Package 'trio'

January 30, 2026

Type Package

Title Testing of SNPs and SNP Interactions in Case-Parent Trio Studies

Version 3.48.0

Date 2024-12-28

Author
Holger Schwender, Qing Li, Philipp Berger, Christoph Neumann, Margaret Taub, Ingo Ruczinski

Maintainer Holger Schwender <holger.schw@gmx.de>

Depends R (>= 3.0.1)

Suggests haplo.stats, mcbiopi, splines, logicFS (>= 1.28.1), KernSmooth, VariantAnnotation

Imports grDevices, graphics, methods, stats, survival, utils, siggenes, LogicReg (>= 1.6.1), data.table

Description Testing SNPs and SNP interactions with a genotypic TDT. This package furthermore contains functions for computing pairwise values of LD measures and for identifying LD blocks, as well as functions for setting up matched case pseudo-control genotype data for case-parent trios in order to run trio logic regression, for imputing missing genotypes in trios, for simulating case-parent trios with disease risk dependent on SNP interaction, and for power and sample size calculation in trio data.

License LGPL-2

biocViews SNP, GeneticVariability, Microarray, Genetics

git_url https://git.bioconductor.org/packages/trio

git_branch RELEASE_3_22

git_last_commit 2a3434d

git_last_commit_date 2025-10-29

Repository Bioconductor 3.22

Date/Publication 2026-01-29

Contents

allelicTDT 2
colEMlrt 4
colGxE 6
colGxGPerms 10

colNtrios . . . 11
colPOlrt . . . 13
colTDTmaxTest . . . 15
colTDTsam . . . 17
findLDblocks . . . 19
getLD . . . 21
getMatPseudo . . . 23
lrControl . . . 24
ntrios2Dom . . . 26
ped2geno . . . 27
plot.getLD . . . 28
plot.LDblocks . . . 30
plot.trioLR . . . 31
poly4root . . . 33
print.colGxE . . . 34
print.trioFS . . . 36
print.trioLR . . . 38
probTDT . . . 40
read.pedfile . . . 41
removeSNPs . . . 43
scoreTDT . . . 44
tdt . . . 46
tdtGxG . . . 48
trio.check . . . 51
trio.data . . . 53
trio.permTest . . . 54
trio.power . . . 56
trio.prepare . . . 58
trio.sim . . . 59
trioFS . . . 61
trioLR . . . 64
vcf2geno . . . 68

Index **70**

`allelicTDT` *Allelic TDT*

Description

Performs the allelic Transmission/Disequilibrium Test for each SNP contained in a genotype matrix.

Usage

```
allelicTDT(mat.snp, size = 50, correct = FALSE)

## S3 method for class 'aTDT'
print(x, top = 5, digits = 4, ...)
```

Arguments

mat.snp	a numeric matrix in which each column represents a SNP. Each column must be a numeric vector of length $3 * t$ representing a SNP genotyped at t trios. Each of the t blocks must consist of the genotypes of father, mother, and offspring (in this order). The genotypes must be coded by 0, 1, and 2. Missing values are allowed and need to be coded by NA. This matrix might be generated from a ped-file by, e.g., employing ped2geno.
size	the number of SNPs considered simultaneously when computing the parameter estimates.
correct	should the test statistic be continuity corrected? If FALSE, $(b-c)^2/(b+d)$ will be used as test statistic, where b and c are the off-diagonal elements of the 2x2-table summarizing the transmitted and not transmitted alleles from the heterozygous parents. If TRUE, $(\|b-c\|-1)^2/(b+d)$ will be used as test statistic.
x	an object of class aTDT, i.e. the output of allelicTDT.
digits	number of digits that should be printed.
top	number of interactions that should be printed. If top is less than or equal to zero, set to NA, or larger than the number of SNPs, then the statistics for all SNPs are printed in the order as they were in the genotype matrix used as input into colTDT. Otherwise, the top interactions with the smallest p-values are printed.
...	ignored.

Value

An object of class aTDT containing the following numeric vectors:

stat	values of the test statistic of the allelic TDT,
pval	the corresponding p-values.

Author(s)

Holger Schwender, <holger.schwender@udo.edu>

References

Spielman, R.S., McGinnis, R.E., and Ewens, W.J. (1993). Trsnmmission Test for Linkage Disequilibrium: The Insulin Gene Region and Insulin-Dependent Diabetes Mellitus (IDDM). *American Journal of Human Genetics*, 52, 506-516.

See Also

colTDT

Examples

```
# Load the simulated data for the analysis.
data(trio.data)

# Perform an allelic TDT
a.out <- allelicTDT(mat.test)

# By default, the top 5 SNPs are shown.
# Another number of SNPs, e.g., 10, are displayed by
```

```
print(a.out, top=10)

# If the results for all SNPs should be shown (or returned), use
print(a.out, top=0)
```

`colEMlrt` *EM Likelihood Ratio Test*

Description

Performs the Expectation Maximimization Likelihood Ratio Test (EMLRT) proposed by Weinberg (1999) for each SNP in a matrix in genotype format.

Usage

```
colEMlrt(mat.snp, model = c("general", "dominant", "recessive"), maternal = FALSE,
     parentMissing = c("father", "mother", "either"), iter = 40, eps = 10^-16)

## S3 method for class 'colEMlrt'
print(x, top = 5, digits = 4, ...)
```

Arguments

`mat.snp`	a numeric matrix in which each column represents a SNP. Each column must be a numeric vector of length $3 * t$ representing a SNP genotyped at t trios. Each of the t blocks must consist of the genotypes of father, mother, and offspring (in this order). The genotypes must be coded by 0, 1, and 2. Missing values are allowed and need to be coded by `NA`. This matrix in genotype format might be generated from a ped-file by, e.g., employing `ped2geno`.
`model`	a character string specifying the genetic effect that should be considered in the Poisson regression. By default, the general model proposed by Weinberg (1999) is fitted. Alternatively, a dominant or recessive mode of inheritance might be considered by setting `model = "dominant"` or `model = "recessive"`, respectively. Abbreviations such as `"dom"` or `"rec"` are also accepted.
`maternal`	logical specifying whether parameters for a maternal effects should be added to the Poisson regression model. If `TRUE` and `model = "general"`, the most general model described by Weinberg (1999) is used.
`parentMissing`	a character string specifying whether the genotype of the `father`, the `mother`, or `either` the mother or the father is allowed to be missing. By default, only the genotype of the father is allowed to be missing in a trio. Thus, in this case, all complete trios and all trios in which data are available for the father and the offspring are used in the testing of the considered SNP. If `parentMissing = "either"`, all trios in which the genotype of the offspring or the genotypes of both parents are missing for a particular SNP are ignored in the analysis of this SNP.
`iter`	a non-negative numeric value specifying the maximum number of iterations that should be used in the application of the expectation maximization algorithm.
`eps`	a non-negative small numeric value specifying the accuracy for the stopping criterion of the EM algorithm. If the sum of the squared differences of the estimated expected numbers of trios over the 15 possible genotype combinations

	in trios between two consecutive iterations of the EM algorithm is smaller than `eps`, the EM algorithm stops. Often, less than ten genotype combinations are required.
`x`	an object of class `colEMlrt`, i.e. the output of `colEMlrt`.
`digits`	number of digits that should be printed.
`top`	number of SNPs that should be printed. If `top` is less than or equal to zero or larger than the total number of SNPs in `mat.snp`, then the statistics for all SNPs are printed in the order as they were in the genotype matrix used as input into `colEMlrt`. Otherwise, the `top` interactions with the smallest p-values are printed.
`...`	ignored.

Details

While in functions such as `colTDT` all trios in which the genotype of one or more of the members of this trio is missing for a particular SNP are removed from the analysis of this SNP, the procedure proposed by Weinberg (1999) can handle missing genotypes by employing an expectation maximization (EM) algorithm to estimate the expected numbers of trios for the 15 different genotype combinations possible in a trio (when considering the genotypes of mothers and fathers individually) and a likelihood ratio test based on two nested Poisson regression models using the estimated expected numbers of trios as outcome.

Value

An object of class `colEMlrt` consisting of the following numeric vectors:

`stat`	the values of the test statistic of the likelihood ratio test for all SNPs in `mat.snp`,
`pval`	the corresponding p-values,
`ll.full`	the values of the maximized likelihoods of the full models,
`ll.red`	the values of the maximied likelihoods of the reduced models not containing the parameter(s) corresponding to the genetic `model` of interest.

Author(s)

Philipp Berger, <philipp.berger@hhu.de>

References

Weinberg, C.R. (1999). Allowing for Missing Parents in Genetic Studies of Case-Parent Triads. *American Journal of Human Genetics*, 64, 1186-1193.

See Also

`colTDT`, `ped2geno`

Examples

```
# Load the simulated data.
data(trio.data)

# The EM Likelihood Ratio Test can be applied
# to the SNPs in mat.test by
```

```
em.out <- colEMlrt(mat.test)

# If a dominant mode of inheritance is of interest,
# the corresponding EM Likelihood Ratio Test can be
# performed by
emd.out <- colEMlrt(mat.test, model="dominant")

# By default, statistics for the top 5 SNPs are displayed.
# If another number of SNPs, say 10, should be displayed,
# then this can be done by
print(em.out, top = 10)

# The statistics for all SNPs (not ordered by their
# significance) can be obtained by
print(em.out, top = 0)
```

---

colGxE *Genotypic TDT for Gene-Environment Interactions*

Description

Performs a genotypic TDT for gene-environment interactions for each SNP represented by a column of a matrix in genotype format and a binary environmental factor. If `alpha1` is set to a value smaller than 1, then the two-step procedure of Gauderman et al. (2010) will be used to first select all SNPs showing a p-value smaller than `alpha1` in a logistic regression of the environmental factor against the sums of the codings for the parents' genotypes at the respective SNP. In the second step, the genotypic TDT is then applied to the selected SNPs.

If `unstructured = TRUE`, all fully parameterized model is considered and a likelihood ratio test is performed.

While `colGxE` computes the p-values based on asymptotic ChiSquare-distributions, `colGxEPerms` can be used to determine permutation-based p-values for the basic genotypic TDT (i.e. for `colGxE` using `alpha = 1` and `unstructured = FALSE`.

Usage

```
colGxE(mat.snp, env = NULL, listNumber = NULL,
   model = c("additive", "dominant", "recessive"), alpha1 = 1, size = 50,
   addGandE = TRUE, whichLRT = c("both", "2df", "1df", "none"),
   add2df = TRUE, addCov = FALSE, famid = NULL, unstructured = FALSE,
   step1Stats = c("alpha", "add", "only"))

colGxEPerms(mat.snp, env, model = c("additive", "dominant", "recessive"),
   B = 10000, size = 20, addPerms = TRUE, famid = NULL, rand = NA)
```

Arguments

`mat.snp`	a numeric matrix in which each column represents a SNP. Each column must be a numeric vector of length $3 * t$ representing a SNP genotyped at t trios. Each of the t blocks must consist of the genotypes of father, mother, and offspring (in this order). The genotypes must be coded by 0, 1, and 2. Missing values are allowed and need to be coded by `NA`. This matrix might be generated from a ped-file by, e.g., employing ped2geno.

`env` a vector of length t (see `mat.snp`) containing for each offspring the value of a binary environmental variable, which must take the values 0 and 1. Ignored if `listNumber` is specified.

`listNumber` a list consisting of two numeric matrices that contain, on the one hand, for trios affected by a (binary) environmental variable, and on the other hand, for trios not being affected by this environmental variable, the numbers of trios showing the different combinations of genotypes of the offsprings and their parents, i.e. the output of `colNtrios` with specified argument `env`. Each column of the matrices in `listNumber` corresponds to a genotype combination, and each row to a SNP.

`model` type of model that should be fitted. Abbreviations are allowed. Thus, e.g., `model = "dom"` will fit a dominant model, and `model = "r"` an recessive model.

`alpha1` a numeric value between 0 and 1 (excluding 0). If `alpha1 = 1`, all SNPs will be tested with a genotypic TDT. Otherwise, the two-step procedure of Gauderman et al. (2010) will be used to select all SNPs showing a p-value smaller than or equal to `alpha1` in a logistic regression in which the environmental factor is used as response and the sums over the codings for the genotypes of the parents are employed as predictor. The genotypic TDT will then be applied to the selected SNPs. Since a logistic regression is employed in the first step, which requires a numerical determination of the parameter estimates, the two-step procedure will not lead to a reduction in computing time, but will increase the computing time.

`size` the number of SNPs considered simultaneously when computing the parameter estimates.

`addGandE` should the relative risks and their confidence intervals for the exposed cases be added to the output?

`whichLRT` character string specifying which likelihood ratio test should be added to the output. If `"2df"`, 2 degree of freedom likelihood ratio tests comparing the fitted models (containing one parameter for the SNP and one for the gene-environment interaction) with models containing no factor will be performed. If `"1df"`, one degree of freedom likelihood ratio tests comparing the fitted model (containing two parameters, one for the SNP and the other for the interaction) with models only containing the respective SNP will be added to the output. If `"both"` (default), both tests will be performed, whereas none test will be done, if `whichLRT = "none"`.

`add2df` should the results of a 2 df Wald test for testing both the SNP and the interaction effect simultaneously be added to the model?

`addCov` should the covariance between the parameter estimations for the SNP and the gene-environment interaction be added to the output? Default is `addCov = FALSE`, as this covariance is given by the negative variance of the parameter estimate for the SNP.

`famid` a vector of the same length as `env` specifying the family IDs for the corresponding values of the environmental variable in `env`. Can be used to reorder the vector `env` when the order of the trios differs between `env` and `mat.snp`.

`unstructured` should a fully parameterized model be fitted? If `TRUE`, a 2 df likelihood ratio test is performed comparing a gTDT model containing one indicator variable for the heterozygous genotype and one for the homozygous variant genotype with a gTDT model additionally containing two terms for the interactions between these variables and the environmental factor. In this case, only the arguments `mat.snp`, `env`, and `famid` are considered.

`step1Stats` character string specifying when the statistics of the first step in the two-step procedure of Gauderman et al. (2010) are returned. If set to `"alpha"` (default),

a matrix called `statsStep1` containing the parameter estimates, their standard errors, the values of the test statistic, and the corresponding p-values of this first step procedure is only added to the output, if `alpha1` < 1. If set to `"add"`, `statsStep1` is always added to the output. This can be used if no preselection based on the first step of the Gauderman procedure should be done, i.e. if `alpha1` = 1. If set to `"only"`, only the first step statistics of the Gauderman procedure will be computed (and no further test are performed). In this case, only `statsStep1` is returned. If set to `"add"` or `"only"`, the specification of `alpha1` is ignored.

`B` number of permutations.

`addPerms` should the matrices containing the permuted values of the test statistics for the SNP and the gene-environment interaction be added to the output?

`rand` integer for setting the random number generator into a reproducible state.

Details

A conditional logistic regression model including two parameters, one for G, and the other for GxE, is fitted, where G is specified according to `model`.

Value

If `step1Stats` is set to `"only"`, only a matrix containing the parameter estimates, their standard errors, the values of the test statistic, and the corresponding p-values of the first step of the two-step procedure of Gauderman et al. (2010) is returned.

Otherwise, if `unstructured` = `FALSE`, `colGxE` returns an object of class `colGxE` consisting of the following numeric matrices with two columns (one for each parameter):

`coef` the estimated parameter,

`se` the estimated standard deviation of the parameter estimate,

`stat` Wald statistic,

`RR` the relative risk, i.e.\ in the case of trio data, `exp(coef)` (see Schaid, 1996),

`lowerRR` the lower bound of the 95% confidence interval for `RR`,

`upperRR` the upper bound of the 95% confidence interval for `RR`,

`usedTrios` the number of trios affecting the parameter estimation,

`env` vector containing the values of the environmental factor,

`type` `model`,

`addGandE` the value of `addGandE`,

`addOther` a logical vector specifying which of the likelihood ratio tests and if the 2 df Wald test was performed,

and depending on the specifications in `colGxE`

`cov` numeric vector containing the covariances,

`lrt2df` a numeric matrix with two columns, in which the first column contains the values of the 1 df likelihood ratio test statistic and the second the corresponding p-values,

`wald2df` a numeric matrix with two columns, in which the first column contains the values of the 2 df Wald test statistics and the second the corresponding p-values,

`lrt1df` a numeric matrix with two columns, in which the first column contains the values of the 2 df likelihood ratio test statistic and the seocnd the corresponding p-values.

For `colGxE` with `unstructured=TRUE`, an object of class `colGxEunstruct` consisting of the following vectors:

`ll.main` the loglikelihoods of the models containing only the two main effects,

`ll.full` the loglikelihoods of the models additionally containing the two main effects and the two interaction effects,

`stat` the values of the test statistic of the likelihood ratio test,

`pval` the corresponding p-values.

For `colGxEPerms`,

`stat` a matrix with two columns containing the values of gTDT statistics for the main effects of the SNPs and the gene-environment interactions when considering the original, unpermuted case-pseudo-control status,

`pval` a matrix with two columns comprising the permutation-based p-values corresponding to the test statistics in `stat`,

and if `addPerms = TRUE`

`matPermG` a matrix with `B` columns containing the values of the gTDT statistic for the SNPs when considering the `B` permutations of the case-pseudo-control status,

`matPermGxE` a matrix with `B` columns containing the values of the gTDT statistic for the gene-environment interactions when considering the `B` permutations of the case-pseudo-control status.

Author(s)

Holger Schwender, <holger.schwender@udo.edu>

References

Gauderman, W.J., Thomas, D.C., Murcray, C.E., Conti, D., Li, D., and Lewinger, J.P. (2010). Efficient Genome-Wide Association Testing of Gene-Environment Interaction in Case-Parent Trios. *American Journal of Epidemiology*, 172, 116-122.

Schaid, D.J. (1996). General Score Tests for Associations of Genetic Markers with Disease Using Cases and Their Parents. *Genetic Epidemiology*, 13, 423-449.

Schwender, H., Taub, M.A., Beaty, T.H., Marazita, M.L., and Ruczinski, I. (2011). Rapid Testing of SNPs and Gene-Environment Interactions in Case-Parent Trio Data Based on Exact Analytic Parameter Estimation. *Biometrics*, 68, 766-773.

