Package: ctqr (via r-universe)

September 10, 2024

Type Package

Title Censored and Truncated Quantile Regression

Version 2.1

Date 2024-02-12

Author Paolo Frumento

Maintainer Paolo Frumento <paolo.frumento@unipi.it>

Description Estimation of quantile regression models for survival data.

Depends survival, pch (>= 2.1)

Imports stats

Suggests car, lmtest

License GPL-2

RoxygenNote 7.2.3

NeedsCompilation no

Repository CRAN

Date/Publication 2024-02-12 19:30:10 UTC

Contents

ctqr-package *Censored and Truncated Quantile Regression*

Description

Fit quantile regression models to survival data, allowing for right censoring, left truncation, and interval censoring.

Details

Package:	ctqr
Type:	Package
Version:	2.1
Date:	2024-02-12
License:	GPL-2

The main function `ctqr` is used for model fitting. Other documented functions are `predict.ctqr`, to obtain prediction from a `ctqr` object, `plot.ctqr`, to plot quantile regression coefficients, and `ctqr.control`, that can be used to set the operational parameters for the estimation algorithm.

Author(s)

Paolo Frumento

Maintainer: Paolo Frumento <paolo.frumento@unipi.it>

References

Frumento, P., and Bottai, M. (2017). An estimating equation for censored and truncated quantile regression. *Computational Statistics and Data Analysis*, Vol.113, pp.53-63. ISSN: 0167-9473.

Frumento, P. (2022). A quantile regression estimator for interval-censored data. *The International Journal of Biostatistics*, 19 (1), pp. 81-96.

See Also

`pchreg`, that is used to compute a preliminary estimate of the conditional outcome distribution.

ctqr *Censored and Truncated Quantile Regression*

Description

Fits a quantile regression model to possibly censored and truncated data, e.g., survival data.

Usage

```
ctqr(formula, data, weights, p = 0.5, CDF, control = ctqr.control(), ...)
```

Arguments

`formula`	an object of class "`formula`": a symbolic description of the regression model. The response must be a `Surv` object as returned by `Surv` (see 'Details').
`data`	an optional data frame containing the variables in the model.
`weights`	an optional vector of weights to be used in the fitting process. The weights will always be normalized to sum to the sample size. This implies that, for example, using double weights will not halve the standard errors.
`p`	numerical vector indicating the order of the quantile(s) to be fitted.
`CDF`	an object of class "pch", i.e., the result of a call to `pchreg`. If missing, it will be computed internally with default settings. See 'Details'.
`control`	a list of operational parameters for the optimization algorithm, usually passed via `ctqr.control`.
`...`	for future arguments.

Details

This function implements the method described in Frumento and Bottai (2017) for censored, truncated quantile regression, and Frumento (2022) for interval-censored quantile regression.

The left side of `formula` must be of the form `Surv(time, event)` if the data are right-censored, `Surv(time0, time, event)` if the data are right-censored and left-truncated (`time0 < time`, `time0` can be `-Inf`), and `Surv(time1, time2, type = "interval2")` if the data are interval-censored (use `time1 = time2` for exact observations, `time1 = -Inf` or `NA` for left-censored, and `time2 = Inf` or `NA` for right-censored). Using `Surv(time)` is also allowed and indicates that the data are neither censored nor truncated.

The conditional distribution function (`CDF`) of the response variable represents a nuisance parameter and is estimated preliminarily via `pchreg`. If missing, `CDF = pchreg(formula)` is used as default. See the "Note" and the documentation of `pchreg`.

Estimation is carried out using an algorithm for gradient-based optimization. To estimate the asymptotic covariance matrix, standard two-step procedures are used (e.g., Ackerberg et al., 2012).

Value

An object of class “ctqr”, which is a list with the following items:

p	the quantile(s) being estimated.
coefficients	a named vector or matrix of quantile regression coefficients.
call	the matched call.
n.it	the number of iterations.
converged	logical. The convergence status.
fitted	the fitted values.
terms	the terms object used.
mf	the model frame used.
covar	the estimated asymptotic covariance matrix.
CDF	the used CDF.

Note that the dimension of all items, except call, terms, mf, and CDF, is the same as the dimension of p. For example, if p = c(0.25,0.5,0.75), coefficients and fitted will be 3-columns matrices; n.it and converged will be vectors of 3 elements; and covar will be a list of three covariance matrices.

