Package 'Qtools'

July 21, 2025

Type Package

Title Utilities for Quantiles

Version 1.5.9

Date 2023-10-28

Maintainer Marco Geraci <marco.geraci@uniroma1.it>

Depends R (>= 3.5.0)

Imports boot, conquer, glmx, graphics, grDevices, gtools, MASS, Matrix, np, numDeriv (>= 2016.8-1), quantdr, quantreg, Rcpp (>= 0.12.13), stats, utils

LinkingTo Rcpp, RcppArmadillo

Suggests knitr, mice, rmarkdown, survey

VignetteBuilder knitr

Description Functions for unconditional and conditional quantiles. These include methods for transformation-based quantile regression, quantile-based measures of location, scale and shape, methods for quantiles of discrete variables, quantile-based multiple imputation, restricted quantile regression, directional quantile classification, and quantile ratio regression.

A vignette is given in Geraci (2016, The R Journal) <doi:10.32614/RJ-2016-037> and included in the package.

License GPL (>= 2)

LazyLoad yes

NeedsCompilation yes

Author Marco Geraci [aut, cph, cre] (ORCID: <https://orcid.org/0000-0002-6311-8685>), Alessio Farcomeni [ctb] (Contributions to midrq and qrr code, ORCID: <https://orcid.org/0000-0002-7104-5826>), Cinzia Viroli [ctb] (Contributions to dqc code, ORCID: <https://orcid.org/0000-0002-3278-5266>)

Repository CRAN

Date/Publication 2023-10-28 15:10:02 UTC

Contents

`Qtools-package` *Utilities for Quantilies*

Description

The package Qtools provides functions for unconditional and conditional quantiles. These include methods for transformation-based quantile regression, quantile-based measures of location, scale and shape, methods for quantiles of discrete variables, quantile-based multiple imputation, restricted quantile regression, and directional quantile classification.

Details

Package:	Qtools
Type:	Package
Version:	1.5.9
Date:	2023-10-28
License:	GPL (>=2)
LazyLoad:	yes

Author(s)

Marco Geraci

Maintainer: Marco Geraci <marco.geraci@uniroma1.it>

ao *Transformations*

Description

Functions used in quantile regression transformation models

Usage

```
ao(theta, lambda, symm = TRUE, omega = 0.001)
invao(x, lambda, symm = TRUE, replace = TRUE)
bc(x, lambda)
invbc(x, lambda, replace = TRUE)
mcjI(x, lambda, symm = TRUE, dbounded = FALSE, omega = 0.001)
invmcjI(x, lambda, symm = TRUE, dbounded = FALSE)
mcjII(x, lambda, delta, dbounded = FALSE, omega = 0.001)
invmcjII(x, lambda, delta, dbounded = FALSE)
```

Arguments

x, theta	numeric vector of singly (x) or doubly (theta) bounded observations; theta must be between 0 and 1 (see map to map generic [a,b] intervals to [0,1]).
lambda, delta	transformation parameters.
symm	logical flag. If TRUE (default) a symmetric transformation is used.
dbounded	logical flag. If TRUE the argument x is assumed to be bounded between 0 and 1.
omega	small constant to avoid numerical problems when theta is exactly 0 or 1.
replace	logical flag. If TRUE (default), values that are outside the admissible range after the Box-Cox or the Aranda-Ordaz back-transformations are replaced by the range bounds.

Details

These functions transform (back-transform) x or theta conditional on the parameters lambda and theta, using the Box–Cox (bc), Aranda-Ordaz (ao), Proposal I (mcjI) and Proposal II (mcjII) transformations.

Value

Transformed or back-transformed values.

Author(s)

Marco Geraci

References

Aranda-Ordaz FJ. On two families of transformations to additivity for binary response data. Biometrika 1981;68(2):357-363.

Box GEP, Cox DR. An analysis of transformations. Journal of the Royal Statistical Society Series B-Statistical Methodology 1964;26(2):211-252.

Dehbi H-M, Cortina-Borja M, and Geraci M. Aranda-Ordaz quantile regression for student performance assessment. Journal of Applied Statistics. 2016;43(1):58-71.

Geraci M and Jones MC. Improved transformation-based quantile regression. Canadian Journal of Statistics 2015;43(1):118-132.

Jones MC. Connecting distributions with power tails on the real line, the half line and the interval. International Statistical Review 2007;75(1):58-69.

See Also

`tsrq`, `tsrq2`, `rcrq`, `nlrq2`

`Chemistry` *A-level Chemistry Scores*

Description

The `Chemistry` data frame has 31022 rows and 7 columns of the A-level scores in Chemistry for England and Wales students, 1997.

Format

This data frame contains the following columns:

lea school district ID.

school school ID.

id subject ID.

score a numeric vector of A-level scores in Chemistry.

sex a factor with levels `male` and `female`

age a numeric vector of ages of the subjects (months).

gcse a numeric vector of average GCSE scores.

Source

Fielding, A., Yang, M., and Goldstein, H. (2003) "Multilevel ordinal models for examination grades". Statistical Modelling, 3, 127–53.

cmidecdf *Mid-distribution Functions*

Description

Compute conditional mid-cumulative probabilities

Usage

```
cmidecdf(formula, data, ecdf_est = "npc", npc_args = list(),
theta = NULL, subset, weights, na.action,
contrasts = NULL)
cmidecdf.fit(x, y, intercept, ecdf_est, npc_args = list(),
theta = NULL)
```

Arguments

formula	an object of class "formula" (or one that can be coerced to that class): a symbolic description of the model to be fitted. The details of model specification are given under 'Details'.
data	an optional data frame, list or environment (or object coercible by as.data.frame to a data frame) containing the variables in the model. By default the variables are taken from the environment from which the call is made.
ecdf_est	estimator of the (standard) conditional cumulative distribution. The options are: npc (default) for kernel estimator (Li and Racine, 2008); logit, probit, cloglog for binomial regression; ao for Aranda-Ordaz binomial regression.
npc_args	named list of arguments for npcdistbw when ecdf_est = npc.
theta	values of the Aranda-Ordaz transformation parameter for grid search when ecdf_est = "ao".
subset	an optional vector specifying a subset of observations to be used in the fitting process.
weights	an optional vector of weights to be used in the fitting process. Not currently implemented.
na.action	a function which indicates what should happen when the data contain NAs.
contrasts	an optional list. See the contrasts.arg of model.matrix.default.
x	design matrix of dimension $n * p$.
y	vector of observations of length n.
intercept	logical flag. Does x include a vector of ones?

Value

An object of class `class` `cmidecdf` with mid-cumulative probabilities. This is a list that contains:

`G`	Estimated conditional mid-probabilities. This is a $n * k$ matrix, where n is the sample size and k is the number of unique values of `y`.
`Fhat`	Estimated (standard) cumulative probabilities.
`Fse`	Standard error for Fhat.
`yo`	unique values of `y`.
`bw`	`npcdistbw` object.
`ecdf_est`	estimator used.

Author(s)

Marco Geraci with contributions from Alessio Farcomeni

References

Geraci, M. and A. Farcomeni. Mid-quantile regression for discrete responses. arXiv:1907.01945 [stat.ME]. URL: https://arxiv.org/abs/1907.01945.

Li, Q. and J. S. Racine (2008). Nonparametric estimation of conditional cdf and quantile functions with mixed categorical and continuous data. Journal of Business and Economic Statistics 26(4), 423-434.

Peracchi, F. (2002). On estimating conditional quantiles and distribution functions. Computational Statistics and Data Analysis 38(4), 433-447.

See Also

`midecdf`

Examples

```
## Not run:
n <- 100
x <- rnorm(n, 0, 3)
y <- floor(1 + 2*x) + sample(1:5, n, replace = TRUE)
cmidecdf(y ~ x, ecdf_est = "logit")

## End(Not run)
```

coef.midrq *Extract Coefficients*

Description

coef extracts model coefficients from midrq objects.

Usage

```
## S3 method for class 'midrq'
coef(object, ...)
## S3 method for class 'midrq'
coefficients(object, ...)
```

Arguments

object	an midrq object.
...	not used.

Value

a vector for single quantiles or a matrix for multiple quantiles.

Author(s)

Marco Geraci

See Also

midrq

coef.qrr *Extract Coefficients*

Description

coef extracts model coefficients from qrr objects.

Usage

```
## S3 method for class 'qrr'
coef(object, ...)
## S3 method for class 'qrr'
coefficients(object, ...)
```

Arguments

object	an object of class qrr.
...	not used.

Value

a vector of estimated coefficients.

Author(s)

Marco Geraci

See Also

qrr

---

coef.rq.counts *Extract Coefficients*

Description

coef extracts model coefficients from rq.counts objects.

Usage

```
## S3 method for class 'rq.counts'
coef(object, ...)
## S3 method for class 'rq.counts'
coefficients(object, ...)
```

Arguments

object	an rq.counts object.
...	not used.

Value

a vector for single quantiles or a matrix for multiple quantiles.

Author(s)

Marco Geraci

See Also

rq.counts

coef.rqt *Extract Coefficients*

Description

coef extracts model coefficients from rqt objects.

Usage

```
## S3 method for class 'rqt'
coef(object, all = FALSE, ...)
## S3 method for class 'rqt'
coefficients(object, all = FALSE, ...)
```

Arguments

object	an rqt object.
all	logical flag. If FALSE (default), only the regression coefficients are returned. If TRUE, the transformation parameter(s) too is returned.
...	not used.

Value

a vector for single quantiles or a matrix for multiple quantiles.

Author(s)

Marco Geraci

See Also

tsrq

confint.midquantile *Mid-distribution Functions*

Description

Compute mid-quantiles confidence intervals

Usage

```
## S3 method for class 'midquantile'
confint(object, parm = NULL, level = 0.95, ...)
```

Arguments

object	an object of class midquantile.
parm	not used (included for consistency with confint.default).
level	nominal coverage level of the confidence interval.
...	not used.

Author(s)

Marco Geraci

References

Ma Y., Genton M., and Parzen E. Asymptotic properties of sample quantiles of discrete distributions. Annals of the Institute of Statistical Mathematics 2011;63(2):227-243

Parzen E. Quantile probability and statistical data modeling. Statistical Science 2004;19(4):652-62.

Examples

```
x <- rpois(100, lambda = 3)
mq <- midquantile(x)
confint(mq, level = 0.95)

# print standard errors
attributes(confint(mq, level = 0.95))$stderr
```

---

dqc *Directional Quantile Classification*

Description

This function is used to classify multivariate observations by means of directional quantiles.

Usage

```
dqc(formula, data, df.test, subset, weights, na.action, control = list(),
fit = TRUE)
dqc.fit(x, z, y, control)
```

Arguments

formula	an object of class formula: a two-sided formula of the form y ~ x1 + ... + xn where y represents the groups (i.e., labels) for the observations and x1, ..., xn are the variables used for classification.

data	an optional data frame, list or environment (or object coercible by as.data.frame to a data frame) containing the variables for classification (training). If not found in data, the variables are taken from environment(formula), typically the environment from which dqc is called.
df.test	a required data frame, list or environment (or object coercible by as.data.frame to a data frame) containing the variables for prediction.
subset	an optional vector specifying a subset of observations to be used in the fitting process.
weights	an optional vector of weights to be used in the fitting process.
na.action	a function which indicates what should happen when the data contain NAs.
control	list of control parameters of the fitting process. See dqcControl.
fit	logical flag. If FALSE the function returns a list of arguments for fitting.
x	design matrix of dimension $nx * p$ for training.
z	design matrix of dimension $nz * p$ for prediction.
y	vector of labels of length nx.

Details

Directional quantile classification is described in the article by Viroli et al (2020).

Value

a list of class dqc containing the following components

call	the matched call.
ans	a data frame with predictions.
nx	number of observations in the training dataset.
nz	number of observations in the prediction dataset.
p	number of variables.
control	control parameters used for fitting.

