Package 'glmtoolbox'

October 4, 2021

Type Package

Title Set of Tools to Data Analysis using Generalized Linear Models

Version 0.1.2

Description Set of tools to the statistical analysis of data using: (1) normal linear models; (2) generalized linear models; (3) negative binomial regression models as alternative to the Poisson regression models under the presence of overdispersion; (4) beta-binomial regression models as alternative to the binomial regression models under the presence of overdispersion; (5) generalized estimating equations for cluster correlated data.

License GPL-2 | GPL-3

Imports methods, stats, graphics

Suggests aplore3, MASS, ISLR

Depends numDeriv, Rfast, splines

Encoding UTF-8

LazyData true

RoxygenNote 7.1.1

NeedsCompilation no

Author Luis Hernando Vanegas [aut, cre],
Luz Marina Rondón [aut],
Gilberto A. Paula [aut]

Maintainer Luis Hernando Vanegas <lhvanegasp@unal.edu.co>

Repository CRAN

Date/Publication 2021-10-04 04:20:02 UTC

R topics documented:

AGPC *AGPC for Generalized Estimating Equations*

Description

Computes the Akaike-type penalized Gaussian pseudo-likelihood criterion (AGPC) for one or more objects of the class glmgee.

Usage

```
AGPC(..., k = 2, verbose = TRUE)
```

Arguments

`...`	one or several objects of the class glmgee which are obtained from the fit of generalized estimating equations.
`k`	an (optional) non-negative value giving the magnitude of the penalty. By default, `k` is set to be 2.
`verbose`	an (optional) logical switch indicating if should the report of results be printed. By default, `verbose` is set to be TRUE.

Details

If `k` is set to be 0 then the AGPC reduces to the Gaussian pseudo-likelihood criterion (GPC), proposed by Carey and Wang (2011), which corresponds to the logarithm of the multivariate normal density function.

Value

A `data.frame` with the values of the gaussian pseudo-likelihood, the number of parameters in the linear predictor plus the number of parameters in the correlation matrix, and the value of AGPC for each glmgee object in the input.

References

Carey V.J. and Wang Y.-G. (2011) Working covariance model selection for generalized estimating equations. *Statistics in Medicine* 30, 3117–3124.

Zhu X. and Zhu Z. (2013) Comparison of criteria to select working correlation matrix in generalized estimating equations. *Chinese Journal of Applied Probability and Statistics* 29, 515-530.

Fu L., Hao Y. and Wang Y.-G. (2018) Working correlation structure selection in generalized estimating equations. *Computational Statistics* 33, 983-996.

See Also

QIC, CIC, RJC, GHYC, SGPC

Examples

```
## Example 1
mod <- size ~ poly(days,4) + treat
fit1 <- glmgee(mod, id=tree, family=Gamma("log"), data=spruces, corstr="Exchangeable")
fit2 <- update(fit1, corstr="AR-1")
fit3 <- update(fit1, corstr="Stationary-M-dependent(2)")
fit4 <- update(fit1, corstr="Independence")
AGPC(fit1, fit2, fit3, fit4)

## Example 2
mod <- dep ~ visit + group
fit1 <- glmgee(mod, id=subj, family=gaussian, corstr="Exchangeable", data=depression)
fit2 <- update(fit1, corstr="AR-1")
fit3 <- update(fit1, corstr="Non-Stationary-M-dependent(2)")
fit4 <- update(fit1, corstr="Independence")
AGPC(fit1, fit2, fit3, fit4)

## Example 3
mod <- depressd ~ visit + group
fit1 <- glmgee(mod, id=subj, family=binomial, corstr="Exchangeable", data=depression)
fit2 <- update(fit1, corstr="AR-1")
fit3 <- update(fit1, corstr="Stationary-M-dependent(2)")
fit4 <- update(fit1, corstr="Independence")
AGPC(fit1, fit2, fit3, fit4)
```

`anova.glmgee` *Comparison of nested Generalized Estimating Equations*

Description

Allows to compare nested generalized estimating equations using the Wald and generalized score tests.

Usage

```
## S3 method for class 'glmgee'
anova(
  object,
  ...,
  test = c("wald", "score"),
  verbose = TRUE,
  varest = c("robust", "df-adjusted", "model", "bias-corrected")
)
```

Arguments

`object`	an (object) of the class glmgee which is obtained from the fit of a generalized estimating equation.
`...`	another objects of the class glmgee which are obtained from the fit of generalized estimating equations.
`test`	an (optional) character string indicating the required test. The available options are: Wald ("wald") and generalized score ("score") tests. By default, `test` is set to be "wald".
`verbose`	an (optional) logical switch indicating if should the report of results be printed. By default, `verbose` is set to be TRUE.
`varest`	an (optional) character string indicating the type of estimator which should be used to the variance-covariance matrix of the interest parameters in the Wald test. The available options are: robust sandwich-type estimator ("robust"), degrees-of-freedom-adjusted estimator ("df-adjusted"), bias-corrected estimator ("bias-corrected"), and the model-based or naive estimator ("model"). By default, `varest` is set to be "robust". See vcov.glmgee.

Value

A matrix with three columns which contains the following:

- `Chi`: The value of the statistic of the test.
- `df`: The number of degrees of freedom.
- `Pr(>Chi)`: The *p*-value of the test computed using the Chi-square distribution.

References

Boos D. (1992) On Generalized Score Tests. *American Statistician* 46, 327–33.

Rotnitzky A. and Jewell N.P. (1990). Hypothesis Testing of Regression Parameters in Semiparametric Generalized Linear Models for Cluster Correlated Data. *Biometrika* 77, 485-497.

Examples

```
## Example 1
mod <- size ~ poly(days,4)
fit1 <- glmgee(mod, id=tree, family=Gamma("log"), data=spruces, corstr="AR-1")
```

```
fit2 <- update(fit1, . ~ . + treat)
fit3 <- update(fit2, . ~ . + poly(days,4):treat)
anova(fit1,fit2,fit3,test="wald")
anova(fit3,test="wald")
anova(fit1,fit2,fit3,test="score")
anova(fit3,test="score")

## Example 2
mod2 <- depressd ~ group
fit1 <- glmgee(mod2, id=subj, family=binomial("logit"), corstr="Exchangeable", data=depression)
fit2 <- update(fit1, . ~ . + visit)
fit3 <- update(fit2, . ~ . + group:visit)
anova(fit1,fit2,fit3,test="wald")
anova(fit3,test="wald")
anova(fit1,fit2,fit3,test="score")
anova(fit3,test="score")

## Example 3
mod3 <- dep ~ group
fit1 <- glmgee(mod3, id=subj, family=gaussian("identity"), corstr="AR-1", data=depression)
fit2 <- update(fit1, . ~ . + visit)
fit3 <- update(fit2, . ~ . + group:visit)
anova(fit1,fit2,fit3,test="wald")
anova(fit3,test="wald")
anova(fit1,fit2,fit3,test="score")
anova(fit3,test="score")
```

`anova.overglm` *Comparison of nested Negative Binomial and Beta-Binomial Regression Models*

Description

Allows to compare nested negative binomial and beta-binomial regression models using Wald, score, gradient and likelihood ratio tests.

Usage

```
## S3 method for class 'overglm'
anova(object, ..., test = c("wald", "lr", "score", "gradient"), verbose = TRUE)
```

Arguments

`object`	an object of the class overglm which is obtained from the fit of a negative binomial or beta-binomial model.
`...`	another objects of the class overglm which are obtained from the fit of negative binomial or beta-binomial models.
`test`	an (optional) character string indicating the required test. The available options are: Wald ("wald"), Rao's score ("score"), likelihood ratio ("lr") and Terrell's gradient ("gradient") tests. By default, `test` is set to be "wald".

verbose — an (optional) logical indicating if should the report of results be printed. By default, verbose is set to be TRUE.

Details

The Wald, Rao's score and Terrell's gradient tests are performed using the observed Fisher information matrix.

Value

A matrix with three columns which contains the following:

- Chi: The value of the statistic of the test.
- Df: The number of degrees of freedom.
- Pr(>Chi): The *p*-value of the test computed using the Chi-square distribution.

References

Buse A. (1982) The Likelihood Ratio, Wald, and Lagrange Multiplier Tests: An Expository Note. *The American Statistician* 36, 153 - 157.

Terrell G.R. (2002) The gradient statistic. *Computing Science and Statistics* 34, 206 – 215.

Examples

```
## Example 1
fit1 <- overglm(cbind(cells,200-cells) ~ tnf, family="bb(logit)", data=cellular)
fit2 <- update(fit1, . ~ . + ifn)
fit3 <- update(fit2, . ~ . + tnf:ifn)
anova(fit1, fit2, fit3, test="wald")
anova(fit1, fit2, fit3, test="score")
anova(fit1, fit2, fit3, test="lr")
anova(fit1, fit2, fit3, test="gradient")

## Example 2
fit1 <- overglm(infections ~ frequency, family="nb1(log)", data=swimmers)
fit2 <- update(fit1, . ~ . + location)
fit3 <- update(fit2, . ~ . + age)
fit4 <- update(fit3, . ~ . + gender)
anova(fit1, fit2, fit3, fit4, test="wald")
anova(fit1, fit2, fit3, fit4, test="score")
anova(fit1, fit2, fit3, fit4, test="lr")
anova(fit1, fit2, fit3, fit4, test="gradient")
```

anova2 *Comparison of nested Generalized Linear Models*

Description

Allows to compare nested generalized linear models using Wald, score, gradient, and likelihood ratio tests.

Usage

```
anova2(
  object,
  ...,
  test = c("wald", "lr", "score", "gradient"),
  verbose = TRUE
)
```

Arguments

`object`	an object of the class glm which is obtained from the fit of a generalized linear model.
`...`	another objects of the class glm which are obtained from the fit of generalized linear models.
`test`	an (optional) character string indicating the required type of test. The available options are: Wald ("wald"), Rao's score ("score"), Terrell's gradient ("gradient"), and likelihood ratio ("lr") tests. By default, `test` is set to be "wald".
`verbose`	an (optional) logical indicating if should the report of results be printed. By default, `verbose` is set to be TRUE.

Details

The Wald, Rao's score and Terrell's gradient tests are performed using the expected Fisher information matrix.

Value

A matrix with three columns which contains the following:

- `Chi:` The value of the statistic of the test.
- `Df:` The number of degrees of freedom.
- `Pr(>Chi):` The *p*-value of the test computed using the Chi-square distribution.

References

Buse A. (1982) The Likelihood Ratio, Wald, and Lagrange Multiplier Tests: An Expository Note. *The American Statistician* 36, 153-157.

Terrell G.R. (2002) The gradient statistic. *Computing Science and Statistics* 34, 206 – 215.

Examples

```
## Example 1
Auto <- ISLR::Auto
fit1 <- glm(mpg ~ weight, family=inverse.gaussian("log"), data=Auto)
fit2 <- update(fit1, . ~ . + horsepower)
fit3 <- update(fit2, . ~ . + horsepower:weight)
anova2(fit1, fit2, fit3, test="lr")
anova2(fit1, fit2, fit3, test="score")
anova2(fit1, fit2, fit3, test="wald")
anova2(fit1, fit2, fit3, test="gradient")

## Example 2
burn1000 <- aplore3::burn1000
mod <- death ~ age + tbsa + inh_inj
fit1 <- glm(mod, family=binomial("logit"), data=burn1000)
fit2 <- update(fit1, . ~ . + inh_inj + age*inh_inj + tbsa*inh_inj)
anova2(fit1, fit2, test="lr")
anova2(fit1, fit2, test="score")
anova2(fit1, fit2, test="wald")
anova2(fit1, fit2, test="gradient")

## Example 3
fit <- glm(lesions ~ 1 + time, family=poisson("log"), data=aucuba)
anova2(fit, test="lr")
anova2(fit, test="score")
anova2(fit, test="wald")
anova2(fit, test="gradient")
```

---

aucuba *Lesions of Aucuba mosaic virus*

Description

The investigators counted the number of lesions of *Aucuba mosaic* virus developing after exposure to X rays for various times. See Snedecor and Cochran (1980, page 404).

Usage

```
data(aucuba)
```

Format

A data frame with 7 rows and 2 variables:

time a numeric vector giving the minutes of exposure.

lesions a numeric vector giving the counts of lesions, in hundreds.

References

Snedecor G.W. and Cochran W.G. (1989) *Statistical Methods, Eight Edition*, Iowa State University Press, Ames.

Examples

```
barplot(lesions ~ time, col="red", data=aucuba)
```

`bladder` *Bladder cancer in mice*

Description

Female mice were continuously fed dietary concentrations of 2-Acetylaminofluorene (2-AAF), a carcinogenic and mutagenic derivative of fluorene. Serially sacrificed, dead or moribund mice were examined for tumors and dates of deaths were recorded. These data consist of the incidences of bladder neoplasms in the mice observed during 33 months.

Usage

```
data(bladder)
```

Format

A data frame with 8 rows and 3 variables:

dose a numeric vector giving the dose, in parts per 10^4, of 2-AAF.

exposed a numeric vector giving the number of mice exposed to each dose of 2-AAF.

cancer a numeric vector giving the number of mice with bladder cancer for each dose of 2-AAF.

References

Zhang H. and Zelterman D. (1999) Binary Regression for Risks in Excess of Subject-Specific Thresholds. *Biometrics* 55, 1247-1251.

See Also

liver

Examples

```
barplot(100*cancer/exposed ~ dose, beside=TRUE, data=bladder, col="red",
        xlab="Dose of 2-AAF", ylab="% of mice with bladder cancer")
```

cellular *Agents to stimulate cellular differentiation*

Description

In a biomedical study of the immuno-activating ability of two agents, TNF (tumor necrosis factor) and IFN (interferon), to induce cell differentiation, the number of cells that exhibited markers of differentiation after exposure to TNF and IFN was recorded. At each of the 16 dose combinations of TNF/INF, 200 cells were examined. The main question is whether the two agents stimulate cell differentiation synergistically or independently.

Usage

```
data(cellular)
```

Format

A data frame with 16 rows and 3 variables:

cells a numeric vector giving the number of cells that exhibited markers of differentiation after exposure to the dose of the two agents

tnf a numeric vector giving the dose (U/ml) of TNF

ifn a numeric vector giving the dose (U/ml) of IFN

References

Piegorsch W.W., Weinberg C.R. and Margolin B.H. (1988) Exploring simple independent action in multifactor tables of proportions. *Biometrics* 44, 595-603.

Vanegas L.H. and Rondon L.M. (2020) A data transformation to deal with constant under/over-dispersion in binomial and poisson regression models. *Journal of Statistical Computation and Simulation* 90, 1811-1833.

Examples

```
barplot(100*cells/200 ~ ifn + tnf, beside=TRUE, data=cellular, col=terrain.colors(4),
        xlab="Dose of TNF", ylab="% of cells with markers of differentiation")
legend(1, 90, c("0","4","20","100"), fill=terrain.colors(4), bty="n", cex=0.9,
       title="Dose of IFN")
```

CIC *Correlation Information Criterion for Generalized Estimating Equations*

Description

Computes the Correlation Information Criterion (CIC) for one or more objects of the class glmgee.

Usage

```
CIC(..., verbose = TRUE)
```

Arguments

...	one or several objects of the class glmgee which are obtained from the fit of generalized estimating equations.
verbose	an (optional) logical switch indicating if should the report of results be printed. By default, verbose is set to be TRUE.

Value

A data.frame with the values of the CIC for each glmgee object in the input.

