

Package ‘glmglrt’

March 19, 2020

Type Package

Title GLRT P-Values in Generalized Linear Models

Version 0.1.0

Maintainer André GILLIBERT <andre.gillibert@chu-rouen.fr>

Author André GILLIBERT [aut, cre]

Description Provides functions to compute Generalized Likelihood Ratio Tests (GLRT) and Rao's score tests of simple and complex contrasts of Generalized Linear Models (GLMs). It provides the same interface as `summary.glm()`, adding GLRT P-values, less biased than Wald's P-values and consistent with profile-likelihood confidence interval generated by `confint()`.

License GPL-2

Encoding UTF-8

LazyData true

Imports base, stats, parameters (>= 0.1.0), MASS

RoxygenNote 7.1.0

Suggests testthat (>= 2.1.0), lme4, nlme, datasets

Enhances nnet, survival, lmerTest

NeedsCompilation no

Repository CRAN

Date/Publication 2020-03-19 17:20:02 UTC

R topics documented:

<code>df_for_wald</code>	2
<code>fixcoef</code>	3
<code>glmglrt</code>	4
<code>override_summary</code>	5
<code>print.summary.glmglrt</code>	6
<code>p_value.glm</code>	7
<code>p_value_contrast</code>	8

summarylr	10
vcov_fixcoef	12

Index	14
--------------	-----------

df_for_wald	<i>Gets the degree of freedom for Wald tests involving the model</i>
-------------	--

Description

This generic function is used by `p_value_contrast.default` to get the number of degrees of freedom of the [t distribution](#) that approximates the point estimate of the contrast divided by its standard error.

Usage

```
df_for_wald(object, ...)

## S3 method for class 'glm'
df_for_wald(object, ...)

## Default S3 method:
df_for_wald(object, ...)
```

Arguments

object	statistical model;
...	Unused by <code>p_value_contrast.default</code> , but may be useful to some custom specializations.

Details

This function is quite similar to `df.residual` but it should return Inf when the Student's t distribution is less appropriate than the normal distribution.

Value

A finite value or Inf for normal distribution approximation.

Methods (by class)

- `glm`: Returns `df.residual` for linear gaussian models and Inf for all other models in order to make Wald's tests consistent with the behavior of `stats::summary.glm(object)`
- `default`: Simple proxy to `df.residual` but replaces NAs with Inf

See Also

Other Wald-related functions: `fixcoef()`, `p_value_contrast()`, `vcov_fixcoef()`

Examples

```
# 10 observations, one coefficient, 9 degrees of freedom
df_for_wald(glm(I(1:10) ~ 1))
# returns Inf (non-gaussian-identity model)
df_for_wald(glm(family="poisson", c(10,20,30) ~ 1))
data(mtcars)
# returns Inf (non-gaussian-identity model)
df_for_wald(glm(family="binomial", data=mtcars, I(hp > median(hp)) ~ cyl))
```

fixcoef

Generic function to get fixed effects of a model

Description

This is a generic S3 function that gets point estimates of fixed effects of a statistical model, implemented on a wide range of models and that can be extended to new models.

Usage

```
fixcoef(model, ...)

## S3 method for class 'lmerMod'
fixcoef(model, ...)

## S3 method for class 'glmerMod'
fixcoef(model, ...)

## S3 method for class 'lmerModLmerTest'
fixcoef(model, ...)

## S3 method for class 'lme'
fixcoef(model, ...)

## S3 method for class 'multinom'
fixcoef(model, ...)

## S3 method for class 'mlm'
fixcoef(model, ...)

## Default S3 method:
fixcoef(model, ...)
```

Arguments

model	a fitted statistical model
...	argument unused by <code>p_value_contrast.default</code> but that may be useful to some specializations.

Details

It must return only estimates of fixed-effects of a model. Random effects are ignored. The `names` of the element of this vector must be consistent with the `rownames` and `colnames` of the variance-covariance matrix that `vcov_fixcoef` returns. The `vcov_fixcoef` function, on the same model, must return a matrix with the same number and names of rows and columns as the length of the vector returned by `fixcoef`.

