

# Package ‘cdfquantreg’

July 4, 2021

**Type** Package

**Title** Quantile Regression for Random Variables on the Unit Interval

**Version** 1.2.5

**Date** 2021-07-05

**Description** Employs a two-parameter family of distributions for modelling random variables on the (0, 1) interval by applying the cumulative distribution function (cdf) of one parent distribution to the quantile function of another.

**BugReports** <https://ummlab.wordpress.com/resources/cdfquantreg-bugs-report/>

**Depends** R (>= 3.1.0)

**License** GPL-3

**Imports** pracma (>= 2.3), Formula (>= 1.2), stats, MASS

**Suggests** knitr, rmarkdown

**VignetteBuilder** knitr

**LazyData** true

**RoxygenNote** 7.1.1

**NeedsCompilation** no

**Author** Yiyun Shou [aut, cre],  
Michael Smithson [aut]

**Maintainer** Yiyun Shou <yiyun.shou@anu.edu.au>

**Repository** CRAN

**Date/Publication** 2021-07-04 08:10:02 UTC

## R topics documented:

cdfquantreg-package . . . . .	2
Ambdata . . . . .	3
anova.cdfqr . . . . .	4
AnxStrData . . . . .	5
bugsLikelihood . . . . .	5

bugsModel	6
cdfqr.control	7
cdfqrFamily	7
cdfquantreg	9
cdfquantregC	11
cdfquantregH	13
ExtEvent	15
influence.cdfqr	15
IPCC	17
IPCCAUS	18
IPCC_Wide	19
JurorData	19
plot.cdfqr	20
pq	21
predict.cdfqr	22
qrBoot	23
qrGrad	24
qrLogLik	24
qrLogLikFun	25
qrPwlm	25
qrStart	26
residuals.cdfqr	27
scaleTR	28
summary.cdfqr	29
summary.cdfqrH	30
<b>Index</b>	<b>33</b>

---

cdfquantreg-package     *Quantile Regression for Random Variables on the Unit Interval*

---

## Description

Employs a two-parameter family of distributions for modelling random variables on the  $(0, 1)$  interval by applying the cumulative distribution function (cdf) of one parent distribution to the quantile function of another.

## Details

Package: cdfquantreg  
 Type: Package  
 Version: 1.1.0  
 Date: 2016-09-10  
 License: GPL-3

The `cdfquantreg` package includes 36 members of a two-parameter family of distributions for modelling random variables on the (0, 1) interval (see [cdfqrFamily](#)). This family has explicit pdfs, cdfs, and quantile functions. The two parameters consist of a location parameter and a dispersion parameter. The location parameter models the median and the dispersion parameter models the spread of other quantiles around the median (see Smithson and Shou, 2016, for details about the distribution family and the models). Separate submodels may be specified for the location and for the dispersion parameters, permitting different or overlapping sets of predictors in each.

The package offers maximum likelihood (see [cdfquantreg](#)), Bayesian MCMC (see [bugsModel](#)), and bootstrap (see [qrBoot](#)) estimation methods. All model functions return S3 objects. The `bugsModel()` function runs OpenBUGS in R. In addition to the usual goodness of fit information, the package provides root-mean-squared errors in both the raw and logit scales, and the gradient. Model diagnostics include raw, Pearson, and deviance residuals (see [residuals.cdfqr](#)), and `dfbetas` (see [influence.cdfqr](#)).

For each distribution, the package provides evaluations of the pdf (`dq`), cdf (`pq`), and quantile (`qq`), as well as random samples from any of them (`rq`). Evaluations of skew and kurtosis (`qrPwlm`) also are available using probability-weighted L-moments.

### Author(s)

Yiyun Shou (<yiyun.shou@anu.edu.au>) and Michael Smithson (<Michael.Smithson@anu.edu.au>)

Maintainer: Yiyun Shou

### References

Shou, Y. and Smithson, M., (2019). `cdfquantreg`: An R Package for CDF-Quantile Regression. *Journal of Statistical Software*, 88(1), pp.1–30, doi: 10.18637/jss.v088.i01

### See Also

[cdfqrFamily](#)

---

Ambdata

*Ambiguity-Conflict data*

---

### Description

A data from a study that investigates the judgment under ambiguity and conflict

### Usage

Ambdata

**Format**

A data frame with 166 rows and 2 variables:

**ID** subject ID

**value** Rating in each judgment scenario

**scenario** Index for judgment scenarios

**Source**

<https://psycnet.apa.org/record/2006-03820-004>

---

anova.cdfqr	<i>Model comparison test for fitted cdfqr models</i>
-------------	--

---

**Description**

Likelihood Ratio Tests for fitted cdfqr Objects.

**Usage**

```
## S3 method for class 'cdfqr'
anova(object, ..., test = "LRT")
```

```
## S3 method for class 'cdfqrH'
anova(object, ..., test = "LRT")
```

**Arguments**

object	The fitted cdfqr model.
...	One or more cdfqr model objects for model comparison.
test	The model comparison test, currently only 'LRT' is implemented.

**Examples**

```
data(cdfqrExampleData)
fit_null <- cdfquantreg(crc99 ~ 1 | 1, 't2','t2', data = JurorData)
fit_mod1 <- cdfquantreg(crc99 ~ vert | conf1, 't2','t2', data = JurorData)
anova(fit_null, fit_mod1)
```

---

AnxStrData	<i>Stress-Anxiety data</i>
------------	----------------------------

---

**Description**

A data from a study that investigates the relationship between stress and anxiety.

**Usage**

AnxStrData

**Format**

A data frame with 166 rows and 2 variables:

**Anxiety** Scores on Anxiety subscale

**Stress** Scores on Stress subscale

**Source**

<https://psycnet.apa.org/record/2006-03820-004>

---

bugsLikelihood	<i>Likelihood Functions for Generating OpenBUGS Model File</i>
----------------	--

---

**Description**

Likelihood functions for generating OpenBUGS model file.