Author(s)

Marco Geraci with contributions from Cinzia Viroli

References

Viroli C, Farcomeni A, Geraci M (2020). Directional quantile-based classifiers (in preparation).

See Also

dqcControl

Examples

```
## Not run:
# Iris data
data(iris)

# Create training and prediction datasets

n <- nrow(iris)
ng <- length(unique(iris$Species))
df1 <- iris[c(1:40, 51:90, 101:140),]
df2 <- iris[c(41:50, 91:100, 141:150),]

# Classify
ctrl <- dqcControl(nt = 10, ndir = 5000, seed = 123)
fit <- dqc(Species ~ Sepal.Length + Petal.Length,
data = df1, df.test = df2, control = ctrl)

# Data frame with predictions
fit$ans

# Confusion matrix
print(cm <- xtabs( ~ fit$ans$groups + df2$Species))

# Misclassification rate
1-sum(diag(cm))/nrow(df2)

## End(Not run)
```

`dqcControl` *Control parameters for dqc estimation*

Description

A list of parameters for controlling the fitting process.

Usage

```
dqcControl(tau.range = c(0.001, 0.999), nt = 10, ndir = 50, seed = NULL)
```

Arguments

`tau.range`	vector with range of quantile probabilities. See details.
`nt`	length of grid of quantiles within `tau.range`.
`ndir`	number of directions.
`seed`	seed for `set.seed`.

Details

A directional quantile classifier (Viroli et al, 2020) is computed over a grid of quantile probabilities. The vector `tau.range` must be of length 2, providing a minimum and a maximum for the grid, or of length 1, in which case the grid will have only one probability equal to tau.range. In the latter case `nt` is ignored and set equal to 1.

Value

a list of control parameters.

Author(s)

Marco Geraci

References

Viroli C, Farcomeni A, Geraci M (2020). Directional quantile-based classifiers (in preparation).

See Also

dqc

---

esterase *Esterase Essay Data*

Description

The `esterase` data frame has 113 rows and 2 columns with the results of an essay for the concentration of an enzyme esterase.

Format

This data frame contains the following columns:

Esterase amount of esterase.

Count observed count.

Details

The esterase essay data were reported by Caroll and Ruppert (1988) and successively analyzed by Zhao (2000).

Source

R. J. Carroll and D. Ruppert, Transformation and Weighting in Regression. London: Chapman and Hall, 1988.

References

Zhao QS. Restricted regression quantiles. Journal of Multivariate Analysis 2000;72(1):78-99.

`fitted.midrq` *Extract Fitted Values from Mid-Quantile Transformation Models*

Description

This function extracts fitted values from objects of class `midrq`.

Usage

```
## S3 method for class 'midrq'
fitted(object, ...)
```

Arguments

`object`	an object of `class` `midrq`.
`...`	other arguments.

Value

a vector or a matrix or an array of fitted values.

Author(s)

Marco Geraci

See Also

`predict.midrq`

`fitted.rq.counts` *Extract Fitted Values from Quantile Regression Models for Counts*

Description

This function extracts fitted values from objects of class `rq.counts`.

Usage

```
## S3 method for class 'rq.counts'
fitted(object, ...)
```

Arguments

object	an object of class rq.counts.
...	other arguments.

Value

a vector or a matrix or an array of fitted values.

Author(s)

Marco Geraci

See Also

predict.rq.counts

---

fitted.rqt *Extract Fitted Values from Quantile Regression Transformation Models*

Description

This function extracts fitted values from objects of class rqt.

Usage

```
## S3 method for class 'rqt'
fitted(object, ...)
```

Arguments

object	an object of class rqt.
...	other arguments.

Value

a vector or a matrix or an array of fitted values.

Author(s)

Marco Geraci

See Also

predict.rqt

GOFTest *Goodness-of-Fit Tests for Quantile Regression Models*

Description

This function calculates a goodness-of-fit test for quantile regression models.

Usage

```
GOFTest(object, type = "cusum", alpha = 0.05, B = 100, seed = NULL)
```

Arguments

object	an object of class "rq","rqs", "rqt", "rrq", or "rq.counts".
type	the type of the test. See details.
alpha	the significance level for the test. This argument is relevant for type = "cusum" only.
B	the number of Monte Carlo samples. This argument is relevant for type = "cusum" only.
seed	see for random numbers. This argument is relevant for type = "cusum" only.

Details

This function provides goodness-of-fit tests for quantile regression. Currently, there is only one method available (type = "cusum"), for a test based on the cusum process of the gradient vector (He and Zhu, 2013). The critical value at level alpha is obtained by resampling. Other methods will be implemented in future versions of the package.

Value

GOFTest returns an object of class GOFtest.

Author(s)

Marco Geraci

References

He XM, Zhu LX. A lack-of-fit test for quantile regression. Journal of the American Statistical Association (2003);98:1013-1022.

Examples

```
## Not run:
data(barro, package = "quantreg")
fit <- quantreg::rq(y.net ~ lgdp2 + fse2 + gedy2 + Iy2 + gcony2, data = barro, tau = c(.1,.5,.9))
GOFTest(fit)

## End(Not run)
```

`KhmaladzeFormat` *Khmaladze Test*

Description

This function provides significance levels of the Khmaladze Test using a (hard-coded) table of asymptotic critical values.

Usage

```
KhmaladzeFormat(object, epsilon)
```

Arguments

`object`	an object of `class` "KhmaladzeTest".
`epsilon`	trimming value. One of `c(0.05,0.10,0.15,0.20,0.25,0.30)`.

Details

This function is applied to an object produced by `KhmaladzeTest`. The Khmaladze test is used to test for location–shift and location-scale–shift hypotheses (Koenker, 2005). The test statistic is computed over the interval *[epsilon, 1 - epsilon]*, where *epsilon* is the trimming value.

Author(s)

Marco Geraci

References

Appendix B in Koenker R. Quantile regression. New York, NY: Cambridge University Press; 2005.

Koenker R. and Xiao Z. Inference on the quantile regression process. Avalilable at http://www.econ.uiuc.edu/~roger/research/inference/khmal6ap.pdf.

Examples

```
data(barro, package = "quantreg")
eps <- 0.05
kt <- quantreg::KhmaladzeTest( y.net ~ lgdp2 + fse2 + gedy2 + Iy2 + gcony2,
data = barro, taus = seq(.05,.95,by = .01), trim = c(eps, 1 - eps))
class(kt)
KhmaladzeFormat(kt, epsilon = eps)
```

---

labor *Labor Pain Data*

Description

The `labor` data frame has 358 rows and 4 columns of the change in pain over time for several 83 women in labor.

Format

This data frame contains the following columns:

subject an ordered factor indicating the subject on which the measurement was made. The levels are labelled `1` to `83`.

pain a numeric vector of self–reported pain scores on a 100mm line.

treatment a dummy variable with values `1` for subjects who received a pain medication and `0` for subjects who received a placebo.

time a numeric vector of times (minutes since randomization) at which `pain` was measured.

Details

The labor pain data were reported by Davis (1991) and successively analyzed by Jung (1996) and Geraci and Bottai (2007). The data set consists of repeated measurements of self–reported amount of pain on N = 83 women in labor, of which 43 were randomly assigned to a pain medication group and 40 to a placebo group. The response was measured every 30 min on a 100–mm line, where 0 means no pain and 100 means extreme pain. A nearly monotone pattern of missing data was found for the response variable and the maximum number of measurements for each woman was six.

Source

Davis CS (1991). Semi–parametric and non–parametric methods for the analysis of repeated measurements with applications to clinical trials. Statistics in Medicine 10, 1959–80.

References

Geraci M and Bottai M (2007). Quantile regression for longitudinal data using the asymmetric Laplace distribution. Biostatistics 8(1), 140–154.

Jung S (1996). Quasi–likelihood for median regression models. Journal of the American Statistical Association 91, 251–7.

maref.rqt *Marginal Effects*

Description

This function computes marginal effects for `rqt` and `rq.counts` objects.

Usage

```
maref(object, namevec)
## S3 method for class 'rqt'
maref(object, namevec)
## S3 method for class 'rq.counts'
maref(object, namevec)
```

Arguments

`object`	an `rqt` or an `rq.counts` object.
`namevec`	character giving the name of the covariate with respect to which the marginal effect is to be computed.

Details

Given the τth conditional quantile function $Q_{h(Y)|X}(\tau) = \eta = Xb$, where Y is the response variable, X a design matrix, and h is a one-parameter transformation with inverse $h^{-1} = g$, `maref` computes the marginal effect:

$$\frac{dQ_{Y|X}(\tau)}{dx_j} = \frac{dg\{Q_{h(Y)|X}(\tau)\}}{dx_j}$$

where x_j is the *j*-th covariate with respect to which the marginal effect is to be computed and its name is given in the argument `namevec`.

The derivative of the quantile function is the the product of two components

$$\frac{dQ_{Y|X}(\tau)}{dx_j} = \frac{dg(\eta)}{d\eta} \cdot \frac{d\eta}{dx_j}$$

The derivative w.r.t. the linear predictor η is calculated symbolically after parsing the `object`'s formula and is evaluated using the `object`'s model frame. The function that parses formulae has a limited scope. It recognizes interactions and basic operators (e.g., log, exp, etc.). Therefore, it is recommended to use simple expressions for the model's formula.

This function can be applied to models of class `rqt` and `rq.counts`. Note that marginal effects can be similarly obtained using `predict.rqt` or `predict.rq.counts` with argument `type = "maref"` which, in addition, allows for an optional data frame to be specified via `newdata`.

Value

a vector for single quantiles or a matrix for multiple quantiles of marginal effects.

Author(s)

Marco Geraci

See Also

tsrq

Examples

```
## Not run:
# Box-Cox quantile regression model (dataset trees from package 'datasets')
fit <- tsrq(Volume ~ Height, data = trees, tsf = "bc", tau = 0.9)

# Coefficients (transformed scale)
coef(fit)

# Design matrix
head(fit$x)

# Marginal effect of 'Height'
maref(fit, namevec = "Height")

# Predict marginal effects over grid of values for Height
nd <- data.frame(Height = seq(min(trees$Height), max(trees$Height), length = 100))
x <- predict(fit, newdata = nd, type = "maref", namevec = "Height")

# Plot
plot(nd$Height, x, xlab = "Height", ylab = "Marginal effect on volume")

# Include 'Girth' and interaction between 'Height' and 'Girth'
fit <- tsrq(Volume ~ Height * Girth, data = trees, tsf = "bc", tau = 0.5)
head(fit$x)

# Predict marginal effects over grid of values for Height (for fixed girth)
nd$Girth <- rep(mean(trees$Girth), 100)
x <- predict(fit, newdata = nd, type = "maref", namevec = "Height")
plot(nd$Height, x, xlab = "Height", ylab = "Marginal effect on volume")

# Quantile regression for counts (log transformation)
data(esterase)
fit <- rq.counts(Count ~ Esterase, tau = 0.25, data = esterase, M = 50)
maref(fit, namevec = "Esterase")

## End(Not run)
```

---

mice.impute.rq *QR-based Multiple Imputation*

Description

This function is used to multiply impute missing values using quantile regression imputation models.

