Package: drgee (via r-universe)

September 27, 2024

Type Package

Title Doubly Robust Generalized Estimating Equations

Version 1.1.10

Date 2020-01-08

Author Johan Zetterqvist <drjohanzetterqvist at gmail.com> , Arvid Sjölander <arvid.sjolander at ki.se> with contributions from Alexander Ploner.

Maintainer Johan Zetterqvist <drjohanzetterqvist@gmail.com>

Description Fit restricted mean models for the conditional association between an exposure and an outcome, given covariates. Three methods are implemented: O-estimation, where a nuisance model for the association between the covariates and the outcome is used; E-estimation where a nuisance model for the association between the covariates and the exposure is used, and doubly robust (DR) estimation where both nuisance models are used. In DR-estimation, the estimates will be consistent when at least one of the nuisance models is correctly specified, not necessarily both. For more information, see Zetterqvist and Sjölander (2015) <doi:10.1515/em-2014-0021>.

Imports nleqslv, survival, Rcpp, data.table

LinkingTo Rcpp, RcppArmadillo

License GPL-2 | GPL-3

NeedsCompilation yes

Encoding UTF-8

Repository CRAN

Date/Publication 2020-01-09 07:20:02 UTC

Contents

`drgee-package` *Doubly Robust Generalized Estimating Equations*

Description

The main function is `drgee`, which estimates a parameter β in a model defined as $g\{E(Y|A,L)\} - g\{E(Y|A=0,L)\} = \beta^T\{A \cdot X(L)\}$. By supplying (nuisance) models for $g\{E(Y|A=0,L)$ and $E(A|L)$, a consistent estimate of β is obtained when at least one of these models is correctly specified.

In the function `drgee`, three estimation methods are implemented: O-estimation, where a model for $g\{E(Y|A=0,L)$ is used; E-estimation, where a model for $E(A|L)$ is used; and DR-estimation where models for both $g\{E(Y|A=0,L)$ and $E(A|L)$ are used.

The function `gee` is an implementation of standard GEE with independent working correlation matrix.

For conditional methods with clustered data, cluster-specific intercepts are assumed in all models.

The function `drgeeData` is used for extraction and manipulation of data, and is called by `drgee` and `gee`.

The function `RobVcov` is used to calculate standard errors of the estimates, given a vector of the residuals from estimating equations, the Jacobian, and a cluster-identifying variable.

The function `findRoots` solves a system of non linear equations.

Author(s)

Johan Zetterqvist, Arvid Sjölander with contributions from Alexander Ploner.

`centerX` *Cluster-center a design matrix around cluster means*

Description

Given a design matrix and a cluster identification variable. A new design matrix is returned, where each column is centered around its cluster mean. Using C++ code for speed.

Usage

```
centerX(x, id)
```

Arguments

`x`	A design matrix
`id`	A cluster identifying variable.

Details

`centerX` is written as a help function with the aim of being fast.

Value

A matrix with cluster centered columns.

Author(s)

Johan Zetterqvist, Arvid Sjölander

`drgee` *Doubly Robust Generalized Estimating Equations*

Description

`drgee` is used to estimate an exposure-outcome effect adjusted for additional covariates. The estimation is based on regression models for the outcome, exposure or a combination of both. For clustered data the models may have cluster-specific intercepts.

Usage

```
drgee(outcome, exposure,
      oformula, eformula, iaformula = formula(~1),
      olink = c("identity", "log", "logit"),
      elink = c("identity", "log", "logit"),
      data, subset = NULL, estimation.method = c("dr", "o", "e"),
      cond = FALSE, clusterid, clusterid.vcov, rootFinder = findRoots,
      intercept = TRUE, ...)
```

Arguments

`outcome`	The outcome as variable or as a character string naming a variable in the `data` argument. If missing, the outcome is assumed to be the response of `oformula`.
`exposure`	The exposure as variable or as a character string naming a variable in the `data` argument. If missing, the exposure is assumed to be the response of `eformula`.
`oformula`	An expression or formula for the outcome nuisance model.
`eformula`	An expression or formula for the exposure nuisance model.
`iaformula`	An expression or formula where the RHS should contain the variables that "interact" (i.e. are supposed to be multiplied with) with the exposure in the main model. "1" will always added. Default value is no interactions, i.e. `iaformula = formula(~1)`.

