Package ‘aod’

February 19, 2015

Version 1.3
Date 2012-04-10
Title Analysis of Overdispersed Data
Author Matthieu Lesnoff <matthieu.lesnoff@cirad.fr> and Renaud Lancelot <renaud.lancelot@cirad.fr>
Maintainer Renaud Lancelot <renaud.lancelot@cirad.fr>
Depends R (>= 2.0.0), methods, stats
Suggests MASS, boot, lme4
Description This package provides a set of functions to analyse overdispersed counts or proportions. Most of the methods are already available elsewhere but are scattered in different packages. The proposed functions should be considered as complements to more sophisticated methods such as generalized estimating equations (GEE) or generalized linear mixed effect models (GLMM).
License GPL (>= 2)
URL http://cran.r-project.org/package=aod
LazyData yes
Repository CRAN
Date/Publication 2012-04-10 07:30:34
NeedsCompilation no

R topics documented:

aic-class ................................................................. 2
AIC-methods ............................................................. 3
anova-methods ......................................................... 4
antibio ................................................................. 5
aod-pkg ................................................................. 6
betabin ................................................................. 7
coef-methods ......................................................... 10
Index

aic-class  

Representation of Objects of Formal Class "aic"

Description

Representation of the output of function AIC.
Slots

\begin{itemize}
\item \texttt{istats} A data frame with 3 columns describing the models indicated by the row names:
\begin{itemize}
\item \texttt{df} number of parameters in the model,
\item \texttt{AIC} Akaike information criterion for the model (see \texttt{AIC}),
\item \texttt{AICc} small-sample corrected Akaike information criterion for the model (see \texttt{AIC}).
\end{itemize}
\end{itemize}

Methods

\begin{itemize}
\item \texttt{summary} signature\(\text{object = "aic"}\)
\item \texttt{show} signature\(\text{object = "aic"}\)
\end{itemize}

\begin{tabular}{l}
\hline
\textbf{AIC-methods} & \textit{Akaike Information Criteria} \\
\hline
\end{tabular}

\section*{Description}

Extracts the Akaike information criterion (AIC) and the corrected AIC (AICc) from fitted models of formal class “glimML” and possibly computes derived statistics.

\section*{Usage}

\begin{verbatim}
## S4 method for signature 'glimML'
AIC(object, ..., k = 2)
\end{verbatim}

\section*{Arguments}

\begin{itemize}
\item \texttt{object} fitted model of formal class “glimML” (functions \texttt{betabin} or \texttt{negbin}).
\item \texttt{...} optional list of fitted models separated by commas.
\item \texttt{k} numeric scalar, with a default value set to 2, thus providing the regular AIC.
\end{itemize}

\section*{Details}

\[ AIC = -2 \times \text{log-likelihood} + 2 \times n_{\text{par}}, \]
where \(n_{\text{par}}\) represents the number of parameters in the fitted model.

\[ AICc = AIC + 2 \times n_{\text{par}} \times (n_{\text{par}} + 1)/(n_{\text{obs}} - n_{\text{par}} + 1), \]
where \(n_{\text{obs}}\) is the number of observations used to compute the log-likelihood. It should be used when the number of fitted parameters is large compared to sample size, i.e., when \(n_{\text{obs}}/n_{\text{par}} < 40\) (Hurvich and Tsai, 1995).

\section*{Methods}

\begin{itemize}
\item \texttt{glimML} Extracts the AIC and AICc from models of formal class “glimML”, fitted by functions \texttt{betabin} and \texttt{negbin}.
\end{itemize}
References


See Also

Examples in `betabin` and see `AIC` in package `stats`.

---

**Description**

Performs likelihood-ratio tests on nested models. Currently, one method was implemented for beta-binomial models (`betabin`) or negative-binomial models (`negbin`).

