums des Innern*, 8, 1–54.

Green, P. J., and B. W. Silverman (1994): *Nonparametric Regression and Generalized Linear Models: A Roughness Penalty Approach*. ChapmnHall:Lnd.

Gutenbrunner, C., J. Jurečková, R. Koenker, and S. Portnoy (1993): "Tests of linear hypotheses based on regression rank scores," *J. of Nonparametric Statistics*, 2, 307–33.

He, X., and F. Hu (2002): "Markov chain marginal bootstrap," *J. of Am. Stat. Assoc.*, 97, 783–795.

Hendricks, W., and R. Koenker (1991): "Hierarchical spline models for conditional quantiles and the demand for electricity," *J. of Am. Stat. Assoc.*, 87, 58–68.

Ihaka, R., and R. Gentleman (1996): "R: A Language for Data Analysis and Graphics," *J. of Computation and Graphical Stat.*, 5, 299–314.

Khmaladze, E. V. (1981): "Martingale Approach in the Theory of Goodness-of-fit Tests," *Theory of Prob. and its Apps*, 26, 240–257.

Knuth, D. E. (1992): *Literate Programming*. Center for the Study of Language and Information.

Kocherginsky, M., X. He, and Y. Mu (2004): "Practical Confidence Intervals for Regression Quantiles," *J. of Comp. and Graphical Stat.*, forthcoming.

Koenker, R. (2005): *Quantile Regression*. Cambridge U. Press.

Koenker, R., and G. Bassett (1982): "Robust tests for heteroscedasticity based on regression quantiles," *Econometrica*, 50, 43–61.

Koenker, R., and V. d'Orey (1987): "Computing Regression Quantiles," *Applied Statistics*, 36, 383–393.

Koenker, R., and I. Mizera (2004): "Penalized Triograms: Total Variation Regularization for Bivariate Smoothing," *J. Royal Stat. Soc. (B)*, 66, 145–163.

Koenker, R., P. Ng, and S. Portnoy (1994): "Quantile Smoothing Splines," *Biometrika*, 81, 673–80.

Koenker, R., and Z. Xiao (2002): "Inference on the quantile regression process," *Econometrica*, 70, 1583–1612.

Leisch, F. (2003): "Sweave, Part II: Package Vignettes," *R News*, 3(2), 21–24.

Parzen, M. I., L. Wei, and Z. Ying (1994): "A resampling method based on pivotal estimating functions," *Biometrika*, 81, 341–350.

Portnoy, S., and R. Koenker (1997): "The Gaussian Hare and the Laplacian Tortoise: Computability of squared-error versus absolute-error estimators, with discusssion," *Stat. Science*, 12, 279–300.

R Development Core Team (2003): *R: A language and environment for statistical computing*http://www.R-project.org.

Silverman, B. (1986): *Density Estimation for Statistics and Data Analysis*. Chapman-Hall: New York.

Stigler, S. (1984): "Boscovich, Simpson and a 1760 manuscript note on fitting a linear relation," *Biometrika*, 71, 615–620.

Venables, W., and B. D. Ripley (2002): *Modern applied statistics with S-PLUS*. Springer-Verlag, fourth edn.