Package 'crctStepdown'

October 12, 2022

Title Univariate Analysis of Cluster Trials with Multiple Outcomes

Version 0.2.1

Description Frequentist statistical inference for cluster randomised trials with multiple outcomes that controls the family-wise error rate and provides nominal coverage of confidence sets. A full description of the methods can be found in Watson et al. (2021) <arXiv:2107.10017>.

License CC BY-SA 4.0

Encoding UTF-8

RoxygenNote 7.1.1

LinkingTo Rcpp

Imports fastglm, Rcpp, RcppArmadillo, rlang, ggplot2, ggpubr, methods, stringr, lme4

Depends R (>= 2.10)

Suggests testthat

NeedsCompilation yes

Author Sam Watson [aut, cre] (<https://orcid.org/0000-0002-8972-769X>)

Maintainer Sam Watson <s.i.watson@bham.ac.uk>

Repository CRAN

Date/Publication 2021-12-16 08:40:05 UTC

R topics documented:

`conf_int_search` *Randomisation test confidence interval limit search*

Description

A multi-variate Robbins-Monroe search process to estimate the upper of lower limits for a confidence set for parameters from a list of fitted model objects.

Usage

```
conf_int_search(
  fitlist,
  data,
  actual_tr,
  start,
  nsteps = 1000,
  alpha = 0.05,
  plots = TRUE,
  cl_var = "cl",
  rand_func = NULL,
  verbose = TRUE,
  type = "rw"
)
```

Arguments

`fitlist`	A list of p fitted mer class model objects
`data`	A data frame used to fit the models in fitlist
`actual_tr`	A vector of length p with the original point estimates of the treatment effects
`start`	A vector of length p with a set of starting values for the search. The uncorrected confidence set limits (tr_eff +/- 2*SE) usually provide a good starting point.
`nsteps`	Number of steps for the search process.
`alpha`	Numeric value. The process searches for the 100(1-2*alpha) confidence intervals
`plots`	Logical value indicating whether to plot the search process. Default to TRUE
`cl_var`	String indicating the name of the column in data with the IDs of the clusters
`rand_func`	The name of a function that re-randomises the clusters. The function should produce a data frame that identifies the clusters in the treatment group under the new randomisation scheme. The data frame can either have a single column with name cl_var or two columns of cl_var and t identifying the cluster ID and time period a cluster joins the treatment group. If NULL then clusters are randomised in a 1:1 ratio to treatment and control
`verbose`	Logical indicating whether to provide verbose output showing progress and estimates

type — Method of correction: options are "rw" = Romano-Wolf randomisation test based stepdown, "h" = Holm standard stepdown, "b" = Bonferroni, or "none" for no correction.

Details

A version of the search process proposed by Garthwaite (1996) adapted for multiple limits. Given a set of estimates of the upper or lower, the process calculates the test statistics for the two-sided null hypotheses that the treatment effects equal these values, and then conducts a single iteration randomisation test of the same null hypotheses. The estimates are then updated based on whether the actual test statistic is higher or lower than the randomisation test statistic it would be compared to under the resampling stepdown approach of Romano & Wolf (2005). At the limits of the confidence set all values should be rejected in a two-sided hypothesis test with a family-wise error rate of alpha, which provides a probabilistic basis for the search process. See Watson (2021) for more details.

Value

A vector of length p with the estimates of the limits

Examples

```
out <- twoarm_sim()
data <- out[[1]]
fit1 <- lme4::glmer(y1 ~ treat + (1|cl) ,
                    data=data,
                    family="poisson")

fit2 <- lme4::glmer(y2 ~ treat + (1|cl),
                    data=data,
                    family="poisson")
fitlist <- list(fit1,fit2)
tr_eff <- rep(NA,2)
for(i in 1:2){
    res <- summary(fitlist[[i]])
    tr_eff[i] <- res$coefficients["treat",'Estimate']
}
conf_int_search(fitlist,
                data = data,
                actual_tr=tr_eff,
                start=tr_eff+1,
                nsteps=100,
                alpha=0.025,
                plots = FALSE,
                cl_var = "cl",
                verbose = FALSE)
```

`est_null_model` *Estimates null model*

Description

Given an lme4 model object and the value of the treatment effect parameter under the null hypothesis, the function returns a glm or lm object fitted under the null model with no cluster effects. For linear models (lmer) the value of the null is subtracted from the value of the outcome for those in receipt of the treatment and an lm model is fitted with no treatment effect. For generalised linear models (glmer) the model is refitted as a glm model with the treatment effect specified as an offset.