**Usage**

```r
## S4 method for signature 'glimML'
anova(object, ...)```

**Arguments**

- `object` Fitted model of class “glimML”.
- `...` Further models to be tested or arguments passed to the `print` function.

**Details**

The `anova` method for models of formal class “glimML” needs at least 2 nested models of the same type (either beta-binomial or negative-binomial models: they cannot be mixed). The quantity of interest is the deviance difference between the compared models: it is a log-likelihood ratio statistic. Under the null hypothesis that 2 nested models fit the data equally well, the deviance difference has an approximate $\chi^2$ distribution with degrees of freedom = the difference in the number of parameters between the compared models (McCullagh and Nelder, 1989).

**Value**

An object of formal class “anova.glimML” with 3 slots:

- `models` A vector of character strings with each component giving the name of the models and the formulas for the fixed and random effects.
**Description**

Hypothetical drug trial to compare the effect of four antibiotics against Shipping fever in calves (Shoukri and Pause, 1999, Table 3.11).
Usage

data(antibio)

Format

A data frame with 24 observations on the following 3 variables.

treatment A factor with levels 1, 2, 3 and 4

n A numeric vector: the number of treated animals within a two-week period.

y A numeric vector: the number of deaths at the end of the two weeks.

References


Description

This package provides a set of functions to analyse overdispersed counts or proportions. Most of the methods are already available elsewhere but are scattered in different packages. The proposed functions should be considered as complements to more sophisticated methods such as generalized estimating equations (GEE) or generalized linear mixed effect models (GLMM).

Details

<table>
<thead>
<tr>
<th>Package:</th>
<th>aod</th>
</tr>
</thead>
<tbody>
<tr>
<td>Version:</td>
<td>1.1-32</td>
</tr>
<tr>
<td>Date:</td>
<td>2010-04-02</td>
</tr>
<tr>
<td>Depends:</td>
<td>R (&gt;= 2.0.0), methods, stats</td>
</tr>
<tr>
<td>Suggests:</td>
<td>MASS, nlme, boot</td>
</tr>
<tr>
<td>License:</td>
<td>GPL version 2 or newer</td>
</tr>
<tr>
<td>URL:</td>
<td><a href="http://cran.r-project.org/package=aod">http://cran.r-project.org/package=aod</a></td>
</tr>
<tr>
<td>LazyLoad:</td>
<td>yes</td>
</tr>
<tr>
<td>LazyData:</td>
<td>yes</td>
</tr>
</tbody>
</table>

Index:

- AIC-methods
- aic-class
- anova-method
- antibio
- betabin

Akaike Information Criteria
Representation of Objects of Formal Class "aic"
Likelihood-Ratio Tests for Nested ML Models
Antibiotics against Shipping Fever in Calves
Beta-Binomial Model for Proportions
**Author(s)**

Matthieu Lesnoff <matthieu.lesnoff@cirad.fr> and Renaud Lancelot <renaud.lancelot@cirad.fr>
Maintainer: Renaud Lancelot

---

**betabin**

_Beta-Binomial Model for Proportions_
Description
Fits a beta-binomial generalized linear model accounting for overdispersion in clustered binomial data \((n, y)\).

Usage
```r
betabin(formula, random, data, link = c("logit", "cloglog"), phi.ini = NULL,
    warnings = FALSE, na.action = na.omit, fixpar = list(),
    hessian = TRUE, control = list(maxit = 2000), ...)
```

Arguments
- `formula`: A formula for the fixed effects b. The left-hand side of the formula must be of the form `cbind(y, n - y)` where the modelled probability is \(y/n\).
- `random`: A right-hand formula for the overdispersion parameter(s) \(\phi\).
- `link`: The link function for the mean \(p\): “logit” or “cloglog”.
- `data`: A data frame containing the response (\(n\) and \(y\)) and explanatory variable(s).
- `phi.ini`: Initial values for the overdispersion parameter(s) \(\phi\). Default to 0.1.
- `warnings`: Logical to control the printing of warnings occurring during log-likelihood maximization. Default to FALSE (no printing).
- `na.action`: A function name: which action should be taken in the case of missing value(s).
- `fixpar`: A list with 2 components (scalars or vectors) of the same size, indicating which parameters are fixed (i.e., not optimized) in the global parameter vector \((b, \phi)\) and the corresponding fixed values. For example, `fixpar = list(c(4, 5), c(0, 0))` means that 4th and 5th parameters of the model are set to 0.
- `hessian`: A logical. When set to FALSE, the hessian and the variances-covariances matrices of the parameters are not computed.
- `control`: A list to control the optimization parameters. See `optim`. By default, set the maximum number of iterations to 2000.
- `...`: Further arguments passed to `optim`.

Details
For a given cluster \((n, y)\), the model is:
\[
y^\dagger \lambda \sim \text{Binomial}(n, \lambda)
\]
with \(\lambda\) following a Beta distribution \(\text{Beta}(a_1, a_2)\).
If \(B\) denotes the beta function, then:
\[
P(\lambda) = \frac{\lambda^{a_1 - 1} * (1 - \lambda)^{a_2 - 1}}{B(a_1, a_2)}
\]
\[
E[\lambda] = \frac{a_1}{a_1 + a_2}
\]
The marginal beta-binomial distribution is:
\[ P(y) = \frac{C(n, \tilde{y}) \cdot B(a_1 + y, a_2 + n - y)}{B(a_1, \tilde{a_2})} \]

The function uses the parameterization \( p = \frac{a_1}{a_1 + a_2} = h(Xb) = h(\eta) \) and \( \phi = \frac{1}{a_1 + a_2 + 1} \), where \( h \) is the inverse of the link function (logit or complementary log-log), \( X \) is a design-matrix, \( b \) is a vector of fixed effects, \( \eta = Xb \) is the linear predictor and \( \phi \) is the overdispersion parameter (i.e., the intracluster correlation coefficient, which is here restricted to be positive).

The marginal mean and variance are:
\[ E[y] = n \cdot p \]
\[ Var[y] = n \cdot p \cdot (1 - p) \cdot [1 + (n - 1) \cdot \phi] \]

The parameters \( b \) and \( \phi \) are estimated by maximizing the log-likelihood of the marginal model (using the function \texttt{optim})..

### Value
An object of formal class “glimML”: see \texttt{glimML-class} for details.

### Author(s)
Matthieu Lesnoff \texttt{<matthieu.lesnoff@cirad.fr>}, Renaud Lancelot \texttt{<renaud.lancelot@cirad.fr>}

### References

### See Also
\texttt{glimML-class}, \texttt{glm} and \texttt{optim}

### Examples
```r
data(orbit2)
fm1 <- betabin(cbind(y, n - y) ~ seed, ~ 1, data = orbit2)
fm2 <- betabin(cbind(y, n - y) ~ seed + root, ~ 1, data = orbit2)
fm3 <- betabin(cbind(y, n - y) ~ seed * root, ~ 1, data = orbit2)
# show the model
fm1; fm2; fm3
# AIC
AIC(fm1, fm2, fm3)
```
Methods for Function "coef" in Package "aod"

Description

Extract the fixed-effect coefficients from fitted objects.

Methods

ANY  Generic function: see coef.

glimML Extract the estimated fixed-effect coefficients from objects of formal class "glimML". Presently, these objects are generated by functions betabin and negbin.

glimQL Extract the estimated fixed-effect coefficients from objects of formal class "glimQL". Presently, these objects are generated by functions quasibin and quasipois.
Description

Number of prostate cancer deaths and midperiod population for nonwhites in the USA by age and period. The cohort index $k$ is related to age and period indices ($i$ and $j$, respectively) by $k = j + I - i$, where $I = \max(i)$ (Holford, 1983, Table 2).

Usage

data(cohorts)

Format

A data frame with 49 observations on the following 4 variables.

- **period**: A factor with levels 1935-, 1940-, ..., 1965-.
- **age**: A factor with levels 50-, 55-, ..., 80-.
- **y**: Numeric: the number of prostate cancer deaths.
- **n**: Numeric: the midperiod population size.

References


deviance-methods

Methods for Function "deviance" in Package "aod"

Description

Extracts the deviance fitted models.

Methods

- **ANY**: Generic function: see deviance.
- **glimML**: Extracts the deviance from models fitted by betabin or negbin.
df.residual-methods  Methods for Function "df.residual" in Package "aod"

Description
Computes the number of degrees of freedom of the residuals from fitted objects.

Methods
- ANY  Generic function: see df.residual.
- glimML  Computes the df of residuals for models fitted by betabin or negbin.
- glimQL  Computes the df of residuals for models fitted by quasibin or quasipois.

dja  Mortality of Djallonke Lambs in Senegal

Description
Field trial to assess the effect of ewes deworming (prevention of gastro-intestinal parasitism) on the mortality of their offspring (age < 1 year). This data set is extracted from a large database on small ruminants production and health in Senegal (Lancelot et al., 1998). Data were collected in a sample of herds in Kolda (Upper Casamance, Senegal) during a multi-site survey (Faugère et al., 1992). See also the references below for a presentation of the follow-up survey (Faugère and Faugère, 1986) and a description of the farming systems (Faugère et al., 1990).

Usage
data(dja)

Format
A data frame with 21 observations on the following 4 variables.
- group  a factor with 2 levels: CTRL and TREAT, indicating the treatment.
- village  a factor indicating the village of the herd.
- herd  a factor indicating the herd.
- n  a numeric vector: the number of animals exposed to mortality.
- trisk  a numeric vector: the exposition time to mortality (in year).
- y  a numeric vector: the number of deaths.
References


doner

Test of Proportion Homogeneity using Donner's Adjustment

deker

Description
Tests the homogeneity of proportions between I groups (H0: p_1 = p_2 = ... = p_I ) from clustered binomial data \((n, y)\) using the adjusted \(\chi^2\) statistic proposed by Donner (1989).

Usage

doner(formula = NULL, response = NULL,
weights = NULL, group = NULL, data = c = NULL)

Arguments

formula An optional formula where the left-hand side is either a matrix of the form \(\text{cbind}(y, n-y)\), where the modelled probability is \(y/n\), or a vector of proportions to be modelled \((y/n)\). In both cases, the right-hand side must specify a single grouping variable. When the left-hand side of the formula is a vector of proportions, the argument weight must be used to indicate the denominators of the proportions.

response An optional argument indicating either a matrix of the form \(\text{cbind}(y, n-y)\), where the modelled probability is \(y/n\), or a vector of proportions to be modelled \((y/n)\).

weights An optional argument used when the left-hand side of formula or response is a vector of proportions: weight is the denominator of the proportion.

group An optional argument only used when response is used. In this case, this argument is a factor indicating a grouping variable.

data A data frame containing the response \((n \text{ and } y)\) and the grouping variable.

c If not NULL, a numerical vector of \(I\) cluster correction factors.
Details

The $\chi^2$ statistic is adjusted with the correction factor $C_i$ computed in each group $i$. The test statistic is given by:

$$X^2 = \sum_i \left( \frac{(y_i - n_i * p)^2}{C_i * n_i * p * (1 - p)} \right)$$

where $C_i = 1 + (nA_i - 1) * \rho$, $nA_i$ is a scalar depending on the cluster sizes, and $\rho$ is the ANOVA estimate of the intra-cluster correlation, assumed common across groups (see Donner, 1989 or Donner et al., 1994). The statistic is compared to a $\chi^2$ distribution with $I - 1$ degrees of freedom. Fixed correction factors can be specified with the argument $\text{c}$.

