Package 'AssotesteR'

February 19, 2015

Type Package

Title Statistical Tests for Genetic Association Studies

Version 0.1-10

Date 2013-12-29

Author Gaston Sanchez

Maintainer Gaston Sanchez <gaston.stat@gmail.com>

Depends mvtnorm

Description R package with statistical tests and methods for genetic association studies with emphasis on rare variants and binary (dichotomous) traits

License GPL (>= 3)

URL http://www.gastonsanchez.com

LazyLoad yes

Collate 'ASCORE.Ord.R' 'ASCORE.R' 'ASSU.Ord.R' 'ASSU.R' 'ASSUW.Ord.R' 'ASSUW.R' 'ASUM.Ord.R' 'ASUM.R' 'AssotesteR-internal.R' 'BST.R' 'CALPHA.R' 'CARV.R' 'CAST.R' 'CMAT.R' 'CMC.R' 'GDBR.R' 'MULTI.R' 'ORWSS.R' 'RARECOVER.R' 'RBT.R' 'RVT1.R' 'RVT2.R' 'RWAS.R' 'SCORE.R' 'SEQSUM.R' 'SKAT.R' 'SSU.R' 'SSUW.R' 'SUM.R' 'TTEST.R' 'UMINP.R' 'VT.R' 'WSS.R' 'WST.R' 'gdbr_IBS.R' 'gdbr_wIBS.R' 'kernel_IBS.R' 'kernel_twowayx.R' 'kernel_wIBS.R' 'my_check.R' 'print.assoctest.R'

NeedsCompilation yes

Repository CRAN

Date/Publication 2013-12-19 00:09:14

R topics documented:

AssotesteR-package 2
ASCORE 3
ASSU 5
ASSUW 7

ASUM 8
BST 10
CALPHA 12
CARV 14
CAST 16
CMAT 18
CMC 20
GDBR 22
genodata 24
MULTI 25
ORWSS 26
RARECOVER 28
RBT 30
RVT1 31
RVT2 33
RWAS 35
SCORE 36
SEQSUM 38
SKAT 40
SSU 42
SSUW 44
SUM 45
TTEST 47
UMINP 48
VT 50
WSS 52
WST 53

Index **56**

`AssotesteR-package` *Statistical Tests for Genetic Association Studies*

Description

Statistical tests and methods for genetic association studies with emphasis on rare variants, binary (dichotomous) traits, and missing genotypes

Details

Package:	AssotesteR
Type:	Package
Version:	0.1-9
Date:	2013-01-23
Depends:	mvtnorm
License:	GPL (>= 3)
URL:	http://www.gastonsanchez.com/assotester
LazyLoad:	yes

The package contains several methods on testing for association between phenotype and genotype data with no covariates

Author(s)

Author: Gaston Sanchez

Maintainer: Gaston Sanchez <gaston.sanchez@berkeley.edu>

References

http://www.gastonsanchez.com/assotester

`ASCORE` *ASCORE: Adaptive Score Test*

Description

The Adaptive Score test has been proposed by Hand and Pan (2010) in an attempt to overcome some of the drawbacks of the Score test (from logistic regression) by extending the idea of the adaptive Neymans test. The approach behind the adaptive test is to use the the first components of the U-score vector in order to construct a test statistic

Usage

```
ASCORE(y, X, perm = 100)

ASCORE.Ord(y, X, perm = 100)
```

Arguments

`y`	numeric vector with phenotype status: 0=controls, 1=cases. No missing data allowed
`X`	numeric matrix or data frame with genotype data coded as 0, 1, 2. Missing data is allowed
`perm`	positive integer indicating the number of permutations (100 by default)

Details

`ASCORE` gives the normal (unordered) test.
`ASCORE.Ord` gives the ordered (decreasing) test.

There is no imputation for the missing data. Missing values are simply ignored in the computations.

Value

An object of class `"assoctest"`, basically a list with the following elements:

`ascore.stat`	ascore statistic
`perm.pval`	permuted p-value
`args`	descriptive information with number of controls, cases, variants, and permutations
`name`	name of the statistic

Author(s)

Gaston Sanchez

References

Han F, Pan W (2010) A Data-Adaptive Sum Test for Disease Association with Multiple Common or Rare Variants. *Human Heredity*, **70**: 42-54

Pan W, Shen X (2011) Adaptive Tests for Association of Rare Variants. *Genetic Epidemiology*, **35**: 381-388

See Also

SCORE

Examples

```
## Not run:

# number of cases
cases = 500

# number of controls
controls = 500

# total (cases + controls)
total = cases + controls

# phenotype vector
phenotype = c(rep(1, cases), rep(0, controls))

# genotype matrix with 10 variants (random data)
set.seed(123)
genotype = matrix(rbinom(total*10, 2, 0.05), nrow=total, ncol=10)

# apply ASCORE with 500 permutations
myascore = ASCORE(phenotype, genotype, perm=500)
myascore

# apply ASCORE.Ord with 500 permutations
```

```
  myascoreord = ASCORE.Ord(phenotype, genotype, perm=500)
  myascoreord

## End(Not run)
```

ASSU *ASSU: Adaptive Sum of Squared Score U Statistic*

Description

The adaptive SSU test has been proposed by Han and Pan (2010) in an attempt to overcome some of the drawbacks of the SSU test, by extending the idea of the adaptive Neyman's test (Fan, 1996). The approach behind the adaptive test is to use the U-statistics of the score test (from logistic regression models) in order to construct a statistic with the first components of the score vector U.

Usage

```
ASSU(y, X, perm = 100)

ASSU.Ord(y, X, perm = 100)
```

Arguments

y	numeric vector with phenotype status: 0=controls, 1=cases. No missing data allowed
X	numeric matrix or data frame with genotype data coded as 0, 1, 2. Missing data is allowed
perm	positive integer indicating the number of permutations (100 by default)

Details

`ASSU` gives the normal (unordered) test.
`ASSU.Ord` gives the ordered (decreasing) test.

There is no imputation for the missing data. Missing values are simply ignored in the computations.

Value

An object of class `"assoctest"`, basically a list with the following elements:

assu.stat	assu statistic
perm.pval	permuted p-value
args	descriptive information with number of controls, cases, variants, and permutations
name	name of the statistic

Author(s)

Gaston Sanchez

References

Han F, Pan W (2010) A Data-Adaptive Sum Test for Disease Association with Multiple Common or Rare Variants. *Human Heredity*, **70**: 42-54

Pan W, Shen X (2011) Adaptive Tests for Association of Rare Variants. *Genetic Epidemiology*, **35**: 381-388

See Also

SSU

Examples

```
## Not run:

# number of cases
cases = 500

# number of controls
controls = 500

# total (cases + controls)
total = cases + controls

# phenotype vector
phenotype = c(rep(1, cases), rep(0, controls))

# genotype matrix with 10 variants (random data)
set.seed(123)
genotype = matrix(rbinom(total*10, 2, 0.05), nrow=total, ncol=10)

# apply ASSU with 500 permutations
myassu = ASSU(phenotype, genotype, perm=500)
myassu

# apply ASSU.Ord with 500 permutations
myassuord = ASSU.Ord(phenotype, genotype, perm=500)
myassuord

## End(Not run)
```

ASSUW *ASSUW: Adaptive Sum of Squared Score U Statistic*

Description

The adaptive Weighted Score test has been proposed by Han and Pan (2010) in an attempt to overcome some of the drawbacks of the SSUW test, by extending the idea of the adaptive Neyman's test (Fan, 1996). The approach behind the adaptive test is to use the U-statistics of the score test (from logistic regression models) in order to construct a statistic with the first components of the score vector U.

Usage

```
ASSUW(y, X, perm = 100)

ASSUW.Ord(y, X, perm = 100)
```

Arguments

y	numeric vector with phenotype status: 0=controls, 1=cases. No missing data allowed
X	numeric matrix or data frame with genotype data coded as 0, 1, 2. Missing data is allowed
perm	positive integer indicating the number of permutations (100 by default)

Details

`ASSUW` gives the normal (unordered) test.
`ASSUW.Ord` gives the ordered (decreasing) test.

There is no imputation for the missing data. Missing values are simply ignored in the computations.