The generic accessor functions summary, plot, predict, coef, terms, nobs, can be used to extract information from the model. The functions waldtest (from the package **lmtest**), and linearHypothesis (from the package **car**) can be used to perform Wald test, and to test linear restrictions. These functions, however, will only work if p is scalar.

Note

NOTE 1. The first-step estimator (the CDF argument) is computed using the pchreg function of the **pch** package. To be correctly embedded in ctqr, a pch object must be constructed using the same observations, in the same order.

If the first-step estimator is biased, and there is censoring or truncation, the estimates of the quantile regression coefficients and their standard errors will also be biased.

If the data are neither censored nor truncated, the CDF does not enter the estimating equation of the model. However, since the first-step estimator is used to compute the starting points, the final estimates may be sensitive to the supplied CDF.

NOTE 2. Right-censoring is a special case of interval censoring, in which exact events are identified by time2 = time1, while censored observations have time2 = Inf. Note, however, that ctqr(Surv(time1, time2, type = "interval2") ~ x) will not be identical to ctqr(Surv(time = time1, event = (time2 < Inf)) ~ x). The estimating equation used for interval-censored data is that described in Frumento (2022), while that used for right-censored data is that of Frumento and Bottai (2017). The two estimating equations are only asymptotically equivalent.

Author(s)

Paolo Frumento <paolo.frumento@unipi.it>

References

Ackerberg, D., Chen, X., and Hahn, J. (2012). A practical asymptotic variance estimator for two-step semiparametric estimators. The Review of Economics and Statistics, 94 (2), 481-498.

Frumento, P., and Bottai, M. (2017). An estimating equation for censored and truncated quantile regression. *Computational Statistics and Data Analysis*, Vol.113, pp.53-63. ISSN: 0167-9473.

Frumento, P. (2022). A quantile regression estimator for interval-censored data. *The International Journal of Biostatistics*, 19 (1), pp. 81-96.

See Also

`plot.ctqr`, `predict.ctqr`, `pchreg`

Examples

```
# Using simulated data


# Example 1 - censored data ####################################################

n <- 1000
x1 <- runif(n); x2 <- runif(n)   # covariates
t <- runif(n, 0, 1 + x1 + x2)    # time variable (e.g., time to death)
c <- runif(n,0,5)                # censoring variable (e.g., end of follow-up)
y <- pmin(t,c)                   # observed variable = min(t,c)
d <- (t <= c)                    # 1 = event (e.g., death), 0 = censored

CDF1 <- pchreg(Surv(y,d) ~ x1 + x2)
model1 <- ctqr(Surv(y,d) ~ x1 + x2, p = 0.5, CDF = CDF1)
model2 <- ctqr(Surv(y,d) ~ x1, p = 0.5, CDF = CDF1)

# model1 is identical to ctqr(Surv(y,d) ~ x1 + x2, p = 0.5)
# model2 is NOT identical to ctqr(Surv(y,d) ~ x1, p = 0.5),
  # which would have default CDF = pchreg(Surv(y,d) ~ x1)


# Example 2 - censored and truncated data ######################################

n <- 1000
x1 <- runif(n); x2 <- runif(n)   # covariates
t <- runif(n, 0, 1 + x1 + x2)    # time variable
c <- runif(n,0,5)                # censoring variable
y <- pmin(t,c)                   # observed variable = min(t,c)
d <- (t <= c)                    # 1 = event, 0 = censored

z <- rnorm(n) # truncation variable (e.g., time at enrollment)
w <- which(y > z) # data are only observed when y > z
z <- z[w]; y <- y[w]; d <- d[w]; x1 <- x1[w]; x2 <- x2[w]

# implement various CDFs and choose the model with smallest AIC

CDFs <- list(
```

```
  pchreg(Surv(z,y,d) ~ x1 + x2, breaks = 5),
  pchreg(Surv(z,y,d) ~ x1 + x2, breaks = 10),
  pchreg(Surv(z,y,d) ~ x1 + x2 + x1:x2, breaks = 5),
  pchreg(Surv(z,y,d) ~ x1 + x2 + x1^2 + x2^2, breaks = 10)
)

CDF <- CDFs[[which.min(sapply(CDFs, function(obj) AIC(obj)))]]
summary(ctqr(Surv(z,y,d) ~ x1 + x2, p = 0.5, CDF = CDF))


# Example 3 - interval-censored data ###########################################
# t is only known to be in the interval (t1,t2) ################################

n <- 1000
x1 <- runif(n); x2 <- runif(n)     # covariates
t <- runif(n, 0, 10*(1 + x1 + x2)) # time variable
t1 <- floor(t)                     # lower extreme of the interval
t2 <- ceiling(t)                   # upper extreme of the interval

model <- ctqr(Surv(t1,t2, type = "interval2") ~ x1 + x2, p = 0.5)
```

`ctqr.control` *Auxiliary Function for Root Search*

Description

This functions can be used within a call to `ctqr`, to control the operational parameters of the root search algorithm.