**Usage**

```
bugsLikelihood(fd, sd)
```

**Arguments**

fd                    A string that specifies the parent distribution.

sd                    A string that specifies the sub-family distribution.

**Value**

A string to be written in the BUGS model file.

**Examples**

```
bugsLikelihood('t2', 't2')
```

bugsModel

*Generating OpenBUGS Model File***Description**

Generating OpenBUGS model file

**Usage**

```
bugsModel(formula, fd, sd, random = NULL, modelName = "bugmodel", wd = getwd())
```

**Arguments**

formula	A formula object, with the DV on the left of an ~ operator, and predictors on the right. For the part on the right of '~', the specification of submodels can be separated by ' '. So $y \sim X1   X2$ means the DV is y, X1 is the term in the mean submodel, and X2 is the term in the dispersion submodel.
fd	A string that specifies the parent distribution (see <a href="#">cdfqrFamily</a> ).
sd	A string that specifies the sub-family distribution.
random	Character or vector of characters that indicates the random effect factors.
modelName	The name of the model file; optional.
wd	The working directory in which OpenBUGS will work (i.e., generate the model files and chain information).

**Value**

A model '.txt' file is generated in the specified working directory. The function also returns a list of values:

**init1,init2** Default initial values for MCMC two chain procedure.

**vars** A list of variables that are included in the estimation.

**nodes\_sample** a list of characters that specify the nodes to be monitored.

**Examples**

```
## Not run:
# Need write access in the working directory before executing the code.
# No random component
bugsModel(y ~ x1 | x2, 't2', 't2', random = NULL)
# Random component as subject ID
bugsModel(y ~ x1 | x2, 't2', 't2', random = 'ID')

## End(Not run)
```

---

cdfqr.control	<i>Control Optimization Parameters for CDF-Quantile Probability Distributions</i>
---------------	---

---

**Description**

Control Optimization Parameters for CDF-Quantile Probability Distributions.

**Usage**

```
cdfqr.control(method = "BFGS", maxit = 5000, trace = FALSE)
```

**Arguments**

method	Characters string specifying the method argument passed to <a href="#">optim</a> .
maxit	Integer specifying the maxit argument (maximal number of iterations) passed to <a href="#">optim</a> .
trace	Logical or integer controlling whether tracing information on the progress of the optimization should be produced

**Value**

A list with the arguments specified.

**Examples**

```
data(cdfqrExampleData)
fit <- cdfquantreg(crc99 ~ vert | conf1, 't2', 't2',
  data = JurorData, control = cdfqr.control(trace = TRUE))
```

---

cdfqrFamily	<i>Overview of the family of distributions</i>
-------------	--

---

**Description**

The cdfquantreg family consists of the currently available distributions that can be used to fit quantile regression models via the cdfquantreg() function.

**Usage**

```
cdfqrFamily(shape = "all")
```

## Arguments

shape	To show all distributions or the set of distribution for a specific type of shape. Can be BM, TM,LL or FT for Bimodal, Trimodal, Logit-logistic or Finite-tailed shapes, respectively.
-------	--

## Details

The cdfquantreg package includes a two-parameter family of distributions for modeling random variables on the (0, 1) interval by applying the cumulative distribution function (cdf) of one “parent” distribution to the quantile function of another.

The naming of these distributions is “parent - child” or “fd - sd”, where “fd” is the parent distribution, and “sd” is the child distribution.

The distributions have four characteristic shapes: Logit-logistic, bimodal, trimodal, and finite-tailed. Here is the list of currently available distributions.

### Bimodal Shape Distributions

Distribution	R input	Alternative Input	Shape
Burr VII-ArcSinh	fd = "burr7", sd = "arcsinh"	family = "burr7-arcsinh"	Bimodal
Burr VII-Cauchy	fd = "burr7", sd = "cauchy"	family = "burr7-cauchy"	Bimodal
Burr VII-T2	fd = "burr7", sd = "t2"	family = "burr7-t2"	Bimodal
Burr VIII-ArcSinh	fd = "burr8", sd = "arcsinh"	family = "burr8-arcsinh"	Bimodal
Burr VIII-Cauchy	fd = "burr8", sd = "cauchy"	family = "burr8-cauchy"	Bimodal
Burr VIII-T2	fd = "burr8", sd = "t2"	family = "burr8-t2"	Bimodal
Logit-ArcSinh	fd = "logit", sd = "arcsinh"	family = "logit-arcsinh"	Bimodal
Logit-Cauchy	fd = "logit", sd = "cauchy"	family = "logit-cauchy"	Bimodal
Logit-T2	fd = "logit", sd = "t2"	family = "logit-t2"	Bimodal
T2-ArcSinh	fd = "t2", sd = "arcsinh"	family = "t2-arcsinh"	Bimodal
T2-Cauchy	fd = "t2", sd = "cauchy"	family = "t2-cauchy"	Bimodal

### Trimodal Shape Distributions

Distribution	R input	Alternative Input	Shape
ArcSinh-Burr VII	fd = "arcsinh", sd = "burr7"	family = "arcsinh-burr7"	Trimodal
ArcSinh-Burr VIII	fd = "arcsinh", sd = "burr8"	family = "arcsinh-burr8"	Trimodal
ArcSinh-Logistic	fd = "arcsinh", sd = "logistic"	family = "arcsinh-logistic"	Trimodal
ArcSinh-T2	fd = "arcsinh", sd = "t2"	family = "arcsinh-t2"	Trimodal
Cauchit-Burr VII	fd = "cauchit", sd = "burr7"	family = "cauchit-burr7"	Trimodal
Cauchit-Burr VIII	fd = "cauchit", sd = "burr8"	family = "cauchit-burr8"	Trimodal
Cauchit-Logistic	fd = "cauchit", sd = "logistic"	family = "cauchit-logistic"	Trimodal
Cauchit-T2	fd = "cauchit", sd = "t2"	family = "cauchit-t2"	Trimodal
T2-Burr VII	fd = "t2", sd = "burr7"	family = "t2-burr7"	Trimodal
T2-Burr VIII	fd = "t2", sd = "burr8"	family = "t2-burr8"	Trimodal
T2-Logistic	fd = "t2", sd = "logistic"	family = "t2-logistic"	Trimodal