Value

An object of class `"assoctest"`, basically a list with the following elements:

assuw.stat	assuw statistic
perm.pval	permuted p-value
args	descriptive information with number of controls, cases, variants, and permutations
name	name of the statistic

Author(s)

Gaston Sanchez

References

Han F, Pan W (2010) A Data-Adaptive Sum Test for Disease Association with Multiple Common or Rare Variants. *Human Heredity*, **70**: 42-54

Pan W, Shen X (2011) Adaptive Tests for Association of Rare Variants. *Genetic Epidemiology*, **35**: 381-388

See Also

SSUW

Examples

```
## Not run:

# number of cases
cases = 500

# number of controls
controls = 500

# total (cases + controls)
total = cases + controls

# phenotype vector
phenotype = c(rep(1, cases), rep(0, controls))

# genotype matrix with 10 variants (random data)
set.seed(123)
genotype = matrix(rbinom(total*10, 2, 0.05), nrow=total, ncol=10)

# apply ASSUW with 500 permutations
myassuw = ASSUW(phenotype, genotype, perm=500)
myassuw

# apply ASSUW.Ord with 500 permutations
myassuword = ASSUW.Ord(phenotype, genotype, perm=500)
myassuword

## End(Not run)
```

ASUM *ASUM: Adaptive Sum Statistic*

Description

The adaptive Adaptive Sum test has been proposed by Han and Pan (2010) in an attempt to overcome some of the drawbacks of the SUM test, by extending the idea of the adaptive Neyman's test

(Fan, 1996). The approach behind the adaptive test is to use the U-statistics of the score test (from logistic regression models) in order to construct a statistic with the first components of the score vector U.

Usage

```
ASUM(y, X, perm = 100)

ASUM.Ord(y, X, perm = 100)
```

Arguments

y	numeric vector with phenotype status: 0=controls, 1=cases. No missing data allowed
X	numeric matrix or data frame with genotype data coded as 0, 1, 2. Missing data is allowed
perm	positive integer indicating the number of permutations (100 by default)

Details

`ASUM` gives the normal (unordered) test.
`ASUM.Ord` gives the ordered (decreasing) test.

There is no imputation for the missing data. Missing values are simply ignored in the computations.

Value

An object of class `"assoctest"`, basically a list with the following elements:

asum.stat	asum statistic
perm.pval	permuted p-value
args	descriptive information with number of controls, cases, variants, and permutations
name	name of the statistic

Author(s)

Gaston Sanchez

References

Han F, Pan W (2010) A Data-Adaptive Sum Test for Disease Association with Multiple Common or Rare Variants. *Human Heredity*, **70**: 42-54

Pan W, Shen X (2011) Adaptive Tests for Association of Rare Variants. *Genetic Epidemiology*, **35**: 381-388

See Also

SUM

Examples

```
## Not run:

# number of cases
cases = 500

# number of controls
controls = 500

# total (cases + controls)
total = cases + controls

# phenotype vector
phenotype = c(rep(1, cases), rep(0, controls))

# genotype matrix with 10 variants (random data)
set.seed(123)
genotype = matrix(rbinom(total*10, 2, 0.05), nrow=total, ncol=10)

# apply ASUM with 500 permutations
myasum = ASUM(phenotype, genotype, perm=500)
myasum

# apply ASUM.Ord with 500 permutations
myasumord = ASSU.Ord(phenotype, genotype, perm=500)
myasumord

## End(Not run)
```

BST *BST: Bayesian Score Test*

Description

BST is based on the test statistic of Goeman et al (2005) following a general empirical Bayes method to test on a large number of parameters in a logistic regression model. BST is closely related to the usual Score test, although it assumes an empirical Bayesian model with an independent prior on the genetic variant effects. The null distribution of the BST statistic is unknown and has to be estimated by permutation.

Usage

```
BST(y, X, perm = 100)
```

Arguments

y	numeric vector with phenotype status: 0=controls, 1=cases. No missing data allowed
X	numeric matrix or data frame with genotype data coded as 0, 1, 2. Missing data is allowed
perm	positive integer indicating the number of permutations (100 by default)

Details

The BST statistic does not offer an asymptotic p-value. Permutations are required

There is no imputation for the missing data. Missing values are simply ignored in the computations.

Value

An object of class "assoctest", basically a list with the following elements:

bst.stat	bst statistic
perm.pval	permuted p-value
args	descriptive information with number of controls, cases, variants, and permutations
name	name of the statistic

Author(s)

Gaston Sanchez

References

Goeman JJ, van de Geer SA, van Houwelingen HC (2006) Testing against a high dimensional alternative. *Journal of the Royal Statistical Society*, **68**: 477-493

Chapman J, Whittaker J (2008) Analysis of Multiple SNPs in a Candidate Gene or Region. *Genetic Epidemiology*, **32**: 560-566

Pan W (2009) Asymptotic Tests of Association with Multiple SNPs in Linkage Disequilibrium. *Genetic Epidemiology*, **33**: 497-507

See Also

SCORE

Examples

```
## Not run:

# number of cases
cases = 500

# number of controls
controls = 500

# total (cases + controls)
total = cases + controls

# phenotype vector
phenotype = c(rep(1,cases), rep(0,controls))

# genotype matrix with 10 variants (random data)
set.seed(123)
genotype = matrix(rbinom(total*10, 2, 0.05), nrow=total, ncol=10)

# apply BST with 500 permutations
mybst = BST(phenotype, genotype, perm=500)
mybst

## End(Not run)
```

CALPHA *CALPHA: C-alpha Score Test*

Description

The C-alpha score-test of Neyman and Scott (1966) has been proposed to be used in association astudies by Neale et al (2011) in order to test the observed distributino of rare variants in cases versis controls. Under the null hypothesis of no association between the variants and the phenotype, C-alpha assumes that the distribution of counts (copies of an observed variant) should follow a binomial distribution. The C-alpha test statistic contrasts the variance of each observed count with the expected variance, assuming the binomial distribution. Under the null hypothesis, the test statistic follows a standard normal distribution

Usage

```
CALPHA(y, X, perm = NULL)
```

Arguments

y	numeric vector with phenotype status: 0=controls, 1=cases. No missing data allowed
X	numeric matrix or data frame with genotype data coded as 0, 1, 2. Missing data is allowed
perm	positive integer indicating the number of permutations (NULL by default)

Details

There is no imputation for the missing data. Missing values are simply ignored in the computations.

Value

An object of class `"assoctest"`, basically a list with the following elements:

`calpha.stat`	c-alpha statistic
`asym.pval`	asymptotic p-value
`perm.pval`	permuted p-value; only when `perm` is used
`args`	descriptive information with number of controls, cases, variants, and permutations
`name`	name of the statistic

Author(s)

Gaston Sanchez

References

Neyman J, Scott E (1966) On the use of c-alpha optimal tests of composite hypothesis. *Bulletin of the International Statistical Institute*, **41**: 477-497

Neale BM, Rivas MA, Voight BF, Altshuler D, Devlin B, Orho-Melander M, Kathiresan S, Purcell SM, Roeder K, Daly MJ (2011) Testing for an unusual distribution of rare variants. *PLoS Genetics*, **7(3)**: e1001322

Basu S, Pan W (2011) Comparison of Statistical Tests for Disease Association With Rare Variants. *Genetic Epidemiology*, **35(7)**: 606-619

Examples

```
## Not run:

# number of cases
cases = 500

# number of controls
controls = 500

# total (cases + controls)
total = cases + controls

# phenotype vector
phenotype = c(rep(1,cases), rep(0,controls))

# genotype matrix with 10 variants (random data)
set.seed(123)
genotype = matrix(rbinom(total*10, 2, 0.10), nrow=total, ncol=10)
```

```
  # apply CALPHA with 500 permutations
  mycalpha = CALPHA(phenotype, genotype, perm=500)

  # this is what we get
  mycalpha

## End(Not run)
```

CARV *CARV: Comprehrensive Approach to Analyzing Rare Variants*

Description

The CARV method has been proposed by Hoffmann et al (2010) as an approach that determines an optimal grouping of rare variants while avoiding assumptions required by other methods for grouping such variants. The idea behind CARV is to try multiple models for rare variants, since prior information is generally not very accurate. Statistical significance is obtained by permutation.

Usage

```
CARV(y, X, waf = FALSE, signs = FALSE, approach = "hard", maf = 0.05, perm = 100)
```

Arguments

y	numeric vector with phenotype status: 0=controls, 1=cases. No missing data allowed
X	numeric matrix or data frame with genotype data coded as 0, 1, 2. Missing data is allowed
waf	logical value to indicate whether weights for the variants should be calculated as in the WSS method (default waf=FALSE). See deatils below
signs	logical value to indicate whether signs for the variants should be calculated based on allele prevalence (signs=FALSE by default). See details below
approach	character string to indicate the type of approach to be used for variable selection; i.e. whether each variant belongs in the model for variable selection (approach="hard" by default). Possible options are "hard", "variable", and "stepup". See details below
maf	numeric value between 0 and 1 to indicate the threshold of minor allele frequency for rare variants. Only used when approach="hard"
perm	positive integer indicating the number of permutations (100 by default)

Details

The argument `waf` is used to specify weights of the variants in order to incorporate allele frequency information. When `waf=FALSE`, all variants have a constant unit weight. When `waf=TRUE`, the weights are calculated as in the function `WSS`, that is, weights are the inverse variance of allele frequency in controls.