Usage

```
mice.impute.rq(y, ry, x, tsf = "none", symm = TRUE, dbounded = FALSE,
lambda = NULL, x.r = NULL, par = NULL, conditional = TRUE,
epsilon = 0.001, method.rq = "fn", ...)
mice.impute.rrq(y, ry, x, tsf = "none", symm = TRUE, dbounded = FALSE,
lambda = NULL, epsilon = 0.001, method.rq = "fn", ...)
```

Arguments

y	numeric vector of length n with nmis missing values.
ry	missing data indicator. Logical vector of length n: FALSE if y is missing, TRUE if y is observed.
x	matrix n x p of completely observed covariates.
tsf	transformation to be used. Possible options are mcjI for Proposal I, bc for Box-Cox and ao for Aranda-Ordaz transformation models. No transformation is used by default.
symm	logical flag. If TRUE (default) a symmetric transformation is used.
dbounded	logical flag. If TRUE the response y is assumed to be bounded between 0 and 1.
lambda	if conditional = TRUE, a numerical value for the transformation parameter. This is provided by the user or set to zero if not specified. If conditional = FALSE, this argument is ignored.
x.r	range of the mapping for doubly bounded variables.
par	if conditional = FALSE, starting values for nlrq1 can be provided via this argument. See argument start in nlrq1 for details.
conditional	logical flag. If TRUE (default), the transformation parameter is assumed to be known and this must be provided via the argument lambda. Otherwise, it is estimated via nlrq1.
epsilon	constant used to trim the values of the sample space.
method.rq	linear programming algorithm (see rq).
...	additional arguments.

Details

This function implements the methods proposed by Geraci (2016) and Geraci and McLain (2018) to impute missing values using quantile regression models. Uniform values are sampled from *[epsilon, 1 - epsilon]*, therefore allowing the interval to be bounded away from 0 and 1 (default is 0.001). It is possible to specify a quantile regression transformation model with parameter lambda (Geraci and Jones). The function mice.impute.rrq performs imputation based on restricted regression quantiles to avoid quantile crossing (see Geraci 2016 for details).

Value

A vector of length `nmis` with imputations.

Author(s)

Marco Geraci

References

Bottai, M., & Zhen, H. (2013). Multiple imputation based on conditional quantile estimation. Epidemiology, Biostatistics, and Public Health, 10(1), e8758.

Geraci, M. (2016). Estimation of regression quantiles in complex surveys with data missing at random: An application to birthweight determinants. Statistical Methods in Medical Research, 25(4), 1393-1421.

Geraci, M., and Jones, M. C. (2015). Improved transformation-based quantile regression. Canadian Journal of Statistics, 43(1), 118-132.

Geraci, M., and McLain, A. (2018). Multiple imputation for bounded variables. Psychometrika, 83(4), 919-940.

van Buuren, S., and Groothuis-Oudshoorn, K. (2011). mice: Multivariate imputation by chained equations in R. Journal of Statistical Software, 45(3), 1–67.

See Also

`ao`, `tsrq`

Examples

```
## Not run:

# Load package 'mice'
require(mice)

# Load data nhanes
data(nhanes)
nhanes2 <- nhanes
nhanes2$hyp <- as.factor(nhanes2$hyp)

# Impute continuous variables using quantile regression
set.seed(199)
imp <- mice(nhanes2, meth = c("polyreg", "rq", "logreg", "rq"), m = 5)

# estimate linear regression and pool results
fit <- lm.mids(bmi ~ hyp + chl, data = imp)
pool(fit)

# Impute using restricted quantile regression
set.seed(199)
imp <- mice(nhanes2, meth = c("polyreg", "rrq", "logreg", "rrq"), m = 5)
fit <- lm.mids(bmi ~ hyp + chl, data = imp)
```

```
pool(fit)

# Impute using quantile regression + Box-Cox transformation with parameter
# lambda = 0 (ie, log transformation)

set.seed(199)
imp <- mice(nhanes2, meth = c("polyreg", "rq", "logreg", "rq"), m = 5, tsf = "bc", lambda = 0)
fit <- lm.mids(bmi ~ hyp + chl, data = imp)
pool(fit)



## End(Not run)
```

`midq2q` *Recover Ordinary Conditional Quantiles from Conditional Mid-Quantiles*

Description

This function recovers ordinary conditional quantile functions based on fitted mid-quantile regression models.

Usage

```
midq2q(object, newdata, observed = FALSE, ...)
```

Arguments

`object`	an object of `class` `midrq`.
`newdata`	a required data frame in which to look for variables with which to predict.
`observed`	logical flag. If `TRUE`, ordinary quantiles are recovered from observed sample values. Otherwise, they are calcuated as rounded mid-quantiles. See details.
`...`	not used.

Details

If the values of the support of the random variable are equally spaced integers, then `observed` should ideally be set to `FALSE` so that the ordinary quantile is obtained by rounding the predicted mid-quantile. Otherwise, the function returns an integer observed in the sample. See Geraci and Farcomeni for more details.

Value

a vector or a matrix of estimated ordinary quantiles. The attribute `Fhat` provides the corresponding estimated cumulative distribution.

Author(s)

Marco Geraci

References

Geraci, M. and A. Farcomeni. Mid-quantile regression for discrete responses. arXiv:1907.01945 [stat.ME]. URL: https://arxiv.org/abs/1907.01945.

See Also

plot.midq2q, predict.midrq

Examples

```
## Not run:
# Esterase data
data(esterase)

# Fit quantiles 0.1, 0.15, ..., 0.85
fit <- midrq(Count ~ Esterase, tau = 2:17/20, data = esterase, type = 3, lambda = 0)

# Recover ordinary quantile function
xx <- seq(min(esterase$Esterase), max(esterase$Esterase), length = 5)
print(Qhat <- midq2q(fit, newdata = data.frame(Esterase = xx)))

# Plot
plot(Qhat, sub = TRUE)


## End(Not run)
```

---

midquantile *Mid-distribution Functions*

Description

Compute mid-cumulative probabilities and mid-quantiles

Usage

```
midecdf(x, na.rm = FALSE)
midquantile(x, probs = 1:3/4, na.rm = FALSE)
```

Arguments

x	numeric vector of observations used to estimate the mid-cumulative distribution or the mid-quantiles.
probs	numeric vector of probabilities with values in [0,1].
na.rm	logical value indicating whether NA values should be stripped before the computation proceeds.

Value

An object of class class midecdf or midquantile with mid-cumulative probabilities and mid-quantiles. For midecdf, this is a list that contains:

x	unique values of the vector x at which mid-cumulative probabilities are calculated.
y	estimated mid-cumulative probabilities.
fn	interpolating function of the points (x,y).
data	input values.

For midquantile, this is a list that contains:

x	probabilities probs at which mid-quantiles are calculated.
y	estimated mid-cumulative probabilities.
fn	interpolating function of the points (x,y).
data	input values.

Author(s)

Marco Geraci

References

Ma Y., Genton M., and Parzen E. Asymptotic properties of sample quantiles of discrete distributions. Annals of the Institute of Statistical Mathematics 2011;63(2):227-243

Parzen E. Quantile probability and statistical data modeling. Statistical Science 2004;19(4):652-62.

See Also

confint.midquantile, plot.midquantile

Examples

```
x <- rpois(100, lambda = 3)
midquantile(x)
```

midrq *Mid-Quantile Regression for Discrete Responses*

Description

This function is used to fit a mid-quantile regression model when the response is discrete.

Usage

```
midrq(formula, data, tau = 0.5, lambda = NULL, subset, weights, na.action,
contrasts = NULL, offset, type = 3, midFit = NULL, control = list())
midrq.fit(x, y, offset, lambda, binary, midFit, type, tau, method)
```

Arguments

formula	an object of class formula: a symbolic description of the model to be fitted.
data	an optional data frame, list or environment (or object coercible by as.data.frame to a data frame) containing the variables in the model. If not found in data, the variables are taken from environment(formula), typically the environment from which midrq is called.
tau	quantile to be estimated. This can be a vector of quantiles in midrq, but must be one single quantile in midrq.fit.
lambda	a numerical value for the transformation parameter. This is provided by the user or set to NULL. The transformation is always Box-Cox, unless the response is binary (0-1) in which case the trasformation is Aranda-Ordaz. See bc and ao.
subset	an optional vector specifying a subset of observations to be used in the fitting process.
weights	an optional vector of weights to be used in the fitting process.
na.action	a function which indicates what should happen when the data contain NAs.
contrasts	an optional list. See the contrasts.arg of model.matrix.default.
offset	an optional offset to be included in the model frame. This must be provided in midrq.fit (e.g., a vector of zeros).
type	estimation method for the fitting process. See details.
midFit	cmidecdf object used for fitting conditional mid-quantiles. If set to NULL in midrq, it is automatically created. It must be provided in midrq.fit.
control	list of control parameters of the fitting process. See midrqControl.
x	design matrix of dimension $n * p$.
y	vector of observations of length n.
binary	logical flag. Is the response binary?
method	character vector that specifies the optimization algorithm in optim to fit a conditional mid-quantile model when type = 1 or type = 2. Only "Nelder-Mead" has been tested.

Details

A linear mid-quantile regression model is fitted to the transformed response. The transformation of the response can be changed with the argument `lambda`. If `lambda = NULL`, then no transformation is applied (i.e., identity); if `lambda` is a numeric value, then the Box-Cox transformation is applied (e.g., 0 for log-transformation). However, `midrq` will automatically detect whether the response is binary, in which case the Aranda-Ordaz transformation is applied. In contrast, the user must declare whether the response is binary in `midrq.fit`.

There are 3 different estimators. `type = 1` is based on a general-purpose estimator (i.e., `optim`). `type = 2` is similar to `type = 1`, except the loss function is averaged over the space of the predictors (i.e., CUSUM). `type = 3` is the least-squares estimator discussed by Geraci and Farcomeni (2019).

The warning 'tau is outside mid-probabilities range' indicates that there are observations for which tau is below or above the range of the corresponding estimated conditional mid-probabilities. This affects estimation in a way similar to censoring.

Value

a list of class `midrq` containing the following components

`call`	the matched call.
`x`	the model matrix.
`y`	the model response.
`hy`	the tranformed model response.
`tau`	the order of the estimated quantile(s).
`coefficients`	regression quantile (on the log–scale).
`fitted.values`	fitted values (on the response scale).
`offset`	offset.
`terms`	the terms object used.
`term.labels`	names of coefficients.

Author(s)

Marco Geraci with contributions from Alessio Farcomeni

References

Geraci, M. and A. Farcomeni. Mid-quantile regression for discrete responses. arXiv:1907.01945 [stat.ME]. URL: https://arxiv.org/abs/1907.01945.

See Also

`residuals.midrq`, `predict.midrq`, `coef.midrq`

Examples

```
## Not run:
# Esterase data
data(esterase)

# Fit quantiles 0.25 and 0.75
fit <- midrq(Count ~ Esterase, tau = c(0.25, 0.75), data = esterase, type = 3, lambda = 0)
coef(fit)

# Plot
with(esterase, plot(Count ~ Esterase))
lines(esterase$Esterase, fit$fitted.values[,1], col = "blue")
lines(esterase$Esterase, fit$fitted.values[,2], col = "red")
legend(8, 1000, lty = c(1,1), col = c("blue", "red"), legend = c("tau = 0.25","tau = 0.75"))


## End(Not run)
```

`midrqControl` *Control parameters for midrq estimation*

Description

A list of parameters for controlling the fitting process.

Usage

```
midrqControl(method = "Nelder-Mead", ecdf_est = "npc", npc_args = list())
```

Arguments

`method`	character vector that specifies the optimization algorithm in `optim` to fit a conditional mid-quantile model when `type = 1` or `type = 2`. Only `"Nelder-Mead"` has been tested.
`ecdf_est`	estimator of the (standard) conditional cumulative distribution. The options are: `npc` (default) for kernel estimator (Li and Racine, 2008); `logit`, `probit`, `cloglog` for binomial regression; `ao` for Aranda-Ordaz binomial regression.
`npc_args`	named list of arguments for `npcdistbw` when `ecdf_est = npc`.

Value

a list of control parameters.

Author(s)

Marco Geraci

References

Geraci, M. and A. Farcomeni. Mid-quantile regression for discrete responses. arXiv:1907.01945 [stat.ME]. URL: https://arxiv.org/abs/1907.01945.

Li, Q. and J. S. Racine (2008). Nonparametric estimation of conditional cdf and quantile functions with mixed categorical and continuous data. Journal of Business and Economic Statistics 26(4), 423-434.

See Also

midrq

`nlControl` *Control parameters for gradient search estimation*

Description

A list of parameters for controlling the fitting process.