`olink`	A character string naming the link function in the outcome nuisance model. Has to be `"identity"`, `"log"` or `"logit"`.Default is `"identity"`.
`elink`	A character string naming the link function in the exposure nuisance model. Has to be `"identity"`, `"log"` or `"logit"`. Default is `"identity"`.
`data`	A data frame or environment containing the variables used. If missing, variables are expected to be found in the calling environment of the calling environment.
`subset`	An optional vector defining a subset of the data to be used.
`estimation.method`	A character string naming the desired estimation method. Choose `"o"` for O-estimation, `"e"` for E-estimation or `"dr"` for DR-estimation. Default is `"dr"`.
`cond`	A logical value indicating whether the nuisance models should have cluster-specific intercepts. Requires a `clusterid` argument.
`rootFinder`	A function to solve a system of non-linear equations. Default is `findRoots`.
`clusterid`	A cluster-defining variable or a character string naming a cluster-defining variable in the `data` argument. If it is not found in the `data` argument, it will be searched for in the calling frame. If missing, each observation will be considered to be a separate cluster. This argument is required when `cond = TRUE`.
`clusterid.vcov`	A cluster-defining variable or a character string naming a cluster-defining variable in the `data` argument to be used for adding contributions from the same cluster. These clusters can be different from the clusters defined by `clusterid`. However, each cluster defined by `clusterid` needs to be contained in exactly one cluster defined by `clusterid.vcov`. This variable is useful when the clusters are hierarchical.
`intercept`	A boolean to choose whether the nuisance parameters in doubly robust conditional logistic regression should be fitted with a model with an intercept. Only used for doubly robust condtional logistic regression.
`...`	Further arguments to be passed to the function `rootFinder`.

Details

`drgee` estimates the parameter β in a main model $g\{E(Y|A,L)\} - g\{E(Y|A=0,L)\} = \beta^T\{A \cdot X(L)\}$, where Y is the outcome of interest, A is the exposure of interest, and L is a vector of covariates that we wish to adjust for. $X(L)$ is a vector valued function of L. Note that $A \cdot X(L)$ should be interpreted as a columnwise multiplication and that $X(L)$ will always contain a column of 1's. Given a specification of an outcome nuisance model $g\{E(Y|A=0,L) = \gamma^T V(L)$ (where $V(L)$ is a function of L) O-estimation is performed. Alternatively, leaving $g\{E(Y|A=0,L)$ unspecified and using an exposure nuisance model $h\{E(A|L)\} = \alpha^T Z(L)$ (where h is a link function and $Z(L)$ is a function of L) E-estimation is performed. When g is logit, the exposure nuisance model is required be of the form $logit\{E(A|Y=0,L)\} = \alpha^T Z(L)$. In this case the exposure needs to binary.

Given both an outcome and an exposure nuisance model, DR-estimation can be performed. DR-estimation gives a consistent estimate of the parameter β when either the outcome nuisance model or the exposure nuisance model is correctly specified, not necessarily both.

Usage is best explained through an example. Suppose that we are interested in the parameter vector (β_0, β_1) in a main model $logit\{E(Y|A,L_1,L_2)\} - logit\{E(Y|A=0,L_1,L_2)\} = \beta_0 A + \beta_1 A \cdot L_1$

where L_1 and L_2 are the covariates that we wish to adjust for. To adjust for L_1 and L_2, we can use an outcome nuisance model $E(Y|A = 0, L_1, L_2; \gamma_0, \gamma_1) = \gamma_0 + \gamma_1 L_1$ or an exposure nuisance model $logit\{E(A|Y = 0, L_1, L_2)\} = \alpha_0 + \alpha_1 L_1 + \alpha_2 L_2$ to calculate estimates of β_0 and β_1 in the main model. We specify the outcome nuisance model as `oformula=Y~L_1` and `olink = "logit"`. The exposure nuisance model is specified as `eformula = A~L_1+L_2` and `elink = "logit"`. Since the outcome Y and the exposure A are identified as the LHS of `oformula` and `eformla` respectively and since the outcome link is specified in the `olink` argument, the only thing left to specify for the main model is the (multiplicative) interactions $A \cdot X(L) = A \cdot (1, L_1)^T$. This is done by specifying $X(L)$ as `iaformula = ~L_1`, since 1 is always included in $X(L)$. We can then perform O-estimation, E-estimation or DR-estimation by setting `estimation.method` to `"o"`, `"e"` or `"dr"` respectively. O-estimation uses only the outcome nuisance model, and E-estimation uses only the exposure nuisance model. DR-estimation uses both nuisance models, and gives a consistent estimate of (β_0, β_1) if either nuisance model is correct, not necessarily both.