### Logit-logistic Shape Distributions



Distribution	R input	Alternative Input	Shape
Burr VII-Burr VII	fd = "burr7", sd = "burr7"	family = "burr7-burr7"	Logit-logistic
Burr VII-Burr VIII	fd = "burr7", sd = "burr8"	family = "burr7-burr8"	Logit-logistic
Burr VII-Logistic	fd = "burr7", sd = "logistic"	family = "burr7-logistic"	Logit-logistic
Burr VIII-Burr VII	fd = "burr8", sd = "burr7"	family = "burr8-burr7"	Logit-logistic
Burr VIII-Burr VIII	fd = "burr8", sd = "burr8"	family = "burr8-burr8"	Logit-logistic
Burr VIII-Logistic	fd = "burr8", sd = "logistic"	family = "burr8-logistic"	Bimodal
Logit-Burr VII	fd = "logit", sd = "burr7"	family = "logit-burr7"	Logit-logistic
Logit-Burr VIII	fd = "logit", sd = "burr8"	family = "logit-burr8"	Logit-logistic
Logit-Logistic	fd = "logit", sd = "logistic"	family = "logit-logistic"	Logit-logistic

### Finite-tailed Shape Distributions

Distribution	R input	Alternative Input	Shape
ArcSinh-ArcSinh	fd = "arcsinh", sd = "arcsinh"	family = "arcsinh-arcsinh"	Finite-tailed
ArcSinh-Cauchy	fd = "arcsinh", sd = "cauchy"	family = "arcsinh-cauchy"	Finite-tailed
Cauchit-ArcSinh	fd = "cauchit", sd = "arcsinh"	family = "cauchit-arcsinh"	Finite-tailed
Cauchit-Cauchy	fd = "cauchit", sd = "cauchy"	family = "cauchit-cauchy"	Finite-tailed
T2-T2	fd = "t2", sd = "t2"	family = "t2-t2"	Finite-tailed

### Kumaraswamy Distribution

Distribution	R input	Alternative Input	Shape
Kumaraswamy	fd = "", sd = ""	family = "-"	

### Value

A list of distributions that are available in the current version of package.

### Examples

```
cdfqrFamily()
```

---

cdfquantreg

*CDF-Quantile Probability Distributions*

---

### Description

cdfquantreg is the main function to fit a cdf quantile regression with a variety of distributions.

**Usage**

```

cdfquantreg(
  formula,
  fd = NULL,
  sd = NULL,
  data,
  family = NULL,
  start = NULL,
  control = cdfqr.control(...),
  ...
)

```

**Arguments**

<code>formula</code>	A formula object, with the dependent variable (DV) on the left of an <code>~</code> operator, and predictors on the right. For the part on the right of <code>'~'</code> , the specification of the location and dispersion submodels can be separated by <code>' '</code> . So <code>y ~ X1   X2</code> specifies that the DV is <code>y</code> , <code>X1</code> is the predictor in the location submodel, and <code>X2</code> is the predictor in the dispersion submodel.
<code>fd</code>	A string that specifies the parent distribution.
<code>sd</code>	A string that specifies the child distribution.
<code>data</code>	The data in a <code>data.frame</code> format
<code>family</code>	If <code>'fd'</code> and <code>'sd'</code> are not provided, the name of a member of the family of distributions can be provided (See <a href="#">cdfqrFamily</a> for details of family functions)
<code>start</code>	The starting values for model fitting. If not provided, default values will be used.
<code>control</code>	Control optimization parameters (See <a href="#">cdfqr.control</a> )
<code>...</code>	Currently ignored.

**Details**

The `cdfquantreg` function fits a quantile regression model with a distributions from the cdf-quantile family selected by the user (Smithson and Shou, 2015). The model is specified in a two-part formula, one part containing the predictors of the location parameter, and the second part containing the predictors of the dispersion parameter. The models are fitted in two stages, the first of which uses the Nelder-Mead algorithm and the second of which takes the estimates from the first stage and applies the BFGS algorithm to refine the estimates.

**Value**

An object of class `cdfquantreg` will be returned. Generic functions such as [summary.print](#) (e.g., [print.cdfqr](#)) and [coef](#) can be used to extract output (see [summary.cdfqr](#) for more details about the generic functions that can be used). Class of object is a list with the following output:

**coefficients** A named vector of coefficients.

**residuals** Raw residuals, the difference between the fitted values and the data.

**fitted** The fitted values, including full model fitted values, fitted values for the mean component, and fitted values for the dispersion component.

**rmse** The model root mean squared errors

**rmseLogit** The root mean squared errors between the logit of the fitted values, and the logit of the response values.

**vcov** The variance-covariance matrix of the coefficient estimates.

**AIC, BIC** Akaike's Information Criterion and Bayesian Information Criterion.

**deviance** The deviance for the model.

### Examples

```
data(cdfqrExampleData)
fit <- cdfquantreg(crc99 ~ vert | conf1, fd ='t2',sd ='t2', data = JurorData)

summary(fit)
```

---

cdfquantregC

*Censored CDF-Quantile Probability Distributions*

---

### Description

cdfquantregC is the a function to fit a censored cdf quantile regression with a variety of distributions .

### Usage

```
cdfquantregC(
  formula,
  fd = NULL,
  sd = NULL,
  data,
  family = NULL,
  censor = "DB",
  c1 = NULL,
  c2 = NULL,
  start = NULL,
  control = cdfqr.control(...),
  ...
)
```

### Arguments

formula	A formula object, with the dependent variable (DV) on the left of an ~ operator, and predictors on the right. For the part on the right of '~', the specification of the location and dispersion submodels can be separated by ' '. So $y \sim X1   X2$ specifies that the DV is y, X1 is the predictor in the location submodel, and X2 is the predictor in the dispersion submodel.
fd	A string that specifies the parent distribution.