See Also

`colTDT`, `ped2geno`, `colNtrios`

Examples

```
# Load the simulated data for the analysis.
data(trio.data)

# Set up a vector with the binary environmental variable.
# Here, we consider the gene-gender interactions and
# assume that the children in the first 50 trios are
# girls, and the remaining 50 are boys.
sex <- rep(0:1, each = 50)

# Test the interaction of sex with each of the SNPs in mat.test
gxe.out <- colGxE(mat.test, sex)

# By default, an additive mode of inheritance is considered.
# If, e.g., a dominant mode should be considered, then this can
# be done by calling
gxeDom.out <- colGxE(mat.test, sex, model="dominant")
```

`colGxGPerms` *Permutation-Based gTDT for Two-Way Interactions*

Description

Computes the original and permuted values of the test statistic of the gTDT test as proposed by Cordell (2002) for each interaction between the pairs of SNPs in mat.snp.

Usage

```
colGxGPerms(mat.snp, n.perm = 1000, genes = NULL, col.out = NULL,
   warnError = TRUE, verbose = TRUE, rand = NA)
```

Arguments

`mat.snp`	a numeric matrix in which each column represents a SNP. Each column must be a numeric vector of length $3 * t$ representing a SNP genotyped at t trios. Each of the t blocks must consist of the genotypes of father, mother, and offspring (in this order). The genotypes must be coded by 0, 1, and 2. Missing values are allowed and need to be coded by `NA`. This matrix might be generated from a ped-file by, e.g., employing `ped2geno`.
`n.perm`	number of permutations of the response for which the permuted values of the test statistic should be computed.
`genes`	a character vector containing the names of the genes to which the SNPs belong. If specified, only the two-way interactions between SNPs from different genes are tested. If `NULL`, all two-way interactions between all possible pairs of SNPs are tested.
`col.out`	the output of `colGxG` with `epistatic = TRUE` (which is the default in `colGxG`). If `NULL`, `compPermTDT2way` computes the values of the test statistic for the original permutation of the response.
`warnError`	logical indicating whether the statistics for the gTDT should be returned as `NA` if the fitting of the conditional logistic regression model fails. This might in particular happen when the two considered SNPs are in (strong) LD.

verbose	logical indicating whether some information on what is currently computed should be printed.
rand	numeric value. If specified, the random number generator is set into a reproducible state.

Value

A list consisting of

stat	a numeric vector containing the original values of the test statistic,
permStat	a numeric matrix containing the permuted values of the test statistic,
y.perm	a matrix containing the permutations of the response.

Author(s)

Holger Schwender, <holger.schwender@udo.edu>

References

Cordell, H. J. (2002). Epistasis: What it Means, what it Doesn't mean, and Statistical Methods to Detect it in Humans. Human Molecular Genetics, 11, 2463-2468.

See Also

colGxG

Examples

```
# Load the simulated data.
data(trio.data)

# Cordell's LRT for all pairs of SNPs in mat.test can be performed
# and the values of the LRT statistic for 10 permutations of the
# case-pseudo-controls status can be computed by
gxg <- colGxGPerms(mat.test, n.perm = 10)

# where we here consider only 10 permutations to keep the computing
# time of this example small. Usually, at least a few thousand
# permutations should be considered.
```

---

colNtrios *Frequency of Genotype Combinations*

Description

Computes the numbers of trios showing the ten different genotype combinations for each SNP in a matrix in genotype format.

Usage

```
colNtrios(mat.snp, env = NULL, onlyContributing = FALSE, famid = NULL,
   size = 50)
```

Arguments

mat.snp	a numeric matrix in which each column represents a SNP. Each column must be a numeric vector of length $3 * t$ representing a SNP genotyped at t trios. Each of the t blocks must consist of the genotypes of father, mother, and offspring (in this order). The genotypes must be coded by 0, 1, and 2. Missing values are allowed and need to be coded by NA. This matrix in genotype format might be generated from a ped file by, e.g., employing ped2geno.
env	a vector of length t (see mat.snp) containing for each offspring the value of a binary environmental variable, which must take the values 0 and 1. If specified the numbers of trios showing the ten different genotype combinations are computed separately for the trios for which env = 0 and for the trios for which env = 1.
onlyContributing	if TRUE, only the seven numbers of trios that contribute to the likelihood estimation in the performance of the genotypic TDT, e.g., with colTDT, are determined. If FALSE (which is the default), all ten numbers of trios for the ten different genotype combinations are computed.
famid	a vector of the same length as env specifying the family IDs for the corresponding values of the environmental variable in env. Can be used to reorder the vector env when the order of the trios differs between env and mat.snp. Ignored if env is not specified.
size	the number of SNPs considered simultaneously when computing the parameter estimates. Ignored if fast = FALSE.

Details

When considering SNPs coded by the numbers of minor alleles so that the genotypes of the SNPs are coded by 0, 1, and 2, there exist ten possible combinations of the genotypes of an offspring and its parents that this trio can show (see Table 1 in Schwender et al., 2011). Seven of these genotype combinations contribute to the likelihood estimation considered when performing a genotypic TDT.

Depending on the specification of the argument onlyContributing, colNtrios computes for each SNP in mat.snp the numbers of trios showing the ten or seven different genotype combinations, respectively.

In colNtrios, an additive mode of inheritance is considered. If the numbers of trios showing the different genotype combinations that exist when considering a dominant or recessive mode of inheritance are of interest (see, e.g., Table 2 or 3, respectively, in Schwender et al., 2011), then ntrios2Dom or ntrios2Rec, respectively, can be applied to the output of colNtrios.

Value

If env is not specified, a matrix with the numbers of trios for the different genotype combinations, where the names of the columns corresponding to the different genotype combinations are given by an starting "G" for genotype followed by the genotypes of the parents and the offspring (as in Table 1 of Schwender et al., 2011).

If env is specified, a list consisting of two matrices as the one described above, one matrix for env = 0, and the other for env = 1.

Author(s)

Holger Schwender, <holger.schwender@udo.edu>

References

Schwender, H., Taub, M.A., Beaty, T.H., Marazita, M.L., and Ruczinski, I. (2011). Rapid Testing of SNPs and Gene-Environment Interactions in Case-Parent Trio Data Based on Exact Analytic Parameter Estimation. *Biometrics*, 68, 766-773.

See Also

colTDT, ntrios2Dom

Examples

```
# Load the simulated data.
data(trio.data)

# Compute the numbers of trios.
mat.ntrios <- colNtrios(mat.test)

# A genotypic TDT can then be applied for each SNP
# in mat.test by
colTDT(matNumber = mat.ntrios)

# This leads to the same results as
colTDT(mat.test)

# If env is also specified, e.g., by
facEnv <- rep(0:1, each = 50)
list.ntriosEnv <- colNtrios(mat.test, env = facEnv)

# then a genotypic TDT for gene-environment interactions
# can be performed by
gxe.Nout <- colGxE(listNumber = list.ntriosEnv,
    env = facEnv)
```

---

colPOlrt *Parent-of-Origin Tests*

Description

Computes the test statistics and the corresponding p-values either for the Parent-of-Origin Likelihood Ratio Test proposed by Weinberg (1999) or the Transmission Asymmetry Test proposed by Weinberg et al. (1998).

Usage

```
colPOlrt(mat.snp, size = 20)

colTAT(mat.snp, stratified = FALSE, size = 50, bothHet = 0)

## S3 method for class 'polrt'
print(x, top = 5, digits = 4, ...)

## S3 method for class 'tat'
print(x, top = 5, digits = 4, ...)
```

Arguments

mat.snp	a numeric matrix in which each column represents a SNP. Each column must be a numeric vector of length $3 * t$ representing a SNP genotyped at t trios. Each of the t blocks must consist of the genotypes of father, mother, and offspring (in this order). The genotypes must be coded by 0, 1, and 2. Missing values are allowed and need to be coded by NA. This matrix might be generated from a ped-file by, e.g., employing ped2geno.
size	the number of SNPs considered simultaneously when computing the test statistics.
stratified	should also test statistics and p-values stratified by paternal and maternal transmission be computed?
bothHet	a numeric value between 0 and 1 specifying how trios in which both parents are heterozygous are weighted in determination of the TAT statistic. By default, such trios are ignored (as proposed by Weinberg, 1999). If bothHet = 1, such trios are treated in the same way as trios with one heterozygous parent. Other values (e.g., bothHet = 0.5) are also sometimes used for bothHet.
x	an object of class polrt or tat, i.e. the output of colPOlrt or colTAT, respectively.
digits	number of digits that should be printed.
top	number of interactions that should be printed. If top is less than or equal to zero, set to NA, or larger than the number of SNPs, then the statistics for all SNPs are printed in the order as they were in the genotype matrix used as input into colTDT. Otherwise, the top interactions with the smallest p-values are printed.
...	ignored.

Value

For colPOlrt, an object of class polrt consisting of the following numeric vectors:

stat	the values of the test statistic of the likelihood ratio test for all SNPs in mat.snp,
pval	the corresponding p-values,
full	the values of the maximized likelihoods of the full models containing also a parameter for the parent-of-origin effect,
red	the values of the maximied likelihoods of the reduced models not containing this parameter.

For colTAT, an object of class tat consisting of the following numeric vectors:

stat	the values of the test statistic of transmission asymmetry test for all SNPs in mat.snp,
pval	the corresponding p-values,
usedTrios	the number of trios affecting the determination of the TAT statistic,

and if stratified = TRUE

matStrat	a matrix with four columns containing the number of minor alleles transmitted and not-transmitted by heterozygous fathers and mothers,
statPaternal	a numeric vector containing for each SNP the value of the test statistic for testing whether the numbers of paternal transmissions and non-transmissions differ,

pvalPaternal	the p-values corresponding to statPaternal,
statMaternal	a numeric vector containing for each SNP the value of the test statistic for testing whether the numbers of maternal transmissions and non-transmissions differ,
pvalMaternal	the p-values corresponding to statMaternal.

Author(s)

Holger Schwender, <holger.schwender@udo.edu>

References

Weinberg, C.R.,Wilcox, A.J., and Lie, R.T. (1998). A Log-Linear Approach to Case-Parent-??Triad Data: Assessing Effects of Disease Genes that act Either Directly or Through Maternal Effects and that may be Subject to Parental Imprinting. *American Journal of Human Genetics*, 62, 969-978.

Weinberg, C.R. (1999). Methods for Detection of Parent-of-Origin Effects in Genetic Studies of Case-Parents Triads. *American Journal of Human Genetics*, 65, 229-235.

See Also

colTDT, ped2geno

Examples

```
# Load the simulated data.
data(trio.data)

# The Parent-of-Origin Likelihood Ratio Test can be applied
# to the SNPs in mat.test by
po.out <- colPOlrt(mat.test)

# The Transmission Asymmetry Test can be applid to the SNPs
# in mat.test by
tat.out <- colTAT(mat.test)

# By default, statistics for the top 5 SNPs are displayed.
# If another number of SNPs, say 10, should be displayed,
# then this can be done by
print(po.out, top = 10)

# The statistics for all SNPs (not ordered by their
# significance) can be obtained by
print(po.out, top = 0)
```

---

colTDTmaxTest *Maximum Genotypic TDT*

Description

Computes the maximum over the gTDT statistics for an additive, dominant, and recessive model. colTDTmaxTest additionally computes permutation-based p-values.

Usage

```
colTDTmaxTest(geno, perm = 10000, size = 50, chunk = 10000,
    minimum = 0.001, verbose = FALSE)
colTDTmaxStat(geno, size = 50)

## S3 method for class 'maxTestTrio'
print(x, top = 5, digits = 4, ...)
## S3 method for class 'maxStatTrio'
print(x, top = 5, digits = 4, ...)
```

Arguments

geno	a numeric matrix in which each column represents a SNP. Each column must be a numeric vector of length $3 * t$ representing a SNP genotyped at t trios. Each of the t blocks must consist of the genotypes of father, mother, and offspring (in this order). The genotypes must be coded by 0, 1, and 2. Missing values are allowed and need to be coded by NA. This matrix might be generated from a ped-file by, e.g., employing ped2geno.
perm	number of permutations of the response for which the permuted values of the test statistic should be computed.
size	number of SNPs that should be considered simultaneously when estimating the parameter.
chunk	number of permutations that should be considered simultaneously in the computation of the p-values
minimum	minimum value that a test statistic must show that for the corresponding SNP the p-value is computed.
verbose	logical indicating whether some information on what is currently computed should be printed.
x	an object of class maxTestTrio or maxTestStat, i.e. the output of colTDTmaxTest or of colTDTmaxStat.
digits	number of digits that should be printed.
top	number of interactions that should be printed. If the number of interactions is smaller than or equal to top, then the statistics for all interactions are printed in the order of their computation. Otherwise, they the top Top interactions are printed.
...	ignored.

Value

For colTDTmaxStat, an object of class maxStatTrio consisting of a vector stat containing the values of the Max statistic for the SNPs in geno, a matrix max.stat containing the values of the gTDT statistic for testing an additive, a dominant, and a recessive effect, and additional information required by colTDTmaxTest.

For colTDTmaxTest, an object of class maxTestTrio consisting of stat, max.stat, and the unadjusted p-values pval corresponding to stat.

Author(s)

Holger Schwender, <holger.schwender@udo.edu>

References

Schwender, H., Taub, M.A., Beaty, T.H., Marazita, M.L., and Ruczinski, I. (2011). Rapid Testing of SNPs and Gene-Environment Interactions in Case-Parent Trio Data Based on Exact Analytic Parameter Estimation. *Biometrics*, 68, 766-773.

See Also

tdt

Examples

```
# Load the simulated data.
data(trio.data)

# Perform a MAX test by only computing the MAX statistics.
max.out <- colTDTmaxStat(mat.test)

# Permutation-based p-values are additionally computed when using
max.out2 <- colTDTmaxTest(mat.test)
```

---

colTDTsam *SAM and EBAM for Trio Data*

Description

Performs a Significance Analysis of Microarrays (SAM; Tusher et al., 2001) or an Empirical Bayes Analysis of Microarrays (EBAM; Efron et al., 2001), respectively, based on the genotypic transmission/disequilibrium test statistic.

Usage

```
colTDTsam(mat.snp, model = c("additive", "dominant", "recessive", "max"),
   approx = NULL, B = 1000, size = 10, chunk = 100, rand = NA)

colTDTebam(mat.snp, model = c("additive", "dominant", "recessive", "max"),
   approx = NULL, B = 1000, size = 10, chunk = 100,
   n.interval = NULL, df.ratio = 3, df.dens = 3, knots.mode = TRUE,
   type.nclass = c("wand", "FD", "scott"), fast = FALSE, rand = NA)
```

Arguments

mat.snp	a matrix in genotype format, i.e. a numeric matrix in which each column is a vector of length $3 * t$ representing a SNP genotyped at t trios. Each of the t blocks of rows in mat.snp must consist of the genotypes of father, mother, and offspring (in this order), where the genotypes must be coded by 0, 1, and 2. Missing values are allowed and need to be coded by NA. This matrix might be generated from a data frame in ped format by, e.g., employing ped2geno.
model	type of genetic mode of inheritance that should be considered. Either "additive" (default), "dominant", "recessive", or "max". If model = "max", the maximum over the gTDT statistics for testing an additive, dominant, and recessive model is used as gTDT statistic. Abbreviations are allowed. Thus, e.g., model = "dom" will fit a dominant model, and model = "r" an recessive model.

approx	logical specifying whether the null distribution should be approximated by a χ^2-distribution with one degree of frredom. If approx = FALSE, the null distribution is estimated based on a permutation method. If not specified, i.e. NULL, approx is set to TRUE, when an additive, dominant, or recessive mode of inheritance is considered, and approx = FALSE, when model = "max". If model = "max", it is not allowed to set approx = TRUE.
B	number of permutations used in the estimation of the null distribution, and thus, the computation of the null statistics. Ignored if approx = TRUE.
size	number of SNPs considered simultaneously when computing the gTDT statistics.
chunk	number of permutations considered simultaneously in the permutation procedure.
n.interval	the number of intervals used in the logistic regression with repeated observations for estimating the ratio of the null density to the density of the observed gTDT values in an EBAM analysis (if approx = FALSE), or in the Poisson regression used to estimate the density of the observed gTDT values (if approx = TRUE). For details, see Efron et al., 2001, or Schwender and Ickstadt, 2008, respectively. If NULL, n.interval is determined by the maximum of 139 (see Efron et al., 2001) and the number of intervals estimated by the method specified by type.nclass.
df.ratio	integer specifying the degrees of freedom of the natural cubic spline used in the logistic regression with repeated observations for estimating the ratio of the null density to the density of the observed gTDT values in an EBAM analysis. Only used when approx is set to FALSE.
df.dens	integer specifying the degrees of freedom of the natural cubic spline used in the Poisson regression to estimate the density of the observed gTDT values in an EBAM analysis. Only used when approx is set to TRUE.
knots.mode	logical specifying whether the df.dens - 1 knots of the natural cubic spline are centered around the mode and not the median of the density when fitting the Poisson regression model to estimate the density of the observed gTDT values in an EBAM analysis. Only used when approx is set to TRUE. For details on this density estimation, see denspr.
type.nclass	character string specifying the procedure used to estimate the number of intervals of the histogram used in the logistic regression with repeated observations or the Poisson regression, respectively (see n.interval). Can be either "wand" (default), "FD", or "scott". Ignored if n.interval is specified. For details, see denspr.
fast	logical specifying whether a crude estimate for the number of permuted test scores larger than the respective observed gTDT value should be used. If FALSE, the exact number of permuted test scores larger than the respective observed gTDT value is computed.
rand	numeric value. If specified, i.e. not NA, the random number generator will be set into a reproducible state.

Value

The output of colTDTsam or colTDTebam is an object of class SAM or EBAM, respectively. All the features implemented in the R package siggenes for an SAM or EBAM analysis, respectively, can therefore be used in the SAM or EBAM analysis of case-parent trio data implemented in colTDTsam or colTDTebam, respectively. For details, see sam or ebam, respectively.

Author(s)

Holger Schwender, <holger.schwender@udo.edu>

References

Efron, B., Tibshirani, R., Storey, J.D., and Tusher, V. (2001). Empirical Bayes Analysis of a Microarray Experiment, *Journal of the American Statistical Association*, 96, 1151-1160.

Schwender, H. and Ickstadt, K. (2008). Empirical Bayes Analysis of Single Nucleotide Polymorphisms. *BMC Bioinformatics*, 9, 144.