When `estimation.method = "o"`, the RHS of `eformula` will be ignored. The `eformula` argument can also be replaced by an `exposure` argument specifying what the exposure of interest is.

When `estimation.method = "e"`, the RHS of `oformula` will be ignored. The `oformula` argument can also be replaced by an `outcome` argument specifying what the outcome of interest is.

When `cond = TRUE` the nuisance models will be assumed to have cluster-specific intercept. These intercepts will not estimated.

When E-estimation or DR-estimation is chosen with `olink = "logit"`, the exposure link will be changed to `"logit"`. Note that this choice of outcome link does not work for DR-estimation when `cond = TRUE`.

Robust variance for the estimated parameter is calculated using the function `robVcov`. A cluster robust variance is calculated when a character string naming a cluster variable is supplied in the `clusterid` argument.

For E-estimation when `cond = FALSE` and g is the identity or log link, see Robins et al. (1992).

For DR-estimation when `cond = TRUE` and g is the identity or log link, see Robins (1999). For DR-estimation when g is the logit link, see Tchetgen et al. (2010).

O-estimation can also be performed using the `gee` function.

Value

`drgee` returns an object of class `drgee` containing:

`coefficients`	Estimates of the parameters in the main model.
`vcov`	Robust variance for all main model parameters.
`coefficients.all`	Estimates of all estimated parameters.
`vcov.all`	Robust variance of the all parameter estimates.
`optim.object`	An estimation object returned from the function specified in the `rootFinder`, if this function is called for the estimation of the main model parameters.
`optim.object.o`	An estimation object returned from the function specified in the `rootFinder`, if this function is called for the estimation of the outcome nuisance parameters.
`optim.object.e`	An estimation object returned from the function specified in the `rootFinder`, if this function is called for the estimation of the outcome nuisance parameters.

`call`	The matched call.
`estimation.method`	The value of the input argument `estimation.method`.
`data`	The original data object, if given as an input argument
`oformula`	The original oformula object, if given as an input argument
`eformula`	The original eformula object, if given as an input argument
`iaformula`	The original iaformula object, if given as an input argument

The class methods `coef` and `vcov` can be used to extract the estimated parameters and their covariance matrix from a `drgee` object. `summary.drgee` produces a summary of the calculations.

Author(s)

Johan Zetterqvist, Arvid Sjölander

References

Orsini N., Belocco R., Sjölander A. (2013), Doubly Robust Estimation in Generalized Linear Models, *Stata Journal*, **13**, 1, pp. 185–205

Robins J.M., Mark S.D., Newey W.K. (1992), Estimating Exposure Effects by Modelling the Expectation of Exposure Conditional on Confounders, *Biometrics*, **48**, pp. 479–495

Robins JM (1999), Robust Estimation in Sequentially Ignorable Missing Data and Causal Inference Models, *Proceedings of the American Statistical Association Section on Bayesian Statistical Science*, pp. 6–10

Tchetgen E.J.T., Robins J.M., Rotnitzky A. (2010), On Doubly Robust Estimation in a Semiparametric Odds Ratio Model, *Biometrika*, **97**, 1, 171–180

Zetterqvist J., Vansteelandt S., Pawitan Y., Sjölander (2016), Doubly Robust Methods for Handling Confounding by Cluster, *Biostatistics*, **17**, 2, 264–276

See Also

`gee` for O-estimation, `findRoots` for nonlinear equation solving and `robVcov` for estimation of variance.

Examples

```
## DR-estimation when
## the main model is
## E(Y|A,L1,L2)-E(Y|A=0,L1,L2)=beta0*A+beta1*A*L1
## and the outcome nuisance model is
## E(Y|A=0,L1,L2)=gamma0+gamma1*L1+gamma2*L2
## and the exposure nuisance model is
## E(A|Y=0,L1,L2)=expit(alpha0+alpha1*L1+alpha2*l2)

library(drgee)

expit<-function(x) exp(x)/(1+exp(x))
```

```
n<-5000

## nuisance
l1<-rnorm(n, mean = 0, sd = 1)
l2<-rnorm(n, mean = 0, sd = 1)

beta0<-1.5
beta1<-1
gamma0<--1
gamma1<--2
gamma2<-2
alpha0<-1
alpha1<-5
alpha2<-3

## Exposure generated from the exposure nuisance model
a<-rbinom(n,1,expit(alpha0 + alpha1*l1 + alpha2*l2))
## Outcome generated from the main model and the
## outcome nuisance model
y<-rnorm(n,
mean = beta0 * a + beta1 * a * l1 + gamma0 + gamma1 * l1 + gamma2 * l2,
sd = 1)

simdata<-data.frame(y,a,l1,l2)

## outcome nuisance model misspecified and
## exposure nuisance model correctly specified

## DR-estimation
dr.est <- drgee(oformula = formula(y~l1),
eformula = formula(a~l1+l2),
iaformula = formula(~l1),
olink = "identity", elink = "logit",
data = simdata, estimation.method = "dr")
summary(dr.est)

## O-estimation
o.est <- drgee(exposure = "a", oformula = formula(y~l1),
iaformula = formula(~l1), olink = "identity",
data = simdata, estimation.method = "o")
summary(o.est)

## E-estimation
e.est <- drgee(outcome = "y", eformula = formula(a~l1+l2),
iaformula = formula(~l1), elink="logit",
data = simdata, estimation.method = "e")
summary(e.est)
```

