Package ‘robcbi’

July 23, 2019

Type Package
Title Conditionally Unbiased Bounded Influence Estimates
Version 1.1-3
Date 2019-07-23
Depends R (>= 3.2.0)
Author Alfio Marazzi <Alfio.Marazzi@unisante.ch>
Maintainer A. Randriamiharisoa <exelami@gmail.com>
Description Conditionally unbiased bounded influence estimates as described in
Kuensch et al. (1989) <doi:10.1080/01621459.1989.10478791> in three special cases of
the generalized linear model: Bernoulli, Binomial, and Poisson distributed responses.
Imports robeth, stats, graphics, utils
License GPL (>= 2)
LazyLoad yes
LazyData yes
NeedsCompilation no
Repository CRAN
Date/Publication 2019-07-23 15:00:05 UTC

R topics documented:

robcbi-package .................................................. 1
Breslow ............................................................ 3
correl ............................................................. 4
cubinf ............................................................. 5
cubinf.control ................................................... 7
cubinf.missing .................................................... 9
cubinf.summaries ............................................... 10
Finney ............................................................ 12
fits.compare ...................................................... 13
glm.summaries ................................................... 14
plot.cubinf ...................................................... 15
robcbi-package

Robust Fit for Discrete Generalized Linear Model

Description

Conditionally unbiased bounded influence estimates as described in Kuensch et al. (1989) in three special cases of the Generalized Linear Model: Bernoulli, Binomial, and Poisson distributed responses.

Details

Package: cubinf
Version: 1.0
Date: 2013-07-04
License: GPL (>= 2)

Author(s)

A. Marazzi <Alfio.Marazzi@chuv.ch>
Maintainer: A. Randriamiharisoa <Alex.Randriamiharisoa@chuv.ch>

References


Examples

```r
library(robcbi)
# First example
data(Finney)
Vol <- Finney$Vol; Rate <- Finney$Rate; Resp <- Finney$Resp
## Not run:
plot(Vol,Rate,type="n")
points(Vol[Resp==0],Rate[Resp==0],pch=5, cex=1.2)
points(Vol[Resp==1],Rate[Resp==1],pch=16,cex=1.2)
```
## End(Not run)
lVol <- log(Vol); lRate <- log(Rate)
z.glm <- glm(Resp~lVol+lRate,family=binomial)
summary(z.glm)
z.cub <- glm(Resp~lVol+lRate,family=binomial,method="cubinf", ufact=3.2)
summary(z.cub)
weights(z.cub)
## Not run:
plot(z.cub, smooth=TRUE, ask=TRUE)

## Not run:
comp <- fits.compare(z.glm,z.cub)
comp
## Not run:
plot(comp)

## Not run:
# Second example
data(Breslow)
## Not run:
help(Breslow)

## Not run:
y = Breslow$sumY
x1 = Breslow$Age10
x2 = Breslow$Base4
x3 = rep(0,length(y))
x3[Breslow$Trt=="progabide"] = 1
x4 = x2*x3
CBA = glm(y~x1+x2+x3+x4,family=poisson,method=cubinf,ufact=3.2)
## Not run:
plot(CBA,num=5)

## Not run:
weights(CBA)
#
# compute the $R_n^2$ statistic (Section 2.5) to compare CBA
# with a reduced model with three variables:
#
CBA.red = update(CBA, .~.-x3-x4)
np = 5 # number of parameters of the full model
nq = 3 # number of parameters of the reduced model
CVR = covar(CBA)
CFF = coef(CBA)
K22 = CVR[(nq+1):np,(nq+1):np]
cff = as.matrix(CFF[(nq+1):np])
Rn2 = t(cff) %*% solve(K22) %*% cff
Rn2
Description

Patients suffering from simple or complex partial seizures were randomized to receive either the antiepileptic drug progabide or a placebo. At each of four successive postrandomization clinic visits, the number of seizures occurring over the previous two weeks was reported.

Usage

data(Breslow)

Format

A data frame with 59 observations and the following 4 variables

- **Trt**: The treatment: a factor with levels "placebo" and "progabide".
- **sumY**: An integer value, the sum of seizures during the 1st, 2nd, 3rd and 4th two week periods.
- **Age10**: Age divided by 10.
- **Base4**: The eight-week baseline seizure count divided by 4.