References

Hin L.-Y. and Wang Y.-G. (2009) Working-Correlation-Structure Identification in Generalized Estimating Equations. *Statistics in Medicine*, 28, 642-658.

Hin L.-Y., Carey V.J., Wang Y.-G. (2007) Criteria for Working–Correlation–Structure Selection in GEE: Assessment via Simulation. *The American Statistician* 61, 360–364.

See Also

QIC, GHYC, RJC, AGPC, SGPC

Examples

```
## Example 1
mod <- size ~ poly(days,4) + treat
fit1 <- glmgee(mod, id=tree, family=Gamma("log"), data=spruces, corstr="Exchangeable")
fit2 <- update(fit1, corstr="AR-1")
fit3 <- update(fit1, corstr="Stationary-M-dependent(2)")
fit4 <- update(fit1, corstr="Independence")
CIC(fit1, fit2, fit3, fit4)

## Example 2
mod <- dep ~ visit + group
fit1 <- glmgee(mod, id=subj, family=gaussian, corstr="Exchangeable", data=depression)
fit2 <- update(fit1, corstr="AR-1")
```

```
fit3 <- update(fit1, corstr="Non-Stationary-M-dependent(2)")
fit4 <- update(fit1, corstr="Independence")
CIC(fit1, fit2, fit3, fit4)

## Example 3
mod <- depressd ~ visit + group
fit1 <- glmgee(mod, id=subj, family=binomial, corstr="Exchangeable", data=depression)
fit2 <- update(fit1, corstr="AR-1")
fit3 <- update(fit1, corstr="Stationary-M-dependent(2)")
fit4 <- update(fit1, corstr="Independence")
CIC(fit1, fit2, fit3, fit4)
```

`confint2` *Confidence Intervals for Generalized Linear Models*

Description

Computes confidence intervals based on Wald, likelihood-ratio, Rao's score or Terrell's gradient tests for a generalized linear model.

Usage

```
confint2(
  model,
  level = 0.95,
  test = c("wald", "lr", "score", "gradient"),
  digits = 4,
  verbose = TRUE
)
```

Arguments

`model`	an object of the class glm which is obtained from the fit of a generalized linear model.
`level`	an (optional) value indicating the required confidence level. By default, `level` is set to be 0.95.
`test`	an (optional) character string indicating the required type of test. The available options are: Wald ("wald"), Rao's score ("score"), Terrell's gradient ("gradient"), and likelihood ratio ("lr") tests. By default, `test` is set to be "wald".
`digits`	an (optional) integer value indicating the number of decimal places to be used. By default, `digits` is set to be 4.
`verbose`	an (optional) logical indicating if should the report of results be printed. By default, `verbose` is set to be TRUE.

Details

The approximate 100(level)% confidence interval for β based on the test test is the set of values of β_0 for which the hypothesis H_0: $\beta = \beta_0$ versus H_1: $\beta! = \beta_0$ is not rejected at the approximate significance level of 100(1-level)%. The Wald, Rao's score and Terrell's gradient tests are performed using the expected Fisher information matrix.

Value

A matrix with so many rows as parameters in the linear predictor and two columns: "Lower limit" and "Upper limit".

References

Buse A. (1982) The Likelihood Ratio, Wald, and Lagrange Multiplier Tests: An Expository Note. *The American Statistician* 36, 153-157.

Terrell G.R. (2002) The gradient statistic. *Computing Science and Statistics* 34, 206 – 215.

Examples

```
## Example 1
Auto <- ISLR::Auto
fit1 <- glm(mpg ~ weight*horsepower, family=inverse.gaussian("log"), data=Auto)
confint2(fit1)

## Example 2
burn1000 <- aplore3::burn1000
mod <- death ~ age + tbsa + inh_inj + age*inh_inj + tbsa*inh_inj
fit2 <- glm(mod, family=binomial("logit"), data=burn1000)
confint2(fit2)
```

---

cooks.distance.glmgee *Cook's Distance for Generalized Estimating Equations*

Description

Produces an approximation, better known as the *one-step aproximation*, of the Cook's distance, which is aimed to measure the effect on the estimates of the parameters in the linear predictor of deleting each cluster in turn. This function also can produce a cluster-index plot of the Cook's distance for all parameters in the linear predictor or for some subset of them.

Usage

```
## S3 method for class 'glmgee'
cooks.distance(
  model,
  method = c("Preisser-Qaqish", "full"),
  level = c("clusters", "observations"),
```

```
  plot.it = FALSE,
  coefs,
  identify,
  varest = c("robust", "df-adjusted", "model", "bias-corrected"),
  ...
)
```

Arguments

`model`	an object of class glmgee obtained from the fit of a generalized estimating equation.
`method`	an (optional) character string indicating the method of calculation for the *one-step approximation*. The options are: the *one-step approximation* described by Preisser and Qaqish (1996) in which the working-correlation matrix is assumed to be known ("Preisser-Qaqish"); and the "authentic" *one-step approximation* ("full"). By default, `method` is set to be "Preisser-Qaqish".
`level`	an (optional) character string indicating the level for which the Cook's distance is required. The options are: cluster-level ("clusters") and observation-level ("observations"). By default, `level` is set to be "clusters".
`plot.it`	an (optional) logical indicating if the plot of Cook's distance is required or just the data matrix in which that plot is based. By default, `plot.it` is set to be FALSE.
`coefs`	an (optional) character string which (partially) match with the names of some of the parameters in the linear predictor.
`identify`	an (optional) integer indicating the number of clusters to identify on the plot of Cook's distance. This is only appropriate if `plot.it=TRUE`.
`varest`	an (optional) character string indicating the type of estimator which should be used to the variance-covariance matrix of the interest parameters. The available options are: robust sandwich-type estimator ("robust"), degrees-of-freedom-adjusted estimator ("df-adjusted"), bias-corrected estimator ("bias-corrected"), and the model-based or naive estimator ("model"). By default, `varest` is set to be "robust".
`...`	further arguments passed to or from other methods. If `plot.it=TRUE` then `...` may be used to include graphical parameters to customize the plot. For example, `col`, `pch`, `cex`, `main`, `sub`, `xlab`, `ylab`.

Details

The Cook's distance consists of the *distance* between two estimates of the parameters in the linear predictor using a metric based on the (estimate of the) variance-covariance matrix. The first one set of estimates is computed from a dataset including all clusters, and the second one is computed from a dataset in which the *i*-th cluster is excluded. To avoid computational burden, the second set of estimates is replaced by its *one-step approximation*. See the dfbeta.glmgee documentation.

Value

A matrix as many rows as clusters in the sample and one column with the values of the Cook's distance.

Examples

```
## Cook's distance for all parameters in the linear predictor
mod <- size ~ poly(days,4) + treat
fit <- glmgee(mod, id=tree, family=Gamma("log"), data=spruces, corstr="Exchangeable")
cooks.distance(fit, col="red", lty=1, lwd=1, col.lab="blue", main="Cook's distance",
               col.axis="blue", col.main="black", family="mono", cex=0.8)

## Cook's distance for the parameter associated to the variable treat
cooks.distance(fit, coef="treat", col="red", lty=1, lwd=1, col.lab="blue",
               main="Cook's distance", col.axis="blue", col.main="black",
               family="mono", cex=0.8)
```

---

cooks.distance.overglm

Cook's Distance for Negative Binomial and Beta-Binomial Models

---

Description

Produces an approximation, better known as the *one-step approximation*, of the Cook's distance, which is aimed to measure the effect on the estimates of the parameters in the linear predictor of deleting each observation in turn. This function also can produce an index plot of the Cook's distance for all parameters in the linear predictor or for some subset of them.

Usage

```
## S3 method for class 'overglm'
cooks.distance(model, plot.it = TRUE, coefs, identify, ...)
```

Arguments

`model`	an object of class overglm obtained from the fit of a negative binomial or beta-binomial model.
`plot.it`	an (optional) logical indicating if the plot is required or just the data matrix in which that plot is based. By default, `plot.it` is set to be TRUE.
`coefs`	an (optional) character string which (partially) match with the names of some model parameters.
`identify`	an (optional) integer indicating the number of individuals to identify on the plot of the Cook's distance. This is only appropriate if `plot.it=TRUE`.
`...`	further arguments passed to or from other methods. If `plot.it=TRUE` then `...` may be used to include graphical parameters to customize the plot. For example, `col`, `pch`, `cex`, `main`, `sub`, `xlab`, `ylab`.

Details

The Cook's distance consists of the *distance* between two estimates of the parameters in the linear predictor using a metric based on the (estimate of the) variance-covariance matrix. The first one set of estimates is computed from a dataset including all individuals, and the second one is computed from a dataset in which the *i*-th individual is excluded. To avoid computational burden, the second set of estimates is replaced by its *one-step approximation*. See the dfbeta.overglm documentation.

Value

A matrix as many rows as individuals in the sample and one column with the values of the Cook's distance.

Examples

```
## Cook's distance for all parameters in the linear predictor
fit <- overglm(cbind(cells,200-cells) ~ tnf + ifn + tnf*ifn, family="bb(logit)", data=cellular)
cooks.distance(fit, col="red", lty=1, lwd=1, col.lab="blue", main="Cook's distance",
               col.axis="blue", col.main="black", family="mono", cex=0.8)

## Cook's distance for the parameter associated to the interaction
cooks.distance(fit, coef="tnf:ifn", col="red", lty=1, lwd=1, col.lab="blue",
               main="Cook's distance", col.axis="blue", col.main="black",
               family="mono", cex=0.8)
```

---

depression *Treatment for severe postnatal depression*

Description

These data arose from a study on the efficacy of oestrogen give transdermally for treatment of severe postnatal depression. Women with major depression were randomly assigned to either a placebo control group or estrogen patch group. Prior to the treatment all women were assessed by self-ratings of depressive symptoms on the Edinburgh Postnatal Depression Scale (EPDS). The data on EPDS were collected monthly for six months once the treatment began. Higher scores on the EDPS are indicative of higher levels of depression.

Usage

```
data(depression)
```

Format

A data frame with 427 rows and 5 variables:

subj a numeric vector giving the identifier of each woman .

group a factor giving the received treatment: "placebo" or "estrogen".

visit a numeric vector giving the number of months since the treatment began, where -1 indicates the pretreatment assessment of the EDPS.

dep a numeric vector giving the value of the EDPS.

depressd a numeric vector coded as 1 when the value of the EDPS is greater than or equal to 11 and coded as 0 in other cases.

Source

https://stats.idre.ucla.edu/spss/library/spss-librarypanel-data-analysis-using-gee/

References

Gregoire A.J.P., Kumar R., Everitt B., Henderson A.F. and Studd J.W.W. (1996) Transdermal oestrogen for treatment of severe postnatal depression, *The Lancet* 347, 930-933.

Examples

```
boxplot(dep ~ visit, data=subset(depression,group=="placebo"), at=c(0:6)-0.2,
        col="yellow", boxwex=0.25, outline=FALSE, xaxt="n", ylab="EDPS",
        xlab="Months since the treatment began", ylim=range(na.omit(depression$dep)))
boxplot(dep ~ visit, data=subset(depression,group=="estrogen"), add=TRUE,
        at=c(0:6)+0.2, col="blue", boxwex=0.25, outline=FALSE, xaxt="n")
axis(1, at=0:6, labels=c(-1,1:6))
legend(-1, 8, legend=c("placebo","estrogen"), fill=c("yellow","blue"),
       bty="n", title="Group")
```

---

dfbeta.glmgee *Dfbeta for Generalized Estimating Equations*

Description

Produces an approximation, better known as the *one-step approximation*, of the effect of deleting each cluster in turn on the estimates of the parameters in the linear predictor of a generalized estimating equation. This function also can produce a plot of those effects for a subset of the parameters in the linear predictor.

Usage

```
## S3 method for class 'glmgee'
dfbeta(
  model,
  level = c("clusters", "observations"),
  method = c("Preisser-Qaqish", "full"),
  coefs,
  identify,
  ...
)
```

Arguments

`model`	an object of class glmgee which is obtained from the fit of a generalized estimating equation.
`level`	an (optional) character string indicating the level for which the dfbeta statistic is required. The options are: cluster-level ("clusters") and observation-level ("observations"). By default, `level` is set to be "clusters".
`method`	an (optional) character string indicating the method of calculation for the *one-step approximation*. The options are: the *one-step approximation* described by Preisser and Qaqish (1996) in which the working-correlation matrix is assumed to be known ("Preisser-Qaqish"); and the "authentic" *one-step approximation* ("full"). By default, `method` is set to be "Preisser-Qaqish".
`coefs`	an (optional) character string which (partially) match with the names of some parameters in the linear predictor.
`identify`	an (optional) integer indicating the number of clusters to identify on the plot of dfbeta. This is only appropriate if `coefs` is specified.
`...`	further arguments passed to or from other methods. If `coefs` is specified then `...` may be used to include graphical parameters to customize the plot. For example, `col`, `pch`, `cex`, `main`, `sub`, `xlab`, `ylab`.

Details

The *one-step approximation* of the estimates of the parameters in the linear predictor of a GEE when the *i*-th cluster is excluded from the dataset is given by the vector obtained as the result of the first iteration of the fitting algorithm of that GEE when it is performed using: (1) a dataset in which the *i*-th cluster is excluded; and (2) a starting value which is the solution to the same GEE but based on the dataset inluding all clusters.

Value

A matrix with so many rows as clusters in the sample and so many columns as parameters in the linear predictor. The i-th row of that matrix corresponds to the difference between the estimates of the parameters in the linear predictor using all clusters and the *one-step approximation* of those estimates when the *i*-th cluster is excluded from the dataset.

References

Pregibon D. (1981). Logistic regression diagnostics. *The Annals of Statistics*, 9, 705-724.

Examples

```
mod <- size ~ poly(days,4) + treat
fit <- glmgee(mod, id=tree, family=Gamma("log"), data=spruces, corstr="Exchangeable")
dfbs <- dfbeta(fit, coefs="treat" ,col="red", lty=1, lwd=1, col.lab="blue",
               col.axis="blue", col.main="black", family="mono", cex=0.8, main="Dfbeta")

# Calculation by hand of dfbeta for the tree labeled by "N1T01"
idtree <- "N1T01"
onestep <- glmgee(mod, id=tree, family=Gamma("log"), data=spruces, corstr="Exchangeable",
```

```
                  start=coef(fit), subset=c(tree!=idtree), maxit=1)
coef(fit)-coef(onestep)
dfbs[rownames(dfbs)==idtree,]
```

`dfbeta.overglm` *Dfbeta for Negative Binomial and Beta-binomial Models*

Description

Produces an approximation, better known as the *one-step approximation*, of the effect on the parameter estimates of a negative binomial or beta-binomial model of deleting each individual in turn. This function also can produce a plot of those effects for a subset of the parameters in the linear predictor.

Usage

```
## S3 method for class 'overglm'
dfbeta(model, coefs, identify, ...)
```

Arguments

`model`	an object of class overglm which is obtained from the fit of a negative binomial or beta-binomial model.
`coefs`	an (optional) character string which (partially) match the names of some parameters in the linear predictor.
`identify`	an (optional) integer indicating the number of individuals to identify on the plot of dfbeta. This is only appropriate if the argument `coefs` is specified.
`...`	further arguments passed to or from other methods. If `coefs` is specified then `...` may be used to include graphical parameters to customize the plot. For example, `col`, `pch`, `cex`, `main`, `sub`, `xlab`, `ylab`.