---

drgeeData *Extracting Variables and Model Matrices for Generalized Estimating equations*

Description

Given a main model, an outcome nuisance model and an exposure nuisance model `drgeeData` extracts the model variables and matrices from a `data.frame` or an `environment` object. It also performs some data cleaning and error checking.

Usage

```
drgeeData(outcome, exposure,
          oformula, eformula, iaformula = formula(~1),
          olink = c("identity", "log", "logit"),
          elink = c("identity", "log", "logit"),
          data, subset = NULL,
          estimation.method = c("dr", "o", "e"),
          cond = FALSE, clusterid, clusterid.vcov)
```

Arguments

`outcome`	The outcome as a variable or as a character string naming a variable in the `data` argument. If it is not found in the `data` argument, it will be searched for in the calling frame. If missing, the outcome is assumed to be the response of `oformula`.
`exposure`	The exposure as a variable or as a character string naming a variable in the `data` argument. If it is not found in the `data` argument, it will be searched for in the calling frame. If missing, the outcome is assumed to be the response of `eformula`.
`oformula`	An expression or formula for the outcome nuisance model. The outcome is identified as the response in this formula.
`eformula`	An expression or formula for the exposure nuisance model. The exposure is identified as the response in this formula.
`iaformula`	An expression or formula where the RHS should contain the variables that "interact" (i.e. are supposed to be multiplied with) with the exposure in the main model to create the terms associated with the parameters of interest. "1" will always added. Default value is no interactions, i.e. `formula(~1)`.
`olink`	A character string naming the link function in the outcome nuisance model. Have to be `"identity"`, `"log"` or `"logit"`. Default is `"identity"`.
`elink`	A character string naming the link function in the exposure nuisance model. Have to be `"identity"`, `"log"` or `"logit"`. Default is `"identity"`. When `olink="logit"` this is replaced by `"logit"`.
`data`	A data frame or environment containing the variables in `iaformula`, `oformula` and `eformula`. If missing, variables are expected to be found in the calling frame.
`subset`	An optional vector defining a subset of the data to be used.
`estimation.method`	A character string naming the desired estimation method. Choose `"o"` for O-estimation, `"e"` for E-estimation or `"dr"` for DR-estimation. Default is `"dr"`.

`cond`	A logical value indicating whether the nuisance models should have cluster-specific intercepts. If `cond=TRUE` the design matrices for the nuisance models do not have an intercept. Requires a `clusterid` argument.
`clusterid`	A cluster-defining variable or a character string naming a cluster-defining variable in the `data` argument. If it is not found in the `data` argument, it will be searched for in the calling frame. If missing, each observation will be considered to be a separate cluster. This argument is required when `cond = TRUE`.
`clusterid.vcov`	A cluster-defining variable or a character string naming a cluster-defining variable in the `data` argument to be used for adding contributions from the same cluster. These clusters can be different from the clusters defined by `clusterid`. However, each cluster defined by `clusterid` needs to be contained in exactly one cluster defined by `clusterid.vcov`. This variable is useful when the clusters are hierarchical.

Details

`drgeeData` is called by `drgee` and `gee` to extract data from a `data.frame` or `environment` object. The data can then be used to for O-estimation, E-estimation or DR-estimation. `drgeeData` uses `model.frame` and `model.matrix` to remove incomplete observations and to convert factors to dummy variables. It also performs check the supplied data for errors or inconsistencies.

The class method `summary.drgeeData` produces strings for the formulas with terms referring to the columns in the produced design matrices.

