R documentation
of all in ‘quantreg/man’

December 15, 2001

R topics documented:

akj .................................................. 1
bandwidth.rq ...................................... 2
barro .............................................. 3
khmaladzize ....................................... 4
lm.fit.recursive ................................. 5
plot.khmal ................................. 5
plot.rq.process ................................. 6
plot.table.rq ................................. 7
print.rq ........................................ 8
print.summary.rq .......................... 8
ranks ........................................... 9
rq ................................................. 10
rq.fit ........................................... 12
rq.fit.br ...................................... 12
rq.fit.fn ...................................... 14
rq.fit.pfn ................................... 15
rq.object ..................................... 16
rq.process.object ............................ 17
rq.test.khmal .................................. 18
rq.wfit ......................................... 19
rrs.test ......................................... 20
standardize .................................. 21
summary.rq ...................................... 22
table.rq ......................................... 23

Index 24

akj Density estimation using adaptive kernel method

Description

univariate adaptive kernel density estimation a la Silverman
Usage

```r
akj(x, z, p, h, alpha, kappa, iker1, iker2)
```

Arguments

- `x`: points used for centers of kernel assumed to be sorted
- `z`: points at which density is calculated; default to `seq( min(x), max(x), 2*length(x) )`
- `p`: vector of probabilities associated with x's; default to `1/len(x)` for each x.
- `h`: initial window size (overall); default to Silverman's normal reference
- `alpha`: a sensitivity parameter that determines the sensitivity of the local bandwidth to variations in the pilot density; default to .5
- `kappa`: constant determining initial (default) window width
- `iker1`: kernel indicator, 0 for normal kernel (default) while 1 for cauchy kernel
- `iker2`:.xxx

Value

- a R structure is returned
  - `dens`: the vector of estimated density
  - `psi`: a vector of $\psi = -f'/f$ function
  - `score`: a vector of score $(f'/f)^2 - f'/f$ function
  - `h`: same as the input argument h

References


---

`bandwidth.rq`  
*bandwidth selection for rq functions*

Description

function to compute bandwidth for sparsity estimation

Usage

```r
bandwidth.rq(p, n, hs=T, alpha=0.05)
```

Arguments

- `p`: quantile(s) of interest
- `n`: sample size
- `hs`: flag for hall-sheather method
- `alpha`: alpha level for intended confidence intervals
Details

If hs=T (default) then the Hall-Sheather(1988) rule $O(n^{-1/3})$ is used, if hs=F then the Bofinger $O(n^{-1/5})$ is used.

Value

returns a vector of bandwidths corresponding to the argument p.

Author(s)

Roger Koenker rkoenker@uiuc.edu

References

Hall and Sheather(1988, JRSS(B)), Bofinger (1975, Aus. J. Stat)

__barro__

<table>
<thead>
<tr>
<th>Barro Data</th>
</tr>
</thead>
</table>

Description

Version of the Barro Growth Data used in Koenker and Machado(1999). This is a regression data set consisting of 161 observations on determinants of cross country GDP growth rates. There are 13 covariates with dimnames corresponding to the original Barro and Lee source. See http://www.nber.org/pub/barro.lee/. The first 71 observations are on the period 1965-75, remainder on 1987-85.

Usage

data(barro)

Format

A data frame containing 161 observations on 14 variables:

[.1] "Annual Change Per Capita GDP"
[.2] "Initial Per Capita GDP"
[.3] "Male Secondary Education"
[.4] "Female Secondary Education"
[.5] "Female Higher Education"
[.6] "Male Higher Education"
[.7] "Life Expectancy"
[.8] "Human Capital"
[.9] "Education/GDP"
[.10] "Investment/GDP"
[.11] "Public Consumption/GDP"
[.12] "Black Market Premium"
[.13] "Political Instability"
[.14] "Growth Rate Terms Trade"
References


khmaladzize

Function to compute Khmaladze Transformation

Description

Function to compute the recursive least squares transformation of the quantile regression process for the rq.test.khmal procedure.

Usage

khmaladzize(tau, atau, Z, location.scale)

Arguments

tau

quantiles specified in the fitted model

atau

xbar’betahat(tau) at these quantiles

Z

full rq process

location.scale

if T do location-scale transformation, if F do location transformation

Details

Uses adaptive kernel density estimation akj() to estimate score functions.