References


Examples

```r
library(robcbi)
data(Breslow)
y <- Breslow$sumY
x1 <- Breslow$Age10
x2 <- Breslow$Base4
x3 <- rep(0, length(y))
x3[Breslow$Trt == "progabide"] <- 1
```

correl

*Generic functions for objects of classes "glm" and "cubinf"*

Description

Correlation and covariance matrix of the parameter estimates, rank, scale estimate, and weights.
Usage

correl(object, tl = 1e-10)
covar(object)
Rank(object)
rscale(object)
weights(object)

Arguments

object An object inheriting from class "glm" or "cubinf".
tl Tolerance for a scale factor (denominator) close to zero.

Details

The generic functions coef, residuals, fitted, formula, deviance, rscale, covar, correl, weights, Rank can be used to extract elements from an object returned by glm.

See Also

The model fitting function glm

Examples

library(robcbi)
data(Finney)
Vol <- Finney$Vol; Rate <- Finney$Rate; Resp <- Finney$Resp
lVol <- log(Vol); lRate <- log(Rate)
z.cub <- glm(Resp~lVol+lRate,family=binomial,method="cubinf", ufact=3.2)
correl(z.cub)
covar(z.cub)
Rank(z.cub)
rscale(z.cub)
weights(z.cub)

---

cubinf Conditionally unbiased bounded influence estimates of discrete Generalized Linear Models

Description

Conditionally unbiased bounded influence estimates as described in Kuensch et al. (1989) in three special GLM cases: Bernoulli, Binomial, and Poisson distributed responses. The result is an object of class "cubinf".
Usage

cubinf(x, y, weights = NULL, start=NULL, etastart=NULL, mustart=NULL, offset = NULL, family = binomial(), control = cubinf.control(...), intercept = FALSE, ...)

Arguments

x  Vector or matrix of explanatory variable(s). Columns represent variables and rows are observations.
y  Vector of observed responses. In the case of Binomial responses, y is a two column matrix: the 1st column contains the number of successes, the 2nd column the number of failures. The Bernoulli case, is treated as a special Binomial case. However, the response y is a categorical variable (not a matrix with two columns) with two levels.
weights  Optional weights for weighted regression. Components must be non negative integers.
start  Starting values for the parameters in the linear predictor. Not used but required for compatibility with the glm function.
etastart  Starting values for the linear predictor. Not used but required for compatibility with the glm function.
mustart  Starting values for the vector of means. Not used but required for compatibility with the glm function.
offset  Optional offset added to the linear predictor.
family  A family object. Only two options are available for cubinf: 'family=binomial()' and 'family=poisson()'.
control  A list of control parameters for the numerical algorithms. See cubinf.control for the possible control parameters and their defaults.
intercept  Logical flag: if TRUE, an intercept term is added to the model.
...
Further named control arguments as singular.ok or qr.out used in the case where the x matrix is singular.

Details

The initial values of the coefficients (theta), the matrix A and the bias correction c are computed using the ROBETH subroutine GINTAC (Marazzi, 1993). Then an initial covariance matrix (for the convergence criterion) is computed by means of the ROBETH subroutines GFEDCA and KTASKW. Finally, the main algorithm (subroutine GYMAIN) alternates between improving values of - theta, for fixed A and c (theta-step, subroutine GYTSTP), - c, for fixed theta and A (c-step, subroutine GYCSTP), - A, for fixed theta and c (A-step, subroutine GYASTP).

For the different available options see the function cubinf.control.