<code>sd</code>	A string that specifies the child distribution.
<code>data</code>	The data in a data.frame format
<code>family</code>	If ‘fd’ and ‘sd’ are not provided, the name of a member of the family of distributions can be provided (See <a href="#">cdfqrFamily</a> for details of family functions)
<code>censor</code>	A string variable to indicate how many censored point is used- only left censored ‘LC’, or only right-hand censored ‘RC’, or both sides ‘DB’.
<code>c1</code>	The left censored value, if NULL, the minimum value in the data will be used
<code>c2</code>	The right censored value, if NULL, the maximum value in the data will be used
<code>start</code>	The starting values for model fitting. If not provided, default values will be used.
<code>control</code>	Control optimization parameters (See <a href="#">cdfqr.control</a> )
<code>...</code>	Currently ignored.

### Details

The `cdfquantreg` function fits a quantile regression model with a distributions from the cdf-quantile family selected by the user (Smithson and Shou, 2015). The model is specified in a two-part formula, one part containing the predictors of the location parameter, and the second part containing the predictors of the dispersion parameter. The models are fitted in two stages, the first of which uses the Nelder-Mead algorithm and the second of which takes the estimates from the first stage and applies the BFGS algorithm to refine the estimates.

### Value

An object of class `cdfquantreg` will be returned. Generic functions such as [summary.print](#) (e.g., [print.cdfqr](#)) and [coef](#) can be used to extract output (see [summary.cdfqr](#) for more details about the generic functions that can be used). Class of object is a list with the following output:

**coefficients** A named vector of coefficients.

**residuals** Raw residuals, the difference between the fitted values and the data.

**fitted** The fitted values, including full model fitted values, fitted values for the mean component, and fitted values for the dispersion component.

**rmse** The model root mean squared errors

**rmseLogit** The root mean squared errors between the logit of the fitted values, and the logit of the response values.

**vcov** The variance-covariance matrix of the coefficient estimates.

**AIC, BIC** Akaike’s Information Criterion and Bayesian Information Criterion.

**deviance** The deviance for the model.

### Examples

```
data(cdfqrExampleData)
fit <- cdfquantregC(crc99 ~ vert | conf1, c1 = 0.001, c2= 0.999,
  fd = 't2',sd = 't2', data = JurorData)

summary(fit)
```

cdfquantregH

*Zero/One inflated CDF-Quantile Probability Distributions***Description**

`cdfquantregH` is the a function to fit a Zero/One inflated CDF-Quantile regression with a variety of distributions .

**Usage**

```
cdfquantregH(
  formula,
  zero.fo = ~1,
  one.fo = ~1,
  fd = NULL,
  sd = NULL,
  data,
  family = NULL,
  type = "ZI",
  start = NULL,
  control = cdfqr.control(...),
  ...
)
```

**Arguments**

<code>formula</code>	A formula object, with the dependent variable (DV) on the left of an <code>~</code> operator, and predictors on the right. For the part on the right of <code>'~'</code> , the specification of the location and dispersion submodels can be separated by <code>' '</code> . So <code>y ~ X1   X2</code> specifies that the DV is y, X1 is the predictor in the location submodel, and X2 is the predictor in the dispersion submodel.
<code>zero.fo</code>	A formula object to indicate the predictors for the zero component, only input as <code>~ predictors</code>
<code>one.fo</code>	A formula object to indicate the predictors for the one component, only input as <code>~ predictors</code>
<code>fd</code>	A string that specifies the parent distribution.
<code>sd</code>	A string that specifies the child distribution.
<code>data</code>	The data in a data.frame format
<code>family</code>	If <code>'fd'</code> and <code>'sd'</code> are not provided, the name of a member of the family of distributions can be provided (See <a href="#">cdfqrFamily</a> for details of family functions)
<code>type</code>	A string variable to indicate whether the model is zero-inflated <code>'ZI'</code> , or one-inflated <code>'OI'</code> , or zero-one inflated <code>'ZO'</code> .
<code>start</code>	The starting values for model fitting. If not provided, default values will be used.
<code>control</code>	Control optimization parameters (See <a href="#">cdfqr.control</a> )
<code>...</code>	Currently ignored.

## Details

The `cdfquantreg` function fits a quantile regression model with a distributions from the cdf-quantile family selected by the user (Smithson and Shou, 2015). The model is specified in a two-part formula, one part containing the predictors of the location parameter, and the second part containing the predictors of the dispersion parameter. The models are fitted in two stages, the first of which uses the Nelder-Mead algorithm and the second of which takes the estimates from the first stage and applies the BFGS algorithm to refine the estimates.

## Value

An object of class `cdfquantreg` will be returned. Generic functions such as `summary`, `print` (e.g., `print.cdfqr`) and `coef` can be used to extract output (see `summary.cdfqr` for more details about the generic functions that can be used). Class of object is a list with the following output:

**coefficients** A named vector of coefficients.

**residuals** Raw residuals, the difference between the fitted values and the data.

**fitted** The fitted values, including full model fitted values, fitted values for the mean component, and fitted values for the dispersion component.

**vcov** The variance-covariance matrix of the coefficient estimates.

**AIC, BIC** Akaike's Information Criterion and Bayesian Information Criterion.

## Examples

```
data(cdfqrExampleData)
# For one-inflated model
ipcc_high <- subset(IPCC, mid == 1 & high == 1 & prob!=0)
fit <- cdfquantregH(prob ~ valence | valence, one.fo = ~valence,
  fd = 't2', sd = 't2', type = "OI", data = ipcc_high)

summary(fit)

# For zero-inflated model
ipcc_low <- subset(IPCC, mid == 0 & high == 0 & prob!=1)
fit <- cdfquantregH(prob ~ valence | valence, zero.fo = ~valence,
  fd = 't2', sd = 't2', type = "ZI", data = ipcc_low)

# For zero & one-inflated model
ipcc_mid <- subset(IPCC, mid == 1 & high == 0)
fit <- cdfquantregH(prob ~ valence | valence, zero.fo = ~valence,
  one.fo = ~valence,
  fd = 't2', sd = 't2', type = "ZO", data = ipcc_mid)
```