Schwender, H., Taub, M.A., Beaty, T.H., Marazita, M.L., and Ruczinski, I. (2011). Rapid Testing of SNPs and Gene-Environment Interactions in Case-Parent Trio Data Based on Exact Analytic Parameter Estimation. *Biometrics*, 68, 766-773.

Tusher, V.G., Tibshirani, R., and Chu, G. (2001). Significance Analysis of Microarrays Applied to the Ionizing Radiation Response. *Proceedings of the National Academy of Science of the United States of America*, 98, 5116-5121.

See Also

colTDT, colTDTmaxStat, sam, ebam, SAM-class, EBAM-class

Examples

```
# Load the simulated data.
data(trio.data)

# Perform a Significance Analysis of Microarrays (SAM).
sam.out <- colTDTsam(mat.test)

# By default an additive mode of inheritance is considered.
# If another mode, e.g., the dominant mode, should be
# considered, then this can be done by
samDom.out <- colTDTsam(mat.test, model="dominant")

# Analogously, an Empirical Bayes Analysis of Microarrays based
# on the genotypic TDT can be performed by
ebam.out <- colTDTebam(mat.test)
```

---

findLDblocks *Identifying LD blocks*

Description

Finds LD blocks using the procedure proposed by Gabriel et al. (2002).

Usage

```
findLDblocks(x, alpha = 0.1, ciLD = c(0.7, 0.98), cuRecomb = 0.9,
   ratio = 9, alsoOthers = FALSE, parentsOnly = FALSE, iter = 50,
   snp.in.col = TRUE)

splitBlocks(blocks)
```

Arguments

x	either the output of getLD or getLDlarge, respectively, or a numeric matrix consisting of the integers 0, 1, and 2, where these integers are assumed to be the number of minor alleles that the respective SNPs shows at the respective subject. Missing values are allowed. By default, each column of this matrix represents a SNP, and each row a subject (for details, see snp.in.col). The SNPs must be ordered by their position on the considered chromosome.
alpha	numeric value between 0 and 1. For each pair of SNPs, a two-sided 100 * (1 - alpha)% confidence interval of D' is computed, and used to specify pairs of SNPs that are either in strong LD, or show historical evidence of recombination (see ciLD and cuRecomb). All SNP pairs not falling into these two categories are specified as 'Others'.
ciLD	numeric vector consisting of two values between 0 and 1. If the lower bound of the confidence interval of D' for a SNP pair is larger than or equal to the first value in ciLD and the upper bound is larger than or equal to the second value, then this pair of SNP is considered to be in strong LD.
cuRecomb	numeric value between 0 and 1. If the upper bound of the confidence interval of D' for a SNP pair is smaller than cuRecomb, then this pair of SNP is considered to show evidence of recombination.
ratio	numeric value larger than 1. If in a block of SNPs, the ratio of the number of SNP pairs being in strong LD to the number of SNPs showing evidence of recombination is larger than or equal to ratio, then this block will be identified as an LD-block. (Note that Gabriel et al. (2002) use ratio = 19 instead of ratio = 9.) Overlapping blocks are avoided by employing the approach described in Wall and Pritchard (2003).
alsoOthers	logical value. Following the description of Wall and Pritchard (2003) the endmarkers of a LD block must be in strong LD. By default (i.e.\ if alsoOthers = FALSE), this condition is used. If alsoOthers = TRUE, the endmarkers can also be categorized as 'Others'.
parentsOnly	logical indicating whether only the genotypes of the parents, i.e.\ rows 1, 2, 4, 5, ... of x, should be used in the computation of the LD measures when x is in genotype format and contains case-parent trio data (see ped2geno and read.pedfile). If FALSE (default), all rows are used in the determination of the pairwise LD measure. Ignored if x is the output of getLD or getLDlarge.
iter	integer specifying the number of iterations used in the computation of D (for details, see getLD). Ignored if x is the output of getLD.
snp.in.col	logical specifying whether each column of x represents a SNP (and each row a subject). If FALSE, each row represents a SNP (and each column a subject). Ignored if x is the output of getLD or getLDlarge.
blocks	output of findLDblocks. See Details.

Details

The LD-blocks are estimated using the method of Gabriel et al. (2002) as described in Wall and Pritchard (2003), where we use the approximate variance estimates of D' proposed by Zabaleta et al. (1997).

Since in trio.prepare the LD blocks are restricted to a maximum of 7 SNPs, splitBlocks can be used to split LD blocks composed of more than 7 SNPs into smaller blocks, if the output of findLDblocks should be used in trio.prepare to prepare a matrix for a trioLR or trioFS analysis.

Value

An object of class LDblocks consisting of

ld	the output of getLD,
blocks	a vector specifying which SNP belongs to which LD-block,
vec.blocks	a list in which each entry contains the names of the SNPs belonging to a specific LD-block,
param	a list of the input parameters.

Author(s)

Holger Schwender, <holger.schwender@udo.edu>

References

Gabriel, S.B. et al. (2002). The Structure of Haplotype Blocks in the Human Genome. *Science*, 296, 2225-2229.

Wall, J.D. and Pritchard J.K. (2003). Assessing the Performance of the Haplotype Block Model of Linkage Disequilibrium. *American Journal of Human Genetics*, 73, 502-515.

Zapata, C., Alvarez, G., and Carollo, C. (1997). Approximate Variance of the Standardized Measure of Gametic Disequilibrium D'. *American Journal of Human Genetics*, 61, 771-774.

See Also

plot.LDblocks, getLD

Examples

```
# Load the simulated data.
data(trio.data)

# Estimate LD blocks.
blocks <- findLDblocks(LDdata)

# Alternatively, the LD blocks can be estimated by
ld.out <- getLD(LDdata, addVarN=TRUE)
blocks2 <- findLDblocks(ld.out)
```

---

getLD *Computation of LD Measures*

Description

While getLD computes the value of D' and r^2 for each pair of SNPs in a matrix, getLDlarge determines D' and r^2 between each SNP and a user-specified number of SNPs closest to the SNP on the corresponding chromosome. Thus, getLDlarge can be applied to much more SNPs than getLD.

Usage

```
getLD(x, which = c("both", "rSquare", "Dprime"), parentsOnly = FALSE,
   iter = 50, snp.in.col = TRUE, asMatrix = FALSE, addVarN = FALSE)

getLDlarge(x, neighbors=25, which=c("both", "rSquare", "Dprime"),
      parentsOnly=FALSE, iter=50, snp.in.col=TRUE, addVarN=FALSE)
```

Arguments

x	a numeric matrix consisting of 0, 1, and 2, where it is assumed that the values represent the numbers of minor alleles that the SNPs show. Missing values are allowed. By default, each column represents a SNP and each row a subject. This can be changed by setting snp.in.col = FALSE. It is assumed that the SNPs are ordered by their position on the considered chromosome.
neighbors	positive integer specifying the number of neighbors of a SNP (in both directions) on a chromosome for which D' or r^2 should be computed. Thus, for each SNP (except for the SNPs in the first and last neighbors columns of x), 2 * neighbors r^2 or D' values are computed.
which	which LD measures should be computed? Either "rSquare", or "Dprime", or the values of "both" measures are computed. The latter is the default.
parentsOnly	logical indicating whether only the genotypes of the parents, i.e.\ rows 1, 2, 4, 5, ... of x, should be used in the computation of the LD measures when x is in genotype format and contains case-parent trio data (see ped2geno and read.pedfile). If FALSE (default), all rows are used in the determination of the pairwise LD measure.
iter	integer specifying how many iterations are used in the procedure of Hill (1974) which is used to estimate D.
snp.in.col	logical indicating whether each column of x represents a SNP (and each row a subject). If FALSE, each row represents a SNP (and each column a subject).
asMatrix	logical indicating whether the LD values are returned as a m x m matrix, where m is the number of SNPs. If FALSE, the LD values are returned as a vector of length $m * (m-1)/2$.
addVarN	logical indicating whether for each pair of SNPs the number of non-missing values and the variance estimates of D' proposed by Zabaleta et al. (1997) should be added to the output. The variance estimates are required for the identification of LD-blocks with findLDblocks.

Value

An object of class getLD or getLDlarge consisting (depending of the specification of which) the D' (Dprime) or r^2 (rSquare) values for each SNP pair, and (depending of the specification of addVarN) the variance estimates for D' (varDprime) and the numbers of non-missing values (n). Furthermore, the names of the SNPs (rn) will be added (in getLD, if asMatrix = FALSE).

Author(s)

Holger Schwender, <holger.schwender@udo.edu>

References

Hill, W.O. (1974). Estimation of Linkage Disequilibrium in Randomly Mating Populations. *Heredity*, 33, 229-239.

Zapata, C., Alvarez, G., and Carollo, C. (1997). Approximate Variance of the Standardized Measure of Gametic Disequilibrium D'. *American Journal of Human Genetics*, 61, 771-774.

See Also

plot.getLD, findLDblocks

Examples

```
# Load the simulated data.
data(trio.data)

# The values of Dprime and Rsquare for each pair of SNPs
# in LDdata can be computed by
ld.out <- getLD(LDdata)

# By default, the LD measures are returned as a vector.
# If they should be returned as a matrix, then use
ld.out2 <- getLD(LDdata, asMatrix = TRUE)
```

---

getMatPseudo *Generates Case-Pseudo-Control Matrix*

Description

Generates a matrix containing the genotypes of the cases and the corresponding three pseudo-controls (i.e. the genotypes of the children and the respective corresponding three genotypes not transmitted from the parents).

Usage

```
getMatPseudo(mat.snp)
```

Arguments

mat.snp	a numeric matrix in which each column represents a SNP. Each column must be a numeric vector of length $3 * t$ representing a SNP genotyped at t trios. Each of the t blocks must consist of the genotypes of father, mother, and offspring (in this order). The genotypes must be coded by 0, 1, and 2. Missing values are allowed and need to be coded by NA. This matrix might be generated from a ped-file by, e.g., employing ped2geno.

Value

A matrix with $4 * t$ rows, in which each block of four consecutive rows consists of the genotypes of the SNPs in mat.snp for the case and the three matched pseudo-controls corresponding to the respective block in mat.snp.

Author(s)

Holger Schwender, <holger.schwender@udo.edu>

See Also

colTDT, colTDT2way, colGxE

Examples

```
# Load the simulated data.
data(trio.data)

# The matrix with the genotypes of the offspring and the three
# pseudo-controls for each of the trios in mat.test can be
# generated by
matPseudo <- getMatPseudo(mat.test)
```

---

lrControl *Control Parameters for Trio Logic Regression*

Description

Specifies the control parameters for the search algorithms (i.e. either simulated annealing or MCMC) and the logic tree considered when fitting a trio logic regression model.

Usage

```
lrControl(start = 0, end = 0, iter = 0, earlyout = 0, update = 0,
   treesize = 8, opers = 1, minmass = 0, nburn = 1000, hyperpars = 0,
   output = 4)
```

Arguments

start	a numeric value specifying the upper temperature (on log10 scale) used as start temperature in simulated annealing. Must be larger than end. If both start = 0 and end = 0, these temperatures will be chosen automatically (which is not the optimal way to specify these parameters).
end	a numeric value specifying the lowest temperature (on log10 scale) used in simulated annealing. Must be smaller than start. If both start = 0 and end = 0, these temperatures will be chosen automatically (which is not the optimal way to specify these parameters).
iter	the number of iterations used in the (stochastic) search for the best trio logic regression model, i.e. either in simulated annealing (if the argument search in trioLR or trioFS is set to "sa") or in MCMC (if search = "mcmc"). If iter = 0, iter will be chosen automatically (similar to start and end) when simulated annealing is used, and will be set to iter = 50000 when MCMC is employed.
earlyout	a non-negative integer providing an option to end the search before all iter iterations in simulated annealing are considered. If during five consecutive blocks of earlyout iterations, 10 or fewer moves proposed in simulated annealing are accepted in each of the blocks, then the search will terminate. Can help to stop the search earlier, when there is no progress in the search anymore. By default, all iter iterations are considered.

update	the number of iterations in simulated annealing or MCMC after which statistics for the current trio logic regression model are displayed. This argument allows to evaluate the progress in the search for the best trio logic regression model. By default, no updates are shown.
treesize	a positive integer specifying the maximum number of leaves allowed in the logic tree of a trio logic regression model.
opers	either 1, 2, or 3 specifying if both the AND and the OR operator (opers = 1), or only the AND operator (opers = 2), or only the OR operator (opers = 3) is considered when building the logic tree.
minmass	a non-negative integer specifying the number of cases and pseudo-controls for which the logic expression (i.e. the logic tree) needs to be 1 or for which the logic expression needs to be 0 to be considered as a logic tree in the trio logic regression model. By default, minmass is either set to 20% of the trios or to 15, whatever is less.
nburn	number of initial iterations in MCMC considered as burn-in MC trio logic regression, and therefore, ignored when computing the summaries.
hyperpars	a numeric value specifying the hyperparameter for the prior on the model size when performing a MC trio logic regression. More exactly, hyperpars is assumed to be $log(P(size = k)/P(size = k + 1))$, where P is the prior on the model size.
output	a value specifying which statistics are returned in an MCMC trio logic regression analysis. If output > 0, then all fitted models are saved in a text file called "triolrlisting.tmp" in the current working directory. By setting output < 0, this can be avoided. If abs(output) > 1, bivariate statistics are gathered. If abs(output) > 2, trivariate statistics are gathered. Otherwise, only univariate statistics are determined.

Details

More details on the different control parameters and their specification can be found on the help pages of the functions logreg.anneal.control, logreg.tree.control, and logreg.mc.control for the different types of control parameters available in the R package LogicReg for a standard logic regressions.

Value

A list containing all required control parameters.

Author(s)

Holger Schwender, <holger.schwender@udo.edu>

Examples

```
# The default values for the parameters in trio logic regression
# can be specified by
myControl <- lrControl()

# If the starting temperature of Simulated Annealing should be set
# to 100 and the lowest temperature to 0.001, then this can be done by
myControl2 <- lrControl(start = 2, end = -3)
```

`ntrios2Dom` *Frequency of Dominant or Recessive Genotype Combinations*

Description

Transform the numbers of trios showing the different combinations of genotypes in trios coded by 0, 1, and 2 (i.e. considering an additive mode of inheritance) to the numbers of trios showing the different genotype combinations when the three genotypes of a SNP are coded for a dominant or a recessive mode of inheritance.

Usage

```
ntrios2Dom(matNumber, check = TRUE, quiet = FALSE)

ntrios2Rec(matNumber, check = TRUE, quiet = FALSE)
```

Arguments

`matNumber`	a numeric matrix or list of two matrices containing the numbers of trios showing the different combinations of genotypes of the offsprings and their parents, i.e. the output of `colNtrios`. Each column of each matrix corresponds to a genotype combination, and each row to a SNP.
`check`	logical specifying whether it should be checked if it is plausible that `matNumber` is the output of `colNtrios`. Meant for internal purposes only. So the default should actually not be changed.
`quiet`	logical indicating whether a message should be surpressed when only the seven numbers of trios contributing to the likelihood estimation considered in the genotypic TDT have been computed.

Value

If in `colNtrios` the argument `env` has not been specified, a matrix with the numbers of trios, where each column of this matrix corresponds to a genotype combination and each row to a SNP. Depending on whether `ntrios2Dom` or `ntrios2Rec` has been applied, the column names of this matrix start with a D or R (for dominant or recessive mode of inheritance), respectively, followed by the coding for the offspring and then the three pseudo controls (see Table 2 or 3, respectively, in Schwender et al., 2011).

If `env` has been specified in `colNtrios`, a list consisting of two matrices as the one described above is returend, one matrix for `env = 0`, and the other for `env = 1`.

Author(s)

Holger Schwender, <holger.schwender@udo.edu>

References

Schwender, H., Taub, M.A., Beaty, T.H., Marazita, M.L., and Ruczinski, I. (2011). Rapid Testing of SNPs and Gene-Environment Interactions in Case-Parent Trio Data Based on Exact Analytic Parameter Estimation. *Biometrics*, 68, 766-773.

See Also

colNtrios, colTDT

Examples

```
# Load the simulated data.
data(trio.data)

# Compute the numbers of trios.
mat.ntrios <- colNtrios(mat.test)

# Transform these numbers to the numbers
# when considering genotypes coding for a
# dominant mode of inheritance.
ntrios2Dom(mat.ntrios)
```

---

ped2geno *Transformation of Ped-File*

Description

Transforms a ped-file into a genotype file as required by, e.g., the functions for computing the genotypic TDT.

Usage

```
ped2geno(ped, snpnames = NULL, coded = c("12", "AB", "ATCG", "1234"),
   naVal = 0, cols4ID = FALSE)
```

Arguments

ped	a data frame or data table in ped format, i.e. the first six columns must contain information on the families as typically presented in ped files, where the column names of these six columns must be "famid", "pid", "fatid", "motid", "sex","affected". The last two of these six columns are ignored. The IDs of individuals in the second column must be unique (not only within the family, but among all individuals). The columns following the six columns are assumed to contain the alleles of the SNPs, where the alleles are coded using the letters/numbers in coded, and missing values are coded by naVal. Thus, the seventh and the eigth column contain the two alleles for the first SNP, the ninth and tenth the two alleles for the second SNP, and so on. Contrary to the names of the first six columns, the names of the columns representing the SNPs are ignored, and SNP names can be specified using snpnames.
snpnames	a character vector containing the names of the SNPs. If not specified, generic names are assigned (i.e. SNP1, SNP2, ...). Ignored if ped just contains one SNPs.
coded	the coding used for the alleles of the SNPs. coded = "12", e.g., means that one of the alleles is coded by 1, and the other by 0. coded = "ATCG" means that the alleles are coded by the actual base.
naVal	the value used for specifying missing values.
cols4ID	logical indicating whether columns should be added to output matrix containing the family ID and the individual ID. If FALSE, the individual IDs are used as the row names of the output matrix.

Value

A vector (if ped consists of alleles for one SNP) or matrix (otherwise) containing one column for each SNP representing the genotypes of the respective SNP, where the genotypes are coded by 0, 1, 2 (i.e. the number of minor alleles), and missing values are represented by NA. The vector or matrix contains $3 * t$ values for each SNP genotyped at the t trios, where each block of 3 values is composed of the genotypes of the father, the mother, and the offspring (in this order) of a specific trio. If data for a family with more than one children are available, each of the children is treated as a separate trio.