Details

The *one-step approximation* of the estimates of the parameters in a negative binomial or beta-binomial model when the *i*-th individual is excluded from the dataset consists of the vector obtained as result of the first iteration of the Newthon-Raphson algorithm when it is performed using: (1) a dataset in which the *i*-th individual is excluded; and (2) a starting value which is the estimate of the same negative binomial or beta-binomial model but based on the dataset inluding all individuals.

Value

A matrix with so many rows as individuals in the sample and so many columns as parameters. The i-th row of that matrix corresponds to the difference between the estimates of the parameters using all individuals and the *one-step approximation* of those estimates when the *i*-th individual is excluded from the dataset.

References

Pregibon D. (1981). Logistic regression diagnostics. *The Annals of Statistics*, 9, 705-724.

Examples

```
fit <- glm(cbind(cells,200-cells) ~ tnf + ifn + tnf*ifn, family=binomial, data=cellular)
dfbs <- dfbeta(fit, coefs="tnf:ifn", col="red", lty=1, lwd=1, col.lab="blue",
               col.axis="blue", col.main="black", family="mono", cex=0.8, main="Dfbeta")
```

---

envelope *Normal QQ-plot with simulated envelope of model residuals*

Description

Generic function for building a normal QQ-plot with simulated envelope of residuals obtained from a fitted model.

Usage

```
envelope(object, ...)
```

Arguments

`object`	a fitted model object.
`...`	further arguments passed to or from other methods.

Value

A matrix with the simulated envelope and, optionally, a plot of it.

---

envelope.glm *Normal QQ-plot with simulated envelope of model residuals*

Description

Produces a normal QQ-plot with simulated envelope of residuals obtained from the fit of a generalized linear model.

Usage

```
## S3 method for class 'glm'
envelope(
  object,
  rep = 100,
  conf = 0.95,
  type = c("quantile", "deviance", "pearson"),
  standardized = FALSE,
  plot.it = TRUE,
  identify,
  ...
)
```

Arguments

`object`	an object of the class glm which is obtained from the fit of a generalized linear model.
`rep`	an (optional) positive integer indicating the number of replicates which should be used to build the simulated envelope. By default, `rep` is set to be 100.
`conf`	an (optional) value in the interval (0,1) indicating the confidence level which should be used to build the pointwise confidence intervals, which form the envelope. By default, `conf` is set to be 0.95.
`type`	a character string indicating the type of residuals which should be used. The available options are: randomized quantile ("quantile"), deviance ("deviance") and pearson ("pearson") residuals. By default, `type` is set to be "quantile".
`standardized`	an (optional) logical switch indicating if the residuals should be standardized by dividing by the square root of $(1 - h)$, where h is a measure of leverage. By default, `standardized` is set to be FALSE.
`plot.it`	an (optional) logical switch indicating if the normal QQ-plot with simulated envelope of residuals is required or just the data matrix in which it is based. By default, `plot.it` is set to be TRUE.
`identify`	an (optional) positive integer indicating the number of individuals to identify on the QQ-plot with simulated envelope of residuals. This is only appropriate if `plot.it=TRUE`.
`...`	further arguments passed to or from other methods. If `plot.it=TRUE` then `...` may be used to include graphical parameters to customize the plot. For example, `col`, `pch`, `cex`, `main`, `sub`, `xlab`, `ylab`.

Details

The simulated envelope is builded by simulating `rep` independent realizations of the response variable for each individual, which is accomplished taking into account the following: (1) the model assumption about the distribution of the response variable; (2) the estimates of the parameters in the linear predictor; and (3) the estimate of the dispersion parameter. The interest model is re-fitted `rep` times, as each time the vector of observed responses is replaced by one of the simulated samples. The residuals type *type* are computed and then ordered for each replicate, so that for each

$i = 1, 2, ..., n$, where n is the number of individuals in the sample, there is a random sample of size `rep` of the i-th order statistic of the residuals type *type*. Therefore, the simulated envelope is composed of the quantiles (1-`conf`)/2 and (1+`conf`)/2 of the random sample of size `rep` of the i-th order statistic of the residuals type *type* for $i = 1, 2, ..., n$. Families `quasi()`, `quasipoisson` and `quasibinomial` are not supported.

Value

A matrix with n rows and four columns: the first three (Lower limit, Median, and Upper limit) describe the simulated envelope, that is, each row corresponds to the quantiles (1-`conf`)/2, 0.5 and (1+`conf`)/2 of the random sample of size `rep` of the i-th order statistic of the residuals type `type` for $i = 1, 2, ..., n$; and the last one column (Residuals) contains the observed type `type` residuals.

References

Atkinson A.C. (1985) *Plots, Transformations and Regression*. Oxford University Press, Oxford.

Davison A.C. and Gigli A. (1989) Deviance Residuals and Normal Scores Plots. *Biometrika* 76, 211-221.

Dunn P.K. and Smyth G.K. (1996) Randomized Quantile Residuals. *Journal of Computational and Graphical Statistics* 5, 236-244.

Pierce D.A. and Schafer D.W. (1986) Residuals in Generalized Linear Models. *Journal of the American Statistical Association* 81, 977-986.

See Also

envelope.lm, envelope.overglm

Examples

```
# Example 1
fit1 <- glm(infections ~ frequency + location, family=poisson, data=swimmers)
envelope(fit1, type="quantile", col="red", pch=20,col.lab="blue",
         col.axis="blue",col.main="black",family="mono",cex=0.8)

# Example 2
fit2 <- glm(cbind(cells,200-cells) ~ tnf + ifn + tnf*ifn, family=binomial, data=cellular)
envelope(fit2, type="deviance",col="red", pch=20,col.lab="blue",
         col.axis="blue",col.main="black",family="mono",cex=0.8)

# Example 3
fit3 <- glm(cancer/exposed ~ dose, family=binomial, weights=exposed, data=bladder)
envelope(fit3, type="pearson", col="red", pch=20,col.lab="blue",
         col.axis="blue",col.main="black",family="mono",cex=0.8)

# Example 4
fit4 <- glm(cases ~ offset(log(population)) + city + age, family=poisson("log"), data=skincancer)
envelope(fit4, type="quantile", col="red", pch=20,col.lab="blue",
         col.axis="blue",col.main="black",family="mono",cex=0.8)
```

envelope.lm *Normal QQ-plot with simulated envelope of model residuals*

Description

Produces a normal QQ-plot with simulated envelope of residuals obtained from the fit of a normal linear model.

Usage

```
## S3 method for class 'lm'
envelope(
  object,
  rep = 100,
  conf = 0.95,
  type = c("external", "internal"),
  plot.it = TRUE,
  identify,
  ...
)
```

Arguments

object	an object of the class lm which is obtained from the fit of a normal linear model.
rep	an (optional) positive integer indicating the number of replicates which should be used to build the simulated envelope. By default, rep is set to be 100.
conf	an (optional) value in the interval (0,1) indicating the confidence level which should be used to build the pointwise confidence intervals, which form the envelope. By default, conf is set to be 0.95.
type	a character string indicating the type of residuals which should be used. The available options are: internally Studentized ("internal") and externally Studentized ("external") residuals. See Cook and Weisberg (1982, pages 18-20).
plot.it	an (optional) logical switch indicating if the normal QQ-plot with simulated envelope of residuals is required or just the data matrix in which it is based. By default, plot.it is set to be TRUE.
identify	an (optional) positive integer value indicating the number of individuals to identify on the QQ-plot with simulated envelope of residuals. This is only appropriate if plot.it=TRUE.
...	further arguments passed to or from other methods. If plot.it=TRUE then ... may be used to include graphical parameters to customize the plot. For example, col, pch, cex, main, sub, xlab, ylab.

Details

The simulated envelope is builded by simulating rep independent realizations of the response variable for each individual, which is accomplished taking into account the following: (1) the estimates of the parameters in the linear predictor; and (2) the estimate of the dispersion parameter. The interest model is re-fitted `rep` times, as each time the vector of observed responses is replaced by one of the simulated samples. The residuals type `type` are computed and then ordered for each replicate, so that for each $i = 1, 2, ..., n$, where n is the number of individuals in the sample, there is a random sample of size `rep` of the i-th order statistic of the residuals type `type`. Therefore, the simulated envelope is composed of the quantiles (1-`conf`)/2 and (1+`conf`)/2 of the random sample of size `rep` of the i-th order statistic of the residuals type `type` for $i = 1, 2, ..., n$.

Value

A matrix with n rows and four columns: the first three (Lower limit, Median, and Upper limit) describe the simulated envelope, that is, each row corresponds to the quantiles (1-`conf`)/2, 0.5 and (1+`conf`)/2 of the random sample of size `rep` of the i-th order statistic of the residuals type `type` for $i = 1, 2, ..., n$; and the last one column (Residuals) contains the observed type `type` residuals.

References

Atkinson A.C. (1985) *Plots, Transformations and Regression*. Oxford University Press, Oxford.

Cook R.D. and Weisberg S. (1982) *Residuals and Influence in Regression*. Chapman and Hall, New York.

See Also

envelope.glm, envelope.overglm

Examples

```
# Example 1
fit1 <- lm(Species ~ Biomass + pH + Biomass*pH, data=richness)
envelope(fit1, type="internal", col="red", pch=20, col.lab="blue",
         col.axis="blue", col.main="black", family="mono", cex=0.8)

# Example 2
fit2 <- lm(mpg ~ log(hp) + log(wt), data=mtcars)
envelope(fit2, type="external", col="red", pch=20, col.lab="blue",
         col.axis="blue", col.main="black", family="mono", cex=0.8)
```

`envelope.overglm` *Normal QQ-plot with simulated envelope of model residuals*

Description

Produces a normal QQ-plot with simulated envelope of residuals obtained from the fit of a negative binomial or beta-binomial regression model.

Usage

```
## S3 method for class 'overglm'
envelope(
  object,
  rep = 100,
  conf = 0.95,
  type = c("quantile", "response", "standardized"),
  plot.it = TRUE,
  identify,
  ...
)
```

Arguments

`object`	an object of the class overglm which is obtained from the fit of a negative binomial or beta-binomial model.
`rep`	an (optional) positive integer indicating the number of replicates which should be used to build the simulated envelope. By default, `rep` is set to be 100.
`conf`	an (optional) value in the interval (0,1) indicating the confidence level which should be used to build the pointwise confidence intervals, which form the envelope. By default, `conf` is set to be 0.95.
`type`	a character string indicating the type of residuals which should be used. The available options are: (1) the difference between the observed response and the fitted mean ("response"); (2) the standardized difference between the observed response and the fitted mean ("standardized"); (3) the randomized quantile residuals ("quantile"). By default, `type` is set to be "quantile".
`plot.it`	an (optional) logical switch indicating if the normal QQ-plot with simulated envelope of residuals is required or just the data matrix in which it is based. By default, `plot.it` is set to be TRUE.
`identify`	an (optional) positive integer value indicating the number of individuals to identify on the QQ-plot with simulated envelope of residuals. This is only appropriate if `plot.it=TRUE`.
`...`	further arguments passed to or from other methods. If `plot.it=TRUE` then `...` may be used to include graphical parameters to customize the plot. For example, `col`, `pch`, `cex`, `main`, `sub`, `xlab`, `ylab`.

Details

The simulated envelope is builded by simulating rep independent realizations of the response variable for each individual, which is accomplished taking into account the following: (1) the model assumption about the distribution of the response variable; (2) the estimates of the parameters in the linear predictor; and (3) the estimate of the dispersion parameter. The interest model is re-fitted `rep` times, as each time the vector of observed responses is replaced by one of the simulated samples. The residuals type `type` are computed and then ordered for each replicate, so that for each $i = 1, 2, ..., n$, where n is the number of individuals in the sample, there is a random sample of size `rep` of the i-th order statistic of the residuals type `type`. Therefore, the simulated envelope is

composed of the quantiles (1-conf)/2 and (1+conf)/2 of the random sample of size rep of the i-th order statistic of the residuals type type for $i = 1, 2, ..., n$.

Value

A matrix with n rows and four columns: the first three (Lower limit, Median, and Upper limit) describe the simulated envelope, that is, each row corresponds to the quantiles (1-conf)/2, 0.5 and (1+conf)/2 of the random sample of size rep of the i-th order statistic of the residuals type type for $i = 1, 2, ..., n$; and the last one column (Residuals) contains the observed type type residuals.

References

Atkinson A.C. (1985) *Plots, Transformations and Regression*. Oxford University Press, Oxford.

Dunn P.K. and Smyth G.K. (1996) Randomized Quantile Residuals. *Journal of Computational and Graphical Statistics* 5, 236-244.

See Also

envelope.lm, envelope.glm

Examples

```
## Example 1
fit1 <- overglm(infections ~ frequency + location, family="nb1(log)", data=swimmers)
envelope(fit1, rep=100, conf=0.95, type="quantile", col="red", pch=20, col.lab="blue",
         col.axis="blue", col.main="black", family="mono", cex=0.8)

## Example 2
fit2 <- overglm(cbind(fetuses,litter-fetuses) ~ tcpo + pht, family="bb(logit)", data=ossification)
envelope(fit2, rep=100, conf=0.95, type="quantile", col="red", pch=20, col.lab="blue",
         col.axis="blue", col.main="black", family="mono", cex=0.8)
```

---

estequa *Function to extract estimating equations*

Description

Extracts estimating equations evaluated at the parameter estimates and the observed data for a fitted model object.

Usage

```
estequa(model, ...)
```

Arguments

model	a fitted model object.
...	further arguments passed to or from other methods.

Value

A vector with the value of the estimating equations evaluated at the parameter estimates and the observed data.

`estequa.glm` *Estimating Equations in Generalized Linear Models*

Description

Extracts estimating equations evaluated at the parameter estimates and the observed data for a generalized linear model fitted to the data.

Usage

```
## S3 method for class 'glm'
estequa(model, ...)
```

Arguments

`model`	an object of the class glm which is obtained from the fit of a generalized linear model.
`...`	further arguments passed to or from other methods.

Value

A vector with the value of the estimating equations evaluated at the parameter estimates and the observed data.

Examples

```
## Example 1
Auto <- ISLR::Auto
mod <- mpg ~ cylinders + displacement + acceleration + origin + horsepower*weight
fit1 <- glm(mod, family=inverse.gaussian("log"), data=Auto)
estequa(fit1)

## Example 2
burn1000 <- aplore3::burn1000
burn1000 <- within(burn1000, death <- factor(death, levels=c("Dead","Alive")))
mod2 <- death ~ age + gender + race + tbsa + inh_inj + flame + age*inh_inj + tbsa*inh_inj
fit2 <- glm(mod2, family=binomial("logit"), data=burn1000)
estequa(fit2)

## Example 3
fit3 <- glm(cases ~ offset(log(population)) + city + age, family=poisson("log"), data=skincancer)
estequa(fit3)
```

estequa.glmgee *Estimating Equations in Generalized Estimating Equations*

Description

Extracts estimating equations evaluated at the parameter estimates and the observed data for a generalized estimating equation fitted to the data.

Usage

```
## S3 method for class 'glmgee'
estequa(model, ...)
```

Arguments

model	an object of class glmgee which is obtained from the fit of a generalized estimating equation.
...	further arguments passed to or from other methods.

Value

A vector with the value of the estimating equations evaluated at the parameter estimates and the observed data.

Examples

```
mod <- size ~ poly(days,4) + treat
fit <- glmgee(mod, id=tree, family=Gamma("log"), data=spruces, corstr="AR-1")
estequa(fit)
```

estequa.overglm *Estimating Equations in Negative Binomial and Beta-Binomial Models*

Description

Extracts estimating equations evaluated at the estimates of the parameters for a negative binomial or beta-binomial model fitted to the data.

