Package ‘flipscores’

November 22, 2024

Version 1.3.1

Date 2024-11-22

Title Robust Score Testing in GLMs, by Sign-Flip Contributions

Description Provides robust tests for testing in GLMs, by sign-flipping score contributions. The tests are robust against overdispersion, heteroscedasticity and, in some cases, ignored nuisance variables. See Hemerik, Goeman and Finos (2020) <doi:10.1111/rssb.12369>.

Imports MASS, methods

Encoding UTF-8

License GPL-2

RoxygenNote 7.3.2

NeedsCompilation no

Author Livio Finos [cre, aut] (<https://orcid.org/0000-0003-3181-8078>),
Jelle J. Goeman [ctb],
Jesse Hemerik [ctb],
Riccardo De Santis [ctb]

Maintainer Livio Finos <livio.finos@unipd.it>

Repository CRAN

Date/Publication 2024-11-22 09:00:09 UTC

Contents

anova.flipscores *anova.flipscores*

Description

This is the anova method for flipscores object. Remark: it performs type III deviance decomposition as in car::Anova.

Usage

```
## S3 method for class 'flipscores'
anova(object, model1 = NULL, score_type = NULL, n_flips = 5000, id = NULL, ...)
```

Arguments

`object` (the object) glm (or flipscores) object with the model under the null hypothesis (i.e. the covariates, the nuisance parameters).

`model1` a glm (or flipscores) or a matrix (or vector). If it is a glm object, it has the model under the alternative hypothesis. The variables in model1 are the same variables in object plus one or more variables to be tested. Alternatively, if model1 is a matrix, it contains the tested variables column-wise.

`score_type` The type of score that is computed. It can be "orthogonalized", "effective" or "basic". Default is "orthogonalized". "effective" and "orthogonalized" take into account the nuisance estimation. The default is NULL, in this case the value is taken from object.

`n_flips` The number of random flips of the score contributions. When n_flips is equal or larger than the maximum number of possible flips (i.e. n^2), all possible flips are performed. Default is 5000.

`id` a vector identifying the clustered observations. If NULL (default) observations are assumed to be independent. NOTE: if object is a flipscores and model$flip_param_call$id is not NULL, this is considered in the inference.

`...` other parameters allowed in stats::anova.

Examples

```
set.seed(1)
dt=data.frame(X=scale(rnorm(50)),
   Z=factor(rep(LETTERS[1:3],length.out=50)))
dt$Y=rpois(n=nrow(dt),lambda=exp(dt$X*(dt$Z=="C")))
mod0=flipscores(Y~Z+X,data=dt,family="poisson")
summary(mod0)
anova(mod0)

mod1=flipscores(Y~Z*X,data=dt,family="poisson")
summary(mod1)
anova(mod0,model1 = mod1)
```

compute_scores *compute_scores*

Description

Same usage as `anova.glm`. The parameter `id` is used too, if present in `model0` (with priority) or in `model1`.

Usage

```
compute_scores(model0, model1, score_type = "standardized", ...)
```

Arguments

`model0`	a `glm` object with the model under the null hypothesis (i.e. the covariates, the nuisance parameters).
`model1`	a `glm` or a `matrix` (or `vector`). If it is a `glm` object, it has the model under the alternative hypothesis. The variables in `model1` are the same variables in `model0` plus one or more variables to be tested. Alternatively, if `model1` is a `matrix`, it contains the tested variables column-wise.
`score_type`	The type of score that is computed. It is "orthogonalized", "effective" or "basic". "effective" and "orthogonalized" take into account the nuisance estimation.
`...`	other arguments.

Author(s)

Jesse Hemerik, Riccardo De Santis, Vittorio Giatti, Jelle Goeman and Livio Finos

Examples

```
set.seed(1)
Z=rnorm(20)
X=Z+rnorm(20)
Y=rpois(n=20,lambda=exp(Z+X))
mod0=glm(Y~Z,family="poisson")
X=data.frame(X=X)
scr0=compute_scores(model0 = mod0, model1 = X)
head(scr0)
```

flipscores *Robust testing in GLMs, by sign-flipping score contributions*

Description

Provides robust tests for testing in GLMs, by sign-flipping score contributions. The tests are often robust against overdispersion, heteroscedasticity and, in some cases, ignored nuisance variables.