Value

An object of formal class “drs”: see `drs-class` for details. The slot `tab` provides the proportion of successes and the correction factor for each group.

Author(s)

Matthieu Lesnoff<matthieu.lesnoff@cirad.fr>, Renaud Lancelot<renaud.lancelot@cirad.fr>

References


See Also

`chisq.test`, `raoscott`, `drs-class`

Examples

data(rats)
donner(formula = cbind(y, n - y) ~ group, data = rats)
donner(formula = y/n ~ group, weights = n, data = rats)
donner(response = cbind(y, n - y), group = group, data = rats)
donner(response = y/n, weights = n, group = group, data = rats)
# standard test
donner(cbind(y, n - y) ~ group, data = rats, C = c(1, 1))
data(antibio)
donner(cbind(y, n - y) ~ treatment, data = antibio)
Description

Representation of the output of functions `donner` and `raoscott`.

Objects from the Class

Objects can be created by calls of the form `new("drs", ...)` or, more commonly, via the `donner` or `raoscott` functions.

Slots

- `CALL` - The call of the function.
- `tab` - A data frame containing test information. The content of the data frame depends on the type of the function which generated it.
- `rho` - The ANOVA estimate of the intra-cluster correlation (function `donner`).
- `X2` - The adjusted $\chi^2$ statistic.

Methods

- `donner` signature(object = "drs"): see `donner`.
- `raoscott` signature(object = "drs"): see `raoscott`.

Description

Extracts the fitted values from models.

Methods

- `ANY` - Generic function: see `fitted`.
- `glimML` - Extract the fitted values from models of formal class “glimML”, presently generated by functions `betabin` and `negbin`.
- `glimQL` - Extract the fitted values from models of formal class “glimQL”, presently generated by functions `quasibin` and `quasibin`. 
**glimML-class**  

*Representation of Models of Formal Class "glimML"*

**Description**

Representation of models of formal class "glimML" fitted by maximum-likelihood method.

**Objects from the Class**

Objects can be created by calls of the form `new("glimML", ...)` or, more commonly, via the functions `betabin` or `negbin`.

**Slots**

- `call` The call of the function.
- `link` The link function used to transform the mean: "logit", "cloglog" or "log".
- `method` The type of fitted model: “BB” for beta-binomial and “NB” for negative-binomial models.
- `formula` The formula used to model the mean.
- `random` The formula used to model the overdispersion parameter $\phi$.
- `data` Data set to which model was fitted. Different from the original data in case of missing value(s).
- `param` The vector of the ML estimated parameters $b$ and $\phi$.
- `varparam` The variance-covariance matrix of the ML estimated parameters $b$ and $\phi$.
- `fixed.param` The vector of the ML estimated fixed-effect parameters $b$.
- `random.param` The vector of the ML estimated random-effect (correlation) parameters $\phi$.
- `logl` The log-likelihood of the fitted model.
- `logl.max` The log-likelihood of the maximal model (data).
- `dev` The deviance of the model, i.e., $-2 \times (\log L - \log L_{\text{max}})$.
- `df.residual` The residual degrees of freedom of the fitted model.
- `nbpar` The number of estimated parameters, i.e., nbpar = total number of parameters - number of fixed parameters. See argument `fixpar` in `betabin` or `negbin`.
- `iterations` The number of iterations performed in `optim`.
- `code` An integer (returned by `optim`) indicating why the optimization process terminated.
  - 1 Relative gradient is close to 0, current iterate is probably solution.
  - 2 Successive iterates within tolerance, current iterate is probably solution.
  - 3 Last global step failed to locate a point lower than estimate. Either estimate is an approximate local minimum of the function or `steptol` is too small.
  - 4 Iteration limit exceeded.
  - 5 Maximum step size `stepmax` exceeded 5 consecutive times. Either the function is unbounded below, becomes asymptotic to a finite value from above in some direction or `stepmax` is too small.
glimQL-class

msg Message returned by optim.
singular.hessian Logical: true when fitting provided a singular hessian, indicating an overparameterized model.
param.ini The initial values provided to the ML algorithm.
na.action A function defining the action taken when missing values are encountered.

glimQL-class Representation of Models of Formal Class "glimQL"

Description

Representation of models of formal class "glimQL" fitted by quasi-likelihood method.

Objects from the Class

Objects can be created by calls of the form new("glimQL", ...) or, more commonly, via the quasibin or quasipois functions.

Slots

CALL The call of the function.
fm A fitted model of class "glm".
phi The overdispersion parameter.

Methods

show signature(object = "glimQL"): Main results of "glimQL" models.

iccbin Intra-Cluster Correlation for Binomial Data

Description

This function calculates point estimates of the intraclass correlation $\rho$ from clustered binomial data $(n_1, y_1), (n_2, y_2), ..., (n_K, y_K)$ (with $K$ the number of clusters), using a 1-way random effect model. Three estimates, following methods referred to as "A", "B" and "C" in Goldstein et al. (2002), can be obtained.

Usage

iccbin(n, y, data, method = c("A", "B", "C"), nAGQ = 1, M = 1000)
Arguments

- **n**: Vector of the denominators of the proportions.
- **y**: Vector of the numerators of the proportions.
- **data**: A data frame containing the variables `n` and `y`.
- **method**: A character ("A", "B" or "C") defining the calculation method. See Details.
- **nAGQ**: Same as in function `glmer` of package `lme4`. Only for methods "A" and "B". Default to 1.
- **M**: Number of Monte Carlo (MC) replicates used in method "B". Default to 1000.

Details

Before computations, the clustered data are split into binary (0/1) observations $y_{ij}$ (obs. $j$ in cluster $i$). The calculation methods are described in Goldstein et al. (2002). Methods "A" and "B" assume a 1-way generalized linear mixed model, and method "C" a 1-way linear mixed model.

For "A" and "B", function `iccbin` uses the logistic binomial-Gaussian model:

$$y_{ij}|p_{ij} \sim \text{Bernoulli}(p_{ij}),$$

$$\text{logit}(p_{ij}) = b_0 + u_i,$$

where $b_0$ is a constant and $u_i$ a cluster random effect with $u_i \sim N(0, s_u^2)$. The ML estimate of the variance component $s_u^2$ is calculated with the function `glmer` of package `lme4`. The intra-class correlation $\rho = Corr[y_{ij}, y_{ij'}]$ is then calculated with a first-order model linearization around $E[u_i] = 0$ in method "A", and with Monte Carlo simulations in method "B".

For "C", function `iccbin` provides the common ANOVA (moments) estimate of $\rho$. For details, see for instance Donner (1986), Searle et al. (1992) or Ukoumunne (2002).

Value

An object of formal class “iccbin”, with 3 slots:

- **CALL**: The call of the function.
- **features**: A character vector summarizing the main features of the method used.
- **rho**: The point estimate of the intra-class correlation $\rho$.

Author(s)

Matthieu Lesnoff <matthieu.lesnoff@cirad.fr>, Renaud Lancelot <renaud.lancelot@cirad.fr>

References


See Also

iccbin-class, glmer

Examples

data(rats)
tmp <- rats[rats$group == "TREAT", ]
# A: glmm (model linearization)
iccbin(n, y, data = tmp, method = "A")
iccbin(n, y, data = tmp, method = "A", nAGQ = 10)
# B: glmm (Monte Carlo)
iccbin(n, y, data = tmp, method = "B")
iccbin(n, y, data = tmp, method = "B", nAGQ = 10, M = 1500)
# C: lmm (ANOVA moments)
iccbin(n, y, data = tmp, method = "C")

## Not run:
# Example of confidence interval calculation with nonparametric bootstrap
require(boot)
foo <- function(X, ind) {
    n <- X$n[ind]
    y <- X$y[ind]
    X <- data.frame(n = n, y = y)
iccbin(n = n, y = y, data = X, method = "C"}@rho[1]
}
res <- boot(data = tmp[, c("n", "y")], statistic = foo, R = 500, sim = "ordinary", stype = "i")
res.boot.ci(res, conf = 0.95, type = "basic")

## End(Not run)

iccbin-class  Representation of Objects of Formal Class "iccbin"

Description

Representation of the output of function iccbin.

Objects from the Class

Objects can be created by calls of the form new("iccbin", ...) or, more commonly, via the function iccbin.
invlink

Transformation from the Link Scale to the Observation Scale

Description

The function transforms a variable from the link scale to the observation scale: probability or count.

Usage

invlink(x, type = c("cloglog", "log", "logit"))

Arguments

x
A vector of real numbers.

type
A character string. Legal values are “cloglog”, “log” and “logit”.

Value

\[
\text{anti-logit}(x) = \frac{\exp(x)}{1 + \exp(x)} \\
\text{anti-cloglog}(x) = 1 - \exp(-\exp(x))
\]

See Also

link

Examples

x <- seq(-5, 5, length = 100)
plot(x, invlink(x, type = "log"),
     type = "l", lwd = 2, ylab = "Probability")
lines(x, invlink(x, type = "cloglog"), lty = 2, lwd = 2)
grid(col = "black")
legend(-5, 1, legend = c("alogit(x)", "acloglog(x)"),
       lty = c(1, 2), bg = "white")
Transformation from the Observation Scale to the Link Scale

Description

The function transforms a variable from the observation scale (probability or count) to the link scale.

Usage

\[
\text{link}(x, \text{type} = c(\text{"cloglog"}, \text{"log"}, \text{"logit"}))
\]

Arguments

- \(x\): A vector of real numbers.
- \(\text{type}\): A character string. Legal values are “cloglog”, “log” and “logit”.

Value

- \(\text{logit}(x) = \log(x/(1-x))\)
- \(\text{cloglog}(x) = \log(-\log(1-x))\)

See Also

\text{invlink}

Examples

\[
x <- \text{seq}(0.001, 0.999, \text{length} = 100)
\text{plot}(x, \text{link}(x, \text{type} = \text{"logit"}),
\text{type} = \text{"l"}, \text{lwd} = 2, \text{ylab} = \text{"link(proba.")})
\text{lines}(x, \text{link}(x, \text{type} = \text{"cloglog"}), \text{lty} = 2, \text{lwd} = 2)
\text{grid}(\text{col} = \text{"black"})
\text{legend}(0, 6, \text{legend} = c(\text{"logit(x)"}, \text{"cloglog(x)"}),
\text{lty} = c(1, 2), \text{bg} = \text{"white"})
\]

A Comparison of Site Preferences of Two Species of Lizard

Description

“These data describe the daytime habits of two species of lizards, \textit{grahami} and \textit{opalinus}. They were collected by observing occupied sites or perches and recording the appropriate description, namely species involved, time of the day, height and diameter of the perch and whether the site was sunny or shaded. Time of the day is recorded as early, mid-day or late.” (McCullagh and Nelder, 1989, p.129).
Usage

data(lizards)

Format

A data frame with 24 observations on the following 6 variables.

Site  A factor with levels Sun and Shade.
Diameter A factor with levels D <= 2 and D > 2 (inches).
Height  A factor with levels H < 5 and H >= 5 (feet).
Time  A factor with levels Early, Mid-day and Late.
grahami A numeric vector giving the observed sample size for *grahami* lizards.
opalinus A numeric vector giving the observed sample size for *opalinus* lizards.

Details

The data were originally published in Fienberg (1970).

Source


References


Examples

data(lizards)

---

**Description**

Extracts the maximized log-likelihood from fitted models of formal class “glimML”.

Usage