The functions `vcov_fixcoef` and `fixcoef` would be pointless if the behavior of `vcov` and `coef` were not inconsistent from package to package.

`fixcoef` and `vcov_fixcoef`, together with `df_for_wald` are used by `p_value_contrast.default`

Value

Simple numeric vector with one item for each fixed effect of the model.

Methods (by class)

- `lmerMod`: implementation for `lme4::lmer`
- `glmerMod`: implementation for `lme4::glmer`
- `lmerModLmerTest`: implementation for `lmerTest::lmer`
- `lme`: implementation for `nlme::lme`
- `multinom`: implementation for `nnet::multinom`
- `mlm`: implementation for multiple responses linear models generated by `stats::lm` when the response is a matrix.
- `default`: default implementation, simply calls `coef(model)`.

See Also

Other Wald-related functions: `df_for_wald()`, `p_value_contrast()`, `vcov_fixcoef()`

Examples

```
data(mtcars)
fixcoef(lm(data=mtcars, hp ~ 1)) # get mean horse power of cars listed in mtcars
```

glmglrt

glmglrt: GLRT P-Values in Generalized Linear Models

Description

This package has been developed to provide Generalized Likelihood Ratio Tests (GLRT) also known as Likelihood Ratio Tests (LRT) to Generalized Linear Models (GLMs). The `stats` package do support LRT P-values with `anova` and derived confidence intervals with `confint()`, but provides Wald's P-values with the `summary` function. This is unfortunate for two reasons: Wald's P-values may be inconsistent with profile-likelihood confidence intervals and Wald's P-values, on small samples are more biased than LRT P-values, for non-gaussian models. The `anova` function is not as simple as `summary`, since it requires manually fitting two models.

Summary function

This package provides a way to override (see [override_summary](#)) the standard [summary.glm](#) function by a [summarylr](#) function that provides LRT and/or Rao's score P-values.

Generic p_value functions

It also provides a function [p_value_contrast](#) to make LRT or Rao hypothesis tests on contrasts of coefficients of GLMs. This is an alternative to [multcomp::glht](#) without Wald's approximation ! It also provides a less powerful [p_value.glm](#) method for the S3 generic [parameters::p_value](#). It also extends this S3 generic for a variety of models as [p_value.default](#). That time, the only method supported for all models, is Wald's method.

override_summary

Overrides the Generalized Linear Models summary methods

Description

This function overrides the [summary.glm](#) and [summary.negbin](#) S3 methods by the [summarylr](#) function in the calling environment.

Usage

```
override_summary()
```

Details

Although some minor compatibility issues may exist when calling this function in the global environment, most scripts should work with it. Indeed [summarylr](#) behaves like [summary.glm](#) but adds a `$extra` field containing P-value info. The first letter of the field name ('e') is unique, avoiding problems with scripts that access fields with short names (e.g. `model$x` for `model$xlevels`).

See Also

Other Extended GLM summary functions: [print.summary.glmglrt\(\)](#), [summarylr\(\)](#)

Examples

```
model = glm(family="binomial", cbind(50,30) ~ 1)
override_summary()
summary(model) # Additional 'LRT P-value' column
```

```
print.summary.glmglrt Prints the summary generated by summarylr
```

Description

This function prints a `summary.glmglrt` object generated by `summarylr`. It works like the standard `summary.glm` function but additionally displays columns showing Rao or LRT P-values.