Author(s)

Holger Schwender, <holger.schwender@udo.edu>

See Also

tdt, tdt2way, trio.check

Examples

```
## Not run:
# Assuming there is a ped-file called pedfile.ped in the
# R working directory, this file can be read into R by
ped <- read.pedfile("pedfile.ped")

# The resulting data frame is in the typical ped format
# which needs to be transformed into the genotype format
# for applications of most of the functions in the trio
# package. This transformation can be done by
geno <- ped2geno(ped)

# This transformation can also be done directly when
# reading the ped-file into R by
geno2 <- read.pedfile("pedfile.ped", p2g = TRUE)

## End(Not run)
```

---

plot.getLD *Plotting a getLD or getLDlarge Object*

Description

Plots either the pairwise r^2 or D' values computed by either getLD or getLDlarge. Can also be used to plot the categorizations used in the procedure of Gabriel et al. (2002).

Usage

```
## S3 method for class 'getLD'
plot(x, y = "rSquare", start = 1, end = NA, squared = TRUE,
   col = NULL, xlab = "", ylab = "", cexAxis = 0.8, alpha = 0.1,
   ciLD = c(0.7, 0.98), cuRecomb = 0.9, ...)

## S3 method for class 'getLDlarge'
```

```
plot(x, y = "rSquare", start = NA, end = NA, squared = TRUE,
   col = NULL, xlab = "", ylab = "", cexAxis = 0.8, alpha = 0.1,
   ciLD = c(0.7,0.98), cuRecomb = 0.9, ...)
```

Arguments

x	the output of getLD or getLDlarge.
y	either "rSquare" (default), "Dprime", or "gabriel" specifying the LD values that should be plotted.
start	integer or character string specifying the index or the name of the first SNP, respectively, that should be plotted, where the index corresponds to the column (or row if snp.in.col = FALSE) in the matrix used as input in getLD or getLDlarge.
end	integer or character string specifying the index or the name of the last SNP, respectively, that should be plotted.
squared	should the r^2 values be plotted? If FALSE, the r values are plotted. Only considered if y = "rSquare".
col	a vector specifying the colors used in plotting of the LD values. If y = "rSquare" or y = "Dprime", different levels of gray will be used by default (the darker, the higher is the LD value). If y = "gabriel", strong LD is by default marked by blue fields, evidence of recombination by white color, and others by yellow.
xlab	character string naming the label of the x-axis.
ylab	character string naming the label of the y-axis.
cexAxis	a numeric value specifying the relative size of the SNP names displayed at the axes of the plot.
alpha	numeric value between 0 and 1. Only considered if y = "gabriel". For each pair of SNPs, a two-sided 100 * (1 - alpha)% confidence interval of D' is computed, and used to specify pairs of SNPs that are either in strong LD, or show historical evidence of recombination (see ciLD and cuRecomb). All SNP pairs not falling into these two categories are specified as 'Others'.
ciLD	numeric vector consisting of two values between 0 and 1. Only considered if y = "gabriel". If the lower bound of the confidence interval of D' for a SNP pair is larger than or equal to the first value in ciLD and the upper bound is larger than or equal to the second value, then this pair of SNP is considered to be in strong LD.
cuRecomb	numeric value between 0 and 1. Only considered if y = "gabriel". If the upper bound of the confidence interval of D' for a SNP pair is smaller than cuRecomb, then this pair of SNP is considered to show evidence of recombination.
...	further arguments of image

Author(s)

Holger Schwender, <holger.schwender@udo.edu>

References

Gabriel, S.B. et al. (2002). The Structure of Haplotype Blocks in the Human Genome. *Science*, 296, 2225-2229.

See Also

getLD, plot.LDblocks

Examples

```
# Load the simulated data.
data(trio.data)

# The values of Dprime and Rsquare for each pair of SNPs
# in LDdata can be computed by
ld.out <- getLD(LDdata)

# By default, the LD measures are returned as a vector.
# If they should be returned as a matrix, then use
ld.out2 <- getLD(LDdata, asMatrix = TRUE)

# The matrix of the Rsquare values can be plotted by
plot(ld.out)

# The matrix of the Dprime values can be plotted by
plot(ld.out, "Dprime")
```

---

plot.LDblocks *Plotting a LDblock Object*

Description

Plots either the pairwise D' values or the pairwise LD categorization used in the procedure of Gabriel et al. (2002). Additionally, the LD blocks are marked in this plot.

Usage

```
## S3 method for class 'LDblocks'
plot(x, y = "gabriel", col = NULL, start = 1, end = NA, xlab = "",
   ylab = "", cexAxis = 0.8, block.col = 2, block.lwd = 3, ...)
```

Arguments

x	the output of findLDblocks.
y	either "Dprime" or "gabriel" (default) specifying the LD values that should be plotted.
col	a vector specifying the colors used in plotting of the LD values. If y = "Dprime", different levels of gray will be used by default (the darker, the higher is the LD value). If y = "gabriel", strong LD is by default marked by blue fields, evidence of recombination by white color, and others by yellow.
start	integer or character string specifying the index or name of the first SNP, respectively, that should be plotted, where the index corresponds to the column (or row if snp.in.col = FALSE) of the matrix used as input in getLD or findLDblocks.
end	integer or character string specifying the index or name of the last SNP, respectively, that should be plotted.
xlab	character string naming the label of the x-axis.

ylab	character string naming the label of the y-axis.
cexAxis	a numeric value specifying the relative size of the SNP names displayed at the axes of the plot.
block.col	the color of the lines used to show the borders of the LD blocks.
block.lwd	numeric value specifying the size of the lines used to show the borders of the LD blocks
...	further arguments of image.

Author(s)

Holger Schwender, <holger.schwender@udo.edu>

References

Gabriel, S.B. et al.~(2002). The Structure of Haplotype Blocks in the Human Genome. *Science*, 296, 2225-2229.

See Also

findLDblocks, plot.getLD

Examples

```
# Load the simulated data.
data(trio.data)

# Estimate LD blocks.
blocks <- findLDblocks(LDdata)

# Alternatively, the LD blocks can be estimated by
ld.out <- getLD(LDdata, addVarN=TRUE)
blocks2 <- findLDblocks(ld.out)

# Plot the LD blocks showing the Gabriel categorization.
plot(blocks)

# Plot the LD blocks showing the Dprime values.
plot(blocks, "Dprime")
```

---

plot.trioLR *Plotting for trioLR Objects*

Description

Plots the logic trees or information on the visited models generated in a the trio logic regression analyis with trioLR.

Usage

```
## S3 method for class 'trioLR'
plot(x, whichTree = NA, freqType = 1, useNames = FALSE,
        addStats = TRUE, digits = 3, main = NULL, cexOper=1.5,
      cexLeaf=1.5, sizeLeaf=7, cexPar=1.3, ...)
```

Arguments

x	an object of class trioLR, i.e.\ the output of trioLR.
whichTree	positive integer specifying the model for which the logic tree should be plotted when several trio logic regression models with different maximum numbers of leaves have been fitted. Ignored if just one model has been fitted using simulated annealing or MCMC has been employed to perform a Trio Logic Regression.
freqType	positive integer between 1 and 3 specifying which statistics from the MC Trio Logic Regression analysis should be plotted. If freqType = 1, then for each variable, the percentage of models visited (after the burn-in) in the MCMC chain that contain this variable will be plotted. If freqType = 2, then for each pair of variables, this percentage will be shown. If freqType = 3, then for each pair of variables, the observed-to-expected ratio for being jointly in the models will be plotted. Ignored if simulated annealing or the greedy algorithm was used in the application of trioLR.
useNames	should the names of the variables be used in the plots? If FALSE, the index of the column is shown.
addStats	should the coefficient in the trio logic regression model and the score for the fitted model be shown in the plot? Ignored if MCMC has been used in trioLR.
digits	number of digits used in the presentation of the coefficient and score (see addStats). Ignored if addStats = FALSE or MCMC has been used in trioLR.
main	character string specifying the title that should be added to the plot. If NULL, a standard title will be added to the plot.
cexOper	the relative size of the AND- and OR-operators in the plotting of the logic tree. Ignored if MCMC has been used in trioLR.
cexLeaf	the relative size of the variable names shown in the logic tree. Ignored if MCMC has been used in trioLR.
sizeLeaf	the relative size of the boxes representing the leaves in the logic trees. Ignored if MCMC has been used in trioLR.
cexPar	the relative size of the coefficient and the score (see addStats) when plotting the logic tree. Ignored if addStats = FALSE or if MCMC has been used in trioLR.
...	ignored.

Author(s)

Holger Schwender, <holger.schwender@udo.edu>, based on the plot functions implemented by Ingo Ruczinski and Charles Kooperberg in the R package LogicReg.

References

Kooperberg, C. and Ruczinski, I. (2005). Identifying Interacting SNPs Using Monte Carlo Logic Regression. *Genetic Epidemiology*, 28, 157-170.

Li, Q., Fallin, M.D., Louis, T.A., Lasseter, V.K., McGrath, J.A., Avramopoulos, D., Wolyniec, P.S., Valle, D., Liang, K.Y., Pulver, A.E., and Ruczinski, I. (2010). Detection of SNP-SNP Interactions in Trios of Parents with Schizophrenic Children. *Genetic Epidemiology*, 34, 396-406.

Ruczinski, I., Kooperberg, C., and LeBlanc, M.L. (2003). Logic Regression. *Journal of Computational and Graphical Statistics*, 12, 475-511.

See Also

trioLR

Examples

```
# Load the simulated data.
data(trio.data)

# Prepare the data in trio.ped1 for a trio logic
# regression analysis by first calling
trio.tmp <- trio.check(dat = trio.ped1)

# and then applying
set.seed(123456)
trio.bin <- trio.prepare(trio.dat=trio.tmp, blocks=c(1,4,2,3))

# where we here assume the block structure to be
# c(1, 4, 2, 3), which means that the first LD "block"
# only consists of the first SNP, the second LD block
# consists of the following four SNPs in trio.bin,
# the third block of the following two SNPs,
# and the last block of the last three SNPs.
# set.seed() is specified to make the results reproducible.

# For the application of trio logic regression, some
# parameters of trio logic regression are changed
# to make the following example faster.
my.control <- lrControl(start=1, end=-3, iter=1000, output=-4)

# Please note typically you should consider much more
# than 1000 iterations (usually, at least a few hundred
# thousand).

# Trio regression can then be applied to the trio data in
# trio.ped1 by
lr.out <- trioLR(trio.bin, control=my.control, rand=9876543)

# where we specify rand just to make the results reproducible.

# The logic tree representing the logic expression found in
# the trio logic regression analysis can then be plotted by
plot(lr.out)
```

---

poly4root *Roots of a Fourth Degree Polynomial*

Description

While `poly4root` computes the (real-valued) roots of a polynomial of fourth degree, `poly4rootMat` can be applied to several polynomials of fourh degree at once by assuming that each row the input matrix contains the coefficients for one of the polynomials.

Usage

```
poly4root(a)

poly4rootMat(amat)
```

Arguments

a	a numeric vector of length five specifying the coefficients of the polynomial a[1]*x^4 + a[2]*x^3 + a[3]*x^2 + a[4]*x + a[5].
amat	a numeric matrix with five columns in which each row contains the five coefficients of a polynomial of fourth degree.

Value

For poly4root, a vector containing the real-valued roots of the polynomial. For poly4rootMat, a matrix with four columns in which each row contains the real-valued roots of the corresponding polynomial. If a polynomial has less than four real-valued roots, the remaining entries in the corresponding row are set to NA.

Author(s)

Holger Schwender, <holger.schwender@udo.edu>

Examples

```
# The roots of
# 2 * x^4 + 3 * x^3 - x^2 + 5 * x^1 - 4
# can be determined by
poly4root(c(2, 3, -1, 5, -4))
```

---

print.colGxE *Printing and Storing of colGxE objects*

Description

Prints the statistics computed with colGxE. getGxEstats generates a data frame containing these statistics.

Usage

```
## S3 method for class 'colGxE'
print(x, top = 5, digits = 4, onlyGxE = FALSE, ...)

## S3 method for class 'colGxEunstruct'
print(x, top = 5, digits = 4, ...)

getGxEstats(x, top = NA, sortBy = c("none", "gxe", "lrt2df", "wald2df", "lrt1df", "g"))
```

Arguments

x	an object of class colGxE, i.e. the output of the function colGxE.
top	number of top interactions that should be printed or stored in a data frame. If top is set to NA, 0, or to a value that is negative of larger than the number of interactions, then the statistics for all interactions are printed or stored in the same order as they were in the genotype matrix mat.snp used in colGxE. Otherwise, the top interactions with the smallest p-values are printed or stored, where print uses the p-values of the GxE effect to order the interactions, while

	in generateGxEstats the p-values of test specified by sortBy are employed. Ignored if sortBy = "none".
onlyGxE	logical indicating whether only the statistics for the parameter of the GxE interaction should be printed. If FALSE, the statistics for both parameters in the model as well as the relative risks for the exposed trios and statistics for the 2 df likelihood ratio test and the 2 df Wald test (if these relative risks and statistics were computed by colGxE) are shown.
digits	number of digits that should be printed.
...	ignored.
sortBy	character string specifying by the p-value of which test the SNPs should be sorted. If "none" (default), the SNPs are not sorted and the SNPs are in the same order as in the genotype matrix used to specify mat.snp in colGxE.

Author(s)

Holger Schwender, <holger.schwender@udo.edu>

References

Schwender, H., Taub, M.A., Beaty, T.H., Marazita, M.L., and Ruczinski, I. (2011). Rapid Testing of SNPs and Gene-Environment Interactions in Case-Parent Trio Data Based on Exact Analytic Parameter Estimation. *Biometrics*, 68, 766-773.

See Also

colGxE

Examples

```
# Load the simulated data for the analysis.
data(trio.data)

# Set up a vector with the binary environmental variable.
# Here, we consider the gene-gender interactions and
# assume that the children in the first 50 trios are
# girls, and the remaining 50 are boys.
sex <- rep(0:1, each = 50)

# Test the interaction of sex with each of the SNPs in mat.test
gxe.out <- colGxE(mat.test, sex)

# By default, the statistics are shown for the parameters of
# the top 5 GxE interactions and the parameters of the
# corresponding SNPs.
gxe.out

# If the top 10 GxE interactions should be displayed, then this
# can be done by
print(gxe.out, top = 10)

# The statististics for all GxE interactions (and SNPs) are
# shown, when calling
print(gxe.out, top = 0)
```

```
# If only the statistics for the GxE parameters, but not for
# the SNPs should be displayed, then use
print(gxe.out, onlyGxE = TRUE)

# A convenient way to generate a data frame with all the statistics
# computed by colGxE either for the top SNPs or for all SNPs (here,
# the top 10 SNPs) ordered by the p-values of one of the considered
# tests, e.g., the 2 df likelihood ratio test, is
dat.top3 <- getGxEstats(gxe.out, top = 10, sortBy = "lrt2df")
```

`print.trioFS` *Printing and plotting of a trioFS object*

Description

Prints or plots the most important interactions found in a trioFS analysis.

Usage

```
## S3 method for class 'trioFS'
print(x, topX = 5, show.prop = TRUE, coded = FALSE, digits = 2, ...)

## S3 method for class 'trioFS'
plot(x, topX = 15, show.prop = FALSE, coded = TRUE, cex = 0.9,
   pch = 16, col = 1, force.topX = FALSE, include0 = TRUE, add.v0 = TRUE,
   v0.col = "grey50", main = NULL, ...)
```

Arguments

`x`	an object of class `trioFS`, i.e. the output of `trioFS`.
`topX`	integer specifying how many interactions should be shown. If `topX` is larger than the number of interactions contained in `x`, all the interactions are shown. Additionally to the `topX` most important interactions, any interaction having the same importance as the `topX` most important one are `printed` or (if `force.topX = FALSE`) `plotted`.
`show.prop`	should the proportions of models containing the respective interactions be added to the output (if `print` is used)? If the output of `trioFS` should be `plotted`, then the proportions of models can be plotted instead of the values of the importance measure by setting `show.prop = TRUE`.
`coded`	should the coded variable names be displayed? Might be useful if the actual variable names are pretty long. The coded variable name of the *j*-th variable is `Xj`.
`digits`	number of digits shown in the `printed` output.
`cex`	a numeric value specifying the relative size of the text and symbols.
`pch`	specifies the used symbol. See the help of `par` for details.
`col`	the color of the text and the symbols. See the help of `par` for how colors can be specified.
`force.topX`	if `TRUE` exactly `topX` interactions are `plotted`. If `FALSE` (default) all interactions up to the `topX`th most important one and all interactions having the same importance as the `topX`th most important one are `plotted`.

include0	should the x-axis include zero regardless whether the importances of the shown interactions are much higher than 0?
add.v0	should a vertical line be drawn at $x = 0$? Ignored if include0 = FALSE and all importances are larger than zero.
v0.col	the color of the vertical line at $x = 0$. See the help page of par for how colors can be specified.
main	character string naming the title of the plot. If NULL, a standard title is added to the plot.
...	Ignored.

Author(s)

Holger Schwender, <holger.schwender@udo.edu>

See Also

trioFS

Examples

```
# Load the simulated data.
data(trio.data)

# Prepare the data in trio.ped1 for a trioFS analysis
# by first calling
trio.tmp <- trio.check(dat = trio.ped1)

# and then applying
set.seed(123456)
trio.bin <- trio.prepare(trio.dat=trio.tmp, blocks=c(1,4,2,3))

# where we here assume the block structure to be
# c(1, 4, 2, 3), which means that the first LD "block"
# only consists of the first SNP, the second LD block
# consists of the following four SNPs in trio.bin,
# the third block of the following two SNPs,
# and the last block of the last three SNPs.
# set.seed() is specified to make the results reproducible.

# For the application of trioFS, some parameters of trio
# logic regression are changed to make the following example faster.
my.control <- lrControl(start=1, end=-3, iter=1000, output=-4)

# Please note typically you should consider much more
# than 1000 iterations (usually, at least a few hundred
# thousand).

# TrioFS can then be applied to the trio data in trio.ped1 by
fs.out <- trioFS(trio.bin, control=my.control, rand=9876543)

# where we specify rand just to make the results reproducible.

# The output of trioFS can be printed by
fs.out
```

```
# By default, the five most important interactions are displayed.
# If another number of interactions, e.g., 10, should be shown,
# then this can be done by
print(fs.out, topX = 10)

# The importances can also be plotted by
plot(fs.out)
```

`print.trioLR` *Printing of trioLR Objects*

Description

Prints information on the trio logic regression model(s) fitted with `trioLR`.