The argument `signs` is used to specify the direction of the variant effect (deleterious or protective). When `signs=FALSE`, all variants have a positive sign indicating a likely deleterious effect. When `signs=TRUE`, all rare alleles that are more prevalent in controls than cases will have an associated sk=-1 (see paper in the reference); conversely, all the rare alleles that are more prevalent in cases than controls will have an associated sk=1.

The argument `approach` is used to specify whether the alle belongs in the model for variable selection. When `approach="hard"`, only variants below the predefined `maf` are included in the analysis. When `approach="variable"`, a variable threshold approach is used as in the `VT` method: all possible minor allele frequencies are considered, selecting the maximum statistic among all of them. When `approach="stepup"`, the step-up strategy described in Hoffman et al (2010) is applied.

There is no imputation for the missing data. Missing values are simply ignored in the computations.

Value

An object of class `"assoctest"`, basically a list with the following elements:

`carv.stat`	carv statistic
`perm.pval`	permuted p-value
`args`	descriptive information with number of controls, cases, variants, permutations, waf, signs, approach, and maf
`name`	name of the statistic

Author(s)

Gaston Sanchez

References

Hoffmann TJ, Marini NJ, Witte JS (2010) Comprehensive Approach to Analyzing Rare Genetic Variants. *PLoS One*, **5(11)**: e13584

See Also

`RARECOVER`

Examples

```
## Not run:

# number of cases
cases = 500

# number of controls
controls = 500

# total (cases + controls)
total = cases + controls

# phenotype vector
phenotype = c(rep(1, cases), rep(0, controls))

# define genotype matrix with 10 variants (random data)
set.seed(1234)
genotype = matrix(rbinom(total*10, 2, 0.051), nrow=total, ncol=10)

# apply CARV with "hard" approach and maf=0.05
mycarv1 = CARV(phenotype, genotype, waf=FALSE, signs=FALSE,
   approach="hard", maf=0.05, perm=500)
mycarv1

# apply CARV with "variable" approach and waf=TRUE
mycarv2 = CARV(phenotype, genotype, waf=TRUE, signs=FALSE,
   approach="variable", perm=500)
mycarv2

# apply CARV with "stepup" approach, waf=TRUE, and signs=TRUE
mycarv3 = CARV(phenotype, genotype, waf=TRUE, signs=TRUE,
   approach="stepup", perm=500)
mycarv3


## End(Not run)
```

CAST *CAST: Cohort Allelic Sums Test*

Description

CAST is a pooled association test applied to discover if the difference in the sums of allelic mutation frequencies in case and control cohorts is greater than would be expected by chance. CAST works by first collapsing the genotypes across rare variants to generate a super-variant. It then tests the association between the trait and this new super-variant.

Usage

```
CAST(y, X, maf = 0.05,  test = "fisher")
```

Arguments

y	numeric vector with phenotype status: 0=controls, 1=cases. No missing data allowed
X	numeric matrix or data frame with genotype data coded as 0, 1, 2. Missing data is allowed
maf	numeric value indicating the minor allele frequency threshold for rare variants (maf=0.05 by default)
test	character string indicating the type of test to be applied. Possible values are "fisher" and "chisq" (test="fisher" by default)

Details

If no variants are below the specified maf threshold, the function will stop and return an error message

The argument test="fisher" involves a fisher exact test. Conversely, test="chisq" indicates a chi-square test.

There is no imputation for the missing data. Missing values are simply ignored in the computations.

Value

An object of class "assoctest", basically a list with the following elements:

cast.stat	cast statistic
asym.pval	asymptotic p-value
args	descriptive information with number of controls, cases, variants, maf, and applied test
name	name of the statistic

Author(s)

Gaston Sanchez

References

Morgenthaler S, Thilly WG (2007) A strategy to discover genes that carry multi-allelic or mono-allelic risk for common diseases: A cohort allelic sums test (CAST). *Mutation Research*, **615**: 28-56

See Also

link{TTEST}

Examples

```
## Not run:

# number of cases
cases = 500

# number of controls
controls = 500

# total (cases + controls)
total = cases + controls

# phenotype vector
phenotype = c(rep(1,cases), rep(0,controls))

# genotype matrix with 10 variants (random data)
set.seed(1234)
genotype = matrix(rbinom(total*10, 2, 0.051), nrow=total, ncol=10)

# apply CAST with fisher exact test
mycast1 = CAST(phenotype, genotype, maf=0.05, test = "fisher")
mycast1

# apply CAST with chi-square test
mycast2 = CAST(phenotype, genotype, maf=0.05, test = "chisq")
mycast2

## End(Not run)
```

CMAT *CMAT: Cumulative Minor Allele Test*

Description

CMAT is a pooling method proposed by Zawistowski et al (2010). CMAT works by comparing weighted minor-allele counts (for cases and controls) against the weighted major-allele counts (for cases and controls). Although the CMAT test statistic is based on a chi-square statistic, it does not follow a known distribution and its significance has to be determined by a permutation procedure.

Usage

```
CMAT(y, X, maf = NULL, weights = NULL, perm = 100)
```

Arguments

y	numeric vector with phenotype status: 0=controls, 1=cases. No missing data allowed
X	numeric matrix or data frame with genotype data coded as 0, 1, 2. Missing data is allowed

maf	optional numeric value to specify a threshold for the minor allele frequency of rare variants (NULL by default)
weights	optional vector of weights for the variants (NULL by default)
perm	positive integer indicating the number of permutations (100 by default)

Details

By default, argument maf=NULL meaning that no rare variants are selected

By default, argument weights=NULL but different values for the variants can be provided

Statistical significance is determined by permutation procedure

There is no imputation for the missing data. Missing values are simply ignored in the computations.

Value

An object of class "assoctest", basically a list with the following elements:

cmat.stat	cmat statistic
perm.pval	permuted p-value
args	descriptive information with number of controls, cases, variants, rare variants, maf, and permutations
name	name of the statistic

Author(s)

Gaston Sanchez

References

Zawistowski M, Gopalahrishnan S, Ding J, Li Y, Grimm S, Zollner S (2010) *The American Journal of Human Genetics*, **87**: 604-617

See Also

CMC, WSS

Examples

```
## Not run:

# number of cases
cases = 500

# number of controls
controls = 500
```

```
  # total (cases + controls)
  total = cases + controls

  # phenotype vector
  phenotype = c(rep(1,cases), rep(0,controls))

  # genotype matrix with 10 variants (random data)
  set.seed(1234)
  genotype = matrix(rbinom(total*10, 2, 0.051), nrow=total, ncol=10)

  # apply CMAT with 500 permutations
  mycmat1 = CMAT(phenotype, genotype, perm=500)
  mycmat1

  # apply CMAT with maf=0.05 and 500 permutations
  mycmat2 = CMAT(phenotype, genotype, maf=0.05, perm=500)
  mycmat2

## End(Not run)
```

CMC *CMC: Combined Multivariate and Collapsing Method*

Description

The CMC method is a pooling approach proposed by Li and Leal (2008) that uses allele frequencies to determine the partition of the variants into groups. After the rare variants are selected, they are collapsed into an indicator variable, and then a multivariate test such as Hotelling's T2 test is applied to the collection formed by the common variants and the collapsed super-variant.

Usage

```
CMC(y, X, maf = 0.05, perm = 100)
```

Arguments

y	numeric vector with phenotype status: 0=controls, 1=cases. No missing data allowed
X	numeric matrix or data frame with genotype data coded as 0, 1, 2. Missing data is allowed
maf	numeric value indicating the minor allele frequency threshold for rare variants (maf=0.05 by default)
perm	positive integer indicating the number of permutations (100 by default)

Details

Those variants with minor allele frequency below the specified `maf` threshold are collapsed into a single super variant

There is no imputation for the missing data. Missing values are simply ignored in the computations.

