Package ‘iGasso’

February 20, 2015

Type Package
Title Genetic association tests and utilities
Version 1.2
Date 2014-06-11
Author Dr. Kai Wang
Maintainer Kai Wang <kai-wang@uiowa.edu>
Depends lattice, CompQuadForm
Description A collection of statistical tests and utilities for genetic association studies
License GPL (>= 2)
LazyLoad yes
NeedsCompilation no
Repository CRAN
Date/Publication 2014-06-18 08:43:55

R topics documented:

iGasso-package ................................................................. 2
genome.plot ................................................................. 3
kinf.BN ................................................................. 4
LMM.test ................................................................. 5
MFree.test ................................................................. 6
mlmp.test ................................................................. 7
null.par ................................................................. 9
SNPass.test .............................................................. 10
VSTF.test .............................................................. 11

Index 13
Description

iGasso is a collection of statistical tests developed by our group for genetic association studies. So far it contains functions for rare variants association, for association with multiple phenotypes, for linear mixed model analysis, and for model-free association analysis. There is also a function for genome plot. It will keep growing as more tests are developed. Use ?iGasso to see an introduction.

Details

Package: iGasso
Type: Package
Version: 1.2
Date: 2014-06-11
License: GPL (>=2)
LazyLoad: yes

Functions for various tests have .test as their extension name.

Author(s)

Kai Wang <kai-wang@uiowa.edu>

References


Examples

```r
y = rnorm(100)
chr = c(rep(1, 20), rep(3, 20), rep(10, 20), rep(19, 30), rep("X", 10))
pos = c(1:20, 1:20, 1:30, 1:10)
mydata = data.frame(y=y, chr=chr, pos=pos)
genome.plot(mydata, sig.line=c(1, -1), ylab="T Statistic")

G = rbind(c(14, 999), c(3, 1081))
VSTF.test(G)

G = rbind(c(161, 474, 489), c(231, 444, 380))
MFree.test(G)

G = matrix(sample(c(0, 1, 2), 200, replace=TRUE), ncol=10)
y = rnorm(10)
X = matrix(rnorm(10), ncol=1)
BN = kinf.BN(G, whole=TRUE)
tmp = null.par(BN, y, X)
LMM.test(G[,2], tmp$y1, BN, tmp$Sigma0.1)
```

Description

genome.plot plots the value of a variable across the genome.

Usage

```r
genome.plot(mydata, style=1, type="h", sig.line=c(4, -4),
             sig.color=c("red", "red"), ...)
```

Arguments

- **mydata**: a data frame containing three variables: `y` (numeric, the value of the variable to be plotted), `chr` (character, chromosome label), and `pos` (numeric, position, for instance, in base pair or centi-Morgan). Examples of `y` include -log10 of p-values and test statistic values.
- **style**: either 1 (default) or 2.
- **type**: a generic graphic parameter. Recommended values are "h" (default) and "b".
- **sig.line**: vertical locations of significance lines.
- **sig.color**: colors of significance lines.
- **...**: other parameters to be passed to function `xyplot` in the lattice package.

Details

This function makes use of the function xyplot from package lattice.
Author(s)
Kai Wang <kai-wang@uiowa.edu>

Examples
```r
y = rnorm(100)
chr = c(rep(1, 20), rep(3, 20), rep(10, 20), rep(19, 30), rep("X", 10))
pos = c(1:20, 1:20, 1:20, 1:30, 1:10)
mydata = data.frame(y=y, chr=chr, pos=pos)
mydata2 = data.frame(y=y^2, chr=chr, pos=pos)

genome.plot(mydata, sig.line=c(1, -1), ylab="T Statistic")
genome.plot(mydata, sig.line=c(1, -1), ylab="T Statistic", type="b")
genome.plot(mydata2, sig.line=c(2), ylab="y squared")
genome.plot(mydata2, style=2, sig.line=c(1, -1), ylab="T Statistic")
genome.plot(mydata2, style=2, sig.line=c(1, -1), ylab="T Statistic", type="b")
```

Description
kinf.bn computes a BN relatedness matrix from a matrix of SNP dose scores.

Usage
```r
kinf.bn(G, whole=FALSE)
```

Arguments
- `G` a (# SNP)x(# individual) matrix of SNP scores. Each SNP score can be 0, 1, or 2, or imputed dose. (# individual) includes only individuals with non-missing phenotype values.
- `whole` a logical variable. Are all SNPs included in G? If not, this function needs to be called more than once. Then the sum of total is devised by the sum of count. The default is FALSE.

Details
SNPs with MAF <= 0.01 are excluded. G can contain missing values.