Value

Returns transformed Z process.

Author(s)

R. Koenker

References


See Also

rq.test.khmal
lm.fit.recursive  Recursive Least Squares

Description

This function fits a linear model by recursive least squares. It is a utility routine for the 
rq.test.khmal function of the quantile regression package.

Usage

lm.fit.recursive(X, y, int=T)

Arguments

X  Design Matrix
y  Response Variable
int if T then append intercept to X

Value

return p by n matrix of fitted parameters, where p. The ith column gives the solution up
to "time" i.

Author(s)

R. Koenker

References


See Also

khmaladzize

plot.khmal  Plots Standardized and Khmaladzized Residual Processes

Description

The function makes 6 arrays of p plots based on the object of class "khmal" created by
rq.test.khmal of quantile regression results. The 6 arrays are: (i) estimated coefficients; (ii)
regression of slopes on the intercept; (iii) standardized residuals for the joint; (iv)
standardized residuals for the coef by coef; (v) khmaladzized residuals for the joint, and
(vi) khmaladzized residuals for the coef by coef hypothesis testing.

Usage

plot.khmal(x, nrow= ceiling(length(x$var.list)/2), ncol= 2, plotn = 1:6, bcolor="gray", ... )
plot.rq.process

Arguments

x
output of ‘rq.test.khmal’. plot.khmal() requires the output of ‘rq.test.khmal’.

var.list
numerical list of variables to be plotted. By default all variables are plotted. A restricted set of variables can be specified by providing a numerical vector indicating the desired variables. The convention is that 1 corresponds to the intercept, 2 to the first independent variable entered in "formula" and so on. See example for further details.

nrow
number of rows per page of plots. Automatically set by assuming that the number of columns is 2.

ncol
number of plots per page of plots. Default 2.

plotn
a numerical vector indicating which array of plots will be graphed. By default the 6 arrays described in ‘Description’ are plotted. Useful to produce individual postscript files of each array. For example, specifying plotn = 1 in conjunction will postscript("01.ps") will yield an array of plots of the quantile regression estimated coefficients.

bcolor
color of the confidence band by default "gray".

... other optional arguments passed to ‘plot’.

Value
Generates plots of object of class ‘khmal’. Please refer to "Description" for further details.

References

Examples

data(barro)
fit.Lonly <- rq.test.khmal(y.net ~ lgdp2 + fse2 + gedy2 + Iy2 + gcony2, data = barro, location.scale = FALSE)
par(ask=interactive())
plot(fit.Lonly, var.list=c(2,4))

plot.rq.process plot the coordinates of the quantile regression process

Description
Function to plot quantile regression process.

Usage
plot.rq.process(x, nrow=3, ncol=2, ...)
**plot.table.rq**

Arguments

- **x**: an object produced by `rq()` fitting
- **nrow**: rows in `mfrow`
- **ncol**: columns in `mfrow`
- **...**: optional arguments to `plot`

Author(s)

Roger Koenker rkoenker@uiuc.edu

See Also

- rq

**plot.table.rq**  
Plot Table of Quantile Regression Results

Description

The function makes an array of plots based on an array produced by `table.rq` of quantile regression results. The plots each represent one parameter of the model specified in the formula argument to `table.rq`; the plots consist of the point estimates `betahat(tau)` plotted against the taus specified in the `table.rq` command with a confidence band as produced by `rq()`.

Usage

```r
plot.table.rq(x, nrow=3, ncol=2, ...)
```

Arguments

- **x**: object of class `table.rq` containing the array to be plotted.
- **nrow**: number of rows per page of plots
- **ncol**: number of columns per page of plots
- **...**: optional arguments to `plot`

Details

See `rq()` and `rq.fit.br()` and `table.rq()` for further details on control of data. Obviously, further plotting parameters could be added in a more full-blown version. This version is meant just to be illustrative.

Side Effects

plots an array of figures on the current graphics device.

See Also

- rq, rq.fit.br, table.rq
print.rq  Print an rq object

Description
Print an object generated by rq

Usage
print.rq(x, ...)