Usage

```
nlControl(tol_ll = 1e-05, tol_theta = 0.001, check_theta = FALSE,
step = NULL, beta = 0.5, gamma = 1.25, reset_step = FALSE,
maxit = 1000, smooth = FALSE, omicron = 0.001, verbose = FALSE)
```

Arguments

`tol_ll`	tolerance expressed as relative change of the objective function.
`tol_theta`	tolerance expressed as relative change of the estimates.
`check_theta`	logical flag. If `TRUE` the algorithm performs a check on the change in the estimates in addition to the likelihood.
`step`	step size (default standard deviation of response).
`beta`	decreasing step factor for line search (0,1).
`gamma`	nondecreasing step factor for line search (>= 1).
`reset_step`	logical flag. If `TRUE` the step size is re-setted to the initial value at each iteration.
`maxit`	maximum number of iterations.
`smooth`	logical flag. If `TRUE` the standard loss function is replaced with a smooth approximation.
`omicron`	small constant for smoothing the loss function when using `smooth = TRUE`. See details.
`verbose`	logical flag.

Details

The optimization algorithm is along the lines of the gradient search algorithm (Bottai et al, 2015). If `smooth = TRUE`, the classical non-differentiable loss function is replaced with a smooth version (Chen and Wei, 2005).

Value

a list of control parameters.

Author(s)

Marco Geraci

References

Bottai M, Orsini N, Geraci M (2015). A Gradient Search Maximization Algorithm for the Asymmetric Laplace Likelihood, Journal of Statistical Computation and Simulation, 85(10), 1919-1925.

Chen C, Wei Y (2005). Computational issues for quantile regression. Sankhya: The Indian Journal of Statistics, 67(2), 399-417.

See Also

`nlrq1`

Orthodont *Growth curve data on an orthdontic measurement*

Description

The `Orthodont` data frame has 108 rows and 4 columns of the change in an orthdontic measurement over time for several young subjects.

Format

This data frame contains the following columns:

distance a numeric vector of distances from the pituitary to the pterygomaxillary fissure (mm). These distances are measured on x-ray images of the skull.

age a numeric vector of ages of the subject (yr).

Subject an ordered factor indicating the subject on which the measurement was made. The levels are labelled `M01` to `M16` for the males and `F01` to `F13` for the females. The ordering is by increasing average distance within sex.

Sex a factor with levels `Male` and `Female`

Details

Investigators at the University of North Carolina Dental School followed the growth of 27 children (16 males, 11 females) from age 8 until age 14. Every two years they measured the distance between the pituitary and the pterygomaxillary fissure, two points that are easily identified on x-ray exposures of the side of the head.

Source

Pinheiro, J. C. and Bates, D. M. (2000), *Mixed-Effects Models in S and S-PLUS*, Springer, New York. (Appendix A.17)

Potthoff, R. F. and Roy, S. N. (1964), "A generalized multivariate analysis of variance model useful especially for growth curve problems", Biometrika, 51, 313–326.

Jose Pinheiro, Douglas Bates, Saikat DebRoy, Deepayan Sarkar and the R Development Core Team (2011). nlme: Linear and Nonlinear Mixed Effects Models. R package version 3.1-100.

`plot.midq2q` *Plot Quantile Functions*

Description

Plot an object generated by `midq2q`.

Usage

```
## S3 method for class 'midq2q'
plot(x, ..., xlab = "p", ylab = "Quantile",
main = "Ordinary Quantile Function", sub = TRUE, verticals = TRUE,
col.steps = "gray70", cex.points = 1, jumps = FALSE)
```

Arguments

`x`	a `midq2q` object.
`...`	additional arguments for `plot.default`.
`xlab`	a label for the x axis.
`ylab`	a label for the y axis.
`main`	a main title for the plot.
`sub`	if `TRUE`, a subtitle with indication of the row of `x`.
`verticals`	logical. If `TRUE`, draw vertical lines at steps.
`col.steps`	the color for the steps of ordinary quantiles.
`cex.points`	amount by which plotting characters and symbols should be scaled relative to the default.
`jumps`	logical flag. Should values at jumps be marked?

Author(s)

Marco Geraci

See Also

midq2q, midecdf, midquantile

---

plot.midquantile *Plot Mid-distribution Functions*

Description

Plot an object generated by midecdf or midquantile.

Usage

```
## S3 method for class 'midecdf'
plot(x, ..., ylab = "p", main = "Ordinary and Mid-ECDF", verticals = FALSE,
col.01line = "gray70", col.steps = "gray70", col.midline ="black", cex.points = 1,
lty.midline = 2, lwd = 1, jumps = FALSE)
## S3 method for class 'midquantile'
plot(x, ..., xlab = "p", ylab = "Quantile", main = "Ordinary and Mid-Quantiles",
col.steps = "gray70", col.midline = "black", cex.points = 1, lty.midline = 2,
lwd = 1, jumps = FALSE)
```

Arguments

x	a midecdf or a midquantile object.
...	additional arguments for plot.default.
xlab	a label for the x axis.
ylab	a label for the y axis.
main	a main title for the plot.
verticals	logical. If TRUE, draw vertical lines at steps.
col.01line	numeric or character specifying the color of the horizontal lines at y = 0 and 1.
col.steps	the color for the steps of ordinary quantiles.
col.midline	the color for the mid-ecdf or the mid-quantile line.
cex.points	amount by which plotting characters and symbols should be scaled relative to the default.
lty.midline	line type for the mid-ecdf or the mid-quantile line.
lwd	line width of the mid-ecdf or the mid-quantile line.
jumps	logical flag. Should values at jumps be marked (with the convention that, at the point of discontinuity or 'jump', the function takes its value corresponding to the ordinate of the filled circle as opposed to that of the hollow circle)?

Author(s)

Marco Geraci

See Also

midecdf, midquantile

---

plot.qlss *Quantile-based Summary Statistics for Location, Scale and Shape*

Description

This function plots location, scale and shape of a conditional distribution.

Usage

```
## S3 method for class 'qlss'
plot(x, z, whichp = NULL, interval = FALSE, type = "l", ...)
```

Arguments

x	an object of class qlss as returned by qlss.formula.
z	numeric vector of values against which LSS measures are plotted. This argument is required.
whichp	when probs in qlss is a vector, the argument whichp specifies one of the probabilities (and one only) in probs that should be used for plotting. If whichp = NULL (default), the first value in probs is used.
interval	logical flag. If TRUE, confidence intervals for the predictions are plotted.
type	1-character string giving the type of plot desired. See plot.default.
...	other arguments for plot.default.

Details

This function plots a qlss object from qlss or predict.qlss.

Author(s)

Marco Geraci

See Also

qlss

Examples

```
trees2 <- trees[order(trees$Height),]
fit <- qlss(Volume ~ Height, data = trees2, probs = c(.05, .1))
# Plot the results for probs = 0.1
plot(fit, z = trees2$Height, whichp = 0.1, xlab = "height")
```

`predict.midrq` *Predictions from Mid-Quantile Regression Models*

Description

This function computes predictions based on fitted mid-quantile regression models.

Usage

```
## S3 method for class 'midrq'
predict(object, newdata, offset, na.action = na.pass,
type = "response", ...)
```

Arguments

`object`	an object of `class` `midrq`.
`newdata`	an optional data frame in which to look for variables with which to predict. If omitted, the fitted values are used.
`offset`	an optional offset to be included in the model frame (when `newdata` is provided).
`na.action`	function determining what should be done with missing values in `newdata`. The default is to predict `NA`.
`type`	the type of prediction required. The default `"response"` is on the scale of the response variable, i.e. the values are back-transformed using the inverse of the transformation $h^{-1}(Xb)$; the alternative `"link"` is on the scale of the linear predictors $h(y) = Xb$.
`...`	not used.

Value

a vector or a matrix or an array of predictions.

Author(s)

Marco Geraci

See Also

`residuals.midrq`, `midrq`, `coef.midrq`

predict.qlss *Predictions from Conditional LSS Objects*

Description

This function computes predictions based on fitted conditional QLSS objects.

Usage

```
## S3 method for class 'qlss'
predict(object, newdata, interval = FALSE, level = 0.95, R = 200,
na.action = na.pass, trim = 0.05, ...)
```

Arguments

object	an object as returned by qlss.formula.
newdata	an optional data frame in which to look for variables with which to predict. If omitted, the fitted values are used.
interval	logical flag. If TRUE, confidence intervals for predictions are computed by bootstrap.
level	nominal coverage level of the confidence interval.
R	number of bootstrap replications used to compute confidence intervals.
na.action	function determining what should be done with missing values in newdata. The default is to predict NA.
trim	proportion of extreme bootstrap replications to be trimmed before standard errors are computed.
...	not used.

Author(s)

Marco Geraci

See Also

qlss.formula

Examples

```
## Not run:
# Fit QLSS object
trees2 <- trees[order(trees$Height),]
fit <- qlss(Volume ~ Height, data = trees2)

## Predict using newdata. Calculate confidence intervals using 200 bootstrap replications
# large confidence intervals for shape index due to small IQR at low values of height
#xx <- seq(min(trees2$Height), max(trees2$Height), length = 100)
```

```
#new <- data.frame(Height = xx)
#set.seed(121)
#fit.pred <- predict(fit, newdata = new, interval = TRUE, level = 0.95, R = 200)
#plot(fit.pred, z = xx, interval = TRUE, xlab = "height")

# Restrict range for Height

xx <- seq(65, 87, length = 100)
new <- data.frame(Height = xx)
set.seed(121)
fit.pred <- predict(fit, newdata = new, interval = TRUE, level = 0.95, R = 200)
plot(fit.pred, z = xx, interval = TRUE, xlab = "height") # better


## End(Not run)
```

`predict.qrr` *Predictions from Quantile Ratio Regression Models*

Description

This function computes predictions based on quantile ratio regression models.

Usage

```
## S3 method for class 'qrr'
predict(object, newdata, na.action = na.pass,
type = "response", ...)
```

Arguments

`object`	an object of `class` `qrr`.
`newdata`	an optional data frame in which to look for variables with which to predict. If omitted, the fitted values are used.
`na.action`	function determining what should be done with missing values in `newdata`. The default is to predict `NA`.
`type`	the type of prediction required. The default `"response"` is on the scale of the response variable, i.e. the values are back-transformed using the inverse of the link function $g^{-1}(Xb) = 1 + exp(Xb)$; the alternative `"link"` is on the scale of the linear predictor.
`...`	not used.

Value

a vector of predictions.

Author(s)

Marco Geraci

See Also

qrr

---

predict.rq.counts *Predictions from rq.counts Objects*

Description

This function computes predictions based on fitted linear quantile models.

Usage

```
## S3 method for class 'rq.counts'
predict(object, newdata, offset,
na.action = na.pass, type = "response",
namevec = NULL, ...)
```

Arguments

object	an rq.counts object.
newdata	an optional data frame in which to look for variables with which to predict. If omitted, the fitted values are used.
offset	an offset to be used with newdata.
na.action	function determining what should be done with missing values in newdata. The default is to predict NA.
type	the type of prediction required. The default "response" is on the scale of the response variable, i.e. the values are back-transformed using the inverse of the transformation $h^{-1}(Xb)$; the alternative "link" is on the scale of the linear predictors $h(y) = Xb$; finally, predictions for marginal effects are given with "maref".
namevec	character giving the name of the covariate with respect to which the marginal effect is to be computed. If type = "maref", this argument is required. See maref.rq.counts.
...	not used.

Value

a vector or a matrix or an array of predictions.

Author(s)

Marco Geraci

See Also

residuals.rq.counts, rq.counts, coef.rq.counts, maref.rq.counts

Examples

```
# Esterase data
data(esterase)

# Fit quantiles 0.25 and 0.75
fit <- rq.counts(Count ~ Esterase, tau = 0.5, data = esterase, M = 50)
cbind(fit$fitted.values, predict(fit, type = "response"))
```

---

predict.rqt *Predictions from Quantile Regression Transformation Models*

Description

This function computes predictions based on fitted quantile regression transformation models.