Usage

```
ctqr.control(tol = 1e-06, maxit = 1000, a = 0.5, b = 1.25)
```

Arguments

`tol`	positive convergence tolerance: the algorithm stops when the maximum absolute change between two consecutive estimates is smaller than `tol`.
`maxit`	maximum number of iterations.
`a, b`	numeric scalar with `0 < a < 1` and `b > 1`. See 'Details'.

Details

For a current estimate `beta`, a new estimate is computed as `beta_new = beta + delta*s(beta)`, where `s(beta)` is the current value of the estimating equation and `delta` is a positive multiplier. If `sum(s(beta_new)^2) < sum(s(beta)^2)`, the iteration is accepted and `delta` is multiplied by `b`. Otherwise, `beta_new` is rejected and `delta` is multiplied by `a`. By default, `a = 0.5` and `b = 1.25`. Choosing `a,b` closer to 1 may result in a more accurate estimate, but will require a larger number of iterations.

Value

The function returns its arguments. If some was not correctly specified, it is set to its default and a warning message is returned.

See Also

`ctqr`

`plot.ctqr` *Plot Quantile Regression Coefficients*

Description

Plots quantile regression coefficients $\beta(p)$ as a function of p, based on a fitted model of class "`ctqr`".

Usage

```
## S3 method for class 'ctqr'
plot(x, which = NULL, ask = TRUE, ...)
```

Arguments

`x`	an object of class "`ctqr`".
`which`	an optional numerical vector indicating which coefficient(s) to plot. If `which = NULL`, all coefficients are plotted.
`ask`	logical. If `which = NULL` and `ask = TRUE` (the default), you will be asked interactively which coefficients to plot.
`...`	additional graphical parameters, that can include `xlim, ylim, xlab, ylab, col, lwd`. See `par`.

Details

With this command, a plot of $\beta(p)$ versus p is created, provided that at least two quantiles have been estimated. Dashed lines represent 95% confidence intervals, while the horizontal dotted line indicates the zero.

Author(s)

Paolo Frumento <paolo.frumento@unipi.it>

See Also

`ctqr`

Examples

```
# using simulated data

n <- 1000
x <- runif(n)
t <- 1 + x + rexp(n)
c <- runif(n, 1,10)
y <- pmin(c,t)
d <- (t <= c)

par(mfrow = c(1,2))
plot(ctqr(Surv(y,d) ~ x, p = seq(0.05,0.95,0.05)), ask = FALSE)
```

---

predict.ctqr *Prediction After Quantile Regression*

Description

This function returns predictions for an object of class "ctqr".

Usage

```
## S3 method for class 'ctqr'
predict(object, newdata, se.fit = FALSE, ...)
```

Arguments

object	a ctqr object.
newdata	optional data frame in which to look for variables with which to predict. It must include all the covariates that enter the quantile regression model. If omitted, the fitted values are used.
se.fit	logical. If TRUE, standard errors of the predictions are also computed.
...	for future methods.

Details

This function produces predicted values obtained by evaluating the regression function at newdata (which defaults to model.frame(object)).

Value

If se = FALSE, a matrix of fitted values, with rows corresponding to different observations, and one column for each value of object$p. If se = TRUE, a list with two items:

fit	a matrix of fitted values, as described above.
se.fit	a matrix of estimated standard errors.

Author(s)

Paolo Frumento <paolo.frumento@unipi.it>

See Also

ctqr

Examples

```
# Using simulated data

n <- 1000
x1 <- runif(n)
x2 <- runif(n)
t <- 1 + x1 + x2 + runif(n, -1,1)
c <- rnorm(n,3,1)
y <- pmin(t,c)
d <- (t <= c)

model <- ctqr(Surv(y,d) ~ x1 + x2, p = c(0.25,0.5))
pred <- predict(model) # the same as fitted(model)
predict(model, newdata = data.frame(x1 = c(0.2,0.6), x2 = c(0.1,0.9)), se.fit = TRUE)
```

Index