Usage

```
## S3 method for class 'overglm'
estequa(model, ...)
```

Arguments

model	an object of the class overglm which is obtained from the fit of a negative binomial or beta-binomial regression model.
...	further arguments passed to or from other methods.

Value

A vector with the value of the estimating equations evaluated at the parameter estimates and the observed data.

Examples

```
## Example 1
fit1 <- overglm(tumors ~ group, family="nb3(log)", data=mammary)
estequa(fit1)

fit2 <- overglm(infections ~ frequency + location, family="nb1(log)", data=swimmers)
estequa(fit2)

fit3 <- overglm(cbind(cells,200-cells) ~ tnf + ifn + tnf*ifn, family="bb(logit)", data=cellular)
estequa(fit3)

fit4 <- overglm(cbind(fetuses,litter-fetuses) ~ pht*tcpo, family="bb(logit)", data=ossification)
estequa(fit4)
```

GHYC *Gosho-Hamada-Yoshimura's Criterion for Generalized Estimating Equations*

Description

Computes the Gosho-Hamada-Yoshimura's criterion (GHYC) for one or more objects of the class glmgee.

Usage

```
GHYC(..., verbose = TRUE)
```

Arguments

...	one or several objects of the class glmgee which are obtained from the fit of generalized estimating equations.
verbose	an (optional) logical switch indicating if should the report of results be printed. By default, verbose is set to be TRUE.

Value

A data.frame with the values of the GHYC for each glmgee object in the input.

References

Gosho M., Hamada C., Yoshimura I. (2011) Criterion for the Selection of a Working Correlation Structure in the Generalized Estimating Equation Approach for Longitudinal Balanced Data. *Communications in Statistics – Theory and Methods* 40, 3839–3856.

Gosho M. (2014) Criteria to Select a Working Correlation Structure in SAS. *Journal of Statistical Software, Code Snippets* 57.

See Also

QIC, CIC, RJC, AGPC, SGPC

Examples

```
## Example 1
mod <- size ~ poly(days,4) + treat
fit1 <- glmgee(mod, id=tree, family=Gamma("log"), data=spruces, corstr="Exchangeable")
fit2 <- update(fit1, corstr="AR-1")
fit3 <- update(fit1, corstr="Stationary-M-dependent(2)")
fit4 <- update(fit1, corstr="Independence")
GHYC(fit1, fit2, fit3, fit4)

## Example 2
mod <- dep ~ visit + group
fit1 <- glmgee(mod, id=subj, family=gaussian, corstr="Exchangeable", data=depression)
fit2 <- update(fit1, corstr="AR-1")
fit3 <- update(fit1, corstr="Non-Stationary-M-dependent(2)")
fit4 <- update(fit1, corstr="Independence")
GHYC(fit1, fit2, fit3, fit4)

## Example 3
mod <- depressd ~ visit + group
fit1 <- glmgee(mod, id=subj, family=binomial, corstr="Exchangeable", data=depression)
fit2 <- update(fit1, corstr="AR-1")
fit3 <- update(fit1, corstr="Stationary-M-dependent(2)")
fit4 <- update(fit1, corstr="Independence")
GHYC(fit1, fit2, fit3, fit4)
```

`glmgee` *Generalized Estimating Equations*

Description

Produces an object of the class `glmgee` in which the main results of a Generalized Estimating Equation fitted to the data are stored.

Usage

```
glmgee(
  formula,
  family = gaussian(),
  weights,
  id,
  waves,
  data,
  subset,
  corstr,
  corr,
  start = NULL,
  scale.fix = FALSE,
  scale.value = 1,
  toler = 1e-05,
  maxit = 50,
  adjr2 = FALSE,
  ...
)
```

Arguments

formula	a formula expression of the form response ~ predictors, which is a symbolic description of the linear predictor of the model to be fitted to the data.
family	a family object, that is, a list of functions and expressions for defining link and variance functions. Families supported are gaussian, binomial, poisson, Gamma, inverse gaussian and quasi. See the glm and family documentation. By default, family is set to be gaussian("identity").
weights	an (optional) vector of positive "prior weights" to be used in the fitting process. The length of weights should be the same as the number of observations.
id	a vector which identifies the subjects or clusters. The length of id should be the same as the number of observations.
waves	an (optional) positive integer-valued variable that is used to identify the order and spacing of observations within clusters. This argument is crucial when there are missing values and gaps in the data. By default, waves is equal to the integers from 1 to the size of each cluster.
data	a data frame in which to look for variables involved in the formula expression, as well as the variables specified in the arguments id and weights. The data are assumed to be sorted by id and time.
subset	an (optional) vector specifying a subset of observations to be used in the fitting process.
corstr	a character string specifying the working-correlation structure. The available options are: "Independence", "Unstructured", "Stationary-M-dependent(m)", "Non-Stationary-M-dependent(m)", "AR-1", "Exchangeable" and "User-defined", where *m* represents the lag of the dependence. By default, corstr is set to be "Independence".

`corr`	an (optional) square matrix of the same dimension of the maximum cluster size containing the user specified correlation. This is only appropriate if `corstr` is specified to be "User-defined".
`start`	an (optional) vector of starting values for the parameters in the linear predictor.
`scale.fix`	an (optional) logical variable. If TRUE, the scale parameter is fixed at the value of `scale.value`. By default, `scale.fix` is set to be FALSE.
`scale.value`	an (optional) numeric variable giving the value at which the scale parameter should be fixed. This is only appropriate if `scale.fix=TRUE`. By default, `scale.value` is set to be 1.
`toler`	an (optional) positive value which represents the convergence tolerance. The convergence is reached when the maximum of the relative differences between the values of the parameters in the linear predictor in consecutive iterations of the fitting algorithm is lower than `toler`. By default, `toler` is set to be 0.00001.
`maxit`	an (optional) integer value which represents the maximum number of iterations allowed for the fitting algorithm. By default, `maxit` is set to be 50.
`adjr2`	an (optional) logical variable. If TRUE, the adjusted R-squared based on the deviance is computed. By default, `adjr2` is set to be FALSE.
`...`	further arguments passed to or from other methods.

Details

If the maximum cluster size is 6 and for a cluster of size 4 `waves` is set to be 2, 4, 5, 6 then it means that the data on times 1 and 3 are missing, which should be taken into account by `glmgee` when the structure of the correlation matrix is assumed to be "Unstructured", "Stationary-M-dependent", "Non-Stationary-M-dependent" or "AR-1". If in this scenario `waves` is not specified then `glmgee` assumes that the available data for this cluster were taken on point times 1, 2, 3 and 4.

Value

an object of the class glmgee in which are stored the main results of a Generalized Estimating Equation fitted to the data. Some of those results can be easily accessed using functions as, for example, `print()`, `summary()`, `model.matrix()`, `estequa()`, `coef()`, `vcov()`, `fitted()`, `confint()` and `predict()`. In addition, the model fitted to the data can be assessed using functions as, for instance, anova.glmgee, residuals.glmgee, leverage.glmgee, dfbeta.glmgee, cooks.distance.glmgee and localInfluence.glmgee. The variable selection may be accomplished using stepCriterion.glmgee whereas the working–correlation–structure can be chosen by using criteria as QIC, CIC, GHYC, RJC, AGPC and SGPC.

References

Liang K.Y. and Zeger S.L. (1986) Longitudinal data analysis using generalized linear models. *Biometrika* 73, 13-22.

Zeger S.L. and Liang K.Y. (1986) Longitudinal data analysis for discrete and continuous outcomes. *Biometrics* 42, 121-130.

Hardin J.W. and Hilbe J.M. (2013). *Generalized Estimating Equations*. Chapman & Hall, London.

Examples

```
## Example 1
mod1 <- size ~ poly(days,4) + treat
fit1 <- glmgee(mod1, id=tree, family=Gamma("log"), corstr="AR-1", data=spruces)
summary(fit1)

## Example 2
mod2 <- depressd ~ visit + group
fit2 <- glmgee(mod2, id=subj, family=binomial("logit"), corstr="AR-1", data=depression)
summary(fit2)

## Example 3
mod3 <- dep ~ visit + group
fit3 <- glmgee(mod3, id=subj, corstr="Exchangeable", data=depression)
summary(fit3)

## Example 4
mod4 <- score ~ rinse + age + gender + smoke + time
fit4 <- glmgee(mod4, family=Gamma(log), id=subject, corstr="Exchangeable", data=rinse)
summary(fit4)

## Example 5
OME <- MASS::OME
mod5 <- cbind(Correct, Trials-Correct) ~ Loud + Age + OME
fit5 <- glmgee(mod5, family = binomial(cloglog), id = ID, corstr = "Exchangeable", data = OME)
summary(fit5)
```

GUIDE *Guidelines for Urinary Incontinence Discussion and Evaluation*

Description

These data arose from a randomized controlled trial that assessed if provider adherence to a set of guidelines for treatment of patients with urinary incontinence (UI) affected patient outcomes. Data were collected on 137 elderly patients from 38 medical practices. The number of patients per practice ranged from 1 to 8 and the median was 4 patients. The interest of the present analysis is to determine what predicts whether or not a patient considers their UI a problem that interferes with him/her daily life.

Usage

```
data(GUIDE)
```

Format

A data frame with 137 rows and 7 variables:

bothered a numeric vector giving the answer to the following: Do you consider this accidental loss of urine a problem that interferes with your day to day activities or bothers you in other ways? 1 for "Yes" and 0 for "No".

gender a factor giving the patient's gender: "Male" or "Female".

age a numeric vector giving the standardized age: (age in years - 76)/10.

dayacc a numeric vector giving the patient's report of the number of leaking accidents they experience in an average day (derived from number of accidents reported per week).

severe a factor giving the severity of the loss of urine: "1" if there is only some moisture; "2" if the patient wet the underwear; "3" if the urine trickled down the thigh; and "4" if the patient wet the floor.

toilet a numeric vector giving the patient's report on the number of times during the day he (or she) usually go to the toilet to urinate.

practice a character string giving the identifier of the medical practice.

Source

http://www.bios.unc.edu/~preisser/personal/uidata/preqaq99.dat

References

Hammill B.G. and Preisser J.S. (2006) A SAS/IML software program for GEE and regression diagnostics. *Computational Statistics & Data Analysis* 51, 1197-1212.

Jung K.-M. (2008) Local Influence in Generalized Estimating Equations. *Scandinavian Journal of Statistics* 35, 286-294.

Examples

```
mod <- bothered ~ gender + age + dayacc + severe + toilet
fit <- glmgee(mod, family=binomial(logit), id=practice, corstr="Exchangeable", data=GUIDE)
summary(fit)
```

---

gvif *Generalized Variance Inflaction Factor*

Description

Computes the generalized variance inflaction factor (GVIF) for a fitted model object.

Usage

```
gvif(model, ...)
```

Arguments

`model`	a fitted model object.
`...`	further arguments passed to or from other methods.

Value

An object with the values of the GVIF for all effects in the model.

gvif.glm *Generalized Variance Inflaction Factor*

Description

Computes the generalized variance inflaction factor (GVIF) for a generalized linear model.

Usage

```
## S3 method for class 'glm'
gvif(model, verbose = TRUE, ...)
```

Arguments

model	an object of the class glm which is obtained from the fit of a generalized linear model.
verbose	an (optional) logical switch indicating if should the report of results be printed. By default, verbose is set to be TRUE.
...	further arguments passed to or from other methods.

Value

A matrix with so many rows as effects in the model and three columns: *(1)* the values of GVIF; *(2)* the number of degrees of freedom; and *(3)* the values of GVIF^1/(2*df).

If the number of degrees of freedom is 1 then the GVIF reduces to the Variance Inflation Factor (VIF).

References

Fox J. and Monette G. (1992) Generalized collinearity diagnostics, *JASA* 87, 178–183.

See Also

gvif.lm

Examples

```
## Example 1
Auto <- ISLR::Auto
fit1 <- glm(mpg ~ weight*horsepower, family=inverse.gaussian("log"), data=Auto)
gvif(fit1)

## Example 2
burn1000 <- aplore3::burn1000
mod <- death ~ age + tbsa + inh_inj + age*inh_inj + tbsa*inh_inj
fit2 <- glm(mod, family=binomial("logit"), data=burn1000)
gvif(fit2)
```

```
## Example 3
fit3 <- glm(rtime ~ log(distance) + log(cclimb), family=Gamma("log"), data=races)
gvif(fit3)
```

`gvif.lm` *Generalized Variance Inflaction Factor*

Description

Computes the generalized variance inflaction factor (GVIF) for a normal linear model.

Usage

```
## S3 method for class 'lm'
gvif(model, verbose = TRUE, ...)
```

Arguments

`model`	an object of the class lm which is obtained from the fit of a normal linear model.
`verbose`	an (optional) logical switch indicating if should the report of results be printed. By default, `verbose` is set to be TRUE.
`...`	further arguments passed to or from other methods.

Value

A matrix with so many rows as effects in the model and three columns: *(1)* the values of GVIF; *(2)* the number of degrees of freedom; and *(3)* the values of GVIF^1/(2*df).

If the number of degrees of freedom is 1 then the GVIF reduces to the Variance Inflation Factor (VIF).

References

Fox J. and Monette G. (1992) Generalized collinearity diagnostics, *JASA* 87, 178–183.

See Also

gvif.glm

Examples

```
## Example 1
fit1 <- lm(mpg ~ log(hp) + log(wt), data=mtcars)
gvif(fit1)

## Example 2
fit2 <- lm(Species ~ Biomass + pH, data=richness)
gvif(fit2)
```

```
## Example 3
whiteside <- MASS::whiteside
fit3 <- lm(Gas ~ Temp + Insul + Temp*Insul, data=whiteside)
gvif(fit3)
```

`hltest` *The Hosmer-Lemeshow Goodness-of-Fit Test*

Description

Computes the Hosmer-Lemeshow goodness-of-fit test for a generalized linear model fitted to binary responses.

Usage

```
hltest(model, verbose = TRUE, ...)
```

Arguments

`model`	an object of the class glm which is obtained from the fit of a generalized linear model where the distribution for the response variable is assumed to be binomial.
`verbose`	an (optional) logical switch indicating if should the report of results be printed. By default, `verbose` is set to be TRUE.
`...`	further arguments passed to or from other methods.

Value

A list which contains the following objects:

- `hm`: A matrix with the values of Group, Size, Observed and Expected, which are required to compute the statistic of the test.
- `statistic`: The value of the statistic of the test.
- `df`: The number of degrees of freedom, given by the number of groups minus 2.
- `p.value`: The *p*-value of the test computed using the Chi-square distribution.

References

Hosmer, D.W. and Lemeshow, S. (2000) *Applied Logistic Regression. 2nd ed.* John Wiley & Sons, New York.

Examples

```
## Example 1
fit1 <- glm(cancer/exposed ~ dose, weights=exposed, family=binomial("logit"), data=bladder)
hltest(fit1)

## Example 2
fit2 <- glm(cancer/exposed ~ dose, weights=exposed, family=binomial("logit"), data=liver)
hltest(fit2)

## Example 3
burn1000 <- aplore3::burn1000
mod <- death ~ age + tbsa + inh_inj + age*inh_inj + tbsa*inh_inj
fit3 <- glm(mod, family=binomial("logit"), data=burn1000)
hltest(fit3)
```

---

leverage *Leverage*

Description

Computes leverage measures for a fitted model object.