Value

An object of class `"assoctest"`, basically a list with the following elements:

`cmc.stat`	cmc statistic
`asym.pval`	asymptotic p-value
`perm.pval`	permuted p-value
`args`	descriptive information with number of controls, cases, variants, rare variants, maf threshold, and permutations
`name`	name of the statistic

Author(s)

Gaston Sanchez

References

Li B, Leal SM (2008) Methods for Detecting Associations with Rare Variants for Common Diseases: Application to Analysis of Sequence Data. *The American Journal of Human Genetics*, **83**: 311-321

See Also

`WSS`, `CMAT`, `TTEST`

Examples

```
## Not run:

# number of cases
cases = 500

# number of controls
controls = 500

# total (cases + controls)
total = cases + controls

# phenotype vector
phenotype = c(rep(1,cases), rep(0,controls))

# genotype matrix with 10 variants (random data)
set.seed(1234)
```

```
  genotype = matrix(rbinom(total*10, 2, 0.051), nrow=total, ncol=10)

  # apply CMC with maf=0.05 and 500 permutations
  mycmc = CMC(phenotype, genotype, maf=0.05, perm=500)
  mycmc

## End(Not run)
```

GDBR *GDBR: Genomic Distance-Based Regression*

Description

The Genomic Distance-Based Regression has been developed by Wessel et al (2006). This approach captures genotype information across multiple loci through a similarity measure between any two individuals. GDBR is unique in its regression analysis relating variation in the measure of genomic similarity to variation in their trait values. Note that this approach is computationally expensive.

Usage

```
GDBR(y, X, distance = "IBS", weights = NULL, perm = NULL)
```

Arguments

y	numeric vector with phenotype status: 0=controls, 1=cases. No missing data allowed
X	numeric matrix or data frame with genotype data coded as 0, 1, 2.
distance	character string indicating the type of distance to be used. Possible options are "IBS" or "wIBS" (distance="IBS" by default)
weights	optional numeric vector with weights for the genetic variants (NULL by default)
perm	positive integer indicating the number of permutations (NULL by default)

Details

The argument distance is used to specify the similarity distance. "IBS" indicates Identity-By-Share, "wIBS" indicates weighted IBS.

Value

An object of class "assoctest", basically a list with the following elements:

gdbr.stat	gdbr statistic
perm.pval	permuted p-value
args	descriptive information with number of controls, cases, variants, permutations, and selected distance
name	name of the statistic

Note

This method is computationally expensive

Author(s)

Gaston Sanchez

References

Wessel J, Schork NJ (2006) Generalized Genomic Distance-Based Regression Methodology for Multilocus Association Analysis. *The American Journal of Human Genetics*, **79**: 792-806

Schaid DJ (2010) Genomic Similarity and Kernel Methods I: Advancements by Building on Mathematical and Statistical Foundations. *The American Journal of Human Heredity*, **70**: 109-131

See Also

SKAT

Examples

```
## Not run:

# number of cases
cases = 250

# number of controls
controls = 250

# total (cases + controls)
total = cases + controls

# phenotype vector
phenotype = c(rep(1,cases), rep(0,controls))

# genotype matrix with 10 variants (random data)
set.seed(123)
genotype = matrix(rbinom(total*10, 2, 0.05), nrow=total, ncol=10)

# apply GDBR with 50 permutations
# (it takes some time to run the permutations!)
mygdbr = GDBR(phenotype, genotype, perm=50)
mygdbr

## End(Not run)
```

genodata *genodata*

Description

Simulated genotype data

Usage

```
data(genodata)
```

Format

A data frame with 2000 observations on the following 21 variables.

`pheno` a numeric vector
`snp1` a numeric vector
`snp2` a numeric vector
`snp3` a numeric vector
`snp4` a numeric vector
`snp5` a numeric vector
`snp6` a numeric vector
`snp7` a numeric vector
`snp8` a numeric vector
`snp9` a numeric vector
`snp10` a numeric vector
`snp11` a numeric vector
`snp12` a numeric vector
`snp13` a numeric vector
`snp14` a numeric vector
`snp15` a numeric vector
`snp16` a numeric vector
`snp17` a numeric vector
`snp18` a numeric vector
`snp19` a numeric vector
`snp20` a numeric vector

Details

The first column is the phenotype (1000 cases coded as 1, 1000 controls coded as 0)

Examples

```
data(genodata)
```

MULTI *MULTI: Multiple Tests*

Description

Performs multiple association tests.

Usage

```
MULTI(y, X, tests, maf = 0.05, perm = 100, weights = NULL, c.param = NULL)
```

Arguments

y	numeric vector with phenotype status: 0=controls, 1=cases. No missing data allowed
X	numeric matrix or data frame with genotype data coded as 0, 1, 2. Missing data is allowed
tests	character vector with names of the tests to be applied
maf	numeric value indicating the minor allele frequency threshold for rare variants (maf=0.05 by default)
perm	positive integer indicating the number of permutations (100 by default)
weights	optional vector of weights for the variants (NULL by default
c.param	Optional value to specify the c parameter when applying ORWSS

Details

The available tests are: "WSS", "ORWSS", "RWAS", "CMC", "CMAT", "CALPHA", "RBT", "SCORE", "SUM", "SSU", "SSU

There is no imputation for the missing data. Missing values are simply ignored in the computations.

Value

A data frame with test statistics and permutated p-values

Author(s)

Gaston Sanchez

Examples

```
## Not run:

# number of cases
cases = 250

# number of controls
```

```
controls = 250

# total (cases + controls)
total = cases + controls

# phenotype vector
phenotype = c(rep(1, cases), rep(0, controls))

# genotype matrix with 10 variants (random data)
set.seed(1234)
genotype = matrix(rbinom(total*10, 2, 0.051), nrow=total, ncol=10)

# apply MULTI with "BST", "CMC", "RWAS" and 100 permutations
mymulti1 = MULTI(phenotype, genotype, c("BST", "CMC", "RWAS"), perm=100)

# this is what we get
mymulti1

# create list with the following tests
test_list = c("BST", "CMC", "CMAT", "CALPHA", "ORWSS", "RWAS",
    "RBT", "SCORE", "SUM", "SSU", "SSUW", "UMINP", "WSS", "WST")

# apply MULTI with 100 permutations
mymulti2 = MULTI(phenotype, genotype, test_list, perm=100)

# this is what we get
mymulti2

## End(Not run)
```

ORWSS *ORWSS: Odds Ratio Weighted Sum Statistic*

Description

The ORWSS method has been proposed by Feng et al (2011) and it is based on a weighted sum statistic like the WSS method of Madsen and Browning (2009). ORWSS uses the logarithm of the odds ratio of a genetic variant as the weight for that variant, rather than the variance estimated in controls.

Usage

```
ORWSS(y, X, c.param = NULL, perm = 100)
```

Arguments

y	numeric vector with phenotype status: 0=controls, 1=cases. No missing data allowed
X	numeric matrix or data frame with genotype data coded as 0, 1, 2. Missing data is allowed

c.param	optional value to specify the c parameter. See reference Feng et al, 2011
perm	positive integer indicating the number of permutations (100 by default)

Details

When c.param=NULL, the weights of the sum statistic are simply the logarithm of the amended Odds Ratio of each variant (as in Dai et al 2012). Alternative values like c.param=1.64 or c.param=1.28 are suggested in Feng et al (2011).

There is no imputation for the missing data. Missing values are simply ignored in the computations.