```r
## S4 method for signature 'glimML'
logLik(object, ...)
```

Arguments

- `object` A fitted model of formal class “glimML” (functions `betabin` or `negbin`).
- `...` Other arguments passed to methods.
Value

A numeric scalar with 2 attributes: “df” (number of parameters in the model) and “nobs” (number of observations = degrees of freedom of the residuals + number of parameters in the model).

Methods

ANY  Generic function: see logLik.

glimML  Extract the maximized log-likelihood from models of formal class “glimML”, fitted by functions betabin and negbin.

See Also

logLik in package stats.

mice  

Pregnant Female Mice Experiment

Description

Unpublished laboratory data on the proportion of affected foetuses in two groups (control and treatment) of 10 pregnant female mice (Kupper and Haseman, 1978, p. 75).

Usage

data(mice)

Format

A data frame with 20 observations on the following 3 variables.

  group  a factor with levels CTRL and TREAT
  n  a numeric vector: the total number of foetuses.
  y  a numeric vector: the number of affected foetuses.

References

Description

The function fits a negative-binomial log linear model accounting for overdispersion in counts $y$.

Usage

```r
nenbin(formula, random, data, phi.ini = NULL, warnings = FALSE,
        na.action = na.omit, fixpar = list(),
        hessian = TRUE, control = list(maxit = 2000), ...)
```

Arguments

- `formula`: A formula for the fixed effects. The left-hand side of the formula must be the counts $y$ i.e., positive integers ($y \geq 0$). The right-hand side can involve an offset term.
- `random`: A right-hand formula for the overdispersion parameter(s) $\phi$.
- `data`: A data frame containing the response ($y$) and explanatory variable(s).
- `phi.ini`: Initial values for the overdispersion parameter(s) $\phi$. Default to 0.1.
- `warnings`: Logical to control printing of warnings occurring during log-likelihood maximization. Default to FALSE (no printing).
- `na.action`: A function name. Indicates which action should be taken in the case of missing value(s).
- `fixpar`: A list with 2 components (scalars or vectors) of the same size, indicating which parameters are fixed (i.e., not optimized) in the global parameter vector $(b, \phi)$ and the corresponding fixed values.
  For example, `fixpar = list(c(4, 5), c(0, 0))` means that 4th and 5th parameters of the model are set to 0.
- `hessian`: A logical. When set to FALSE, the hessian and the variances-covariances matrices of the parameters are not computed.
- `control`: A list to control the optimization parameters. See `optim`. By default, set the maximum number of iterations to 2000.
- `...`: Further arguments passed to `optim`.

Details

For a given count $y$, the model is:

$$y \sim \text{Poisson}(\lambda)$$
with \( \lambda \) following a Gamma distribution \( \text{Gamma}(r, \theta) \).

If \( G \) denote the gamma function, then:

\[
P(\lambda) = r^{-\theta} \lambda^{\theta-1} e^{-\frac{\lambda}{r}} \frac{\exp(-\frac{\lambda}{r})}{G(\theta)}
\]

\[
E[\lambda] = \theta \cdot r
\]

\[
\text{Var}[\lambda] = \theta \cdot r^2
\]

The marginal negative-binomial distribution is:

\[
P(y) = G(y + \theta) \cdot \left( \frac{1}{1 + r} \right)^\theta \cdot \left( \frac{r}{1 + r} \right)^y \frac{y!}{y! \cdot G(\theta)}
\]

The function uses the parameterization \( \mu = \theta \cdot r = \exp(Xb) = \exp(\eta) \) and \( \phi = 1/\theta \), where \( X \) is a design-matrix, \( b \) is a vector of fixed effects, \( \eta = Xb \) is the linear predictor and \( \phi \) the overdispersion parameter.

The marginal mean and variance are:

\[
E[y] = \mu
\]

\[
\text{Var}[y] = \mu + \phi \cdot \mu^2
\]

The parameters \( b \) and \( \phi \) are estimated by maximizing the log-likelihood of the marginal model (using the function \text{optim}()). Several explanatory variables are allowed in \( b \). Only one is allowed in \( \phi \).

An offset can be specified in the \text{formula} argument to model rates \( y/T \). The offset and the marginal mean are \( \log(T) \) and \( \mu = \exp(\log(T) + \eta) \), respectively.

Value

An object of formal class “glimML”: see \text{glimML-class} for details.

Author(s)

Matthieu Lesnoff <matthieu.lesnoff@cirad.fr>, Renaud Lancelot <renaud.lancelot@cirad.fr>

References


See Also

\text{glimML-class}, \text{glm} and \text{optim},
\text{glm.nb} in the recommended package \text{MASS},
\text{gnlr} in package \text{gnlm} available at www.luc.ac.be/~jlindsey/rcode.html.
Examples