Usage

```
## S3 method for class 'rqt'
predict(object, newdata, na.action = na.pass,
type = "response", namevec = NULL, ...)
```

Arguments

object	an object of class rqt.
newdata	an optional data frame in which to look for variables with which to predict. If omitted, the fitted values are used.
na.action	function determining what should be done with missing values in newdata. The default is to predict NA.
type	the type of prediction required. The default "response" is on the scale of the response variable, i.e. the values are back-transformed using the inverse of the transformation $h^{-1}(Xb)$; the alternative "link" is on the scale of the linear predictors $h(y) = Xb$; finally, predictions for marginal effects are given with "maref".
namevec	character giving the name of the covariate with respect to which the marginal effect is to be computed. If type = "maref", this argument is required. See maref.rqt.
...	not used.

Value

a vector or a matrix or an array of predictions.

Author(s)

Marco Geraci

See Also

residuals.rqt, tsrq, coef.rqt, maref.rqt

---

predict.rrq *Predictions from Restricted Quantile Regression Models*

Description

This function computes predictions based on fitted restricted quantile regression models.

Usage

```
## S3 method for class 'rrq'
predict(object, newdata, na.action = na.pass, ...)
```

Arguments

object	an object of class rrq.
newdata	an optional data frame in which to look for variables with which to predict. If omitted, the fitted values are used.
na.action	function determining what should be done with missing values in newdata. The default is to predict NA.
...	not used.

Value

a vector or a matrix or an array of predictions.

Author(s)

Marco Geraci

print.cmidecdf *Print Mid-distribution Functions*

Description

Print an object generated by cmidecdf.

Usage

```
## S3 method for class 'cmidecdf'
print(x, ...)
```

Arguments

x	a midecdf or a midquantile object.
...	not used.

Author(s)

Marco Geraci

See Also

cmidecdf

print.dqc *Print Directional Quantile Classification Objects*

Description

Print an object of class dqc.

Usage

```
## S3 method for class 'dqc'
print(x, ...)
```

Arguments

x	an object of class dqc.
...	other arguments used by print.default.

Author(s)

Marco Geraci

See Also

dqc

---

print.GOFTest *Print Goodness-of-Fit Test for Quantile Regression Models*

Description

Print an object generated by GOFTest.

Usage

```
## S3 method for class 'GOFTest'
print(x, digits = max(3, getOption("digits") - 3), ...)
```

Arguments

x	an GOFTest object.
digits	a non-null value for digits specifies the minimum number of significant digits to be printed in values.
...	not used.

Author(s)

Marco Geraci

See Also

GOFTest

---

print.midquantile *Print Mid-distribution Functions*

Description

Print an object generated by midecdf or midquantile.

Usage

```
## S3 method for class 'midecdf'
print(x, ...)
## S3 method for class 'midquantile'
print(x, ...)
```

Arguments

x	a midecdf or a midquantile object.
...	not used.

Author(s)

Marco Geraci

See Also

midecdf, midquantile

---

print.midrq *Print Mid-Quantile Models*

Description

Print an object of class midrq or summary.midrq.

Usage

```
## S3 method for class 'midrq'
print(x, ...)
## S3 method for class 'summary.midrq'
print(x, ...)
```

Arguments

x	an object of class midrq or summary.midrq.
...	other arguments used by print.default.

Author(s)

Marco Geraci

See Also

midrq

print.qlss *Print Quantile-based Summary Statistics for Location, Scale and Shape*

Description

Print an object generated by `qlss`.

Usage

```
## S3 method for class 'qlss'
print(x, ...)
```

Arguments

`x`	an `qlss` object.
`...`	not used.

Author(s)

Marco Geraci

See Also

`qlss`

print.qrr *Print Quantile Ratio Regression Models*

Description

Print an object of class `qrr` or `summary.qrr`.

Usage

```
## S3 method for class 'qrr'
print(x, ...)
## S3 method for class 'summary.qrr'
print(x, ...)
```

Arguments

`x`	an object of `class` `qrr` or `summary.qrr`.
`...`	other arguments used by `print.default`.

Author(s)

Marco Geraci

See Also

qrr

`print.rq.counts` *Print rq.counts*

Description

Print an object generated by `rq.counts`.

Usage

```
## S3 method for class 'rq.counts'
print(x, digits = max(3, getOption("digits") - 3), ...)
```

Arguments

`x`	an `rq.counts` object.
`digits`	a non-null value for digits specifies the minimum number of significant digits to be printed in values.
`...`	not used.

Author(s)

Marco Geraci

See Also

`rq.counts`

print.rqt *Print Transformation Models*

Description

Print an object of class `rqt` or `summary.rqt`.

Usage

```
## S3 method for class 'rqt'
print(x, ...)
## S3 method for class 'summary.rqt'
print(x, ...)
```

Arguments

`x`	an object of `class` `rqt` or `summary.rqt`.
`...`	other arguments used by `print.default`.

Author(s)

Marco Geraci

See Also

`tsrq`, `rcrq`, `tsrq2` or `nlrq2`

print.rrq *Print Restricted Quantile Regression Models*

Description

Print an object of class `rrq` or `summary.rrq`.

Usage

```
## S3 method for class 'rrq'
print(x, ...)
## S3 method for class 'summary.rrq'
print(x, ...)
```

Arguments

`x`	an object of `class` `rrq` or `summary.rrq`.
`...`	other arguments used by `print.default`.

Author(s)

Marco Geraci

See Also

rrq

---

qexact *Exact Confidence Intervals for Quantiles*

Description

Compute exact confidence intervals for quantiles of continuous random variables using binomial probabilities

Usage

```
qexact(x, probs = 0.5, level = 0.95)
```

Arguments

x	numeric vector whose sample quantile and confidence intervals are to be calculated.
probs	numeric vector of probabilities with values in [0,1].
level	nominal coverage level of the confidence interval.

Details

This function calculates exact confidence intervals for quantiles at level probs from a vector x of length n. It does so by first determining the confidence level for all possible pairwise combinations of order statistics from 1 to n. This entails "n choose 2" possible confidence intervals before selecting the one with the level closest to level. If the procedure yields more than one such confidence intervals, then the interval with smallest width is returned.

Caution: for large n, the procedure may reach the limit on the number of nested expressions. See gtools::combinations and options(expressions) for additional information. However, if you have a large n, then consider estimating an asymptotic approximation of the confidence interval.

Author(s)

Marco Geraci

References

Thompson W. R. On confidence ranges for the median and other expectation distributions for populations of unknown distribution form. The Annals of Mathematical Statistics 1936;7(3):122-128.

Examples

```
x <- rnorm(100)
qexact(x, p = c(0.1,0.5), level = 0.9)
```

---

qlss *Quantile-based Summary Statistics for Location, Scale and Shape*

Description

This function calculates quantile-based summary statistics for location, scale and shape of a distribution, unconditional or conditional.

Usage

```
qlss(...)
## Default S3 method:
qlss(fun = "qnorm", probs = 0.1, ...)
## S3 method for class 'numeric'
qlss(x, probs = 0.1, ...)
## S3 method for class 'formula'
qlss(formula, probs = 0.1, data = sys.frame(sys.parent()), subset, weights,
na.action, contrasts = NULL, method = "fn", type = "rq", tsf = "mcjI",
symm = TRUE, dbounded = FALSE, lambda = NULL, conditional = FALSE, ...)
```

Arguments

fun	quantile function.
x	a numeric vector.
formula	an object of class formula: a symbolic description of the model to be fitted. The details of model specification are given under "Details".
probs	a vector of probabilities.
data	an optional data frame, list or environment (or object coercible by as.data.frame to a data frame) containing the variables in the model. By default the variables are taken from the environment from which the call is made.
subset	an optional vector specifying a subset of observations to be used in the fitting process.
weights	an optional vector of weights to be used in the fitting process. Should be NULL or a numeric vector.
na.action	a function which indicates what should happen when the data contain NAs.
contrasts	an optional list. See the contrasts.arg of model.matrix.default.
method	the algorithm used to solve the linear program. See rq for further details. The Frisch-Newton interior point method is the default.
type	possible options are rq for linear quantile regression (default) or rqt for transformation-based quantile regression.

tsf	transformation to be used. Possible options are mcjI for Proposal I transformation models (default), bc for Box-Cox and ao for Aranda-Ordaz transformation models. See tsrq for further details.
symm	logical flag. If TRUE (default) a symmetric transformation is used.
dbounded	logical flag. If TRUE the response is assumed to be doubly bounded on [a,b]. If FALSE the response is assumed to be singly bounded (ie, strictly positive).
lambda	values of transformation parameters for grid search.
conditional	logical flag. If TRUE, the transformation parameter is assumed to be known and this must be provided via the arguments lambda using a vector of length 3 + 2 x length(probs) (see details).
...	other arguments for fun, rq or tsrq.

Details

This function computes a number of quantile-based summary statistics for location (median), scale (inter-quartile range and inter-quantile range), and shape (Bowley skewness and shape index) of a distribution. These statistics can be computed for unconditional and conditional distributions.

Let Y be a continuous random variable and let $Q(p)$ be its pth quantile. The function qlss computes the median $Q(0.5)$, the inter-quartile range $IQR = Q(0.75) - Q(0.25)$, the inter-quantile range $IPR(p) = Q(1-p) - Q(p)$, the Bowley skewness index $A(p) = (Q(1-p) + Q(p) - 2Q(0.5))/IPR(p)$, and the shape index $T(p) = IPR(p)/IQR$, for $0 < p < 0.25$.

The default qlss function computes the summary statistics of a standard normal distribution or any other theoretical distribution via the argument fun. The latter must be a function with p as its probability argument (see for example qnorm, qt, qchisq, qgamma, etc.). When a variable x is provided, LSS measures are computed using empirical (sample) quantiles.

The argument formula specifies a quantile function for Y conditional on predictors X. Linear models are fitted via standard quantile regression with type = "rq". Nonlinear models are fitted via transformation-based quantile regression with type = "rqt" (proposal II transformation models are not available.). When conditional = TRUE, lambda is a vector of transformation parameters of length 3 + 2 x np, where np = length(probs) (3 quartiles, np quantiles at level p, np quantiles at level $1-p$).

Value

qlss returns an object of class qlss. This is a list that contains at least three elements:

location	summary statistic(s) for location.
scale	summary statistic(s) for scale.
shape	summary statistic(s) for shape.

Author(s)

Marco Geraci

References

Geraci M and Jones MC. Improved transformation-based quantile regression. Canadian Journal of Statistics 2015;43(1):118-132.

Gilchrist W. Statistical modelling with quantile functions. Chapman and Hall/CRC; 2000.