Usage

```
## S3 method for class 'trioLR'
print(x, asDNF=FALSE, posBeta=FALSE, digits = 3, ...)
```

Arguments

`x`	an object of class `trioLR`, i.e.\ the output of `trioLR`.
`asDNF`	should the disjunctive normal form of the logic expression represented by the logic tree be printed? If `FALSE`, the logic expression is printed as found by the search algorithm in trio logic regression. An advantage of the disjunctive normal form representation is that the interactions comprised by the logic expression are given by the AND-combinations in the disjunctive normal form. Note that not necessarily the minimum disjunctive normal form is printed so that all interactions comprised by the model are shown, even if some of the interactions are redundant for the evaluating the logic tree.
`posBeta`	should the disjunctive normal form be determined as if the sign of the coefficient in trio logic regression model is positive? If `FALSE`, the sign is ignored when transforming the logic tree into its disjunctive normal form. If `TRUE` and the coefficient is negative, the complement of the logic expression is transformed into its disjunctive normal form and the coefficient is multiplied by -1. Ignored if `asDNF = FALSE` or the fitted logic tree only contains one leaf.
`digits`	number of digits used in the printing of the score and the parameter estimate of the fitted trio logic regression model(s).
`...`	ignored.

Author(s)

Holger Schwender, <holger.schwender@udo.edu>, based on the `plot` functions implemented by Ingo Ruczinski and Charles Kooperberg in the `R` package `LogicReg`.

References

Kooperberg, C. and Ruczinski, I. (2005). Identifying Interacting SNPs Using Monte Carlo Logic Regression. *Genetic Epidemiology*, 28, 157-170.

Li, Q., Fallin, M.D., Louis, T.A., Lasseter, V.K., McGrath, J.A., Avramopoulos, D., Wolyniec, P.S., Valle, D., Liang, K.Y., Pulver, A.E., and Ruczinski, I. (2010). Detection of SNP-SNP Interactions in Trios of Parents with Schizophrenic Children. *Genetic Epidemiology*, 34, 396-406.

Ruczinski, I., Kooperberg, C., and LeBlanc, M.L. (2003). Logic Regression. *Journal of Computational and Graphical Statistics*, 12, 475-511.

See Also

trioLR

Examples

```
# Load the simulated data.
data(trio.data)

# Prepare the data in trio.ped1 for a trio logic
# regression analysis by first calling
trio.tmp <- trio.check(dat = trio.ped1)

# and then applying
set.seed(123456)
trio.bin <- trio.prepare(trio.dat=trio.tmp, blocks=c(1,4,2,3))

# where we here assume the block structure to be
# c(1, 4, 2, 3), which means that the first LD "block"
# only consists of the first SNP, the second LD block
# consists of the following four SNPs in trio.bin,
# the third block of the following two SNPs,
# and the last block of the last three SNPs.
# set.seed() is specified to make the results reproducible.

# For the application of trio logic regression, some
# parameters of trio logic regression are changed
# to make the following example faster.
my.control <- lrControl(start=1, end=-3, iter=1000, output=-4)

# Please note typically you should consider much more
# than 1000 iterations (usually, at least a few hundred
# thousand).

# Trio regression can then be applied to the trio data in
# trio.ped1 by
lr.out <- trioLR(trio.bin, control=my.control, rand=9876543)

# where we specify rand just to make the results reproducible.

# The output of trioLR can then be displayed by
lr.out

# This output shows the detected logic expression. If this
# expression should be displayed in disjunctive normal form,
# then this can be done by
```

```
print(lr.out, asDNF = TRUE)
```

---

probTDT *TDT on genotype probabilities matrix*

Description

Computes the genotypic TDT for a a matrix representing SNP genotype probabilities.

Usage

```
probTDT(mat.geno, model = c("additive", "dominant", "recessive"),
   size = 50)
```

Arguments

mat.geno	a numeric matrix with one row for each SNP and $9 * t$ columns representing genotype probabilities for t trios. Each of the t blocks (i.e. snp[1:9], snp[10:18], ...) must consist of sets of the three genotype probabilities for AA, AB and BB calls, of father, mother, and offspring (in this order), as would be output by BEAGLE, for example. The genotype probabilities must sum to 1 (up to slight imprecision) in each individual. Missing values are allowed and need to be coded by NA. Note that the order of the columns is not checked to be in terms of minor allele – any dominant or recessive tests are for allele B, as ordered in the mat.geno, not necessarily for the minor allele.
model	type of model that should be fitted. Abbreviations are allowed. Thus, e.g., model = "dom" will fit a dominant model, and model = "r" an recessive model. See description of mat.geno for a caveat about allele ordering.
size	the number of SNPs considered simultaneously when computing the parameter estimates. Ignored if fast = FALSE.

Value

An object of class colTDT consisting of the following numeric values or vectors, respectively:

coef	the estimated parameter,
se	the estimated standard deviation of the parameter estimate,
stat	Wald statistic,
RR	the relative risk, i.e.\ for trio data, exp(coef) (see Schaid, 1996),
lowerRR	the lower bound of the 95% confidence interval for RR,
upperRR	the upper bound of the 95% confidence interval for RR,
usedTrios	the number of trios affecting the parameter estimation,
pMendelErr	the sum across families of probabilities of Mendelian errors.

Author(s)

Margaret Taub, <mtaub@jhsph.edu>

References

Schaid, D.J. (1996). General Score Tests for Associations of Genetic Markers with Disease Using Cases and Their Parents. *Genetic Epidemiology*, 13, 423-449.

Schwender, H., Taub, M.A., Beaty, T.H., Marazita, M.L., and Ruczinski, I. (2011). Rapid Testing of SNPs and Gene-Environment Interactions in Case-Parent Trio Data Based on Exact Analytic Parameter Estimation. *Biometrics*, 68, 766-773.

Taub M.A., Schwender H., Beatty T.H., Louis T.A., Ruczinski I. (2012). Incorporating genotype uncertainties into the genotypic TDT for main effects and gene-environment interactions. *Genetic Epidemiology*, 36, 225-234.

See Also

tdt

Examples

```
# Load the simulated data.
data(trio.data)

# All SNPs in prob.mat.test can be tested by
prob.tdt.out <- probTDT(prob.mat.test)

# By default, an additive mode of inheritance is considered.
# If another mode, e.g., the dominant mode, should be
# considered, then this can be done by
prob.tdt.out2 <- probTDT(prob.mat.test, model = "dominant")

# By default, statistics for the top 5 SNPs are displayed.
# If another number of SNPs, say 10, should be displayed,
# then this can be done by
print(prob.tdt.out2, top = 10)

# The statistics for all SNPs (not ordered by their
# significance) can be obtained by
print(prob.tdt.out2, top = 0)
```

---

read.pedfile *Reading a Ped File*

Description

Reads a ped file into R and creates a data frame or data table in ped format, or transform the ped file into a matrix in genotype format.

Usage

```
read.pedfile(file, first.row = NA, coded = NULL, naVal = 0, sep = " ",
   p2g = FALSE, non.rs.IDs = FALSE, asDataTable = FALSE, cols4ID=FALSE)
```

Arguments

file	the filename (if necessary with path) of a ped file that should be read into R.
first.row	logical indicating whether the first row of file also contains data for a subject. If FALSE, the first row is assumed to contain the SNP names. By default, read.pedfile tries to figure out automatically if the first column contains the SNP names or data for a subject.
coded	a character string stating how the alleles of the SNPs are coded. Possible values are "12", "AB", "1234", "ATCG". For details, see ped2geno. By default, read.pedfile tries to figure out automatically how the alleles are coded.
naVal	value or character string specifying how missing values in the SNP data are coded.
sep	character string specifying how the SNP names in the first row of file are separated. Ignored if first.row = TRUE.
p2g	logical indicating whether the ped file should be transformed into a matrix in genotype format. If FALSE, depending on the specification of asDataTable, either a data frame or data table in ped format is returned. Otherwise, ped2geno is called within read.pedfile to transform the data table into a matrix in genotype format, and the matrix is returned.
non.rs.IDs	logical indicating whether (some of) the SNP names are specified by other names than rs-IDs.
asDataTable	logical indication whether a data table or a data frame should be returned, if p2g is set to FALSE. By default, a data frame is returned in the case of p2g = FALSE.
cols4ID	logical indicating whether columns should be added to output matrix containing the family ID and the individual ID. If FALSE, the individual IDs are used as the row names of the output matrix.

Value

If p2g = FALSE, depending on the specification of asDataTable, either a data frame or data table in ped format. If p2g = TRUE, a matrix in genotype format is returned.

Author(s)

Holger Schwender, <holger.schwender@udo.edu>

See Also

ped2geno

Examples

```
## Not run:
# Assuming there is a ped-file called pedfile.ped in the
# R working directory, this file can be read into R by
ped <- read.pedfile("pedfile.ped")

# The resulting data frame is in the typical ped format
# which needs to be transformed into the genotype format
# for applications of most of the functions in the trio
# package. This transformation can be done by
geno <- ped2geno(ped)
```

```
# This transformation can also be done directly when
# reading the ped-file into R by
geno2 <- read.pedfile("pedfile.ped", p2g = TRUE)

## End(Not run)
```

`removeSNPs` *Remove SNPs or Trios*

Description

Functions for removing SNPs with a low minor allele frequency or a high percentage of missing values, for removing trios in which at least one member shows a high percentage of missing values, for ordering the SNPs by their position in the genome, and for computing the minor allele frequencies of the SNPs based on only the genotypes of the parents, where each parent is only used once in this computation, even if this person is part of more than one of the trios.

Usage

```
removeSNPs(geno, maf = NA, perc.na = NA)

removeTrios(geno, perc.na = 1)

orderSNPs(geno, map, snp = "SNP", orderBy = c("Chr", "Position"))

colMAFtrio(geno, changeMinor = FALSE)
```

Arguments

`geno`	a matrix in genotype format, i.e.\ the output of `ped2geno` or `read.pedfile` with `p2g` set to `TRUE`.
`maf`	a numeric value. If specified, i.e.\ not `NA`, all SNPs with a minor allele frequency less than `maf` are removed, where `maf` can range from 0 and 0.2. If, e.g., `maf = 0`, monomorphic SNPs are removed.
`perc.na`	a numeric value between 0 and 1 specifying a cutoff for the percentage of missing values that a SNP or a subject is allowed to have. If more than 100 * `perc.na`% of the genotypes of a SNP or a subject is missing, then this SNP or the trio to which this subject belong, respectively, is removed `geno`.
`map`	a data frame containing the chromosome and the position for all the SNPs in `geno`.
`snp`	a character string giving the (case-sensitive) name of the column of `map` containing the SNP IDs used as column names in `geno`.
`orderBy`	character string of length 2 specifying the (case-sensitive) names of the columns of `map` containing the chromosomes and the positions of the SNPs in `geno`.
`changeMinor`	logical specifying whether 1 - minor allele frequency should be returned when the MAF is larger than 0.5. The MAF might be larger than 0.5, if the minor allele was specified on another data set than the one considered in `colMAFtrio`.

Value

For `removeSNPs`, `removeTrios`, and `orderSNPs`, a reduced or ordered version of `geno`. For `colMAFtrio`, a vector containing the minor allele frequencies of the SNPs in `geno`.

Author(s)

Holger Schwender, <holger.schwender@udo.edu>

Examples

```
# Load the simulated data.
data(trio.data)

# All SNPs with a minor allele frequency smaller than 0.1
# can be removed from mat.test by
mat2 <- removeSNPs(mat.test, maf = 0.1)

# The minor allele frequencies for all SNPs can be
# determined (based on the genotypes of the parents) by
maf <- colMAFtrio(mat.test)
```

`scoreTDT` *Score Tests for SNPs, GxE, and GxG Interactions*

Description

Performs score tests for all individual SNPs (`scoreTDT`), all interactions of each SNP with an environmental variable (`scoreGxE`), or all interactions of two SNPs (`scoreGxG`) comprised by an input matrix based on the same log-likelihood considered in the corresponding genotypic TDT, where in `scoreGxG` the conditional logistic regression model including only one parameter (for the interaction effect) is used.

Additionally, the maximum over the score statistics for testing an additive, dominant, and recessive effect can be determined using `scoreMaxStat`.

Usage

```
scoreTDT(mat.snp, model = c("additive", "dominant", "recessive"), size = 20)

scoreGxE(mat.snp, env, model = c("additive", "dominant", "recessive"), size = 20,
   famid = NULL)

scoreGxG(mat.snp, model = c("additive", "dominant", "recessive"), genes = NULL,
   size = 20)

scoreMaxStat(mat.snp, size = 20)

## S3 method for class 'scoreTDT'
print(x, top = 5, digits = 4, ...)

## S3 method for class 'scoreGxE'
print(x, top = 5, digits = 4, onlyGxE = FALSE, ...)
```

```
## S3 method for class 'maxScoreTrio'
print(x, top = 5, digits = 4, ...)
```

Arguments

mat.snp	a numeric matrix in which each column represents a SNP. Each column must be a numeric vector of length $3 * t$ representing a SNP genotyped at t trios. Each of the t blocks must consist of the genotypes of father, mother, and offspring (in this order). The genotypes must be coded by 0, 1, and 2. Missing values are allowed and need to be coded by NA. This matrix might be generated from a ped-file by, e.g., employing ped2geno.
model	type of model that should be fitted. Abbreviations are allowed. Thus, e.g., model = "dom" will fit a dominant model, and model = "r" an recessive model.
size	the number of models considered simultaneously when computing the parameter estimates.
env	a vector of length t (see mat.snp) containing for each offspring the value of a binary environmental variable, which must take the values 0 and 1.
famid	a vector of the same length as env specifying the family IDs for the corresponding values of the environmental variable in env. Can be used to reorder the vector env when the order of the trios differs between env and mat.snp.
genes	a character vector containing the names of the genes (or LD-blocks or other genetic sets of SNPs) to which the SNPs belong. If specified, only the two-way interactions between SNPs from different genes (or LD-blocks or other genetic sets of SNPs) are tested. If NULL, all two-way interactions between all possible pairs of SNPs are tested.
x	an object of class scoreTDT, scoreGxE, or maxScoreTrio, i.e. the output of the function scoreTDT / scoreGxG, scoreGxE, or scoreMaxStat, respectively.
digits	number of digits that should be printed.
top	number of interactions that should be printed. If the number of interactions is smaller than or equal to top, then the statistics for all interactions are printed in the order of their computation. Otherwise, the top interactions with the smallest p-values are printed.
onlyGxE	logical indicating whether only the statistics for the parameter of the GxE interaction should be printed. If FALSE, the statistics for both parameters in the model are shown.
...	ignored.

Value

For scoreTDT and scoreGxG, an object of class scoreTDT containing numeric vectors

score	the scores for all SNPs or SNP interactions,
info	the denominators of the corresponding score statistics
,	
stat	the values of the score statistics for all SNPs or SNP interactions
,	
pval	the corresponding p-values computed based on a ChiSquare-distribution with 1 degree of freedom.

Author(s)

Holger Schwender, <holger.schwender@udo.edu>

See Also

colTDT, colGxE, colTDT2way

Examples

```
# Load the simulated data.
data(trio.data)

# A score test can be applied to the SNPs in
# mat.test by
s.out <- scoreTDT(mat.test)

# By default, an additive mode of inheritance is considered.
# Another mode, e.g., the dominant mode can be considered by
sDom.out <- scoreTDT(mat.test, model = "dominant")

# The test statistic of the MAX score test can be computed by
sMax.out <- scoreMaxStat(mat.test)


# The interaction between a binary environmental factor,
# e.g., the gender, and each SNP in mat.test can be tested
# by setting up the vector containing the value of the
# environmental factor for each trio. If we, e.g., assume
# that the children in the first 50 trios are girls
# and in the remaininge 50 trios boys, then this vector
# can be generated by
sex <- rep(0:1, each = 50)

# and the interaction between sex and each SNP in mat.test
# can be tested with a score test by
sgxe.out <- scoreGxE(mat.test, sex)

# The interactions between all pairs of SNPs in mat.test
# can be tested with a score test by
sgxg.out <- scoreGxG(mat.test)
```

---

tdt *Genotypic TDT*

Description

Computes the genotypic TDT for a SNP or for each column of a matrix representing a SNP.

Usage

```
tdt(snp, model = c("additive", "dominant", "recessive"))

colTDT(mat.snp, matNumber = NULL, model = c("additive", "dominant", "recessive"),
```

```
   size = 50)

## S3 method for class 'tdt'
print(x, digits = 4, ...)

## S3 method for class 'colTDT'
print(x, top = 5, digits = 4, ...)
```

Arguments

snp	a numeric vector of length $3 * t$ representing a SNP genotyped at t trios. Each of the t blocks (i.e. snp[1:3], snp[4:6], ...) must consist of the genotypes of father, mother, and offspring (in this order). The genotypes must be coded by 0, 1, and 2. Missing values are allowed and need to be coded by NA. The vector must thus have the same structure as the output of trio.check, or the genotype example data sets such as trio.gen1 (see data(trio.gen1)), and can be generated from a ped-file by, e.g., employing ped2geno.
mat.snp	a numeric matrix in which each column represents a SNP. Each of the SNPs must have the same structure as snp, and can, e.g., be generated from a ped-file by employing ped2geno.
matNumber	a numeric matrix containing the numbers of trios showing the different combinations of genotypes of the offsprings and their parents, i.e. the output of colNtrios (with unspecified argument env). Each column of matNumber corresponds to a genotype combination, and each row to a SNP.
model	type of model that should be fitted. Abbreviations are allowed. Thus, e.g., model = "dom" will fit a dominant model, and model = "r" an recessive model.
size	the number of SNPs considered simultaneously when computing the parameter estimates. Ignored if fast = FALSE.
x	an object of class tdt or colTDT, i.e. the output of the function tdt (or tdtGxG) or the function colTDT.
digits	number of digits that should be printed.
top	number of interactions that should be printed. If top is less than or equal to zero, set to NA, or larger than the number of SNPs, then the statistics for all SNPs are printed in the order as they were in the genotype matrix used as input into colTDT. Otherwise, the top interactions with the smallest p-values are printed.
...	ignored.