Usage

```
flipscores(formula, family, data, score_type = "standardized",
n_flips = 5000, alternative = "two.sided", id = NULL,
seed = NULL, to_be_tested = NULL, flips = NULL,
precompute_flips = TRUE, output_flips = FALSE, ...)
```

Arguments

`formula`	see `glm` function. It can also be a model (usually generated by a call to `glm`); in this case, any other glm-related parameter (e.g. `family, data, etc.`) are discarded, the function will make use of the ones used to generate the model. (i.e. `formula`, `family`, `data`, etc) are not considered. It is `NULL` by default (i.e. not used).
`family`	see `glm` function.
`data`	see `glm` function.
`score_type`	The type of score that is computed. It can be "standardized" "orthogonalized", "effective" or "basic". Both "orthogonalized" and "effective" take into account the nuisance estimation and they provide the same test statistic. In case of small samples "effective score" might have a slight anti-conservative behaviour. "standardized effective score" gives a solution for this issue. "orthogonalized" has a similar intent, note however that in case of a big model matrix, it may be slow.
`n_flips`	The number of random flips of the score contributions. Overwritten with the `nrow(flips)` when `flips` is not `NULL` (see parameter `flips` for more details). When `n_flips` is equal or larger than the maximum number of possible flips (i.e. n^2), all possible flips are performed.
`alternative`	It can be "greater", "less" or "two.sided" (default)
`id`	a `vector` identifying the clustered observations. If `NULL` (default) observations are assumed to be independent. If `id` is not `NULL`, only `score_type=="effective"` is allowed, yet.
`seed`	`NULL` by default.
`to_be_tested`	vector of indices or names of coefficients of the glm model to be tested (it is faster than computing every scores and p-values of course).
`flips`	matrix fo +1 or -1, the matrix has `n_flips` rows and n (number of observations) columns

`precompute_flips`
: `TRUE` by default. Overwritten if `flips` is not `NULL`. If `FALSE` the matrix of flips is not computed and the flips are made ’on-the-fly’ before computing the test statistics; it may be usefull when `flips` is very large (see parameter `flips` for more details).

`output_flips`
: `FALSE` by default. If `TRUE` the `flips` matrix is returned. Useful when the same flips are needed for more glms, for example in the case of multivariate glms where the joint distribution of test statistis if used for multivariate inference.

`...`
: see `glm` function.

Details

`flipscores` borrows the same parameters from function `glm` (and `glm.nb`). See these helps for more details about parameters such as `formula`, `data`, `family`. Note: in order to use Negative Binomial family, `family` reference must have quotes (i.e. `family="negbinom"`). Furthermore, `flipscores` object contains two extra elements: `scores` – i.e. a matrix of `n` score contributions, one column for each tested coefficient – and `Tspace` – i.e. a matrix of size `n_flips` times `ncol(scores)`. The fist row of `Tspace` contains column-wise the test statistics generated by randomly flipping the score contributions, each column refers to the same column of `scores`, the vector of observed test statistics (i.e. no flips) is in the first row of `Tspace`.

Value

an object of class `flipscores`. See also its methods (`summary.flipscores`, `anova.flipscores`, `print.flipscores`).

Author(s)

Livio Finos, Riccardo De Santis, Jesse Hemerik and Jelle Goeman

References

"Robust testing in generalized linear models by sign-flipping score contributions" by J.Hemerik, J.Goeman and L.Finos.

See Also

`anova.flipscores`, `summary.flipscores`, `flip`

Examples

```
set.seed(1)
dt=data.frame(X=rnorm(20),
   Z=factor(rep(LETTERS[1:3],length.out=20)))
dt$Y=rpois(n=20,lambda=exp(dt$Z=="C"))
mod=flipscores(Y~Z+X,data=dt,family="poisson",n_flips=1000)
summary(mod)

# Equivalent to:
model=glm(Y~Z+X,data=dt,family="poisson")
```

```
mod2=flipscores(model)
summary(mod2)
```

`flipscores-method` *Methods for flipscores objects*

Description

Methods for `flipscores` objects. The following are methods to extract and manipulate relevant information from a `flipscores` object.

Usage

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

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

Arguments

`x`	a flipscores object
`...`	additional arguments to be passed
`object`	a flipscores object

`make_flips` *It creates a `n_flipsxn_obs` matrix of random +1 and -1. The first row is made by ones (i.e. the observed test statistic is computed)*

Description

It creates a `n_flipsxn_obs` matrix of random +1 and -1. The first row is made by ones (i.e. the observed test statistic is computed)

Usage

```
make_flips(n_obs, n_flips)
```

Arguments

`n_obs`	number of observations
`n_flips`	number of flips

Examples

```
# example code
make_flips(n_obs=10,n_flips=20)
```

Index