Value

A list with the following components:

coefficients  Coefficient estimates.
residuals  Working residuals.
rsdev     Deviance residuals.
fitted.values  Fitted values.
cov        Estimated covariance matrix of the coefficients.
rank       Rank of the model matrix.
df.residuals Degrees of freedom in the residuals.
ci          Vector of final bias corrections.
A           Final value of the matrix A.
ai          Vector with components $a_i = ufact \times |Ax_i|$ (where $x_i^T$ denotes the ith row of
            the model matrix)
converged   A logical value. FALSE if the maximum number of iterations was reached.
control     Control parameters.
prior.weights Input vector w (when some of its components are different from 1).
family      The family object used in the call to cubinf ‘ics=1’ for the Bernoulli case. ‘ics=2’
            for the Binomial case. ‘ics=3’ for the Poisson case.
linear.predictors Components of the linear predictor (the model matrix multiplied by the coeffi-
            cient vector).
iter        Number of iterations required in the main algorithm.
y           Coded value of the response.
gradient    Vector of the final unscaled negative gradient of the objective function.
inv.hessian Vector of the final inverse of the Hessian matrix in compact storage mode.

References
estimation in general regression models, with application to generalized linear models. Journal of
the American Statistical Association, 84, 460-466.
New York.

See Also
glm(..., method="cubinf"), cubinf.control

Examples
library(robcbi)
y <- c(5,10,15,20,30,40,60,80,100)
x <- matrix(
c(0,1,0,0,0,1,0,0,0,0,0,0,1,0,0,0,1,1,0,0,0,0,0,0,0,0,0,0,0,0,0,1,1,1),
   nrow=9,byrow=FALSE)
z <- cubinf(x,y, family=poisson, control=list(ufact=3.2), intercept=TRUE)
z$iter
z$coeff
z <- cubinf(x, y, family=poisson, control=list(ufact=30), intercept=TRUE)
z$iter
z$coeff

---

**cubinf.control**  
*Control parameters for the function cubinf*

**Description**

Allows the user to set parameters affecting the estimation of the discrete GLMs implemented in cubinf. Most control parameters are parameters of the ROBETH subroutine GYMAIN (Marazzi, 1993).

**Usage**

```r
library(cubinf)

cubinf.control(tlo = 0.001, tua = 1e-06, mxx = 30, mxt = 10, mxf = 10, ntm = 0, gma = 1, 
iug = 1, ipo = 1, ilg = 2, icn = 1, icv = 1, ufact = 0, cpar = 1.5, 
null.dev=TRUE, ...)
```

**Arguments**

- **tlo**
  Relative precision for the convergence criterion of the main algorithm (GYMAIN) called by cubinf. The relative precision for the convergence criterion in the lower level steps (theta-step, A-step and c-step) is '10*tlo'.

- **tua**
  Tolerance used for the determination of the pseudo-rank.

- **mxx**
  Maximum number of cycles for the main algorithm.

- **mxt**
  Maximum number of iterations for the theta-step.

- **mxf**
  Maximum number of iterations for the A-step.

- **ntm**
  Parameter to control iteration monitoring. When the number of iterations in the theta-step reaches a multiple of 'ntm', the current parameter values as well as the corresponding value of the objective function are printed.

- **gma**
  Relaxation factor for the theta-step.

- **iug**
  Parameter for the choice of the u-function in the A-step. See Marazzi, 1993, for details.

- **ipo**
  Parameter for the choice of the steplength algorithm in the theta-step. If 'ipo=1', a quadratic comparison function is minimized. If 'ipo=2', the Goldstein-Armijo step length algorithm is used.

- **ilg**
  Parameter for the choice of the algorithm in the c-step. If 'ilg=1', the H-algorithm is used. If 'ilg=2', the W-algorithm is used.

- **icn**
  Parameter for the choice of the convergence criterion for the theta-step and the main algorithm. If 'icn=1', convergence is assumed when the change in each coefficient is less than the tolerance ('10*tlo') times an estimate of the coefficient variance. See Marazzi (1993, p. 281), for the other options ('icn=2' and 'icn=3').
Parameter for the choice of the convergence criterion for the A-step. If 'icv=1', convergence is assumed when the norm of the difference between two consecutive values of A is less than the tolerance (10*tol). See Marazzi (1993, p.288 and p. 301), for another option ('icv=2').

The tuning constant b is set equal to ufact*sqrt(p), where p is the dimension of the observation vectors. The default value of b is 1.1*sqrt(p); this value is used when 'ufact=0' on input.

Parameter used in determining an initial value of theta (standard Mallows estimate, see Marazzi, 1993, p281).

If 'null.dev=TRUE', the null deviance is computed. The null deviance is the deviance of the model with no predictors.