Usage

```
est_null_model(fit, data, tr_var = "treat", null_par)
```

Arguments

`fit`	An lme4 model object
`data`	The data frame used to fit model fit
`tr_var`	A string indicating the name of the column in data that is a binary indicator for whether the observation was under the treatment (1=treatment, 0=control)
`null_par`	Numeric the value of tr_var parameter under the null hypothesis

Value

An lm or glm model fit under the null model

Examples

```
out <- twoarm_sim()
data <- out[[1]]
  fit1 <- lme4::glmer(y1 ~ treat + (1|cl) ,
data=data,
family="poisson")

fit2 <- lme4::glmer(y2 ~ treat + (1|cl),
                    data=data,
                    family="poisson")
fitlist <- list(fit1,fit2)
nullfitlist <- list()
for(i in 1:length(fitlist)){
  nullfitlist[[i]] <- est_null_model(fitlist[[i]],
                                     data,
                                     tr_var = "treat",
                                     null_par = 0)
}
```

gen_rand_order *Function to generate a stepped-wedge cRCT randomisation allocation*

Description

Function to generate a stepped-wedge cRCT randomisation allocation. Assumes a baseline and endline period in which no clusters and all clusters have the intervention, respectively.

Usage

```
gen_rand_order(nJ, nT)
```

Arguments

nJ	Number of clusters
nT	Number of time points

Value

A data frame with columns cl and t indicating the time

lme_permute2 *Generate a new permutation*

Description

Returns the test statistic from a specified null hypothesis and model under a single new permutation

Usage

```
lme_permute2(
  fitlist,
  data,
  null_par = rep(0, length(fitlist)),
  cl_var = "cl",
  rand_func = NULL
)
```

Arguments

fitlist	A list of glm model objects fitted under the null hypotheses
data	A data frame containing the data used to fit the models in fitlist
null_par	A vector of the same length as fitlist specifying the value(s) of the treatment effect parameter(s) under the null hypotheses
cl_var	String specifying the name of the column identifying the clusters/cluster-time

rand_func — String of the name of a function that re-randomises the clusters. The function should produce a data frame that identifies the clusters in the treatment group under the new randomisation scheme. The data frame can either have a single column with name cl_var or two columns of cl_var and t identifying the cluster ID and time period a cluster joins the treatment group. If NULL then clusters are randomised in a 1:1 ratio to treatment and control

Value

A vector of the length of fitlist with the test statistics for each model and null hypothesis

Examples

```
out <- twoarm_sim()
data <- out[[1]]
  fit1 <- lme4::glmer(y1 ~ treat + (1|cl) ,
data=data,
family="poisson")

fit2 <- lme4::glmer(y2 ~ treat + (1|cl),
                    data=data,
                    family="poisson")
fitlist <- list(fit1,fit2)
nullfitlist <- list()
for(i in 1:length(fitlist)){
  nullfitlist[[i]] <- est_null_model(fitlist[[i]],
                                     data,
                                     tr_var = "treat",
                                     null_par = 0)
}
out <- lme_permute2(nullfitlist,
               data=data,
               cl_var = "cl")
```

---

outname_fit — *Extracts the dependent variable name from glm, lm, or mer model*

Description

Extracts the dependent variable name from glm, lm, or mer model

Usage

```
outname_fit(fit)
```

Arguments

fit — A fitted model object of class glm, lm, or *merMod

Value

A string with the name of the dependent variable from the model

Examples

```
out <- twoarm_sim()
data <- out[[1]]
fit1 <- lme4::glmer(y1 ~ treat + (1|cl) ,
                    data=data,
                    family="poisson")
outname_fit(fit1)
```

---

permute *Wrapper function to replicate permutations*

Description

Calls lme_permute2 to replicate the permutations

Usage

```
permute(
  fitlist,
  data,
  n_permute = 100,
  null_pars = rep(0, length(fitlist)),
  cl_var = "cl",
  rand_func = NULL
)
```

Arguments

fitlist	A list of glm model objects fitted under the null hypotheses
data	A data frame containing the data used to fit the models in fitlist
n_permute	Number of permutations to conduct
null_pars	A vector of the same length as fitlist specifying the value(s) of the treatment effect parameter(s) under the null hypotheses
cl_var	String specifying the name of the column identifying the clusters/cluster-time
rand_func	The name of a function that re-randomises the clusters. The function should produce a data frame that identifies the clusters in the treatment group under the new randomisation scheme. The data frame can either have a single column with name cl_var or two columns of cl_var and t identifying the cluster ID and time period a cluster joins the treatment group. If NULL then clusters are randomised in a 1:1 ratio to treatment and control

Value

An array of dimension length(fitlist)*n_permute containing the test statistics for each model and each iteration

Examples

```
out <- twoarm_sim()
data <- out[[1]]
  fit1 <- lme4::glmer(y1 ~ treat + (1|cl) ,
data=data,
family="poisson")

fit2 <- lme4::glmer(y2 ~ treat + (1|cl),
                    data=data,
                    family="poisson")
fitlist <- list(fit1,fit2)
nullfitlist <- list()
for(i in 1:length(fitlist)){
  nullfitlist[[i]] <- est_null_model(fitlist[[i]],
                                     data,
                                     tr_var = "treat",
                                     null_par = 0)
}
out <- permute(nullfitlist,
               data=data,
               n_permute = 10,
               cl_var = "cl")
```

`perm_dist` *Extracts the test statistics*

Description

Extracts the test statistics from the output of the `permute` function. Returns the largest value from a specified subset of rows, each row is the test statistic from a different null hypothesis.