Value

An object of class tdt or colTDT consisting of the following numeric values or vectors, respectively:

coef	the estimated parameter,
se	the estimated standard deviation of the parameter estimate,
stat	Wald statistic,
RR	the relative risk, i.e.\ for trio data, exp(coef) (see Schaid, 1996),
lowerRR	the lower bound of the 95% confidence interval for RR,
upperRR	the upper bound of the 95% confidence interval for RR,
usedTrios	the number of trios affecting the parameter estimation (only for colTDT),
...	further internal parameters

Author(s)

Holger Schwender, <holger.schwender@udo.edu>

References

Schaid, D.J. (1996). General Score Tests for Associations of Genetic Markers with Disease Using Cases and Their Parents. *Genetic Epidemiology*, 13, 423-449.

Schwender, H., Taub, M.A., Beaty, T.H., Marazita, M.L., and Ruczinski, I. (2011). Rapid Testing of SNPs and Gene-Environment Interactions in Case-Parent Trio Data Based on Exact Analytic Parameter Estimation. *Biometrics*, 68, 766-773.

See Also

tdt2way, ped2geno

Examples

```
# Load the simulated data.
data(trio.data)

# One particular SNP (e.g., the one in the first
# column of mat.test) can be tested by
tdt.out <- tdt(mat.test[,1])

# All SNPs in mat.test can be tested by
tdt.out2 <- colTDT(mat.test)

# By default, an additive mode of inheritance is considered.
# If another mode, e.g., the dominant mode, should be
# considered, then this can be done by
tdt.out3 <- colTDT(mat.test, model = "dominant")

# By default, statistics for the top 5 SNPs are displayed.
# If another number of SNPs, say 10, should be displayed,
# then this can be done by
print(tdt.out2, top = 10)

# The statistics for all SNPs (not ordered by their
# significance) can be obtained by
print(tdt.out2, top = 0)
```

---

tdtGxG *Genotypic TDT for Two-Way Interactions*

Description

tdtGxG and colGxG perform the genotypic TDT for the interaction of two SNPs or of each pair of columns of a genotype matrix, respectively.

fastGxG provides a fast implementation for the genotypic TDT for two-way interactions when considering the simplest conditional logistic regression model only containing one parameter for the interaction effect. It thus leads to the same results as colGxG with test = "screen". In fastGxGrec, an analytic solution to the genotypic TDT based on the simplest model for testing a recessive x

recessive model is implemented, which is even faster than `fastGxG` with `model = "recessive"`. In future versions of this package, `fastGxG` and `fastGxGrec` will be joint with `colGxG`.

The genotypic TDT for testing two-way interactions makes use of the 16 possible genotypes that can be obtained from combining the parents' genotypes of the two considered SNPs. Thus, for each family, genotypes for one case (i.e. the affected offspring) and 15 pseudo-controls are used.

Usage

```
tdtGxG(snp1, snp2, test = c("epistatic", "lrt", "full", "screen"),
     model = c("additive", "dominant", "recessive"))

colGxG(mat.snp, test = c("epistatic", "lrt", "full", "screen"), genes = NULL,
   maf = FALSE, model = c("additive", "dominant", "recessive"))

fastGxG(mat.snp, model = c("additive", "dominant", "recessive"),
   genes = NULL, interval = c(-10, 10), tol = 10^-8, maxiter = 1000,
   size = 20)

fastGxGrec(mat.snp, genes = NULL, size = 20)
```

Arguments

`snp1`, `snp2`	numeric vectors of length $3 * t$ representing two SNPs genotyped at t trios. Each of the t blocks (i.e. `snp1[1:3]`, `snp1[4:6]`, ..., and `snp2[1:3]`, `snp2[4:6]`, ...) must consist of the genotypes of father, mother, and offspring (in this order). The genotypes must be coded by 0, 1, and 2. Missing values are allowed and need to be coded by `NA`. The vectors must thus have the same structure as the output of `trio.check`, or the genotype example data sets such as `trio.gen1` (see `data(trio.gen1)`), and can be generated from a ped-file by, e.g., employing `ped2geno`.
`mat.snp`	a numeric matrix in which each column represents a SNP. Each of the SNPs must have the same structure as `snp`, and can, e.g., be generated from a ped-file by employing `ped2geno`.
`test`	character string naming the GxG test that should be performed. If `test = "epistatic"`, then a conditional logistic regression version of the test proposed by Cordell (2002) is used to test for epistatistical interactions. If `test = "full"`, a conditional logistic regression model containing one parameter for each SNP and one parameter for the interaction of these two SNPs will be fitted and a Wald test for the interaction term will be performed, where a genetic model specified by `model` is assumed for both SNPs. If `test = "lrt"`, a likelihood ratio test is performed comparing the fit of this model with the fit of a conditional logistic regression model only containing the two parameters for the main effects of the SNPs. If `test = "screen"`, a conditional logistic regression model only composed of one parameter for the interaction of the two SNPs will be fitted and a Wald test will be performed, where the genetic model specified by `model` is assumed for both SNPs.
`genes`	a character vector containing the names of the genes to which the SNPs belong. If specified, only the two-way interactions between SNPs from different genes are tested. If `NULL`, all two-way interactions between all possible pairs of SNPs are tested.
`maf`	logical indicating whether the minor allele frequency (computed by considering the genotypes of only the parents) should be added to the output.

model	type of model that should be considered. Abbreviations are allowed. Thus, e.g., model = "dom" will consider a dominant model for each of the respective two SNPs, and model = "r" an recessive model. Ignored if epistatic = TRUE.
interval	the end-points of the interval to be searched for the root. For details, see uniroot.
tol	the desired accuracy/convergence tolerance. For details, see uniroot.
maxiter	the maximum number of iterations. For details, see uniroot.
size	the number of interactions considered simultaneously when computing the parameter estimates.

Value

Depending on test, the output contains statistics and p-values either of a likelihood ratio test (test = "epistatic" or test = "lrt") or the Wald statistics and the corresponding p-values for the interaction term in the conditional logistic regression model (test = "full" or test = "screen"). If maf = TRUE, a vector maf containing the minor allele frequencies of each SNP and a matrix mat.maf with two columns containing the SNP-wise minor allele frequencies for each tested pair of SNPs are added to the output of colGxG.

Author(s)

Holger Schwender, <holger.schwender@udo.edu>

References

Cordell, H. J. (2002). Epistasis: What it Means, what it Doesn't mean, and Statistical Methods to Detect it in Humans. Human Molecular Genetics, 11, 2463-2468.

Schwender, H., Taub, M.A., Beaty, T.H., Marazita, M.L., and Ruczinski, I. (2011). Rapid Testing of SNPs and Gene-Environment Interactions in Case-Parent Trio Data Based on Exact Analytic Parameter Estimation. *Biometrics*, 68, 766-773.

See Also

tdt, ped2geno

Examples

```
# Load the simulated data.
data(trio.data)

# The interaction between a particular pair of SNPs
# (e.g., the ones in the first and second column of
# mat.test) can be tested by
gxg.out <- tdtGxG(mat.test[,1], mat.test[,2])

# All pairs of SNPs in mat.test can be tested by
gxg.out2 <- colGxG(mat.test)

# By default, Cordell's likelihood ratio test for
# epistatistic interactions is used. This is the
# most sophisticated, but also most time-consuming
# test. If another test, e.g., the one considering
# a conditional logistic regression model only
# containing a term for the interaction, should
# be used, then this can be done by
```

```
gxg.out3 <- colGxG(mat.test, test = "screen")

# In this case, different modes of inheritance can
# be considered (by default, the additive mode is
# considered). If a dominant model (for both SNPs)
# should be tested, this can be done by
gxg.out4 <- colGxG(mat.test, test = "screen", model ="dom")

# If just a subset of all pairs of SNPs should be
# tested, e.g., only pairs of SNPs belonging to different
# genes, then this can be done by first specifying a
# vector specifying which SNP belongs to which genes.
# If we, e.g., assume that the first two SNPs in mat.test
# belong to gene G1 and the other four SNPs to G2, then
# this vector can be specified by
genes <- paste("G", rep(1:2, c(2,4)), sep="")

# and only the pairs of SNPs in which the two SNPs belong
# to different genes can be tested with Cordell's
# likelihood ratio test by
gxg.out5 <- colGxG(mat.test, genes = genes)
```

`trio.check` *Check Case-Parent Trio Data for Mendelian Errors*

Description

This function checks case-parent trio data in linkage or genotype format for Mendelian errors. If no errors are found, the function returns an object suitable for input to the `trio.prepare` function. Otherwise, an object identifying the Mendelian errors is returned.

Usage

```
trio.check(dat, is.linkage=TRUE, replace=FALSE)
```

Arguments

`dat`	A matrix or data frame of pedigree data in linkage format, or in genotype format. If the data are in **linkage format**, the file has to have the standard linkage/pedigree format. Each row describes an individual, and the columns are <*famid*> <*pid*> <*fatid*> <*motid*> <*sex*> <*affected*> <*genotype:1_1*> <*genotype:1_2*> ... <*genotype:n_1*> <*genotype:n_2*>. Here, <*famid*> is a unique identifier for each family, <*pid*> is a unique identifier for an individual within each family, <*fatid*> and <*motid*> identify the father and mother of the individual, <*sex*> denotes the gender, using the convention 1=male, 2=female, <*affected*> denotes the disease status (0=unknown, 1=unaffected, 2=affected). Only one phenotype column is allowed. Each genotype is encoded using two columns (<*genotype:k_1*> and <*genotype:k_2*>), identifying the alleles (1 for the major allele, 2 for the minor allele, 0 if missing). Other values for the alleles will result in an error. Please see the data frames `trio.ped1` and `trio.ped2` contained in this package as examples for trio data in linkage file format (complete and with missing records, respectively).

If the data are in **genotype format**, each row in the object describes an individual, and each block of three consecutive rows describes the two parents and the affected child in a trio. The columns in the object are <*famid*> <*pid*> <*genotype_1*> ... <*genotype_n*>. Here, <*famid*> is a unique identifier for each family, <*pid*> is a unique identifier for an individual within each family (with each block of three consecutive rows describing the two parents and the affected child in a trio). Each <*genotype*> is encoded as an integer indicating the number of variant alleles (e.g. 0=common homozygote, 1=heterozygote, and 2=rare homozygote, and NA=missing genotype). Please see the data frames `trio.gen1` and `trio.gen2` contained in this package as examples for trio data in linkage file format (complete and with missing records, respectively).

`is.linkage` A logical value indicating if the case parent data are in linkage file format (`TRUE`) or in genotype format (`FALSE`).

`replace` A logical value indicating whether existing Mendelian errors should be replaced by missing values. For each Mendelian error found (for a particular trio at a particular locus), all three genotypes are replaced by `NA`, and an object suitable for input to the `trio.prepare` function is returned.

Details

The first function used from this package should always be `trio.check`. Unless otherwise specified, this function assumes that the data are in linkage format, however, genotype data can also be accommodated. If no Mendelian inconsistencies in the data provided are identified, `trio.check` creates an object that can be processed in the subsequent analysis with the `trio.prepare` function. If the data were in linkage format, the genotype information for each SNP will be converted into a single variable, denoting the number of variant alleles.

To delineate the genotype information for the pseudo-controls in the subsequent analysis, the trio data must not contain any Mendelian errors. The function `trio.check` returns a warning, and an R object with relevant information when Mendelian errors are encountered in the supplied trio data. It is the users responsibility to find the cause for the Mendelian errors and correct those, if possible. However, Mendelian inconsistencies are often due to genotyping errors and thus, it might not be possible to correct those in a very straightforward manner. In this instance, the user might want to encode the genotypes that cause theses Mendelian errors in some of the trios as missing data. The function `trio.check` allows for this possibility, using the argument `replace=T`.

Value

The function `trio.check` returns a list with the following elements:

`trio` A data frame with the genotypes of the trios, suitable for input to the function `trio.prepare`. This element will be `NULL` if Mendelian errors are detected.

`errors` This element will be `NULL` if no Mendelian errors are detected. Otherwise, this element will be a data frame with five columns, indicating the Mendelian errors detected in the object `dat`. The five columns of the data frame refer to the trio (`trio`), the family id (`famid`), the genotype (`snp`), the row numbers (`r`), and the column numbers (`c`).

`trio.err` This element will be `NULL` if no Mendelian errors are detected. Otherwise, this element will be a data frame with the trio genotype data. If the input was a linkage file, the data will be converted from alleles to genotypes. If the input was a genotype file, this element will be identical to the input.

Author(s)

Qing Li, mail2qing@yahoo.com

References

Li, Q., Fallin, M.D., Louis, T.A., Lasseter, V.K., McGrath, J.A., Avramopoulos, D., Wolyniec, P.S., Valle, D., Liang, K.Y., Pulver, A.E., and Ruczinski, I. (2010). Detection of SNP-SNP Interactions in Trios of Parents with Schizophrenic Children. Genetic Epidemiology, 34, 396-406.

See Also

trio.prepare

Examples

```
data(trio.data)
trio.tmp <- trio.check(dat=trio.ped1)
str(trio.tmp, max=1)
trio.tmp$trio[1:6,]

trio.tmp <- trio.check(dat=trio.ped.err)
str(trio.tmp, max=1)
trio.tmp$errors
trio.tmp$trio.err[1:3, c(1,2, 11:12)]
trio.ped.err[1:3,c(1:2, 23:26)]

trio.tmp <- trio.check(dat=trio.gen.err, is.linkage=FALSE)
trio.tmp$errors
trio.tmp$trio.err[1:6, c(1,2,7), drop=FALSE]

trio.rep <- trio.check(dat=trio.gen.err, is.linkage=FALSE, replace=TRUE)
trio.rep$trio[1:6,c(1,2,7)]
```

---

trio.data *Case-Parent Trio Data*

Description

trio.data contains several simulated data sets used in the different examples for the analyses with the functions in the R package trio.

For the applications of genotypic TDTs for individual SNPs and two-way interactions with, for example, tdt and tdt2way, respectively, trio.data contains a 300 x 6 matrix called mat.test consisting of genotype data for 100 trios genotyped at 6 SNPs.

For the application of probTDT to genotype probabilities, trio.data contains a 334 x 180 matrix called prob.mat.test containing genotype probabilities for 334 SNPs and 20 trios.

For the preparation of the trio data for an application of trio logic regression with trio.check and trio.prepare, trio.data contains different data set containing genotype data for 10 SNPs in 100 trios in different formats.

trio.gen1, trio.gen2, and trio.gen.err consist of 12 columns and 300 rows, where the first two columns contain family identifier and individual identifier. In the columns afterwards, each SNPs is encoded in one variable denoting the number of minor alleles.

trio.ped1, trio.ped2, and trio.ped.err consist of 26 columns and 300 rows, where the first six columns identify the family structure of the data, and the phenotype. Besides the variables providing information on the family structure and the phenotypes (columns 1 to 6), each SNPs is encoded in two variables denoting the alleles.

Contrary to the other data sets, trio.gen.err and trio.ped.err contain Mendelian errors.

For the application of the functions getLD and findLDblocks for computing the pairwise LD values and for detecting the LD blocks, respectively, trio.data contains a 500 x 50 matrix called LDblock that is composed of genotype data for 10 LD blocks each consisting of 5 SNPs in strong LD.

Finally, for the simulation of trio data with trio.sim, trio.data contains examples for haplotype frequencies used in these simulations. Both freq.hap and simuBkMap are data.frames containing haplotype information, including the haplotype block identifier, haplotype, and haplotype frequency. While freq.hap is a data frame consisting of 20 rows and 3 columns, simuBkMap consists of 66 rows and 3 columns. step3way is a list internally used for simulation, containing some indexes and sampling frequencies.

Author(s)

LDdata and mat.test: Holger Schwender, <holger.schwender@udo.edu>; prob.mat.test: Margaret Taub, <mtaub@jhsph.edu>; all other data sets: Qing Li, <mail2qing@yahoo.com>

Examples

```
# Data can be loaded by
data(trio.data)
```

---

trio.permTest *Permutation Tests for Trio Logic Regression*

Description

Performs either a null-model or a conditional permutation test for a trio logic regression analysis.

Usage

```
trio.permTest(object, conditional = FALSE, n.perm = 10, nleaves = NULL,
   control = NULL, rand = NA)
```

Arguments

object	an object of class trioLR, i.e. the output of the function trioLR. This object must be the result of a trio logic regression analysis in which a single model has been fitted (i.e. in trioLR, search must have been set to "sa" and nleaves must have been a single integer).
conditional	should the conditional permutation test be performed? If FALSE, a null-model permutation test is done analogously to the null-model permutation test for a standard logic regression for population-based data implemented in the function logreg of the R package LogicReg. If TRUE, a test analogous to the conditional permutation test for a standard logic regression is performed.
n.perm	integer specifying the number of permutations.

`nleaves` integer specifying the maximum number of leaves that the logic tree in the trio logic regression model is allowed to have. If `NULL`, the maximum number of leaves saved in `object` is used.

`control` a list containing the control parameters for the search algorithms and the logic tree considered in `trioLR`, where the parameters for an MCMC run and the logic tree are ignored. If `NULL` (i.e. by default), the same values for the parameters are used that have been employed in the original analysis with `trioLR`. If other values should be used, it is highly recommended to specify `control` by employing `lrControl`.

`rand` an integer. If specified, the random number generator will be set into a reproducible state.

Value

A list consisting of

`origScore` `NA`, if `conditional = FALSE`, and otherwise, the score, i.e.\ the value of the partial likelihood, of the original model saved in `object`

,

`permScore` a vector of length `n.perm` containing the scores for the trio logic regression models built in the iterations of the permutation test.

Author(s)

Qing Li, <mail2qing@yahoo.com>. Modified by Holger Schwender.

References

Li, Q., Fallin, M.D., Louis, T.A., Lasseter, V.K., McGrath, J.A., Avramopoulos, D., Wolyniec, P.S., Valle, D., Liang, K.Y., Pulver, A.E., and Ruczinski, I. (2010). Detection of SNP-SNP Interactions in Trios of Parents with Schizophrenic Children. Genetic Epidemiology, 34, 396-406.