Value

`drgee.data` returns an object of class `drgeeData` containing

`used.rows`	The rows numbers in the original data for the used rows (after subset selection and exlusions).
`orig.order`	The original order of the observations.
`y`	The outcome matrix.
`a`	The exposure matrix.
`x`	The matrix of of interactions defined in `iaformula`. This matrix will always contain a column with 1's.
`ax`	The matrix of elementwise product(s) of `a` and each column in `x`.
`v`	The matrix of terms in the outcome nuisance model.
`z`	The matrix of terms in the exposure nuisance model.
`yx`	The matrix of elementwise product(s) of `y` and each column in `x`.
`id`	A factor defining clusters. For independent observations, the number of levels equals the number of complete observations.
`clustname`	A string for the name of the cluster defining variable.
`y.names`	A string for the name of the outcome.
`a.names`	A string for the name of the exposure.
`x.names`	A string vector for the variable names in `x`.

`ax.names`	A string vector for the variable names in `ax`.
`v.names`	A string vector for the variable names in `v`.
`z.names`	A string vector for the variable names in `z`.
`yx.names`	A string vector for the variable names in `yx`.
`olink`	A character string naming the link function in the outcome nuisance model.
`elink`	A character string naming the link function in the outcome nuisance model.
`cond`	A logical value indicating whether cluster-specific intercepts should be assumed. If `TRUE`, the is no column for the intercept in `v` and `z`. Outcome concordant will also be removed.
`oterms`	The `terms` object corresponding to the outcome nuisance model.
`eterms`	The `terms` object corresponding to the exposure nuisance model.

Author(s)

Johan Zetterqvist, Arvid Sjölander

See Also

`drgee`, `gee`, `model.frame` and `model.matrix`.

`findRoots` *Non Linear Equation System Solving*

Description

A wrapper around `nleqslv` from the **nleqslv** package to solve a non linear system of equations.

Usage

```
findRoots(beta.init, eq.func, d.eq.func = NULL, arg.list, ...)
```

Arguments

`beta.init`	An initial guess for the zero.
`eq.func`	A function of two variables for which the zero are sought. Its first argument `beta` should be a vector over which the zeros are sought and the second argument `arg.list` a list of additional arguments.
`d.eq.func`	A function to return the Jacobian of `eq.func` taking the same arguments as `eq.func`. Supplying this function can speed up calculations. Default is `NULL`.
`arg.list`	The second argument to `eq.func` and `d.eq.func`
`...`	A list of additional arguments to be passed to `nleqslv`

Details

`findRoots` calculates zeros fo the function `eq.func` and is the default equation solving function in `drgee`. It is supplied as a separate function in order to allow users to use other equation solvers by writing their own wrapper with the same interface as `findRoots`.

Value

The value is a list containing the following arguments:

`roots`	The zero(s) of the function `eq.func`.
`optim.object`	The optimization object returned from `nleqslv`.

Author(s)

Johan Zetterqvist, Arvid Sjölander

See Also

`nleqslv` in package **nleqslv**.

`gee` *Generalized Estimating Equations*

Description

gee performs estimation of parameters in a restricted mean model using standard GEEs with independent working correlation matrix. For clustered data, cluster-robust standard errors are calculated. When `cond=TRUE`, cluster-specific intercepts are assumed.

Usage

```
gee(formula, link = c("identity", "log", "logit"), data, subset, cond = FALSE,
clusterid, clusterid.vcov, rootFinder = findRoots, ...)
```

Arguments

`formula`	An expression or formula representing the expected outcome conditional on covariates.
`link`	A character string naming the link function to use. Has to be `"identity"`, `"log"` or `"logit"`. Default is `"identity"`.
`data`	A data frame or environment containing the variables appearing in `formula`. If missing, the variables are expected to be found in the environment of the `formula` argument.
`subset`	An optional vector defining a subset of the data to be used.
`cond`	A logical value indicating whether cluster-specific intercepts should be assumed. Requires a `clusterid` argument.

`clusterid`	A cluster-defining variable or a character string naming a cluster-defining variable in the `data` argument. If it is not found in the `data` argument, it will be searched for in the calling frame. If missing, each observation will be considered to be a separate cluster. This argument is required when `cond = TRUE`.
`clusterid.vcov`	A cluster-defining variable or a character string naming a cluster-defining variable in the `data` argument to be used for adding contributions from the same cluster. These clusters can be different from the clusters defined by `clusterid`. However, each cluster defined by `clusterid` needs to be contained in exactly one cluster defined by `clusterid.vcov`. This variable is useful when the clusters are hierarchical.
`rootFinder`	A function to solve a system of non linear equations. Default is `findRoots`.
`...`	Further arguments to be passed to the function `rootFinder`.