Arguments
x  Object returned from rq representing the fit of the model.
...  Optional arguments passed to fitting routines

See Also
rq

print.summary.rq  Print Quantile Regression Summary Object

Description
Print summary of quantile regression object

Usage
print.summary.rq(x, digits=max(5, .Options$digits - 2), ...)

Arguments
x  This is an object of class "summary.rq" produced by a call to summary.rq().
digits  Significant digits reported in the printed table.
...  Optional arguments passed to printing function

See Also
summary.rq
ranks

Quantile Regression Ranks

Description

Function to compute ranks from the dual (regression rankscore) process.

Usage

ranks(v, score="wilcoxon", tau=0.5)

Arguments

v object of class "rq.process" generated by rq()

score The score function desired. Currently implemented score functions are "wilcoxon", "normal", and "sign" which are asymptotically optimal for the logistic, Gaussian and Laplace location shift models respectively. Also implemented are the "tau" which generalizes sign scores to an arbitrary quantile, and "interquartile" which is appropriate for tests of scale shift.

tau the optional value of tau if the "tau" score function is used.

Details

See GJKP(1993) for further details.

Value

The function returns two components. One is the ranks, the other is a scale factor which is the $L_2$ norm of the score function. All score functions should be normalized to have mean zero.

References


See Also

rq, rrs.test

Examples

data(stackloss)
ranks(rq(stack.loss ~ stack.x, tau=-1))
rq

Quantile Regression

Description

Returns an object of class "rq" or "rq.process" that represents a quantile regression fit.

Usage

rq(formula, tau=.5, data, weights, na.action, method="br", contrasts, ...)

Arguments

formula a formula object, with the response on the left of a ~ operator, and the terms, separated by + operators, on the right.

tau the quantile to be estimated, this is generally a number between 0 and 1, but if specified outside this range, it is presumed that the solutions for all values of \( \tau \) in (0,1) are desired. In the former case an object of class "rq" is returned, in the latter, an object of class "rq.process" is returned.

data a data.frame in which to interpret the variables named in the formula, or in the subset and the weights argument. If this is missing, then the variables in the formula should be on the search list. This may also be a single number to handle some special cases – see below for details.

weights vector of observation weights; if supplied, the algorithm fits to minimize the sum of the weights multiplied into the absolute residuals. The length of weights must be the same as the number of observations. The weights must be nonnegative and it is strongly recommended that they be strictly positive, since zero weights are ambiguous.

na.action a function to filter missing data. This is applied to the model.frame after any subset argument has been used. The default (with na.fail) is to create an error if any missing values are found. A possible alternative is na.omit, which deletes observations that contain one or more missing values.

method the algorithmic method used to compute the fit. There are currently three options: The default method is the modified version of the Barrodale and Roberts algorithm for \( l_1 \)-regression, used by l1fit in S, and is described in detail in Koenker and d'Orey(1987, 1994), default = "br". This is quite efficient for problems up to several thousand observations, and may be used to compute the full quantile regression process. It also implements a scheme for computing confidence intervals for the estimated parameters, based on inversion of a rank test described in Koenker(1994). For larger problems it is advantageous to use the Frisch–Newton interior point method "fn". And very large problems one can use the Frisch-Newton approach after preprocessing "pfn". Both of the latter methods are described in detail in Portnoy and Koenker(1997).

contrasts a list giving contrasts for some or all of the factors default = NULL appearing in the model formula. The elements of the list should have the same
name as the variable and should be either a contrast matrix (specifically, any full-rank matrix with as many rows as there are levels in the factor), or else a function to compute such a matrix given the number of levels.

... additional arguments for the fitting routines (see \texttt{rq.fit.br} and \texttt{rq.fit.fn} and the functions they call).

\section*{Value}

See \texttt{rq.object} and \texttt{rq.process.object} for details.