Usage

```
leverage(object, ...)
```

Arguments

object	a fitted model object.
...	further arguments passed to or from other methods.

Value

An object with the values of the leverage measures.

---

leverage.glmgee *Leverage for Generalized Estimating Equations*

Description

Computes and, optionally, displays a graph of the leverage measures at the cluster- and observation-level.

Usage

```
## S3 method for class 'glmgee'
leverage(
  object,
  level = c("clusters", "observations"),
  plot.it = FALSE,
  identify,
  ...
)
```

Arguments

object	an object of class glmgee which is obtained from the fit of a generalized estimating equation.
level	an (optional) character string indicating the level for which the leverage measures are required. The options are: cluster-level ("clusters") and observation-level ("observations"). By default, level is set to be "clusters".
plot.it	an (optional) logical indicating if the plot of the measures of leverage are required or just the data matrix in which that plot is based. By default, plot.it is set to be FALSE.
identify	an (optional) integer indicating the number of (level=``clusters'') or observations (level=``observations'') to identify on the plot of the leverage measures. This is only appropriate if plot.it is specified to be TRUE.
...	further arguments passed to or from other methods. If plot.it is specified to be TRUE then ... may be used to include graphical parameters to customize the plot. For example, col, pch, cex, main, sub, xlab, ylab.

Value

A vector with the values of the leverage measures with so many rows as clusters (level=``clusters'') or observations (level=``observations'') in the sample.

References

Preisser, J.S. and Qaqish, B.F. (1996). Deletion diagnostics for generalised estimating equations. *Biometrika*, 83, 551-562.

Hammill, B.G. and Preisser J.S. (2006). A SAS/IML software program for GEE and regression diagnostics. *Computational Statistics & Data Analysis*, 51, 1197-1212.

Examples

```
#Example 1
OME <- MASS::OME
mod <- cbind(Correct, Trials-Correct) ~ Loud + Age + OME
fit1 <- glmgee(mod, family = binomial(cloglog), id = ID, corstr = "Exchangeable", data = OME)
leverage(fit1,level="clusters",plot.it=TRUE)
```

```
#Example 2
mod <- bothered ~ gender + age + dayacc + severe + toilet
fit2 <- glmgee(mod, family=binomial(logit), id=practice, corstr="Exchangeable", data=GUIDE)
summary(fit2)
par(mfrow=c(1,2))
leverage(fit2,level="clusters",plot.it=TRUE)
leverage(fit2,level="observations",plot.it=TRUE)
```

`liver` *Liver cancer in mice*

Description

Female mice were continuously fed dietary concentrations of 2-Acetylaminofluorene (2-AAF), a carcinogenic and mutagenic derivative of fluorene. Serially sacrificed, dead or moribund mice were examined for tumors and dates of deaths were recorded. These data consist of the incidences of liver neoplasms in the mice observed during 18 months.

Usage

```
data(liver)
```

Format

A data frame with 8 rows and 3 variables:

dose a numeric vector giving the dose, in parts per 10^4, of 2-AAF.

exposed a numeric vector giving the number of mice exposed to each dose of 2-AAF.

cancer a numeric vector giving the number of mice with liver cancer for each dose of 2-AAF.

References

Zhang H. and Zelterman D. (1999) Binary Regression for Risks in Excess of Subject-Specific Thresholds. *Biometrics* 55, 1247-1251.

See Also

bladder

Examples

```
barplot(100*cancer/exposed ~ dose, beside=TRUE, data=liver, col="red",
        xlab="Dose of 2-AAF", ylab="% of mice with liver cancer")
```

localInfluence *Local Influence*

Description

Computes measures of local influence for a fitted model object.

Usage

```
localInfluence(object, ...)
```

Arguments

object	a fitted model object.
...	further arguments passed to or from other methods.

Value

An object with the measures of local influence.

localInfluence.glmgee *Local Influence for Generalized Estimating Equations*

Description

Computes some measures and, optionally, display graphs of them to perform influence analysis based on the approaches described in Cook (1986) and Jung (2008).

Usage

```
## S3 method for class 'glmgee'
localInfluence(
  object,
  type = c("total", "local"),
  perturbation = c("cw-clusters", "cw-observations", "response", "covariate"),
  covariate,
  coefs,
  plot.it = FALSE,
  identify,
  ...
)
```

Arguments

object	an object of class glmgee obtained from the fit of a generalized estimating equation.
type	an (optional) character string indicating the type of approach to study the local influence. The options are: the absolute value of the elements of the eigenvector which corresponds to the maximum absolute eigenvalue ("local"); and the absolute value of the elements of the main diagonal ("total"). By default, type is set to be "total".
perturbation	an (optional) character string indicating the perturbation scheme to apply. The options are: case weight perturbation of clusters ("cw-clusters"); Case weight perturbation of observations ("cw-observations"); perturbation of covariates ("covariate"); and perturbation of response ("response"). By default, perturbation is set to be "cw-clusters".
covariate	an character string which (partially) match with the names of one of the parameters in the linear predictor. This is only appropriate if perturbation="covariate".
coefs	an (optional) character string which (partially) match with the names of some of the parameters in the linear predictor.
plot.it	an (optional) logical indicating if the plot of the measures of local influence is required or just the data matrix in which that plot is based. By default, plot.it is set to be FALSE.
identify	an (optional) integer indicating the number of clusters/observations to identify on the plot of the measures of local influence. This is only appropriate if plot.it=TRUE.
...	further arguments passed to or from other methods. If plot.it=TRUE then ... may be used to include graphical parameters to customize the plot. For example, col, pch, cex, main, sub, xlab, ylab.

Value

A matrix as many rows as clusters/observations in the sample and one column with the values of the measures of local influence.

References

Cook D. (1986) Assessment of Local Influence. *Journal of the Royal Statistical Society: Series B (Methodological)* 48, 133-155.

Jung K.-M. (2008) Local Influence in Generalized Estimating Equations. *Scandinavian Journal of Statistics* 35, 286-294.

Examples

```
mod <- size ~ poly(days,4) + treat
fit <- glmgee(mod, id=tree, family=Gamma("log"), corstr="AR-1", data=spruces)
summary(fit)
localInfluence(fit,type="total",perturbation="cw-clusters",coefs="treat",plot.it=TRUE)
```

mammary *Ability of retinyl acetate to prevent mammary cancer in rats*

Description

A total of 76 female rats were injected with a carcinogen for mammary cancer. Then, all animals were given retinyl acetate (retinoid) to prevent mammary cancer for 60 days. After this phase, the 48 animals that remained tumor-free were randomly assigned to continue the retinoid prophylaxis or control. Rats were then palpated for tumors twice weekly, and observations ended 182 days after the initial carcinogen injections began. The main objective of analysis was to assess the difference in the development of tumors between the treated and control groups. See Morel and Nagaraj (2012, page 63).

Usage

```
data(mammary)
```

Format

A data frame with 48 rows and 2 variables:

group a factor giving the group to which the rat was assigned: "retinoid" or "control".

tumors a numeric vector giving the number of tumors identified on the rat.

References

Lawless J.F. (1987) Regression Methods for Poisson Process Data. *Journal of the American Statistical Association*, 82, 808-815.

Morel J.G. and Nagaraj N.K. (2012) *Overdispersion Models in SAS*. SAS Institute Inc., Cary, North Carolina, USA.

Examples

```
boxplot(tumors ~ group, data=mammary, outline=FALSE, xlab="Group",
        ylab="Number of tumors", col=c("yellow","blue"))
```

ossification *Teratogenic effects of phenytoin and trichloropropene oxide*

Description

The data come from a 2x2 factorial design with 81 pregnant mice. In the experiment each pregnant mouse was randomly allocated to an control group and three treated groups, which received daily, by gastric gavages, 60 mg/kg of phenytoin, 100 mg/kg of trichloropropene oxide, or 60 mg/kg phenytoin and 100 mg/kg of trichloropropene oxide. On day 18 of gestation, fetuses were recovered, stained, and cleared. Then, by visual inspection, the presence or absence of ossification was determined for the different joints of the right and left forepaws. The purpose of the experiment was to investigate the synergy of phenytoin and trichloropropene oxide to produce ossification at the phalanges, that is, teratogenic effects. See Morel and Nagaraj (2012, page 103).

Usage

```
data(ossification)
```

Format

A data frame with 81 rows and 4 variables:

fetuses a numeric vector giving the number of fetuses showing ossification on the left middle third phalanx.

litter a numeric vector giving the litter size.

pht a factor giving the dose (mg/kg) of phenytoin: "0 mg/kg" or "60 mg/kg".

tcpo a factor giving the dose (mg/kg) of trichloropropene oxide: "0 mg/kg" or "100 mg/kg".

References

Morel J.G. and Neerchal N.K. (1997) Clustered binary logistic regression in teratology data using a finite mixture distribution. *Statistics in Medicine* 16, 2843-2853.

Morel J.G. and Nagaraj N.K. (2012) *Overdispersion Models in SAS*. SAS Institute Inc., Cary, North Carolina, USA.

Examples

```
boxplot(100*fetuses/litter ~ pht, data=subset(ossification, tcpo=="0 mg/kg"),
        at=c(1:2)-0.2, col="yellow", boxwex=0.25, outline=FALSE, xaxt="n",
        xlab="Dose of PHT", ylab="% of fetuses showing ossification")
boxplot(100*fetuses/litter ~ pht, data=subset(ossification, tcpo=="100 mg/kg"),
        add=TRUE, at=c(1:2)+0.2, col="blue", boxwex=0.25, outline=FALSE, xaxt="n")
axis(1, at=1:2, labels=levels(ossification$pht))
legend(0.25, 20, legend=c("0 mg/kg","100 mg/kg"), fill=c("yellow","blue"),
       bty="n", cex=0.9, title="Dose of TCPO")
```

`overglm` *Negative Binomial and Beta-Binomial Regression Models*

Description

Produces an object of the class overglm in which are stored the main results of the negative binomial or beta-binomial regression model fitted to the data.

Usage

```
overglm(formula, family, weights, data, subset, start = NULL, ...)
```

Arguments

`formula`	a `formula` expression of the form `response ~ predictors`, which is a symbolic description of the linear predictor of the model to be fitted to the data.
`family`	a character string which describe the distribution of the response variable and the link function. The following distributions are supported: negative binomial I ("nb1"), negative binomial II ("nb2"), negative binomial III ("nb3"), and beta-binomial ("bb"). Link functions available for negative binomial and beta-binomial regression models are the same than those available for Poisson and binomial regression models, respectively. See family documentation.
`weights`	an (optional) vector of positive "prior weights" to be used in the fitting process. The length of `weights` should be the same as the number of observations.
`data`	an (optional) `data frame` in which to look for variables involved in the `formula` expression, as well as the variable specified in the argument `weights`.
`subset`	an (optional) vector specifying a subset of observations to be used in the fitting process.
`start`	an (optional) vector of starting values for the parameters in the linear predictor.
`...`	further arguments passed to or from other methods.

Details

The negative binomial distributions can be obtained as mixture of the Poisson and Gamma distributions. Let $Y|\lambda$ ~ Poisson(λ), where E$(Y|\lambda) = \lambda$ and Var$(Y|\lambda) = \lambda$, and λ ~ Gamma(β, α), where E$(\lambda) = \beta$ and Var$(\lambda) = \alpha\beta^2$. Therefore,

(1) If $\beta = \mu$ and $\alpha = \phi$ then Y ~ Binomial Negativa I(μ, ϕ), E$(Y) = \mu$ and Var$(Y) = \mu + \phi\mu^2$.

(2) If $\beta = \mu$ and $\alpha = \phi/\mu$ then Y ~ Binomial Negativa II(μ, ϕ), E$(Y) = \mu$ and Var$(Y) = (\phi+1)\mu$.

(3) If $\beta = \mu$ and $\alpha = \phi\mu$ then Y ~ Binomial Negativa III(μ, ϕ), E$(Y) = \mu$ and Var$(Y) = \mu + \phi\mu^3$.

So, the regression models based on the negative binomial distributions are alternatives to the Poisson regression model under the presence of overdispersion.

The beta-binomial distribution can be obtained as mixture of the binomial and Beta distributions. Let $mY|\pi$ ~ Binomial(m, π), where E$(Y|\pi) = \pi$ and Var$(Y|\pi) = (1/m)\pi(1-\pi)$, and π ~

Beta(μ, ϕ), where $\mathrm{E}(\pi) = \mu$ and $\mathrm{Var}(\pi) = [\phi/(\phi+1)]\mu(1-\mu)$, then mY ~ Beta Binomial(m, μ, ϕ), $\mathrm{E}(Y) = \mu$ and $\mathrm{Var}(Y) = (1/m)\mu(1-\mu)[1+(m-1)\phi/(\phi+1)]$.

So, the regression model based on the beta-binomial distribution is an alternative to the binomial regression model under the presence of overdispersion.

Value

an object of the class overglm in which are stored the main results of the model fitted to the data. Some of those results can be easily accessed using functions as, for example, `print()`, `summary()`, `model.matrix()`, `estequa()`, `coef()`, `vcov()`, `logLik()`, `fitted()`, `confint()` and `predict()`. In addition, the model fitted to the data can be assessed using functions as, for instance, anova.overglm, residuals.overglm, dfbeta.overglm, cooks.distance.overglm and envelope.overglm. The variable selection may be accomplished using stepCriterion.overglm.

References

Lawless J.F. (1987) Negative binomial and mixed poisson regression, *The Canadian Journal of Statistics* 15, 209-225.

Crowder M. (1978) Beta-binomial anova for proportions, *Journal of the Royal Statistical Society Series C (Applied Statistics)* 27, 34-37.

Examples

```
fit1 <- overglm(tumors ~ group, family="nb3(log)", data=mammary)
summary(fit1)

fit2 <- overglm(infections ~ frequency + location, family="nb1(log)", data=swimmers)
summary(fit2)

fit3 <- overglm(cbind(cells,200-cells) ~ tnf + ifn + tnf*ifn, family="bb(logit)", data=cellular)
summary(fit3)

fit4 <- overglm(cbind(fetuses,litter-fetuses) ~ pht*tcpo, family="bb(logit)", data=ossification)
summary(fit4)
```

---

predict.glmgee *Predictions for Generalized Estimating Equations*

Description

Produces predictions and optionally estimates standard errors of those predictions from a fitted generalized estimating equation.

Usage

```
## S3 method for class 'glmgee'
predict(
  object,
  ...,
  newdata,
  se.fit = FALSE,
  type = c("link", "response"),
  varest = c("robust", "df-adjusted", "model", "bias-corrected")
)
```

Arguments

object	an object of the class glmgee which is obtained from the fit of a generalized estimating equation.
...	further arguments passed to or from other methods.
newdata	an (optional) data frame in which to look for variables with which to predict. If omitted, the fitted linear predictors are used.
se.fit	an (optional) logical switch indicating if standard errors are required. By default, se.fit is set to be FALSE.
type	an (optional) character string giving the type of prediction required. The default, "link", is on the scale of the linear predictors, and the alternative, "response", is on the scale of the response variable.
varest	an (optional) character string indicating the type of estimator which should be used to the variance-covariance matrix of the interest parameters. The available options are: robust sandwich-type estimator ("robust"), degrees-of-freedom-adjusted estimator ("df-adjusted"), bias-corrected estimator ("bias-corrected"), and the model-based or naive estimator ("model"). By default, varest is set to be "robust".