```r
# without offset
data(salmonella)
negbin(y ~ log(dose + 10) + dose, ~ 1, salmonella)
library(MASS) # function glm.nb in MASS
fm.nb <- glm.nb(y ~ log(dose + 10) + dose,
               link = log, data = salmonella)
coef(fm.nb)
1 / fm.nb$theta # theta = 1 / phi
c(logLik(fm.nb), AIC(fm.nb))
# with offset
data(dja)
negbin(y ~ group + offset(log(trisk)), ~ group, dja)
# phi fixed to zero in group TREAT
negbin(y ~ group + offset(log(trisk)), ~ group, dja,
      fixpar = list(4, 0))
# glm without overdispersion
summary(glm(y ~ group + offset(log(trisk)),
            family = poisson, data = dja))
# phi fixed to zero in both groups
negbin(y ~ group + offset(log(trisk)), ~ group, dja,
      fixpar = list(c(3, 4), c(0, 0)))
```

Germination Data

Description

[Data describing the germination] “for seed Orobanche cernua cultivated in three dilutions of a
bean root extract. The mean proportions of the three sets are 0.142, 0.872 and 0.842, and the overall
mean is 0.614.” (Crowder, 1978, Table 1).

Usage

data(orob1)

Format

A data frame with 16 observations on the following 3 variables.

- **dilution** a factor with 3 levels: 1/1, 1/25 and 1/625.
- **n** a numeric vector: the number of seeds exposed to germination.
- **y** a numeric vector: the number of seeds which actually germinated.

References

**Germination Data**

**Description**

“A 2 x 2 factorial experiment comparing 2 types of seed and 2 root extracts. There are 5 or 6 replicates in each of the 4 treatment groups, and each replicate comprises a number of seeds varying between 4 and 81. The response variable is the proportion of seeds germinating in each replicate.” (Crowder, 1978, Table 3).

**Usage**

```r
data(orob2)
```

**Format**

A data frame with 21 observations on the following 4 variables.

- **seed**: a factor with 2 levels: oWS and oWU.
- **root**: a factor with 2 levels BEAN and CUCUMBER.
- **n**: a numeric vector: the number of seeds exposed to germination.
- **y**: a numeric vector: the number of seeds which actually germinated.

**References**


**predict-methods**

**Methods for Function “predict” in Package “aod”**

**Description**

“predict” methods for fitted models generated by functions in package **aod**.

**Usage**

```r
## S4 method for signature 'glimML'
predict(object, newdata = NULL,
        type = c("response", "link"), se.fit = FALSE, ...)
## S4 method for signature 'glimQL'
predict(object, newdata = NULL,
        type = c("response", "link"), se.fit = FALSE, ...)
```
Arguments

- **object**: A fitted model of formal class “glimML” (functions betabin or negbin) or “glimQL” (functions quasibin or quasipois).
- **newdata**: A data.frame providing all the explanatory variables necessary for predictions.
- **type**: A character string indicating the scale on which predictions are made: either “response” for predictions on the observation scale, or “link” for predictions on the scale of the link.
- **se.fit**: A logical scalar indicating whether pointwise standard errors should be computed for the predictions.
- **...**: Other arguments passed to methods.

Methods

- **glimML**: Compute predictions for models of formal class “glimML”, presently generated by functions betabin and negbin. See the examples for these functions.
- **glimQL**: Compute predictions for models of formal class “glimQL”, presently generated by the functions quasibin and quasibo. See the examples for these functions.

See Also

- predict.glm

---

### quasbin

**Quasi-Likelihood Model for Proportions**

**Description**

The function fits the generalized linear model “II” proposed by Williams (1982) accounting for overdispersion in clustered binomial data \((n, y)\).

**Usage**

```r
quasibin(formula, data, link = c("logit", "cloglog"), phi = NULL, tol = 0.001)
```

**Arguments**

- **formula**: A formula for the fixed effects. The left-hand side of the formula must be of the form `cbind(y, n - y)` where the modelled probability is \(y/n\).
- **link**: The link function for the mean \(p\): “logit” or “cloglog”.
- **data**: A data frame containing the response \((n and y)\) and explanatory variable(s).
- **phi**: When phi is NULL (the default), the overdispersion parameter \(\phi\) is estimated from the data. Otherwise, its value is considered as fixed.
- **tol**: A positive scalar (default to 0.001). The algorithm stops at iteration \(r + 1\) when the condition \(\chi^2[r + 1] - \chi^2[r] \leq tol\) is met by the \(\chi^2\) statistics.
Details

For a given cluster \((n, y)\), the model is:

\[ y \mid \lambda \sim \text{Binomial}(n, \lambda) \]

with \(\lambda\) a random variable of mean \(E[\lambda] = p\) and variance \(Var[\lambda] = \phi \cdot p \cdot (1 - p)\).

The marginal mean and variance are:

\[ E[y] = p \]
\[ Var[y] = p \cdot (1 - p) \cdot [1 + (n - 1) \cdot \phi] \]

The overdispersion parameter \(\phi\) corresponds to the intra-cluster correlation coefficient, which is here restricted to be positive.

The function uses the function \text{glm} and the parameterization: \(p = h(Xb) = h(\eta)\), where \(h\) is the inverse of a given link function, \(X\) is a design-matrix, \(b\) is a vector of fixed effects and \(\eta = Xb\) is the linear predictor.

The estimate of \(b\) maximizes the quasi log-likelihood of the marginal model. The parameter \(\phi\) is estimated with the moment method or can be set to a constant (a regular \text{glm} is fitted when \(\phi\) is set to zero). The literature recommends to estimate \(\phi\) from the saturated model. Several explanatory variables are allowed in \(b\). None is allowed in \(\phi\).

Value

An object of formal class “glimQL”: see \text{glimQL-class} for details.

Author(s)

Matthieu Lesnoff <matthieu.lesnoff@cirad.fr>, Renaud Lancelot <renaud.lancelot@cirad.fr>

References


See Also

\text{glm}, \text{geese} in the contributed package \text{geepack}, \text{glm.binomial.disp} in the contributed package \text{dispmob}.

Examples

data(orob2)
fm1 <- glm(cbind(y, n - y) ~ seed * root,
  family = binomial, data = orob2)
fm2 <- quasibin(cbind(y, n - y) ~ seed * root,
  data = orob2, phi = 0)
fm3 <- quasibin(cbind(y, n - y) ~ seed * root,
  data = orob2)
rbind(fm1 = coef(fm1), fm2 = coef(fm2), fm3 = coef(fm3))
# show the model
fm3
# dispersion parameter and goodness-of-fit statistic
c(phi = fm3@phi,  
X2 = sum(residuals(fm3, type = "pearson")^2))
# model predictions
predfm1 <- predict(fm1, type = "response", se = TRUE)
predfm3 <- predict(fm3, type = "response", se = TRUE)
New <- expand.grid(seed = levels(orob2$seed),  
  root = levels(orob2$root))
predict(fm3, New, se = TRUE, type = "response")
data.frame(orob2, p1 = predfm1$fit,  
  se.p1 = predfm1$se.fit,  
  p3 = predfm3$fit,  
  se.p3 = predfm3$se.fit)
f4 <- quasibin(cbind(y, n - y) - seed + root,  
  data = orob2, phi = fm3@phi)
# Pearson's chi-squared goodness-of-fit statistic
# compare with fm3's X2
sum(residuals(f4, type = "pearson")^2)

---

**quasipois**  
*Quasi-Likelihood Model for Counts*

### Description

The function fits the log linear model ("Procedure II") proposed by Breslow (1984) accounting for overdispersion in counts y.

### Usage

```r
quasipois(formula, data, phi = NULL, tol = 0.001)
```

### Arguments

- **formula**: A formula for the fixed effects. The left-hand side of the formula must be the counts y i.e., positive integers (y >= 0). The right-hand side can involve an offset term.
- **data**: A data frame containing the response (y) and explanatory variable(s).
- **phi**: When phi is NULL (the default), the overdispersion parameter $\phi$ is estimated from the data. Otherwise, its value is considered as fixed.
- **tol**: A positive scalar (default to 0.001). The algorithm stops at iteration $r + 1$ when the condition $\chi^2[r + 1] - \chi^2[r] \leq tol$ is met by the $\chi^2$ statistics.
Details

For a given count $y$, the model is:

$$y \sim \text{Poisson}(\lambda)$$

with $\lambda$ a random variable of mean $E[\lambda] = \mu$ and variance $\text{Var}[\lambda] = \phi \cdot \mu^2$.

The marginal mean and variance are:

$$E[y] = \mu$$
$$\text{Var}[y] = \mu + \phi \cdot \mu^2$$

The function uses the function glm and the parameterization: $\mu = \exp(Xb) = \exp(\eta)$, where $X$ is a design-matrix, $b$ is a vector of fixed effects and $\eta = Xb$ is the linear predictor. The estimate of $b$ maximizes the quasi log-likelihood of the marginal model. The parameter $\phi$ is estimated with the moment method or can be set to a constant (a regular glm is fitted when $\phi$ is set to 0). The literature recommends to estimate $\phi$ with the saturated model. Several explanatory variables are allowed in $b$. None is allowed in $\phi$.

An offset can be specified in the argument formula to model rates $y/T$ (see examples). The offset and the marginal mean are $\log(T)$ and $\mu = \exp(\log(T) + \eta)$, respectively.

Value

An object of formal class “glimQL”: see glmQL-class for details.

Author(s)

Matthieu Lesnoff <matthieu.lesnoff@cirad.fr>, Renaud Lancelot <renaud.lancelot@cirad.fr>

References


See Also

glm, negative.binomial in the recommended package MASS, geese in the contributed package geepack, glm.poisson.disp in the contributed package dispmod.

Examples