See Also

predict.qlss, plot.qlss

Examples

```
# Compute summary statistics of a normal distribution
qlss()

# Compute summary statistics of a t distribution with 3 df
qlss(fun = "qt", df = 3, probs = 0.05)

# Compute summary statistics for a sample using a sequence of probabilities
x <- rnorm(1000)
qlss(x, probs = c(0.1, 0.2, 0.3, 0.4))

# Compute summary statistics for Volume conditional on Height
trees2 <- trees[order(trees$Height),]
fit <- qlss(Volume ~ Height, data = trees2)
plot(fit, z = trees2$Height, xlab = "height")

# Use a quadratic model for Height
fit <- qlss(Volume ~ poly(Height,2), data = trees2)
plot(fit, z = trees2$Height, xlab = "height")
```

---

qrr *Quantile Ratio Regression*

Description

This function fits a quantile ratio regression model

Usage

```
qrr(formula, data, taus, start = "rq", beta = NULL,
tsf = "bc", symm = TRUE, dbounded = FALSE, linearize = TRUE,
kernel = "Gaussian", maxIter = 10, epsilon = 1e-05,
verbose = FALSE, method.rq = "fn", method.nlrq = "L-BFGS-B")
```

Arguments

formula	a formula object, with the response on the left of a ~ operator, and the terms, separated by + operators, on the right.
data	a data frame in which to interpret the variables named in the formula.
taus	a vector of two quantiles for the ratio to be estimated (the order is irrelevant).
start	the algorithm with which obtain the starting values for one of the quantiles in the ratio. Possible options are "rq" (linear regression model – see rq), "tsrq" (quantile regression transformation model – see tsrq), "conquer" (fast linear regression model – see conquer), "llqr" (nonparametric linear regression model – see llqr)
beta	starting values for the regression coefficients. If left NULL, these are set to 0.
tsf	if start = "tsrq", see tsrq.
symm	if start = "tsrq", see tsrq.
dbounded	if start = "tsrq", see tsrq.
linearize	logical flag. If TRUE (default), estimation is carried out with the linearized iterative algorithm of Farcomeni and Geraci (2023) by repeated calls to an appropriate linear estimation algorithm. Otherwise, the algorithm calls a nonlinear estimation routine. See argument method.rq and method.nlrq further below.
kernel	an optional vector of weights to be used in the fitting process. Should be NULL or a numeric vector.
maxIter	maximum number of iterations for fitting.
epsilon	tolerance for convergence.
verbose	logical flag. If TRUE, progress on estimation is print out.
method.rq	the method used to compute the linear fit. If linearize = TRUE, the options are "conquer" or any of those from rq (see the argument method).
method.nlrq	the method used to compute the nonlinear fit. If linearize = FALSE, the options are those from nlrq (see the argument method).

Details

These function implements quantile ratio regression as discussed by Farcomeni and Geraci (see references). The general model is assumed to be $g(Q_{Y|X}(\tau_1)/Q_{Y|X}(\tau_2)) = \eta = Xb$ where Q denotes the conditional quantile function, Y is the response variable, X a design matrix, g is a monotone link function, and τ_1 and τ_2 the levels of the two quantiles in the ratio. In the current implementation, $g(u) = log(u - 1)$, which ensures monotonocity (non-crossing) of the quantiles and leads to the familiar interpretation of the inverse logistic transformation.

Author(s)

Marco Geraci

References

Farcomeni A. and Geraci M. Quantile ratio regression. 2023. Working Paper.

See Also

coef.qrr, predict.qrr, summary.qrr, vcov.qrr

Examples

```
set.seed(123)
n <- 5000
x <- runif(n, -0.5, 0.5)
R <- 1 + exp(0.5 + 0.5*x)

# fit quintile ratio regression
alpha <- 1/log(R)*log(log(1-0.8)/log(1-0.2))
y <- rweibull(n, shape = alpha, scale = 1)
dd <- data.frame(x = x, y = y)
qrr(y ~ x, data = dd, taus = c(.2,.8))

# fit Palma ratio regression
alpha <- 1/log(R)*log(log(1-0.9)/log(1-0.4))
y <- rweibull(n, shape = alpha, scale = 1)
dd <- data.frame(x = x, y = y)
qrr(y ~ x, data = dd, taus = c(.4,.9))
```

---

residuals.midrq *Residuals from a midrq Objects*

Description

This function computes the residuals from a fitted mid-quantile regression model.

Usage

```
## S3 method for class 'midrq'
residuals(object, ...)
```

Arguments

object	an midrq object.
...	not used.

Value

a vector or matrix of residuals.

Author(s)

Marco Geraci

See Also

midrq

`residuals.rq.counts` *Residuals from an rq.counts Object*

Description

This function computes the residuals from a fitted linear quantile model for counts.

Usage

```
## S3 method for class 'rq.counts'
residuals(object, ...)
```

Arguments

`object`	an `rq.counts` object.
`...`	not used.

Value

a vector or matrix of residuals.

Author(s)

Marco Geraci

See Also

rq.counts

`residuals.rqt` *Residuals from an rqt Objects*

Description

This function computes the residuals from a fitted quantile regression transformation model.

Usage

```
## S3 method for class 'rqt'
residuals(object, ...)
```

Arguments

object	an rqt object.
...	not used.

Value

a vector or matrix of residuals.

Author(s)

Marco Geraci

See Also

tsrq

---

rq.counts *Quantile Regression for Counts*

Description

This function is used to fit a (log-linear) quantile regression model when the response is a count variable.

Usage

```
rq.counts(formula, data = sys.frame(sys.parent()), tau = 0.5, subset, weights,
na.action, contrasts = NULL, offset = NULL, method = "fn", M = 50,
zeta = 1e-5, B = 0.999, cn = NULL, alpha = 0.05)
```

Arguments

formula	an object of class formula: a symbolic description of the model to be fitted.
data	an optional data frame, list or environment (or object coercible by as.data.frame to a data frame) containing the variables in the model. If not found in data, the variables are taken from environment(formula), typically the environment from which rq.counts is called.
tau	quantile to be estimated.
subset	an optional vector specifying a subset of observations to be used in the fitting process.
weights	an optional vector of weights to be used in the fitting process.
na.action	a function which indicates what should happen when the data contain NAs.
contrasts	an optional list. See the contrasts.arg of model.matrix.default.
offset	an optional offset to be included in the model frame.

method	estimation method for the fitting process. See rq.
M	number of dithered samples.
zeta	small constant (see References).
B	right boundary for uniform random noise U[0,B] to be added to the response variable (see References).
cn	small constant to be passed to Fn (see Theorem 3, Machado and Santos Silva).
alpha	significance level.

Details

A linear quantile regression model is fitted to the log–transformed response. The notation used here follows closely that of Machado and Santos Silva (2005). This function is based on routines from package quantreg (Koenker, 2016). See also lqm.counts from package lqmm (Geraci, 2014) for Laplace gradient estimation.

As of version 1.4, the transformation of the response cannot be changed. This option may be reinstated in future versions.

Value

a list of class rq.counts containing the following components

call	the matched call.
method	the fitting algorithm for rq.
x	the model matrix.
y	the model response.
tau	the order of the estimated quantile(s).
tsf	tranformation used (see also attributes(tsf)).
coefficients	regression quantile (on the log–scale).
fitted.values	fitted values (on the response scale).
tTable	coefficients, standard errors, etc.
offset	offset.
M	specified number of dithered samples for standard error estimation.
Mn	actual number of dithered samples used for standard error estimation that gave an invertible D matrix (Machado and Santos Silva, 2005).
InitialPar	starting values for coefficients.
terms	the terms object used.
term.labels	names of coefficients.
rdf	the number of residual degrees of freedom.

Author(s)

Marco Geraci

References

Geraci M. Linear quantile mixed models: The lqmm package for Laplace quantile regression. Journal of Statistical Software. 2014;57(13):1-29.

Geraci M and Jones MC. Improved transformation-based quantile regression. Canadian Journal of Statistics 2015;43(1):118-132.

Koenker R. quantreg: Quantile Regression. 2016. R package version 5.29.

Machado JAF, Santos Silva JMC. Quantiles for counts. Journal of the American Statistical Association. 2005;100(472):1226-37.

See Also

`residuals.rq.counts`, `predict.rq.counts`, `coef.rq.counts`, `maref.rq.counts`

Examples

```
# Esterase data
data(esterase)

# Fit quantiles 0.25 and 0.75
fit1 <- rq.counts(Count ~ Esterase, tau = 0.25, data = esterase, M = 50)
coef(fit1)
fit2 <- rq.counts(Count ~ Esterase, tau = 0.75, data = esterase, M = 50)
coef(fit2)

# Plot
with(esterase, plot(Count ~ Esterase))
lines(esterase$Esterase, fit1$fitted.values, col = "blue")
lines(esterase$Esterase, fit2$fitted.values, col = "red")
legend(8, 1000, lty = c(1,1), col = c("blue", "red"), legend = c("tau = 0.25","tau = 0.75"))
```

---

rrq *Restricted Regression Quantiles*

Description

This function fits a restricted quantile regression model to avoid crossing of quantile curves.

Usage

```
rrq(formula, tau, data, subset, weights, na.action, method = "fn",
model = TRUE, contrasts = NULL, ...)
rrq.fit(x, y, tau, method = "fn", ...)
rrq.wfit(x, y, tau, weights, method = "fn", ...)
```

Arguments

formula	a formula object, with the response on the left of a ~ operator, and the terms, separated by + operators, on the right.
x	the design matrix.
y	the response variable.
tau	the quantile(s) to be estimated.
data	a data frame in which to interpret the variables named in the formula.
subset	an optional vector specifying a subset of observations to be used in the fitting process.
weights	an optional vector of weights to be used in the fitting process. Should be NULL or a numeric vector.
na.action	a function which indicates what should happen when the data contain NAs.
method	the algorithm used to compute the fit (see rq).
model	if TRUE then the model frame is returned. This is essential if one wants to call summary subsequently.
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.
...	optional arguments passed to rq.fit or rq.wfit.

Author(s)

Marco Geraci

References

He X. Quantile curves without crossing. The American Statistician 1997;51(2):186-192.

Koenker R. quantreg: Quantile Regression. 2016. R package version 5.29.

Examples

```
data(esterase)

# Fit standard quantile regression
fit <- quantreg::rq(Count ~ Esterase, data = esterase, tau = c(.1,.25,.5,.75,.9))
yhat <- fit$fitted.values

# Fit restricted quantile regression
fitr <- rrq(Count ~ Esterase, data = esterase, tau = c(.1,.25,.5,.75,.9))
yhat2 <- predict(fitr)


# Plot results
par(mfrow = c(1, 2))
```

```
# Plot regression quantiles
with(esterase, plot(Count ~ Esterase, pch = 16, cex = .8))
apply(yhat, 2, function(y,x) lines(x,y,lwd = 1.5), x = esterase$Esterase)

# Plot restricted regression quantiles
with(esterase, plot(Count ~ Esterase, pch = 16, cex = .8))
apply(yhat2, 2, function(y,x) lines(x,y,lwd = 1.5), x = esterase$Esterase)
```

`sparsity.rqt` *Sparsity Estimation*

Description

This function estimates the density and sparsity functions of the residuals from a `rq` or a `rqt` object.

Usage

```
sparsity(object, se = "nid", hs = TRUE)
## S3 method for class 'rq'
sparsity(object, se = "nid", hs = TRUE)
## S3 method for class 'rqs'
sparsity(object, se = "nid", hs = TRUE)
## S3 method for class 'rqt'
sparsity(object, se = "nid", hs = TRUE)
```

Arguments

`object`	a `rq`, `rqs` or `rqt` object.
`se`	"iid" if errors are assumed independent and identically distributed; "nid" (default) if independent but not identically distributed; "ker" which uses a kernel estimate of the sandwich as proposed by Powell (1991).
`hs`	logical flag. If `TRUE` (default) the Hall-Sheather rule is used. Otherwise, the Bofinger's rule is used.

Details

This function is based on the code from `quantreg::summary.rq` and `quantreg::bandwidth.rq` to estimate the sparsity function for linear quantile regression models (Koenker and Bassett, 1978) and transformation models of Geraci and Jones (2014).

Value

`sparsity` returns an object of `class` `list` that contains three elements:

`density`	estimate of the density of the residuals.
`sparsity`	estimate of the sparsity of the residuals.
`bandwidth`	bandwidth used for estimation.

Author(s)

Marco Geraci

References

Geraci M and Jones MC. Improved transformation-based quantile regression. Canadian Journal of Statistics 2015;43(1):118-132.

Koenker R. quantreg: Quantile Regression. 2016. R package version 5.29.

Koenker R, Bassett G. Regression quantiles. Econometrica. 1978;46(1):33-50.

Powell JL. Estimation of monotonic regression models under quantile restrictions. In: Barnett W, Powell J, Tauchen G, editors. Nonparametric and Semiparametric Methods in Econometrics and Statistics: Proceedings of the Fifth International Symposium on Economic Theory and Econometrics. New York, NY: Cambridge University Press 1991. p. 357-84.