Further named control arguments as singular.ok or qr.out used in the case where the x matrix is singular

List of control parameters.


See Also

Examples

#To compute the classical estimates using cubinf, set:
control <- cubinf.control(ufact=300)

Missing methods for an object of class "cubinf"

These functions are not implemented for an object of class "cubinf".

anova(object, ...)
add1(object, ...)
drop1(object, ...)
step(object, ...)

Arguments

object An object inheriting from class "glm" or "cubinf".
...
Optional arguments according to the method.

See Also

The model fitting function glm, cubinf

cubinf.summaries  Functions required by the corresponding access functions

Description

Auxiliary functions for residuals(), summary(), covar(), deviance(), family(), Rank(), rscale(), weights().

Usage

## S3 method for class 'cubinf'
residuals(object, type = c("deviance", "pearson", "response"), ...)

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

## S3 method for class 'cubinf'
covar(object)

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

## S3 method for class 'cubinf'
family(object, ...)

## S3 method for class 'cubinf'
Rank(object)

## S3 method for class 'cubinf'
rscale(object)

## S3 method for class 'cubinf'
weights(object)
Arguments

object An object inheriting from class "cubinf".

type A character string for the residual type.

... Optional arguments. For summary, it can be correlation=TRUE.

Details

The generic functions coef, residuals, fitted, formula, deviance, rscale, r.squared, covar, correl, weights and Rank can be used to extract elements from an object of class "cubinf" returned by glm. The class "lm" functions effects, alias, add1, drop1, codekappa, proj, step, influence, anova and sensitivity are not implemented to objects of class "cubinf".

Value

summary.cubinf returns a list with the following components:

call The model formula used in glm.

terms Terms object used in fitting the model.

coefficients A matrix with three columns, containing the coefficients, their standard errors and the corresponding t-statistics.

dispersion Dispersion coefficient

df Degrees of freedom for model and residuals.

deviance.resid Deviance residuals

family The family function used: binomial or poisson

cov.unscaled Unscaled covariance matrix of coefficient estimates.

correlation Correlation matrix of coefficient estimates.

deviance Deviance.

null.deviance Null deviance.

iter Number of iterations of the main algorithm.

nas A logical vector whose i-th component is TRUE if the i-th coefficient is NA.

See Also

The model fitting function glm, cubinf
Examples

```r
library(robcbi)
data(Finney)
Vol <- Finney$Vol; Rate <- Finney$Rate; Resp <- Finney$Resp
lVol <- log(Vol); lRate <- log(Rate)
z(glm <- glm(Resp~lVol+lRate,family=binomial)
summary(z glm)
z.cub <- glm(Resp~lVol+lRate,family=binomial,method="cubinf", ufact=3.2)
summary(z.cub)
weights(z.cub)
covar(z.cub)
deviance(z.cub)
Rank(z.cub)
residuals(z.cub)
rscale(z.cub)
```

Finney data from 'Annals of Eugenics' 1947

Description

Finney data over 39 observations on occurrence or not of vaso-constriction. The data were obtained in a study of the effect of the rate and volume of air inspired on a transient vaso-constriction in the skin of the digits. The R function, plotFdat, for plotting the variables is included in the list.

Usage

```r
data(Finney)
```

Format

A list with the following components

- **Resp** Occurrence (Resp=1) or not (Resp=0) of vaso-constriction
- **Vol** Volume of air inspired
- **Rate** Observed Rate
- **plotFdat** R function for plotting the response and explanatory variables

Source


References

**fits.compare**

**Examples**

```r
library(robcbi)
data(Finney)
Vol <- Finney$Vol; Rate <- Finney$Rate; Resp <- Finney$Resp
lVol <- log(Vol); lRate <- log(Rate)
```

---

**Description**

The `fits.compare` function accepts a sequence of objects of class "glm", "cubinf", or "aov" (with optional names), and creates a class "fits.compare" object. The `print.fits.compare` function prints summaries of each of the input objects in a manner suitable for comparing the input models.