```r
# without offset
data(salmonella)
quasipois(y ~ log(dose + 10) + dose,
data = salmonella)
quasipois(y ~ log(dose + 10) + dose,
data = salmonella, phi = 0.07180449)
summary(glm(y ~ log(dose + 10) + dose,
family = poisson, data = salmonella))
quasipois(y ~ log(dose + 10) + dose,
data = salmonella, phi = 0)
```
rabbits

Rabbits Foetuses Survival Experiment

Description

Experimental data for analyzing the effect of an increasing dose of a compound on the proportion of live foetuses affected. Four treatment-groups were considered: control “C”, low dose “L”, medium dose “M” and high dose “H”. The animal species used in the experiment was banded Dutch rabbit (Paul, 1982, Table 1).

Usage

data(rabbits)

Format

A data frame with 84 observations on the following 3 variables.

- **group** a factor with levels C, H, L and M
- **n** a numeric vector: the total number of foetuses.
- **y** a numeric vector: the number of affected foetuses.

References

raoscott

Test of Proportion Homogeneity using Rao and Scott’s Adjustment

Description

Tests the homogeneity of proportions between I groups (H0: \( p_1 = p_2 = ... = p_I \)) from clustered binomial data \((n, y)\) using the adjusted \(\chi^2\) statistic proposed by Rao and Scott (1993).

Usage

```r
raoscott(formula = NULL, response = NULL, weights = NULL,
         group = NULL, data, pooled = FALSE, deff = NULL)
```

Arguments

- **formula**: An optional formula where the left-hand side is either a matrix of the form `cbind(y, n-y)`, where the modelled probability is \(y/n\), or a vector of proportions to be modelled \((y/n)\). In both cases, the right-hand side must specify a single grouping variable. When the left-hand side of the formula is a vector of proportions, the argument `weights` must be used to indicate the denominators of the proportions.

- **response**: An optional argument: either a matrix of the form `cbind(y, n-y)`, where the modelled probability is \(y/n\), or a vector of proportions to be modelled \((y/n)\).

- **weights**: An optional argument used when the left-hand side of `formula` or `response` is a vector of proportions: `weight` is the denominator of the proportions.

- **group**: An optional argument only used when `response` is used. In this case, this argument is a factor indicating a grouping variable.

- **data**: A data frame containing the response \((n\text{ and } y)\) and the grouping variable.

- **pooled**: Logical indicating if a pooled design effect is estimated over the \(I\) groups.

- **deff**: A numerical vector of \(I\) design effects.

Details

The method is based on the concepts of design effect and effective sample size.

The design effect in each group \(i\) is estimated by \(def_i = vratio_i / vbin_i\), where \(vratio_i\) is the variance of the ratio estimate of the probability in group \(i\) (Cochran, 1999, p. 32 and p. 66) and \(vbin_i\) is the standard binomial variance. A pooled design effect (i.e., over the \(I\) groups) is estimated if argument `pooled` = TRUE (see Rao and Scott, 1993, Eq. 6). Fixed design effects can be specified with the argument `deff`.

The \(def_i\) are used to compute the effective sample sizes \(nadj_i = n_i / def_i\), the effective numbers of successes \(yadj_i = y_i / def_i\) in each group \(i\), and the overall effective proportion \(padj = \sum_i yadj_i / \sum_i def_i\). The test statistic is obtained by substituting these quantities in the usual \(\chi^2\) statistic, yielding:

\[
X^2 = \sum_i \frac{(yadj_i - nadj_i * padj)^2}{nadj_i * padj * (1 - padj)}
\]
which is compared to a $\chi^2$ distribution with $I - 1$ degrees of freedom.

**Value**

An object of formal class “drs”: see *drs-class* for details. The slot tab provides the proportion of successes, the variances of the proportion and the design effect for each group.

**Author(s)**

Matthieu Lesnoff <matthieu.lesnoff@cirad.fr>, Renaud Lancelot <renaud.lancelot@cirad.fr>

**References**


**See Also**

chisq.test, donner, iccbin, drs-class

**Examples**

data(rats)
# deff by group
raoscott(cbind(y, n - y) ~ group, data = rats)
raoscott(y/n ~ group, weights = n, data = rats)
raoscott(response = cbind(y, n - y), group = group, data = rats)
raoscott(response = y/n, weights = n, group = group, data = rats)
# pooled deff
raoscott(cbind(y, n - y) ~ group, data = rats, pooled = TRUE)
# standard test
raoscott(cbind(y, n - y) ~ group, data = rats, deff = c(1, 1))
data(antibio)
raoscott(cbind(y, n - y) ~ treatment, data = antibio)

---

### rats

**Rats Diet Experiment**

**Description**

“Weil (1970) in Table 1 gives the results from an experiment comprising two treatments. One group of 16 pregnant female rats was fed a control diet during pregnancy and lactation, the diet of a second group of 16 pregnant females was treated with a chemical. For each litter the number $n$ of pups alive at 4 days and the number $x$ of pups that survived the 21 day lactation period were recorded.” (Williams, 1975, p. 951).
Usage

data(rats)

Format

A data frame with 32 observations on the following 3 variables.

- **group**: A factor with levels CTRL and TREAT
- **n**: A numeric vector: the number of pups alive at 4 days.
- **y**: A numeric vector: the number of pups that survived the 21 day lactation.

Source


References

Weil, C.S., 1970. *Selection of the valid number of sampling units and a consideration of their combination in toxicological studies involving reproduction, teratogenesis or carcinogenesis*. Fd. Cosmet. Toxicol. 8, 177-182.

---

Residuals methods

**Description**

Residuals of models fitted with functions `betabin` and `negbin` (formal class “glimML”), or `quasibin` and `quasipois` (formal class “glimQL”).

**Usage**

```r
## S4 method for signature 'glimML'
residuals(object, type = c("pearson", "response"), ...)
## S4 method for signature 'glimQL'
residuals(object, type = c("pearson", "response"), ...)
```

**Arguments**

- `object`: Fitted model of formal class “glimML” or “glimQL”.
- `type`: Character string for the type of residual: “pearson” (default) or “response”.
- `...`: Further arguments to be passed to the function, such as `na.action`.
Details

For models fitted with betabin or quasibin, Pearson’s residuals are computed as:

\[
\frac{y - n \cdot \hat{p}}{\sqrt{n \cdot \hat{p} \cdot (1 - \hat{p}) \cdot (1 + (n - 1) \cdot \phi)}}
\]

where \( y \) and \( n \) are respectively the numerator and the denominator of the response, \( \hat{p} \) is the fitted probability and \( \phi \) is the fitted overdispersion parameter. When \( n = 0 \), the residual is set to 0.

Response residuals are computed as \( y/n - \hat{p} \).

For models fitted with negbin or quasipois, Pearson’s residuals are computed as:

\[
\frac{y - \hat{y}}{\sqrt{\hat{y} + \phi \cdot \hat{y}^2}}
\]

where \( y \) and \( \hat{y} \) are the observed and fitted counts, respectively. Response residuals are computed as \( y - \hat{y} \).

Value

A numeric vector of residuals.

Author(s)

Matthieu Lesnoff <matthieu.lesnoff@cirad.fr>, Renaud Lancelot <renaud.lancelot@cirad.fr>

See Also

residuals.glm

Examples

data(orob2)
fm <- betabin(cbind(y, n - y) ~ seed, ~ 1,
  link = "logit", data = orob2)
#Pearson's chi-squared goodness-of-fit statistic
sum(residuals(fm, type = "pearson")^2)

salmonella

Salmonella Reverse Mutagenicity Assay

Description

“Data for our third example were compiled by Margolin et al. (1981) from an Ames Salmonella reverse mutagenicity assay. Table 1 shows the number of revertant colonies observed on each of 3 replicate plates tested at each of 6 dose levels of quinoline.” (Breslow, 1984, Table 1).
**Usage**