---

 ExtEvent

*Extinction Study data-set*


---

**Description**

Probability of Human Extinction Study

**Usage**

ExtEvent

**Format**

A data frame with 1170 rows and 11 variables:

**ID** Subject ID

**gend** Gender of subjects, '0' is male, '1' is female

**nation** The nation of the participants come from

**UK** effect coding for nation

**IND** effect coding for nation

**political** political orientation of subjects

**format** The format of probability elicitation

**order** the order of probability judgement task.

**SECS\_6** Social conservatism question on attitude toward gun ownership.

**EQ1\_P** Probability estimates for general threats.

**EQ3\_P** Probability estimates for the greatest threat.

**Source**

<https://www.michaelsmithson.online/>

---

 influence.cdfqr

*Influence Diagnosis For Fitted Cdfqr Object*


---

**Description**

Influence Diagnosis (dfbetas) For Fitted Cdfqr Object

**Usage**

```
## S3 method for class 'cdfqr'
influence(
  model,
  method = "dfbeta",
  type = c("full", "location", "dispersion"),
  what = "full",
  plot = FALSE,
  id = FALSE,
  ...
)

## S3 method for class 'cdfqr'
dfbeta(model, type = c("full", "location", "dispersion"), what = "full", ...)

## S3 method for class 'cdfqr'
dfbetas(model, type = c("full", "location", "dispersion"), what = "full", ...)

## S3 method for class 'cdfqrH'
influence(
  model,
  method = "dfbeta",
  type = c("full", "location", "dispersion", "zero", "one"),
  what = "full",
  plot = FALSE,
  id = FALSE,
  ...
)

## S3 method for class 'cdfqrH'
dfbeta(
  model,
  type = c("full", "location", "dispersion", "zero", "one"),
  what = "full",
  ...
)

## S3 method for class 'cdfqrH'
dfbetas(
  model,
  type = c("full", "location", "dispersion", "zero", "one"),
  what = "full",
  ...
)
```

**Arguments**

model            A cdfqr model object



method	Currently only 'dfbeta' method is available.
type	A string that indicates whether the results for all parameters are to be returned, or only the location/dispersion submodel's parameters returned.
what	for influence statistics based on coefficient values, indicate the predictor variables that needs to be tested.
plot	if plot is needed.
id	for plot only, if TRUE, the case ids will be displayed in the plot.
...	currently ignored.s

**Value**

A matrix, each row of which contains the estimated influence on parameters when that row's observation is removed from the sample.

**See Also**

[lm.influence](#), [influence.measures](#)

**Examples**

```
data(cdfqrExampleData)
fit <- cdfquantreg(crc99 ~ vert | conf1, 't2', 't2', data = JurorData)
#It takes some time especially the data is large.
influcne <- influence(fit)
plot(influcne[,2])

## Not run:
# Same as influence(fit)
dfbetval <- dfbetas(fit)

## End(Not run)
```

---

IPCC

*IPCC data-set*

---

**Description**

The IPCC data-set comprises the lower, best, and upper estimates for the phrases "likely" and "unlikely" in six IPCC report sentences.

**Usage**

IPCC

**Format**

A data frame with 4014 rows and 8 variables:

**subj** Subject ID number

**treat** Experimental conditions

**valence** Valence of the sentences

**prob** raw probability estimates

**probm** Linear transformed prob into (0, 1) interval

**mid** Distinguish lower, best and upper estimates

**high** Distinguish lower, best and upper estimates

**Question** IPCC question number

**Source**

<https://journals.sagepub.com/doi/abs/10.1111/j.1467-9280.2009.02284.x>

---

IPCCAUS

*IPCC data-set - Australian data*

---

**Description**

The IPCC-AUS data-set comprises the best estimates for the phrases in IPCC report sentences.

**Usage**

IPCCAUS

**Format**

A data frame with 4014 rows and 8 variables:

**ID** Subject ID

**gender** Gender of subjects, '0' is male, '1' is female

**age** age of subjects

**cfprob** personal probability.

**bestprob** nominated probability.

**Source**

<https://journals.sagepub.com/doi/abs/10.1111/j.1467-9280.2009.02284.x>

---

IPCC_Wide	<i>IPCC data-set - Wide format</i>
-----------	------------------------------------

---

**Description**

The IPCC-wide data-set comprises the best estimates for the phrases "likely" and "unlikely" in six IPCC report sentences.

**Usage**

IPCC\_Wide

**Format**

A data frame with 4014 rows and 8 variables:

- Q4** Each column indicates the estimates for one sentence.
- Q5** Each column indicates the estimates for one sentence.
- Q6** Each column indicates the estimates for one sentence.
- Q8** Each column indicates the estimates for one sentence.
- Q9** Each column indicates the estimates for one sentence.
- Q10** Each column indicates the estimates for one sentence.

**Source**

<https://journals.sagepub.com/doi/abs/10.1111/j.1467-9280.2009.02284.x>

---

JurorData	<i>Juror data</i>
-----------	-------------------

---

**Description**

Juror Judgment Study.

**Usage**

JurorData

**Format**

A data frame with 104 rows and 3 variables:

- cr99** The ratings of confidence levels with rescaling into the (0, 1) interval to avoid 1 and 0 values.
- vert** was the dummy variable for coding the conditions of verdict types, whereas
- confl** was the dummy variable for coding the conflict conditions

**Source**

<https://www.tandfonline.com/doi/abs/10.1375/pplt.2004.11.1.154>

---

plot.cdfqr

*Plot Fitted Values/Residuals of A Cdfqr Object or Distribution*

---

**Description**

Plot Fitted Values/Residuals of A cdfqr Object or Distribution

**Usage**

```
## S3 method for class 'cdfqr'
plot(
  x,
  mu = NULL,
  sigma = NULL,
  fd = NULL,
  sd = NULL,
  n = 10000,
  type = c("fitted"),
  ...
)
```

**Arguments**

x	If the plot is based on the fitted values, provide a fitted cdfqr object.
mu, sigma,	fd, sd alternatively, mu and sigma, and the distribution can be specified
fd	A string that specifies the parent distribution.
sd	A string that specifies the sub-family distribution.
n	The number of random variates to be generated for user specified plot.
type	Currently only fitted values are available for generating plots.
...	other plot parameters pass onto <a href="#">plot</a> .