Usage

```
perm_dist(out, positions)
```

Arguments

`out`	Array output by the `permute` function
`positions`	Vector indicating which rows of out to use

Value

Vector of numeric values of length `ncol(out)`

qscore_stat *Calculates the randomisation test statistic*

Description

Calculates a randomisation test statistic based on the sum of generalised, studentized residuals for a given model fit and null hypothesis of the value of the treatment effect parameter

Usage

```
qscore_stat(
  fit,
  data,
  null_par = 0,
  tr_var = "treat",
  cl_var = "cl",
  tr_assign = "treat"
)
```

Arguments

fit	A glm or lm model fit
data	The data frame used to fit model fit
null_par	Numeric the value of tr_var parameter under the null hypothesis
tr_var	String indicating the name of the column in data that is a binary indicator for whether the observation was under the treatment (1=treatment, 0=control)
cl_var	String specifying the name of the column identifying the clusters/cluster-time
tr_assign	String specifying the treatment assignment under a new permutation. If calculating the test statistics under the original treatment assignment then this should be the same as tr_var

Value

The value of the test statistic

Examples

```
out <- twoarm_sim()
data <- out[[1]]
  fit1 <- glm(y1 ~ treat ,
              data=data,
              family="poisson")
qscore_stat(fit=fit1,
            data=data)
```

`stepdown` *Conduct the randomisation-based stepdown procedure*

Description

For a set of models fit with lme4, the function will conduct the randomisation tests and generate p-values for the null hypotheses of no treatment effect that controls the family-wise error rate, and generates a 100(1-alpha)% confidence set for the treatment effect model parameters.

Usage

```
stepdown(
  fitlist,
  tr_var = "treat",
  cl_var = "cl",
  data,
  alpha = 0.05,
  plots = TRUE,
  n_permute = 1000,
  nsteps = 1000,
  type = "rw",
  rand_func = NULL,
  confint = TRUE,
  verbose = TRUE
)
```

Arguments

`fitlist`	A list of models fitted with lme4. All models should be fit using the same data frame.
`tr_var`	String indicating the name of the column in data that is a binary indicator for whether the observation was under the treatment (1=treatment, 0=control)
`cl_var`	String specifying the name of the column identifying the clusters/cluster-time
`data`	A data frame containing the data used to fit the models in fitlist
`alpha`	Numeric. 100(1-alpha)% confidence intervals are calculated. Default it 0.05
`plots`	Logical indicating whether to plot permutational distributions and confidence interval search during running of function. Default is TRUE
`n_permute`	Number of permutations of the randomisation test to run
`nsteps`	Number of steps of the confidence interval search process
`type`	Method of correction: options are "rw" = Romano-Wolf randomisation test based stepdown, "h" = Holm standard stepdown, "h" = Holm stepdown using randomisation test, "b" = standard Bonferroni, "br" = Bonerroni using randomisation test, or "none" = randomisation test with no correction.

rand_func	String of the name of a function that re-randomises the clusters. The function should produce a data frame that identifies the clusters in the treatment group under the new randomisation scheme. The data frame can either have a single column with name cl_var or two columns of cl_var and t identifying the cluster ID and time period a cluster joins the treatment group. If NULL then clusters are randomised in a 1:1 ratio to treatment and control
confint	Logical indicating whether to run the confidence interval search process
verbose	Logical indicating whether to provide detailed output

Value

A data frame with the point estimates, p-values, and confidence intervals

Examples

```
out <- twoarm_sim()
data <- out[[1]]
  fit1 <- lme4::glmer(y1 ~ treat + (1|cl) ,
data=data,
family="poisson")

fit2 <- lme4::glmer(y2 ~ treat + (1|cl),
                    data=data,
                    family="poisson")
 stepdown(fitlist=list(fit1,fit2),
          data=data,
          n_permute = 100,
          nsteps=100,
          plots=FALSE,
          verbose=TRUE)
```

---

twoarm_sim *Simulates data from a two-arm parallel cluster randomised trial*

Description

Simple simulation of two Poisson distributed outcomes for a two-arm parallel cluster randomised trial with no baseline measures. A log-linear model is specified y~Poisson(lambda) with lambda = exp(mu + beta*D + theta) where D is the treatment effect indicator equal to one in clusters with the treatment and zero otherwise, and theta~N(0,sigma^2) is the cluster random effect. Used for testing error rates of the methods.

Usage

```
twoarm_sim(
  nJ = c(7, 7),
  N = 20,
  mu = rep(1, 2),
```

```
  beta = c(0, 0),
  sig_cl = rep(0.05, 2)
)
```

Arguments

nJ	Vector of two integers with the number of clusters in treatment and control arms
N	Number of individuals per cluster
mu	Vector of two numeric values with the intercept terms for the two models on the log scale
beta	Vector of two numeric values that are the treatment effect parameters in the two models
sig_cl	Vector of two values equal to the variance of the random effect in each model

Value

A list consisting of: (1) data frame with the cluster IDs (cl), treatment effect indicators (treat), and two outcomes (y1, y2), and (2) the values of the treatment effect parameters used in the simulation.

Examples

```
out <- twoarm_sim()
data <- out[[1]]
```

Index