See Also

`trioLR`

Examples

```
# Load the simulated data.
data(trio.data)

# Prepare the data in trio.ped1 for a trio logic
# regression analysis by first calling
trio.tmp <- trio.check(dat = trio.ped1)

# and then applying
set.seed(123456)
trio.bin <- trio.prepare(trio.dat=trio.tmp, blocks=c(1,4,2,3))

# where we here assume the block structure to be
# c(1, 4, 2, 3), which means that the first LD "block"
# only consists of the first SNP, the second LD block
# consists of the following four SNPs in trio.bin,
```

```
# the third block of the following two SNPs,
# and the last block of the last three SNPs.
# set.seed() is specified to make the results reproducible.

# For the application of trio logic regression, some
# parameters of trio logic regression are changed
# to make the following example faster.
my.control <- lrControl(start=1, end=-3, iter=1000, output=-4)

# Please note typically you should consider much more
# than 1000 iterations (usually, at least a few hundred
# thousand).

# Trio regression can then be applied to the trio data in
# trio.ped1 by
lr.out <- trioLR(trio.bin, control=my.control, rand=9876543)

# where we specify rand just to make the results reproducible.

# A null model permutation test can be performed by
trio.permTest(lr.out)

# The conditional permutation test can be performed by
trio.permTest(lr.out, conditional = TRUE)
```

`trio.power` *Power and sample size calculation*

Description

Computes power for genotypic TDT, allelic TDT or Score test given n trios or required sample size to gain given power.

Usage

```
trio.power(maf = 0.5, RR = 1.5, alpha = 5*10^(-8), n = NULL, beta = NULL,
 model = c("additive", "dominant", "recessive"), test = c("gTDT", "Score", "aTDT"))

## S3 method for class 'trio.power'
print(x,digits=4,...)
```

Arguments

`maf`	a numeric vector of population frequencies of a mutant allele.
`RR`	a numeric vector of the assumed relative risks for an individual getting a disease with 1 (in case of recessive model 2) mutant alleles compared to the risk of individuals carnying 0 mutant alleles.
`alpha`	a numeric vector of significance levels (Type I Error probability).
`n`	a numeric vector containing number of trios in a study. Must be filled for power calculation. Must not be NULL for sample size calculation.
`beta`	the desired power of the test. Must be filled for power calculation. Must not be NULL for sample size calculation.

model	a character containing the genotypic model assumed. Possible values are "additive", "dominant" and "recessive". In case of test="aTDT", the standard multiplicative model will be considered. Abbreviations are allowed. Thus, e.g., model = "dom" will fit a dominant model, and model = "r" a recessive model.
test	the chosen test. Must be "aTDT", "gTDT" or "Score". Abbreviations are allowed. Thus, e.g., test = "g" will perform a genotypic TDT, and test = "S" a Score test.
x	an object of class trio.power.
digits	number of digits that should be printed.
...	ignored

Details

Power and sample size calculation is derived on Knapp (1999). The power or the sample size will be calculated for all combinations of p, RR, alpha, test, model and n or beta.

Value

An object of class trio.power containing the following numeric values or vectors, respectively:

model	the chosen model
size	In case of sample size calculation: calculated sample sizes
beta	In case of sample size calculation: desired power
n	In case of power calculation: given number of trios
power	In case of power calculation: calculated power
alpha	Type I error
test	the chosen test
RR	the relative risks assumed
p	the assumed allele frequency
calc	the type of calculation

Author(s)

Christoph Neumann

References

Knapp, M. (1999). A Note on Power Approximations for the Transmission/Disequilibrium Test. *American Journal of Human Genetics*, 64, 1177-1185.

Neumann, C., Taub, M.A., Younkin, S.G., Beaty, T.H., Ruczinski, I., Schwender, H. (2014). Analytic Power and Sample Size Calculation for the Genotypic Transmission/Disequilibrium Test in Case-Parent Trio Studies. Submitted.

Examples

```
# The required samples size to reach of power
# of 0.8 when testing SNPs with minor allele
# frequencies of 0.1 and 0.2 with an additive
# or dominant genotypic TDT and score test
# can be determined by
trio.power(maf = c(0.1, 0.2), beta = 0.8, model = c("add", "dom"))
```

`trio.prepare` *Generate Trio Data Format Suitable for Trio Logic Regression*

Description

This function transforms case-parent data into a format suitable as input for trio logic regression. The function can also be used for the imputation of missing genotypes in case-parent data, while taking the existing SNP block structure into account.

Usage

```
trio.prepare(trio.dat, freq=NULL, blocks=NULL, logic=TRUE, ...)
```

Arguments

`trio.dat`	An object returned from the function `trio.check`.
`freq`	An optional data frame specifying haplotype blocks and frequencies. For an example, see the data frame `simuBkMap` contained in this package. If provided, the following argument `blocks` will be ignored. The object must have three columns in the following order: block identifiers (`key`), haplotypes (`hap`), and haplotype frequencies (`freq`). The block identifiers must be unique for each block. For each block, the haplotypes must be encoded as a string of the integers 1 and 2, where 1 refers to the major allele and 2 refers to the minor allele. The respective haplotype frequencies will be normalized to sum one.
`blocks`	An optional vector of integers, specifying (in sequence) the lengths of the linkage disequilibrium blocks. The sum of these integers must be equal to the total numbers of SNPs in the data set used as input. Using the integer 1 for SNPs not contained in LD blocks is required if this argument is used. If both arguments `freq` and `blocks` are `NULL`, complete linkage equilibrium is assumed (i.e., no correlation between the genotypes).
`logic`	A logical value indicating whether the trio data are returned with genotypes in dominant and recessive coding, suitable as input for trio logic regression (`TRUE`), or if the imputed data should be returned in genotype format, using one variable per SNP (`FALSE`).
`...`	Optional arguments that can be passed to function `haplo.em`.

Details

To create the genotypes for the pseudo-controls it is necessary to take the LD structure of the SNPs into account. This requires information on the LD blocks. It is assumed that the user has already delineated the block structure according to his or her method of choice. The function `trio.prepare`, which operates on an output object of `trio.check`, accepts the block length information as an argument. If this argument is not specified, a uniform block length of 1 (i.e., no LD structure) is assumed. If the haplotype frequencies are not specified, they are estimated from the parents' genotypes using the function `haplo.em`. The function then returns a list that contains the genotype information in binary format, suitable as input for trio logic regression. Since trio logic regression requires complete data, the function `trio.prepare` also performs an imputation of the missing genotypes. The imputation is based on the estimated or supplied haplotype information.

Value

bin	A matrix suitable as input for trio logic regression. The first column specifies the cases and pseudo-controls as required by logic regression using conditional logistic regression (the integer 3 for the probands followed by three zeros indicating the pseudo-controls). The following columns specify the (possibly imputed) genotypes in dominant and recessive coding, with two binary variables for each SNP. This is returned only if `logic = TRUE`.
trio	A data frame with imputed SNPs in genotype format derived from the input. This is returned only if `logic = FALSE`.
miss	A data frame with five columns indicating the missing genotypes in the input object. The five columns of the data frame refer to the family id (`famid`), the individual id (`pid`), the genotype (`snp`), the row numbers (`r`), and the column numbers (`c`). This element will be `NULL` if there are no missing data.
freq	The estimated or supplied haplotype information, in the same format as described in the **Arguments** above.

Acknowledgments

Support was provided by NIH grants R01 DK061662 and HL090577.

Author(s)

Qing Li, mail2qing@yahoo.com

References

Li, Q., Fallin, M.D., Louis, T.A., Lasseter, V.K., McGrath, J.A., Avramopoulos, D., Wolyniec, P.S., Valle, D., Liang, K.Y., Pulver, A.E., and Ruczinski, I. (2010). Detection of SNP-SNP Interactions in Trios of Parents with Schizophrenic Children. Genetic Epidemiology, 34, 396-406.

See Also

`trio.check`, `haplo.em`

Examples

```
data(trio.data)
trio.tmp <- trio.check(dat=trio.ped1)
trio.bin <- trio.prepare(trio.dat=trio.tmp, blocks=c(1,4,2,3))
trio.bin$bin[1:8,]
```

---

trio.sim *Simulate Case-Parent Trios*

Description

`trio.sim` generates case-parents trios when the disease risk of children is specified by (possibly higher-order) SNP-SNP interactions. The SNP minor allele frequencies and/or haplotypes are specified by the user, as are the parameters in the logistic model that describes the disease risk. If `pi.usr` is specified, a specific type of model, namely the well-known Risch model, will be employed.

Usage

```
trio.sim(freq, interaction = "1R and 2D", prev = 1e-3, OR = 1, pi.usr = 0,
   n = 100, rep = 1, step.save = NULL, step.load = NULL, verbose = FALSE)
```

Arguments

freq	A data frame specifying haplotype blocks and frequencies. For an example, see the data frame simuBkMap contained in this package. If provided, the following argument blocks will be ignored. The object must have three columns in the following order: block identifiers (key), haplotypes (hap), and haplotype frequencies (freq). The block identifiers must be unique for each block. For each block, the haplotypes must be encoded as a string of the integers 1 and 2, where 1 refers to the major allele and 2 refers to the minor allele. The respective haplotype frequencies will be normalized to sum one.
interaction	A string that specifies the risk altering genotype interaction as a Boolean term, such as *"7D or 19R"*, or *"(not 10D) or 45D"*. Each locus can appear at most once in the string, and the the Boolean term *not* can appear at most once before each locus, and must be enclosed in parenthesis, e.g., *"(not 3D)"*. Therefore, strings such as *"not (not 3D)"* and *"not 3D or 5R"* are prohibited. Parenthesis are also used to unambiguously define the Boolean expression as a binary tree, i.e., every parent node has exact two children. For example Thus, a long string such as *"1R or 3D or 5R"* must be written as *"(1R or 3D) or 5R"* or as *"1R or (3D or 5R)"*, even though the parenthesis are technically redundant. There is also a limit on the size of the interactions, please see **Details** below.
prev	The prevalence of the disease in the simulated population among non-carriers (the "un-exposed" group).
OR	The odds ratio of disease in the simulated population, comparing carriers to non-carriers.
pi.usr	probability for an individual without the interaction to be affected
n	The number of case-parent trios simulated. The default is 100.
rep	The number of data set replicates generated. The default is 1.
step.save	The name of the binary file (without ".RData" extension) in which the object specifying the simulation mating tables and probabilities will be saved. The default value is NULL In that case, the object will not be saved for re-use in later run. See **Details**.
step.load	The name of an existing binary file (without ".RData" extension) in which the object specifying the simulation mating tables and probabilities have been saved (see above). The default value is NULL. In that case, a new object will be generated.
verbose	A logical value indicating whether or not to print information about memory and time usage.

Details

The function trio.sim simulates case-parent trio data when the disease risk of children is specified by (possibly higher-order) SNP-SNP interactions. The mating tables and the respective sampling probabilities depend on the haplotype frequencies (or SNP minor allele frequencies when the SNP does not belong to a block). This information is specified in the freq argument of the function. The

probability of disease is assumed to be described by the logistic term *logit(p) = a + b* I*[Interaction]*, where *a* = logit (`prev`) and *b* = log(`OR`), with `prev` and `OR` specified by the user. Note that at this point only data for two risk groups (carriers versus non-carriers) can be simulated. Since the computational demands for generating the mating is dependent on the number of loci involved in the interactions and the lengths of the LD blocks that contain these disease loci, the interaction term can only consist of up to six loci, not more than one of those loci per block, and haplotype (block) lengths of at most 5 loci.

Generating the mating tables and the respective sampling probabilities necessary to simulate case-parent trios can be very time consuming for interaction models involving three or more SNPs. In simulation studies, many replicates of similar data are usually required, and generating these sampling probabilities in each instance would be a large and avoidable computational burden (CPU and memory). The sampling probabilities depend foremost on the interaction term and the underlying haplotype frequencies, and as long as these remain constant in the simulation study, the mating table information and the sampling probabilities can be "recycled". This is done by storing the relevant information (denoted as "step-stone") as a binary R file in the working directory (using the argument `step.save`), and loading the binary file again in future simulations (using the argument `step.load`), speeding up the simulation process dramatically. It is even possible to change the parameters `prev` and `OR` (corresponding to *a* and *b* in the logistic model) in these additional simulations, as the sampling probabilities can be adjusted accordingly.

Value

A list of matrices, containing the simulated data sets, in genotype format (indicating the number of variant alleles), including family and subject identifiers.

Author(s)

Qing Li, mail2qing@yahoo.com

References

Li, Q., Fallin, M.D., Louis, T.A., Lasseter, V.K., McGrath, J.A., Avramopoulos, D., Wolyniec, P.S., Valle, D., Liang, K.Y., Pulver, A.E., and Ruczinski, I. (2010). Detection of SNP-SNP Interactions in Trios of Parents with Schizophrenic Children. Genetic Epidemiology, 34, 396-406.

See Also

`trio.prepare`

Examples

```
data(trio.data)
sim <- trio.sim(freq=simuBkMap, interaction="1R and 5R", prev=.001, OR=2, n=20, rep=1)
sim[[1]][1:6, 1:12]
```

---

trioFS *Trio Feature Selection*

Description

Performs a trioFS (trio Feature Selection) analysis as proposed by Schwender et al. (2011) based on bagging/subsampling with base learner trio logic regression (Li et al., 2011).

Usage

```
## Default S3 method:
trioFS(x, y, B = 20, nleaves = 5, replace = TRUE, sub.frac = 0.632,
    control = lrControl(), fast = FALSE, addMatImp = TRUE, addModels = TRUE,
    verbose = FALSE, rand = NA, ...)

## S3 method for class 'trioPrepare'
trioFS(x, ...)

## S3 method for class 'formula'
trioFS(formula, data, recdom = TRUE, ...)
```

Arguments

x	either an object of class trioPrepare, i.e. the output of trio.prepare, or a binary matrix consisting of zeros and ones. If the latter, then each column of x must correspond to a binary variable (e.g., codng for a dominant or a recessive effect of a SNP), and each row to a case or a pseudo-control, where each trio is represented by a block of four consecutive rows of x containing the data for the case and the three matched pseudo-controls (in this order) so that the first four rows of x comprise the data for the first trio, rows 5-8 the data for the second trio, and so on. Missing values are not allowed. A convenient way to generate this matrix is to use the function trio.prepare. Afterwards, trioLR can be directly applied to the output of trio.prepare.
y	a numeric vector specifying the case-pseudo-control status for the observations in x (if x is a binary matrix). Since in trio logic regression, cases are coded by a 3 and pseudo-controls by a 0, y is given by rep(c(3, 0, 0, 0), n.trios), where n.trios is the number of trios for which genotype data is stored in x. Thus, the length of y must be equal to the number of rows in x. No missing values are allowed in y. If not specified, y will be automatically generated.
B	number of bootstrap samples or subsamples used in trioFS
nleaves	maximum number of leaves, i.e.\ variables, in the logic tree considered in each of the B trio logic regression models (please note in trio logic regression the model consists only of one logic tree).
replace	should sampling of the trios be done with replacement? If TRUE, a Bootstrap sample of size n.trios is drawn from the n.trios trios in each of the B iterations. If FALSE, ceiling(sub.frac * n.trios) of the trios are drawn without replacement in each iteration.
sub.frac	a proportion specifying the fraction of trios that are used in each iteration to fit a trio logic regression model if replace = FALSE. Ignored if replace = TRUE.
control	a list of control parameters for the search algorithms and the logic trees considered when fitting the trio logic regression model, where the parameters for an MC logic regression are ignored. For details and the parameters, see lrControl, which is the function that should be used to specify control.
fast	should a greedy search be used instead of simulated annealing, i.e. the standard search algorithm in (trio) logic regression?
addMatImp	should the matrix containing the improvements due to the interactions in each of the iterations be added to the output, where the importance of each interaction is computed by the average over the B improvements due to this interaction?
addModels	should the B trio logic regression models be added to the output

verbose	should some comments on the progress the trioFS analysis be printed?
rand	positive integer. If specified, the random number generator is set into a reproducible state.
formula	an object of class formula describing the model that should be fitted.
data	a data frame containing the variables in the model. Each row of data must correspond to an observation, and each column to a binary variable (coded by 0 and 1) or a factor (for details, see recdom) except for the column comprising the response, where no missing values are allowed in data. For a description of the specification of the response, see y.
recdom	a logical value or vector of length ncol(data) comprising whether a SNP should be transformed into two binary dummy variables coding for a recessive and a dominant effect. If recdom is TRUE (and a logical value), then all factors/variables with three levels will be coded by two dummy variables as described in make.snp.dummy. Each level of each of the other factors (also factors specifying a SNP that shows only two genotypes) is coded by one indicator variable. If recdom isFALSE (and a logical value), each level of each factor is coded by an indicator variable. If recdom is a logical vector, all factors corresponding to an entry in recdom that is TRUE are assumed to be SNPs and transformed into two binary variables as described above. All variables corresponding to entries of recdom that are TRUE (no matter whether recdom is a vector or a value) must be coded either by the integers 1 (coding for the homozygous reference genotype), 2 (heterozygous), and 3 (homozygous variant), or alternatively by the number of minor alleles, i.e. 0, 1, and 2, where no mixing of the two coding schemes is allowed. Thus, it is not allowed that some SNPs are coded by 1, 2, and 3, and others are coded by 0, 1, and 2.
...	for the trioPrepare and the formula method, optional parameters to be passed to the low level function trioFS.default, i.e. all arguments of trioFS.default except for x and y. Otherwise, ignored.

Value

An object of class trioFS consisting of

vim	a numeric vector containing the values of the importance measure for the found interactions,
prop	a numeric vector consisting of the percentage of models that contain the respective found interactions,
primes	a character vector naming the found interactions,
param	a list of parameters used in the trioFS analysis, i.e. B, nleaves, and the sampling method,
mat.imp	if addMatImp = TRUE, a matrix containing the B improvements for each found interaction,
logreg.model	if addModel = TRUE, the B trio logic regression models,
inbagg	if addModel = TRUE, a list of length B in which each object specifies the trios used to fit the corresponding trio logic regression model.

Author(s)

Holger Schwender, <holger.schwender@udo.edu>

References

Li, Q., Fallin, M.D., Louis, T.A., Lasseter, V.K., McGrath, J.A., Avramopoulos, D., Wolyniec, P.S., Valle, D., Liang, K.Y., Pulver, A.E., and Ruczinski, I. (2010). Detection of SNP-SNP Interactions in Trios of Parents with Schizophrenic Children. *Genetic Epidemiology*, 34, 396-406.

Schwender, H., Bowers, K., Fallin, M.D., and Ruczinski, I. (2011). Importance Measures for Epistatic Interactions# in Case-Parent Trios. *Annals of Human Genetics*, 75, 122-132.