Value

A matrix with so many rows as newdata and one column with the predictions. If se.fit=TRUE then a second column with estimates standard errors is included.

Examples

```
mod <- size ~ poly(days,4) + treat
fit <- glmgee(mod, id=tree, family=Gamma("log"), data=spruces, corstr="Stationary-M-dependent(2)")
newdata <- data.frame(days=c(556,556),treat=as.factor(c("normal","ozone_enriched")))
predict(fit,newdata=newdata,type="response",se.fit=TRUE)
```

`predict.overglm` *Predictions for Negative Binomial and Beta-Binomial regression models*

Description

Produces predictions and optionally estimates standard errors of those predictions from a fitted negative binomial or beta-binomial regression model.

Usage

```
## S3 method for class 'overglm'
predict(object, ..., newdata, se.fit = FALSE, type = c("link", "response"))
```

Arguments

`object`	an object of class overglm which is obtained from the fit of a negative binomial or beta-binomial regression model.
`...`	further arguments passed to or from other methods.
`newdata`	an (optional) `data frame` in which to look for variables with which to predict. If omitted, the fitted linear predictors are used.
`se.fit`	an (optional) logical switch indicating if standard errors are required. By default, `se.fit` is set to be FALSE.
`type`	an (optional) character string giving the type of prediction required. The default, "link", is on the scale of the linear predictors, and the alternative, "response", is on the scale of the response variable.

Value

A matrix with so many rows as `newdata` and one column with the predictions. If `se.fit=TRUE` then a second column with estimates standard errors is included.

Examples

```
fit1 <- overglm(tumors ~ group, family="nb3(log)", data=mammary)
newdata <- data.frame(group=as.factor(c("control","retinoid")))
predict(fit1,newdata=newdata,type="response",se.fit=TRUE)

fit2 <- overglm(cbind(cells,200-cells) ~ tnf + ifn + tnf*ifn, family="bb(logit)", data=cellular)
newdata <- data.frame(tnf=c(0,100),ifn=c(100,0))
predict(fit2,newdata=newdata,type="response",se.fit=TRUE)
```

QIC *QIC for Generalized Estimating Equations*

Description

Computes the quasi-likelihood under the independence model criterion (QIC) for one or more objects of the class glmgee.

Usage

```
QIC(..., k = 2, u = FALSE, verbose = TRUE)
```

Arguments

...	one or several objects of the class glmgee which are obtained from the fit of generalized estimating equations.
k	an (optional) non-negative value giving the magnitude of the penalty. By default, k is set to be 2.
u	an (optional) logical switch indicating if QIC should be replaced by QICu. By default, u is set to be FALSE.
verbose	an (optional) logical switch indicating if should the report of results be printed. By default, verbose is set to be TRUE.

Value

A data.frame with the values of -2*quasi-likelihood, the number of parameters in the linear predictor, and the value of QIC (or QICu if u=TRUE) for each glmgee object in the input.

References

Pan W. (2001) Akaike's information criterion in generalized estimating equations, *Biometrics* 57, 120-125.

Hin L-Y, Carey V.J., Wang Y-G (2007) Criteria for Working–Correlation–Structure Selection in GEE: Assessment via Simulation. *The American Statistician* 61, 360–364.

See Also

CIC, GHYC, RJC, AGPC, SGPC

Examples

```
## Example 1
mod <- size ~ poly(days,4) + treat
fit1 <- glmgee(mod, id=tree, family=Gamma("log"), data=spruces, corstr="Exchangeable")
fit2 <- update(fit1, corstr="AR-1")
fit3 <- update(fit1, corstr="Stationary-M-dependent(2)")
fit4 <- update(fit1, corstr="Independence")
```

```
QIC(fit1, fit2, fit3, fit4)

## Example 2
mod <- dep ~ visit + group
fit1 <- glmgee(mod, id=subj, family=gaussian, corstr="Exchangeable", data=depression)
fit2 <- update(fit1, corstr="AR-1")
fit3 <- update(fit1, corstr="Non-Stationary-M-dependent(2)")
fit4 <- update(fit1, corstr="Independence")
QIC(fit1, fit2, fit3, fit4)

## Example 3
mod <- depressd ~ visit + group
fit1 <- glmgee(mod, id=subj, family=binomial, corstr="Exchangeable", data=depression)
fit2 <- update(fit1, corstr="AR-1")
fit3 <- update(fit1, corstr="Stationary-M-dependent(2)")
fit4 <- update(fit1, corstr="Independence")
QIC(fit1, fit2, fit3, fit4)
```

`races` *Hill races in Scotland*

Description

Each year the Scottish Hill Runners Association publishes a list of hill races in Scotland for the year. These data consist of record time, distance, and cumulative climb of 35 of those races. The aim of the statistical analysis of these data is to explain the differences between the record time of the races using their differences on distance and cumulative climb. See Agresti (2015, page 62).

Usage

```
data(races)
```

Format

A data frame with 35 rows and 4 variables:

race a character vector giving the names of the races.

distance a numeric vector giving the distance, in miles, of the races.

cclimb a numeric vector giving the cumulative climb, in thousands of feet, of the races.

rtime a numeric vector giving the record time, in minutes, of the races.

Source

http://users.stat.ufl.edu/~aa/glm/data/

References

Agresti A. (2015) *Foundations of Linear and Generalized Linear Models*. John Wiley & Sons, New Jersey.

Examples

```
races2 <- within(races,cli <- cut(cclimb, include.lowest=TRUE,
                                  breaks=quantile(cclimb, probs=c(0:2)/2),
                                  labels=c("low","high")))
with(races2,{
    plot(log(distance), log(rtime),
         col=apply(as.matrix(cli),1,function(x) switch(x,"low"="red","high"="blue")),
         pch=apply(as.matrix(cli),1,function(x) switch(x,"low"=15,"high"=16)))
    legend(0.7, 5.4, legend=c("low","high"), title="Cumulative climb",
           col=c("red","blue"), pch=c(15,16), bty="n")
})
```

---

residuals.glmgee *Residuals for Generalized Estimating Equations*

Description

Calculates residuals for a fitted generalized estimating equation.

Usage

```
## S3 method for class 'glmgee'
residuals(
  object,
  ...,
  type = c("mahalanobis", "pearson", "deviance"),
  plot.it = FALSE,
  identify
)
```

Arguments

object	a object of the class glmgee obtained from the fit of a generalized estimating equation.
...	further arguments passed to or from other methods
type	an (optional) character string giving the type of residuals which should be returned. The available options are: (1) "pearson"; (2) "deviance"; (3) the distance between the observed response vector and the fitted mean vector using a metric based on the product between the cluster size and fitted variance-covariance matrix ("mahalanobis"). By default, type is set to be "mahalanobis".
plot.it	an (optional) logical switch indicating if a plot of the residuals is required. By default, plot.it is set to be FALSE.
identify	an (optional) integer value indicating the number of individuals/clusters to identify on the plot of residuals. This is only appropriate when plot.it=TRUE.

Value

A vector with the observed residuals type type.

Examples

```
mod <- size ~ poly(days,4) + treat
fit <- glmgee(mod, id=tree, family=Gamma("log"), data=spruces, corstr="AR-1")
residuals(fit, type="mahalanobis", col="red", pch=20, col.lab="blue",
          col.axis="blue", col.main="black", family="mono", cex=0.8)
```

---

residuals.overglm *Residuals for Negative Binomial and Beta-Binomial Regression Models*

Description

Calculates residuals for a fitted negative binomial or beta-binomial model.

Usage

```
## S3 method for class 'overglm'
residuals(
  object,
  ...,
  type = c("quantile", "standardized", "response"),
  plot.it = TRUE,
  identify
)
```

Arguments

object	an object of the class overglm obtained from the fit of a negative binomial or beta-binomial model.
...	further arguments passed to or from other methods.
type	an (optional) character string giving the type of residuals which should be returned. The available options are: (1) the difference between the observed response and the fitted mean ("response"); (2) the standardized difference between the observed response and the fitted mean ("pearson"); (3) the randomized quantile residuals ("quantile"). By default, type is set to be "quantile".
plot.it	an (optional) logical switch indicating if a plot of the residuals is required. By default, plot is set to be TRUE.
identify	an (optional) integer value indicating the number of individuals to identify on the plot of residuals. This is only appropriate when plot.it=TRUE.

Value

A vector with the observed residuals type type.

References

Dunn P.K. and Smyth G.K. (1996) Randomized Quantile Residuals. *Journal of Computational and Graphical Statistics*, 5, 236-244.

Examples

```
## Example 1
fit1 <- overglm(cbind(fetuses,litter-fetuses) ~ pht + tcpo, family="bb(logit)", data=ossification)
residuals(fit1, type="quantile", col="red", pch=20, col.lab="blue",
          col.axis="blue", col.main="black", family="mono", cex=0.8)

## Example 2
fit2 <- overglm(infections ~ location + frequency, family="nb1(log)", data=swimmers)
residuals(fit2, type="quantile", col="red", pch=20, col.lab="blue",
          col.axis="blue", col.main="black", family="mono", cex=0.8)
```

---

residuals2 *Residuals for Linear and Generalized Linear Models*

Description

Computes residuals for a fitted linear or generalized linear model.

Usage

```
residuals2(object, type, standardized = FALSE, plot.it = TRUE, identify, ...)
```

Arguments

`object`	a object of the class lm or glm obtained from the fit of a linear or a generalized linear model.
`type`	an (optional) character string giving the type of residuals which should be returned. The available options for LMs are: (1) externally studentized ("external"); (2) internally studentized ("internal") (default). The available options for GLMs are: (1) "pearson"; (2) "deviance"; (3) "quantile" (default).
`standardized`	an (optional) logical switch indicating if the residuals should be standardized by dividing by the square root of $(1 - h)$, where h is a measure of leverage. By default, `standardized` is set to be FALSE.
`plot.it`	an (optional) logical switch indicating if a plot of the residuals is required. By default, `plot.it` is set to be TRUE.
`identify`	an (optional) integer value indicating the number of individuals to identify on the plot of residuals. This is only appropriate when `plot.it=TRUE`.
`...`	further arguments passed to or from other methods

Value

A vector with the observed residuals type `type`.

Examples

```
# Example 1
fit1 <- lm(Species ~ Biomass + pH + Biomass*pH, data=richness)
residuals2(fit1, type="external", col="red", pch=20, col.lab="blue",
           col.axis="blue", col.main="black", family="mono", cex=0.8)

# Example 2
fit2 <- glm(infections ~ frequency + location, family=poisson, data=swimmers)
residuals2(fit2, type="quantile", col="red", pch=20,col.lab="blue",
           col.axis="blue",col.main="black",family="mono",cex=0.8)
```

---

richness *Species richness*

Description

In these data the response is the species richness represented by a count of the number of plant species on plots that have different biomass and three different soil pH levels: low, mid, and high. See Crawley (2007, page 534).

Usage

```
data(richness)
```

Format

A data frame with 90 rows and 3 variables:

Biomass a numeric vector giving the value of the biomass in the plots.

pH a factor giving the soil pH level in the plots: "low", "mid", and "high".

Species a numeric vector giving the number of plant species in the plots.

References

Crawley M.J. (2007) *The R Book*. John Wiley & Sons, Chichester.

Examples

```
with(richness,{
  plot(Biomass, Species,
     col=apply(as.matrix(pH),1,function(x) switch(x,"low"="red","mid"="black","high"="blue")),
       pch=apply(as.matrix(pH),1,function(x) switch(x,"low"=15,"mid"=16,"high"=17)))
  legend(8.5, 42, legend=c("low","mid","high"), col=c("red","black","blue"),
         pch=c(15,16,17), bty="n", cex=0.95, title="Soil pH level")
})
```

rinse *Dental Clinical Trial*

Description

These data arose from a dental clinical study. In this trial, subjects were generally healthy adult male and female volunteers, ages 18–55, with pre-existing plaque but without advanced periodontal disease. Prior to entry, subjects were screened for a minimum of 20 sound, natural teeth and a minimum mean plaque index of 2.0. Subjects with gross oral pathology or on antibiotic, antibacterial, or anti-inflammatory therapy were excluded from the study. One hundred nine volunteers were randomized in a double-blinded way to one of two new mouth rinses (A and B) or to a control mouth rinse. Plaque was scored at baseline, at 3 months, and at 6 months by the Turesky modification of the Quigley-Hein index, a continuous measure. Four subjects had missing plaque scores. The main objective of the analysis is to measure the effectiveness of the three mouth rinses in inhibiting the development of dental plaque.

Usage

```
data(rinse)
```

Format

A data frame with 315 rows and 7 variables:

subject a character string giving the identifier of the volunteer.

gender a factor indicating the gender of the volunteer: "Female" and "Male".

age a numeric vector indicating the age of the volunteer.

rinse a factor indicating the type of rinse used by the volunteer: "Placebo", "A" and "B".

smoke a factor indicating if the volunteer smoke: "Yes" and "No".

time a numeric vector indicating the time (in months) since the treatment began.

score a numeric vector giving the subject's score of plaque.

References

Hadgu A. and Koch G. (1999) Application of generalized estimating equations to a dental randomized clinical trial. *Journal of Biopharmaceutical Statistics* 9, 161-178.

Examples

```
mod <- score ~ rinse + age + gender + smoke + time
fit <- glmgee(mod, family=Gamma(log), id=subject, corstr="AR-1", data=rinse)
summary(fit)
```

RJC *Rotnitzky–Jewell's Criterion for Generalized Estimating Equations*

Description

Computes the Rotnitzky–Jewell's criterion (RJC) for one or more objects of the class glmgee.

Usage

```
RJC(..., verbose = TRUE)
```

Arguments

`...`	one or several objects of the class glmgee which are obtained from the fit of generalized estimating equations.
`verbose`	an (optional) logical switch indicating if should the report of results be printed. By default, `verbose` is set to be TRUE.

Value

A `data.frame` with the values of the RJC for each glmgee object in the input.

References

Hin L-Y, Carey V.J., Wang Y-G (2007) Criteria for Working–Correlation–Structure Selection in GEE: Assessment via Simulation. *The American Statistician* 61, 360–364.

See Also

QIC, CIC, GHYC, AGPC, SGPC

Examples

```
## Example 1
mod <- size ~ poly(days,4) + treat
fit1 <- glmgee(mod, id=tree, family=Gamma("log"), data=spruces, corstr="Exchangeable")
fit2 <- update(fit1, corstr="AR-1")
fit3 <- update(fit1, corstr="Stationary-M-dependent(2)")
fit4 <- update(fit1, corstr="Independence")
RJC(fit1, fit2, fit3, fit4)

## Example 2
mod <- dep ~ visit + group
fit1 <- glmgee(mod, id=subj, family=gaussian, corstr="Exchangeable", data=depression)
fit2 <- update(fit1, corstr="AR-1")
fit3 <- update(fit1, corstr="Non-Stationary-M-dependent(2)")
fit4 <- update(fit1, corstr="Independence")
RJC(fit1, fit2, fit3, fit4)
```

```
## Example 3
mod <- depressd ~ visit + group
fit1 <- glmgee(mod, id=subj, family=binomial, corstr="Exchangeable", data=depression)
fit2 <- update(fit1, corstr="AR-1")
fit3 <- update(fit1, corstr="Stationary-M-dependent(2)")
fit4 <- update(fit1, corstr="Independence")
RJC(fit1, fit2, fit3, fit4)
```

ROCc *The Receiver Operating Characteristic (ROC) Curve*

Description

Computes the exact area under the ROC curve (AUROC), the Gini coefficient, and the Kolmogorov-Smirnov (KS) statistic for a binary classifier. Optionally, this function can plot the ROC curve, that is, the plot of the estimates of Sensitivity versus the estimates of 1-Specificity.