**Examples**

```
data(cdfqrExampleData)
fit <- cdfquantreg(crc99 ~ vert | conf1, 't2', 't2', data = JurorData)
plot(fit)
```

**Description**

Density function, distribution function, quantile function, and random generation of variates for a specified cdf-quantile distribution with mean equal to mean and standard deviation equal to sd.

**Usage**

```
pq(q, mu, sigma, fd, sd)
```

```
dq(x, mu, sigma, fd, sd)
```

```
rq(n, mu, sigma, fd, sd)
```

```
qq(p, mu, sigma, fd, sd)
```

**Arguments**

q	vector of quantiles.
mu	vector of means.
sigma	vector of standard deviations.
fd	A string that specifies the parent distribution.
sd	A string that specifies the sub-family distribution.
x	vector of quantiles.
n	Number of random samples.
p	vector of probabilities.

**Value**

dq gives the density, rq generates random variates, qq gives the quantile function, and pq gives the cumulative density of specified distribution.

**Examples**

```
x <- rq(5, mu = 0.5, sigma = 1, 't2','t2'); x  
dq(x, mu = 0.5, sigma = 1, 't2','t2')  
qtil <- pq(x, mu = 0.5, sigma = 1, 't2','t2');qtil  
qq(qtil , mu = 0.5, sigma = 1, 't2','t2')
```

predict.cdfqr

*Methods for Cdfqr Objects***Description**

Methods for obtaining the fitted/predicted values for a fitted cdfqr object.

**Usage**

```
## S3 method for class 'cdfqr'
predict(
  object,
  newdata = NULL,
  type = c("full", "mu", "sigma"),
  quant = 0.5,
  ...
)

## S3 method for class 'cdfqr'
fitted(object, type = c("full", "mu", "sigma"), plot = FALSE, ...)

## S3 method for class 'cdfqrH'
predict(
  object,
  newdata = NULL,
  type = c("full", "mu", "sigma", "zero", "one"),
  quant = 0.5,
  ...
)

## S3 method for class 'cdfqrH'
fitted(object, type = c("full", "mu", "sigma", "zero", "one"), ...)
```

**Arguments**

object	A cdfqr model fit object
newdata	Optional. A data frame in which to look for variables with which to predict. If not provided, the fitted values are returned
type	A character that indicates whether the full model prediction/fitted values are needed, or values for the ‘mu’ and ‘sigma’ submodel only.
quant	A number or a numeric vector (must be in (0, 1)) to specify the quantile(s) of the predicted value (when ‘newdata’ is provided, and predicted values for responses are required). The default is to use median to predict response values.
...	currently ignored
plot	if a plot is needed.

## Examples

```
data(cdfqrExampleData)
fit <- cdfquantreg(crc99 ~ vert | conf1, 't2', 't2', data = JurorData)

plot(predict(fit))
plot(predict(fit))
```

---

qrBoot

*Bootstrapping for cdf quantile regression*

---

## Description

qrBoot provides a simple bootstrapping method for estimating the parameters of a cdf quantile regression model.

## Usage

```
qrBoot(object, rn, f = coef, R = 500, ci = 0.95)
```

## Arguments

object	The fitted cdfqr model object
rn	The sample size of bootstrap samples
f	A function whose one argument is the name of a cdfqr object that will be applied to the updated cdfqr object to compute the statistics of interest. The default is coef.
R	Number of bootstrap samples.
ci	The confidence interval level to obtain the bootstrap confidence intervals

## Value

A matrix that includes the original statistics, bootstrap means, and bootstrap confidence intervals

## Examples

```
data(cdfqrExampleData)
fit <- cdfquantreg(crc99 ~ vert | conf1, 't2', 't2', data = JurorData)
qrBoot(fit, rn = 50, R = 50)
```

---

qrGrad	<i>Give the Gradient Function for CDF-Quantile Distribution Modles</i>
--------	--

---

**Description**

Give the Gradient Function for CDF-Quantile Distribution Modles.

**Usage**

```
qrGrad(fd, sd)
```

**Arguments**

fd	A string that specifies the parent distribution.
sd	A string that specifies the sub-family distribution.

**Value**

grad The gradient function of parameter estimates, given a specified cdf-quantile distribution

**Examples**

```
qrGrad('t2', 't2')
```

---

qrLogLik	<i>Log Likelihood for Fitting Cdfquantile Distributions</i>
----------	---

---

**Description**

Function to give the (negative) log likelihood for fitting cdfquantile distributions.

**Usage**

```
qrLogLik(y, mu, sigma, fd, sd, total = TRUE)
```

**Arguments**

y	the vector to be evaluated.
mu	mean of the distribution.
sigma	sigma of the distribution.
fd	A string that specifies the parent distribution.
sd	A string that specifies the sub-family distribution.
total	whether the sum of loglikelihood is calculated



**Value**

The negative log likelihood for fitting the data with a cdfquantile distribution.

**Examples**

```
y <- rbeta(20, 0.5, 0.5)
qrLogLik(y, mu = 0.5, sigma = 1, 't2','t2')
```

---

qrLogLikFun

*Function to Give the Log Likelihood Function*


---

**Description**

Function to compute the (negative) log likelihood for fitting cdfquantile models.

**Usage**

```
qrLogLikFun(fd, sd)
```

**Arguments**

fd                    A string that specifies the parent distribution.  
sd                    A string that specifies the sub-family distribution.

**Value**

The log-likelihood calculation function given a specified cdfquantile distribution.

**Examples**

```
qrLogLikFun('t2','t2')
```

---

qrPwlm

*Probability Weighted L-moment Skewness and Kurtosis*


---

**Description**

Calculate the skew and kurtosis statistics based on probability weighted moments, via simulation method.