```r
data(salmonella)
```

**Format**

A data frame with 18 observations on the following 2 variables.

- **dose** a numeric vector: the dose level of quinoline (microgram per plate).
- **y** a numeric vector: the number of revertant colonies of TA98 *Salmonella*.

**Source**


**References**


---

**splitbin**  
*Split Grouped Data Into Individual Data*

**Description**

The function splits grouped data and optional covariates into individual data. Two types of grouped data are managed by `splitbin`:

- Grouped data with weights;
- Grouped data of form `cbind(success, failure)`.

When weights, successes or failures involve non-integer numbers, these numbers are rounded before splitting.

**Usage**

```r
splitbin(formula, data, id = "idbin")
```

**Arguments**

- **formula** A formula. The left-hand side represents grouped data. The right-hand side defines the covariates. See examples for syntax.
- **data** A data frame where all the variables described in `formula` are found.
- **id** An optional character string naming the identifier (= grouping factor). Default to “idbin”.
summary.aic-method

Value

A data frame built according to the formula and function used in the call.

Examples

```r
# Grouped data with weights
mydata <- data.frame(
  success = c(0, 1, 0, 1),
  f1 = c("A", "A", "B", "B"),
  f2 = c("C", "D", "C", "D"),
  n = c(4, 2, 1, 3)
)
mydata
splitbin(formula = n ~ f1, data = mydata)$tab
splitbin(formula = n ~ f1 + f2 + success, data = mydata)$tab

# Grouped data of form "cbind(success, failure)"
mydata <- data.frame(
  success = c(4, 1),
  failure = c(1, 2),
  f1 = c("A", "B"),
  f2 = c("C", "D")
)
mydata$n <- mydata$success + mydata$failure
mydata
splitbin(formula = cbind(success, failure) ~ 1, data = mydata)$tab
splitbin(formula = cbind(success, failure) ~ f1 + f2, data = mydata)$tab
splitbin(formula = cbind(success, n - success) ~ f1 + f2, data = mydata)$tab
splitbin(formula = cbind(success, n - 0.5 * failure - success) ~ f1 + f2,
  data = mydata)$tab
```

Description

Computes Akaike difference and Akaike weights from an object of formal class “aic”.

Usage

```r
## S4 method for signature 'aic'
summary(object, which = c("AIC", "AICc"))
```

Arguments

object  An object of formal class “aic”.
which  A character string indicating which information criterion is selected to compute
Akaike difference and Akaike weights: either “AIC” or “AICc”.
Methods

**summary**  The models are ordered according to AIC or AICc and 3 statistics are computed:
- the *Akaike difference* $\Delta$: the change in AIC (or AICc) between successive (ordered) models,
- the *Akaike weight* $W$: when $r$ models are compared, $W = e^{-0.5 \times \Delta} / \sum_r e^{-0.5 \times \Delta}$,
- the *cumulative Akaike weight* cum.$W$: the Akaike weights sum to 1 for the $r$ models which are compared.

References


See Also

Examples in betabin and AIC in package stats.

---

summary.glimML-class  *Summary of Objects of Class "summary.glimML"*

Description

Summary of a model of formal class “glimML” fitted by betabin or negbin.

Objects from the Class

Objects can be created by calls of the form `new("summary.glimML", ...)` or, more commonly, via the summary or show method for objects of formal class “glimML”.

Slots

- **object**  An object of formal class “glimML”.
- **Coef**  A data frame containing the estimates, standard error, z and P values for the fixed-effect coefficients which were estimated by the fitting function.
- **FixedCoef**  A data frame containing the values of the fixed-effect coefficients which were set to a fixed value.
- **Phi**  A data frame containing the estimates, standard error, z and P values for the overdispersion coefficients which were estimated by the fitting function. Because the overdispersion coefficients are $> 0$, P values correspond to unilateral tests.
- **FixedPhi**  A data frame containing the values of the overdispersion coefficients which were set to a fixed value.
Methods

```r
show signature(object = "summary.glimML")
show signature(object = "glimML")
summary signature(object = "glimML")
```

Examples

```r
data(orb2)
fm1 <- betabin(cbind(y, n - y) ~ seed, ~ 1, data = orb2)
# show objects of class "glimML"
fm1
# summary for objects of class "glimML"
res <- summary(fm1)
res@Coef
# show objects of class "summary.glimML"
res
```

---

**varbin**

*Mean, Variance and Confidence Interval of a Proportion*

---

**Description**

This function computes the mean and variance of a proportion from clustered binomial data \((n, y)\), using various methods. Confidence intervals are computed using a normal approximation, which might be inappropriate when the proportion is close to 0 or 1.

**Usage**