Usage

```
## S3 method for class 'summary.glmglrt'
print(
  x,
  ...,
  has.Pvalue = TRUE,
  tst.ind = 3,
  debuglevel = NULL,
  keep.wald = NULL
)
```

Arguments

<code>x</code>	a <code>summary.glmglrt</code> object generated by <code>summarylr</code> .
<code>...</code>	additional arguments passed to <code>stats::print.summary.glm</code> then <code>printCoefmat</code> . The most useful ones are <code>digits</code> and <code>signif.stars</code> .
<code>has.Pvalue</code>	logical value; passed to <code>printCoefmat</code> ; if TRUE, the P-value column is formatted by <code>format.pval</code> .
<code>tst.ind</code>	integer vector of length ≥ 0 ; passed to <code>printCoefmat</code> ; it changes the format of these columns, assuming they are statistics columns.
<code>debuglevel</code>	NULL or integer value; set to NULL to use the <code>debuglevel</code> argument that was specified in <code>summarylr</code> , 0 (default) to disable warnings, 1 to enable warnings and 2 to enable warnings and notes.
<code>keep.wald</code>	NULL or logical; set to NULL to use the <code>keep.wald</code> argument that was specified in <code>summarylr</code> . If TRUE, the standard Wald's P-values are displayed. If FALSE, the standard Wald's P-values are hidden.

See Also

Other Extended GLM summary functions: `override_summary()`, `summarylr()`

Examples

```
model = glm(family="binomial", cbind(50,30) ~ 1)
print(summarylr(model), signif.stars=FALSE, digits=10)
```

p_value.glm *Computing p-values of hypothesis tests on coefficients of Generalized Linear Models*

Description

This S3 method is a specialization of `parameters::p_value` for `stats::glm` models. By default, it computes Wald's P-values that are known to be more biased than LRT P-values, but the behavior can be overridden by the `method="LRT"` argument. This is for compatibility with the default method of `parameters::p_value`.

Usage

```
## S3 method for class 'glm'
p_value(
  model,
  method = NULL,
  parm = NULL,
  alternative = c("two.sided", "less", "greater"),
  H0 = 0,
  debuglevel = 1,
  force = FALSE,
  ...
)
```

Arguments

model	glm object; as obtained by calling <code>stats::glm</code> or <code>MASS::glm.nb</code> .
method	character value; may either be "LRT" (synonym "Chisq"), "Rao", "wald" (default value, synonym "Wald" and "F").
parm	integer or character vector or NULL; specify coefficients to test, by name or indexes. the default <code>parm=NULL</code> outputs all coefficients.
alternative	a character string specifying the alternative hypothesis, must be one of "two.sided" (default), "greater" or "less". You can specify just the initial letter.
H0	numeric vector of length 1 or of the same length as <code>parm</code> ; the value of the coefficient under the null hypothesis. Zero by default.
debuglevel	integer value; set to 0 (default) to disable warnings, 1 to enable warnings and 2 to enable warnings and notes.
force	logical; if TRUE, force computation of P-values in case of convergence problems.
...	Ignored arguments. Allows compatibility with the generic <code>parameters::p_value</code> .

Value

a data.frame with two columns; the first column, Parameter represents the name of the coefficient and p (second column) represents the P-value of the hypothesis test against H0

Examples

```

require("parameters")
mod = glm(family="poisson", c(2,30) ~ c(0,1), offset=log(c(8,30)))
# Wald's tests (biased)
p_value(mod)
# Rao score tests (biased)
p_value(mod, method="Rao")
# LRT tests (less biased)
p_value(mod, method="LRT")

# only test slope (faster since only one test is performed)
p_value(mod, method="LRT", parm=2)
# is slope greater than log(2) ?
p_value(mod, method="LRT", parm=2, H0=log(2), alternative="greater")

```

p_value_contrast *Hypothesis tests on contrasts*

Description

This S3 generic function allows the computation of P-values associated to hypothesis tests of contrasts (i.e. linear combinations) of fixed-effects in a model

Usage

```

p_value_contrast(
  model,
  contrast,
  alternative = c("two.sided", "less", "greater"),
  H0 = 0,
  method = NULL,
  ...
)

## S3 method for class 'glm'
p_value_contrast(
  model,
  contrast,
  alternative = c("two.sided", "less", "greater"),
  H0 = 0,
  method = c("LRT", "Rao", "Chisq", "F", "Wald", "wald"),
  ...,
  debuglevel = 1,
  force = FALSE
)

## Default S3 method:

```



```

p_value_contrast(
  model,
  contrast,
  alternative = c("two.sided", "less", "greater"),
  H0 = 0,
  method = "Wald",
  ...,
  debuglevel = 0
)

```

Arguments

model	a fitted statistical model such as a glm or a coxph.
contrast	numeric vector of the same length as the number of coefficients in the model; it describes the contrast <code>sum(contrast*fixcoef(model))</code> .
alternative	a character string specifying the alternative hypothesis, must be one of "two.sided" (default), "greater" or "less". You can specify just the initial letter.
H0	numeric value; the value of the contrast under the null hypothesis.
method	character string value; specification of the algorithm used (implementation dependent). Suggested values are "Wald", "LRT", "Rao" and "exact" for, respectively, Wald's asymptotic normality and/or student test, the Generalized Likelihood Ratio Test, Rao's score test and non-asymptotic exact tests. Other values may be allowed.
...	Additional parameters that may be used by some implementations.
debuglevel	integer value; set to 0 (default) to disable warnings, 1 to enable warnings and 2 to enable warnings and notes.
force	logical; if TRUE, force computation of P-values in case of convergence problems.

Details

Every implementation MUST support specification of the alternative hypothesis (alternative argument) and null hypothesis (H0 argument).

Value

A single numeric value equal to the one-sided (for `alternative="less"` or `"greater"`) or two-sided P-value

Methods (by class)

- `glm`: It supports Wald (`method="Wald"`), Generalized Likelihood Ratio Tests (`method="LRT"`) and Rao's score tests (`method="Rao"`). It works for `stats::glm` models and negative binomial models (`MASS::glm.nb`).
- `default`: Supports Wald's test on a wide range of models: `lm`, `mlm`, `glm`, `negbin`, `polr`, `r1m`, `multinom`, `gls`, `nlme::lme`, `lme4::lmer`, `lme4::glmer`, `coxph`, `survreg`. It can be easily extended by implementing three generic functions: `fixcoef`, `vcov_fixcoef` and `df_for_wald`.

If the implementation of `coef`, `vcov` and `df.residual` is correct, you do not have to implement `fixcoef`, `vcov_fixcoef` and `df_for_wald`.

See Also

`p_value_contrast.default`, `p_value_contrast.glm`

Other Wald-related functions: `df_for_wald()`, `fixcoef()`, `vcov_fixcoef()`

Examples

```
data(mtcars)
model1 = glm(family="gaussian", data=mtcars, hp ~ 0+factor(gear))
# do cars with 5 gears have more horse power (hp) than cars with 4 gears ?
p_value_contrast(model1, c(0,-1,1), alternative="greater")

# now, we fit an equivalent model (same distribution and same predictions)
model2 = glm(family=gaussian(log), data=mtcars, hp ~ 0+factor(gear))

# do cars with 5 gears have at least twice the horse power than cars with 4 gears ?

# the following two tests are equivalent
p_value_contrast(model1, c(0,-1,0.5), alternative="greater", method="LRT", H0=0)
p_value_contrast(model2, c(0,-1,1), alternative="greater", method="LRT", H0=log(2))

# the following two tests are close but not equivalent
p_value_contrast(model1, c(0,-1,0.5), alternative="greater", method="Wald", H0=0)
p_value_contrast(model2, c(0,-1,1), alternative="greater", method="Wald", H0=log(2))
```

summarylr

Summarizes a glm, adding a column of GLRT or Rao score P-values

Description

`summarylr` is an improved summary function for standard `glm` (stats package) adding LRT or Rao score P-values

Usage

```
summarylr(
  object,
  dispersion = NULL,
  correlation = FALSE,
  symbolic.cor = FALSE,
  ...,
  force = FALSE,
  debuglevel = level_warning,
  method = "LRT",
  keep.wald = FALSE
)
```