Value
If `whole` = TRUE, returns the BN relatedness matrix. If `whole` = FALSE, returns a list containing the following components:
- `total` a matrix equal to the sum of genotype score inner products. Missing genotype scores are excluded.
- `count` a matrix of counts of individuals pairs both genotype scores are non-missing.
Author(s)

Kai Wang <kai-wang@uiowa.edu>

References


See Also

null.par for computing y1 and Sigma0.1 and LMM.test for conducting the score test based on the linear mixed model.

Examples

```r
G1 = matrix(sample(c(0,1,2), 200, replace=TRUE), ncol=10)
G2 = matrix(sample(c(0,1,2), 200, replace=TRUE), ncol=10)
BN = kinf.BN(G1, whole=TRUE)
tmp1 = kinf.BN(G1, whole=FALSE)
tmp2 = kinf.BN(G2, whole=FALSE)
BN = (tmp1$total+tmp2$total)/(tmp1$count+tmp2$count)
```

LMM.test

Score Test for the Linear Mixed Model

Description

LMM.test performs tests on association between an SNP and case-control status. It tests whether the frequencies of an allele are the same between cases and controls. It does not require specification of an inheritance model.

Usage

LMM.test(g, y1, K, S0.1)

Arguments

g a length n vector of genotype scores at the SNP being tested.

y1 a numeric vector of length n. It is the variable y1 returned by function null.par.

K an nxn BN relatedness matrix computed by function kinf.BN. Here n is the number of individuals with non-missing phenotype values.

S0.1 an nxn matrix equal to the inverse of Sigma0

Details

Matrix multiplication is implemented using existing R functions.
Value

A list with class "test" containing the following components:

- statistic the value of the test statistic.
- p.value the p-value for the test computed from a 50:50 mixture of 0 and a chi-square
distribution with 1 df.
- method a character string indicating the test performed.
- data.name a character string giving the name of the data.

Author(s)

Kai Wang <kai-wang@uiowa.edu>

References

Submitted.

See Also

null.par for computing y1 and Sigma0.1 and kinf.BN for computing K.

Examples

```r
G = matrix(sample(c(0,1,2), 200, replace=TRUE), ncol=10)
y = rnorm(10)
X = matrix(rnorm(10), ncol=1)
BN = kinf.BN(G, whole=TRUE)
tmp = null.par(BN, y, X)
LMM.test(G[2,], tmp$y1, BN, tmp$Sigma0.1)
apply(G, 1, LMM.test, tmp$y1, BN, tmp$Sigma0.1)
```

MFree.test

Model-free Association Tests

Description

MFree.test performs tests on association between an SNP and case-control status. It tests whether
the frequencies of an allele are the same between cases and controls. It does not require specification
of an inheritance model.

Usage

MFree.test(G, method="score")
mlmp.test

Arguments

G a 2x3 two-dimensional contingency table in matrix form. The first row is for cases and the second one for controls. In each row, the entries are the number of subjects carrying 0, 1, and 2 copies of the reference allele, respective.

method a character string indicating the test statistic to use. One of "score" (default), "Wald", and "LRT".

Details

Each test is named after the author(s) of the corresponding publication.

Value

A list with class "test" containing the following components:

statistic the value of the test statistic.
p.value the p-value for the test computed from a chi-square distribution with 1 df.
method a character string indicating the test performed.
data.name a character string giving the name of the data.

Author(s)

Kai Wang <kai-wang@uiowa.edu>

References


Examples

```r
G = rbind(c(161, 474, 489), c(231, 444, 380))
MFree.test(G)
MFree.test(G, method = "Wald")
MFree.test(G, method = "LRT")
```

mlmp.test A suite of multilocus multiple phenotype tests for association

Description

mlmp.test performs tests of association between multilocus SNPs and one or more traits possibly of different types.

Usage

```r
mlmp.test(g, y, weights = NULL, stat = "score")
```
Arguments

- `g`: an $n \times p$ matrix of SNP genotypes of $n$ study subjects at $p$ loci.
- `y`: an $n \times q$ matrix of phenotype values of $n$ study subjects on $q$ traits. $q$ could be 1. Traits can be dichotomous, continuous, selected data, etc. or a mix of these.
- `weights`: a vector of length $p$. Each element is the weight used for the corresponding variant. The default weight for each variant is the inverse of the sample variance of the genotype score at this variant. This option differs with the SKAT option by the same name in that its elements are $w$, not the square root of $w$.
- `stat`: Test statistic to be used. One of "F", "Wald", or "score" (default).

Details

This method regresses multilocus genotype over multiple phenotypes and test a quadratic null hypothesis. The $p$-value of a test statistic is determined through a linear combination of independent chi-square distributions and is evaluated via Davies’ method implemented in package CompQuadForm.