See Also

rq

Examples

```
## Not run:

data(trees)

# 'rqt' object

fit.rqt <- tsrq(Volume ~ Height, tsf = "bc", symm = FALSE, data = trees,
lambda = seq(-10, 10, by = 0.01), tau = 0.5)
sparsity(fit.rqt)

# 'rq' object

fit.rq <- rq(Volume ~ Height, data = trees)
sparsity(fit.rq, se = "iid")
sparsity(fit.rq, se = "nid")
sparsity(fit.rq, se = "ker")

## End(Not run)
```

---

summary.midrq *Summary for Mid-Quantile Regression Models*

Description

This functions gives a summary list for a mid-quantile regression model.

Usage

```
## S3 method for class 'midrq'
summary(object, alpha = 0.05, numerical = FALSE, robust = FALSE, ...)
```

Arguments

object	an object of class midrq.
alpha	numeric value to determine the confidence level (1-alpha) of the required interval.
numerical	logical flag. If TRUE, the variance-covariance estimate is approximated by the inverse of the numerical Hessian.
robust	logical flag. If TRUE, the Huber-White covariance estimate is computed using the Huberized residuals.
...	not used.

Author(s)

Marco Geraci

References

Geraci, M. and A. Farcomeni. Mid-quantile regression for discrete responses. arXiv:1907.01945 [stat.ME]. URL: https://arxiv.org/abs/1907.01945.

See Also

midrq

---

summary.qrr *Summary for Quantile Ratio Regression Models*

Description

This functions gives a summary list for a quantile ratio regression model.

Usage

```
## S3 method for class 'qrr'
summary(object, se = "approximate", R = 200,
update = TRUE, ...)
```

Arguments

object	an object of class summary.qrr.
se	specifies the method used to compute standard errors. See argument method in vcov.qrr.
R	number of bootstrap replications.
update	see argument update in vcov.qrr.
...	not used.

Author(s)

Marco Geraci

References

Farcomeni A. and Geraci M. Quantile ratio regression. 2023. Working Paper.

See Also

qrr

---

summary.rqt *Summary for Quantile Regression Tranformation Models*

Description

This functions gives a summary list for a quantile regression transformation model.

Usage

```
## S3 method for class 'rqt'
summary(object, alpha = 0.05, se = "boot", R = 50,
sim = "ordinary", stype = "i", conditional = FALSE, ...)
```

Arguments

object	an object of class rqt.
alpha	numeric value to determine the confidence level (1-alpha) of the required interval.
se	specifies the method used to compute standard errors. For conditional inference (conditional = TRUE), see argument se in summary.rq. For unconditional inference (conditional = FALSE), see details below.
R	number of bootstrap replications.
sim	see argument sim in boot.
stype	see argument stype in boot.

conditional	logical flag. If TRUE, the transformation parameter is assumed to be known and conditional inference is carried out.
...	if conditional = TRUE, additional arguments for summary.rq in package quantreg. If conditional = FALSE, additional arguments for boot in package boot.

Details

If inference is carried out conditionally on the transformation parameter (ie, assuming this is *known* rather than estimated), any type of summary for regression quantiles can be used (see summary.rq).

For unconditional inference (conditional = FALSE), there are three methods available: boot for bootstrap; iid for large-*n* approximation of the standard errors under IID assumptions; nid for large-*n* approximation of the standard errors under NID assumptions. See Powell (1991), Chamberlain (1994) and Geraci and Jones (2015).

Author(s)

Marco Geraci

References

Canty A and Ripley B (2014). boot: Bootstrap R (S-Plus) Functions. R package version 1.3-11.

Chamberlain G. Quantile regression, censoring, and the structure of wages. In: Sims C, editor. Advances in Econometrics: Sixth World Congress. 1. Cambridge, UK: Cambridge University Press; 1994.

Davison AC and Hinkley DV (1997). Bootstrap Methods and Their Applications. Cambridge University Press, Cambridge.

Geraci M and Jones MC. Improved transformation-based quantile regression. Canadian Journal of Statistics 2015;43(1):118-132.

Mu YM, He XM. Power transformation toward a linear regression quantile. Journal of the American Statistical Association 2007;102(477):269-279.

Powell JL. Estimation of monotonic regression models under quantile restrictions. In: Barnett W, Powell J, Tauchen G, editors. Nonparametric and Semiparametric Methods in Econometrics and Statistics: Proceedings of the Fifth International Symposium on Economic Theory and Econometrics. New York, NY: Cambridge University Press 1991. p. 357-84.

See Also

tsrq, rcrq, tsrq2 or nlrq2

summary.rrq *Summary for Restricted Quantile Regression Models*

Description

This functions gives a summary list for a restricted quantile regression model.

Usage

```
## S3 method for class 'rrq'
summary(object, alpha = 0.05, se = "boot", R = 50,
sim = "ordinary", stype = "i", ...)
```

Arguments

object	an object of class rrq.
alpha	numeric value to determine the confidence level (1-alpha) of the required interval.
se	specifies the method used to compute standard errors. Currently, bootstrap is the only method available.
R	number of bootstrap replications.
sim	see argument sim in boot.
stype	see argument stype in boot.
...	additional arguments for boot in package boot.

Details

A bootstrap approach is used for inference. Future developments of this function will include asymptotic standard errors.

Author(s)

Marco Geraci

References

Canty A and Ripley B (2014). boot: Bootstrap R (S-Plus) Functions. R package version 1.3-15.

Davison AC and Hinkley DV (1997). Bootstrap Methods and Their Applications. Cambridge University Press, Cambridge.

He X (1997). Quantile Curves without Crossing. The American Statistician, 51(2), 186-192.

tsrq *Quantile Regression Transformation Models*

Description

These functions are used to fit quantile regression transformation models.

Usage

```
tsrq(formula, data = sys.frame(sys.parent()), tsf = "mcjI", symm = TRUE,
dbounded = FALSE, lambda = NULL, conditional = FALSE, tau = 0.5,
subset,weights, na.action,contrasts = NULL, method = "fn")
tsrq2(formula, data = sys.frame(sys.parent()), dbounded = FALSE, lambda = NULL,
delta = NULL, conditional = FALSE, tau = 0.5, subset, weights, na.action,
contrasts = NULL, method = "fn")
rcrq(formula, data = sys.frame(sys.parent()), tsf = "mcjI", symm = TRUE,
dbounded = FALSE, lambda = NULL, tau = 0.5, subset, weights, na.action,
contrasts = NULL, method = "fn")
nlrq1(formula, data = sys.frame(sys.parent()), tsf = "mcjI", symm = TRUE,
dbounded = FALSE, start = NULL, tau = 0.5,
subset,weights, na.action, contrasts = NULL, control = list())
nlrq2(formula, data = sys.frame(sys.parent()), dbounded = FALSE,
start = NULL, tau = 0.5, subset, weights, na.action, contrasts = NULL)
```

Arguments

formula	an object of class "formula" (or one that can be coerced to that class): a symbolic description of the model to be fitted. The details of model specification are given under 'Details'.
data	an optional data frame, list or environment (or object coercible by as.data.frame to a data frame) containing the variables in the model. By default the variables are taken from the environment from which the call is made.
tsf	transformation to be used. Possible options are mcjI for Proposal I transformation models (default), bc for Box-Cox and ao for Aranda-Ordaz transformation models.
symm	logical flag. If TRUE (default) a symmetric transformation is used.
dbounded	logical flag. If TRUE the response is assumed to be doubly bounded on [a,b]. If FALSE (default) the response is assumed to be singly bounded (ie, strictly positive).
lambda, delta	values of transformation parameters for grid search.
conditional	logical flag. If TRUE, the transformation parameter is assumed to be known and this must be provided via the arguments lambda, delta in vectors of the same length as tau.

start	vector of length 1 + p (nlrq1) or 2 + p (nlrq2) of initial values for the parameters to be optimized over. The first one (nlrq1) or two (nlrq2) values for the transformation parameter lambda, or lambda and delta, while the last p values are for the regression coefficients. These initial values are passed to nl.fit.rqt or to optim.
control	list of control parameters of the fitting process (nlrq1). See nlControl.
tau	the quantile(s) to be estimated. See rq.
subset	an optional vector specifying a subset of observations to be used in the fitting process.
weights	an optional vector of weights to be used in the fitting process. Should be NULL or a numeric vector.
na.action	a function which indicates what should happen when the data contain NAs.
contrasts	an optional list. See the contrasts.arg of model.matrix.default.
method	fitting algorithm for rq (default is Frisch-Newton interior point method "fn").

Details

These functions implement quantile regression transformation models as discussed by Geraci and Jones (see references). The general model is assumed to be $Q_{h(Y)|X}(\tau) = \eta = Xb$, where Q denotes the conditional quantile function, Y is the response variable, X a design matrix, and h is a monotone one- or two-parameter transformation. A typical model specified in formula has the form response ~ terms where response is the (numeric) response vector and terms is a series of terms which specifies a linear predictor for the quantile of the transformed response. The response, which is singly or doubly bounded, i.e. response > 0 or 0 <= response <= 1 respectively, undergoes the transformation specified in tsf. If the response is bounded in the generic $[a, b]$ interval, the latter is automatically mapped to $[0, 1]$ and no further action is required. If, however, the response is singly bounded and contains negative values, it is left to the user to offset the response or the code will produce an error.

The functions tsrq and tsrq2 use a two-stage (TS) estimator (Fitzenberger et al, 2010) for, respectively, one- and two-parameter transformations. The function rcrq (one-parameter tranformations) is based on the residual cusum process estimator proposed by Mu and He (2007). The functions nlrq1 (one-parameter tranformations) and nlrq2 (two-parameter tranformations) are based on, respectively, gradient search and Nelder-Mead optimization.

Value

tsrq, tsrq2, rcrq, nlrq2 return an object of class rqt. This is a list that contains as typical components:

...	the first nt = length(tau) elements of the list store the results from fitting linear quantile models on the tranformed scale of the response.
call	the matched call.
method	the fitting algorithm for rq or optim.
y	the response – untransformed scale.
theta	if dbounded = TRUE, the response mapped to the unit interval.

x	the model matrix.
weights	the weights used in the fitting process (a vector of 1's if weights is missing or NULL).
tau	the order of the estimated quantile(s).
lambda	the estimated parameter lambda.
eta	the estimated parameters lambda and delta in the two-parameter Proposal II tranformation.
lambda.grid	grid of lambda values used for estimation.
delta.grid	grid of delta values used for estimation.
tsf	tranformation used (see also attributes(tsf)).
objective	values of the objective function minimised over the tranformation parameter(s). This is an array of dimension c(nl,nt) or c(nl,nd,nt), where nl = length(lambda.grid), nd = length(delta.grid) and nt = length(tau).
optimum	value of the objective function at solution.
coefficients	quantile regression coefficients – transformed scale.
fitted.values	fitted values.
rejected	proportion of inadmissible observations (Fitzenberger et al, 2010).
terms	the terms used.
term.labels	names of coefficients.
rdf	residual degrees of freedom.

Author(s)

Marco Geraci

References

Aranda-Ordaz FJ. On two families of transformations to additivity for binary response data. Biometrika 1981;68(2):357-363.

Box GEP, Cox DR. An analysis of transformations. Journal of the Royal Statistical Society Series B-Statistical Methodology 1964;26(2):211-252.

Dehbi H-M, Cortina-Borja M, and Geraci M. Aranda-Ordaz quantile regression for student performance assessment. Journal of Applied Statistics. 2016;43(1):58-71.

Fitzenberger B, Wilke R, Zhang X. Implementing Box-Cox quantile regression. Econometric Reviews 2010;29(2):158-181.

Geraci M and Jones MC. Improved transformation-based quantile regression. Canadian Journal of Statistics 2015;43(1):118-132.