Arguments

object	glm object; as obtained by calling <code>stats::glm</code> or <code>MASS::glm.nb</code> .
dispersion	the dispersion parameter for the family used. Either a single numerical value or NULL (the default), when it is inferred from object (see <code>stats::summary.glm</code>).
correlation	logical; if TRUE, the correlation matrix of the estimated parameters is returned and printed (see <code>stats::summary.glm</code>).
symbolic.cor	logical; if TRUE, print the correlations in a symbolic form (see <code>symnum</code>) rather than as numbers (see <code>stats::summary.glm</code>).
...	Additional arguments to be passed to <code>stats::summary.glm</code>
force	logical; if TRUE, force computation of P-values in case of convergence problems.
debuglevel	integer value; set to 0 (default) to disable warnings, 1 to enable warnings and 2 to enable warnings and notes.
method	NULL or character vector of length 0, 1 or 2; may be code "LRT" or "Rao" or <code>c("LRT", "Rao")</code> to compute specified P-values. You can set <code>method=NULL</code> to compute no additional P-values.
keep.wald	logical; if TRUE, the standard Wald's P-values are kept in the output of <code>print.summary.glmglrt</code> . Even if <code>keep.wald=FALSE</code> , the standard wald P-values are not erased from the <code>summary.glmglrt</code> object. They are only hidden by <code>print.summary.glmglrt</code> .

Details

This function works the same as the standard `summary.glm` function but provides additional parameters. The core parameter `method="LRT"` makes `summarylr` add a column LRT P-value to the output. This P-value is computed by repeatedly fitting the model dropping one coefficient at a time and using the `anova.glm(test="Chisq")` function to perform generalized likelihood ratio test by approximation of the deviance difference to a chi-square distribution. This provides P-values less biased than the standard Wald P-values that `summary` provides. Moreover, this LRT method is consistent with the profile likelihood confidence intervals that `confint.glm` provides. The option `method="Rao"` generates Rao's score P-values. `method="Chisq"` is synonymous to `method="LRT"`. For exhaustivity, the option `method="Wald"` (synonym "wald", "F") generates Wald's P-values. Several methods can be used, e.g. `method=c("LRT", "Rao")` computes both LRT and Rao P-values. New methods may be added in the future.

Extra parameters are passed-through to the `summary.glm` function.

Value

It returns a summary object of type `summary.glmglrt` that gets pretty printed by `link[glmglrt:print.summary.glmglrt]`. The return value is an S3 object compatible with `stats::summary.glm` but with an additional field `$extra` field having sub-fields. `$extra$pvalues` is a numeric matrix with columns "LRT P-value" and/or "Rao P-value", containing the relevant P-values. As new columns may be added in future, you should rely on column names rather than column indexes. Only P-values of methods requested in the `method` parameter are stored in this matrix. `$extra$debuglevel` is equal to the `debuglevel` passed to `summarylr`. `$extra$keep.wald` is equal to the `keep.wald` passed to `summarylr`. In case of convergence problems, the field `$extra$problem_of_convergence` will be added. It will be a character string with the value "general" (because `model$converged = FALSE`), "all" (because

all coefficients have huge variances) or "specific" (because at least one coefficient has a huge variance). Other problem strings may be added in the future. If `weights` are specified in a way that make P-values invalid, the field `$extra$problem_weights` will be added as a character string describing the problem. Actually, the only known problem is "non-constant".

See Also

Other Extended GLM summary functions: `override_summary()`, `print.summary.glmglrt()`

Examples

```
summarylr(glm(family="binomial", cbind(5,3)~1))
data(mtcars)
# do not properly converge (warnings)
mtcars$outcome = mtcars$disp > median(mtcars$disp)
mod=glm(family=binomial(log), data=mtcars,outcome ~ 0+qsec+wt,start=c(-0.1,0.3))
summarylr(mod) # warns that P-values are not computed because model did not converge
summarylr(mod, force=TRUE) # compute P-values anyway !
# also works with negative binomial models
summarylr(MASS::glm.nb(data=mtcars, I(cyl*gear) ~ 1+wt,link="sqrt"),test="LRT")
```

vcov_fixcoef

Gets the variance-covariance matrix of fixed effects of a fitted model

Description

This is a generic S3 function that gets the variance-covariance matrix of fixed effects of a statistical model, implemented on a wide range of models and that can be extended to new models.