\section*{Method}

The function computes an estimate on the tau-th conditional quantile function of the response, given the covariates, as specified by the formula argument. Like \texttt{lm()}, the function presumes a linear specification for the quantile regression model, i.e. that the formula defines a model that is linear in parameters. For non-linear quantile regression see the function \texttt{nlrq()}. [To appear real soon now on a screen near you.] The function minimizes a weighted sum of absolute residuals that can be formulated as a linear programming problem. As noted above, there are three different algorithms that can be chosen depending on problem size and other characteristics. For moderate sized problems ($n \ll 5,000, p \ll 20$) it is recommended that the default "br" method be used. There are several choices of methods for computing confidence intervals and associated test statistics. Using "br" the default approach produces confidence intervals for each of the estimated model parameters based on inversion of a rank test. See the documentation for \texttt{rq.fit.br} for further details and options. For larger problems, the "fn" and "pfn" are preferred, and there are several methods of computing standard errors and associated test statistics described in the help files for \texttt{rq.fit.fn}, and \texttt{summary.rq}.

\section*{References}


There is also recent information available at the URL: \url{http://www.econ.uiuc.edu}.

\section*{See Also}

\texttt{summary.rq}, \texttt{rq.object}, \texttt{rq.process.object}

\section*{Examples}

data(stackloss)
rq(stack.loss ~ stack.x,.5) \#median (l1) regression fit for the stackloss data. 
rq(stack.loss ~ stack.x,.25) \#the 1st quartile, 
\#note that 8 of the 21 points lie exactly on this plane in 4-space 
rq(stack.loss ~ stack.x, tau=-1) \#this returns the full rq process 
rq(rnorm(50) ~ 1, ci=F) \#ordinary sample median --no rank inversion ci 
rq(rnorm(50) ~ 1, weights=runif(50),ci=F) \#weighted sample median
Function to choose method for Quantile Regression

**Description**

Function to choose method for quantile regression

**Usage**

```r
rq.fit(x, y, tau=0.5, method="br", ...)```

**Arguments**

- **x**: the design matrix
- **y**: the response variable
- **tau**: the quantile desired, if tau lies outside (0,1) the whole process is estimated.
- **method**: method of computation: "br" is Barrodale and Roberts exterior point "fn" is the Frisch-Newton interior point method.
- **...**: Optional arguments passed to fitting routine.

**See Also**

- `rq`
- `rq.fit.br`
- `rq.fit.fn`

Quantile Regression Fitting by Exterior Point Methods

**Description**

This function controls the details of QR fitting by the simplex approach embodied in the algorithm of Koenker and d’Orey based on the median regression algorithm of Barrodale and Roberts. Typically, options controlling the construction of the confidence intervals would be passed via the ...{} argument of `rq()`.

**Usage**

```r
rq.fit.br(x, y, tau=0.5, alpha=0.1, ci=T, iid=T, interp=T, tcrit=T)```

**Arguments**

- **x**: the design matrix
- **y**: the response variable
- **tau**: the quantile desired, if tau lies outside (0,1) the whole process is estimated.
- **alpha**: the nominal coverage probability for the confidence intervals
logical flag if T then compute confidence intervals for the parameters using the rank inversion method of Koenker (1994). See \texttt{rq()} for more details. If F then return only the estimated coefficients. Note that for large problems the default option ci = T can be rather slow. Note also that rank inversion only works for \( p > 1 \), an error message is printed in the case that ci=T and p=1.

logical flag if T then the rank inversion is based on an assumption of iid error model, if F then it is based on an nid error assumption. See Koenker and Machado (1999) for further details on this distinction.

As with typical order statistic type confidence intervals the test statistic is discrete, so it is reasonable to consider intervals that interpolate between values of the parameter just below the specified cutoff and values just above the specified cutoff. If interp = F then the 2 “exact” values above and below on which the interpolation would be based are returned.

Logical flag if T - Student t critical values are used, if F then normal values are used.

If tau lies in \((0,1)\) then an object of class \texttt{"rq"} is returned with various related inference apparatus. If tau lies outside \([0,1]\) then an object of class \texttt{rq.process} is returned. In this case parametric programming methods are used to find all of the solutions to the QR problem for tau in \((0,1)\), the \( p \)-variate resulting process is then returned as the array sol containing the primal solution and dsol containing the dual solution. There are roughly \( O(n \log n) \) distinct solutions, so users should be aware that these arrays may be large and somewhat time consuming to compute for large problems.