See Also

trioLR, print.trioFS, trio.prepare

Examples

```
# Load the simulated data.
data(trio.data)

# Prepare the data in trio.ped1 for a trioFS analysis
# by first calling
trio.tmp <- trio.check(dat = trio.ped1)

# and then applying
set.seed(123456)
trio.bin <- trio.prepare(trio.dat=trio.tmp, blocks=c(1,4,2,3))

# where we here assume the block structure to be
# c(1, 4, 2, 3), which means that the first LD "block"
# only consists of the first SNP, the second LD block
# consists of the following four SNPs in trio.bin,
# the third block of the following two SNPs,
# and the last block of the last three SNPs.
# set.seed() is specified to make the results reproducible.

# For the application of trioFS, some parameters of trio
# logic regression are changed to make the following example faster.
my.control <- lrControl(start=1, end=-3, iter=1000, output=-4)

# Please note typically you should consider much more
# than 1000 iterations (usually, at least a few hundred
# thousand).

# TrioFS can then be applied to the trio data in trio.ped1 by
fs.out <- trioFS(trio.bin, control=my.control, rand=9876543)

# where we specify rand just to make the results reproducible.
```

---

trioLR *Trio Logic Regression*

Description

Performs a trio logic regression analysis as proposed by Li et al. (2011), where trio logic regression is an adaptation of logic regression (Ruczinski et al., 2003) for case-parent trio data.

Usage

```
## Default S3 method:
trioLR(x, y, search = c("sa", "greedy", "mcmc"), nleaves = 5,
   penalty = 0, weights = NULL, control=lrControl(), rand = NA, ...)

## S3 method for class 'trioPrepare'
trioLR(x, ...)

## S3 method for class 'formula'
trioLR(formula, data, recdom = TRUE, ...)
```

Arguments

x	either an object of class trioPrepare, i.e. the output of trio.prepare, or a binary matrix consisting of zeros and ones. If the latter, then each column of x must correspond to a binary variable (e.g., coding for a dominant or a recessive effect of a SNP), and each row to a case or a pseudo-control, where each trio is represented by a block of four consecutive rows of x containing the data for the case and the three matched pseudo-controls (in this order) so that the first four rows of x comprise the data for the first trio, rows 5-8 the data for the seocnd trio, and so on. Missing values are not allowed. A convenient way to generate this matrix is to use the function trio.prepare. Afterwards, trioLR can be directly applied to the output of trio.prepare.
y	a numeric vector specifying the case-pseudo-control status for the observations in x (if x is the binary matrix). Since in trio logic regression, cases are coded by a 3 and pseudo-controls by a 0, y is given by rep(c(3, 0, 0, 0), n.trios), where n.trios is the number of trios for which genotype data is stored in x. Thus, the length of y must be equal to the number of rows in x. No missing values are allowed in y. If not specified, y will be automatically generated.
search	character string naming the search algorithm that should be used in the search for the best trio logic regression model. By default, i.e. search = "sa", simulated annealing, the standard search algorithm for a logic regression is used. In this case, depending on the length of nleaves, either one trio logic regression model is fitted or several trio logic regression models of different sizes are fitted. For details, see nleaves. Alternatively, a greedy search can be used by setting search = "greedy", or a MC logic regression analysis (Kooperberg and Ruczinski, 2005) for case-parent trio data can be performed by setting search = "mcmc".
nleaves	integer or vector of two integers specifying the maximum number of leaves, i.e.\ variables, in the logic tree of the trio logic regression model (please note in trio logic regression the model consists only of one logic tree). Must be a single integer, if search = "greedy" or search = "mcmc". If search = "sa", it can also be a vector of two integers, where the second integer must be larger than the first one. In this case, several trio logic regression models are fitted in which the maximum numbers of leaves range from nleaves[1] to nleaves[2].
penalty	a non-negative value for the penalty parameter used in logic regression. The penalty takes the form penalty times the number of leaves in the model. By default, larger models are not penalized. penalty is only relevant when one logic regression model is fitted.
weights	a numeric vector containing one weight for each trio considered in x. Thus, weights must contain nrow(x) / 4 positive values. By default, all trios are

equally weighted.

`control` a list of control parameters for the search algorithms and the logic tree considered when fitting a (trio) logic regression model. For these parameters, see `lrControl`, which is the function that should be used to specify `control`.

`rand` integer. If specified, the random number generator will be set into a reproducible state.

`formula` an object of class `formula` describing the model that should be fitted.

`data` a data frame containing the variables in the model. Each row of `data` must correspond to an observation, and each column to a binary variable (coded by 0 and 1) or a factor (for details, see `recdom`) except for the column comprising the response, where no missing values are allowed in `data`. For a description of the specification of the response, see `y`.

`recdom` a logical value or vector of length `ncol(data)` comprising whether a SNP should be transformed into two binary dummy variables coding for a recessive and a dominant effect. If `recdom` is `TRUE` (and a logical value), then all factors/variables with three levels will be coded by two dummy variables as described in `make.snp.dummy`. Each level of each of the other factors (also factors specifying a SNP that shows only two genotypes) is coded by one indicator variable. If `recdom` is`FALSE` (and a logical value), each level of each factor is coded by an indicator variable. If `recdom` is a logical vector, all factors corresponding to an entry in `recdom` that is `TRUE` are assumed to be SNPs and transformed into two binary variables as described above. All variables corresponding to entries of `recdom` that are `TRUE` (no matter whether `recdom` is a vector or a value) must be coded either by the integers 1 (coding for the homozygous reference genotype), 2 (heterozygous), and 3 (homozygous variant), or alternatively by the number of minor alleles, i.e. 0, 1, and 2, where no mixing of the two coding schemes is allowed. Thus, it is not allowed that some SNPs are coded by 1, 2, and 3, and others are coded by 0, 1, and 2.

`...` for the `trioPrepare` and the `formula` method, optional parameters to be passed to the low level function `trioLR.default`, i.e. all arguments of `trioLR.default` except for `x` and `y`. Otherwise, ignored.

Details

Trio logic regression is an adaptation of logic regression to case-parent trio data. Virtually all features for a standard logic regression analysis with the function `logreg` available in the `R` package `LogicReg` are also available for a trio logic regression analysis, either directly via `trioLR` or via the function `trio.permTest` for performing permutation tests.

For a detailed, comprehensive description on how to perform a logic regression analysis, and thus, a trio logic regression analysis, see the `Details` section of the help page for the function `logreg` in the `R` package `LogicReg`. For a detailed explanation on how to specify the parameters for simulated annealing, see the man page of the function `logreg.anneal.control` in the `R` package `LogicReg`.

Finally, an example for a trio logic regression analysis is given in the vignette `trio` available in the `R` package `trio`.

Value

An object of class `trioLR` composed of the same objects as an object of class `logreg`. For details, see the `Value` section of the function `logreg` from the `R` package `LogicReg`.

Author(s)

Holger Schwender, <holger.schwender@udo.edu>

References

Kooperberg, C. and Ruczinski, I. (2005). Identifying Interacting SNPs Using Monte Carlo Logic Regression. *Genetic Epidemiology*, 28, 157-170.

Li, Q., Fallin, M.D., Louis, T.A., Lasseter, V.K., McGrath, J.A., Avramopoulos, D., Wolyniec, P.S., Valle, D., Liang, K.Y., Pulver, A.E., and Ruczinski, I. (2010). Detection of SNP-SNP Interactions in Trios of Parents with Schizophrenic Children. *Genetic Epidemiology*, 34, 396-406.

Ruczinski, I., Kooperberg, C., and LeBlanc, M.L. (2003). Logic Regression. *Journal of Computational and Graphical Statistics*, 12, 475-511.

See Also

logreg, trio.prepare, trio.check, trio.permTest

Examples

```
# Load the simulated data.
data(trio.data)

# Prepare the data in trio.ped1 for a trio logic
# regression analysis by first calling
trio.tmp <- trio.check(dat = trio.ped1)

# and then applying
set.seed(123456)
trio.bin <- trio.prepare(trio.dat=trio.tmp, blocks=c(1,4,2,3))

# where we here assume the block structure to be
# c(1, 4, 2, 3), which means that the first LD "block"
# only consists of the first SNP, the second LD block
# consists of the following four SNPs in trio.bin,
# the third block of the following two SNPs,
# and the last block of the last three SNPs.
# set.seed() is specified to make the results reproducible.

# For the application of trio logic regression, some
# parameters of trio logic regression are changed
# to make the following example faster.
my.control <- lrControl(start=1, end=-3, iter=1000, output=-4)

# Please note typically you should consider much more
# than 1000 iterations (usually, at least a few hundred
# thousand).

# Trio regression can then be applied to the trio data in
# trio.ped1 by
lr.out <- trioLR(trio.bin, control=my.control, rand=9876543)

# where we specify rand just to make the results reproducible.
```

vcf2geno *Transformation of VFC File*

Description

Transforms a vcf file into a matrix in genotype format required by, e.g., the functions for computing the genotypic TDT.

Usage

```
vcf2geno(vcf, ped, none = "0/0", one = c("0/1"), both = "1/1", na.string = ".",
    use.rownames = FALSE, allowDifference = FALSE, removeMonomorphic = TRUE,
  removeNonBiallelic = TRUE, changeMinor = FALSE)
```

Arguments

vcf	a matrix resulting from reading a vcf file into R, or an object of class collapsedVCF (i.e. the output of, e.g., the function readVcf from the VariantAnnotation package). If use.rownames = FALSE, the column names of the genotype matrix must correspond to the personal IDs in ped (i.e. either the column pid of ped, if the entries in pid are unique, or otherwise, a combination of the columns famid and pid from ped, combined using an underscore). If use.rownames = TRUE, the column names of the genotype matrix specified by vcf must correspond to the row names of ped.
ped	a data frame containing the family information for the subjects in vcf (might also contain information for other subjects, see allowDifference). This data frame must contain the columns famid, pid, fatid, and motid comprising the family ID, the personal ID as well as the ID of the father and the mother, respectively.
none	a character string or vector specifying the coding for the homozygous reference genotype.
one	a character string or vector specifying the coding for the heterozygous genotype.
both	a character string or vector specifying the coding for the homozygous variant genotype.
na.string	a character string or vector specifying how missing values are coded in the vcf file.
use.rownames	a logical value specifying whether the row names of ped correspond to the sample names in vcf. For details, see vcf.
allowDifference	a logical value specifying whether ped and vcf are allowed to also contain samples not available in the respective other object. If FALSE, all samples in ped must also be available in vcf, and vice versa (matched as described in vcf). If TRUE, at least 10% of the samples must be contained in both vcf and ped.
removeMonomorphic	a logical value specifying whether monomorphic SNVs should be removed from the output.
removeNonBiallelic	a logical value specifying whether SNVs showing other genotypes than the ones specified by none, one, and both (which are, therefore, assumed to show more than two alleles) should be removed.

`changeMinor` a logical value specifying whether the coding of the genotypes should be changed for SNVs for which the default coding leads to a minor allele frequency larger than 0.5. The genotypes are coded by the number of minor alleles, i.e. the genotype(s) specified by `none` is coded by 0, the genotype(s) specified by `one` is coded by 1, and the genotype(s) specified by `both` is coded by 2. If for an SNV this leads to a minor allele frequency larger than 0.5 and `changeMinor = TRUE`, this 0, 1, 2-coding will be changed into a 2, 1, 0-coding.

Value

A matrix in genotype format required, e.g., by functions for performing different types of the genotypic TDT, such as `colTDT`.

Author(s)

Holger Schwender, <holger.schwender@udo.edu>

See Also

`colTDT`, `colGxG`, `colGxE`, `ped2geno`

Index

∗ **IO**
read.pedfile, 41
∗ **array**
allelicTDT, 2
colEMlrt, 4
colGxE, 6
colGxGPerms, 10
colNtrios, 11
colPOlrt, 13
colTDTmaxTest, 15
colTDTsam, 17
getLD, 21
ntrios2Dom, 26
poly4root, 33
probTDT, 40
scoreTDT, 44
tdt, 46
tdtGxG, 48
∗ **datagen**
trio.sim, 59
∗ **datasets**
trio.data, 53
∗ **design**
trio.power, 56
∗ **file**
read.pedfile, 41
∗ **hplot**
plot.getLD, 28
plot.LDblocks, 30
plot.trioLR, 31
print.trioFS, 36
∗ **htest**
allelicTDT, 2
colEMlrt, 4
colGxE, 6
colGxGPerms, 10
colPOlrt, 13
colTDTmaxTest, 15
colTDTsam, 17
probTDT, 40
scoreTDT, 44
tdt, 46
tdtGxG, 48
trio.permTest, 54
∗ **manip**
getMatPseudo, 23
ped2geno, 27
removeSNPs, 43
trio.check, 51
trio.prepare, 58
vcf2geno, 68
∗ **math**
poly4root, 33
∗ **models**
allelicTDT, 2
colEMlrt, 4
colGxE, 6
colGxGPerms, 10
colNtrios, 11
colPOlrt, 13
colTDTmaxTest, 15
colTDTsam, 17
probTDT, 40
tdt, 46
tdtGxG, 48
trio.permTest, 54
∗ **multivariate**
findLDblocks, 19
trioFS, 61
trioLR, 64
∗ **print**
print.colGxE, 34
print.trioFS, 36
print.trioLR, 38
∗ **regression**
trioFS, 61
trioLR, 64
∗ **tree**
trioFS, 61
trioLR, 64
∗ **univar**
getLD, 21
∗ **utilities**
lrControl, 24
print.colGxE, 34

allelicTDT, 2

colEMlrt, 4
colGxE, 6, *24*, *34, 35*, *46*, *69*
colGxEPerms *(colGxE)*, 6
colGxG, *10, 11*, *69*
colGxG *(tdtGxG)*, 48
colGxGPerms, 10
colMAFtrio *(removeSNPs)*, 43
colNtrios, *7*, *9*, 11, *26, 27*, *47*
colPOlrt, 13
colTAT *(colPOlrt)*, 13
colTDT, *3*, *5*, *9*, *12, 13*, *15*, *19*, *24*, *27*, *46*, *69*
colTDT *(tdt)*, 46
colTDT2way, *24*, *46*
colTDT2way *(tdtGxG)*, 48
colTDTebam *(colTDTsam)*, 17
colTDTinter2way *(tdtGxG)*, 48
colTDTmaxStat, *19*
colTDTmaxStat *(colTDTmaxTest)*, 15
colTDTmaxTest, 15
colTDTsam, 17
compPermTDT2way *(colGxGPerms)*, 10

denspr, *18*

ebam, *18, 19*

fastGxG *(tdtGxG)*, 48
fastGxGrec *(tdtGxG)*, 48
findLDblocks, 19, *22, 23*, *31*, *54*
freq.hap *(trio.data)*, 53

getGxEstats *(print.colGxE)*, 34
getLD, *20, 21*, 21, *30*, *54*
getLDlarge, *20*
getLDlarge *(getLD)*, 21
getMatPseudo, 23
gtdt.ebam *(colTDTsam)*, 17
gtdt.stat *(colTDTsam)*, 17

haplo.em, *58, 59*

image, *29*

LDdata *(trio.data)*, 53
logreg, *67*
lrControl, 24, *55*, *62*, *66*

make.snp.dummy, *63*, *66*
mat.test *(trio.data)*, 53

ntrios2Dom, *12, 13*, 26
ntrios2Rec, *12*
ntrios2Rec *(ntrios2Dom)*, 26

orderSNPs *(removeSNPs)*, 43

ped2geno, *3–6*, *9, 10*, *12*, *14–17*, *20*, *22, 23*, *27*, *42, 43*, *45*, *47–50*, *69*
plot.getLD, *23*, 28, *31*
plot.getLDlarge *(plot.getLD)*, 28
plot.LDblocks, *21*, *30*, 30
plot.trioFS *(print.trioFS)*, 36
plot.trioLR, 31
poly4root, 33
poly4rootMat *(poly4root)*, 33
print.aTDT *(allelicTDT)*, 2
print.colEMlrt *(colEMlrt)*, 4
print.colGxE, 34
print.colGxEunstruct *(print.colGxE)*, 34
print.colTDT *(tdt)*, 46
print.colTDTepi *(tdtGxG)*, 48
print.LDblock *(findLDblocks)*, 19
print.maxScoreTrio *(scoreTDT)*, 44
print.maxStatTrio *(colTDTmaxTest)*, 15
print.maxTestTrio *(colTDTmaxTest)*, 15
print.polrt *(colPOlrt)*, 13
print.scoreGxE *(scoreTDT)*, 44
print.scoreTDT *(scoreTDT)*, 44
print.tat *(colPOlrt)*, 13
print.tdt *(tdt)*, 46
print.tdtEpi *(tdtGxG)*, 48
print.trio.power *(trio.power)*, 56
print.trioFS, 36, *64*
print.trioLR, 38
prob.mat.test *(trio.data)*, 53
probTDT, 40, *53*

qingEnv *(trio.prepare)*, 58

read.pedfile, *20*, *22*, 41, *43*
removeSNPs, 43
removeTrios *(removeSNPs)*, 43

sam, *18, 19*
scoreGxE *(scoreTDT)*, 44
scoreGxG *(scoreTDT)*, 44
scoreMaxStat *(scoreTDT)*, 44
scoreTDT, 44
simuBkMap *(trio.data)*, 53
splitBlocks *(findLDblocks)*, 19
step3way *(trio.data)*, 53

tdt, *17*, *28*, *41*, 46, *50*, *53*
tdt2way, *28*, *48*, *53*
tdt2way *(tdtGxG)*, 48
tdtGxG, 48
trio.check, *28*, *47*, *49*, 51, *53*, *58, 59*, *67*
trio.data, 53
trio.gen.err *(trio.data)*, 53

trio.gen1 *(*trio.data*)*, 53
trio.gen2 *(*trio.data*)*, 53
trio.ped.err *(*trio.data*)*, 53
trio.ped1 *(*trio.data*)*, 53
trio.ped2 *(*trio.data*)*, 53
trio.permTest, 54, *67*
trio.power, 56
trio.prepare, *20*, *51–53*, 58, *61, 62*, *64, 65*, *67*
trio.sim, *54*, 59
trioFS, *20*, *24*, *36, 37*, 61
trioLR, *20*, *24*, *32*, *38, 39*, *54, 55*, *64*, 64

uniroot, *50*

vcf2geno, 68