Value

An object of class "assoctest", basically a list with the following elements:

orwss.stat	orwss statistic
perm.pval	permuted p-value
args	descriptive information with number of controls, cases, variants, and permutations
name	name of the statistic

Author(s)

Gaston Sanchez

References

Feng T, Elston RC, Zhu X (2011) Detecting Rare and Common Variants for Complex Traits: Sibpair and Odds Ratio Weighted Sum Statistics (SPWSS, ORWSS). *Genetic Epidemiology*, **35**: 398-409

Dai Y, Jiang R, Dong J (2012) Weighted selective collapsing strategy for detecting rare and common variants in genetic association study. *BMC Genetics*, **13**:7

See Also

WSS

Examples

```
## Not run:

# number of cases
cases = 500

# number of controls
controls = 500

# total (cases + controls)
total = cases + controls
```

```
  # phenotype vector
  phenotype = c(rep(1, cases), rep(0, controls))

  # genotype matrix with 10 variants (random data)
  set.seed(123)
  genotype = matrix(rbinom(total*10, 2, 0.05), nrow=total, ncol=10)

  # apply ORWSS with c.param=NULL and 500 permutations
  myorwss1 = ORWSS(phenotype, genotype, c.param=NULL, perm=100)
  myorwss1

  # apply ORWSS with c.param=1.64 (see Feng et al 2011)
  myorwss2 = ORWSS(phenotype, genotype, c.param=1.64, perm=100)
  myorwss2

## End(Not run)
```

RARECOVER *RARECOVER Algorithm*

Description

RARECOVER is an algorithm proposed by Bhatia et al (2010) that determines the set of variants in a manner of forward variable selection: starting from a null model without any genetic variants, genetic variants are selected one by one based on their statistical significance and then added into the model

Usage

```
RARECOVER(y, X, maf = 0.05, dif = 0.5, perm = 100)
```

Arguments

y	numeric vector with phenotype status: 0=controls, 1=cases. No missing data allowed
X	numeric matrix or data frame with genotype data coded as 0, 1, 2. Missing data is allowed
maf	numeric value indicating the minor allele frequency threshold for rare variants (maf=0.05 by default)
dif	numeric value between 0 and 1 as a threshold for the decision criterion in the RARECOVER algorithm (default dif=0.5)
perm	positive integer indicating the number of permutations (100 by default)

Details

The applied association test statistic (denoted as `XCORR` in Bhatia et al, 2010) is based on the Pearsons chi-square statistic

The argument `maf` is used to specify the threshold of the minor allele frequency for rare variants. By default, only variants below `maf=0.05` are taken into account in the analysis. However, if all variants in `X` are considered as rare variants, setting `maf=1` will consider them all for the analysis

There is no imputation for the missing data. Missing values are simply ignored in the computations.

Value

An object of class `"assoctest"`, basically a list with the following elements:

`rc.stat`	rarecover statistic
`perm.pval`	permuted p-value
`set`	set of selected variants
`args`	descriptive information with number of controls, cases, variants, rare variants, maf, number of selected variants, and permutations
`name`	name of the statistic

Author(s)

Gaston Sanchez

References

Bhatia G, Bansal V, Harismendy O, Schork NJ, Topol EJ, Frazer K, Bafna V (2010) A Covering Method for Detecting Genetic Associations between Rare Variants and Common Phenotypes. *PLoS Computational Biology*, **6(10)**: e1000954

See Also

`WSS`

Examples

```
## Not run:

# number of cases
cases = 500

# number of controls
controls = 500

# total (cases + controls)
total = cases + controls
```

```
  # phenotype vector
  phenotype = c(rep(1, cases), rep(0, controls))

  # genotype matrix with 10 variants (random data)
  set.seed(1234)
  genotype = matrix(rbinom(total*10, 2, 0.051), nrow=total, ncol=10)

  # apply RARECOVER with dif=0.05 and 500 permutations
  myrc = RARECOVER(phenotype, genotype, maf=0.05, perm=500)
  myrc

## End(Not run)
```

RBT *RBT: Replication Based Test*

Description

The Replication-Based Test has been proposed by Ionita-Laza et al (2011). RBT is based on a weighted-sum statistic that is similar to a pooled association test but purposefully designed to deal with possibly different association directions. The significance of the statistic has to be calculated by permutations.

Usage

```
RBT(y, X, perm = 100)
```

Arguments

y	numeric vector with phenotype status: 0=controls, 1=cases. No missing data allowed
X	numeric matrix or data frame with genotype data coded as 0, 1, 2. Missing data is allowed
perm	positive integer indicating the number of permutations (100 by default)

Details

There is no imputation for the missing data. Missing values are simply ignored in the computations.

Value

An object of class "assoctest", basically a list with the following elements:

rbt.stat	rbt statistic
perm.pval	permuted p-value
args	descriptive information with number of controls, cases, variants, and permutations
name	name of the statistic

Author(s)

Gaston Sanchez

References

Ionita-Laza I, Buxbaum JD, Laird NM, Lange C (2011) A New Testing Strategy to Identify Rare Variants with Either risk or Protective Effects on Disease. *PLoS Genetics*, **7(2)**: e1001289

See Also

WSS, CMC

Examples

```
## Not run:

# number of cases
cases = 500

# number of controls
controls = 500

# total (cases + controls)
total = cases + controls

# phenotype vector
phenotype = c(rep(1, cases), rep(0, controls))

# genotype matrix with 10 variants (random data)
set.seed(123)
genotype = matrix(rbinom(total*10, 2, 0.05), nrow=total, ncol=10)

# apply RBT with 500 permutations
myrbt = RBT(phenotype, genotype, perm=500)
myrbt

## End(Not run)
```

RVT1 *RVT1: Rare Variant Test 1 for dichotomous traits*

Description

RVT1 has been proposed by Morris and Zeggini (2010) as a collapsing method based on a regression framework that models the phenotype as a function of a collapsed summary of the variants. More specifically, the considered summary in RVT1 is the proportion of rare variants that carry at least one copy of the minor allele. In this sense, RVT1 is an accumulation approach that regresses phenotype on a genetic score, defined as the proportion of sites within the gene that harbor mutations.

Usage

```
RVT1(y, X, maf = 0.05, perm = 100)
```

Arguments

y	numeric vector with phenotype status: 0=controls, 1=cases. No missing data allowed
X	numeric matrix or data frame with genotype data coded as 0, 1, 2. Missing data is allowed
maf	numeric value indicating the minor allele frequency threshold for rare variants (maf=0.05 by default)
perm	positive integer indicating the number of permutations (100 by default)

Details

If no variants are below the specified maf threshold, the function will stop and return an error message

There is no imputation for the missing data. Missing values are simply ignored in the computations.

Value

An object of class "assoctest", basically a list with the following elements:

rvt1.stat	rvt1 statistic
asym.pval	asymptotic p-value
perm.pval	permuted p-value
args	descriptive information with number of controls, cases, variants, rare variants, maf threshold, and permutations
name	name of the statistic

Author(s)

Gaston Sanchez

References

Morris AP, Zeggini E (2010) An Evaluation of Statistical Approaches to Rare Variants Analysis in Genetic Association Studies. *Genetic Epidemiology*, **34**: 188-193

Asimit J, Zeggini E (2010) Rare Variant Association Analysis Methods for Complex Traits. *Annual Review of Genetics*, **44**: 293-308

See Also

RVT2

Examples

```
## Not run:

# number of cases
cases = 500

# number of controls
controls = 500

# total (cases + controls)
total = cases + controls

# phenotype vector
phenotype = c(rep(1, cases), rep(0, controls))

# genotype matrix with 10 variants (random data)
set.seed(1234)
genotype = matrix(rbinom(total*10, 2, 0.051), nrow=total, ncol=10)

# apply RVT1 with maf=0.05 and 500 permutations
myrvt1 = RVT1(phenotype, genotype, maf=0.05, perm=500)
myrvt1

## End(Not run)
```

RVT2 *RVT2: Rare Variant Test 2 for dichotomous traits*

Description

RVT2 is a collapsing method developed by Morris and Zeggini (2010) based on a regression framework that models the phenotype as a function of a collapsed summary of the variants. In the case of RVT@, the collapsed summary consists of the presence or absence of at least one minor allele at any rare variant. In this sense, RVT2 is an accumulation approach that regresses phenotype on a genetic score, defined as the presence of at least one minor allele at any rare variant

Usage

```
RVT2(y, X, maf = 0.05, perm = 100)
```

Arguments

y	numeric vector with phenotype status: 0=controls, 1=cases. No missing data allowed
X	numeric matrix or data frame with genotype data coded as 0, 1, 2. Missing data is allowed
maf	numeric value indicating minor allele frequency threshold for rare variants (`maf=0.05` by default)
perm	positive integer indicating the number of permutations (100 by default)

Details

If no variants are below the specified `maf` threshold, the function will stop and return an error message

There is no imputation for the missing data. Missing values are simply ignored in the computations.