```r
varbin(n, y, data, alpha = 0.05, R = 5000)
```

**Arguments**

- **n**
  - The denominator of the proportion.
- **y**
  - The numerator of the proportion.
- **data**
  - A data frame containing the data.
- **alpha**
  - The significance level for the confidence intervals. Default to 0.05, providing 95% CI’s.
- **R**
  - The number of bootstrap replicates to compute the bootstrap mean and variance.
Details

Five methods are used for the estimations. Let us consider $N$ clusters of sizes $n_1, \ldots, n_N$ with observed responses (counts) $y_1, \ldots, y_N$. We note $p_i = y_i/n_i$ the observed proportions ($i = 1, \ldots, N$). An underlying assumption is that the theoretical proportion is homogeneous across the clusters.

**Binomial method:** the proportion and its variance are estimated as $p = \frac{\sum_i y_i}{\sum_i n_i}$ and $\frac{p(1-p)}{\sum_i n_i-1}$, respectively.

**Ratio method:** the one-stage cluster sampling formula is used to estimate the variance of the ratio estimate (see Cochran, 1999, p. 32 and p. 66). The proportion is estimated as above ($p$).

**Arithmetic method:** the proportion is estimated as $p_A = \frac{1}{N} \sum_i \frac{y_i}{n_i}$, with estimated variance $\frac{\sum_i (p_i-p_A)^2}{N(N-1)}$.

**Jackknife method:** the proportion $p_j$ is the arithmetic mean of the pseudovalues $p v_i$, with estimated variance $\frac{\sum_i (pv_i-p_j)^2}{N(N-1)}$ (Gladen, 1977, Paul, 1982).

**Bootstrap method:** $R$ samples of size $N$ are drawn with equal probability from the initial sample $(p_1, \ldots, p_N)$ (Efron and Tibshirani, 1993). The bootstrap estimate $p_B$ and its estimated variance are the arithmetic mean and the empirical variance (computed with denominator $R - 1$) of the $R$ binomial estimates, respectively.

Value

An object of formal class “varbin”, with 5 slots:

- **CALL** The call of the function.
- **tab** A 4-column data frame giving for each estimation method the mean, variance, upper and lower limits of the $(1 - \alpha)$ confidence interval.
- **boot** A numeric vector containing the $R$ bootstrap replicates of the proportion. Might be used to compute other kinds of CI’s for the proportion.
- **alpha** The significance level used to compute the $(1 - \alpha)$ confidence intervals.
- **features** A numeric vector with 3 components summarizing the main features of the data: $N =$ number of clusters, $n =$ number of subjects, $y =$ number of cases.

The “show” method displays the slot tab described above, substituting the standard error to the variance.

Author(s)

Matthieu Lesnoff<matthieu.lesnoff@cirad.fr>, Renaud Lancelot <renaud.lancelot@cirad.fr>

References

See Also

varbin-class, boot

Examples

data(rabbits)
varbin(n, y, rabbits[rabbits$group == "M", ])
by(rabbits,
    list(group = rabbits$group),
    function(x) varbin(n = n, y = y, data = x, R = 1000))

varbin-class Representation of Objects of Formal Class "varbin"

Description

Representation of the output of function varbin used to estimate proportions and their variance
under various distribution assumptions.

Objects from the Class

Objects can be created by calls of the form new("varbin", ...) or, more commonly, via the
function varbin.

Slots

CALL  The call of the function.
tab   A data frame containing the estimates, their variance and the confidence limits.
pboot A numeric vector containing the bootstrap replicates.
alpha The α level to compute confidence intervals.
features A named numeric vector summarizing the design.

Methods

varbin signature(object = "varbin"): see varbin.
Description

Extract the approximate var-cov matrix of estimated coefficients from fitted models.

Methods

ANY  Generic function: see vcov.

glimML  Extract the var-cov matrix of estimated coefficients for fitted models of formal class “glimML”.

glimQL  Extract the var-cov matrix of estimated coefficients for fitted models of formal class “glimQL”.

geese  Extract the var-cov matrix of estimated coefficients for fitted models of class “geese” (contributed package geepack).

geeglm  Extract the var-cov matrix of estimated coefficients for fitted objects of class “geeglm” (contributed package geepack).

wald.test  Wald Test for Model Coefficients

Description

Computes a Wald $\chi^2$ test for 1 or more coefficients, given their variance-covariance matrix.

Usage

wald.test(Sigma, b, Terms = NULL, L = NULL, H0 = NULL,
          df = NULL, verbose = FALSE)

# S3 method for class 'wald.test'
print(x, digits = 2, ...)

Arguments

Sigma  A var-cov matrix, usually extracted from one of the fitting functions (e.g., lm, glm,...).

b  A vector of coefficients with var-cov matrix Sigma. These coefficients are usually extracted from one of the fitting functions available in R (e.g., lm, glm,...).

Terms  An optional integer vector specifying which coefficients should be jointly tested, using a Wald $\chi^2$ or $F$ test. Its elements correspond to the columns or rows of the var-cov matrix given in Sigma. Default is NULL.
An optional matrix conformable to \( b \), such as its product with \( b \), i.e., \( L \times b \) gives the linear combinations of the coefficients to be tested. Default is NULL.

A numeric vector giving the null hypothesis for the test. It must be as long as Terms or must have the same number of columns as \( L \). Default to 0 for all the coefficients to be tested.

A numeric vector giving the degrees of freedom to be used in an \( F \) test, i.e. the degrees of freedom of the residuals of the model from which \( b \) and \( \Sigma \) were fitted. Default to NULL, for no \( F \) test. See the section Details for more information.

A logical scalar controlling the amount of output information. The default is FALSE, providing minimum output.

Object of class “wald.test”

Number of decimal places for displaying test results. Default to 2.

Additional arguments to print.

Details

The key assumption is that the coefficients asymptotically follow a (multivariate) normal distribution with mean = model coefficients and variance = their var-cov matrix.

One (and only one) of Terms or \( L \) must be given. When \( L \) is given, it must have the same number of columns as the length of \( b \), and the same number of rows as the number of linear combinations of coefficients. When \( df \) is given, the \( \chi^2 \) Wald statistic is divided by \( m = \text{length(Terms)} \) or \( \text{nrow}(L) \). Under the null hypothesis \( H_0 \), this new statistic follows an \( F(m, df) \) distribution.

Value

An object of class wald.test, printed with print.wald.test.

References


See Also

vcov

Examples

```r
data(orob2)
fm <- quasibin(cbind(y, n - y) ~ seed * root, data = orob2)
# Wald test for the effect of root
wald.test(b = coef(fm), Sigma = vcov(fm), Terms = 3:4)
```
Index

*Topic classes
  aic-class, 2
drs-class, 15
glimML-class, 16
glimQL-class, 17
iccbin-class, 19
summary.glimML-class, 39
varbin-class, 42

*Topic datagen
  splitbin, 37

*Topic datasets
  antibio, 5
cohorts, 11
dja, 12
lizards, 21
mice, 23
orob1, 26
orob2, 27
rabbits, 32
rats, 34
salmonella, 36

*Topic htest
  donner, 13
  iccbin, 17
  raoscott, 33
  varbin, 40
  wald.test, 43

*Topic math
  invlink, 20
  link, 21

*Topic methods
  AIC-methods, 3
  coef-methods, 10
deviance-methods, 11
df.residual-methods, 12
fitted-methods, 15
logLik-methods, 22
predict-methods, 27
summary, aic-method, 38

vcov-methods, 43

*Topic package
  aod-pkg, 6

*Topic regression
  anova-methods, 4
  betabin, 7
  negbin, 24
  quasibin, 28
  quasipois, 30
  residuals-methods, 35

AIC, 3–5, 39
AIC,glimML-method (AIC-methods), 3
aic-class, 2
AIC-methods, 3
anova,glimML-method (anova-methods), 4
anova-methods, 4
anova,glimML-class (anova-methods), 4
anova.glim, 5
antibio, 5
aod (aod-pkg), 6
aod-pkg, 6

betabin, 4, 7, 16, 28, 39
boot, 42

chisq.test, 14, 34
coeff, 10
coef,glimML-method (coef-methods), 10
coeff,glimQL-method (coef-methods), 10
coeff-methods, 10
cohorts, 11
deviance, 11
deviance,glimML-method
  (deviance-methods), 11
deviance-methods, 11
df.residual, 12
df.residual,glimML-method
  (df.residual-methods), 12
INDEX

df.residual, glimQL-method
  (df.residual-methods), 12
df.residual-methods, 12
dja, 12
donner, 13, 15, 34
drs-class, 15
fitted, 15
fitted, glimML-method (fitted-methods), 15
fitted, glimQL-method (fitted-methods), 15
fitted-methods, 15
geeglm-class (vcov-methods), 43
goose, 29, 31
goose-class (vcov-methods), 43
glimML-class, 16
glimQL-class, 17
glm, 9, 25, 29, 31
glm.binomial, disp, 29
glm.nb, 25
glm.poisson, disp, 31
glmer, 19
iccbin, 17, 20, 34
iccbin-class, 19
invlink, 20, 21
link, 20, 21
lizards, 21
logLik, 23
logLik, glimML-method (logLik-methods), 22
logLik-methods, 22
mice, 23
negative.binomial, 31
negbin, 16, 24, 28
optim, 8, 9, 24, 25
orob1, 26
orob2, 27
predict, glimML-method
  (predict-methods), 27
predict, glimQL-method
  (predict-methods), 27
predict-methods, 27
predict.glm, 28
print.wald.test (wald.test), 43
quasibin, 28, 28
quasipois, 30
rabbits, 32
raoscott, 14, 15, 33
rats, 34
residuals, glimML-method
  (residuals-methods), 35
residuals, glimQL-method
  (residuals-methods), 35
residuals-methods, 35
residuals.glm, 36
salmonella, 36
show,aic-method (summary,aic-method), 38
show, anova.glimML-method
  (anova-methods), 4
show, donner-class (donner), 13
show, drs-method (drs-class), 15
show, glimML-class
  (summary.glimML-class), 39
show, glimML-method (glimML-class), 16
show, glimQL-method (glimQL-class), 17
show, iccbin-method (iccbin), 17
show, raoscott-class (raoscott), 33
show, summary.glimML-method
  (summary.glimML-class), 39
show, varbin-class (varbin), 40
show, varbin-method (varbin-class), 42
splitbin, 37
summary, aic-method, 38
summary, glimML-method
  (summary.glimML-class), 39
summary.glimML-class, 39
varbin, 40, 42
varbin-class, 42
vcov, 43, 44
vcov, geeglm-method (vcov-methods), 43
vcov, geese-method (vcov-methods), 43
vcov, glimML-method (vcov-methods), 43
vcov, glimQL-method (vcov-methods), 43
vcov-methods, 43
wald.test, 43