**Usage**

```r
## S3 method for class 'fits.compare'
print(x, digits = max(3, .Options$digits - 3), ...)
fits.compare(...)
```

**Arguments**

- `x`: An object inheriting from class "fits.compare", the result of a call to `fits.compare`
- `digits`: Minimal number of significant digits.
- `...`: In `fits.compare(...)`, ... is a sequence of objects of class "lm", "lm.robust", or "aov". Otherwise ... represents further arguments passed to or from `print` method.

**Details**

It is not recommended to compare objects with different structures.

**See Also**

The model fitting function `glm`, `cubinf`

**Examples**

```r
library(robcbi)
data(Finney)
Vol <- Finney$Vol; Rate <- Finney$Rate; Resp <- Finney$Resp
lVol <- log(Vol); lRate <- log(Rate)
z.glm <- glm(Resp~lVol+lRate,family=binomial)
z.cub <- glm(Resp~lVol+lRate,family=binomial,method="cubinf", ufact=3.2)
comp <- fits.compare(z.glm,z.cub)
comp
```
glm.summaries

Accessor functions for objects the class "glm"

Description
Covariance matrix of the coefficient estimates, rank, scale estimate and the weights for class "glm" objects. All these functions are methods.

Usage

## S3 method for class 'glm'
covar(object)

## S3 method for class 'glm'
Rank(object)

## S3 method for class 'glm'
rscale(object)

## S3 method for class 'glm'
weights(object)

Arguments

object An object inheriting from class "glm".

Details

The generic accessor functions coef, residuals, fitted, formula, deviance, rscale, covar, correl, weights and Rank can be used to extract elements from an object returned by glm.

See Also
The model fitting function glm

Examples

library(robcbi)
data(Finney)
Vol <- Finney$Vol; Rate <- Finney$Rate; Resp <- Finney$Resp
lVol <- log(Vol); lRate <- log(Rate)
z.glm <- glm(Resp~lVol+lRate,family=binomial)
summary(z.glm)
covar(z.glm)
Rank(z.glm)
rscale(z.glm)
weights(z.glm)
plot.cubinf

Diagnostic plots for an object of class "cubinf"

Description

Six plots are available: Residuals vs Fitted Values, Sqrt of abs(Residuals) vs Fitted Values, Response vs Fitted Values" and QQline of Residuals. r-f spread plot is not available and Cook's distances are not available for objects of class "cubinf".

Usage

## S3 method for class 'cubinf'
plot(x, residuals = NULL, smooths = FALSE, rugplot = FALSE, id.n = 0, ask = TRUE, num=0, ...)

Arguments

x An object of class "cubinf"
residuals The residuals to be used in the plots if not null.
smooths Logical indicating if a smoother should be added to most plots.
rugplot Logical indicating if a "rug" representation of the data should be added to the plot.
id.n Number of points to be labelled in each plot, starting with the most extreme.
ask If ask=TRUE, the function operates in interactive mode.
... Optional arguments for par.
um Integer between 0 and 6. If num>0, plot the num-th choice in the previous list in batch mode.

See Also

glm.fit, plot.default

Examples

library(robcbi)
data(Finney)
Vol <- Finney$Vol; Rate <- Finney$Rate; Resp <- Finney$Resp
1Vol <- log(Vol); 1Rate <- log(Rate)
z.cub <- glm(Resp~1Vol+1Rate,family=binomial,method="cubinf", ufact=3.2)
summary(z.cub)
## Not run:
plot(z.cub, smooth=TRUE, ask=TRUE)
## End(Not run)
Plots for comparing fits

Description
Plots the results of a call to fits.compare. Plotting the "fits.compare" object results in a sequence of graphical displays. These displays are designed to be of use in comparing two sets of parameter estimates in linear models.

Usage

```r
## S3 method for class 'fits.compare'
plot(x, xplots = FALSE, ..., ask = TRUE)
```

Arguments

- `x` An object inheriting from class "fits.compare", the result of a call to `fits.compare`.
- `xplots` If TRUE, the graphics are displayed.
- `...` Further arguments passed to or from `plot` method.
- `ask` Graphical parameter, if TRUE (and the R session is interactive) the user is asked for input, before a new figure is drawn.

Details
It is not recommended to compare objects with different structures.