Returns an object of class \texttt{"rq"} for tau in \((0,1)\), or else of class \texttt{"rq.process"}. See \texttt{rq.object} and \texttt{rq.process.object} for further details.


\texttt{rq, rq.fit.fn}

data(stackloss)
\texttt{rq.fit.br(stack.x, stack.loss, tau=.73 , interp=F)}
rq.fit.fn

Quantile Regression Fitting via Interior Point Methods

Description

This is a lower level routine called by \texttt{rq()} to compute quantile regression methods using the Frisch-Newton algorithm.

Usage

\texttt{rq.fit.fn(x, y, tau=0.5, int=F, beta=0.99995, eps=1e-06)}

Arguments

\begin{itemize}
\item \texttt{x} The design matrix
\item \texttt{y} The response vector
\item \texttt{tau} The quantile of interest, must lie in (0,1)
\item \texttt{int} logical flag, if \texttt{T} an intercept should be appended to \texttt{x}, if not, not.
\item \texttt{beta} technical step length parameter – alter at your own risk!
\item \texttt{eps} tolerance parameter for convergence. In cases of multiple optimal solutions there may be some discrepancy between solutions produced by method "fn" and method "br". This is due to the fact that "fn" tends to converge to a point near the centroid of the solution set, while "br" stops at a vertex of the set.
\end{itemize}

Details

The details of the algorithm are explained in Koenker and Portnoy (1997). The basic idea can be traced back to the log-barrier methods proposed by Frisch in the 1950’s for constrained optimization. But the current implementation is based on proposals by Mehrotra and others in the recent (explosive) literature on interior point methods for solving linear programming problems. This version of the algorithm is designed for fairly large problems, for very large problems see \texttt{rq.fit.pfn}.

Value

returns an object of class "\texttt{rq}", which can be passed to \texttt{summary.rq} to obtain standard errors, etc.

References


See Also

\texttt{rq, rq.fit.br, rq.fit.pfn}
Description

A preprocessing algorithm for the Frisch Newton algorithm for quantile regression. This is one possible method for rq().

Usage

rq.fit.pfn(x, y, tau=0.5, int=F, Mm.factor=0.8, max.bad.fixup=3, eps=1e-06)

Arguments

x    design matrix usually supplied via rq()
y    response vector usually supplied via rq()
tau quantile of interest
int include intercept?
Mm.factor constant to determine sub sample size m
max.bad.fixup number of allowed mispredicted signs of residuals
eps convergence tolerance

Details

Preprocessing algorithm to reduce the effective sample size for QR problems with (plausibly) iid samples. The preprocessing relies on subsampling of the original data, so situations in which the observations are not plausibly iid, are likely to cause problems. The tolerance eps may be relaxed somewhat.

Value

Returns an object of type rq

Author(s)

Roger Koenker <rkoener@uiuc.edu>

References


See Also

rq
Description

These are objects of class "rq". They represent the fit of a linear conditional quantile function model.

Details

The coefficients, residuals, and effects may be extracted by the generic functions of the same name, rather than by the $ operator. For pure rq objects this is less critical than for some of the inheritor classes. Note that the extractor function coef returns a vector with missing values omitted.

Generation

This class of objects is returned from the rq function to represent a fitted linear quantile regression model.

Methods

The "rq" class of objects has methods for the following generic functions: coef, effects, formula, labels, model.frame, model.matrix, plot, predict, print, print.summary, residuals, summary

Structure

The following components must be included in a legitimate rq object.

coefficients the coefficients of the quantile regression fit. The names of the coefficients are the names of the single-degree-of-freedom effects (the columns of the model matrix). If the model was fitted by method "br" with ci=TRUE, then the coefficient component consists of a matrix whose first column consists of the vector of estimated coefficients and the second and third columns are the lower and upper limits of a confidence interval for the respective coefficients.

residuals the residuals from the fit.

contrasts a list containing sufficient information to construct the contrasts used to fit any factors occurring in the model. The list contains entries that are either matrices or character vectors. When a factor is coded by contrasts, the corresponding contrast matrix is stored in this list. Factors that appear only as dummy variables and variables in the model that are matrices correspond to character vectors in the list. The character vector has the level names for a factor or the column labels for a matrix.

model optionally the model frame, if model=TRUE.

x optionally the model matrix, if x=TRUE.

y optionally the response, if y=TRUE.