Usage

```
vcov_fixcoef(model, ...)

## Default S3 method:
vcov_fixcoef(model, ...)

## S3 method for class 'survreg'
vcov_fixcoef(model, ...)
```

Arguments

<code>model</code>	a fitted statistical model
<code>...</code>	argument unused by <code>p_value_contrast.default</code> but that may be useful to some specializations.

Details

It must return variance-covariance for fixed effects of a model, not random effects nor scale parameters. The rownames and colnames of the returned matrix must be consistent with `names` of `fixcoef(object)`.

The functions `vcov_fixcoef` and `fixcoef` would be pointless if the behavior of `vcov` and `coef` were not inconsistent from package to package.

`fixcoef` and `vcov_fixcoef`, together with `df_for_wald` are used by `p_value_contrast.default`

Methods (by class)

- `default`: default implementation, simple proxy of `vcov(model)`
- `survreg`: implementation for `survreg`, removing the extra column for Scale

See Also

Other Wald-related functions: `df_for_wald()`, `fixcoef()`, `p_value_contrast()`

Examples

```
data(mtcars)
mod = lm(data=mtcars, hp ~ cyl+wt)
est = fixcoef(mod) # get estimates
SE = sqrt(diag(vcov_fixcoef(mod))) # get standard errors of estimates
z = est/SE # get z-score of estimates
df = df_for_wald(mod) # degrees of freedom
pvalues = 2*pt(-abs(z), df=df) # get two-sided P-values
```

Index

anova, [4](#)
anova.glm(test=Chisq), [11](#)

coef, [4](#), [10](#), [13](#)
confint.glm, [11](#)
coxph, [9](#)

df.residual, [2](#), [10](#)
df_for_wald, [2](#), [4](#), [9](#), [10](#), [13](#)

fixcoef, [2](#), [3](#), [4](#), [9](#), [10](#), [13](#)
format.pval, [6](#)

glmglrt, [4](#)
gls, [9](#)

lme4::glmer, [4](#), [9](#)
lme4::lmer, [4](#), [9](#)
lmerTest::lmer, [4](#)

MASS::glm.nb, [7](#), [9](#), [11](#)
mlm, [9](#)
multcomp::glht, [5](#)

names, [4](#), [13](#)
negbin, [9](#)
nlme::lme, [4](#), [9](#)
nnet::multinom, [4](#)

override_summary, [5](#), [5](#), [6](#), [12](#)

p_value.default, [5](#)
p_value.glm, [5](#), [7](#)
p_value_contrast, [2](#), [4](#), [5](#), [8](#), [13](#)
p_value_contrast.default, [2-4](#), [10](#), [12](#), [13](#)
p_value_contrast.glm, [10](#)
parameters::p_value, [5](#), [7](#)
print.summary.glmglrt, [5](#), [6](#), [11](#), [12](#)
printCoefmat, [6](#)

stats, [4](#)

stats::glm, [7](#), [9](#), [11](#)
stats::lm, [4](#)
stats::print.summary.glm, [6](#)
stats::summary.glm, [11](#)
stats::summary.glm(object), [2](#)
summary, [4](#)
summary.glm, [5](#), [6](#), [11](#)
summary.negbin, [5](#)
summarylr, [5](#), [6](#), [10](#)
survreg, [9](#)
symnum, [11](#)

t distribution, [2](#)

vcov, [4](#), [10](#), [13](#)
vcov_fixcoef, [2](#), [4](#), [9](#), [10](#), [12](#), [13](#)

weights, [12](#)