Jones MC. Connecting distributions with power tails on the real line, the half line and the interval. International Statistical Review 2007;75(1):58-69.

Koenker R. quantreg: Quantile Regression. 2016. R package version 5.29.

Mu YM, He XM. Power transformation toward a linear regression quantile. Journal of the American Statistical Association 2007;102(477):269-279.

See Also

predict.rqt, summary.rqt, coef.rqt, maref.rqt

Examples

```
############################################################
## Example 1 - singly bounded (from Geraci and Jones, 2014)

## Not run:

data(trees)
require(MASS)

dx <- 0.01

lambda0 <- boxcox(Volume ~ log(Height), data = trees,
lambda = seq(-0.9, 0.5, by = dx))
lambda0 <- lambda0$x[which.max(lambda0$y)]
trees$z <- bc(trees$Volume,lambda0)
trees$y <- trees$Volume
trees$x <- log(trees$Height)
trees$x <- trees$x - mean(log(trees$Height))

fit.lm <- lm(z ~ x, data = trees)
newd <- data.frame(x = log(seq(min(trees$Height),
max(trees$Height), by = 0.1)))
newd$x <- newd$x - mean(log(trees$Height))
ylm <- invbc(predict(fit.lm, newdata = newd), lambda0)

lambdas <- list(bc = seq(-10, 10, by=dx),
mcjIs = seq(0,10,by = dx), mcjIa = seq(0,20,by = dx))

taus <- 1:3/4
fit0 <- tsrq(y ~ x, data = trees, tsf = "bc", symm = FALSE,
lambda = lambdas$bc, tau = taus)
fit1 <- tsrq(y ~ x, data = trees, tsf = "mcjI", symm = TRUE,
dbounded = FALSE, lambda = lambdas$mcjIs, tau = taus)
fit2 <- tsrq(y ~ x, data = trees, tsf = "mcjI", symm = FALSE,
dbounded = FALSE, lambda = lambdas$mcjIa, tau = taus)


par(mfrow = c(1,3), mar = c(7.1, 7.1, 5.1, 2.1), mgp = c(5, 2, 0))

cx.lab <- 2.5
cx.ax <- 2
lw <- 2
cx <- 2
xb <- "log(Height)"
yb <- "Volume"
xl <- range(trees$x)
yl <- c(5,80)
```

```
yhat <- predict(fit0, newdata = newd)
plot(y ~ x, data = trees, xlim = xl, ylim = yl, main = "Box-Cox",
cex.lab = cx.lab, cex.axis = cx.ax, cex.main = cx.lab,
cex = cx, xlab = xb, ylab = yb)
lines(newd$x, yhat[,1], lwd = lw)
lines(newd$x, yhat[,2], lwd = lw)
lines(newd$x, yhat[,3], lwd = lw)
lines(newd$x, ylm, lwd = lw, lty = 2)

yhat <- predict(fit1, newdata = newd)
plot(y ~ x, data = trees, xlim = xl, ylim = yl, main = "Proposal I (symmetric)",
cex.lab = cx.lab, cex.axis = cx.ax, cex.main = cx.lab,
cex = cx, xlab = xb, ylab = yb)
lines(newd$x, yhat[,1], lwd = lw)
lines(newd$x, yhat[,2], lwd = lw)
lines(newd$x, yhat[,3], lwd = lw)
lines(newd$x, ylm, lwd = lw, lty = 2)

yhat <- predict(fit2, newdata = newd)
plot(y ~ x, data = trees, xlim = xl, ylim = yl, main = "Proposal I (asymmetric)",
cex.lab = cx.lab, cex.axis = cx.ax, cex.main = cx.lab,
cex = cx, xlab = xb, ylab = yb)
lines(newd$x, yhat[,1], lwd = lw)
lines(newd$x, yhat[,2], lwd = lw)
lines(newd$x, yhat[,3], lwd = lw)
lines(newd$x, ylm, lwd = lw, lty = 2)

## End(Not run)

############################################################
## Example 2 - doubly bounded

## Not run:

data(Chemistry)

Chemistry$gcse_gr <- cut(Chemistry$gcse, c(0,seq(4,8,by=0.5)))
with(Chemistry, plot(score ~ gcse_gr, xlab = "GCSE score",
ylab = "A-level Chemistry score"))


# The dataset has > 31000 observations and computation can be slow
set.seed(178)
chemsub <- Chemistry[sample(1:nrow(Chemistry), 2000), ]

# Fit symmetric Aranda-Ordaz quantile 0.9
tsrq(score ~ gcse, data = chemsub, tsf = "ao", symm = TRUE,
lambda = seq(0,2,by=0.01), tau = 0.9)

# Fit symmetric Proposal I quantile 0.9
tsrq(score ~ gcse, data = chemsub, tsf = "mcjI", symm = TRUE,
dbounded = TRUE, lambda = seq(0,2,by=0.01), tau = 0.9)
```

```
# Fit Proposal II quantile 0.9 (Nelder-Mead)
nlrq2(score ~ gcse, data = chemsub, dbounded = TRUE, tau = 0.9)

# Fit Proposal II quantile 0.9 (grid search)
# This is slower than nlrq2 but more stable numerically
tsrq2(score ~ gcse, data = chemsub, dbounded = TRUE,
lambda = seq(0, 2, by = 0.1), delta = seq(0, 2, by = 0.1),
tau = 0.9)


## End(Not run)

############################################################
## Example 3 - doubly bounded

data(labor)

new <- labor
new$y <- new$pain
new$x <- (new$time-30)/30
new$x_gr <- as.factor(new$x)

par(mfrow = c(2,2))

cx.lab <- 1
cx.ax <- 2.5
cx <- 2.5
yl <- c(0,0.06)

hist(new$y[new$treatment == 1], xlab = "Pain score", main = "Medication group",
freq = FALSE, ylim = yl)

plot(y ~ x_gr, new, subset = new$treatment == 1, xlab = "Time (min)",
ylab = "Pain score", axes = FALSE, range = 0)
axis(1, at = 1:6, labels = c(0:5)*30 + 30)
axis(2)
box()

hist(new$y[new$treatment == 0], xlab = "Pain score", main = "Placebo group",
freq = FALSE, ylim = yl)

plot(y ~ x_gr, new, subset = new$treatment == 0, xlab = "Time (min)",
ylab = "Pain score", axes = FALSE, range = 0)
axis(1, at = 1:6, labels = (0:5)*30 + 30)
axis(2)
box()

#

## Not run:

taus <- c(1:3/4)
ls <- seq(0,3.5,by=0.1)
```

```
fit.aos <- tsrq(y ~ x*treatment, data = new, tsf = "ao", symm = TRUE,
dbounded = TRUE, tau = taus, lambda = ls)
fit.aoa <- tsrq(y ~ x*treatment, data = new, tsf = "ao", symm = FALSE,
dbounded = TRUE, tau = taus, lambda = ls)
fit.mcjs <- tsrq(y ~ x*treatment, data = new, tsf = "mcjI", symm = TRUE,
dbounded = TRUE, tau = taus, lambda = ls)
fit.mcja <- tsrq(y ~ x*treatment, data = new, tsf = "mcjI", symm = FALSE,
dbounded = TRUE, tau = taus, lambda = ls)
fit.mcj2 <- tsrq2(y ~ x*treatment, data = new, dbounded = TRUE, tau = taus,
lambda = seq(0,2,by=0.1), delta = seq(0,1.5,by=0.3))
fit.nlrq <- nlrq2(y ~ x*treatment, data = new, start = coef(fit.mcj2, all = TRUE)[,1],
dbounded = TRUE, tau = taus)

sel <- 0 # placebo (change to sel == 1 for medication group)
x <- new$x
nd <- data.frame(x = seq(min(x), max(x), length=200), treatment = sel)
xx <- nd$x+1

par(mfrow = c(2,2))

fit <- fit.aos
yhat <- predict(fit, newdata = nd)

plot(y ~ x_gr, new, subset = new$treatment == sel, xlab = "",
ylab = "Pain score", axes = FALSE, main = "Aranda-Ordaz (s)",
range = 0, col = grey(4/5))
apply(yhat, 2, function(y,x) lines(x, y, lwd = 2), x = xx)
axis(1, at = 1:6, labels = (0:5)*30 + 30)
axis(2, at = c(0, 25, 50, 75, 100))
box()

fit <- fit.aoa
yhat <- predict(fit, newdata = nd)

plot(y ~ x_gr, new, subset = new$treatment == sel, xlab = "", ylab = "",
axes = FALSE, main = "Aranda-Ordaz (a)", range = 0, col = grey(4/5))
apply(yhat, 2, function(y,x) lines(x, y, lwd = 2), x = xx)
axis(1, at = 1:6, labels = (0:5)*30 + 30)
axis(2, at = c(0, 25, 50, 75, 100))
box()

fit <- fit.mcjs
yhat <- predict(fit, newdata = nd)

plot(y ~ x_gr, new, subset = new$treatment == sel, xlab = "Time (min)",
ylab = "Pain score", axes = FALSE, main = "Proposal I (s)",
range = 0, col = grey(4/5))
apply(yhat, 2, function(y,x) lines(x, y, lwd = 2), x = xx)
axis(1, at = 1:6, labels = (0:5)*30 + 30)
axis(2, at = c(0, 25, 50, 75, 100))
box()
```

```
fit <- fit.mcj2
yhat <- predict(fit, newdata = nd)

plot(y ~ x_gr, new, subset = new$treatment == sel, xlab = "Time (min)",
ylab = "", axes = FALSE, main = "Proposal II", range = 0, col = grey(4/5))
apply(yhat, 2, function(y,x) lines(x, y, lwd = 2), x = xx)
axis(1, at = 1:6, labels = (0:5)*30 + 30)
axis(2, at = c(0, 25, 50, 75, 100))
box()

## End(Not run)
```

`vcov.midrq` *Variance-Covariance Matrix for a Fitted Mid-Quantile Regression Model Object*

Description

This functions returns the variance-covariance matrix of the main parameters of a fitted `midrq` model object. The 'main' parameters of the model correspond to those returned by `coef`.

Usage

```
## S3 method for class 'midrq'
vcov(object, numerical = FALSE, robust = FALSE, ...)
```

Arguments

`object`	an object of `class` `midrq`.
`numerical`	logical flag. If `TRUE`, the variance-covariance estimate is approximated by the inverse of the numerical Hessian.
`robust`	logical flag. If `TRUE`, the Huber-White covariance estimate is computed using the Huberized residuals.
`...`	not used.

Author(s)

Marco Geraci with contributions from Alessio Farcomeni

References

Geraci, M. and A. Farcomeni. Mid-quantile regression for discrete responses. arXiv:1907.01945 [stat.ME]. URL: https://arxiv.org/abs/1907.01945.

See Also

`midrq`

vcov.qrr *Variance-Covariance Matrix for a Fitted Quantile Ratio Regression Model Object*

Description

This functions returns the variance-covariance matrix of the coefficients of a fitted qrr model object.

Usage

```
## S3 method for class 'qrr'
vcov(object, method = "approximate", R = 200, update = TRUE, ...)
```

Arguments

object	an object of class qrr.
method	if "approximate", the variance-covariance estimate is approximated by the inverse of the numerical Hessian. The latter is calculated as detailed in Farcomeni and Geraci (2023). If "boot", the variance-covariance estimate is calculated by means of ordinary bootstrap (see boot).
R	the number of bootstrap replications.
update	logical flag. If TRUE (the default), the statistic to be resampled is obtained via an update of the qrr object. If FALSE, then the statistic to be resampled is obtained via a do.call of the qrr object. See details.
...	not used.

Details

The use of update = FALSE is preferred when the function vcov.qrr is called from within another function.

Author(s)

Marco Geraci with contributions from Alessio Farcomeni

References

Farcomeni A. and Geraci M. Quantile ratio regression. 2023. Working Paper.

See Also

qrr

Index