See Also

rq, coefficients.
**Description**

These are objects of class `rq.process`. They represent the fit of a linear conditional quantile function model.

**Details**

These arrays are computed by parametric linear programming methods using the exterior point (simplex-type) methods of the Koenker–d’Orey algorithm based on Barrodale and Roberts median regression algorithm.

**Generation**

This class of objects is returned from the `rq` function to represent a fitted linear quantile regression model.

**Methods**

The "rq.process" class of objects has methods for the following generic functions: `effects`, `formula`, `labels`, `model.frame`, `model.matrix`, `plot`, `predict`, `print`, `print.summary`, `summary`

**Structure**

The following components must be included in a legitimate `rq.process` object.

**sol** The primal solution array. This is a (p+3) by J matrix whose first row contains the 'breakpoints' \( \tau_1, \tau_2, \ldots, \tau_J \), of the quantile function, i.e. the values in \([0,1]\) at which the solution changes, row two contains the corresponding quantiles evaluated at the mean design point, i.e. the inner product of \( x_{\text{bar}} \) and \( b(\tau_j) \), the third row contains the value of the objective function evaluated at the corresponding \( \tau_j \), and the last \( p \) rows of the matrix give \( b(\tau_j) \). The solution \( b(\tau_j) \) prevails from \( \tau_j \) to \( \tau_{j+1} \). Portnoy (1991) shows that \( J = O_p(n \log n) \).

**dsol** The dual solution array. This is a \( n \) by \( J \) matrix containing the dual solution corresponding to sol, the \( ij \)-th entry is 1 if \( y_i > x_i b(\tau_j) \), is 0 if \( y_i < x_i b(\tau_j) \), and is between 0 and 1 otherwise, i.e. if the residual is zero. See Gutenbrunner and Jureckova(1991) for a detailed discussion of the statistical interpretation of dsol. The use of dsol in inference is described in Gutenbrunner, Jureckova, Koenker, and Portnoy (1994).

**References**


See Also
rq.

---

**rq.test.khmal**

*Estimates Quantile Regression Model and Test Statistics*

Description

Estimates a model and produces the output necessary to test the location and location-scale shift hypotheses. Returns an object of the class "khmal".

Usage

```r
rq.test.khmal( formula, data, taus=seq(0.2,0.8,by=0.002), location.scale = TRUE, trim = c(0.25, 0.75) )
```

Arguments

- **formula**: a symbolic description of the model to be fit. The details of model specification are given below.
- **data**: dataframe containing the regressand and regressors. It should not contain a column of ones.
- **taus**: vector of quantiles to be estimated. Default range is 0.2 to 0.8, increasing by 0.002 units.
- **location.scale**: a logic value indicating whether the location-scale shift hypothesis (default) or the location shift hypothesis (F) should be tested.
- **trim**: a vector indicating the lower and upper bound of the quantiles to included in the computation of the test statistics (only, not estimates). This might be required due to tail behavior.

Value

A R structure is returned

- **formula**: a symbolic description of the fitted model.
- **taus**: vector with estimated quantiles.
- **Jn**: \( X'X \) of least squares.
- **fit**: Estimated coefficients for all quantiles.
- **Hfit**: Array with inverse quantile regression covariance matrices.
- **vars**: Variables names, useful for plotting functions.
**rq.wfit**

Numerical list of variables to be plotted. By default all variables are plotted. A restricted set of variables can be specified by providing a numerical vector indicating the desired variables. The convention is that 1 corresponds to the intercept, 2 to the first independent variable entered in "formula" and so on. See example for further details.

**location.scale**

a logic value indicating whether the location-scale shift hypothesis (default) or the location shift hypothesis (F) was performed.

**b**

Matrix with the coefficients from the least-squares regression of the estimated quantile regression slopes on the intercept.

**J**

Output of standardize().

**Vtilde**

Output of khmaladzize() for the joint hypothesis testing.

**vtilde**

Output of khmaladzize() for the individual hypotheses testing.