Usage

```
ROCc(object, plot.it = TRUE, verbose = TRUE, ...)
```

Arguments

`object`	a matrix with two columns: the first one is a numeric vector of 1's and 0's indicating whether each row is a "success" or a "failure"; the second one is a numeric vector of values indicating the probability (or propensity score) of each row to be a "success". Optionally, `object` can be an object of the class glm which is obtained from the fit of a generalized linear model where the distribution of the response variable is assumed to be binomial.
`plot.it`	an (optional) logical switch indicating if the plot of the ROC curve is required or just the data matrix in which it is based. By default, `plot.it` is set to be TRUE.
`verbose`	an (optional) logical switch indicating if should the report of results be printed. By default, `verbose` is set to be TRUE.
`...`	further arguments passed to or from other methods. For example, if `plot.it=TRUE` then `...` may to include graphical parameters as `col`, `pch`, `cex`, `main`, `sub`, `xlab`, `ylab`.

Value

A list which contains the following objects:

- `roc`: A matrix with the Cutoffs and the associated estimates of Sensitivity and Specificity.
- `auroc`: The exact area under the ROC curve.
- `gini`: The value of the Gini coefficient computed as 2(`auroc`-0.5).
- `ks`: The value of the Kolmogorov-Smirnov statistic computed as the maximum value of |1-Sensitivity-Specificity|.

References

Hanley J.A. and McNeil B.J. (1982) The Meaning and Use of the Area under a Receiver Operating Characteristic (ROC) Curve. *Radiology* 143, 29–36.

Examples

```
burn1000 <- aplore3::burn1000

## splitting the sample
## 70% for the training sample and 30% for the validation sample
burn1000 <- within(burn1000, sampleof <- "validation")
s <- sample(nrow(burn1000),nrow(burn1000)*0.7)
burn1000$sampleof[s] <- "training"

mod <- death ~ age + tbsa + inh_inj + age*inh_inj + tbsa*inh_inj
training <- subset(burn1000,sampleof=="training")
fit <- glm(mod, family=binomial("logit"), data=training)

## ROC curve for the training sample
ROCc(fit, col="red", col.lab="blue", col.axis="black",
     col.main="black", family="mono")

validation <- subset(burn1000, sampleof=="validation")
probs <- predict(fit, newdata=validation, type="response")
responses <- with(validation, ifelse(death=="Dead",1,0))

## ROC curve for the validation sample
ROCc(cbind(responses,probs), col="red", col.lab="blue",
     col.axis="black", col.main="black", family="mono")
```

SGPC *SGPC for Generalized Estimating Equations*

Description

Computes the Schwarz-type penalized Gaussian pseudo-likelihood criterion (SGPC) for one or more objects of the class glmgee.

Usage

```
SGPC(..., verbose = TRUE)
```

Arguments

`...`	one or several objects of the class glmgee which are obtained from the fit of generalized estimating equations.
`verbose`	an (optional) logical switch indicating if should the report of results be printed. By default, `verbose` is set to be TRUE.

Value

A `data.frame` with the values of the gaussian pseudo-likelihood, the number of parameters in the linear predictor plus the number of parameters in the correlation matrix, and the value of SGPC for each glmgee object in the input.

References

Carey V.J. and Wang Y.-G. (2011) Working covariance model selection for generalized estimating equations. *Statistics in Medicine* 30, 3117–3124.

Zhu X. and Zhu Z. (2013) Comparison of criteria to select working correlation matrix in generalized estimating equations. *Chinese Journal of Applied Probability and Statistics* 29, 515-530.

Fu L., Hao Y. and Wang Y.-G. (2018) Working correlation structure selection in generalized estimating equations. *Computational Statistics* 33, 983-996.

See Also

QIC, CIC, RJC, GHYC, AGPC

Examples

```
## Example 1
mod <- size ~ poly(days,4) + treat
fit1 <- glmgee(mod, id=tree, family=Gamma("log"), data=spruces, corstr="Exchangeable")
fit2 <- update(fit1, corstr="AR-1")
fit3 <- update(fit1, corstr="Stationary-M-dependent(2)")
fit4 <- update(fit1, corstr="Independence")
SGPC(fit1, fit2, fit3, fit4)

## Example 2
mod <- dep ~ visit + group
fit1 <- glmgee(mod, id=subj, family=gaussian, corstr="Exchangeable", data=depression)
fit2 <- update(fit1, corstr="AR-1")
fit3 <- update(fit1, corstr="Non-Stationary-M-dependent(2)")
fit4 <- update(fit1, corstr="Independence")
SGPC(fit1, fit2, fit3, fit4)

## Example 3
mod <- depressd ~ visit + group
fit1 <- glmgee(mod, id=subj, family=binomial, corstr="Exchangeable", data=depression)
fit2 <- update(fit1, corstr="AR-1")
fit3 <- update(fit1, corstr="Stationary-M-dependent(2)")
fit4 <- update(fit1, corstr="Independence")
SGPC(fit1, fit2, fit3, fit4)
```

skincancer *Skin cancer in women*

Description

The data describe the incidence of nonmelanoma skin cancer for women stratified by age in Minneapolis (St. Paul) and Dallas (Fort Worth). See Kleinbaum et al. (2013, page 751).

Usage

```
data(skincancer)
```

Format

A data frame with 16 rows and 4 variables:

cases a numeric vector giving the nonmelanoma skin cancer counts.

city a factor giving the city to which correspond the skin cancer counts: "St.Paul" and "Ft.Worth".

age a factor giving the age range to which correspond the skin cancer counts: "15-24", "25-34", "35-44", "45-54", "55-64", "65-74", "75-84" and "85+".

population a numeric vector giving the population of women.

References

Kleinbaum D., Kupper L., Nizam A. and Rosenberg E.S. (2013) *Applied Regression Analysis and other Multivariable Methods, Fifth Edition*, Cengage Learning, Boston.

Examples

```
barplot(1000*cases/population ~ city + age, beside=TRUE, col=c("yellow","blue"), data=skincancer)
legend(1, 11, legend=c("St.Paul","Ft.Worth"), title="City",
       fill=c("yellow","blue"), bty="n", cex=0.9)
```

spruces *Effect of ozone-enriched atmosphere on growth of sitka spruces*

Description

The main objective of the analysis of these data is to assess the effect of the ozone pollution on the tree growth. As ozone pollution is common in urban areas, the impact of increased ozone concentrations on tree growth is of considerable interest. The response variable is tree size, where size is conventionally measured by the product of tree height and stem diameter squared. In a first group, a total of 54 trees were grown under an ozone-enriched atmosphere, that is, ozone exposure at 70 parts per billion, whereas in a second group, 25 were grown under a normal atmosphere. The size of each tree was observed 13 times across the time, that is, 152, 174, 201, 227, 258, 469, 496, 528, 556, 579, 613, 639 and 674 days since the beginning of the experiment. Hence, the objective is to compare the growth patterns of the trees under the two conditions. See Diggle et al. (2002, page 4).

Usage

```
data(spruces)
```

Format

A data frame with 1027 rows and 4 variables:

tree a factor giving an unique identifier for each tree.

days a numeric vector giving the number of days since the beginning of the experiment.

size a numeric vector giving an estimate of the volume of the tree trunk.

treat a factor giving the treatment received for each tree: "normal" and "ozone-enriched".

References

Diggle P.J., Heagarty P., Liang K.-Y. and Zeger S.L. (2002) *Analysis of Longitudinal Data*. Oxford University Press, Oxford.

Crainiceanu C.M., Ruppert D. and Wand M.P. (2005). Bayesian Analysis for Penalized Spline Regression Using WinBUGS. *Journal of Statistical Software* 14(14).

Examples

```
boxplot(size ~ days, data=subset(spruces,treat=="normal"), at=c(1:13)-0.2,
        col="yellow", boxwex=0.3, outline=FALSE, xaxt="n", xlim=c(0.9,13.1))
boxplot(size ~ days, data=subset(spruces,treat=="ozone-enriched"), add=TRUE,
        at=c(1:13)+0.2, col="blue", boxwex=0.3, outline=FALSE, xaxt="n")
axis(1, at=1:13, labels=unique(spruces$days))
axis(2, at=seq(0,1500,250), labels=seq(0,1500,250))
legend(0.5, 1500, legend=c("normal","ozone-enriched"), title="Atmosphere",
       fill=c("yellow","blue"), bty="n")
```

---

stepCriterion *Variable selection in regression models from a chosen criterion*

Description

Generic function for selecting variables from a fitted regression model using a chosen criterion.

Usage

```
stepCriterion(model, ...)
```

Arguments

model	a fitted model object.
...	further arguments passed to or from other methods.

Value

A list which includes the descriptions of the linear predictors of the initial and final models as well as the criterion used to compare the candidate models.

`stepCriterion.glm` *Variable Selection in Generalized Linear Models*

Description

Performs variable selection in generalized linear models using hybrid versions of forward stepwise and backward stepwise.

Usage

```
## S3 method for class 'glm'
stepCriterion(
  model,
  criterion = c("bic", "aic", "adjr2", "p-value", "qicu"),
  test = c("wald", "lr", "score", "gradient"),
  direction = c("forward", "backward"),
  levels = c(0.05, 0.05),
  trace = TRUE,
  scope,
  ...
)
```

Arguments

`model`	an object of the class glm which is obtained from the fit of a generalized linear model.
`criterion`	an (optional) character string indicating the criterion which should be used to compare the candidate models. The available options are: AIC ("aic"), BIC ("bic"), adjusted deviance-based R-squared ("adjr2"), and *p*-value of the `test` test ("p-value"). By default, `criterion` is set to be "bic".
`test`	an (optional) character string indicating the statistical test which should be used to compare nested models. The available options are: Wald ("wald"), Rao's score ("score"), likelihood-ratio ("lr") and gradient ("gradient") tests. By default, `test` is set to be "wald".
`direction`	an (optional) character string indicating the type of procedure which should be used. The available options are: hybrid backward stepwise ("backward") and hybrid forward stepwise ("forward"). By default, `direction` is set to be "forward".
`levels`	an (optional) two-dimensional vector of values in the interval $(0, 1)$ indicating the levels at which the variables should in and out from the model. This is only appropiate if `criterion`="p-value". By default, `levels` is set to be `c(0.05,0.05)`.

trace	an (optional) logical switch indicating if should the stepwise reports be printed. By default, trace is set to be TRUE.
scope	an (optional) list, containing components lower and upper, both formula-type objects, indicating the range of models which should be examined in the stepwise search. By default, lower is a model with no predictors and upper is the linear predictor of the model in model.
...	further arguments passed to or from other methods. For example, k, that is, the magnitude of the penalty in the AIC/QICu, which by default is set to be 2.

Value

A list which contains the following objects:

- initial: a character string indicating the linear predictor of the "initial model".
- direction: a character string indicating the type of procedure which was used.
- criterion: a character string indicating the criterion used to compare the candidate models.
- final: a character string indicating the linear predictor of the "final model".

References

James G., Witten D., Hastie T. and Tibshirani R. (2013, page 210) An Introduction to Statistical Learning with Applications in R, Springer, New York.

See Also

stepCriterion.lm, stepCriterion.overglm, stepCriterion.glmgee

Examples

```
## Example 1
Auto <- ISLR::Auto
Auto2 <- within(Auto, origin <- factor(origin))
mod <- mpg ~ cylinders + displacement + acceleration + origin + horsepower*weight
fit1 <- glm(mod, family=inverse.gaussian("log"), data=Auto2)
stepCriterion(fit1, direction="forward", criterion="p-value", test="lr")
stepCriterion(fit1, direction="backward", criterion="bic")

## Example 2
burn1000 <- aplore3::burn1000
burn1000 <- within(burn1000, death <- factor(death, levels=c("Dead","Alive")))
upper <- ~ age + gender + race + tbsa + inh_inj + flame + age*inh_inj + tbsa*inh_inj
lower <- ~ 1
fit2 <- glm(death ~ age + gender + race + tbsa + inh_inj, family=binomial("logit"), data=burn1000)
stepCriterion(fit2, direction="backward", criterion="bic", scope=list(lower=lower,upper=upper))
stepCriterion(fit2, direction="forward", criterion="p-value", test="score")

## Example 3
upper <- cases ~ city + age + city*age
fit3 <- glm(upper, family=poisson("log"), offset=log(population), data=skincancer)
stepCriterion(fit3, direction="backward", criterion="aic", scope=list(lower=~ 1,upper=upper))
stepCriterion(fit3, direction="forward", criterion="p-value", test="lr")
```

`stepCriterion.glmgee` *Variable selection in Generalized Estimating Equation*

Description

Performs variable selection in generalized estimating equations using hybrid versions of forward stepwise and backward stepwise.

Usage

```
## S3 method for class 'glmgee'
stepCriterion(
  model,
  criterion = c("p-value", "qic", "qicu", "adjr2", "agpc", "sgpc"),
  test = c("wald", "score"),
  direction = c("forward", "backward"),
  levels = c(0.05, 0.05),
  trace = TRUE,
  scope,
  digits = 5,
  varest = c("robust", "df-adjusted", "model", "bias-corrected"),
  ...
)
```

Arguments

`model`	an object of the class glmgee which is obtained from the fit of a generalized estimating equation.
`criterion`	an (optional) character string indicating the criterion which should be used to compare the candidate models. The available options are: QIC ("qic"), QICu ("qicu"), adjusted deviance-based R-squared ("adjr2"), Akaike-type penalized gaussian pseudo-likelihood criterion ("agpc"), Schwarz-type penalized gaussian pseudo-likelihood criterion ("sgpc") and *p*-value of the `test` test ("p-value"). By default, `criterion` is set to be "p-value".
`test`	an (optional) character string indicating the statistical test which should be used to compare nested models. The available options are: Wald ("wald") and generalized score ("score") tests. By default, `test` is set to be "wald".
`direction`	an (optional) character string indicating the type of procedure which should be used. The available options are: hybrid backward stepwise ("backward") and hybrid forward stepwise ("forward"). By default, `direction` is set to be "forward".
`levels`	an (optional) two-dimensional vector of values in the interval $(0, 1)$ indicating the levels at which the variables should in and out from the model. This is only appropiate if `criterion`="p-value". By default, `levels` is set to be `c(0.05,0.05)`.

`trace`	an (optional) logical switch indicating if should the stepwise reports be printed. By default, `trace` is set to be TRUE.
`scope`	an (optional) list, containing components `lower` and `upper`, both formula-type objects, indicating the range of models which should be examined in the stepwise search. By default, `lower` is a model with no predictors and `upper` is the linear predictor of the model in `model`.
`digits`	an (optional) integer indicating the number of digits which should be used to print the most of the criteria to compare the candidate models. By default, `digits` is set to be 5.
`varest`	an (optional) character string indicating the type of estimator which should be used to the variance-covariance matrix of the interest parameters in the Wald-type test. The available options are: robust sandwich-type estimator ("robust"), degrees-of-freedom-adjusted estimator ("df-adjusted"), bias-corrected estimator ("bias-corrected"), and the model-based or naive estimator ("model"). By default, `varest` is set to be "robust".
`...`	further arguments passed to or from other methods. For example, k, that is, the magnitude of the penalty in the AGPC, which by default is set to be 2.