Value

An object of class `"assoctest"`, basically a list with the following elements:

`rvt2.stat`	rvt2 statistic
`asym.pval`	asymptotic p-value
`perm.pval`	permuted p-value
`args`	descriptive information with number of controls, cases, variants, rare variants, maf threshold, and permutations
`name`	name of the statistic

Author(s)

Gaston Sanchez

References

Morris AP, Zeggini E (2010) An Evaluation of Statistical Approaches to Rare Variants Analysis in Genetic Association Studies. *Genetic Epidemiology*, **34**: 188-193

Asimit J, Zeggini E (2010) Rare Variant Association Analysis Methods for Complex Traits. *Annual Review of Genetics*, **44**: 293-308

See Also

RVT1

Examples

```
## Not run:

# number of cases
cases = 500

# number of controls
controls = 500

# total (cases + controls)
total = cases + controls

# phenotype vector
phenotype = c(rep(1, cases), rep(0, controls))
```

```
  # genotype matrix with 10 variants (random data)
  set.seed(1234)
  genotype = matrix(rbinom(total*10, 2, 0.051), nrow=total, ncol=10)

  # apply RVT2 with maf=0.05 and 500 permutations
  myrvt2 = RVT2(phenotype, genotype, maf=0.05)
  myrvt2

## End(Not run)
```

RWAS *RWAS: Rare-Variant Weighted Aggregate Statistic*

Description

The RWAS method has been proposed by Sul et al (2011) as a pooling method that groups variants and computes a weighted sum of differences between case and control mutation counts where weights are estimated from data. Under the null hypothesis the RWAS statistic has an asymptotic standard normal distribution, but a permutation procedure can also be applied to assess statistical significance

Usage

```
RWAS(y, X, maf = 0.05, perm = NULL)
```

Arguments

y	numeric vector with phenotype status: 0=controls, 1=cases. No missing data allowed
X	numeric matrix or data frame with genotype data coded as 0, 1, 2. Missing data is allowed
maf	numeric value indicating the minor allele frequency threshold for rare variants (ma f=0.05 by default)
perm	positive integer indicating the number of permutations (NULL by default)

Details

There is no imputation for the missing data. Missing values are simply ignored in the computations.

Value

An object of class "assoctest", basically a list with the following elements:

rwas.stat	rwas statistic
asym.pval	asymptotic p-value
perm.pval	permuted p-value, only when perm is used
args	descriptive information with number of controls, cases, variants, rare variants, maf and permutations
name	name of the statistic

Author(s)

Gaston Sanchez

References

Sul JH, Han B, He D, Eskin E (2011) An Optimal Weighted Aggregated Association Test for Identification of Rare Variants Involved in Common Diseases. *Genetics*, **188**: 181-188

See Also

CMC

Examples

```
## Not run:

# number of cases
cases = 500

# number of controls
controls = 500

# total (cases + controls)
total = cases + controls

# phenotype vector
phenotype = c(rep(1, cases), rep(0, controls))

# genotype matrix with 10 variants (random data)
set.seed(1234)
genotype = matrix(rbinom(total*10, 2, 0.051), nrow=total, ncol=10)

# apply RWAS with maf=0.05 and 500 permutations
myrwas = RWAS(phenotype, genotype, maf=0.05, perm=500)
myrwas

## End(Not run)
```

SCORE *SCORE: Score Test (from Logistic Regression)*

Description

The Score test is one of the statistical tests used for logistic regression models, which is one of the standard approaches used in genetic association studies. Under the null hypothesis that there is no associated variants within the examined region, the test statistic has an asymptotic chi-square distribution. In addition, a permutation procedure can be applied to assess the significance of the test.

Usage

```
SCORE(y, X, perm = 100)
```

Arguments

y	numeric vector with phenotype status: 0=controls, 1=cases. No missing data allowed
X	numeric matrix or data frame with genotype data coded as 0, 1, 2. Missing data is allowed
perm	positive integer indicating the number of permutations (100 by default)

Details

There is no imputation for the missing data. Missing values are simply ignored in the computations.

Value

An object of class "assoctest", basically a list with the following elements:

score.stat	score statistic
asym.pval	asymptotic p-value
perm.pval	permuted p-value
args	descriptive information with number of controls, cases, variants, and permutations
name	name of the statistic

Author(s)

Gaston Sanchez

References

Clayton D, Chapman J, Cooper J (2004) Use of unphased multilocus genotype data in indirect association studies. *Genetic Epidemiology*, **27**: 415-428

Chapman J, Whittaker J (2008) Analysis of Multiple SNPs in a Candidate Gene or Region. *Genetic Epidemioloy*, **32**: 560-566

See Also

SSU, SSUW, SUM

Examples

```
## Not run:

# number of cases
cases = 500

# number of controls
controls = 500

# total (cases + controls)
total = cases + controls

# phenotype (first column of genodata)
phenotype = c(rep(1, cases), rep(0, controls))

# genotype matrix with 10 variants (random data)
set.seed(123)
genotype = matrix(rbinom(total*10, 2, 0.05), nrow=total, ncol=10)

# apply SCORE with 500 permutations
myscore = SCORE(phenotype, genotype, perm=500)
myscore

## End(Not run)
```

SEQSUM *SEQSUM: Sequential Sum Score Test*

Description

SEQSUM has been proposed by Basu and Pan (2011) as a modification of the Sum test based on a model selection approach, following a similar philosophy as the CARV and RARECOVER methods. Assuming that there are M variants, the main idea behind the Sequential Sum test is to associate a sign to each variant indicating whether it has a positive effect or a negative effect. In other words, the purpose is to give signs to the variants so they reflect their effect (positive or negative).

Usage

```
SEQSUM(y, X, perm = 100)
```

Arguments

y	numeric vector with phenotype status: 0=controls, 1=cases. No missing data allowed
X	numeric matrix or data frame with genotype data coded as 0, 1, 2. Missing data is allowed
perm	positive integer indicating the number of permutations (100 by default)

Details

There is no imputation for the missing data. Missing values are simply ignored in the computations.

Value

An object of class `"assoctest"`, basically a list with the following elements:

`seqsum.stat`	seqsum statistic
`perm.pval`	permuted p-value
`signs`	a numeric vector with signs for the variants (1=positive, -1=negative)
`args`	descriptive information with number of controls, cases, variants, and permutations
`name`	name of the statistic

Author(s)

Gaston Sanchez

References

Basu S, Pan W (2011) Comparison of Statistical Tests for Disease Association with Rare Variants. *Genetic Epidemiology*, **35**: 606-619

See Also

SCORE, SUM

Examples

```
## Not run:

# number of cases
cases = 500

# number of controls
controls = 500

# total (cases + controls)
total = cases + controls

# phenotype vector
phenotype = c(rep(1, cases), rep(0, controls))

# genotype matrix with 10 variants (random data)
set.seed(123)
genotype = matrix(rbinom(total*10, 2, 0.05), nrow=total, ncol=10)

# apply SEQSUM with 500 permutations
myseq = SEQSUM(phenotype, genotype, perm=500)
```

```
  myseq

## End(Not run)
```

SKAT *SKAT: Sequence Kernel Association Test*

Description

SKAT is a regression method to test for association between genetic variants (common and rare) in a region. A score-based variance-component test.

Usage

```
SKAT(y, X, kernel = "linear", weights = NULL, a = 1, b = 25, perm = NULL)
```

Arguments

y	numeric vector with phenotype status: 0=controls, 1=cases. No missing data allowed
X	numeric matrix or data frame with genotype data coded as 0, 1, 2.
kernel	character string indicating the type of kernel to be used. Possible options are "linear", "wlinear", "quadratic", "IBS", "wIBS", "twowayx" (kernel="linear" by default)
weights	optional numeric vector with weights for the genetic variants (NULL by default)
a	positive numeric value for the parameter a in the Beta distribution (a=1 by default)
b	positive numeric vallue for the parameter b in the Beta distribution (b=25 by default)
perm	positive integer indicating the number of permutations (NULL by default)

Details

The argument kernel is used to specify the kernel function. "linear" indicates the linear kernel, "wlinear" indicates a weighted linear kernel, "quadratic" indicates the quadratic polynomial kernel, "IBS" indicates Identity-By-Share, "wIBS" indicates weighted IBS, and "twowayx" indicates a two-way interaction kernel.