**Usage**

```
qrPwlm(x, n = NULL, mu = NULL, sigma = NULL, fd = NULL, sd = NULL)
```

**Arguments**

x	The vector of values for the calculation of Skewness and Kurtosis.
n	The number of samples drawn in the simulation. The higher this value, the greater accuracy.
mu	vector of means.
sigma	vector of standard deviations.
fd	A string that specifies the parent distribution.
sd	A string that specifies the sub-family distribution.

**Details**

This function computes the L-moment measures of skew and kurtosis, which may be computed via linear combinations of probability-weighted moments (Greenwood, Landwehr, Matalas and Wallis, 1979).

**Value**

The tau3(skew) and tau4(kurtosis) values of the L-moment.

**References**

Greenwood, J. A., Landwehr, J. M., Matalas, N. C., & Wallis, J. R. (1979). Probability weighted moments: definition and relation to parameters of several distributions expressible in inverse form. *Water Resources Research*, 15(5), 1049-1054.

**Examples**

```
qrPwlm(n = 1000, mu = 0.5, sigma = 1, fd = 't2', sd = 't2')
```

---

 qrStart

---

*Starting Value Generation for CDF quantile Regressions*


---

**Description**

qrStart is the function for generating starting values for a cdf-quantile GLM null model.

**Usage**

```
qrStart(ydata, fd = NULL, sd = NULL)
```

**Arguments**

ydata	The variable to be modeled
fd	A string that specifies the parent distribution.
sd	A string that specifies the sub-family distribution.

**Details**

The start values for the location parameter in a null model are the median of the empirical distribution, and a starting value for the dispersion parameter based on a specific quantile of the empirical distribution, specified according to the theoretical distribution on which the model is based. The start values for all new predictor coefficients in both the location and dispersion submodels are assigned the value 0.1.

**Value**

A vector that consists initial values for mu and sigma.

**Examples**

```
x <- rbeta(100, 1, 2)
qrStart(x, fd='t2', sd='t2')
#[1] -0.5938286  1.3996999
```

---

residuals.cdfqr

*Register method for cdfqr object functions*


---

**Description**

Register method for cdfqr object functions.

**Usage**

```
## S3 method for class 'cdfqr'
residuals(object, type = c("raw", "pearson", "deviance"), ...)
```

**Arguments**

object	The cdfqr model project
type	The type of residuals to be extracted: 'raw', 'pearson', 'std.pearson', or 'deviance',
...	currently ignored

**Value**

residuals of a specified type.

**Examples**

```
data(cdfqrExampleData)
fit <- cdfquantreg(crc99 ~ vert | conf1, 't2','t2', data = JurorData)

residuals(fit, "pearson")
```

---

scaleTR *Transform Values into (0, 1) Interval*

---

### Description

scaleTR is function that rescales values of a variable into the (0, 1) interval.

### Usage

```
scaleTR(y, high = NULL, low = NULL, data = NULL, N = NULL, scale = 0.5)
```

### Arguments

y	A numeric vector, or a variable in a dataframe.
high	The highest possible value of that variable. The value should be equal or greater than the maximum value of y. If not supplied, the maximum value of y will be used.
low	The lowest possible value of that variable. The value should be equal or smaller than the minimum value of y. If not supplied, the minimum value of y will be used.
data	A dataframe that contains the variable y.
N	A integer, normally is the sample size or the number of values. If not supplied, the length of y will be used.
scale	A compressing parameter that determines the extend to which the boundary values are going to be pushed away from the boundary. See details.

### Details

scaleTR used the method suggested by Smithson and Verkuilen (2006) and applies linear transformation to values into the open interval (0, 1). It first transform the values from their original scale by taking  $y' = (y - a)/(b - a)$ , where a is the lowest possible value of that variable and b is the highest possible value of that variable. Next, it compresses the range to avoid zeros and ones by taking  $y'' = (y'(N - 1) + c)/N$ , where N is the sample size and c is the compressing parameter. The smaller value c is, the boundary values would be more approaching zeros and ones, and have greater impact on the estimation of the dispersion parameters in the cdf quantile model.

### See Also

[cdfquantreg](#)

### Examples

```
y <- rnorm(20, 0, 1)
ynew <- scaleTR(y)
```

---

`summary.cdfqr`*S3 Methods for getting output from fitted cdfqr Objects.*

---

**Description**

Give the S3 Methods for CDF-Quantile Distribution Models

**Usage**

```
## S3 method for class 'cdfqr'  
summary(object, ...)  
  
## S3 method for class 'cdfqr'  
print(x, digits = max(3, getOption("digits") - 3), ...)  
  
## S3 method for class 'cdfqr'  
logLik(object, ...)  
  
## S3 method for class 'cdfqr'  
nobs(object, ...)  
  
## S3 method for class 'cdfqr'  
deviance(object, ...)  
  
## S3 method for class 'cdfqr'  
coef(object, type = c("full", "mean", "sigma"), ...)  
  
## S3 method for class 'cdfqr'  
vcov(object, type = c("full", "mean", "sigma"), ...)  
  
## S3 method for class 'cdfqr'  
update(object, formula., ..., evaluate = TRUE)  
  
## S3 method for class 'cdfqr'  
formula(x, ...)  
  
## S3 method for class 'cdfqr'  
confint(object, parm, level = 0.95, submodel = "full", ...)
```

**Arguments**

<code>object</code>	The fitted cdfqr model.
<code>...</code>	Pass onto other functions or currently ignored
<code>x</code>	The fitted cdfqr model.
<code>digits</code>	Number of digits to be retained in printed output.

type, submodel	The parts of coefficients or variance-covariance matrix to be extracted.Can be "full", "mean",or "sigma".
formula.	Changes to the formula. See <a href="#">update.Formula</a> for details.
evaluate	If true evaluate the new updated model else return the call for the new model.
parm	a specification of which parameters are to be given confidence intervals, either a vector of numbers or a vector of names. If missing, all parameters are considered.
level	the confidence level required.