Value

A list which contains the following objects:

- `initial`: a character string indicating the linear predictor of the "initial model".
- `direction`: a character string indicating the type of procedure which was used.
- `criterion`: a character string indicating the criterion used to compare the candidate models.
- `final`: a character string indicating the linear predictor of the "final model".

References

James G., Witten D., Hastie T. and Tibshirani R. (2013, page 210) An Introduction to Statistical Learning with Applications in R. Springer, New York.

Jianwen X., Jiamao Z. and Liya F. (2019) Variable selection in generalized estimating equations via empirical likelihood and Gaussian pseudo-likelihood. Communications in Statistics - Simulation and Computation, 48:4.

See Also

stepCriterion.lm, stepCriterion.glm, stepCriterion.overglm

Examples

```
## Example 1
mod <- size ~ poly(days,4)*treat
fit1 <- glmgee(mod, id=tree, family=Gamma("log"), data=spruces, corstr="AR-1")
stepCriterion(fit1, criterion="p-value", direction="forward", scope=list(lower=~1,upper=mod))

## Example 2
mod <- depressd ~ visit*group
```

```
fit2 <- glmgee(mod, id=subj, family=binomial("logit"), corstr="AR-1", data=depression)
stepCriterion(fit2, criterion="adjr2", direction="forward", scope=list(lower=~1,upper=mod))
```

`stepCriterion.lm` *Variable Selection in Normal Linear Models*

Description

Performs variable selection in normal linear models using hybrid versions of forward stepwise and backward stepwise.

Usage

```
## S3 method for class 'lm'
stepCriterion(
  model,
  criterion = c("bic", "aic", "adjr2", "prdr2", "cp", "p-value"),
  direction = c("forward", "backward"),
  levels = c(0.05, 0.05),
  trace = TRUE,
  scope,
  ...
)
```

Arguments

`model`	an object of the class lm which is obtained from the fit of a normal linear model.
`criterion`	an (optional) character string indicating the criterion which should be used to compare the candidate models. The available options are: AIC ("aic"), BIC ("bic"), adjusted R-squared ("adjr2"), predicted R-squared ("prdr2"), Mallows' CP ("cp") and *p*-value of the F test ("p-value"). By default, `criterion` is set to be "bic".
`direction`	an (optional) character string indicating the type of procedure which should be used. The available options are: hybrid backward stepwise ("backward") and hybrid forward stepwise ("forward"). By default, `direction` is set to be "forward".
`levels`	an (optional) two-dimensional vector of values in the interval $(0, 1)$ indicating the levels at which the variables should in and out from the model. This is only appropiate if `criterion`="p-value". By default, `levels` is set to be `c(0.05,0.05)`.
`trace`	an (optional) logical switch indicating if should the stepwise reports be printed. By default, `trace` is set to be TRUE.
`scope`	an (optional) list, containing components `lower` and `upper`, both formula-type objects, indicating the range of models which should be examined in the stepwise search. By default, `lower` is a model with no predictors and `upper` is the linear predictor of the model in `model`.
`...`	further arguments passed to or from other methods. For example, `k`, that is, the magnitude of the penalty in the AIC/QICu, which by default is set to be 2.

Value

A list which contains the following objects:

- `initial`: a character string indicating the linear predictor of the "initial model".
- `direction`: a character string indicating the type of procedure which was used.
- `criterion`: a character string indicating the criterion used to compare the candidate models.
- `final`: a character string indicating the linear predictor of the "final model".

References

James G., Witten D., Hastie T. and Tibshirani R. (2013, page 210) An Introduction to Statistical Learning with Applications in R, Springer, New York.

See Also

stepCriterion.glm, stepCriterion.overglm, stepCriterion.glmgee

stepCriterion.glm, stepCriterion.overglm, stepCriterion.glmgee

Examples

```
## Example 1
fit1 <- lm(log(Ozone) ~ Solar.R + Temp + Wind, data=airquality)
scope=list(lower=~1, upper=~Solar.R*Temp*Wind)
stepCriterion(fit1, direction="forward", criterion="adjr2", scope=scope)
stepCriterion(fit1, direction="forward", criterion="bic", scope=scope)
stepCriterion(fit1, direction="forward", criterion="p-value", scope=scope)

## Example 2
fit2 <- lm(mpg ~ log(hp) + log(wt) + qsec, data=mtcars)
scope=list(lower=~1, upper=~log(hp)*log(wt)*qsec)
stepCriterion(fit2, direction="backward", criterion="bic", scope=scope)
stepCriterion(fit2, direction="forward", criterion="cp", scope=scope)
stepCriterion(fit2, direction="backward", criterion="prdr2", scope=scope)

## Example 3
Credit <- ISLR::Credit
fit3 <- lm(Balance ~ Cards + Age + Rating + Income + Student + Limit, data=Credit)
stepCriterion(fit3, direction="forward", criterion="prdr2")
stepCriterion(fit3, direction="forward", criterion="cp")
stepCriterion(fit3, direction="forward", criterion="p-value")
```

`stepCriterion.overglm` *Variable selection in Negative Binomial and Beta-Binomial Regression Models*

Description

Performs variable selection in negative binomial and beta-binomial regression models using hybrid versions of forward stepwise and backward stepwise.

Usage

```
## S3 method for class 'overglm'
stepCriterion(
  model,
  criterion = c("bic", "aic", "p-value"),
  test = c("wald", "score", "lr", "gradient"),
  direction = c("forward", "backward"),
  levels = c(0.05, 0.05),
  trace = TRUE,
  scope,
  ...
)
```

Arguments

`model`	an object of the class overglm which is obtained from the fit of a negative binomial or beta-binomial regression model.
`criterion`	an (optional) character string indicating the criterion which should be used to compare the candidate models. The available options are: AIC ("aic"), BIC ("bic"), and *p*-value of the `test` test ("p-value"). By default, `criterion` is set to be "bic".
`test`	an (optional) character string indicating the statistical test which should be used to compare nested models. The available options are: Wald ("wald"), Rao's score ("score"), likelihood-ratio ("lr") and gradient ("gradient") tests. By default, `test` is set to be "wald".
`direction`	an (optional) character string indicating the type of procedure which should be used. The available options are: hybrid backward stepwise ("backward") and hybrid forward stepwise ("forward"). By default, `direction` is set to be "forward".
`levels`	an (optional) two-dimensional vector of values in the interval $(0, 1)$ indicating the levels at which the variables should in and out from the model. This is only appropiate if `criterion`="p-value". By default, `levels` is set to be `c(0.05,0.05)`.
`trace`	an (optional) logical switch indicating if should the stepwise reports be printed. By default, `trace` is set to be TRUE.

scope	an (optional) list containing components lower and upper, both formula-type objects, indicating the range of models which should be examined in the stepwise search. By default, lower is a model with no predictors and upper is the linear predictor of the model in model.
...	further arguments passed to or from other methods. For example, k, that is, the magnitude of the penalty in the AIC, which by default is set to be 2.

Value

A list which contains the following objects:

- initial: a character string indicating the linear predictor of the "initial model".
- direction: a character string indicating the type of procedure which was used.
- criterion: a character string indicating the criterion used to compare the candidate models.
- final: a character string indicating the linear predictor of the "final model".

References

James G., Witten D., Hastie T. and Tibshirani R. (2013, page 210) An Introduction to Statistical Learning with Applications in R, Springer, New York.

See Also

stepCriterion.lm, stepCriterion.glm, stepCriterion.glmgee

Examples

```
fit <- overglm(infections ~ frequency + location + age + gender, family="nb1(log)", data=swimmers)
stepCriterion(fit,direction="forward",criterion="p-value",test="lr")
stepCriterion(fit,direction="backward",criterion="p-value",test="lr")
stepCriterion(fit,direction="forward",criterion="bic")
stepCriterion(fit,direction="backward",criterion="bic")
```

---

swimmers *Self diagnozed ear infections in swimmers*

Description

The data come from the Pilot Surf/Health Study of NSW Water Board performed in 1990 on 287 recruits. The objective of the study was to determine, in particular, whether beach swimmers run a greater risk of contracting ear infections than non-beach swimmers. See Hand et al. (1994. page 266).

Usage

```
data(swimmers)
```

Format

A data frame with 287 rows and 5 variables:

frequency a factor giving the recruit's perception of whether he or she is a frequent swimmer: "frequent" and "occasional".

location a factor giving the recruit's usually chosen swimming location: "beach" and "non-beach".

age a factor giving the recruit's age range: "15-19", "20-24" and "25-29".

gender a factor giving the recruit's gender: "male" and "female".

infections a numeric vector giving the number of self diagnozed ear infections that were reported by the recruit.

References

Hand D.J., Daly F., Lunn A.D., McConway K.J. and Ostrowsky E. (1994) *A Handbook of Small Data Sets*, Chapman and Hall, London.

Vanegas L.H. and Rondon L.M. (2020) A data transformation to deal with constant under/over-dispersion in binomial and poisson regression models. *Journal of Statistical Computation and Simulation* 90, 1811-1833.

Examples

```
boxplot(infections ~ frequency, data=subset(swimmers,location=="non-beach"),
        at=c(1:2)-0.2, col="yellow", boxwex=0.25, outline=FALSE, xaxt="n")
boxplot(infections ~ frequency, data=subset(swimmers,location=="beach"), add=TRUE,
        at=c(1:2)+0.2, col="blue", boxwex=0.25, outline=FALSE, xaxt="n")
axis(1, at=1:2, labels=levels(swimmers$frequency))
legend(0.3, 6, legend=c("non-beach","beach"), title="Location",
       fill=c("yellow","blue"), bty="n", cex=0.9)
```

---

vcov.glmgee *Estimate of the variance-covariance matrix in GEEs*

Description

Computes the type-type estimate of the variance-covariance matrix from an object of the class glmgee.

Usage

```
## S3 method for class 'glmgee'
vcov(
  object,
  ...,
  type = c("robust", "df-adjusted", "model", "bias-corrected", "jackknife")
)
```

Arguments

object	An object of the class glmgee which is obtained from the fit of a generalized estimating equation.
...	further arguments passed to or from other methods.
type	an (optional) character string indicating the type of estimator which should be used. The available options are: robust sandwich-type estimator ("robust"), degrees-of-freedom-adjusted estimator ("df-adjusted"), bias-corrected estimator ("bias-corrected"), and the model-based or naive estimator ("model"). By default, type is set to be "robust".

Value

A matrix with the type-type estimate of the variance-covariance matrix.

References

Mancl, L.A. and DeRouen T.A. (2001) A Covariance Estimator for GEE with Improved Small-Sample Properties. *Biometrics* 57, 126-134.

Examples

```
## Example 1
mod <- size ~ poly(days,4) + treat
fit1 <- glmgee(mod, id=tree, family=Gamma("log"), data=spruces, corstr="Exchangeable")
vcov(fit1)
vcov(fit1,type="bias-corrected")

## Example 2
mod <- dep ~ visit + group
fit2 <- glmgee(mod, id=subj, family=gaussian, corstr="AR-1", data=depression)
vcov(fit2)
vcov(fit2,type="bias-corrected")

## Example 3
mod <- depressd ~ visit + group
fit3 <- glmgee(mod, id=subj, family=binomial, corstr="Stationary-M-dependent(3)", data=depression)
vcov(fit3)
vcov(fit3,type="bias-corrected")
```

---

vdtest *Test for Varying Dispersion Parameter*

Description

Generic function for testing for varying dispersion parameter from a fitted model.

Usage

```
vdtest(model, ...)
```

Arguments

model	a fitted model object.
...	further arguments passed to or from other methods.

Value

A list which includes the main attributes of the test as, for example, value of the statistic and p-value.

---

vdtest.glm *Test for Varying Dispersion Parameter in Generalized Linear Models*

Description

Performs Rao's score test for varying dispersion parameter in weighted and unweighted generalized linear models in which the response distribution is assumed to be gaussian, Gamma or inverse gaussian.

Usage

```
## S3 method for class 'glm'
vdtest(model, varformula, verbose = TRUE, ...)
```

Arguments

model	an object of the class glm which is obtained from the fit of a weighted or unweighted generalized linear model in which the response distribution is assumed to be gaussian, Gamma or inverse gaussian.
varformula	an (optional) formula expression of the form ~ predictors describing only the potential explanatory variables for the dispersion. By default, the same explanatory variables are taken as in the model for the mean.
verbose	an (optional) logical switch indicating if should the report of results be printed. By default, verbose is set to be TRUE.
...	further arguments passed to or from other methods.

Details

The aim of this test is to assess the assumption of constant dispersion parameter in generalized linear models. If the object model corresponds to an unweighted generalized linear model then this test assess assumptions of constant variance and constant coefficient of variation on models in which the response distribution is assumed to be gaussian and Gamma, respectively.

Value

A list which includes the three main attributes of the test for varying dispersion parameter: statistic ("statistic"), degrees of freedom ("df") and *p*-value ("p.value").

References

Wei BC., Shi JQ., Fung WK. and Hu YQ. (1998) Testing for Varying Dispersion in Exponential Family Nonlinear Models. *Annals of the Institute of Statistical Mathematics* 50, 277–294.

See Also

vdtest.lm

Examples

```
## Example 1
Auto <- ISLR::Auto
fit1 <- glm(mpg ~ weight*horsepower, family=inverse.gaussian("log"), data=Auto)
vdtest(fit1)

## Example 2
fit2 <- glm(rtime ~ log(distance) + log(cclimb), family=Gamma("log"), data=races)
vdtest(fit2)
```

`vdtest.lm` *Test for Varying Dispersion Parameter in Normal Linear Models*

Description

Performs Rao's score test for varying dispersion parameter in weighted and unweighted normal linear models.

Usage

```
## S3 method for class 'lm'
vdtest(model, varformula, verbose = TRUE, ...)
```

Arguments

`model`	an object of the class lm which is obtained from the fit of a weighted or unweighted normal linear model.
`varformula`	an (optional) `formula` expression of the form `~ predictors` describing only the potential explanatory variables for the dispersion. By default, the same explanatory variables are taken as in the model for the mean.
`verbose`	an (optional) logical switch indicating if should the report of results be printed. By default, `verbose` is set to be TRUE.
`...`	further arguments passed to or from other methods.

Details

If the object `model` corresponds to an unweighted normal linear model then this test assess the assumption of constant variance, which coincides with the (non-studentized) Breusch-Pagan test against heteroskedasticity.

Value

A list which includes the three main attributes of the test for varying dispersion parameter: statistic ("statistic"), degrees of freedom ("df") and *p*-value ("p.value").

References

Breusch T.S. and Pagan A.R. (1979) A simple test for heteroscedasticity and random coefficient variation. *Econometrica* 47, 1287–1294.

Cook R.D. and Weisberg S. (1983) Diagnostics for heteroscedasticity in regression. *Biometrika* 70, 1–10.

See Also

vdtest.glm

Examples

```
## Example 1
fit1 <- lm(mpg ~ log(hp) + log(wt), data=mtcars)
vdtest(fit1)

## Example 2
fit2 <- lm(Species ~ Biomass + pH, data=richness)
vdtest(fit2)

fit2a <- lm(Species ~ Biomass + pH, data=richness, subset=-c(1,3,18,20))
vdtest(fit2a)

## Example 3
whiteside <- MASS::whiteside
fit3 <- lm(Gas ~ Temp + Insul + Temp*Insul, data=whiteside)
vdtest(fit3)

fit3a <- lm(Gas ~ Temp + Insul + Temp*Insul, data=whiteside, subset=-c(8,9,36,46,55))
vdtest(fit3a)
```

Index