**Tvtilde**

vtilde appropriately transformed to compute the individual test statistics of the location-scale shift hypothesis.

**trim**

a vector indicating the lower and upper bound of the quantiles included in the computation of the test statistics (only, not estimates).

**Kn**

Joint location-scale shift test statistic. Only returned if location.scale is TRUE.

**KHn**

Individual location-scale shift test statistics. Only returned if location.scale is TRUE.

**Tn**

Joint location shift test statistic. Only returned if location.scale is FALSE.

**THn**

Individual location shift test statistics. Only returned if location.scale is FALSE.

**References**


**Examples**

data(barro)
fit.Lonly <- rq.test.khmal( y.net ~ lgdp2 + fse2 + gedy2 + Iy2 + gcony2, 
data = barro, location.scale = FALSE )

---

**rq.wfit**

*Function to choose method for Weighted Quantile Regression*

**Description**

Weight the data and then call the chosen fitting algorithm.

**Usage**

rq.wfit(x, y, tau=0.5, weights, method="br", ...)

```r
rq.wfit(x, y, tau=0.5, weights, method="br", ...)
```
20

Arguments

\(x\) the design matrix
\(y\) the response variable
\(\tau\) the quantile desired, if \(\tau\) lies outside \((0,1)\) the whole process is estimated.
weights weights used in the fitting
method method of computation: "br" is Barrodale and Roberts exterior point "fn" is the Frisch-Newton interior point method.

... Optional arguments passed to fitting routine.

See Also

\texttt{rq \ rq.fit.br \ rq.fit.fn}

\begin{verbatim}
rrs.test

Quantile Regression Rankscore Test

Description

Function to compute regression rankscore test of a linear hypothesis based on the dual quantile regression process. A test of the hypothesis, is carried out by estimating the restricted model and constructing a test based on the dual process under the restricted model. The details of the test are described in GJKP(1993). The test has a Rao-score, Lagrange-multiplier interpretation since in effect it is based on the value of the gradient of unrestricted quantile regression problem evaluated under the null. This function will eventually be superseded by a more general \texttt{anova()} method for \texttt{rq}.

Usage

\texttt{rrs.test(x0, x1, y, v, score="wilcoxon")}

Arguments

\(x0\) the matrix of maintained regressors, a column of ones is appended automatically.
\(x1\) matrix of covariates under test.
\(y\) response variable, may be omitted if \(v\) is provided.
\(v\) object of class "\texttt{rq.process}" generated e.g. by \texttt{rq(y ~ x0, tau=-1)}

score Score function for test (see \texttt{ranks})

Details

See GJKP(1993)

Value

Test statistic \(sn\) is asymptotically Chi-squared with rank(\(X1\)) dfs. The vector of ranks is also returned as component \texttt{rank}. 
References


See Also

rq, ranks

Examples

# Test that covariates 2 and 3 belong in stackloss model using Wilcoxon scores.
data(stackloss)
rrs.test(stack.x[,1], stack.x[,2:3], stack.loss)

standardize

*Function to standardize the quantile regression process*

Description

Standardize the rq process prior to Khmaladze transformation.

Usage

standardize(rqfit, location.scale=T)

Arguments

- **rqfit**: Object produced by rq.test.khmal rq fitting
- **location.scale**: If T location-scale test, if F location test

Details

Compute standardized rq process.

Value

Produces a list with several objects understood by rq.test.khmal

Author(s)

R. Koenker

References

summary.rq

**Summary method for Quantile Regression**

**Description**

Returns a summary list for a quantile regression fit. A null value will be returned if printing is invoked.