For the weighted kernels ("wlinear" and "wIBS"), there are two options to get the weights. The default option (weights=NULL) involves the calculation of the weights by taking into account the minor allele frequency of the variants. In this case, the weights are calculated from a Beta distribution with parameters a and b. The second option is to specify the weights by providing a vector of weights for the variants; in this case the length of the vector must equal the number of columns in X. For more information see reference Wu et al (2011)

Value

An object of class `"assoctest"`, basically a list with the following elements:

`skat.stat`	skat statistic
`asymp.pval`	asymptotic p-value of the applied statistic (distributed as chi-square with df=1)
`perm.pval`	permuted p-value
`args`	descriptive information with number of controls, cases, variants, permutations, and selected kernel
`name`	name of the statistic

Note

This method is computationally expensive

Author(s)

Gaston Sanchez

References

Wu MC, Kraft P, Epstein MP, Taylor DM, Chanock SJ, Hunter DJ, Lin X (2010) Powerful SNP-Set Analysis for Case-Control Genome-wide Association Studies. *The American Journal of Human Genetics*, **86**: 929-942

Wu MC, Lee S, Cai T, Li Y, Boehnke M, Lin X (2011) Rare-Variant Association Testing for Sequencing Data with the Sequence Kernel Association Test. *The American Journal of Human Genetics*, **89**: 82-93

See Also

WSS

Examples

```
## Not run:

# load data genodata
data(genodata)

# phenotype (first column of genodata)
pheno = genodata[,1]

# genotype (rest of columns of genodata)
geno = genodata[,-1]

# apply SKAT with linear kernel
myskat.linear = SKAT(pheno, geno, kernel="linear")
myskat.linear
```

```
  # apply SKAT with weighted linear kernel
  # weights estimated from distribution beta(MAF, a=1, b=25)
  myskat.wlinear = SKAT(pheno, geno, kernel="wlinear", a=1, b=25)
  myskat.wlinear

  # apply SKAT with quadratic kernel
  myskat.quad = SKAT(pheno, geno, kernel="quadratic")
  myskat.quad

  # apply SKAT with IBS kernel
  myskat.ibs = SKAT(pheno, geno, kernel="IBS")
  myskat.ibs

## End(Not run)
```

SSU *SSU: Sum of Squared Score U Statistic*

Description

SSU has been proposed by Pan (2009) as a modified test based on the score-type U statistic (i.e. logistic regression). More specifically, SSU is a test based on the sum of the squares of the marginal score statistics. Its null distribution have a quadratic form and can be approximated by a chi-square distribution. In addition, SSU can also be regarded as a modification to the sum test.

Usage

```
SSU(y, X, perm = 100)
```

Arguments

y	numeric vector with phenotype status: 0=controls, 1=cases. No missing data allowed
X	numeric matrix or data frame with genotype data coded as 0, 1, 2. Missing data is allowed
perm	positive integer indicating the number of permutations (100 by default)

Details

There is no imputation for the missing data. Missing values are simply ignored in the computations.

Value

An object of class `"assoctest"`, basically a list with the following elements:

ssu.stat	ssu statistic
asym.pval	asymptotic p-value
perm.pval	permuted p-value

args	descriptive information with number of controls, cases, variants, and permutations
name	name of the statistic

Author(s)

Gaston Sanchez

References

Pan W (2009) Asymptotic tests of association with multiple SNPs in linkage disequilibrium. *Genetic Epidemiology*, **33**: 497-507

Pan W, Han F, Shen X (2010) Test Selection with Application to Detecting Disease Association with Multiple SNPs. *Human Heredity*, **69**: 120-130

See Also

SCORE, SUM, SSUW

Examples

```
## Not run:

# number of cases
cases = 500

# number of controls
controls = 500

# total (cases + controls)
total = cases + controls

# phenotype vector
phenotype = c(rep(1, cases), rep(0, controls))

# genotype matrix with 10 variants (random data)
set.seed(123)
genotype = matrix(rbinom(total*10, 2, 0.05), nrow=total, ncol=10)

# apply SSU with 500 permutations
myssu = SSU(phenotype, genotype, perm=500)
myssu

## End(Not run)
```

SSUW *SSUW: Weighted Sum of Squared Score U Statistic*

Description

SSUW s been proposed by Pan (2009) as a modified test based on the score-type U statistic (i.e. logistic regression). More specifically, SSUW is a test based on a weighted sum of the squares of the marginal score statistics. Its null distribution have a quadratic form and can be approximated by a chi-square distribution.

Usage

```
SSUW(y, X, perm = 100)
```

Arguments

y	numeric vector with phenotype status: 0=controls, 1=cases. No missing data allowed
X	numeric matrix or data frame with genotype data coded as 0, 1, 2. Missing data is allowed
perm	positive integer indicating the number of permutations (100 by default)

Details

There is no imputation for the missing data. Missing values are simply ignored in the computations.

Value

An object of class "assoctest", basically a list with the following elements:

ssuw.stat	ssuw statistic
asym.pval	asymptotic p-value
perm.pval	permuted p-value
args	descriptive information with number of controls, cases, variants, and permutations
name	name of the statistic

Author(s)

Gaston Sanchez

References

Pan W (2009) Asymptotic tests of association with multiple SNPs in linkage disequilibrium. *Genetic Epidemiology*, **33**: 497-507

Pan W, Han F, Shen X (2010) Test Selection with Application to Detecting Disease Association with Multiple SNPs. *Human Heredity*, **69**: 120-130

See Also

SCORE, SSU, SUM

Examples

```
## Not run:

# number of cases
cases = 500

# number of controls
controls = 500

# total (cases + controls)
total = cases + controls

# phenotype vector
phenotype = c(rep(1, cases), rep(0, controls))

# genotype matrix with 10 variants (random data)
set.seed(123)
genotype = matrix(rbinom(total*10, 2, 0.05), nrow=total, ncol=10)

# apply SSUW with 500 permutations
myssuw = SSUW(phenotype, genotype, perm=500)
myssuw

## End(Not run)
```

SUM *SUM: Sum Test*

Description

The SUM test has been proposed by Pan (2009) based on a modification of the Score. The idea behind the Sum Test is to test on only one parameter under the assumption of a common association strength between each of multiple genetic variants (e.g. SNPs) and the trait under analysis. The Sum test focuses on a scalar function of the multiple parameters with a resulting degree of freedom DF=1

Usage

```
SUM(y, X, perm = 100)
```

Arguments

y	numeric vector with phenotype status: 0=controls, 1=cases. No missing data allowed

X	numeric matrix or data frame with genotype data coded as 0, 1, 2. Missing data is allowed
perm	positive integer indicating the number of permutations (100 by default)

Details

There is no imputation for the missing data. Missing values are simply ignored in the computations.

Value

An object of class "assoctest", basically a list with the following elements:

sum.stat	sum statistic
asym.pval	asymptotic p-value
perm.pval	permuted p-value
args	descriptive information with number of controls, cases, variants, and permutations
name	name of the statistic

Author(s)

Gaston Sanchez

References

Pan W (2009) Asymptotic tests of association with multiple SNPs in linkage disequilibrium. *Genetic Epidemiology*, **33**: 497-507

Pan W, Han F, Shen X (2010) Test Selection with Application to Detecting Association with Multiple SNPs. *Human Heredity*, **69**: 120-130

See Also

SCORE, SSU, SSUW, WST

Examples

```
## Not run:

# number of cases
cases = 500

# number of controls
controls = 500

# total (cases + controls)
total = cases + controls

# phenotype vector
```

```
  phenotype = c(rep(1, cases), rep(0, controls))

  # genotype matrix with 10 variants (random data)
  set.seed(123)
  genotype = matrix(rbinom(total*10, 2, 0.05), nrow=total, ncol=10)

  # apply SUM with 500 permutations
  mysum = SUM(phenotype, genotype, perm=500)
  mysum

## End(Not run)
```

TTEST *TTEST: Hotelling T2 Test*

Description

Generalized T2 test for testing association between genotype variants and binary trait (case-control)

Usage

```
TTEST(y, X)
```

Arguments

y	numeric vector with phenotype status: 0=controls, 1=cases. No missing data allowed
X	numeric matrix or data frame with genotype data coded as 0, 1, 2. Missing data is allowed

Details

There is no imputation for the missing data. Missing values are simply ignored in the computations.

Value

An object of class "assoctest", basically a list with the following elements:

T2.stat	T2 statistic
asym.pval	asymptotic p-value
args	descriptive information with number of controls, cases, variants, maf, and applied test
name	name of the statistic

Author(s)

Gaston Sanchez

References

Xiong M, Zhao J, Boerwinkle E (2002) Generalized T2 Test for Genome Association Studies. *The American Journal of Human Genetics*, **70**: 1257 - 1268

See Also

CAST

Examples

```
## Not run:

# number of cases
cases =500

# number of controls
controls = 500

# total (cases + controls)
total = cases + controls

# phenotype vector
phenotype = c(rep(1, cases), rep(0, controls))

# genotype matrix with 10 variants (random data)
set.seed(123)
genotype = matrix(rbinom(total*10, 2, 0.05), nrow=total, ncol=10)

# apply TTEST
myttest = TTEST(phenotype, genotype)
myttest

## End(Not run)
```

UMINP *UMINP: Univariate minP (minimum p-value)*

Description

UMINP is the Univariate minP that tests on each single genetic variant (e.g. SNP) one-by-one and then takes the minimum of their p-values, Its null distribution is based on numerical integration with respect to a multivariate normal distribution.