Value

A list with class "htest" containing the following components:

- `statistic`: the value of the test statistic.
- `parameter`: the number of SNPs and the number of traits.
- `p.value`: the $p$-value for the test computed using Davies’ method.
- `method`: a character string indicating the test performed.
- `data.name`: a character string giving the name of the data.

Author(s)

Kai Wang <kai-wang@uiowa.edu>

References


Examples

```r
n = 400
y = cbind(y, rnorm(n))
y = cbind(y, rnorm(n))
maf = seq(0.05, 0.5, 0.05)
g = NULL
for (j in 1:10){
genotype.freq = c(maf[j]^2, 2*maf[j]*(1-maf[j]), (1-maf[j])^2)
g = cbind(g, sample(c(0,1,2), n, replace=TRUE, prob=genotype.freq))
}
mlmp.test(g, y, weights=rep(1,10))
```
null.par

Estimate the Null Segregation Parameters of the Linear Mixed Model

Description

null.par estimates the null segregation parameters of the linear mixed model. Its output is used by lmm.test to compute the linear mixed model score statistic.

Usage

null.par(K, y, X)

Arguments

K
an \( n \times n \) relatedness matrix computed by function kinf.BN. Here \( n \) is the number of individuals with non-missing phenotype values.

y
a numeric vector of length \( n \) containing phenotype values.

X
an \( n \times p \) numeric matrix of covariate values. \( p \) is the number of covariates.

Details

A profile likelihood is maximized.

Value

A list containing the following components:

\( y_1 \)
the centered vector of \( y \).

\( \Sigma_0^{-1} \)
the inverse of matrix \( \Sigma_0 \), the variance matrix of \( y \).

Author(s)

Kai Wang <kai-wang@uiowa.edu>

References


See Also

kinf.BN for computing \( K \) and lmm.test for conducting the score test based on the linear mixed model.
Examples
G = matrix(sample(c(0,1,2), 200, replace=TRUE), ncol=10)
y = rnorm(10)
X = matrix(rnorm(10), ncol=1)
BN = kinf.BN(G, whole=TRUE)
null.par(BN, y, X)

SNPass.test An SNP Association Test using a Proportional Odds Model

Description
SNPass.test performs a test on association between an SNP and one or more phenotypes.

Usage
SNPass.test(geno, pheno)

Arguments
genoa length n vector of SNP genotypes on study subjects. Scoring of the genotypes aa, aA, and AA does not matter as long as the the score for the heterogeneous genotype is between the two homogeneous genotype scores.
phenoan nxp matrix of phenotype values. p could be 1. Phenotype can be dichotomous, continuous, or a mix of both types.

Details
The name of this function may change in the future.

Value
A list with class "htest" containing the following components:
statistic the value of the test statistic.
p.value the p-value for the test computed from a chi-square distribution with 1 df.
method a character string indicating the test performed.
data.name a character string giving the name of the data.

Author(s)
Kai Wang <kai-wang@uiowa.edu>

References
**Examples**

```r
pheno = c(1,1,1,1,0,0,0,0)
geno = c(0,1,1,2,2,1,1,2,2)
SNPass.test(geno, pheno)
```

**Description**

`VSTF.test` performs tests on association between a rare variant and case-control status using a variance-stabilizing transformation.

**Usage**

```r
VSTF.test(G, method = "Anscombe")
```

**Arguments**

- **G**: a 2x2 matrix. The first row is for cases and the second one for controls. In each row, the first element is the number of non-carriers and the second one is the number of carriers with at least 1 copy of the variant.
- **method**: a character string indicating which transformation to use. One of "Anscombe" (default), "arcsine", "Freeman-Tukey", and "Chanter".

**Details**

Each test is named after the author(s) of the corresponding publication.

**Value**

A list with class "test" containing the following components:

- **statistic**: the value of the test statistic.
- **p.value**: the p-value for the test computed from a chi-square distribution with 1 df.
- **method**: a character string indicating the test performed.
- **data.name**: a character string giving the name of the data.

**Author(s)**

Kai Wang <kai-wang@uiowa.edu>
References


Examples

```r
## Example 1 of Li et al. (2010)
G = rbind(c(14, 999), c(3, 1081))
VSTF.test(G)
VSTF.test(G, method = "arcsine")
VSTF.test(G, method = "Freeman-Tukey")
```
Index

genome.plot, 3
iGasso (iGasso-package), 2
iGasso-package, 2
kinf.BN, 4, 5, 6, 9
LMM.test, 5, 5, 9
MFree.test, 6
mlmp.test, 7
null.par, 5, 6, 9
SNPass.test, 10
VSTF.test, 11