Details

Estimates parameters in a regression model, defined by `formula`. When `cond=FALSE`, the estimated coefficients are identical to those obtained with `glm`, but since no distributional assumptions are made, a robust variance is calculated. When `cond=TRUE` and `link` is `"identity"` or `"log"`, the coefficients will be calculated using conditional estimating equations as described in Goetgeluk and Vansteelandt (2008). When `cond=TRUE` and `link="logit"`, the coefficients will be calculated by conditional logistic regression (with robust standard errors).

Value

gee return an object of class `gee` containing:

`coefficients`	Estimates of the parameters.
`vcov`	Robust variance of the estimates.
`call`	The matched call.
`y`	The outcome vector.
`x`	The design matrix. For conditional methods there is no column corresponding to the intercept.
`optim.object`	An estimation object returned from the function specified in the `rootFinder`, if this function is called.
`res`	The residuals from the estimating equations.
`d.res`	The derivative of the residuals from the estimating equations.
`data`	The original data object, if given as an input argument
`formula`	The original formula object, if given as an input argument

The class methods `coef` and `vcov` can be used to extract the estimated parameters and their covariance matrix from a `gee` object. To obtain the 'naive' variance, i.e. the variance obtained from maximum likelihood estimation assuming correct parameteric model and no clustering, use the class method `naiveVcov`. The class method `summary.drgee` produces a summary of the calculations.

Author(s)

Johan Zetterqvist, Arvid Sjölander

References

Goetgeluk S., & Vansteelandt S. (2008), Conditional generalized estimating equations for the analysis of clustered and longitudinal data. *Biometrics*, **64**(3), pp. 772–780.

See Also

`glm`

`getScoreResidualsFromClogit` *Get the (conditional) score residuals from conditional logistic regression*

Description

A function to obtain the conditional score residuals from conditional logistic regression as well as the sum of derivatives of these residuals with respect to the parameters evaluated at the estimated parameters.

Usage

```
getScoreResidualsFromClogit(fit, coefs, y, x, id)
```

Arguments

`fit`	A fitted object. Could be any object that has a class.function `coefficients`, but is intended to be used with `clogit` in the `survival` package.
`coefs`	An estimate of the parameters. If the argument `fit` is supplied, this will be replaced by `coefficients(fit)`.
`y`	The observed outcomes.
`x`	The design matrix.
`id`	The cluster identification variable.

Details

`getScoreResidualsFromClogit` calculates the residuals from the conditional score equations used in conditional logistic regression as we as the sums of their derivatives with respect to the paramaters.

This is useful if one wants to use a calculate the sandwich estimate of a variance where the uncertainty in the estimation of a conditional odds ratio needs to be taken into account.

Value

`getScoreResidualsFromClogit` return a list containing:

`U`	An n x p matrix, where n is the number of observations and p is the length of the estimated parameter.
`dU.sum`	An p x p matrix containing the sums of the derivatives. The rows correspond to the estimating equations and the columns correspond to the partial derivatives.
`dU.mean`	The mean cluster sum of the derivatives of the score residuals from the conditional likelihood.

Author(s)

Johan Zetterqvist, Arvid Sjölander

See Also

`robVcov`.

`robVcov` *Robust Variance Calculation*

Description

`robVcov` and `robustVcov` calculates the asymptotic variance for Z-estimators.

Usage

```
robustVcov(U, d.U.sum, id = NULL)
robVcov(U, d.U, id = NULL)
```

Arguments

`U`	A n x q matrix of the estimating equations evaluated at the estimated model parameters, where n is the number of observations and q is the number of estimating equations.
`d.U.sum`	The sum of the jacobian of `U` evaluated at the solution to the estimating equations, with rows corresponding to the estimating equations and columns corresponding to the model parameters. The number of model parameters is assumed to equal the number of estimating equations such that `d.U.sum` is a q x q square matrix.
`d.U`	The mean of `d.U.sum` taken over all observations.
`id`	A factor with levels corresponding the clusters in the data. Default is `NULL` in which case all observations are considered to be independent.

Details

For robust variance estimation, see van der Vaart (2000).

For clustered data, the rows in `U` are added clusterwise resulting in a cluster robust estimate of the variance.

Value

The estimated covariance matrix.

Author(s)

Johan Zetterqvist, Arvid Sjölander

References

van der Vaart, A.W. (2000), *Asymptotic Statistics*, Cambridge University Press, pp. 52–53.

Index