**Usage**

`summary.rq(object, se="nid", covariance=TRUE, hs = TRUE, ...)`

**Arguments**

- `object` This is an object of class "rq" produced by a call to `rq()`.
- `se` specifies the method used to compute standard errors. There are currently three available methods:
  1. "iid" which presumes that the errors are iid and computes an estimate of the asymptotic covariance matrix as in KB(1978).
  2. "nid" which presumes local (in tau) linearity (in x) of the conditional quantile functions and computes a Huber sandwich estimate using a local estimate of the sparsity.
  3. "ker" which uses a kernel estimate of the sandwich as proposed by Powell(1990).
- `covariance` logical flag to indicate whether the full covariance matrix of the estimated parameters should be returned.
- `hs` Use Hall Sheather bandwidth for sparsity estimation If false revert to Bofinger bandwidth.
- `...` Optional arguments to summary

**Value**

A list is returned with the following components

- `coefficients` a p by 4 matrix consisting of the coefficients, their estimated standard errors, their t-statistics, and their associated p-values.
- `cov` the estimated covariance matrix for the coefficients in the model, provided that `cov=TRUE` in the called sequence.
- `Hinv` inverse of the estimated Hessian matrix returned if `cov=TRUE` and `se != "iid"`.
- `J` Outer product of gradient matrix returned if `cov=TRUE` and `se != "iid"`.

The Huber sandwich is `cov = Hinv %*% J %*% Hinv`.

**References**


**See Also**

`rq` `bandwidth.rq`
Examples

data(stackloss)
y <- stack.loss
x <- stack.x
summary(rq(y ~ x, method="fn")) # Compute se's for fit using "nid" method.
summary(rq(y ~ x, ci=F, se="ker")
# default "br" alg, and compute kernel method se's

<table>
<thead>
<tr>
<th>Table of Quantile Regression Results</th>
</tr>
</thead>
<tbody>
<tr>
<td>Description</td>
</tr>
<tr>
<td>Function to produce a table of quantile regression results for a group of specified quantiles.</td>
</tr>
</tbody>
</table>

Usage

table.rq(formula, taus=c(0.05, 0.25, 0.5, 0.75, 0.95), method="br", ...)

Arguments

- formula: formula for the linear model, see rq()
- taus: quantiles of interest
- method: algorithmic method, for large problems method="fn" would be preferred.
- ...: other optional arguments passed to rq().

Details

This is only implemented for method="br", but modifications for "fn" would be quite straightforward. There is also an implementation for making a latex table in the Splus version, but this wasn't incorporated into the R package due to some incompatibilities in the unix/system commands.

Value

The function returns an array with dimensions (p, m, 3), where p is the dimension of the parameter vector of the model specified by formula, m is the number of quantiles specified by tau. For each coefficient at each tau there is a point estimate and lower and upper limits of a confidence interval. The object returned is of class table.rq and can be plotted, or formatted into a latex style table.

See Also

rq, rq.fit.br, plot.table.rq

Examples

data(stackloss)
plot(table.rq(stack.loss~stack.x))#plot results of a quantile regression
Index

*Topic datasets
  barro, 3

*Topic hplot
  plot.table.rq, 7

*Topic htest
  khmaladzize, 3
  plot.khmal, 5
  rq.test.khmal, 17
  standardize, 20

*Topic methods
  lm.fit.recursive, 4

*Topic regression
  bandwidth.rq, 2
  plot.rq.process, 6
  plot.table.rq, 7
  print.rq, 7
  print.summary.rq, 8
  ranks, 8
  rq, 9
  rq.fit, 11
  rq.fit.br, 12
  rq.fit.fn, 13
  rq.fit.pfn, 14
  rq.object, 15
  rq.process.object, 16
  rq.wfit, 18
  rrs.test, 19
  summary.rq, 21
  table.rq, 22

*Topic smooth
  akj, 1

  akj, 1

  bandwidth.rq, 2, 21
  barro, 3

  coefficients, 15

  khmaladzize, 3

  lm.fit.recursive, 4

  plot.khmal, 5
  plot.rq.process, 6

plot.table.rq, 7, 22
print.rq, 7
print.summary.rq, 8

ranks, 8, 19, 20
rq, 6–8, 9, 9, 11, 13–15, 17, 19–22
rq.fit, 11
rq.fit.br, 7, 10, 11, 12, 14, 19, 22
rq.fit.fn, 10, 11, 13, 15, 19
rq.fit.pfn, 14, 14
rq.object, 10–12, 15
rq.process.object, 10–12, 16
rq.test.khmal, 4, 17
rq.wfit, 18
rrs.test, 9, 19

standardize, 20
summary.rq, 8, 10, 11, 13, 21

table.rq, 7, 22