### Examples

```
data(cdfqrExampleData)
fit <- cdfquantreg(crc99 ~ vert | conf1, 't2','t2', data = JurorData)

summary(fit)
print(fit)
logLik(fit)
coef(fit)
deviance(fit)
vcov(fit)
confint(fit)

#Update the model
fit2 <- update(fit, crc99 ~ vert*conf1 | conf1)
summary(fit2)
```

---

summary.cdfqrH

*S3 Methods for getting output from fitted cdfqrr Objects.*


---

### Description

Give S3 Methods for CDF-Quantile Distribution Models

### Usage

```
## S3 method for class 'cdfqrH'
summary(object, ...)

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

## S3 method for class 'cdfqrH'
logLik(object, ...)

## S3 method for class 'cdfqrH'
nobs(object, ...)
```

```

## S3 method for class 'cdfqrH'
deviance(object, ...)

## S3 method for class 'cdfqrH'
coef(object, type = c("full", "mean", "sigma", "zero", "one"), ...)

## S3 method for class 'cdfqrH'
vcov(object, type = c("full", "mean", "sigma", "zero", "one"), ...)

## S3 method for class 'cdfqrH'
update(object, formula., zero.fo., one.fo., ..., evaluate = TRUE)

## S3 method for class 'cdfqrH'
formula(x, ...)

## S3 method for class 'cdfqrH'
confint(
  object,
  parm,
  level = 0.95,
  type = c("full", "mean", "sigma", "zero", "one"),
  ...
)

```

### Arguments

object	The fitted cdfqrH model.
...	Pass onto other functions or currently ignored
x	The fitted cdfqrH model.
digits	Number of digits to be retained in printed output.
type	The parts of coefficients or variance-covariance matrix to be extracted. Can be "full", "mean", or "sigma".
formula.	Changes to the formula. See <a href="#">update.Formula</a> for details.
zero.fo.	Changes to the formula for the zero component, only input as ~ predictors. See <a href="#">update.Formula</a> for details.
one.fo.	Changes to the formula for the one component, only input as ~ predictors. See <a href="#">update.Formula</a> for details.
evaluate	If true evaluate the new updated model else return the call for the new model.
parm	a specification of which parameters are to be given confidence intervals, either a vector of numbers or a vector of names. If missing, all parameters are considered.
level	the confidence level required.

**Examples**

```
data(cdfqrHExampleData)
ipcc_mid <- subset(IPCC, mid == 1 & high == 0)
fit <- cdfquantregH(prob ~ valence | valence, zero.fo = ~valence,
  one.fo = ~valence,
  fd = 't2', sd = 't2', type = "Z0", data = ipcc_mid)

summary(fit)
print(fit)
logLik(fit)
coef(fit)
deviance(fit)
vcov(fit)
confint(fit)
```



# Index

## \* datasets

- Ambdata, 3
- AnxStrData, 5
- ExtEvent, 15
- IPCC, 17
- IPCC\_Wide, 19
- IPCCAUS, 18
- JurorData, 19

## \* package

- cdfquantreg-package, 2

Ambdata, 3

anova.cdfqr, 4

anova.cdfqrH (anova.cdfqr), 4

AnxStrData, 5

bugsLikelihood, 5

bugsModel, 3, 6

cdfqr.control, 7, 10, 12, 13

cdfqrFamily, 3, 6, 7, 10, 12, 13

cdfquantreg, 3, 9, 28

cdfquantreg-package, 2

cdfquantregC, 11

cdfquantregH, 13

coef, 10, 12, 14

coef.cdfqr (summary.cdfqr), 29

coef.cdfqrH (summary.cdfqrH), 30

confint.cdfqr (summary.cdfqr), 29

confint.cdfqrH (summary.cdfqrH), 30

deviance.cdfqr (summary.cdfqr), 29

deviance.cdfqrH (summary.cdfqrH), 30

dfbeta.cdfqr (influence.cdfqr), 15

dfbeta.cdfqrH (influence.cdfqr), 15

dfbetas.cdfqr (influence.cdfqr), 15

dfbetas.cdfqrH (influence.cdfqr), 15

dq, 3

dq (pq), 21

ExtEvent, 15

fitted.cdfqr (predict.cdfqr), 22

fitted.cdfqrH (predict.cdfqr), 22

formula.cdfqr (summary.cdfqr), 29

formula.cdfqrH (summary.cdfqrH), 30

influence.cdfqr, 3, 15

influence.cdfqrH (influence.cdfqr), 15

influence.measures, 17

IPCC, 17

IPCC\_Wide, 19

IPCCAUS, 18

JurorData, 19

lm.influence, 17

logLik.cdfqr (summary.cdfqr), 29

logLik.cdfqrH (summary.cdfqrH), 30

nobs.cdfqr (summary.cdfqr), 29

nobs.cdfqrH (summary.cdfqrH), 30

optim, 7

plot, 20

plot.cdfqr, 20

pq, 3, 21

predict.cdfqr, 22

predict.cdfqrH (predict.cdfqr), 22

print, 10, 12, 14

print.cdfqr, 10, 12, 14

print.cdfqr (summary.cdfqr), 29

print.cdfqrH (summary.cdfqrH), 30

qq, 3

qq (pq), 21

qrBoot, 3, 23

qrGrad, 24

qrLogLik, 24

qrLogLikFun, 25

qrPwlm, 3, 25

qrStart, 26

residuals.cdfqr, [3](#), [27](#)

rq, [3](#)

rq (pq), [21](#)

scaleTR, [28](#)

summary, [10](#), [12](#), [14](#)

summary.cdfqr, [10](#), [12](#), [14](#), [29](#)

summary.cdfqrH, [30](#)

update.cdfqr (summary.cdfqr), [29](#)

update.cdfqrH (summary.cdfqrH), [30](#)

update.Formula, [30](#), [31](#)

vcov.cdfqr (summary.cdfqr), [29](#)

vcov.cdfqrH (summary.cdfqrH), [30](#)