Usage

```
UMINP(y, X, perm = 100)
```

Arguments

y	numeric vector with phenotype status: 0=controls, 1=cases. No missing data allowed
X	numeric matrix or data frame with genotype data coded as 0, 1, 2. Missing data is allowed
perm	positive integer indicating the number of permutations (100 by default)

Details

There is no imputation for the missing data. Missing values are simply ignored in the computations.

Value

An object of class `"assoctest"`, basically a list with the following elements:

uminp.stat	uminp statistic
asym.pval	asymptotic p-value
perm.pval	permuted p-value
args	descriptive information with number of controls, cases, variants, and permutations
name	name of the statistic

Author(s)

Gaston Sanchez

References

Pan W (2009) Asymptotic tests of association with multiple SNPs in linkage disequilibrium. *Genetic Epidemiology*, **33**: 497-507

Pan W, Han F, Shen X (2010) Test Selection with Application to Detecting Disease Association with Multiple SNPs. *Human Heredity*, **69**: 120-130

See Also

SCORE, SUM

Examples

```
## Not run:

# number of cases
cases = 500

# number of controls
controls = 500
```

```
  # total (cases + controls)
  total = cases + controls

  # phenotype vector
  phenotype = c(rep(1, cases), rep(0, controls))

  # genotype matrix with 10 variants (random data)
  set.seed(123)
  genotype = matrix(rbinom(total*10, 2, 0.05), nrow=total, ncol=10)

  # apply UMINP with 500 permutations
  myuminp = UMINP(phenotype, genotype, perm=500)
  myuminp

## End(Not run)
```

VT *VT: Variable Threshold*

Description

The Variable Threshold (VT) test has been proposed by Price et al (2010) based on the assumption that the minor allele frequencies of the causal rare variants may be different from those nonfunctional rare variants, which, if true, can be utilized to improve the power of the corresponding pooled association tests. The idea behind this approach is that there exists some (unknown) threshold T for which variants with a minor allele frequency (MAF) below T are more likely to be functional than are variants with an MAF above T. VT works by finding the maximum z-score across all possible values for the threshold T.

Usage

```
VT(y, X, maf = 0.05, perm = 100)
```

Arguments

`y`	numeric vector with phenotype status: 0=controls, 1=cases. No missing data allowed
`X`	numeric matrix or data frame with genotype data coded as 0, 1, 2. Missing data is allowed
`maf`	numeric value indicating the minor allele frequency threshold for rare variants (must be a positive number between 0 and 1, `maf=0.05` by default)
`perm`	positive integer indicating the number of permutations (100 by default)

Details

There is no imputation for the missing data. Missing values are simply ignored in the computations.

Value

An object of class `"assoctest"`, basically a list with the following elements:

`vt.stat`	vt statistic
`perm.pval`	permuted p-value
`args`	descriptive information with number of controls, cases, variants, and permutations
`name`	name of the statistic

Author(s)

Gaston Sanchez

References

Price AL, Kryukov GV, de Bakker PIW, Purcell SM, Staples J, Wei LJ, Sunyaev SR (2010) Pooled Association Tests for Rare Variants in Exon-Sequencing Studies. *The American Journal of Human Genetics*, **86**: 832-838

See Also

`WSS`

Examples

```
## Not run:

# number of cases
cases = 500

# number of controls
controls = 500

# total (cases + controls)
total = cases + controls

#  phenotype vector
phenotype = c(rep(1, cases), rep(0, controls))

# genotype matrix with 10 variants (random data)
set.seed(1234)
genotype = matrix(rbinom(total*10, 2, 0.051), nrow=total, ncol=10)

# apply VT with maf=0.05 and 500 permutations
myvt = VT(phenotype, genotype, maf=0.05, perm=500)
myvt

## End(Not run)
```

WSS *WSS: Weighted Sum Statistic*

Description

The WSS method has been proposed by Madsen and Browning (2009) as a pooling approach. In WSS, rare-variant counts within the same gene for each individual are accumulated rather than collapsing on them. Second, it introduces a weighting term to emphasize alleles with a low frequency in controls. Finally, the scores for all samples are ordered, and the WSS is computed as the sum of ranks for cases. The significance is determined by a permutation procedure.

Usage

```
WSS(y, X, perm = 100)
```

Arguments

y	numeric vector with phenotype status: 0=controls, 1=cases. No missing data allowed
X	numeric matrix or data frame with genotype data coded as 0, 1, 2. Missing data is allowed
perm	positive integer indicating the number of permutations (100 by default)

Details

There is no imputation for the missing data. Missing values are simply ignored in the computations.

Value

An object of class `"assoctest"`, basically a list with the following elements:

wss.stat	wss statistic
perm.pval	permuted p-value
args	descriptive information with number of controls, cases, variants, and permutations
name	name of the statistic

Author(s)

Gaston Sanchez

References

Madsen BE, Browning SR (2009) A Groupwise Association Test for Rare Mutations Using a Weighted Sum Statistic. *PLoS Genetics*, **5(2)**: e1000384

See Also

CMC

Examples

```
## Not run:

# number of cases
cases = 500

# number of controls
controls = 500

# total (cases + controls)
total = cases + controls

# phenotype vector
phenotype = c(rep(1, cases), rep(0, controls))

# genotype matrix with 10 variants (random data)
set.seed(123)
genotype = matrix(rbinom(total*10, 2, 0.05), nrow=total, ncol=10)

# apply WSS with 500 permutations
mywss = WSS(phenotype, genotype, perm=500)
mywss

## End(Not run)
```

WST *WST: Weighted Score Test*

Description

The WST method has been proposed by Wang and Elston (2007) and it can be seen as a fixed effects method with transformed predictors based on Fourier Transformations. WST is based on Fourier Transform (FT) coefficients to globally test a set of correlated genetic variants (e.g. SNPs). The sequence of genetic variants values is transformed into a sequence of numbers by discrete FT, but only the real parts of the FT coefficients are taken into account. A weighted score statistic of the FT components is calculated, which follows a standard normal distribution under the null hypothesis

Usage

```
WST(y, X, perm = 100)
```

Arguments

y	numeric vector with phenotype status: 0=controls, 1=cases. No missing data allowed
X	numeric matrix or data frame with genotype data coded as 0, 1, 2. NO missing data is allowed
perm	positive integer indicating the number of permutations (100 by default)

Details

This function does not allow missing genotypes

Value

An object of class "assoctest", basically a list with the following elements:

wst.stat	wst statistic
asym.pval	asymptotic p-value
perm.pval	permuted p-value
args	descriptive information with number of controls, cases, variants, and permutations
name	name of the statistic

Author(s)

Gaston Sanchez

References

Wang T, Elston C (2007) Improved Power by Use of a Weighted Score Test for Linkage Disequilibrium Mapping. *The American Journal of Human Genetics*, **80**: 353-360

See Also

SCORE, SUM

Examples

```
## Not run:

# number of cases
cases = 500

# number of controls
controls = 500

# total (cases + controls)
total = cases + controls

# phenotype vector
```

```
  phenotype = c(rep(1, cases), rep(0, controls))

  # genotype matrix with 10 variants (random data)
  set.seed(123)
  genotype = matrix(rbinom(total*10, 2, 0.05), nrow=total, ncol=10)

  # apply WST with 500 permutations
  mywst = WST(phenotype, genotype, perm=500)
  mywst

## End(Not run)
```

Index

∗Topic **datasets**
genodata, 24
∗Topic **package**
AssotesteR-package, 2

ASCORE, 3
AssotesteR *(*AssotesteR-package*)*, 2
AssotesteR-package, 2
ASSU, 5
ASSUW, 7
ASUM, 8

BST, 10

CALPHA, 12
CARV, 14
CAST, 16, *48*
CMAT, 18, *21*
CMC, *19*, 20, *31*, *36*, *53*

GDBR, 22
genodata, 24

MULTI, 25

ORWSS, 26

RARECOVER, *15*, 28
RBT, 30
RVT1, 31, *34*
RVT2, *32*, 33
RWAS, 35

SCORE, *4*, *11*, 36, *39*, *43*, *45*, *46*, *49*, *54*
SEQSUM, 38
SKAT, *23*, 40
SSU, *6*, *37*, 42, *45*, *46*
SSUW, *8*, *37*, *43*, 44, *46*
SUM, *10*, *37*, *39*, *43*, *45*, 45, *49*, *54*

TTEST, *21*, 47

UMINP, 48

VT, *15*, 50

WSS, *15*, *19*, *21*, *27*, *29*, *31*, *41*, *51*, 52
WST, *46*, 53