See Also
The model fitting function `glm`, `cubinf`

Examples

```r
library(robcbi)
data(Finney)
Vol <- Finney$Vol; Rate <- Finney$Rate; Resp <- Finney$Resp
lVol <- log(Vol); lRate <- log(Rate)
z.glm <- glm(Resp~lVol+lRate,family=binomial)
z.cub <- glm(Resp~lVol+lRate,family=binomial,method="cubinf", ufact=3.2)
comp <- fits.compare(z.glm,z.cub)
comp
## Not run:
plot(comp)
## End(Not run)
```
predict.cubinf  Prediction methods for objects of class "cubinf"

Description

Predictions provided by a model fit when method is "cubinf".

Usage

## S3 method for class 'cubinf'
predict(object, newdata, type = c("link", "response", "terms"),
         se.fit = FALSE, terms = labels(object$terms), ...)

Arguments

  object        An object of class "cubinf" for which predictions are desired.
  newdata      Specify the explanatory variables to used.
  type          The prediction type.
  se.fit       Logical to specify if standard errors are returned or not.
  terms        The terms in newdata.
  ...          Additional arguments affecting the predictions produced.

Value

The value returned depends on type.

References

New York.

estimation in general regression models, with application to generalized linear models. Journal of
the American Statistical Association, 84, 460-466.

See Also

predict.glm

Examples

library(robcbi)
data(Finney)
Vol <- Finney$Vol; Rate <- Finney$Rate; Resp <- Finney$Resp
df <- data.frame(lVol = log(Vol), lRate = log(Rate), Resp = Resp)
z.cub <- glm(Resp~lVol+lRate, family=binomial, data=df, method="cubinf", ufact=3.2)
set.seed(123)
rVol <- runif(20,0.4,3.7); rRate <- runif(20,0.3,3.75)
newdat <- data.frame(lVol=log(rVol),lRate=log(rRate))
predict(z.cub, newdat, type="response")

QQline

Add a theoretical QQ-line in a plot

Description

Adds a QQ-line for the values in x in the current plot.

Usage

QQline(x, ...)

Arguments

x

The sample for QQ-line

...

Graphical parameters

Value

The intercept and the slope of the QQ-line are returned

References


Examples

library(robcbi)
data(Finney)
Vol <- Finney$Vol; Rate <- Finney$Rate; Resp <- Finney$Resp
lVol <- log(Vol); lRate <- log(Rate)
z.cub <- glm(Resp~lVol+lRate,family=binomial,method="cubinf", ufact=3.2)
x <- residuals(z.cub, type="deviance")
## Not run:
qqnorm(x, ylab = "Deviance Residuals")
QQline(x, lty = 2)

## End(Not run)
Print methods for objects of class "cubinf", "cubinf.i", "summary.cubinf" or "glm.i"

Description

Printing linear model fits provided by glm or with method="cubinf"

Usage

```r
## S3 method for class 'cubinf'
print(x, ai = FALSE, ci = FALSE, A.mat = FALSE, ...)

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

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

Arguments

- **x**: An object result of a call to `summary.cubinf` (first usage), to `glm` with method="cubinf" (second usage), to `rscale.cubinf` or to `summary.cubinf` or to `weights.cubinf` or to `covar.cubinf` (third usage) and respectively to `rscale.glm` or to `covar.glm` or to `weights.glm`.
- **ai**: Vector with components $a_i = \text{ufact}/|Ax_i|$ (where $x_i^T$ denotes the $i$th row of the model matrix).
- **ci**: Vector of the final bias corrections.
- **A.mat**: The final value of the matrix A.
- **...**: Further optional arguments according to the print method. Implicit argument in all these functions is digits = max(3, .Options$digits - 3).

References


See Also

The model fitting function `glm`, `cubinf`
Examples

library(robcbi)

## Dobson (1990) Page 93: Randomized Controlled Trial :
counts <- c(18,17,15,20,10,20,25,13,12)
outcome <- gl(3,1,9)
treatment <- gl(3,3)
print(d.AD <- data.frame(treatment, outcome, counts))
zD93 <- glm(counts ~ outcome + treatment, family=poisson,method="cubinf",ufact=3.2)
summary(zD93)
print(zD93)
wi <- weights(zD93)
print(wi)